Data Sheet, DS 1, Mar. 2003

INCA-D

Infineon Codec with DASL Transceiver and embedded Microcontroller Featuring Acoustic Echo Cancellation

PSB 21473 Version 1.3

Wired Communications

Never stop thinking.

.

Data Sheet

Revision History:	**2003-03-31**	DS 1
Previous Version:	02.01 / Data Sheet DS3	

Page	Subjects (major changes since last revision)
52/ 173	Reset value of SYSCON modified
74	Table entries deleted
103	HDLEN bit of register PSW removed
174	Reset value of BUSCON0 corrected
181	Reset value of WDTCON corrected
232	S0CON.RXDI bit moved to position 5
407	Peak Detector has no output
420	Note for PIDDHWCFG added
421ff	Modified formulas marked by vertical change bars
507	DSP register descriptions moved
508	Description of keyscanner interrupt generation modified
515	*14* interfaces and *6* Alternate settings supported (see vertical change bars)
516	Description regarding physical vs. logical endpoints modified
516	Figure for Configuraton 0 added
550	Bit position of AIM and IWIE
559	Description of CIAR modified
560	Reset of bits in GEPIR
562	Two bits in register CIARIE added
566	Device Detach Interrupt not available
567	Description for SUI modified
583	Sleep mode description modified
608	Reset values adapted
633	Absolute Maximum Ratings modified
634ff	DC characteristics modified
656	DC characteristics of transceiver removed (informaton partly moved to DC char.)
659ff	Values for ADC charateristics modified, transmission characteristics for AFE

For questions on technology, delivery and prices please contact the Infineon Technologies Offices in Germany or the Infineon Technologies Companies and Representatives worldwide: see our webpage at http://www.infineon.com.

Edition 2003-03-31

Published by Infineon Technologies AG,
St.-Martin-Strasse 53,
D-81541 München, Germany

Table of Contents **Page**

1 Overview 14
1.1 Features 15
1.1.1 16 bit CPU, Internal RAM and Memory Interface 15
1.1.2 General Purpose Timer Unit 15
1.1.3 Asynchronous/Synchronous Serial Interface (ASC) 16
1.1.4 Two Serial Channel Interfaces (SSC) 16
1.1.5 USB Interface 16
1.1.6 16 bit fixed-point DSP 16
1.1.7 Analog Front End 17
1.1.8 Transceiver 17
1.1.9 Terminal Specific Functions (TSF) 17
1.1.10 IOM-2 Handler 17
1.1.11 General Features 17
1.2 Logical Symbols 18
1.3 Typical Application 19

2 Pin Descriptions 20
2.1 Pin Configuration 20
2.2 Pin Definitions and Functions 22

3 Architectural Overview 31
3.1 Functional Block Diagram 31
3.2 Bus Systems 32

4 Clock Concept 33
4.1 Clock Generation 34
4.2 Terminal Specific Functions 35
4.3 USB 35
4.4 DSP 35
4.5 Microcontroller and Peripherals 36
4.6 AFE 38
4.7 IOM-2 clocks 38
4.8 Register Description of the Clock Concept Unit 39

5 Memory Organization 40
5.1 Internal ROM (Bootstrap Loader) 41
5.2 On Chip XRAM 41
5.3 Internal Dual-Port-RAM and SFR Area 41
5.4 XBUS Peripherals 42
5.5 System Stack 43
5.6 General Purpose Registers 43
5.7 PEC Source and Destination Pointers 44
5.8 Special Function Registers 45
5.9 External Memory Space 46

Table of Contents **Page**

5.10 Crossing Memory Boundaries 47

6 Central Processor Unit 48
6.1 Instruction Pipelining 49
6.2 CPU Special Function Registers 51
6.3 PEC - Extension of Functionality 70
6.4 XBUS System Architecture 77
6.4.1 Bus Access Control 77
6.4.2 XBUS Peripheral Configuration Block 80

7 Integrated OCDS Support 82
7.1 Applications of OCDS 82

8 Interrupts 84
8.1 Interrupt System Structure 85
8.2 Combined Interrupt Sources 89
8.2.1 Interrupt Node 3 (USBINT) 89
8.2.2 Interrupt Node 4 (USBEPINT) 91
8.2.3 Interrupt Node 13 (COMB1INT) 92
8.2.4 Interrupt Node 14 (COMB2INT) 96
8.2.5 Normal Interrupt Processing and PEC Service 98
8.2.6 Interrupt System Register Description 98
8.3 Interrupt Control Registers 99
8.4 Operation of the PEC Channels 104
8.5 Prioritization of Interrupt and PEC Service Requests 107
8.6 Saving the Status during Interrupt Service 109
8.7 Interrupt Response Times 111
8.8 PEC Response Times 113
8.9 External Interrupts 115
8.10 Trap Functions 117

9 Parallel Ports 123
9.1 PORT0 127
9.1.1 Alternate Functions of PORT0 131
9.2 PORT1 132
9.2.1 Alternate Functions of PORT1 136
9.3 PORT2 136
9.3.1 Alternate Functions of PORT2 139
9.4 PORT3 141
9.4.1 Alternate Functions of PORT3 144
9.5 PORT4 146
9.5.1 Alternate Functions of PORT4 148
9.6 PORT6 149
9.6.1 Alternate Functions of PORT6 151

Table of Contents **Page**

9.7 PORT7 152
9.7.1 Alternate Functions of PORT7 155

10 Dedicated Pins 156

11 The External Bus Interface 159
11.1 External Bus Modes 160
11.2 Programmable Bus Characteristics 168
11.3 Controlling the External Bus Controller 172
11.4 EBC Idle State 178

12 The Watchdog Timer (WDT) 180
12.1 Operation of the Watchdog Timer 181

13 The Bootstrap Loader 183
13.1 Activation of ASC Bootstrap Loader 183
13.2 Loading the Startup Code 185
13.3 Transfer of User Program 186

14 General Purpose Timer Unit 187
14.1 Functional Description of Timer Block 1 188
14.1.1 Core Timer T3 189
14.1.2 Auxiliary Timers T2 and T4 198
14.2 Functional Description of Timer Block 2 205
14.2.1 Core Timer T6 206
14.2.2 Auxiliary Timer T5 209
14.2.3 Timer Concatenation 211
14.2.4 Programming the GPT Unit 217
14.3 GPT Registers 218

15 The Asynchronous / Synchr. Serial Interface 229
15.1 Functional Description 229
15.1.1 Features 229
15.1.2 Overview 229
15.1.3 Register Description 231
15.1.4 General Operation 237
15.1.5 Asynchronous Operation 237
15.1.5.1 Asynchronous Data Frames 238
15.1.5.2 Asynchronous Transmission 241
15.1.5.3 Asynchronous Reception 242
15.1.5.4 IrDA Mode 242
15.1.5.5 RXD/TXD Data Path Selection in Asynchronous Modes 244
15.1.6 Synchronous Operation 244
15.1.6.1 Synchronous Transmission 245
15.1.6.2 Synchronous Reception 246

Table of Contents **Page**

15.1.6.3 Synchronous Timing 246
15.1.7 Baudrate Generation 247
15.1.7.1 Baudrates in Asynchronous Mode 248
15.1.7.2 Baudrates in Synchronous Mode 251
15.1.8 Hardware Error Detection Capabilities 253
15.1.9 Interrupts 253

16 The High-Speed Synchronous Serial Interfaces 255
16.1 Full-Duplex Operation 262
16.2 Half Duplex Operation 265
16.3 Baud Rate Generation 267
16.4 Error Detection Mechanisms 268
16.5 SSCx Interrupt Control 270

17 IOM-2 Handler, TIC/CI Handler and HDLC Controller 271
17.1 IOM2 Handler 271
17.1.1 IOM-2 Interface 272
17.1.2 Controller Data Access (CDA) 273
17.1.2.1 Looping and Shifting Data 275
17.1.3 Monitoring Data 279
17.1.4 Synchronous Transfer 280
17.1.5 Data transfer from and to the IOM-2 bus 283
17.1.5.1 Data Access on IOM-2 283
17.1.6 Compare Feature for transfer unit 0 287
17.1.7 Communication between DSP and IOM-2 288
17.1.8 Serial Data Strobe Signal and strobed Data Clock 289
17.1.8.1 Serial Data Strobe Signal 289
17.1.8.2 Strobed IOM Bit Clock 290
17.1.9 Collision 291
17.2 TIC/CI Handler 291
17.2.1 Overview 291
17.2.2 C/I channel 292
17.2.3 CIC Interrupt Logic 292
17.2.4 TIC Bus Handler - Functional Description 293
17.3 HDLC Controller 295
17.3.1 Overview 295
17.3.2 Message Transfer Modes 295
17.3.3 Data Reception 298
17.3.3.1 General Description 298
17.3.3.2 Possible Error Conditions during Reception of Frames 301
17.3.3.3 Data Reception Procedure 301
17.3.4 Receive Frame Structure 304
17.3.5 Data Transmission 305

Table of Contents **Page**

17.3.5.1 General Description 305
17.3.5.2 Possible Error Conditions during Transmission of Frames 307
17.3.5.3 Data Transmission Procedure 307
17.3.5.4 Transmit Frame Structure 309
17.3.6 Extended Transparent Mode 310
17.3.6.1 Transmitter 310
17.3.6.2 Receiver 310
17.4 HDLC Controller Interrupts 311
17.5 Register Description 312
17.5.1 HDLC Control and C/I Registers 318
17.5.1.1 RFIFO - Receive FIFO 318
17.5.1.2 XFIFO - Transmit FIFO 318
17.5.1.3 ISTAH - Interrupt Status Register HDLC 318
17.5.1.4 MASKH - Mask Register HDLC 320
17.5.1.5 STAR - Status Register 320
17.5.1.6 CMDR - Command Register 321
17.5.1.7 MODEH - Mode Register 322
17.5.1.8 EXMR- Extended Mode Register 323
17.5.1.9 SAP1 - SAPI1 Register 325
17.5.1.10 RBCL - Receive Frame Byte Count Low 325
17.5.1.11 SAP2 - SAPI2 Register 325
17.5.1.12 RBCH - Receive Frame Byte Count High 326
17.5.1.13 TEI1 - TEI1 Register 1 326
17.5.1.14 RSTA - Receive Status Register 327
17.5.1.15 TEI2 - TEI2 Register 328
17.5.1.16 TMH -Test Mode Register HDLC 329
17.5.1.17 CIR0 - Command/Indication Receive 0 329
17.5.1.18 CIX0 - Command/Indication Transmit 0 330
17.5.1.19 CIR1 - Command/Indication Receive 1 331
17.5.1.20 CIX1 - Command/Indication Transmit 1 331
17.5.2 IOM Handler (CDA) Register 332
17.5.2.1 CDAxy - Controller Data Access Register xy 332
17.5.2.2 XXX_TSDPxy - Time Slot and Data Port Selection for CHxy 332
17.5.2.3 CDAx_CR - Control Register Controller Data Access CH1x 333
17.5.3 IOM Handler (Control register) 335
17.5.3.1 TR_CR - Control Register Transceiver Data 335
17.5.3.2 BCHx_CR - Control Register B-Channel Data 335
17.5.3.3 DCI_CR - Control Register for HDLC and CI1 Data 336
17.5.3.4 DCIC_CR - Control Register for HDLC and CI1 Data 336
17.5.3.5 SDSx_CR - Control Register Serial Data Strobe x 337
17.5.3.6 IOM_CR - Control Register IOM Data 338
17.5.3.7 STI - Synchronous Transfer Interrupt 339

Table of Contents **Page**

17.5.3.8 ASTI - Acknowledge Synchronous Transfer Interrupt 340
17.5.3.9 MSTI - Mask Synchronous Transfer Interrupt 340
17.5.3.10 SDS_CONF - Configuration Register for Serial Data Strobes 341
17.5.3.11 MCDA - Monitoring CDA Bits 341
17.6 Interrupt and General Configuration 341
17.6.1 ISTA - Interrupt Status Register 341
17.6.2 MASK - Mask Register 342
17.6.3 MODE1 - Mode1 Register 343
17.6.4 ID - Identification Register 343
17.6.5 SRES - Software Reset Register 344
17.6.6 IOM Handler (Transfer Units) 344
17.6.6.1 ITRDU - IOM Transfer Unit, Read/Write Register DU 344
17.6.6.2 ITRDD - IOM Transfer Unit, Read/Write Register DD 345
17.6.6.3 ITRICV - IOM Transfer Unit, Idle Code Value for transfer unit 0 . . . 345
17.6.6.4 ITR_CR - IOM Transfer Control Register Register 345
17.6.6.5 ITR_MSK0 - IOM Transfer Time Slot and Data Port Sel. Reg. 0 . . . 346
17.6.6.6 ITR_MSKx - IOM Transfer Time Slot and Data Port Sel. Reg. 1 - 7 347
17.6.6.7 IWSR - IOM waitstates register 347

18 Line Transceiver 349
18.1 Wiring Configurations 349
18.2 Line Coding, Frame Structure of the DASL interfacee 349
18.3 Data Transfer and Delay between IBUS and DASL Interface 350
18.4 Control of Layer 1 / State Machine 351
18.4.1 State Machine 353
18.4.1.1 States 353
18.4.1.2 C/I Commands 356
18.4.1.3 C/I Indications 357
18.4.1.4 Receive Infos on the Line (Downstream) 357
18.4.1.5 Transmit Infos on the Line (Upstream) 358
18.4.2 C/I Codes 358
18.5 Level Detection and Power Down 359
18.6 Transceiver Enable/Disable 359
18.7 Test Functions 360
18.7.1 Line Transceiver Test 360
18.7.2 Test Signals on the Line Interface 360
18.8 Receive PLL (DPLL-Adjust Unit) 360
18.9 Register Description 361
18.9.1 TR_CONF0 - Transceiver Configuration Register 362
18.9.2 TR_CONF1 - Receiver Configuration Register 362
18.9.3 TR_CONF2 - Transmitter Configuration Register 363
18.9.4 TR_STA - Transceiver Status Register 363
18.9.5 TR_CMD - Transceiver Command Register 364

Table of Contents **Page**

18.9.6 ISTATR - Interrupt Status Register Transceiver 364
18.9.7 MASKTR - Mask Transceiver Interrupt 365

19 Analog Front End 366
19.1 Analog Front End (AFE) Description 366
19.2 External circuitry 368

20 Digital Signal Processor 371
20.1 Functional Units 371
20.1.1 Full Duplex Speakerphone 374
20.1.1.1 Echo Cancellation Unit 376
20.1.1.2 Echo Cancellation (Fullband Mode) 376
20.1.1.3 Echo Cancellation (Subband Mode) 379
20.1.2 Noise Reduction 381
20.1.3 Echo Suppression 383
20.1.3.1 Speech Detector 385
20.1.3.2 Speech Comparators (SC) 387
20.1.3.3 Attenuation Control 390
20.1.3.4 Echo Suppression Status Output 391
20.1.3.5 Automatic Gain Control 391
20.1.3.6 Fixed Gain 393
20.1.3.7 Mode Control 393
20.1.4 Line Echo Cancellation Unit 394
20.1.5 DTMF Detector 396
20.1.6 DTMF Generator 397
20.1.7 Interface to the Analog Front End 399
20.1.8 Universal Attenuator 400
20.1.9 Automatic Gain Control Unit 400
20.1.10 Noise Reduction Unit 401
20.1.11 Equalizer 402
20.1.12 Tone Generator 404
20.1.13 Peak Detector 407
20.1.14 Universal Summation Point 408
20.2 Digital Interfaces 409
20.2.1 Interface to IOM handler (Codec Digital Interface - CDI) 409
20.2.2 Parallel Interface to the DSP Domain PIDD 411
20.2.2.1 Interrupt 412
20.2.2.2 Abort 413
20.2.2.3 Revision Register 413
20.2.3 Hardware Configuration 414
20.3 Restrictions and Mutual Dependencies of Modules 414
20.4 DSP Register Description 418
20.4.1 PIDD Command Register 418

Table of Contents **Page**

20.4.2 Status Register 419
20.4.3 Hardware Configuration Register 420
20.4.4 DSP address domain 421

21 Terminal Specific Functions 507
21.1 Keypad Scanner 507
21.2 LED Matrix Control for up to 24 LED’s 510
21.3 TSF Global Register Description 512
21.3.1 Keypad Scanner Register Description 512
21.3.2 LED Multiplex Unit register description 514

22 USB Module 515
22.1 Introduction 515
22.2 USB Configurations 516
22.3 Transfer Modes 519
22.4 Memory Buffer 520
22.4.1 Overview 520
22.4.2 Single Buffer Mode 521
22.4.2.1 USB Write Access 521
22.4.2.2 USB Read Access 524
22.4.3 Dual Buffer Mode 527
22.4.4 Interruption of data transfers 533
22.5 Memory Buffer Organisation 534
22.5.1 Memory Buffer Address Generation 536
22.6 USB block activation 537
22.7 Initialization of USB Module 537
22.8 USB Device Framework 540
22.8.1 Enumeration Process 540
22.8.2 Control Transfers 540
22.8.2.1 Setup Stage 540
22.8.2.2 Data Stage 540
22.8.2.3 Status Stage 541
22.8.3 Standard Device Requests 541
22.8.4 GET_DESCRIPTOR 543
22.9 Onchip USB Transceiver 544
22.10 USB Registers Description. 546
22.10.1 USB Register handling interrupts 549
22.11 Detailed Register Description - USB Device Registers 559
22.11.1 Global Endpoint Stall Register (GESR) 559
22.11.2 Configuration Request Register (CIAR) 559
22.11.3 Global Endpoint Interrupt Request Register (GEPIR) 560
22.11.4 Configuration Request Interrupt Register (CIARI) 561
22.11.5 Configuration Request Interrupt Enable (CIARIE) 562

Table of Contents **Page**

22.11.6 Device Control Register (DCR) 563
22.11.7 Device Power Down Register (DPWDR) 564
22.11.8 USB Device Interrupt Enable Register (DIER) 565
22.11.9 USB Device Interrupt Request Register (DIRR) 566
22.11.10 Device Setup and Status Register (DSSR) 567
22.11.11 Frame Number Register (FNR) 568
22.11.12 Device Get_Status Register (DGSR) 569
22.11.13 Interface Get_Status Register (IGSR) 570
22.11.14 Interface Select Registers (IFCSEL) 570
22.11.15 Enable Setup Token(ESTR) 571
22.11.16 Active Endpoint Number (ACTEP) 571
22.12 Detailed Register Description - USB Endpoint Registers 572
22.12.1 Endpoint Get_Status Register (EGSRn) 572
22.12.2 Endpoint Interrupt Enable Register (EPIEn) 573
22.12.3 Endpoint Interrupt Request Register (EPIRn) 574
22.12.4 Endpoint Buffer Control Register (EPBCn) 576
22.12.5 Endpoint Buffer Status Register (EPBSn) 577
22.12.6 Endpoint Base Address Register (EPBAn) 580
22.12.7 Endpoint Buffer Length Register (EPLENn) 581
22.12.8 Address Offset Register (ADROFFn) 581
22.12.9 USB Data Register (USBVALn) 582

23 Power Reduction Modes 583
23.1 Sleep Mode of the CPU 583
23.2 Idle Mode of the CPU 584
23.3 Power Down Mode of the CPU 585
23.4 Status of Output Pins during Idle and Power Down Mode 586
23.5 Power Down Mode of Peripherals 587
23.6 Power Down Mode of the DSP 590

24 System Reset 591
24.1 Reset Types 593
24.1.1 Software Reset 594
24.1.2 WDT Reset 594
24.2 Reset Sequence Control 594
24.3 Reset Lengthening Control (Start Delay) 594
24.4 Power on reset 596
24.5 Software Reset for IOM-Handler and Transceiver 596
24.6 The INCA-D's Pins after Reset 596
24.6.1 Reset Output Pin 597
24.6.2 Watchdog Timer Operation after Reset 598
24.6.3 Reset and Power Down Mode of the DSP 598
24.7 Ports and External Bus Configuration during Reset 599

Table of Contents **Page**

24.8 System Startup Configuration 600

25 The Register Set 604
25.1 Register Description Format 606
25.2 CPU General Purpose Registers (GPRs) 606
25.3 XBUS Special Function Registers ordered by Address 608
25.4 PD-Bus Special Function Registers ordered by Address 623
25.5 DSP Register Address Summary 630

26 Electrical Characteristics 633
26.1 Electrical Characteristics (general) 633
26.2 Recommended Operating Conditions 633
26.3 DC-Characteristics 634
26.4 Capacities 637
26.5 Oscillator Specification 637
26.6 AC Characteristics 639
26.6.1 AC Characteristics of the External Bus Interface 640
26.6.2 SSC0 and SSC1 Timing 651
26.6.3 ASC Timing Synchronous mode 652
26.7 IOM-2 Interface Timing 654
26.8 Electrical Characteristics Transceiver 656
26.9 USB Transceiver Characteristics 657
26.10 Electrical Characteristics Analog Front End 659
26.10.1 Transmission Characteristics 659
26.10.2 Analog Front End Input Characteristics 661
26.10.3 Analog Front End Output Characteristics 661
26.11 Electrical Characteristics DSP 662

27 Package Outlines 663

1 Overview

The INCA-D integrates all necessary functions for the completion of a digital voice terminal solution.

The line transceiver of the INCA-D implements the subscriber access functions for a digital terminal to be connected to a two wire DASL interface. It covers complete layer-1 and basic layer-2 functions for digital terminals.

Different interfaces allow the connection to a variety of devices including an Full Speed USB interface for PC host communication.

The basic application of the INCA-D is the use in terminal equipment applications where microphone, loudspeaker, headset or handset can be directly connected to the Analog Frontend.

The Analog Front End and the integrated fixpoint DSP perform encoding, decoding, filtering functions and tone generation (ringing, audible feedback tones and DTMF signal). A full duplex echo cancellation mechanism provides high quality speakerphone functionality.

The INCA-D is a CMOS device and operates with a single 3.3V supply.

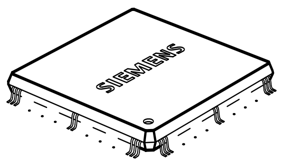
Infineon Codec with DASL Transceiver and embedded Microcontroller Featuring Acoustic Echo Cancellation INCA-D

PSB 21473

Version 1.3 **CMOS**

1.1 Features

1.1.1 16 bit CPU, Internal RAM and Memory Interface


P-TQFP-144

- Bootstrap Loader Function
- On Chip Memory: Dual Port SRAM (2 KBytes) & General Purpose SRAM (4 KBytes)
- On Chip Debug System OCDS (Level 1)
- Interrupt Controller supporting up to 27 nodes
- Peripheral Event Controller (PEC) for up to 8 channels
- External Memory Interface supporting 8 bit or 16 bit data, multiplexed and demultiplexed
- Up to 4 MBytes linear address space for external code and data
- 3 programmable Chip Selects
- Programmable Watchdog Timer

1.1.2 General Purpose Timer Unit

Timer Block 1:
- $f_{Timer}/4$ maximum resolution.
- 3 independent timers/counters.
- Timers/counters can be concatenated.
- 4 operating modes (timer, gated timer, counter, incremental).

Timer Block 2:
- $f_{Timer}/2$ maximum resolution.
- 2 independent timers/counters.
- Timers/counters can be concatenated.
- 3 operating modes (timer, gated timer, counter).
- Extended capture/reload functions via 16-bit Capture/Reload register CAPREL.

Type	Package
PSB 21473	P-TQFP-144

1.1.3 Asynchronous/Synchronous Serial Interface (ASC)

Full duplex asynchronous operating modes
- 8- or 9-bit data frames, LSB first
- Parity bit generation/checking
- One or two stop bits
- Baudrate from 1.5 MBaud to 0.3552 Baud (@24 MHz CPU clock)
- Multiprocessor mode for automatic address/data byte detection
- Loop-back capability

Half-duplex 8-bit synchronous operating mode
- Baudrate from 3 MBaud to 305.76 Baud (@ 24 MHz CPU clock)
- Double buffered transmitter/receiver

1.1.4 Two Serial Channel Interfaces (SSC)

- Master and slave mode operation
- Full-duplex or half-duplex operation
- Flexible data format
- Programmable number of data bits : 2 to 16 bit
- Programmable shift direction : LSB or MSB shift first
- Programmable clock polarity : idle low or high state for the shift clock
- Programmable clock/data phase : data shift with leading or trailing edge of SCLK

- Baudrate generation from 12 MBaud to 183.1 Baud (@ 24 MHz module clock)

Interrupt generation
- on a transmitter empty condition
- on a receiver full condition
- on an error condition (receive, phase, baudrate, transmit error)

1.1.5 USB Interface

- USB specification v1.1 compliant
- 12 Mbit/s Full-Speed Mode
- 15 Interfaces and 7 Alternate Settings supported
- 15 SW-configurable Endpoints, in addition to the bi-directional Control Endpoint 0
- Flexible Memory Management to support Endpoint Buffer Sizes of up to 64 Bytes
- Each non-Control Endpoint can be either Isochronous, Bulk or Interrupt

1.1.6 16 bit fixed-point DSP

- Full-duplex echo cancellation with noise reduction

- PCM A-Law/µ-Law (ITU-T G.711) and 8/16-bit linear data
- Two transducer correction filters
- Side tone gain adjustment
- Set of functional units as described (e.g. Tone Generator, DTMF Receiver)
- Access to two independent 16 bit time slots for up to three voice channels

1.1.7 Analog Front End

- Three differential inputs for the handset, the speakerphone and the headset microphone
- Three differential outputs for a handset ear piece (200 Ω), a headset (200 Ω) and a loudspeaker
- 80 mW @ 25 Ω or 100 mW @ 20 Ω loudspeaker driver capability
- Flexible test and maintenance loop backs in the analog front end
- Gain programmable amplifiers for all analog inputs and outputs
- PCM Codec, fully compatible with the ITU-T G.712 and ETSI (NET33) specification
- Additional summing point to add the Tone Generator output signal and the analog converted audio output from the DSP. The result is fed to the loudspeaker.

1.1.8 Transceiver

- Two wire DASL transceiver with AMI coded 2B+D channels

1.1.9 Terminal Specific Functions (TSF)

- Keypad Scanner for up to 45 keys
- LED Multiplex Unit (3*8 LEDs)

1.1.10 IOM-2 Handler

- IOM-2 Interface
- HDLC controller: Access to B1, B2 or D channels or the combination of them e.g. for 144 kbit data transmission (2B+D)
- 2* 64 byte FIFO buffer for efficient D-channel transfer of data packets
- Implementation of C/I-channel protocol to control peripheral devices
- Test loops and functions
- Data Control Unit for fast data transfers from IOM-2 time slots to memory and vice versa

1.1.11 General Features

- Power Management
- Single 15.36 MHz crystal
- On Chip PLL and a set of clock frequency divider
- 3.3V Single Supply Voltage

1.2 Logical Symbols

Figure 1-1 illustrates the logical symbol of the INCA-D.

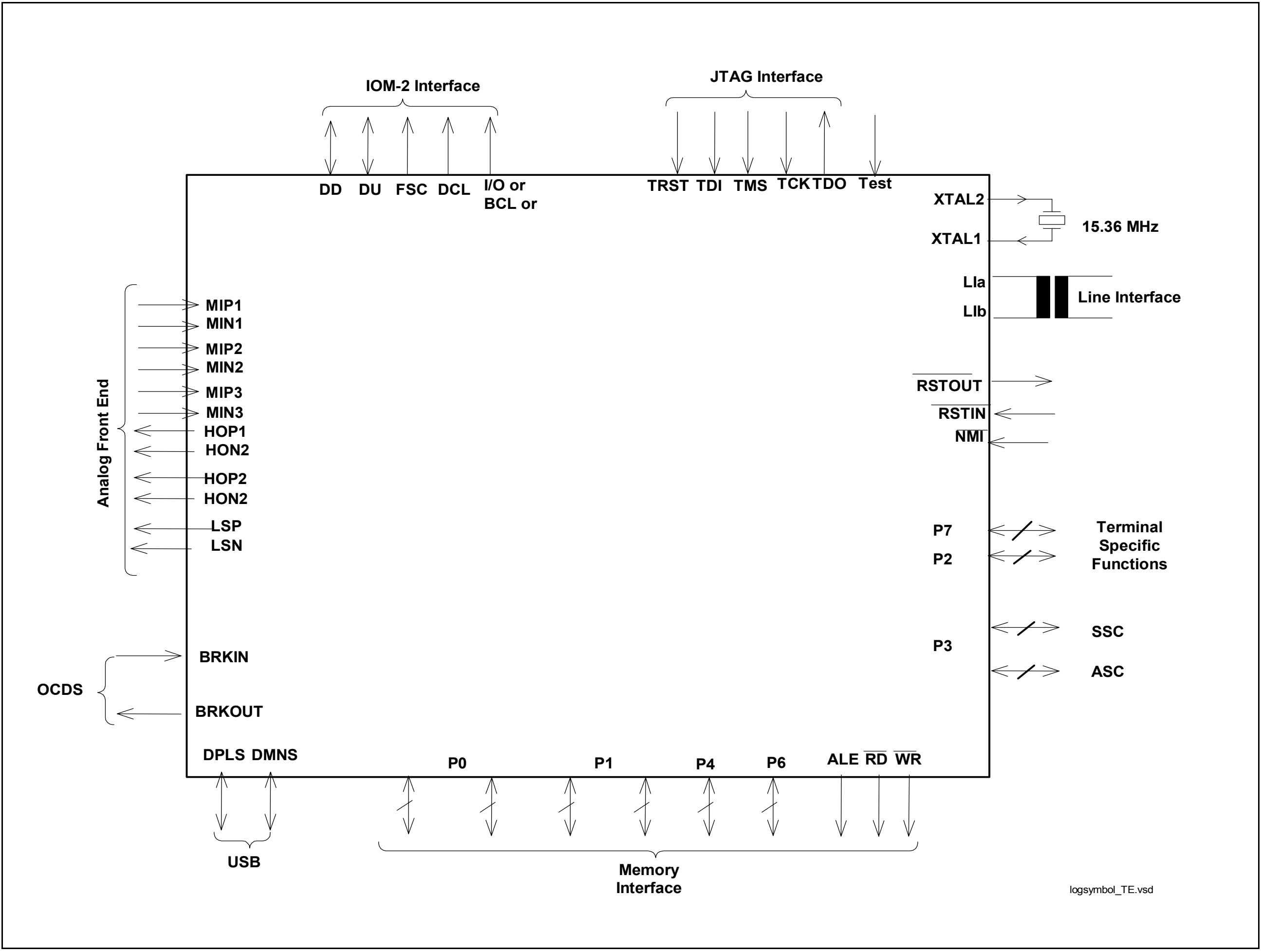


Figure 1-1 Logical Symbol TE mode

1.3 Typical Application

The following figure illustrates the typical application in which the **INCA-D** is usually used.

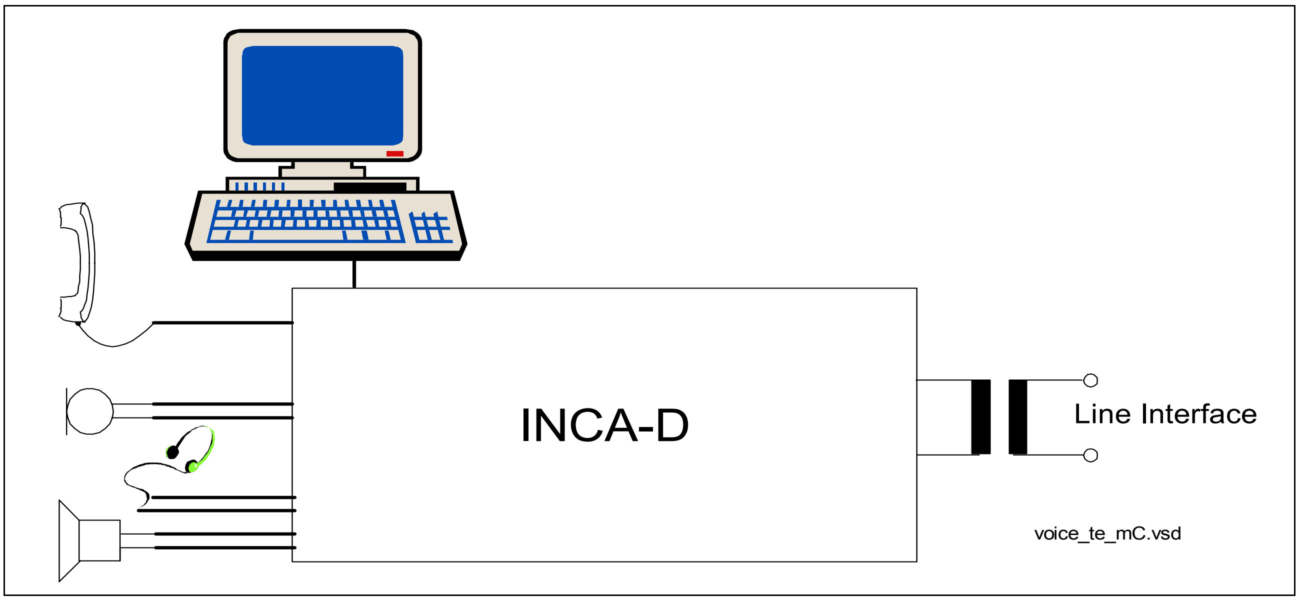


Figure 1-2 Basic Configuration

2 Pin Descriptions

2.1 Pin Configuration

The pin configuration of the INCA-D is shown in figure 2-1.

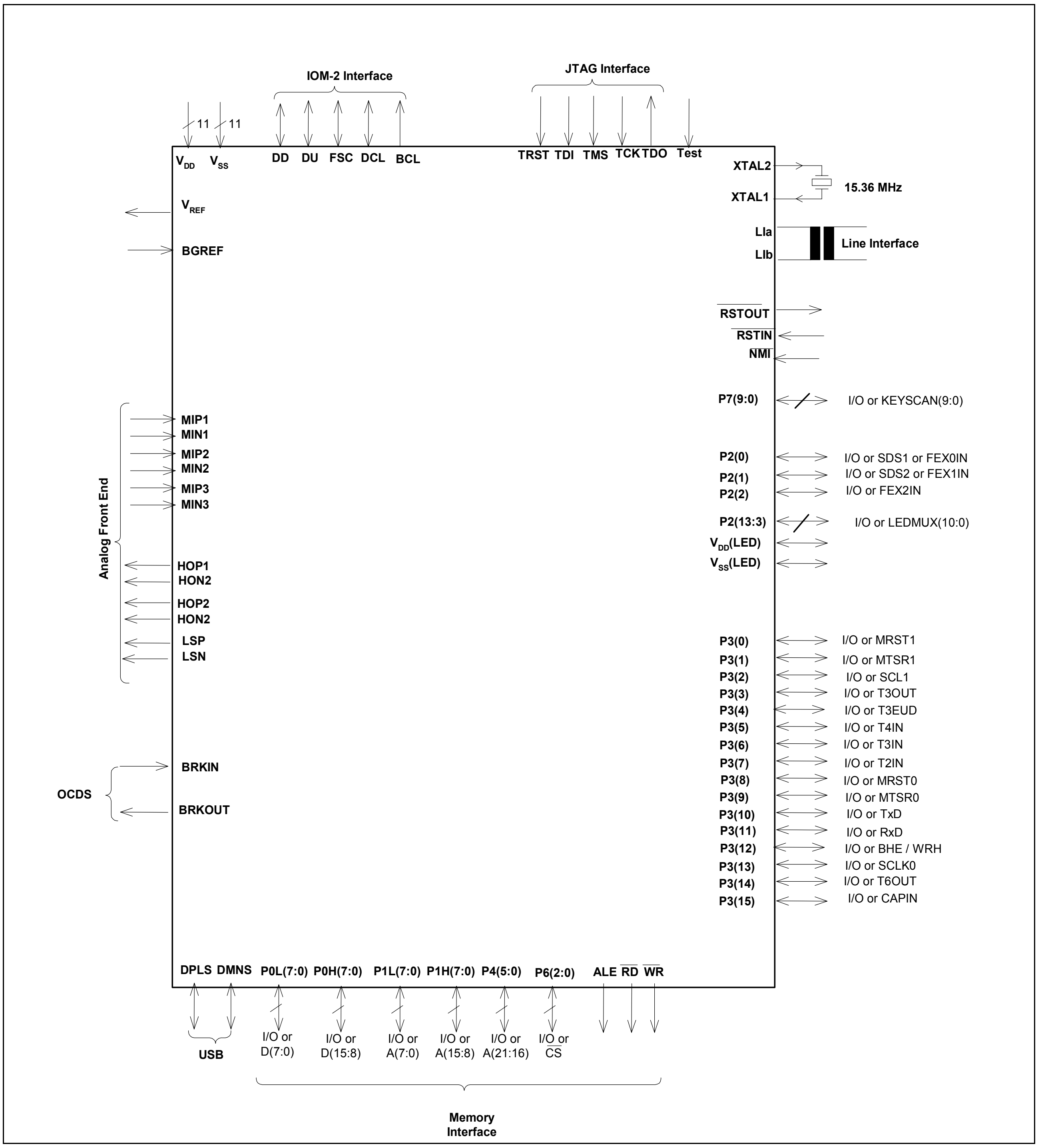


Figure 2-1 Pin Configuration

The mapping of the pin configuration to the physical device is shown in figure 2-2.

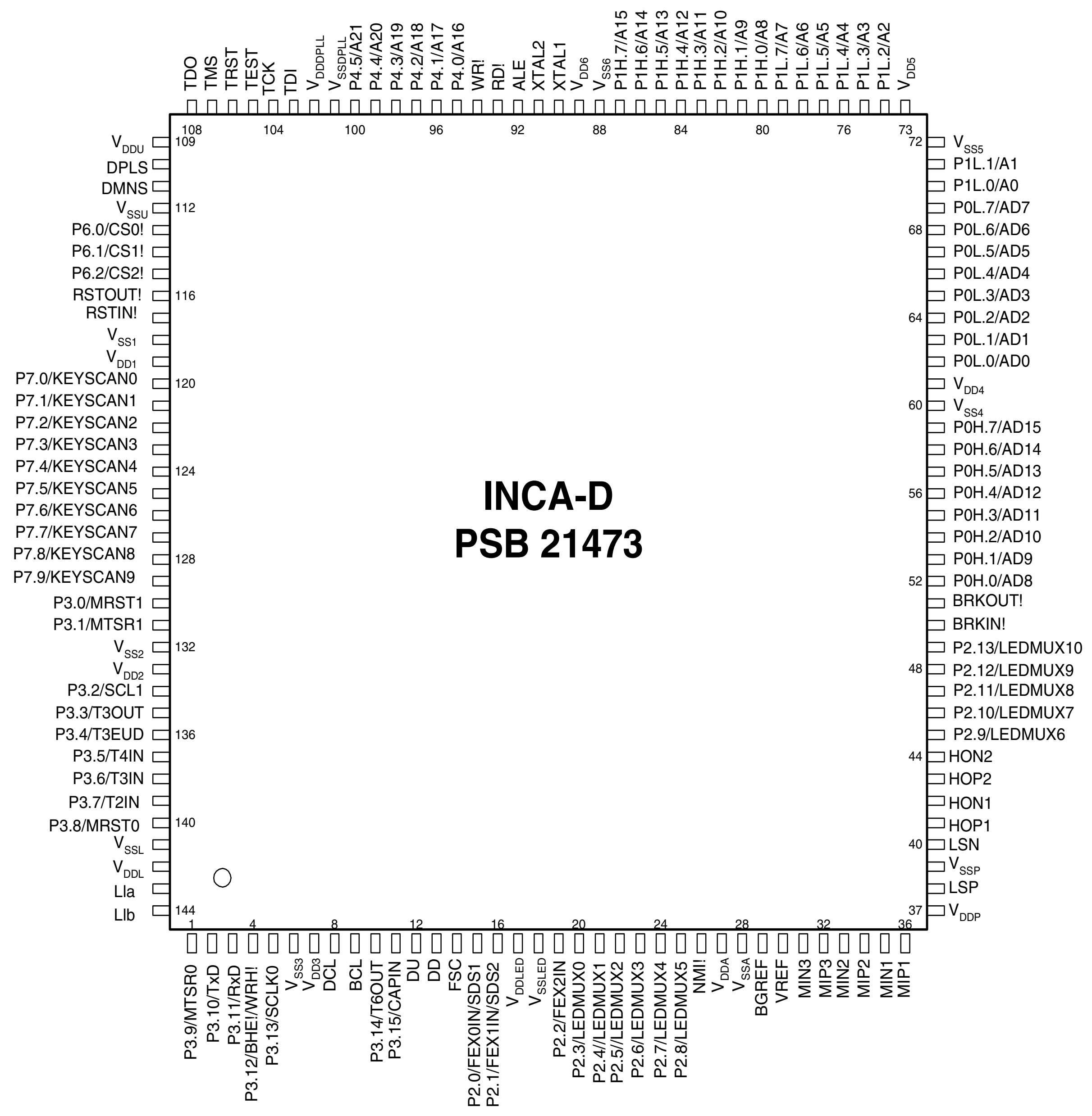


Figure 2-2 Mapping of pins to physical device

2.2 Pin Definitions and Functions

Table 2-1 **Memory Interface and Control Signals**

Pin No.	**Symbol**	Input (I) **Output (O)** **Open Drain** **(OD**	**Function**
   62- 69 52- 59	**PORT0** P0L.0- P0L.7: P0H.0- P0H.7:	I/O (OD)	PORT0 consists of the two 8-bit bidirectional I/O ports POL and POH. It is bitwise programmable for input or output via direction bits. It contains internal pull resistors. For external memory access, PORT0 serves as the data (D) bus in demultiplexed bus modes. Beside these functions, P0 serves for latching in the start-up configuration during reset.
  70, 71, 74-79 80- 87	**PORT1** P1L.0- P1L.7 P1H.0- P1H.7	I/O (OD)	PORT1 consists of the two 8-bit bidirectional I/O ports P1L and P1H. It is bitwise programmable for input or output via direction bits. It contains internal pull resistors. For external memory access PORT1 is used as the 16-bit address bus (A) in demultiplexed bus modes.
     95 100	**PORT4** P4.0- P4.5	I/O (OD) O ... ... O	PORT4 is a 6-bit bidirectional I/O port. It is bit-wise programmable for input or output via direction bits. It contains internal pull resistors. For external memory access Port4 can be used to output the segment address lines: P4.0 A16 : Least Significant Segment Address Line ... P4.5 A21: Most Significant Segment Address Line
   113 114 115	**PORT6** P6.0 - P6.2	I/O, (OD) O O O	PORT6 is a 3-bit bidirectional I/O port. It is bit-wise programmable for input or output via direction bits. It contains internal pull resistors. P6.0 $\overline{\text{CS0}}$ Chip Select 0 P6.1 $\overline{\text{CS1}}$ Chip Select 1 P6.2 $\overline{\text{CS2}}$ Chip Select 2

Table 2-1 Memory Interface and Control Signals (cont'd)

Pin No.	**Symbol**	Input (I) **Output (O) Open Drain (OD**	**Function**
93	$\overline{\text{RD}}$	O	External Memory Read Strobe. $\overline{\text{RD}}$ is activated for every external instruction or data read access. (internal pull-up provided)
94	$\overline{\text{WR}}$/$\overline{\text{WRL}}$	O	External Memory Write Strobe. In $\overline{\text{WR}}$ mode this pin is activated for every external data write access. In $\overline{\text{WRL}}$ mode this pin is activated for low byte data write accesses on a 16-bit bus, and for every data write access on an 8-bit bus. See WRCFG in register SYSCON for mode selection. (internal pull-up provided)
92	ALE	O	Address Latch Enable Output. Can be used for latching the address into external memory or an address latch in the multiplexed bus modes. (internal pull-down provided)

Table 2-2 Serial Interfaces,Terminal Specific Functions

Pin No.	Symbol	Input (I) Output (O) Open Drain (OD)	Function
	PORT3 P3.0 - P3.15	I/O (OD possible at all pins)	PORT3 is a 16-bit bidirectional I/O port. It is bit-wise programmable for input or output via direction bits and it contains internal pull resistors. The following PORT3 pins serve for alternate functions: P3.0 MRST1 Master Receive Slave Transmit
130		I/O	SSC1 / T5IN Timer 5 Input P3.1 MTSR1 Master Transmit Slave Receive
131		I/O	SSC1 / T4EUD Timer 4 External Up Down P3.2 SCLK1 Shift Clock SSC1 / T2EUD Timer 2 External Up Down
134		I/O	P3.3 T3OUT Timer T3 Toggle Latch Output P3.4 T3EUD Timer T3 External Up Down
135		O	P3.5 T4IN Timer T4 Input
136		I	P3.6 T3IN Timer T3 Input
137		I	P3.7 T2IN Timer T2 Input
138		I	P3.8 MRST0 Master Receive Slave Transmit
139		I	SSC0 / T6IN Timer 6 Input
140		I/O	P3.9 MTSR0 Master Transmit Slave Receive SSC0 / T5EUD Timer T5 External Up Down
1		I/O	P3.10 TxD ASC Data transmit P3.11 RxD ASC Data receive P3.12 BHE External Memory High Byte Enable
2		O	Signal WRH External Memory High Byte Write
3		I/O	Strobe
4		O	P3.13 SCLK0 Shift Clock SSC0 / T6EUD Timer 6 External Up Down P3.14 T6OUT Timer 6 output
5		O/I	P3.15 CAPIN GPT2 Capture Input
10		O	
11		I	

Table 2-2 Serial Interfaces,Terminal Specific Functions (cont'd)

Pin No.	Symbol	Input (I) Output (O) Open Drain (OD)	Function
	PORT7 P7.0 - P7.9	I/O (OD)	PORT7 is an 10-bit bidirectional I/O port. It is bit-wise programmable for input or output via direction bits. It contains internal pull resistors. As alternate function the following pins are used by the keyscanner
			P7.0 Keyscan of line 0
120		I/O	P7.1Keyscan of line 1
121		I/O	P7.2 Keyscan of line 2
122		I/O	P7.3Keyscan of line 3
123		I/O	P7.4 Keyscan of line 4
124		I/O	P7.5 Keyscan of line 5
125		I/O	P7.6 Keyscan of line 6
126		I/O	P7.7 Keyscan of line 7
127		I/O	P7.8 Keyscan of line 8
128		I/O	P7.9 Keyscan of line 9
129		I/O	
	PORT2 P2.0- P2.13	I/O (OD)	PORT2 is a 14-bit bidirectional I/O port. It is bit-wise programmable for input or output via direction bits. It contains internal pull resistors. The following pins serve for alternate functions.
			P2.0 FEX0IN or SDS1
			P2.1 FEX1IN or SDS2
15		I/O	P2.2 FEX2IN
16		I/O	P2.3 LED multiplexing line 0
19		I/O	P2.4 LED multiplexing line 1
20		O	P2.5 LED multiplexing line 2
21		O	P2.6 LED multiplexing line 3
22		O	P2.7 LED multiplexing line 4
23		O	P2.8 LED multiplexing line 5
24		O	P2.9 LED multiplexing line 6
25		O	P2.10 LED multiplexing line 7
45		O	P2.11 LED multiplexing line 8
46		O	P2.12 LED multiplexing line 9
47		O	P2.13 LED multiplexing line 10
48		O	
49		O	

Table 2-3 USB Interface

Pin No.	Symbol	Input (I) Output (O) Open Drain (OD)	Function
110	DPLS	I/O	USB Data+ input/output signal.
111	DMNS	I/O	USB Data- input/output signal.

Table 2-4 IOM-2 Interface and Strobe Signals

Pin No.	Symbol	Input (I) Output (O) Open Drain (OD)	Function
13	DD	I/OD/O	IOM-2 Data Downstream Signal pin. For the pin configured as input, the output driver is put into high-impedance state. Open-drain.
12	DU	I/OD/O	IOM-2 Data Upstream Signal pin. For the pin configured as input, the output driver is put into high-impedance state. Open-drain.
14	FSC	I/O	Frame Synchronization Clock
8	DCL	I/O	Data Clock (double-bit clock)
9	BCL	I/O	Data Clock (bit clock)

Table 2-5 RESET

Pin No.	Symbol	Input (I) Output (O) Open Drain (OD)	Function
117	$\overline{\text{RSTIN}}$	I	Reset Input. A low level at this pin for a specified duration while the oscillator is running, resets the device. (Internal pull-up provided).
116	$\overline{\text{RSTOUT}}$	O	Internal Reset Indication Output. This pin is set to a low level when the device is executing either a hardware-, software- or a watchdog timer reset. $\overline{\text{RSTOUT}}$ remains low until the EINIT (end of initialization) instruction is executed.
26	$\overline{\text{NMI}}$	I	Non-Maskable Interrupt Input. A high to low transition at this pin causes the CPU to vector to the NMI trap routine. When the PWRDN (power down) instruction is executed, the $\overline{\text{NMI}}$ pin must be low in order to force the CPU to go into power down mode. If $\overline{\text{NMI}}$ is high, when PWRDN is executed, the device will continue to run in normal mode. If not used, pin $\overline{\text{NMI}}$ should be pulled high externally. However, it is possible to tie $\overline{\text{NMI}}$ permanently to VSS.

Table 2-6 Boundary Scan, JTAG , OCDS

Pin No.	Symbol	Input (I) Output (O) Open Drain (OD)	Function
105	TEST	I	Internal Test Mode Enable (tied to VSS)
104	TCK	I	Test Clock Input (internal pull-up)
103	TDI	I	Test Data Input (internal pull-up)
108	TDO	O	Boundary Scan Test Data Output
107	TMS	I	Test Mode Select (internal pull-up)
106	$\overline{\text{TRST}}$	I	Test Reset (internal pull-down)

Table 2-6 Boundary Scan, JTAG , OCDS

Pin No.	**Symbol**	**Input (I) Output (O) Open Drain (OD)**	**Function**
50	$\overline{\text{BRKIN}}$	I	In OCDS mode, a falling edge from HIGH to LOW signal on $\overline{\text{brkin}}$ forces the system to stop. An internal pull-up resistor is provided.
51	$\overline{\text{BRKOUT}}$	O	In OCDS mode, a falling edge on $\overline{\text{brkout}}$ indicates the trigger of an pre-selected OCDS event.

Table 2-7 Transceiver / XTAL

Pin No.	**Symbol**	**Input (I) Output (O) Open Drain (OD)**	**Function**
143 144	LIa LIb	I/O I/O	DASL transceiver Line Interface
91 90	XTAL2 XTAL1	O I	Oscillator output Oscillator or 15.36 MHz input

Table 2-8 Analog Front End

Pin No.	Symbol	Input (I) Output (O) Open Drain (OD)	Function
30	V_{REF}	O	1.5V reference voltage for biasing external circuitry.
29	BGREF	I	Bandgap Reference Voltage
36 35	MIP1 MIN1	I I	Symmetrical differential Microphone Input 1
34 33	MIP2 MIN2	I I	Symmetrical differential Microphone Input 2
32 31	MIP3 MIN3	I I	Symmetrical differential Microphone Input 3
41 42	HOP1 HON1	O O	Differential Handset earpiece output
43 44	HOP2 HON2	O O	Differential Handset earpiece output
38 40	LSP LSN	O O	Differential Loudspeaker output

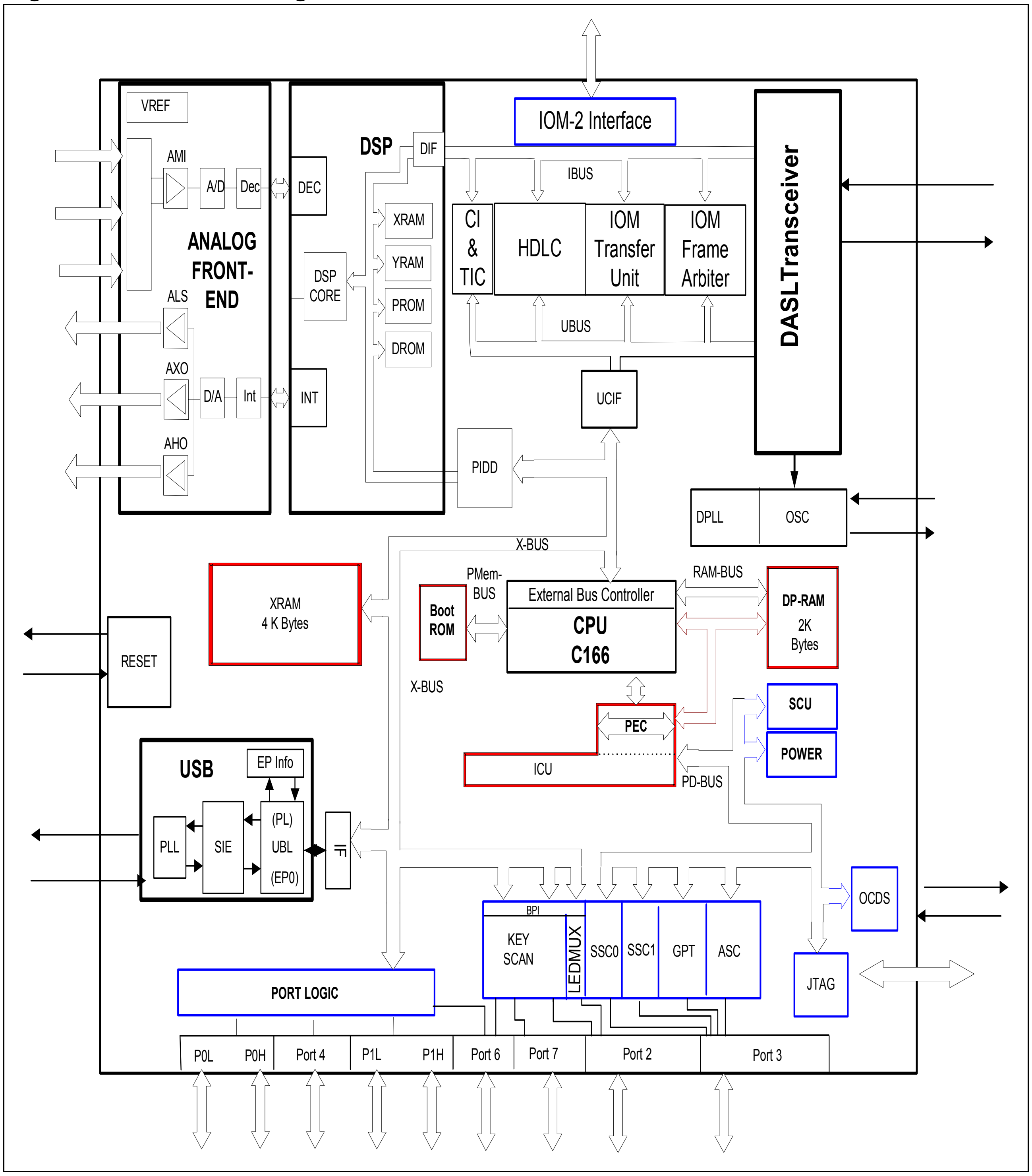
Table 2-9

Pin No.	Symbol	Input (I) Output (O) Open Drain (OD)	Function
			Power supply (3.3 V $-5\%/+10\%$)
142	V_{DDL}	–	Supply voltage for DASL line driver
109	V_{DDU}	–	Supply voltage for USB transceiver
27	V_{DDA}	–	Supply voltage for Analog Front End
37	V_{DDP}	–	Supply voltage for loudspeaker
17	V_{DDLED}	–	Supply voltage for LED mux
102	V_{DDPLL}	–	Supply voltage for PLL
119	V_{DD1}	–	Supply voltage 1 for digital parts
133	V_{DD2}	–	Supply voltage 2 for digital parts
7	V_{DD3}	–	Supply voltage 3 for digital parts
61	V_{DD4}	–	Supply voltage 4 for digital parts
73	V_{DD5}	–	Supply voltage 5 for digital parts
89	V_{DD6}	–	Supply voltage 6 for digital parts
141	V_{SSL}	–	Ground for DASL line driver
112	V_{SSU}	–	Ground for USB transceiver
28	V_{SSA}	–	Ground for Analog Front End
39	V_{SSP}	–	Ground for loudspeaker
18	V_{SSLED}	–	Ground for LED mux
101	V_{SSPLL}	–	Ground for PLL
118	V_{SS1}	–	Ground 1 for digital parts
132	V_{SS2}	–	Ground 2 for digital parts
6	V_{SS3}	–	Ground 3 for digital parts
60	V_{SS4}	–	Ground 4 for digital parts
72	V_{SS5}	–	Ground 5 for digital parts
88	V_{SS6}	–	Ground 6 for digital parts

3 Architectural Overview

3.1 Functional Block Diagram

Figure 3-1 Block Diagram



3.2 Bus Systems

The Analog front End and the DSP are connected using dedicated signals which carry the AD converted and precomputed information. The remaining functional units communicate over different busses.

The **IBUS** carries synchronous data corresponding to the IOM-2 interface.It consists of the Bit Clock, a Chip Select signal, a Read/Write strobe signal, a 3 bit address line and 8 bit lines for data in and data out.

Unlike the IBUS the **UBUS** is used to exchange asynchronous data which can be accessed by the CPU core via the UCIF module which connects XBUS and UBUS.

The INCA-D provides an on-chip interface (the **XBUS** interface), which allows to connect integrated customer/application specific peripherals to the standard controller core. The XBUS is an internal representation of the external bus interface, ie. it is operated in the same way. The XBUS provides a full 24-bit address bus and a 16 bit data bus between the CPU and its X-peripherals (the address width of external bus depends on the selected port configuration). On XBUS, one CPU controlled access can be executed every machine cycle. CPU accesses to XBUS peripherals are synchronous processes, which may be delayed by programmable wait states.

The differentiation between accesses to external bus or internal XBUS is performed within the bus controller by address range detection. For access to internal XBUS peripherals four different address ranges can be selected, each with its own bus type definition. To each address range belongs a chip select signal which also may be shared between more than one X-peripheral.

For accesses to external peripherals 5 address ranges may be selected.

The **Peripheral (PD) -BUS** allows CPU and PEC (Peripheral Event Controller) access to core related and generic peripherals every half machine cycle. The bus cycles are fully synchronized without wait states. The bus also supports read-modify-write cycles without performance losses. Bit modification is provided and bit protection for simultaneous writes is also supported. However, only 512 16-bit registers can be addressed via the demultiplexed address bus.

The **RAM bus** interface provides two 16 bit busses between the dual port RAM and the CPU. Its bus cycles are fully synchronized, with a cycle time of half machine cycle. Thus, four accesses can be made every machine cycle. This performance allows multiple GPR (General Purpose register) accesses to occur without processor stalls. The RAM bus is an internal bus and therefore not visible on core boundary.

4 Clock Concept

The on-chip clock generator provides the INCA-D with all necessary clock signals. Generally, the INCA-D derives its system clocks from an external crystal of 15.36 MHz connected to XTAL1 and XTAL2. An external clock signal may be fed to the input XTAL1, leaving XTAL2 open.

The clock signals for the different device modules are generated as described in **Figure 4-1**. When not disabled by setting bit PPLEN in register CLK_CONF, the on chip PLL generates a signal with a frequency of 96 MHz which is the input signal to a set of fixed and programmable divider. The outputs of the divider are the clock signals for the different device components, which can be disabled separately.

4.1 Clock Generation

The clock generation unit is illustrated in **Figure 4-1**.

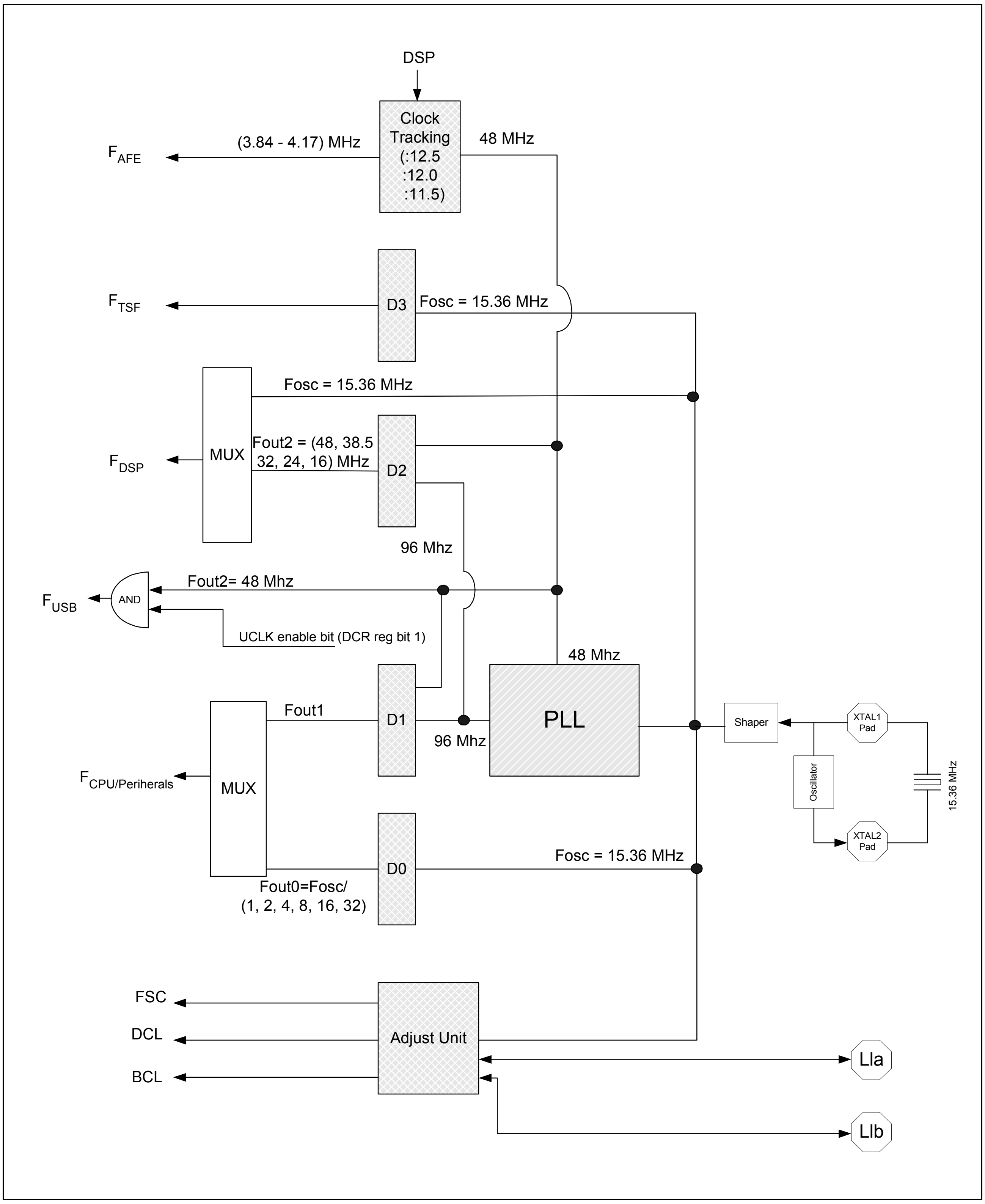


Figure 4-1 Clock Concept

4.2 Terminal Specific Functions

The components realizing the terminal specific functions (Keypad Scanner, LED Multiplex Unit and Pulse Modulation Units) run at a frequency, which is output of divider D3. D3 can not be programmed.

The clock signal for the Terminal Specific Functions can not be disabled globally. As described in **Chapter 21.3**, the different modules can be disabled separately.

4.3 USB

For the USB Device Module the necessary clock of 48 MHz is generated using a clock signal which is being generated from the PLL.

It's important to note that the CPU frequency has to be greater or equal than 12 MHz to ensure a proper operation of the USB module.

The USB module clock can be disabled by setting bit USB_DIS in the CLK_CONF register to '1'. Before the USB interface can be used, the USB clock has to be enabled by setting USB_DIS to '0' and bit UCLK of the USB Device Control Register DCR has to be set to '1' (refer to **Chapter 22**).

A level detect mechanism enables the USB clock as soon as an appropriate level at pins DPLS/DMNS is detected. For further information refer to the USB module description.

4.4 DSP

A bypass mechanism allows the DSP to get a clock signal with the oscillator frequency of 15.36 MHz directly.

If the bypass is not used, the DSP runs with a PLL generated clock signal at a frequency which can be programmed by setting bitfield DSP_DIV in register CLK_CONF according to **Table 4-1**.

A change of the DSP frequency can only be performed synchronously and glitch free, if the following steps will be applied:

- Change to the OSC *as* clock source (if already running on Oscillator, it's okay)
- 5 NOP instructions
- Change the division factor by programming CLK_CONF.DSP_DIV
- At least 50 NOP instructions recommended
- Change the clock source to PLL (if needed)
- 5 NOP instructions

Note: The NOPs are necessary to ensure to have time to change the frequency while the device is running at a much higher frequency as it will be after the change.

The maximum DSP clock frequency of 48 MHz can be reduced, if the full duplex based speakerphone capability is not needed. Without any speakerphone functionality, the lowest frequency of 16 MHz is sufficient for simple phone operation.

Generally, the necessary DSP frequency depends on the functional units that are concurrently in use.

Table 4-1 DSP Frequency

DSP_DIV	Factor	Resulting Frequency
000	2.0	48 MHz
xx1[1)]	2.5	38.5 MHz
010	3.0	32 MHz
100	4.0	24 MHz
110	6.0	16 MHz

1) x means that either a ’0’ or an ’1’ can be entered

4.5 Microcontroller and Peripherals

The peripherals and the CPU are provided with separated clock signals of the same frequency. While the CPU clock is stopped during the idle mode, the peripheral clock keeps running. Both clocks are switched off, when the power down mode is entered. When the powerdown instruction (PWRDWN) is executed, the NMI pin must be low in order to force the CPU to go into power down mode.

A bypass mechanism allows to run the CPU without PLL generated signals. In that case divider D0 may be used to control the clocks for CPU and peripherals.

The bypass may be controlled by bit CPU_BYP in the CLK_CONF register.

After reset, the bypass is active, i.e. the CPU and peripheral frequency will be 7.68 MHz. Afterwards the divider D0 can be modified or PLL generated clock signals can be selected by software. Changing to the PLL clock has to be done only if the bit CLK_CONF.LOCK indicates a stable PLL output clock.

The programmable divider D0 and D1 are programmed by setting bitfield OSC_DIV or CPU_DIV respectively of register CLK_CONF to generate the clock f_{CPU} for the CPU and the peripherals.

The factors have to be programmed according to **Table 4-2** and **Table 4-3**.

Table 4-2 CPU & Peripheral frequency

OSC_DIV	Factor D0	Resulting Frequency if PLL is not used
101	1	15.36 MHz
000	2	7.68 MHz
001	4	3.84 MHz
010	8	1.92 MHz
011	16	0.96 MHz
100	32	0.48 MHz
110	*reserved*	
111	*reserved*	

Table 4-3 CPU & Peripheral frequency

CPU_DIV	Factor D1	Resulting Frequency
0000	3	32 MHz
0001	4	24 MHz
0010	6	16 MHz
0011	8	12 MHz
0100	12	8 MHz
0101	16	6 MHz
0110	24	4 MHz
0111	32	3 MHz
1000	48	2 MHz
1001	64	1.5 MHz
1100	96	1.0 MHz

A change of the CPU frequency can only be performed synchronously and glitch free, if the following steps will be applied:

- Change to the *other* clock source (if running on PLL, switch to Oscillator or vice versa)
- 5 NOP instructions
- Change the division factor by programming CLK_CONF.CPU_DIV or CLK_CONF.OSC_DIV respectively
- At least 50 NOP instructions recommended

- Change the clock source back
- 5 NOP instructions

Note: The NOPs are necessary to ensure to have time to change the frequency while the device is running at a much higher frequency as it will be after the change. The minimum frequency for a proper USB operation is 12 MHz.

4.6 AFE

The input clock of the Analog Front End is a 4 MHz clock signal that is generated by a dedicated divider. If enabled (PIDDHWCFG.ACT = 1), the clock tracking unit takes care that the AFE clock is always synchronous to the frame synchronization provided by the interface to the IOM-2 handler.

4.7 IOM-2 clocks

The INCA-D operates as clock master and the transmit and receive bit clocks are derived, with the help of the Adjust Unit, from the DASL interface receive data stream. The received signal is sampled several times inside the derived receive clock period, and a majority logic is used to additionally reduce bit error rate in severe conditions.

IThe FSC, DCL and BCL clocks are summarized below with the respective duty cycles.

FSC 8 kHz	1:2
BCL 768 kHz	1:1
DCL 1536 kHz	1:1

For further details refer to the IOM-2 handler module description.

4.8 Register Description of the Clock Concept Unit

CLK_CONF (DFAA $_H$) **XBUS**-SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CPU _BYP	**DSP _BYP**	**CPU_DIV**				**DSP_DIV**			**0**	**LOCK**	**OSC_DIV**			**USB _DIS**	**PLL EN**
rw	rw	rw				rw			r	r	rw	rw	rw	rw	rw

Bit	Function
PLLEN	**PLL enable** 0: PLL disabled *Note: The PLL should only be disabled after the bypass for the CPU has been enabled, ie. CPU_BYP set to ’0’. Otherwise the behavior is undefined.* 1: PLL enabled
USB_DIS	**USB clock disable** 0: USB clock enabled 1: USB clock disabled
LOCK	**PLL LOCK** 0: PLL not locked; 1: PLL locked, clock source switched to PLL clock if CPU_BYP is ’1’.
OSC_DIV	**CPU Clock Speed** Divider value for CPU/Peripheral frequency in bypass mode (refer to **Table 4-2**)
DSP_DIV	**DSP Clock Speed** Divider value for DSP frequency when not in bypass mode (refer to **Table 4-1**)
CPU_DIV	**CPU Clock Speed** Divider value for CPU/Peripheral frequency when not in bypass mode (refer to **Table 4-3**)
DSP_BYP	**Bypass for DSP Clock** 1: PLL generated signal is used; 0: Oscillators generated signal is directly used *Note: Because the LOCK bit and the DSP_BYP bit are internally ANDED, the clock source is set to PLL only after the PLL is locked.*
CPU_BYP	**Bypass for CPU/Peripheral Clock** 1: PLL generated signal is used; 0: Oscillators generated signal is directly used *Note: Because the LOCK bit and the CPU_BYP bit are internally ANDED, the clock source is set to PLL only after the PLL is locked.*

5 Memory Organization

The memory space of the INCA-D is configured in a “Von Neumann” architecture. This means that code and data are accessed within the same linear address space. All of the physically separated memory areas, including internal Dual-Port RAM with the internal Special Function Register Areas (SFRs and ESFRs), the address areas for integrated XBUS peripherals and external memory are mapped into one common address space.

Generally, the CPU core of the INCA-D provides a total addressable memory space of 16 MBytes. This address space is arranged as 256 segments of 64 KBytes each, and each segment is again subdivided into four data pages of 16 KBytes each

However, the addressable external memory space of the INCA-D is limited to 4 MBytes[1].

Bytes are stored at even or odd byte addresses. Words are stored in ascending memory locations with the low byte at an even byte address being followed by the high byte at the next odd byte address. Double words (code only) are stored in ascending memory locations as two subsequent words. Single bits are always stored in the specified bit position at a word address. Bit position 0 is the least significant bit of the byte at an even byte address, and bit position 15 is the most significant bit of the byte at the next odd byte address. Bit addressing is supported for a part of the Special Function Registers, a part of the internal RAM and for the General Purpose Registers.

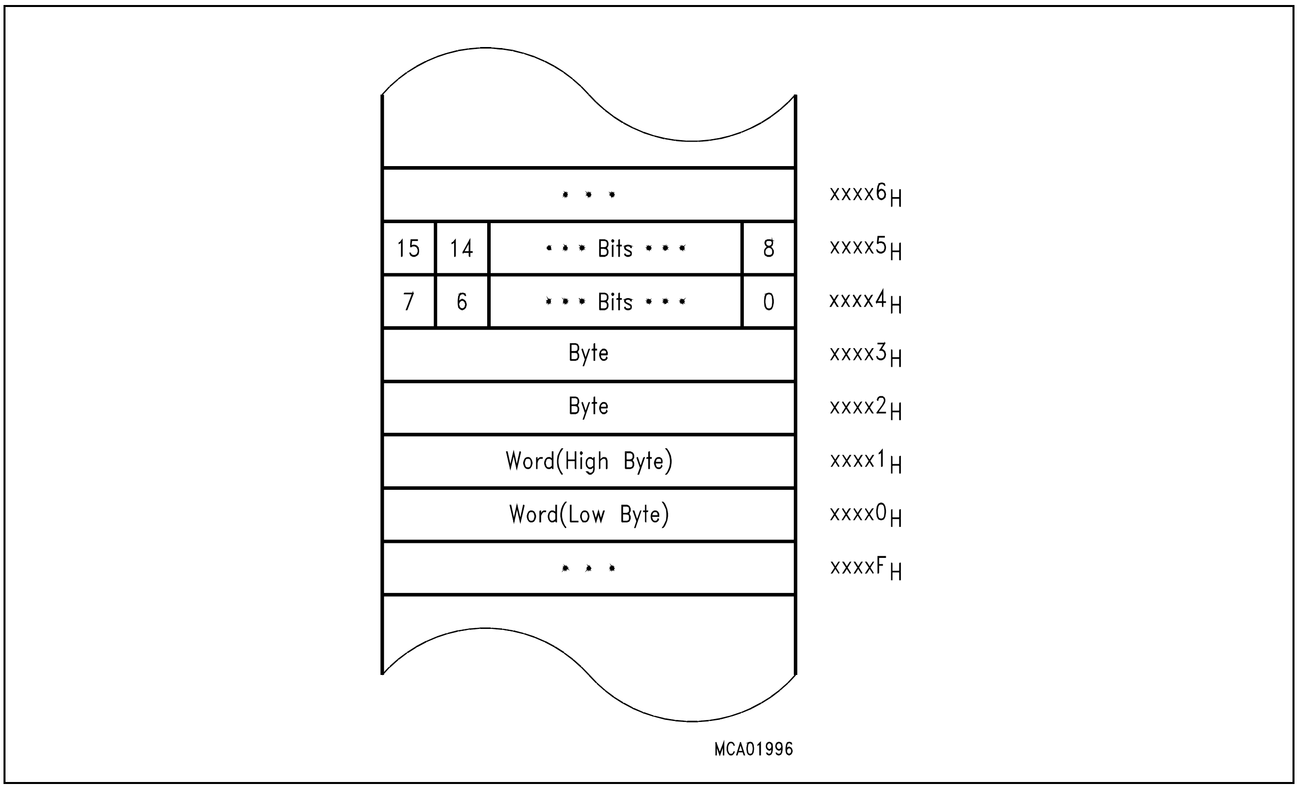


Figure 5-1 Storage of Words, Byte and Bits in a Byte Organized Memory

[1] Each of the 2 chip selects can be used to access a 4 MB window of memory

Note: Byte units forming a single word or a double word must always be stored within the same physical (internal, external, ROM, RAM) and organizational (page, segment) memory area.

5.1 Internal ROM (Bootstrap Loader)

The INCA-D includes an internal ROM for bootstrap loader functionality only. Therefore, there is **no internal ROM** available for customized program code storage.

The BSL mechanism may be used for standard system startup as well as only for special occasions like system maintenance (firmware update) or end-of-line programming or testing

The INCA-D enters BSL mode, if pin P0L.4 is sampled low at the end of a hardware reset (See **Chapter 24.6**). After entering BSL mode and the respective internal initialization the INCA-D scans the RXD0 line to receive a zero byte, ie. one start bit, eight ‘0’ data bits and one stop bit. From the duration of this zero byte it calculates the corresponding baudrate factor with respect to the current CPU clock, initializes the serial interface ASC accordingly and switches pin TxD0 to output.

In order to execute a program in normal mode, the BSL mode has to be terminated first. The INCA-D exits BSL mode upon software reset (ignores the level on P0L.4) or a hardware reset (P0L.4 must be high then). After reset the INCA-D will start executing from location 00’0000$_H$ of the external memory.

5.2 On Chip XRAM

The INCA-D provides 4 KBytes of On Chip XRAM. The XRAM is mapped on data page 3 from address 00’E000$_H$ to 00’EFFF$_H$ as shown in **Figure 5-2**.

5.3 Internal Dual-Port-RAM and SFR Area

The RAM/SFR area is located within data page 3 and provides access to the internal DP-RAM (IRAM, organized as X*16) and to two 512 Byte blocks of Special Function Registers (SFRs). The INCA-D provides 2 KBytes of IRAM, see **Figure 5-2**.

The IRAM serves for several purposes:

- System Stack (programmable size)
- General Purpose Register Banks (GPRs)
- Source and destination pointers for the Peripheral Event Controller (PEC)
- Variable and other data storage, or
- Code storage.

5.4 XBUS Peripherals

The peripherals can be grouped into peripherals which are connected to the PD-bus (GPT, ASC, SSC) and peripherals which are connected to the XBUS (see **Figure 3-1**). While the PD-bus peripherals are configured by the SFR or ESFR respectively, the X-bus peripherals (e.g. USB, I2C, IOM handler) are configured by XBUS SFR'S which are located in the XPER area as described in **Figure 5-2**.

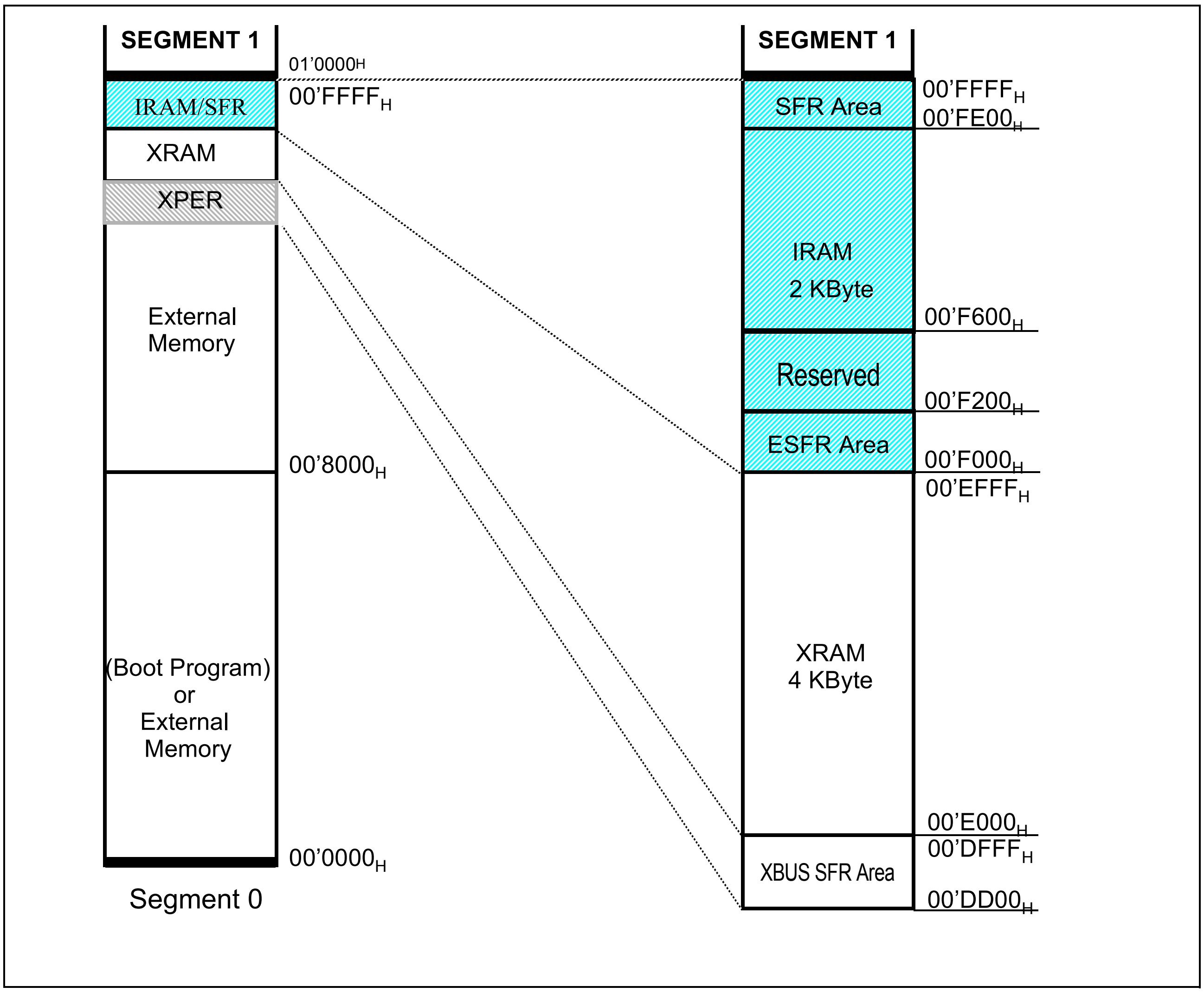


Figure 5-2 Memory Areas and Address Space

Code accesses are always made on even byte addresses. The highest possible code storage location in the internal DP-RAM is either 00'FDFE$_H$ for single word instructions or 00'FDFC$_H$ for double word instructions. The respective location must contain a branch instruction (unconditional), because sequential boundary crossing from internal RAM to the SFR area is not supported and causes erroneous results.

Any word and byte data in the internal DP-RAM can be accessed via indirect or long 16-bit addressing modes, if the selected DPP register points to data page 3.

Any word data access is made on an even byte address. The highest possible word data storage location in the internal RAM is 00'FDFE$_H$. For PEC data transfers, the internal RAM can be accessed independent of the contents of the DPP registers via the PEC source and destination pointers.

The upper 256 Byte of the internal RAM (00'FD00$_H$ through 00'FDFF$_H$) and the GPRs of the current bank are provided for single bit storage, and thus they are bit addressable.

5.5 System Stack

The system stack may be defined within the internal DP-RAM. The size of the system stack is controlled by bitfield STKSZ in register SYSCON (see table below).

<STKSZ>	Stack Size (Words)	Internal RAM Addresses (Words)
0 0 0 $_B$	256	00'FBFE$_H$...00'FA00$_H$ (Default after Reset)
0 0 1 $_B$	128	00'FBFE$_H$...00'FB00$_H$
0 1 0 $_B$	64	00'FBFE$_H$...00'FB80$_H$
0 1 1 $_B$	32	00'FBFE$_H$...00'FBC0$_H$
1 0 0 $_B$	512	00'FBFE$_H$...00'F800$_H$
1 0 1 $_B$	---	Reserved. Do not use this combination.
1 1 0 $_B$	---	Reserved. Do not use this combination.
1 1 1 $_B$	1024	00'FDFE$_H$...00'F600$_H$ (Note: No circular stack)

For all system stack operations the on-chip DP-RAM is accessed via the Stack Pointer (SP) register. The stack grows downward from higher towards lower RAM address locations. Only word accesses are supported to the system stack. A stack overflow (STKOV) and a stack underflow (STKUN) register are provided to control the lower and upper limits of the selected stack area. These two stack boundary registers can be used not only for protection against data destruction, but also allow to implement a circular stack with hardware supported system stack flushing and filling (except for option '111').

5.6 General Purpose Registers

The General Purpose Registers (GPRs) use a block of 16 consecutive words within the internal DP-RAM. The Context Pointer (CP) register determines the base address of the currently active register bank. This register bank may consist of up to 16 word GPRs (R0, R1, ..., R15) and/or of up to 16 byte GPRs (RL0, RH0, ..., RL7, RH7). The sixteen byte GPRs are mapped onto the first eight word GPRs (see table below).

In contrast to the system stack, a register bank grows from lower towards higher address locations and occupies a maximum space of 32 Byte. The GPRs are accessed via short 2-, 4- or 8-bit addressing modes using the Context Pointer (CP) register as base address (independent of the current DPP register contents).

Additionally, each bit in the currently active register bank can be accessed individually.

Mapping of General Purpose Registers to RAM Addresses

Internal RAM Address	Byte Registers		Word Register
<CP> + $1E_H$	---		R15
<CP> + $1C_H$	---		R14
<CP> + $1A_H$	---		R13
<CP> + 18_H	---		R12
<CP> + 16_H	---		R11
<CP> + 14_H	---		R10
<CP> + 12_H	---		R9
<CP> + 10_H	---		R8
<CP> + $0E_H$	RH7	RL7	R7
<CP> + $0C_H$	RH6	RL6	R6
<CP> + $0A_H$	RH5	RL5	R5
<CP> + 08_H	RH4	RL4	R4
<CP> + 06_H	RH3	RL3	R3
<CP> + 04_H	RH2	RL2	R2
<CP> + 02_H	RH1	RL1	R1
<CP> + 00_H	RH0	RL0	R0

The INCA-D supports fast register bank (context) switching. Multiple register banks can physically exist within the internal DP-RAM at the same time. Only the register bank selected by the Context Pointer register (CP) is active at a given time, however. Selecting a new active register bank is simply done by updating the CP register. A particular Switch Context (SCXT) instruction performs register bank switching and an automatic saving of the previous context. The number of implemented register banks (arbitrary sizes) is only limited by the size of the available internal RAM.

5.7 PEC Source and Destination Pointers

The 16 word locations in the internal DP-RAM from 00’FCE0_H to 00’FCFE_H (just below the bit-addressable section) are provided as source and destination address pointers for data transfers using the eight PEC channels. Each channel uses a pair of pointers stored in two subsequent word locations with the source pointer (SRCPx) on the lower and the destination pointer (DSTPx) on the higher word address (x = 7...0).

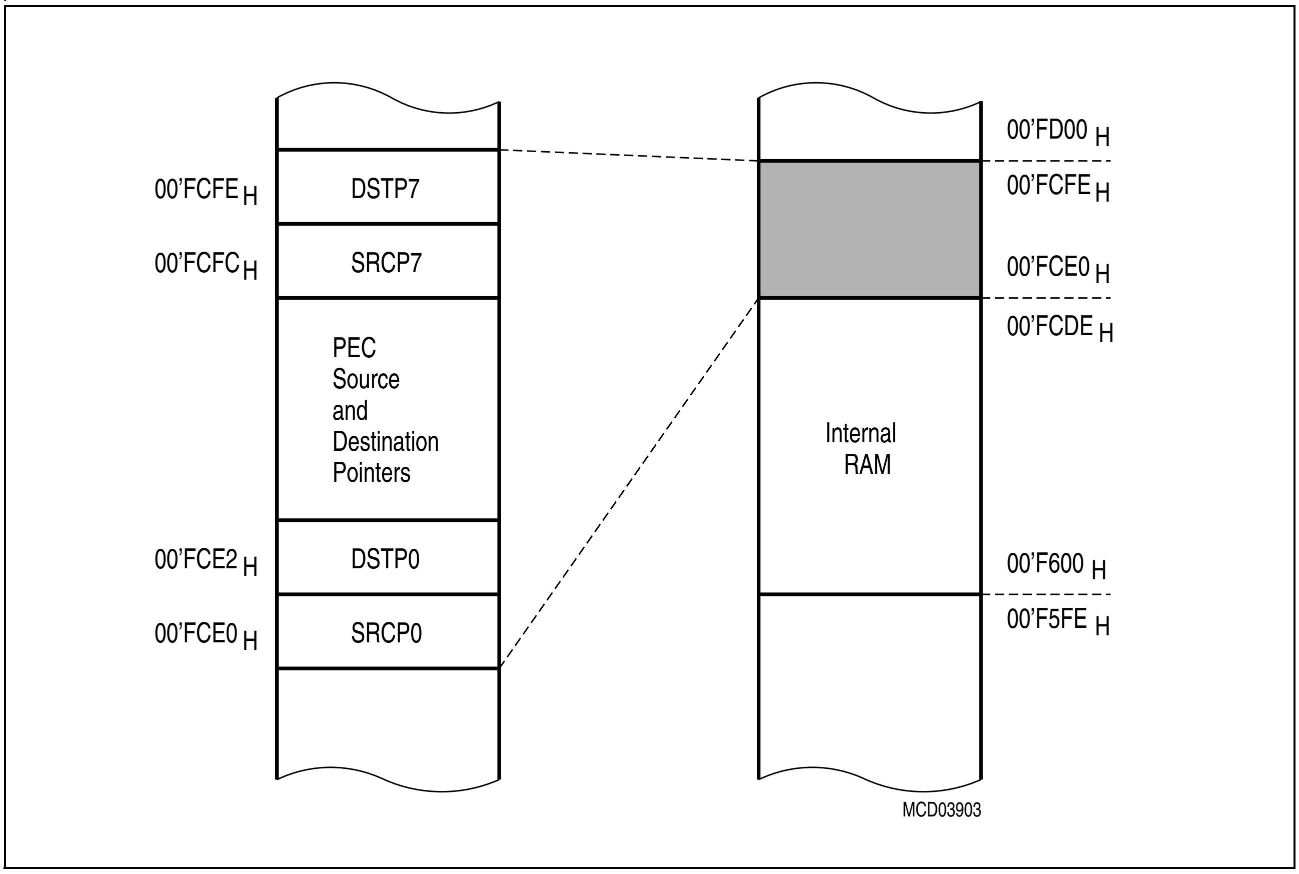


Figure 5-3 Location of the PEC Pointers

Whenever a PEC data transfer is performed, the pair of source and destination pointers, which is selected by the specified PEC channel number, is accessed independent of the current DPP register contents and also the locations referred to by these pointers are accessed independent of the current DPP register contents. If a PEC channel is not used, the corresponding pointer location area is available and can be used for word or byte data storage.

5.8 Special Function Registers

The functions of the CPU, the bus interface, the I/O ports and the on-chip peripherals of the INCA-D are controlled via a number of so-called Special Function Registers (SFRs). These SFRs are arranged within two areas of 512 Byte size each. The first register block, the SFR area, is located in the 512 Bytes above the internal DP-RAM (00'FFFF$_H$...00'FE00$_H$), the second register block, the Extended SFR (ESFR) area, is located in the 512 Bytes below the internal RAM (00'F1FF$_H$...00'F000$_H$).

Special function registers can be addressed via indirect and long 16-bit addressing modes. Using an 8-bit offset together with an implicit base address allows to address word SFRs and their respective low bytes. However, this **does not work** for the respective high bytes!

Note: Writing to any byte of an SFR causes the non-addressed complementary byte to be cleared!

The upper half of each register block is bit-addressable, so the respective control/status bits can directly be modified or checked using bit addressing.

5.9 External Memory Space

The INCA-D is capable to address 4 MBytes of external memory space (for each chip select).

This external memory is accessed via the INCA-D's external bus interface.

Four memory bank sizes are supported:

- Non-segmented mode: 64 KBytes with A15...A0 on PORT0 or PORT1
- 2-bit segmented mode: 256 KByteswith A17...A16 on Port 4 and A15...A0 on PORT0 or PORT1
- 4-bit segmented mode: 1 MBytes with A19...A16 on Port 4 and A15...A0 on PORT0 or PORT1
- 6-bit segmented mode: 4 MBytes with A21...A16 on Port 4 and A15...A0 on PORT0 or PORT1

Each bank can be directly addressed via the address bus, while the programmable chip select signals can be used to select various memory banks.

The INCA-D also supports **four different bus types**:

- Multiplexed 16-bit Bus with address and data on PORT0
- Multiplexed 8-bit Bus with address and data on PORT0 or P0L(pins 0-7 of Port0)respectively
- Demultiplexed 16-bit Bus with address on PORT1 and data on PORT0
- Demultiplexed 8-bit Bus with address on PORT1 and data on P0L (Default after Reset)

Memory model and bus mode are selected during reset by PORT0 pins. (For further details refer to Chapter 'The External Bus Interface')

Please note that the external memory size of segment 0 is limited to the lower 56 KBytes. If the program code exceeds this boundary, it has to be continued in segment 1.

Note: When operating in non-segmented mode only addresses from 00'0000$_H$ to 00'CFFF$_H$ can be used for external memory access.

The different memory areas must be switched explicitly via branch instructions. Sequential boundary crossing is not supported and leads to erroneous results.

External word and byte data can only be accessed via indirect or long 16-bit addressing modes using one of the four DPP registers. There is no short addressing mode for external operands. Any word data access is made to an even byte address.

For PEC data transfers the external memory can be accessed independent of the contents of the DPP registers via the PEC source and destination pointers.

The external memory is not provided for single bit storage and therefore it is not bit addressable.

5.10 Crossing Memory Boundaries

The address space of the INCA-D is implicitly divided into equally sized blocks of different granularity and into logical memory areas. Crossing the boundaries between these blocks (code or data) or areas requires special attention to ensure that the controller executes the desired operations.

Memory Areas are partitions of the address space that represent different kinds of memory (if provided at all). These memory areas are the internal RAM/SFR area, the on-chip general purpose RAM and the external memory.

Accessing subsequent data locations that belong to different memory areas is no problem. However, when executing code, the different memory areas must be switched explicitly via branch instructions. Sequential boundary crossing is not supported and leads to erroneous results.

Segments are contiguous blocks of 64 KByte each. They are referenced via the code segment pointer CSP for code fetches and via an explicit segment number for data accesses overriding the standard DPP scheme.
During code fetching segments are not changed automatically, but rather must be switched explicitly. The instructions JMPS, CALLS and RETS will do this.
In larger sequential programs make sure that the highest used code location of a segment contains an unconditional branch instruction to the respective following segment, to prevent the prefetcher from trying to leave the current segment.

Data Pages are contiguous blocks of 16 KByte each. They are referenced via the data page pointers DPP3...0 and via an explicit data page number for data accesses overriding the standard DPP scheme. Each DPP register can select one of the possible 1024 data pages. The DPP register that is used for the current access is selected via the two upper bits of the 16-bit data address. Subsequent 16-bit data addresses that cross the 16 KByte data page boundaries therefore will use different data page pointers, while the physical locations need not be subsequent within memory.

6 Central Processor Unit

Basic tasks of the CPU are to fetch and decode instructions, to supply operands for the arithmetic and logic unit (ALU), to perform operations on these operands in the ALU, and to store the previously calculated results.

Since a four stage pipeline is implemented in the INCA-D, up to four instructions can be processed in parallel. Most instructions of the INCA-D are executed in one machine cycle (2 CPU clock cycles) due to this parallelism. This chapter describes how the pipeline works for sequential and branch instructions in general, and which hardware provisions have been made to speed the execution of jump instructions in particular. The general instruction timing is described including standard and exceptional timing.

While internal memory accesses are normally performed by the CPU itself, external peripheral or memory accesses are performed by a particular on-chip External Bus Controller (EBC), which is automatically invoked by the CPU whenever a code or data address refers to the external address space. If possible, the CPU continues operating while an external memory access is in progress. If external data are required but are not yet available, or if a new external memory access is requested by the CPU, before a previous access has been completed, the CPU will be held by the EBC until the request can be satisfied. The EBC is described in a dedicated chapter.

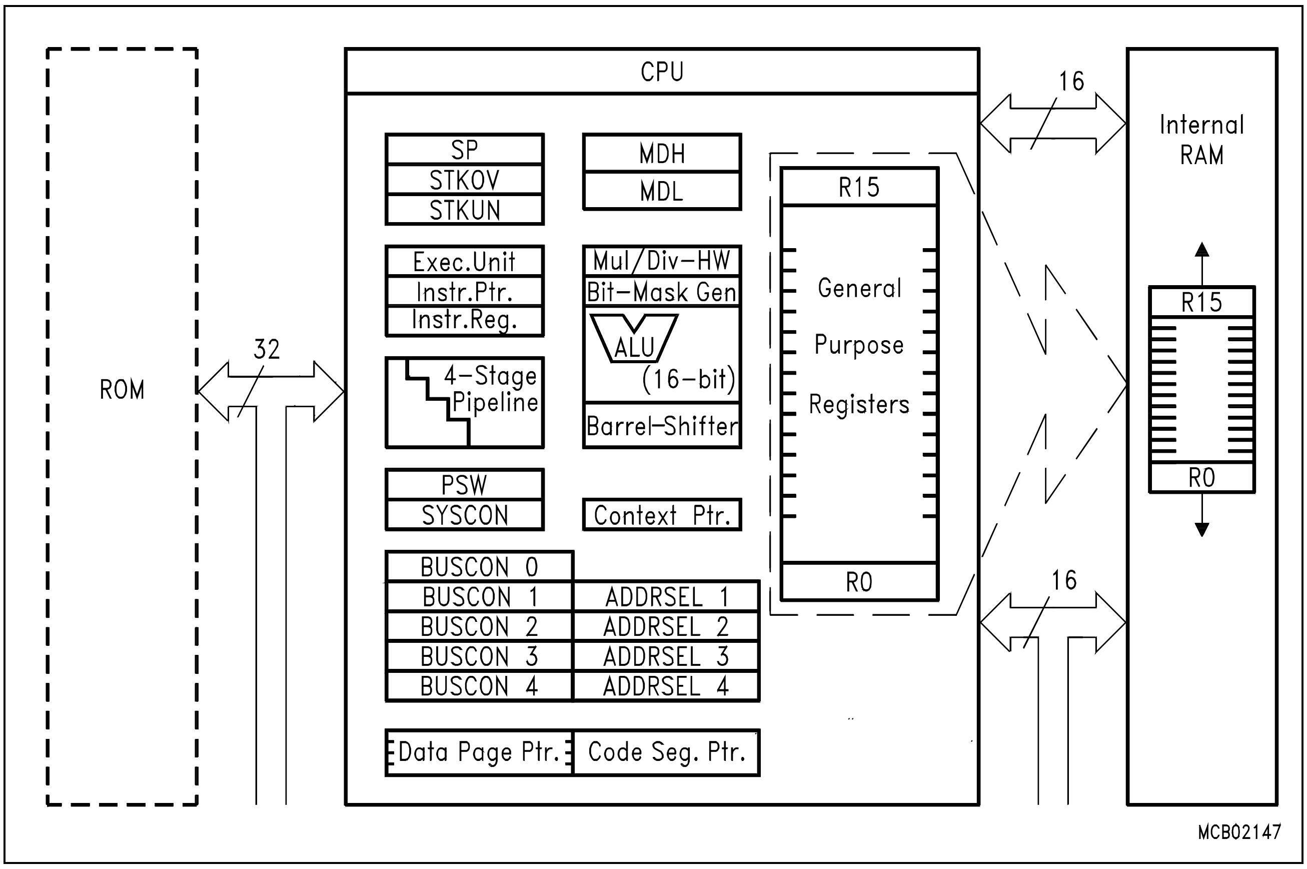


Figure 6-1 CPU Block Diagram

The on-chip peripheral units of the INCA-D work nearly independent of the CPU. Data and control information is interchanged between the CPU and these peripherals via Special Function Registers (SFRs). Whenever peripherals need a non-deterministic CPU action, an on-chip Interrupt Controller compares all pending peripheral service requests against each other and prioritizes one of them. If the priority of the current CPU operation is lower than the priority of the selected peripheral request, an interrupt will occur.

A set of Special Function Registers is dedicated to the functions of the CPU core:

•General System Configuration: **SYSCON (RP0H)**

•CPU Status Indication and Control: **PSW**

•Code Access Control: **IP, CSP**

•Data Paging Control: **DPP0, DPP1, DPP2, DPP3**

•GPRs Access Control: **CP**

•System Stack Access Control: **SP, STKUN, STKOV**

•Multiply and Divide Support: **MDL, MDH, MDC**

•ALU Constants Support: **ZEROS, ONES**

6.1 Instruction Pipelining

The instruction pipeline of the INCA-D partitiones instruction processing into four stages of which each one has its individual task:

1st –>FETCH:
In this stage the instruction selected by the Instruction Pointer (IP) and the Code Segment Pointer (CSP) is fetched from either the internal ROM (bootstrap loader), internal RAM, or external memory.

2nd –>DECODE:
In this stage the instructions are decoded and, if required, the operand addresses are calculated and the respective operands are fetched. For all instructions, which implicitly access the system stack, the SP register is either decremented or incremented, as specified. For branch instructions the Instruction Pointer and the Code Segment Pointer are updated with the desired branch target address (provided that the branch is taken).

3rd –>EXECUTE:
In this stage an operation is performed on the previously fetched operands in the ALU. Additionally, the condition flags in the PSW register are updated as specified by the instruction. All explicit writes to the SFR memory space and all auto-increment or auto-decrement writes to GPRs used as indirect address pointers are performed during the execute stage of an instruction, too.

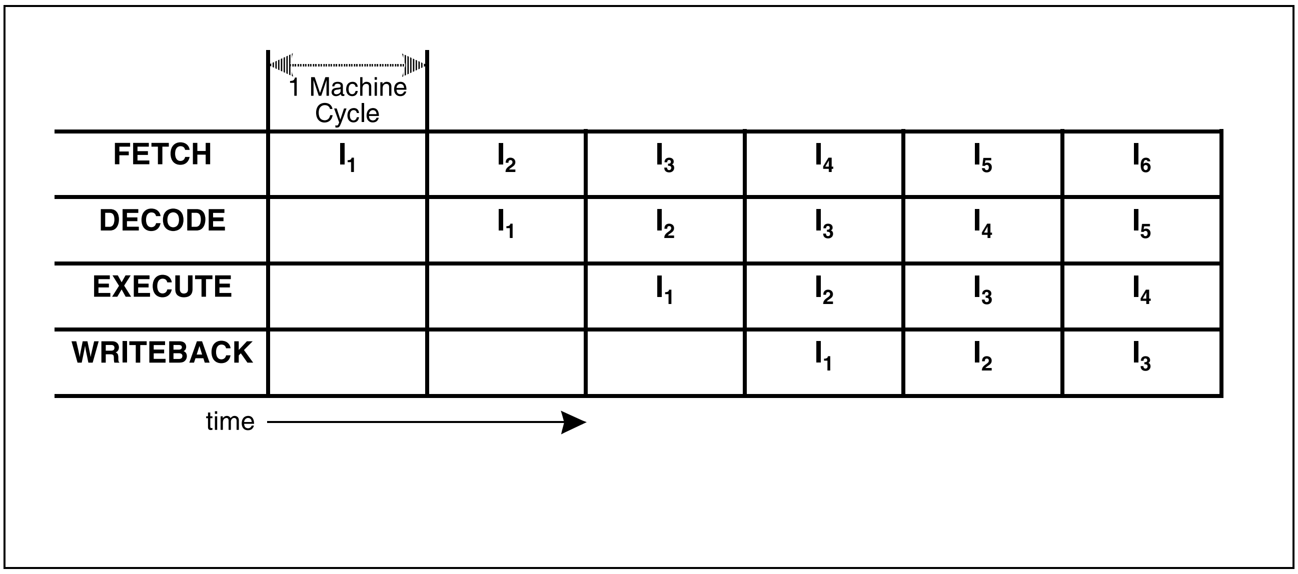
4th –>WRITE BACK:
In this stage all external operands and the remaining operands within the internal RAM space are written back.

A particularity of the INCA-D are the so-called injected instructions. These injected instructions are generated internally by the machine to provide the time needed to process instructions, which cannot be processed within one machine cycle. They are automatically injected into the decode stage of the pipeline, and then they pass through the remaining stages like every standard instruction. Program interrupts are performed by means of injected instructions, too. Although these internally injected instructions will not be noticed in reality, they are introduced here to ease the explanation of the pipeline in the following.

Sequential Instruction Processing

Each single instruction has to pass through each of the four pipeline stages regardless of whether all possible stage operations are really performed or not. Since passing through one pipeline stage takes at least one machine cycle, any isolated instruction takes at least four machine cycles to be completed. Pipelining, however, allows parallel (ie. simultaneous) processing of up to four instructions. Thus, most of the instructions seem to be processed during one machine cycle as soon as the pipeline has been filled once after reset (see figure below).

Instruction pipelining increases the average instruction throughput considered over a certain period of time. In the following, any execution time specification of an instruction always refers to the average execution time due to pipelined parallel instruction processing.



FETCH	I_1	I_2	I_3	I_4	I_5	I_6
DECODE		I_1	I_2	I_3	I_4	I_5
EXECUTE			I_1	I_2	I_3	I_4
WRITEBACK				I_1	I_2	I_3

Figure 6-2 Sequential Instruction Pipelining

6.2 CPU Special Function Registers

The core CPU requires a set of Special Function Registers (SFRs) to maintain the system state information, to supply the ALU with register-addressable constants and to control system and bus configuration, multiply and divide ALU operations, code memory segmentation, data memory paging, and accesses to the General Purpose Registers and the System Stack.

The access mechanism for these SFRs in the CPU core is identical to the access mechanism for any other SFR. Since all SFRs can simply be controlled by means of any instruction, which is capable of addressing the SFR memory space, a lot of flexibility has been gained, without the need to create a set of system-specific instructions.

Note, however, that there are user access restrictions for some of the CPU core SFRs to ensure proper processor operations. The instruction pointer IP and code segment pointer CSP cannot be accessed directly at all. They can only be changed indirectly via branch instructions.

The PSW, SP, and MDC registers can be modified not only explicitly by the programmer, but also implicitly by the CPU during normal instruction processing. Note that any explicit write request (via software) to an SFR supersedes a simultaneous modification by hardware of the same register.

Note: Any write operation to a single byte of an SFR clears the non-addressed complementary byte within the specified SFR.
Non-implemented (reserved) SFR bits cannot be modified, and will always supply a read value of '0'.

The System Configuration Register SYSCON

This bit-addressable register provides general system configuration and control functions. The reset value for register SYSCON depends on the state of the PORT0 pins during reset .

SYSCON (FF12$_H$ / 89$_H$) **SFR-b** **Reset Value: EA84$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	STKSZ		**0**	**SGT DIS**	**0**	**BYT DIS**	**CLK EN**	**WR CFG**	**CS CFG**	**0**	**0**	**0**	**XPEN**	**VISI BLE**	**0**
	rw		r	rw	r	rw	rw	rw	rw	r	r	r	rw	rw	r

Bit	Function
VISIBLE	**Visible Mode Control** 0: Accesses to XBUS peripherals are done internally 1: XBUS peripheral accesses are made visible on the external pins
XPEN	**XBUS Peripheral Enable Bit** 0: Accesses to the on-chip X-Peripherals and their functions are disabled 1: The on-chip X-Peripherals are enabled and can be accessed
CSCFG	**Chip Select Configuration Control** 0: Latched $\overline{CS}$ mode. The $\overline{CS}$ signals are latched internally and driven to the (enabled) port pins synchronously. 1: Unlatched $\overline{CS}$ mode. The $\overline{CS}$ signals are directly derived from the address and driven to the (enabled) port pins.
WRCFG	**Write Configuration Control** (Set according to pin P0H.0 during reset) 0: Pins $\overline{WR}$ and $\overline{BHE}$ retain their normal function 1: Pin $\overline{WR}$ acts as $\overline{WRL}$, pin $\overline{BHE}$ acts as $\overline{WRH}$
CLKEN	**System Clock Output Enable** (CLKOUT [1]) 0: CLKOUT disabled 1: CLKOUT enabled;
BYTDIS	**Disable/Enable Control for Pin $\overline{BHE}$** (Set according to data bus width) 0: Pin $\overline{BHE}$ enabled 1: Pin $\overline{BHE}$ disabled, pin may be used for general purpose I/O
SGTDIS	**Segmentation Disable/Enable Control** ‘0’: Segmentation enabled (CSP is saved/restored during interrupt entry/exit) ‘1’: Segmentation disabled (Only IP is saved/restored)
STKSZ	**System Stack Size** Selects the size of the system stack (in the internal RAM) from 32 to 1024 words

[1] The implemented pad type for CLKOUT supports only an open-drain output driver

Note: Register SYSCON cannot be changed after execution of the EINIT instruction.

Note: Only exception: bit VISIBLE can also be changed by the on-chip debug support with DPEC access.

System Clock Output Enable (CLKEN)

The system clock output function is enabled by setting bit CLKEN in register SYSCON to '1'. If enabled, port pin P3.15 takes on its alternate function as CLKOUT output pin. The clock output is a 50 % duty cycle clock whose frequency equals the CPU operating frequency ($f_{OUT} = f_{CPU}$).

Note: The output driver of port pin P3.15 is switched on automatically, when the CLKOUT function is enabled. The port direction bit is disgarded.
After reset, the clock output function is disabled (CLKEN = ‘0’).

Note: The implemented pad type for CLKOUT supports only an open-drain output driver

Segmentation Disable/Enable Control (SGTDIS)

Bit SGTDIS allows to select either the segmented or non-segmented memory mode.
In non-segmented memory mode (SGTDIS='1') it is assumed that the code address space is restricted to 64 KBytes (segment 0) and thus 16 bits are sufficient to represent all code addresses. For implicit stack operations (CALL or RET) the CSP register is totally ignored and only the IP is saved to and restored from the stack.
In segmented memory mode (SGTDIS='0') it is assumed that the whole address space is available for instructions. For implicit stack operations (CALL or RET) the CSP register and the IP are saved to and restored from the stack. After reset the segmented memory mode is selected.

Note: Bit SGTDIS controls if the CSP register is pushed onto the system stack in addition to the IP register before an interrupt service routine is entered, and it is repopped when the interrupt service routine is left again.

System Stack Size (STKSZ)

This bitfield defines the size of the physical system stack, which is located in the internal RAM of the INCA-D. An area of 32...1024 words may be dedicated to the system stack. A so-called “circular stack” mechanism allows to use a bigger virtual stack than this dedicated RAM area.

The Processor Status Word PSW

This bit-addressable register reflects the current state of the microcontroller. Two groups of bits represent the current ALU status, and the current CPU interrupt status. A separate bit (USR0) within register PSW is provided as a general purpose user flag.

PSW (FF10$_H$ / 88$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ILVL				IEN	0	0	0	0	0	MULIP	E	Z	V	C	N
rw				rw	r	r	r	r	r	rw	rw	rw	rw	rw	rw

Bit	Function
N	**Negative Result** Set, when the result of an ALU operation is negative.
C	**Carry Flag** Set, when the result of an ALU operation produces a carry bit.
V	**Overflow Result** Set, when the result of an ALU operation produces an overflow.
Z	**Zero Flag** Set, when the result of an ALU operation is zero.
E	**End of Table Flag** Set, when the source operand of an instruction is 8000$_H$ or 80$_H$.
MULIP	**Multiplication/Division In Progress** ‘0’: There is no multiplication/division in progress. ‘1’: A multiplication/division has been interrupted.
ILVL, IEN	**Interrupt and EBC Control Fields** Define the response to interrupt requests and enable external bus arbitration. (Described in section “Interrupt and Trap Functions”)

.

ALU Status (N, C, V, Z, E, MULIP)

The condition flags (N, C, V, Z, E) within the PSW indicate the ALU status due to the last recently performed ALU operation. They are set by most of the instructions due to specific rules, which depend on the ALU or data movement operation performed by an instruction.

After execution of an instruction which explicitly updates the PSW register, the condition flags cannot be interpreted as described in the following, because any explicit write to the PSW register supersedes the condition flag values, which are implicitly generated by the CPU. Explicitly reading the PSW register supplies a read value which represents the state of the PSW register after execution of the immediately preceding instruction.

Note: After reset, all of the ALU status bits are cleared.

• **N-Flag:** For most of the ALU operations, the N-flag is set to '1', if the most significant bit of the result contains a '1', otherwise it is cleared. In the case of integer operations the N-flag can be interpreted as the sign bit of the result (negative: N=’1’, positive: N=’0’). Negative numbers are always represented as the 2's complement of the corresponding positive number. The range of signed numbers extends from '–8000$_H$' to '+7FFF$_H$' for the word data type, or from '–80$_H$' to '+7F$_H$' for the byte data type.For Boolean bit operations with only one operand the N-flag represents the previous state of the specified bit. For Boolean bit operations with two operands the N-flag represents the logical XORing of the two specified bits.

• **C-Flag:** After an addition the C-flag indicates that a carry from the most significant bit of the specified word or byte data type has been generated. After a subtraction or a comparison the C-flag indicates a borrow, which represents the logical negation of a carry for the addition.
This means that the C-flag is set to '1', if **no** carry from the most significant bit of the specified word or byte data type has been generated during a subtraction, which is performed internally by the ALU as a 2's complement addition, and the C-flag is cleared when this complement addition caused a carry.
The C-flag is always cleared for logical, multiply and divide ALU operations, because these operations cannot cause a carry anyhow.
For shift and rotate operations the C-flag represents the value of the bit shifted out last. If a shift count of zero is specified, the C-flag will be cleared. The C-flag is also cleared for a prioritize ALU operation, because a '1' is never shifted out of the MSB during the normalization of an operand.
For Boolean bit operations with only one operand the C-flag is always cleared. For Boolean bit operations with two operands the C-flag represents the logical ANDing of the two specified bits.

• **V-Flag:** For addition, subtraction and 2's complementation the V-flag is always set to '1', if the result overflows the maximum range of signed numbers, which are representable by either 16 bits for word operations ('–8000$_H$' to '+7FFF$_H$'), or by 8 bits for byte operations ('–80$_H$' to '+7F$_H$'), otherwise the V-flag is cleared. Note that the result of an integer addition, integer subtraction, or 2's complement is not valid, if the V-flag indicates an arithmetic overflow.
For multiplication and division the V-flag is set to '1', if the result cannot be represented in a word data type, otherwise it is cleared. Note that a division by zero will always cause an overflow. In contrast to the result of a division, the result of a multiplication is valid regardless of whether the V-flag is set to '1' or not.
Since logical ALU operations cannot produce an invalid result, the V-flag is cleared by these operations.

The V-flag is also used as 'Sticky Bit' for rotate right and shift right operations. With only using the C-flag, a rounding error caused by a shift right operation can be estimated up to a quantity of one half of the LSB of the result. In conjunction with the V-flag, the C-flag allows evaluating the rounding error with a finer resolution (see table below).
For Boolean bit operations with only one operand the V-flag is always cleared. For

Boolean bit operations with two operands the V-flag represents the logical ORing of the two specified bits.

Table 6-1 Shift Right Rounding Error Evaluation

C-Flag	V-Flag	Rounding Error Quantity		
0	0	-	No rounding error	-
0	1	0 <	Rounding error	< $^1/_2$ LSB
1	0		Rounding error	= $^1/_2$ LSB
1	1		Rounding error	> $^1/_2$ LSB

• **Z-Flag:** The Z-flag is normally set to '1', if the result of an ALU operation equals zero, otherwise it is cleared.
For the addition and subtraction with carry the Z-flag is only set to '1', if the Z-flag already contains a '1' and the result of the current ALU operation additionally equals zero. This mechanism is provided for the support of multiple precision calculations.
For Boolean bit operations with only one operand the Z-flag represents the logical negation of the previous state of the specified bit. For Boolean bit operations with two operands the Z-flag represents the logical NORing of the two specified bits. For the prioritize ALU operation the Z-flag indicates, if the second operand was zero or not.

• **E-Flag:** The E-flag can be altered by instructions, which perform ALU or data movement operations. The E-flag is cleared by those instructions which cannot be reasonably used for table search operations. In all other cases the E-flag is set depending on the value of the source operand to signify whether the end of a search table is reached or not. If the value of the source operand of an instruction equals the lowest negative number, which is representable by the data format of the corresponding instruction ('8000_H' for the word data type, or '80_H' for the byte data type), the E-flag is set to '1', otherwise it is cleared.

• **MULIP-Flag:** The MULIP-flag will be set to '1' by hardware upon the entrance into an interrupt service routine, when a multiply or divide ALU operation was interrupted before completion. Depending on the state of the MULIP bit, the hardware decides whether a multiplication or division must be continued or not after the end of an interrupt service. The MULIP bit is overwritten with the contents of the stacked MULIP-flag when the return-from-interrupt-instruction (RETI) is executed. This normally means that the MULIP-flag is cleared again after that.

Note: The MULIP flag is a part of the task environment! When the interrupting service routine does not return to the interrupted multiply/divide instruction (ie. in case of a task scheduler that switches between independent tasks), the MULIP flag must be saved as part of the task environment and must be updated accordingly for the new task before this task is entered.

CPU Interrupt Status (IEN, ILVL)

The Interrupt Enable bit allows to globally enable (IEN=’1’) or disable (IEN=’0’) interrupts. The four-bit Interrupt Level field (ILVL) specifies the priority of the current CPU activity. The interrupt level is updated by hardware upon entry into an interrupt service routine, but it can also be modified via software to prevent other interrupts from being acknowledged. In case an interrupt level '15' has been assigned to the CPU, it has the highest possible priority, and thus the current CPU operation cannot be interrupted except by hardware traps or external non-maskable interrupts. For details please refer to chapter “Interrupt and Trap Functions”.

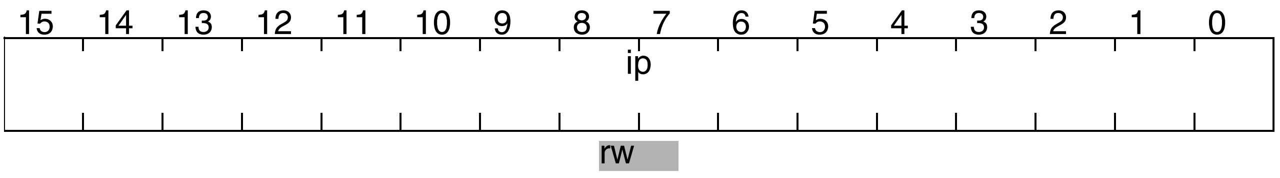
After reset all interrupts are globally disabled, and the lowest priority (ILVL=0) is assigned to the initial CPU activity.

The Instruction Pointer IP

This register determines the 16-bit intra-segment address of the currently fetched instruction within the code segment selected by the CSP register. The IP register is not mapped into the INCA-D's address space, and thus it is not directly accessable by the programmer. The IP can, however, be modified indirectly via the stack by means of a return instruction.

The IP register is implicitly updated by the CPU for branch instructions and after instruction fetch operations.

IP (---- / --) --- **Reset Value: 0000_H**



Bit	**Function**
ip	Specifies the intra segment offset, from where the current instruction is to be fetched. IP refers to the current segment <SEGNR>.

The Code Segment Pointer CSP

This non-bit addressable register selects the code segment being used at run-time to access instructions. The lower 8 bits of register CSP select one of up to 256 segments of 64 KBytes each, while the upper 8 bits are reserved for future use.

CSP (FE08$_H$ / 04$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								SEGNR							
-	-	-	-	-	-	-	-	r							

Bit	Function
SEGNR	**Segment Number** Specifies the code segment, from where the current instruction is to be fetched. SEGNR is ignored, when segmentation is disabled.

Code memory addresses are generated by directly extending the 16-bit contents of the IP register by the contents of the CSP register as shown in the figure below.

In the case of the segmented memory mode, the selected number of segment address bits (via bitfield SALSEL) of register CSP is output on the respective segment address pins of Port 4 for all external code accesses. For non-segmented memory mode, the content of this register is not significant, because all code acccesses are automatically restricted to segment 0.

Note: The CSP register can only be read but not written by data operations. It is, however, modified either directly by means of the JMPS and CALLS instructions, or indirectly via the stack by means of the RETS and RETI instructions.
Upon the acceptance of an interrupt or the execution of a software TRAP instruction, the CSP register is automatically set to zero.

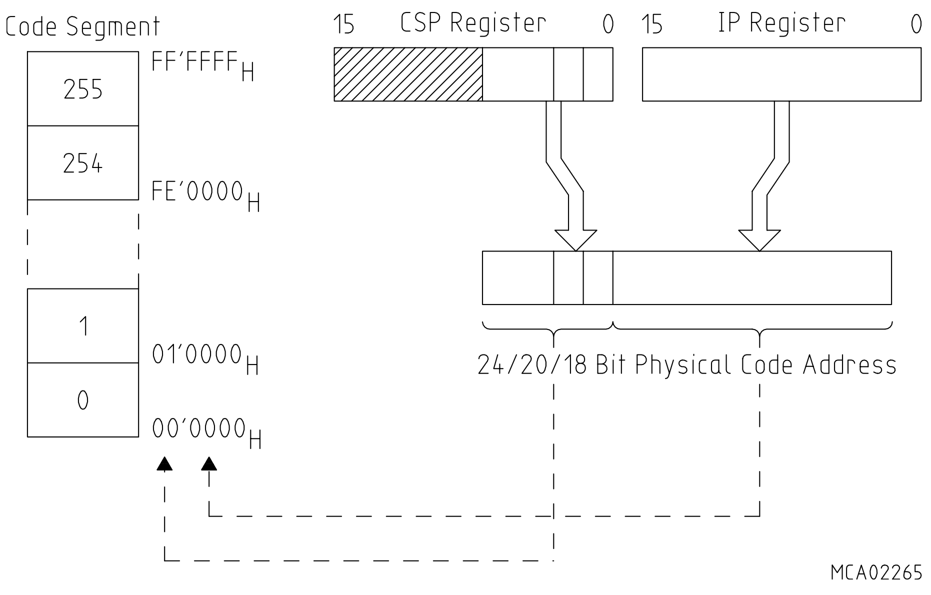


Figure 6-3 Addressing via the Code Segment Pointer

Note: When segmentation is disabled, the IP value is used directly as the 16-bit address.

The Data Page Pointers DPP0, DPP1, DPP2, DPP3

These four non-bit addressable registers select up to four different data pages being active simultaneously at run-time. The lower 10 bits of each DPP register select one of the 1024 possible 16-Kbyte data pages while the upper 6 bits are reserved for future use. The DPP registers allow to access the entire memory space in pages of 16 Kbytes each.

The DPP registers are implicitly used, whenever data accesses to any memory location are made via indirect or direct long 16-bit addressing modes (except for override accesses via EXTended instructions and PEC data transfers). After reset, the Data Page Pointers are initialized in a way that all indirect or direct long 16-bit addresses result in identical 18-bit addresses. This allows to access data pages 3...0 within segment 0 as shown in the figure below. If the user does not want to use any data paging, no further action is required.

DPP0 (FE00$_H$ / 00$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						DPP0PN									
-	-	-	-	-	-	rw									

DPP1 (FE02$_H$ / 01$_H$) SFR **Reset Value: 0001$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						DPP1PN									
-	-	-	-	-	-	rw									

DPP2 (FE04$_H$ / 02$_H$) SFR **Reset Value: 0002$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						DPP2PN									
-	-	-	-	-	-	rw									

DPP3 (FE06$_H$ / 03$_H$) **SFR** **Reset Value: 0003$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						DPP3PN									
-	-	-	-	-	-	rw									

Bit	**Function**
DPPxPN	**Data Page Number of DPPx** Specifies the data page selected via DPPx. Only the least significant two bits of DPPx are significant, when segmentation is disabled.

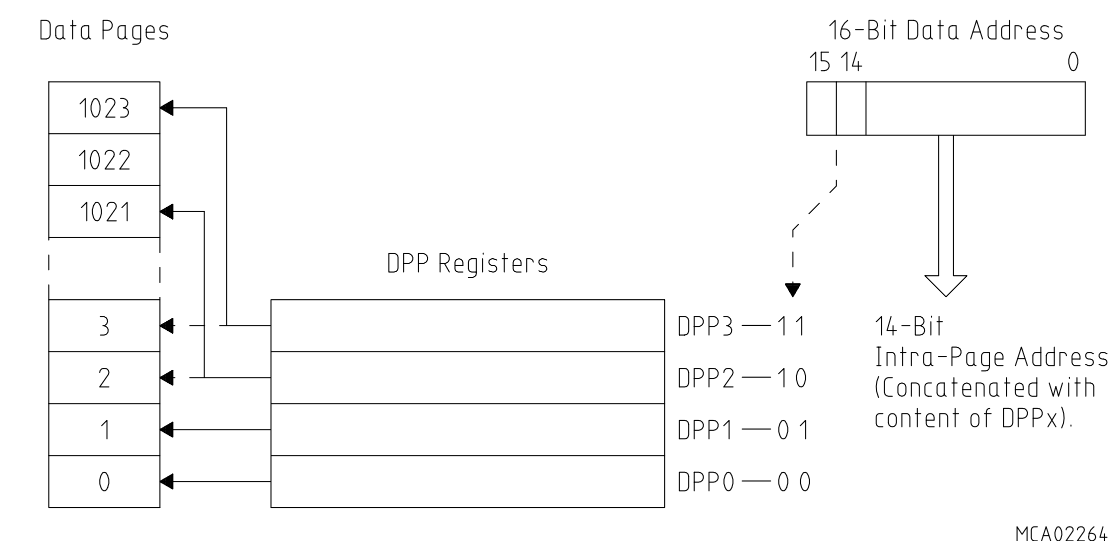
Data paging is performed by concatenating the lower 14 bits of an indirect or direct long 16-bit address with the contents of the DPP register selected by the upper two bits of the 16-bit address. The content of the selected DPP register specifies one of the 1024 possible data pages. This data page base address together with the 14-bit page offset forms the physical 24-bit address (selectable part is driven to the address pins).

In case of non-segmented memory mode, only the two least significant bits of the implicitly selected DPP register are used to generate the physical address. Thus, extreme care should be taken when changing the content of a DPP register, if a non-segmented memory model is selected, because otherwise unexpected results could occur.

In case of the segmented memory mode the selected number of segment address bits (via bitfield SALSEL) of the respective DPP register is output on the respective segment address pins of Port 4 for all external data accesses.

A DPP register can be updated via any instruction, which is capable of modifying an SFR.

Note: Due to the internal instruction pipeline, a new DPP value is not yet usable for the operand address calculation of the instruction immediately following the instruction updating the DPP register.



After reset or with segmentation disabled the DPP registers select data pages 3...0. All of the internal memory is accessible in these cases.

Figure 6-4 Addressing via the Data Page Pointers

The Context Pointer CP

This non-bit addressable register is used to select the current register context. This means that the CP register value determines the address of the first General Purpose Register (GPR) within the current register bank of up to 16 wordwide and/or bytewide GPRs.

CP (FE10$_H$ / 08$_H$) SFR **Reset Value: FC00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1						cp						0
r	r	r	r						rw						r

Bit	Function
cp	**Modifiable portion of register CP** Specifies the (word) base address of the current register bank. When writing a value to register CP with bits CP.11...CP.9 = ‘000’, bits CP.11...CP.10 are set to ‘11’ by hardware, in all other cases all bits of bit field “cp” receive the written value.

Note: It is the user's responsibility that the physical GPR address specified via CP register plus short GPR address must always be an internal RAM location. If this condition is not met, unexpected results may occur.
- *Do not set CP below the IRAM start address, ie. 00’F600$_H$*
- *Do not set CP above 00’FDFE$_H$*
- *Be careful using the upper GPRs with CP above 00’FDE0$_H$*

The CP register can be updated via any instruction which is capable of modifying an SFR.

Note: Due to the internal instruction pipeline, a new CP value is not yet usable for GPR address calculations of the instruction immediately following the instruction updating the CP register.

The Switch Context instruction (SCXT) allows to save the content of register CP on the stack and updating it with a new value in just one machine cycle.

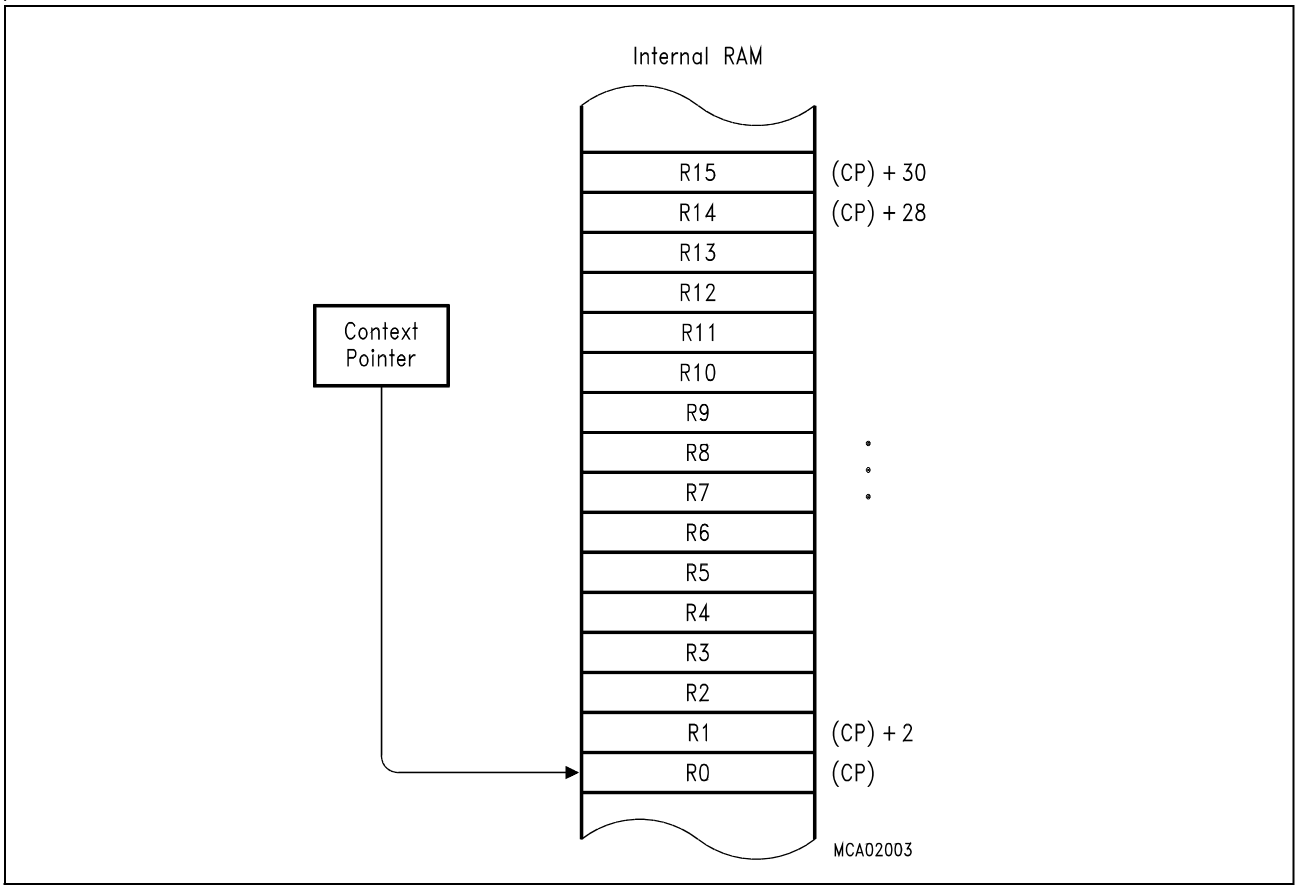


Figure 6-5 Register Bank Selection via Register CP

Several addressing modes use register CP implicitly for address calculations. The addressing modes mentioned below are described in chapter “Instruction Set Summary”.

Short 4-Bit GPR Addresses (mnemonic: Rw or Rb) specify an address relative to the memory location specified by the contents of the CP register, ie. the base of the current register bank.
Depending on whether a relative word (Rw) or byte (Rb) GPR address is specified, the short 4-bit GPR address is either multiplied by two or not before it is added to the content of register CP (see figure below). Thus, both byte and word GPR accesses are possible in this way.

GPRs used as indirect address pointers are always accessed wordwise. For some instructions only the first four GPRs can be used as indirect address pointers. These GPRs are specified via short 2-bit GPR addresses. The respective physical address calculation is identical to that for the short 4-bit GPR addresses.

Short 8-Bit Register Addresses (mnemonic: reg or bitoff) within a range from $F0_H$ to FF_H interpret the four least significant bits as short 4-bit GPR address, while the four most significant bits are ignored. The respective physical GPR address calculation is identical to that for the short 4-bit GPR addresses.

For single bit accesses on a GPR, the GPR's word address is calculated as just described, but the position of the bit within the word is specified by a separate additional 4-bit value.

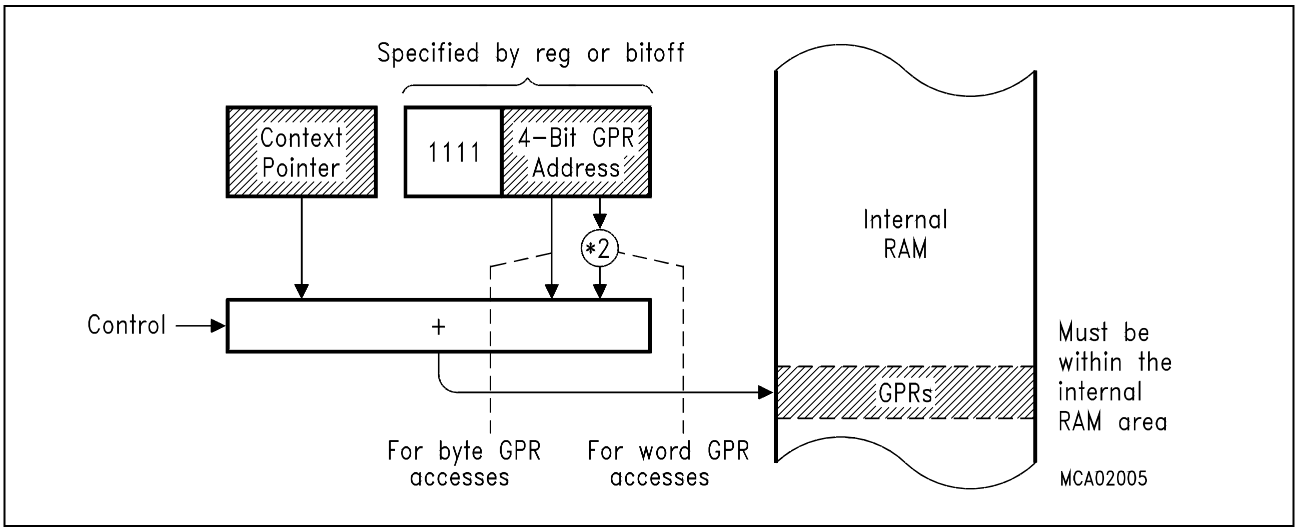


Figure 6-6 Implicit CP Use by Short GPR Addressing Modes

The Stack Pointer SP

This non-bit addressable register is used to point to the top of the internal system stack (TOS). The SP register is pre-decremented whenever data is to be pushed onto the stack, and it is post-incremented whenever data is to be popped from the stack. Thus, the system stack grows from higher toward lower memory locations.

Since the least significant bit of register SP is tied to '0' and bits 15 through 12 are tied to '1' by hardware, the SP register can only contain values from F000$_H$ to FFFE$_H$. This allows to access a physical stack within the internal RAM of the INCA-D. A virtual stack (usually bigger) can be realized via software. This mechanism is supported by registers STKOV and STKUN (see respective descriptions below).

The SP register can be updated via any instruction, which is capable of modifying an SFR.

Note: Due to the internal instruction pipeline, a POP or RETURN instruction must not immediately follow an instruction updating the SP register.

SP (FE12$_H$ / 09$_H$) SFR **Reset Value: FC00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	sp											0
r	r	r	r	rw											r

Bit	Function
sp	**Modifiable portion of register SP** Specifies the top of the internal system stack.

The Stack Overflow Pointer STKOV

This non-bit addressable register is compared against the SP register after each operation, which pushes data onto the system stack (eg. PUSH and CALL instructions or interrupts) and after each subtraction from the SP register. If the content of the SP register is less than the content of the STKOV register, a stack overflow hardware trap will occur.

Since the least significant bit of register STKOV is tied to '0' and bits 15 through 12 are tied to '1' by hardware, the STKOV register can only contain values from F000$_H$ to FFFE$_H$.

STKOV (FE14$_H$ / 0A$_H$) SFR **Reset Value: FA00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	stkov											0
r	r	r	r	rw											r

Bit	Function
stkov	**Modifiable portion of register STKOV** Specifies the lower limit of the internal system stack.

The Stack Overflow Trap (entered when (SP) < (STKOV)) may be used in two different ways:

- **Fatal error indication** treats the stack overflow as a system error through the associated trap service routine. Under these circumstances data in the bottom of the stack may have been overwritten by the status information stacked upon servicing the stack overflow trap.
- **Automatic system stack flushing** allows to use the system stack as a 'Stack Cache' for a bigger external user stack. In this case register STKOV should be initialized to a value, which represents the desired lowest Top of Stack address plus 12 according to the selected maximum stack size. This considers the worst case that will occur, when a stack overflow condition is detected just during entry into an interrupt service routine. Then, six additional stack word locations are required to push IP, PSW, and CSP for both the interrupt service routine and the hardware trap service routine.

More details about the stack overflow trap service routine and virtual stack management are given in chapter "System Programming".

The Stack Underflow Pointer STKUN

This non-bit addressable register is compared against the SP register after each operation, which pops data from the system stack (eg. POP and RET instructions) and after each addition to the SP register. If the content of the SP register is greater than the the content of the STKUN register, a stack underflow hardware trap will occur.

Since the least significant bit of register STKUN is tied to '0' and bits 15 through 12 are tied to '1' by hardware, the STKUN register can only contain values from F000$_H$ to FFFE$_H$.

STKUN (FE16$_H$ / 0B$_H$) SFR **Reset Value: FC00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	stkun											0
r	r	r	r	rw											r

Bit	Function
stkun	**Modifiable portion of register STKUN** Specifies the upper limit of the internal system stack.

The Stack Underflow Trap (entered when (SP) > (STKUN)) may be used in two different ways:

• **Fatal error indication** treats the stack underflow as a system error through the associated trap service routine.

• **Automatic system stack refilling** allows to use the system stack as a 'Stack Cache' for a bigger external user stack. In this case register STKUN should be initialized to a value, which represents the desired highest Bottom of Stack address.

More details about the stack underflow trap service routine and virtual stack management are given in chapter “System Programming”.

Scope of Stack Limit Control

The stack limit control realized by the register pair STKOV and STKUN detects cases where the stack pointer SP is moved outside the defined stack area either by ADD or SUB instructions or by PUSH or POP operations (explicit or implicit, ie. CALL or RET instructions).

This control mechanism is not triggered, ie. no stack trap is generated, when

• the stack pointer SP is directly updated via MOV instructions
• the limits of the stack area (STKOV, STKUN) are changed, so that SP is outside of the new limits.

The Multiply/Divide High Register MDH

This register is a part of the 32-bit multiply/divide register, which is implicitly used by the CPU, when it performs a multiplication or a division. After a multiplication, this non-bit addressable register represents the high order 16 bits of the 32-bit result. For long divisions, the MDH register must be loaded with the high order 16 bits of the 32-bit dividend before the division is started. After any division, register MDH represents the 16-bit remainder.

MDH (FE0C$_H$ / 06$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								mdh							
								rw							

Bit	Function
mdh	Specifies the high order 16 bits of the 32-bit multiply and divide register MD.

Whenever this register is updated via software, the Multiply/Divide Register In Use (MDRIU) flag in the Multiply/Divide Control register (MDC) is set to '1'.

When a multiplication or division is interrupted before its completion and when a new multiply or divide operation is to be performed within the interrupt service routine, register MDH must be saved along with registers MDL and MDC to avoid erroneous results.

A detailed description of how to use the MDH register for programming multiply and divide algorithms can be found in chapter "System Programming".

The Multiply/Divide Low Register MDL

This register is a part of the 32-bit multiply/divide register, which is implicitly used by the CPU, when it performs a multiplication or a division. After a multiplication, this non-bit addressable register represents the low order 16 bits of the 32-bit result. For long divisions, the MDL register must be loaded with the low order 16 bits of the 32-bit dividend before the division is started. After any division, register MDL represents the 16-bit quotient.

MDL (FE0E$_H$ / 07$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								mdl							
								rw							

Bit	Function
mdl	Specifies the low order 16 bits of the 32-bit multiply and divide register MD.

Whenever this register is updated via software, the Multiply/Divide Register In Use (MDRIU) flag in the Multiply/Divide Control register (MDC) is set to '1'. The MDRIU flag is cleared, whenever the MDL register is read via software.

When a multiplication or division is interrupted before its completion and when a new multiply or divide operation is to be performed within the interrupt service routine, register MDL must be saved along with registers MDH and MDC to avoid erroneous results.

A detailed description of how to use the MDL register for programming multiply and divide algorithms can be found in chapter “System Programming”.

The Multiply/Divide Control Register MDC

This bit addressable 16-bit register is implicitly used by the CPU, when it performs a multiplication or a division. It is used to store the required control information for the corresponding multiply or divide operation. Register MDC is updated by hardware during each single cycle of a multiply or divide instruction.

MDC (FF0E$_H$ / 87$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								!!	!!	!!	MDRIU	!!	!!	!!	!!
-	-	-	-	-	-	-	-	rw)	rw)	rw)	rw)	rw)	rw)	rw)	rw)

Bit	Function
MDRIU	**Multiply/Divide Register In Use** ‘0’: Cleared, when register MDL is read via software. ‘1’: Set when register MDL or MDH is written via software, or when a multiply or divide instruction is executed.
!!	**Internal Machine Status** The multiply/divide unit uses these bits to control internal operations. Never modify these bits without saving and restoring register MDC.

When a division or multiplication was interrupted before its completion and the multiply/divide unit is required, the MDC register must first be saved along with registers MDH and MDL (to be able to restart the interrupted operation later), and then it must be cleared prepare it for the new calculation. After completion of the new division or multiplication, the state of the interrupted multiply or divide operation must be restored.

The MDRIU flag is the only portion of the MDC register which might be of interest for the user. The remaining portions of the MDC register are reserved for dedicated use by the hardware, and should never be modified by the user in another way than described above. Otherwise, a correct continuation of an interrupted multiply or divide operation cannot be guaranteed.

A detailed description of how to use the MDC register for programming multiply and divide algorithms can be found in chapter “System Programming”.

The Constant Zeros Register ZEROS

All bits of this bit-addressable register are fixed to '0' by hardware. This register can be read only. Register ZEROS can be used as a register-addressable constant of all zeros, ie. for bit manipulation or mask generation. It can be accessed via any instruction, which is capable of addressing an SFR.

ZEROS (FF1C$_H$ / 8E$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

The Constant Ones Register ONES

All bits of this bit-addressable register are fixed to '1' by hardware. This register can be read only. Register ONES can be used as a register-addressable constant of all ones, ie. for bit manipulation or mask generation. It can be accessed via any instruction, which is capable of addressing an SFR.

ONES (FF1E$_H$ / 8F$_H$) SFR **Reset Value: FFFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

6.3 PEC - Extension of Functionality

Introduction

Compared to existing C16x architecture, the PEC transfer function is enhanced by extended functionality. For information regarding the general PEC transfer refer to the interrupt description.

However, the extended PEC function is a further step into DMA control functionality. It especially supports integrated system design with XBUS as system bus.

The extended PEC functions are defined as follows:

– Source pointer and destination pointer are extended to 24-bit pointer, thus enabling PEC controlled data transfer between any two locations within the total address space. Both 8-bit segment numbers of every source/destination pointer pair are defined in one 16-bit SFR register; thus, 8 PEC segment number registers are available for the 8 PEC channels.
– Two of the PEC channels are expanded by additional 16-bit transfer count registers; when enabled, the original 8-bit bytecount in the control register serves as package length count, thus defining the amount of bytes or words to be transferred with one request. In INCA-D the package size is always limited to one transfer.
– For always two channels a chaining feature is provided. When enabled in the PEC control register, a termination interrupt of one channel will automatically switch transfer control to the other channel of the channel pair.

24-bit Extension of Source and Destination Pointers

The source and destination pointers specify the locations between which the data is to be moved. For each of the eight PEC channels the source and destination pointers are specified by one SFR register and two IRAM memory locations. One SFR register stores the 8-bit segment number of the source (PECSSN) and the 8-bit segment number of the destination (PECDSN) location in a respective 16-bit PEC Segment Number register (PECSNx). The respective segment offset of source and destination are stored in IRAM memory location identical to the IRAM locations of SRCPx and DSTPx pointers of Full-Custom C16x standard PEC channels - thus the extension is fully compatible. With the segment number extension of source and destination, data can be transferred by a PEC transfer between any two locations within the 16 MByte address space of the INCA-D.

Note: The segment number extension of source and destination is provided for all 8 PEC channels. After reset, all 8 segment number registers PECSNx are cleared, providing full compatibility to FC-C16x PEC channels.

The PEC segment number registers PECSNx are defined as follows:

PECSNx (Addresses see table) **SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PECDSN								PECSSN							
rw								rw							

Bit	Function
PECSSN	**PEC Source Segment Number** 8-bit Segment Number (address bits A23-A16) used for addressing the source of the respective PEC transfer.
PECDSN	**PEC Destination Segment Number** 8-bit Segment Number (address bits A23-A16) used for addressing the destination of the respective PEC transfer.

Table 6-2 PEC Segment Number Register Addresses

Register	Address	Reg. Space	Register	Address	Reg. Space
PECSN0	FED0$_H$ / 68$_H$	SFR	PECSN4	FED8$_H$ / 6C$_H$	SFR
PECSN1	FED2$_H$ / 69$_H$	SFR	PECSN5	FEDA$_H$ / 6D$_H$	SFR
PECSN2	FED4$_H$ / 6A$_H$	SFR	PECSN6	FEDC$_H$ / 6E$_H$	SFR
PECSN3	FED6$_H$ / 6B$_H$	SFR	PECSN7	FEDE$_H$ / 6F$_H$	SFR

Extended PEC Channel Control

The PEC control registers with the extended functionality and their application for new PEC control are defined as follows:

PECCx (Addresses: see table) **SFR** **Reset Value: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PT	**-**	**-**	**CLT**	**CL**	**INC**		**BWT**	**COUNT**							
rw	-	-	rw	rw	rw		rw	rw							

PECXC0/1 (Addresses: see table) **SFR** **Reset Value: 0000_H**

15	14											3	2	1	0
COUNT2															
rw															

Bit	Function
COUNT	**PEC Transfer Count** Counts PEC transfers (bytes or words) and influences the channel's action
BWT	**Byte / Word Transfer Selection** 0: Transfer a Word 1: Transfer a Byte
INC	**Increment Control** (Modification of SRCPx or DSTPx) 0 0: Pointers are not modified 0 1: Increment DSTPx by 1 or 2 (BWT) 1 0: Increment SRCPx by 1 or 2 (BWT) 1 1: Reserved. Do not use this combination. (changed to 10 by hardware)
CL	**Channel Link Control** 0: PEC channels work independent 1: Pairs of channels are linked together
CLT	**Channel Link Toggle State** 0: Even numbered PEC channel of linked channels active 1: Odd numbered PEC channel of linked channels active
PT	**Package Transfer** 0: Single Transfer; extended Count2 not enabled 1: Package Transfer; extended Count2 enabled (only for channels 1 and 2) **Note :** Package Transfer is only supported in PECC2 and PECC0

Bit	Function
COUNT2	**PEC Extended Transfer Count** PEC transfer count extension (see table below)

Table 6-3 PEC Control Register Addresses

Register	Address	Reg. Space	Register	Address	Reg. Space
PECC0	FEC0$_H$ / 60$_H$	SFR	PECC4	FEC8$_H$ / 64$_H$	SFR
PECC1	FEC2$_H$ / 61$_H$	SFR	PECC5	FECA$_H$ / 65$_H$	SFR
PECC2	FEC4$_H$ / 62$_H$	SFR	PECC6	FECC$_H$ / 66$_H$	SFR
PECC3	FEC6$_H$ / 63$_H$	SFR	PECC7	FECE$_H$ / 67$_H$	SFR
PECXC0	FEF0$_H$/78$_H$	SFR	PECXC1	FEF2$_H$/79$_H$	SFR

Long Transfer Count with Package Transfer

The features of the C16x PEC functions are expanded by long transfer counts and package transfers, where a package is a container with always the same defined number of bytes or words. If enabled by the PT flag in PECCx, each service request initiates an entire package of data to be transferred by the PEC channel. The length of data packages is determined by the PEC Transfer COUNT in the PECCx register, the amount of package transfers is defined by the long 16-bit transfer count COUNT2 in the new PECXCx register. In PT mode, packages of up to 256 bytes or words are supported.

The PEC transfer COUNT2 allows to service up to 64K transfer requests by the respective PEC channel, and then (when COUNT2 reaches 00H) activate the interrupt service routine in the CPU as known from standard PEC channels. After each PEC package transfer the COUNT2 field is decremented by one and the request flag is cleared to indicate that the request has been serviced. Note: Within a package transfer only every second machine cycle is a PEC transfer cycle.

Note: In INCA-D, the package length is limited to 1 transfer (COUNT field must be 01$_H$). Therefore only the long transfer count COUNT2 which enables block lengths of up to 64K transfers, is available in Package Transfer mode (the PT flag is set in PECCx register). Package transfers with COUNT>1 are not supported.

Note: The PT mode is available for two PEC channels: PECC2 and PECC0. These two channels with long transfer counts can also be used in Channel Link Mode (see below), to be concatenated with channel 3 and channel 1.

Channel Link Mode for Data Chaining

Data chaining with linked PEC channels is enabled, if the Channel Link Control Bit in PECCx register is set to '1', either in one or both PEC channel control registers of a channel pair. In this case, two PEC channels are linked together and handle chained block transfers alternatively to each other. The whole data transfer is divided into several block transfers where each block is controlled by one PEC channel of a channel pair. When a data block is completely transferred a **channel link interrupt** is generated and the PEC service request processing is automatically switched to the 'other' PEC channel of the channel-pair. Thus, PEC service requests addressed to a linked PEC channel are either handled by linked PEC channel A or by linked PEC channel B. This channel toggle allows to set up shadow and multiple buffers for PEC transfers by changing pointer and count values of one channel while the other channel is active. The following table list the channels that can be linked together and the channel numbers to address the linked channels.

Table 6-4 PEC Channels which could be linked together

Linked PEC Channels		Linked PEC Channel
PEC Channel A	**PEC Channel B**	
channel 0	channel 1	channel 0
channel 2	channel 3	channel 2
channel 4	channel 5	channel 4
channel 6	channel 7	channel 6

For each pair of linked channels, an internal channel flag, the Channel Link Toggle flag CLT identifies which of the two PEC channels will serve the next PEC request. The CLT flag is indicated in both PECCx registers of two linked PEC channels, where the CLT bit in channel B always is inverse to the CLT bit in channel A. The very first transfer is always started with the channel A if the CLT bit was not programmed otherwise before. The CLT bit is only valid in case of linked PEC channels, indicated by the CL bits of linked channels. If linking is not enabled, the CLT bit of both channels is always zero (compatibility!).

The internal channel link flag CLT toggles, and the other channel begins service with the next request if the "old" channel stops the service (COUNT=0 or COUNT2=0, dependent on the mode), and if the new channel has in its PEC control register the CL flag enabled and its transfer count is more than zero. Note: With the last transfer of a block transfer (COUNT=0 or COUNT2=0), the channel link control flag CL of that channel is cleared in its PECCx register. If the channel link flag CL of the new (chained) PEC control register is found to be zero the whole data transfer is finished and the channel link interrupt is coincidently a termination interrupt.

The channel link mode is finished and the internal channel toggle flag is cleared after the last transfer of the block, if the CL flags of both pair channels are cleared.

Additional Interrupt Request Node for Channel Link Interrupts

The PEC Unit has one dedicated service request node (trap number) for all channel link interrupts. This service request node requests CPU interrupt service in case of one or more channel link request flag and the respective enable control bit is set in the channel link interrupt subnode control register (CLISNC). These flags indicate a channel link interrupt condition of linked PEC channels (A and B channels) which requires support by the CPU. The following channel link interrupt conditions requesting CPU service are possible:

– In single transfer mode a COUNT value change from 01_H to 00_H in a linked PEC channel and CL flag is set in the respective PEC control register.
– In package transfer mode a COUNT2 value change from 0001_H to 0000_H in a linked PEC channel and CL flag is set in the belonging PEC control register.

In these cases the CPU service is requested to update the PEC control and pointer registers while the next block transfer is executed (the whole transfer is divided into separately controlled block transfers). The last block transfer is determined by the missing link bit in the new (linked) PEC control register. If a new service request hits a linked channel with count equal to zero and channel link flag disabled, a standard interrupt is performed as known from standard PEC channels.

The channel link interrupt subnode register CLISNC is defined as follows:

CLISNC (FFA8$_H$ / D4$_H$) **SFR-b** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	**-**	**C6 IR**	**C6 IE**	**-**	**-**	**C4 IR**	**C4 IE**	**-**	**-**	**C2 IR**	**C2 IE**	**-**	**-**	**C0 IR**	**C0 IE**
-	-	rw	rw	-	-	rw	rw	-	-	rw	rw	-	-	rw	rw

Bit	Function
xxIE	**PEC Channel Link Interrupt Enable Control Bit** (individually enables/disables a specific channel pair interrupt request) ‘0’: PEC interrupt request is disabled ‘1’: PEC interrupt request is enabled
xxIR	**PEC Channel Service Request Flag** ‘0’: No channel link service request pending ‘1’: This source (channel pair) has raised an request to service a PEC channel after channel linking

6.4 XBUS System Architecture

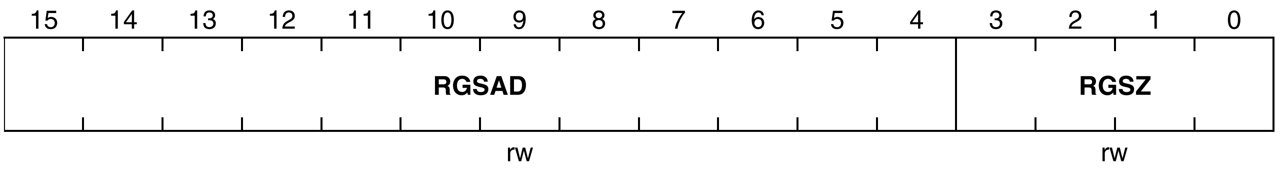
6.4.1 Bus Access Control

CPU accesses to internal and external busses, thus internal or external memories or peripherals are controlled with the respective address ranges. These address ranges are supported for on-chip XBUS resources or for external off-chip resources. In INCA-D **four** address ranges with according bus definitions must be programmed for **on-chip** XBUS peripherals (including memories) .

Address ranges and thus address mappings of peripherals, which communicate over the internal XBUS are controlled by address selection registers XADRSx. The respective bus type definitions are controlled with registers XBCONx.

Generally, the definition for the XADRSx and XBCONx registers is very similar to the defintion of the registers of the External Bus Interface. The XADRSx registers and XBCONx registers are defined as follows:

XADRS1(2)(3)(4) **F014$_H$ (F016$_H$)(F018$_H$) (F01A$_H$)** **ESFR** **Reset Value: 0000$_H$**



Range Size RGSZ	Selected Address Range	Relvant(R) bits of RGSAD	Selected Range Start Address (Relevant(R) bits of RGSAD)
0000	256 Byte	RRRR RRRR RRRR	0000 RRRR RRRR RRRR 0000 0000
0001	512 Bytes	RRRR RRRR RRR0	0000 RRRR RRRR RRR0 0000 0000
0010	1 KBytes	RRRR RRRR RR00	0000 RRRR RRRR RR00 0000 0000
0011	2 KBytes	RRRR RRRR R000	0000 RRRR RRRR R000 0000 0000
0100	4 KBytes	RRRR RRRR 0000	0000 RRRR RRRR 0000 0000 0000
0101	8 KBytes	RRRR RRR0 0000	0000 RRRR RRR0 0000 0000 0000
0110	16 KBytes	RRRR RR00 0000	0000 RRRR RR00 0000 0000 0000
0111	32 KBytes	RRRR R000 0000	0000 RRRR R000 0000 0000 0000
1000	64 KBytes	RRRR 0000 0000	0000 RRRR 0000 0000 0000 0000
1001	128 KBytes	RRR0 0000 0000	0000 RRR0 0000 0000 0000 0000
1010	256 KBytes	RR00 0000 0000	0000 RR00 0000 0000 0000 0000
1011	512 KBytes	R000 0000 0000	0000 R000 0000 0000 0000 0000
11xx	- reserved		

Table 6-5 Address Range and Address Range Start Definition of XADRSx register

Bit	Function
RGSAD	Address Range Start Address Selection
RGSZ	Address Range Size Selection

XBCON1(2)(3)(4) **F114$_H$ (F116$_H$)(F118$_H$)(F11A$_H$)** **ESFR-b** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	RDY ENx	BS WCx	BUS ACTx	ALE CTLx	EW ENx	BTYPx		MT TCx	RW DCx	MCTCx			
-	-	-	rw	rw	rw	rw	rw	rw		rw	rw	rw			

Bit	Function
MCTCx	**Memory Cycle Time Control** 0 0 0 0 : 15 waitstates (Number = 15 - <MCTC>) . . . 1 1 1 1 : No waitstates
RWDCx	**READ/WRITE Delay Control** ‘0’: With read/write delay: activate command 1 TCL after falling edge of ALE ‘1’: No read/write delay: activate command with falling edge of ALE
MTTCx	**Memory Tri-state Time Control** ‘0’: 1 waitstate ‘1’: No waitstate
BTYPx	**Bus Type Selection**; only demultiplexed busses are supported on XBUS; ’00’: 8 bit bus ’10’: 16 bit bus; ’x1’: reserved.
EWENx	**Early Write Enable** ’0’: Standard write enable signal control ’1’: Write active state is disabled one TCL earlier
ALECTLx	**ALE Lengthening Control Bit** (see BUSCON)
BUSACTx	**Bus Active Control** ‘0’: XBUS (peripheral) disabled ‘1’: XBUS (peripheral) enabled Enables the XBUS and the according chip select $\overline{\text{XCSx}}$ for the respective address window (respective XBUS peripheral), selected with according XADRSx window; after reset, all address windows on XBUS are disabled.

Bit	Function
BSWCx	**BUSCON Switch Control** '0': Standard switch of bustype (switch of XBCON) '1': A bus wait state (Tri-state cycle) is included after execution of last old-bustype cycle and before the first new-bustype cycle after switch of XBCON or BUSCON; the BSWC bit is indicated in the old-bustype XBCON/BUSCON.
RDYENx	**READY Enable** '0': The bus cycle length is controlled by the bus controller using MCTC '1': The bus cycle length is controlled by the peripheral using READY

The smallest possible adress range, which is covered with respect to the XADRSx register, is 256 bytes. For a better utilization of that memory , the address ranges are in two cases shared by some XBUS peripherals as described in the table below.

After reset, no address selection register is selected; thus the default address range is enabled and controlled with BUSCON0 and additionally the chip select output $\overline{\text{CS0}}$ is activated (as in standard C16x architecture). In order to use the on-chip XBUS peripherals , the XADRSx and XBCONx registers must be programmed in a certain way. The XBUS peripherals correspond to the register pairs XADRSx and XBCONx as follows:

Table 6-6 XBUS peripherals groups

Group number	corresonding peripherals	registers
1	PIDD, TSF, I2C	XBCON1 & XADRS1
2	IOM handler, HDLC controller, Transceiver, CI Handler	XBCON2 & XADRS2
3	USB	XBCON3 & XADRS3
4	XRAM	XBCON4 & XADRS4

All XADRSx/ADDRSELx registers as well as XBCONx/BUSCONx registers are user programmable SFR registers. All BUSCONx registers are mapped into the bitaddressable SFR memory space, all XBCONx registers are located in the bitaddressable ESFR memory space. Although they are free programmable, programming should be performed during the initialisation phase before the first accesses are controlled with XBCONx or BUSCONx.

Wait states of IOM2 block

Since the IOM2 block contains modules which have wide variations in their uC access times, wait states assignment for the different blocks is dynamic. To implement this dynamic wait state assignment scheme the IOM2 access by the uC is implemented by using synchronous handshaking (via the *READY* signal). The assertion of the READY signal by the IOM2 is based on the values programmed in the IOM2 Wait States Register (**IWSR**). Without wait states a single c166 access takes 2 *XCLK* cycles. Hence with a wait state of 2, the access will last 4 *XCLK* cycles.

Thus the **waitstates for the IOM-2 handler block** are not controlled by the XBCON2 register, but by the dedicated IOM-waitstate register **IWSR**. Please refer to **Chapter 17.6.6.7** for further information.

Wait state for accesses to the USB block

Because the memory management unit of the USB module needs 3 clock cycles for an read access to the setup token memory, but a CPU access to the XBUS takes 2 CPU cycles (= 1 machine cycle), one 1 wait state has to be programmed to read the setup token. (compare XBCON3 programming in **Table 6-7**)

In order to ensure correct access of the on-chip XBUS peripheral groups, the registers XADRSx must be set to the values shown in table 6-7 prior to the use of the corresponding peripherals. See also **Figure 25-1**.

Table 6-7 ADRSx and XBCONx values to be programmed

(E)SFR	Value
XADRS1	0DF0$_H$
XADRS2	0DE0$_H$
XADRS3	0DD0$_H$
XADRS4	0E04$_H$
XBCON1	04BF$_H$
XBCON2	1437$_H$
XBCON3	043F$_H$ 043E$_H$ (for *setup token* read only (for details refer to **Chapter 22**)
XBCON4	04BF$_H$

6.4.2 XBUS Peripheral Configuration Block

Because of compatibility (to C16x) the XBUS peripheral groups can be separately selected for being visible by means of corresponding selection bits in the XPERCON register. If not selected and therefore not enabled (not activated with XPERCON bit), the peripheral's address space including SFR addresses and port pins are not occupied by the peripheral, thus the peripheral is not visible and not available.

XPERCON (F024$_H$ / 12$_H$) **ESFR** **Reset Value : 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											**XPER 4**	**XPER 3**	**XPER 2**	**XPER 1**	***R***

Bit Field	Bits	Type	Value	Description
XPER1	1	rw	0 1	PIDD, IIC and TSF modules are not visible PIDD, IIC and TSF modules are selected and visible
XPER2	1	rw	0 1	IOM handler, CI handler, HDLC controller and line transceiver are not visible IOM handler, CI handler, HDLC controller and line transceiver are selected and visible
XPER3	1	rw	0 1	USB interface is not visible USB interface is selected and visible
XPER4	1	rw	0 1	XRAM is not visible XRAM is selected and visible
XPERx	remaining bits			R = Reserved

To make an XBUS peripheral group visible, its related bit in XPERCON register must be set before the XPERs are globally enabled with XPEN-bit in SYSCON register (during system initialization be fore EINIT instruction). After reset, **no** XBUS peripheral is selected in XPERCON register.

7 Integrated OCDS Support

On Chip Debug Support (OCDS) is implemented to provide the most important hardware emulation features without special emulation chips at minimum cost.

Note: For more detailed information please refer to 'OCDS C166CBC Target Specification V.1.5'

Features of OCDS:
- Hardware, software and external pin breakpoints (BRKIN, BRKOUT)
- Up to 4 instruction pointer breakpoints
- Masked comparisons for hardware breakpoints
- Independent modules for OCDS and debug port (interface to external debugger)
- The OCDS can also be configured by a monitor
- Trace support in conjunction with the debug port
- Support of multi CPU systems
- Single stepping with monitor or CPU halt

Basically, debug operations are controlled with debug events and event actions:
- Debug events:
- Hardware trigger combination
- Execution of a DEBUG instruction
- Break pin input
- Debug event actions:
- Halt the CPU
- Call a monitor
- Trigger a data transfer (DPEC)
- Activate external pin

7.1 Applications of OCDS

The application of OCDS is to debug the user software running on the CPU in the customer's system. This is done with an external debugger, that controls the OCDS via the independent debug port.
An standard (IEEE 1149.1) JTAG interface can be used as independent port to the OCDS within C166CBC systems.
Via the JTAG interface and the integrateds modulle called *Cerberus*, the external debug hardware can access the OCDS registers and arbitrary memory locations with the new DPEC mechanism. DPEC transfers are PEC transfers, executed by the core CPU, but controlled by the Cerberus. With DPEC transfers, injected into the CPU's pipeline or executed in CPU-Hold state, data exchange can be executed by the Cerberus module within the whole address space, including all SFRs and dual-port RAM as well as program or data Flash modules on chip (for example).

Features of Cerberus:
- Generic serial link to address the whole user address space
- External host controls all transactions
- JTAG interface is used as control and data channel
- Generic memory read/write functionality (RW mode)
- Full support for communication between monitor and external debugger (communication mode)
- Optional error protection
- Pending reads (writes) can be optional triggered from the OCDS module (low end tracing)

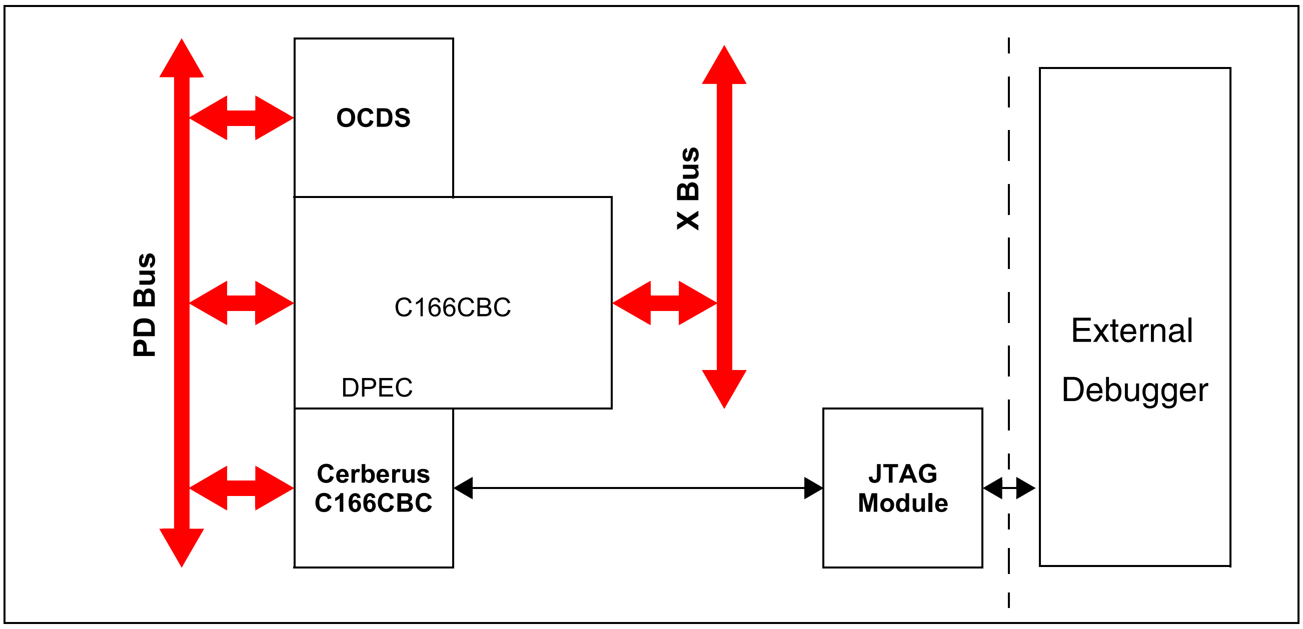


Figure 7-1 Block Diagram of Cerberus/OCDS integration

8 Interrupts

The architecture of the INCA-D supports several mechanisms for fast and flexible response to service requests that can be generated from various sources. These mechanisms include:

Normal Interrupt Processing

The CPU temporarily suspends the current program execution and branches to an interrupt service routine in order to service an interrupt requesting device. The current program status (IP, PSW, in segmentation mode also CSP) is saved on the internal system stack. A prioritization scheme with 16 priority levels allows the user to specify the order in which multiple interrupt requests are to be handled.

Interrupt Processing via the Peripheral Event Controller (PEC)

A faster alternative to normal software controlled interrupt processing is servicing an interrupt requesting device with the INCA-D's integrated Peripheral Event Controller (PEC). Triggered by an interrupt request, the PEC performs a single word or byte data transfer between any two locations in the whole memory space through one of eight programmable PEC Service Channels. During a PEC transfer the normal program execution of the CPU is halted for just 1 instruction cycle. No internal program status information needs to be saved. The same prioritization scheme is used for PEC service as for normal interrupt processing. PEC transfers share the 2 highest priority levels.

Trap Functions

Trap functions are activated in response to special conditions that occur during the execution of instructions. A trap can also be caused externally by the Non-Maskable Interrupt pin $\overline{\text{NMI}}$. Several hardware trap functions are provided for handling erroneous conditions and exceptions that arise during the execution of an instruction. Hardware traps always have highest priority and cause immediate system reaction. The software trap function is invoked by the TRAP instruction, which generates a software interrupt for a specified interrupt vector. For all types of traps the current program status is saved on the system stack.

External Interrupt Processing

The INCA-D allows to connect external interrupt sources and provides several mechanisms to react on external events, including standard inputs, non-maskable interrupts and fast external interrupts.

8.1 Interrupt System Structure

The INCA-D provides up to 27 separate interrupt nodes that may be assigned to 16 priority levels. Each source of an interrupt or PEC request is supplied with a separate interrupt control register and interrupt vector. The control register contains the interrupt request flag, the interrupt enable bit, and the interrupt priority of the associated source.

The INCA-D provides a vectored interrupt system. In this system specific vector locations in the memory space are reserved for the reset, trap, and interrupt service functions. Whenever a request occurs, the CPU branches to the location that is associated with the respective interrupt source. This allows direct identification of the source that caused the request. The only exceptions are the class B hardware traps, which all share the same interrupt vector. The status flags in the Trap Flag Register (TFR) can then be used to determine which exception caused the trap. For the special software TRAP instruction, the vector address is specified by the operand field of the instruction, which is a seven bit trap number.

The reserved vector locations build a jump table in the low end of the INCA-D’s address space (segment 0). The jump table is made up of the appropriate jump instructions that transfer control to the interrupt or trap service routines, which may be located anywhere within the address space.

Each jump table entry occupies 2 words, except for the reset vector and the hardware trap vectors, which occupy 4 or 8 words.

The table below lists all sources that are capable of requesting interrupt or PEC service in the INCA-D, the associated interrupt vectors, their locations, their trap numbers and the SFR addresses of associated interrupt control registers. It also lists the mnemonics of the corresponding Interrupt Enable flags. The mnemonics are composed of a part that specifies the respective source, followed by a part that specifies their function (IE=Interrupt Enable flag). The same composition is used for the mnemonics of according interrupt request flags (IR=Interrupt Request flag; example: CC0IR belongs to interrupt source CC0INT) and for the names of according interrupt control registers (IC=Interrupt Control; example: CC0IC) which are not included in **Table 8-1**.

Table 8-1 INCA-D Interrupts and PEC Service Requests

Nr.	Source of Interrupt or PEC Service Request	Interrupt Name	Enable Flag	Vector Location	Trap Number	SFR Adr.
irq(0)	GPT Timer 2	T2INT	T2IE	00’005C$_H$	17$_H$ / 23$_D$	FF86
irq(1)	GPT Timer 3	T3INT	T3IE	00’0084$_H$	21$_H$ / 33$_D$	FF9E
irq(2)	USB Endpoint 1	USBEP1INT	USBEP1IE	00’0080$_H$	20$_H$ / 32$_D$	FF9C
irq(3)	USB Device Interrupts	USBINT	USBIE	00’00A4$_H$	29$_H$ / 41$_D$	FF9A

Nr.	Source of Interrupt or PEC Service Request	Interrupt Name	Enable Flag	Vector Location	Trap Number	SFR Adr.
irq(4)	USB Endpoint 0 USB Endpoint 5-15	USBEPINT	USBEPIE	00’00A0$_H$	28$_H$ / 40$_D$	FF98
irq(5)	IOM Data Transfer Unit	IOMTRA1INT	IOM1IE	00’0058$_H$	16$_H$ /22$_D$	FF84
irq(6)	IOM Data Transfer Unit	IOMTRA2INT	IOM2IE	00’0054$_H$	15$_H$ / 21$_D$	FF82
irq(7)	IOM Data Transfer Unit	IOMTRA3INT	IOM3IE	00’0050$_H$	14$_H$ / 20$_D$	FF80
irq(8)	IOM Data Transfer Unit	IOMTRA4INT	IOM4IE	00’004C$_H$	13$_H$ / 19$_D$	FF7E
irq(9)	IOM Data Transfer Unit	IOMTRA5INT	IOM5IE	00’0088$_H$	22$_H$ / 34$_D$	FF60
irq(10)	IOM Data Transfer Unit	IOMTRA6INT	IOM6IE	00’008C$_H$	23$_H$ / 35$_D$	FF62
irq(11)	IOM Data Transfer Unit	IOMTRA7INT	IOM7IE	00’0090$_H$	24$_H$ / 36$_D$	FF64
irq(12)	IOM Data Transfer Unit	IOMTRA8INT	IOM8IE	00’0094$_H$	25$_H$ / 37$_D$	FF66
irq(13)	IOM Handler Keyscanner Parallel Interface to DSP	COMB1INT	COMB1IE	00’0098$_H$	26$_H$ / 38$_D$	FF68
irq(14)	Software 0-2 Ext. Interrupt 3-7 ASC, SSC, GPT4-6, CAPREL	COMB2INT	COMB2IE	00’00A8$_H$	2A$_H$ / 42$_D$	FF6C
irq(15)	ASC Transmit Buffer	S0TBIR	S0TBIE	00’011C$_H$	47$_H$ / 71$_D$	F19C
irq(16)	ASC Receive	S0RIR	S0RIE	00’00AC$_H$	2B$_H$ / 43$_D$	FF6E
irq(17)	USB Endpoint 3	USBEP3INT	USBEP3IE	00’00B0$_H$	2C$_H$ / 44$_D$	FF70
irq(18)	SSC Receive	SSCRINT	SSCRIE	00’00B8$_H$	2E$_H$ /46$_D$	FF74
irq(19)	I2C Data Transmision Event	ICINTE	ICINTEIE	00’0118$_H$	46$_H$ / 70$_D$	F194
irq(20)	USB Endpoint 2	USBEP2INT	USBEP2IE	00’00BC$_H$	2F$_H$ / 47$_D$	FF76

Nr.	Source of Interrupt or PEC Service Request	Interrupt Name	Enable Flag	Vector Location	Trap Number	SFR Adr.
irq(21)	USB Endpoint 4	USBEP4INT	USBEP4IE	00’0040$_H$	10$_H$ / 16$_D$	FF78
irq(22)	I2C Data Transfer Event Interrupt	IICINTDINT	IICINTDIE	00’009C$_H$	27$_H$ / 39$_D$	FF6A
irq(23)	SSC Transmit	SSCTINT	SSCTIE	00’00B4$_H$	2D$_H$ /45$_D$	FF72
irq(24)	Fast Ext. Interrupt	FEX0INT	FEX0IE	00’0060$_H$	18$_H$ / 24$_D$	FF88
irq(25)	Fast Ext. Interrupt	FEX1INT	FEX1IE	00’0064$_H$	19$_H$ / 25$_D$	FF8A
irq(26)	Fast Ext. Interrupt	FEX2INT	FEX2IE	00’0068$_H$	1A$_H$ / 26$_D$	FF8C

Note: Each entry of the interrupt vector table provides space for two word instructions or one doubleword instruction. The respective vector location results from multiplying the trap number by 4 (4 bytes per entry).

Per interrupt node one Interrupt Control register is provided. The names of Interrupt Control registers are composed of a part that specifies the interrupt source followed by two letters that specify the function of register.

Table 8-2 lists the vector locations for hardware traps and the corresponding status flags in register TFR. It also lists the priorities of trap service for cases, where more than one trap condition might be detected within the same instruction. After any reset (hardware reset, software reset instruction SRST, or reset by watchdog timer overflow) program execution starts at the reset vector at location 00’0000$_H$. Reset conditions have priority over every other system activity and therefore have the highest priority (trap priority III).

Software traps may be initiated to any vector location between 00’0000$_H$ and 00’01FC$_H$. A service routine entered via a software TRAP instruction is always executed on the current CPU priority level which is indicated in bit field ILVL in register PSW. This means that routines entered via the software TRAP instruction can be interrupted by all hardware traps or higher level interrupt requests.

Table 8-2 Hardware Traps and Vector Locations

Exception Condition	Trap Flag	Trap Vector	Vector Location	Trap Number	Trap Priority
Reset Functions:					
Hardware Reset		RESET	00'0000$_H$	00$_H$	III
Software Reset		RESET	00'0000$_H$	00$_H$	III
Watchdog Timer Overflow		RESET	00'0000$_H$	00$_H$	III
Class A Hardware Traps:					
Non-Maskable Interrupt	NMI	NMITRAP	00'0008$_H$	02$_H$	II
Stack Overflow	STKOF	STOTRAP	00'0010$_H$	04$_H$	II
Stack Underflow	STKUF	STUTRAP	00'0018$_H$	06$_H$	II
Debug Trap	DEBUG	DEBTRAP	00'0020$_H$	08$_H$	II
Class B Hardware Traps:					
Undefined Opcode	UNDOPC	BTRAP	00'0028$_H$	0A$_H$	I
Protected Instruction Fault	PRTFLT #	BTRAP	00'0028$_H$	0A$_H$	I
Illegal Word Operand Access	ILLOPA	BTRAP	00'0028$_H$	0A$_H$	I
Illegal Instruction Access	ILLINA	BTRAP	00'0028$_H$	0A$_H$	I
Illegal External Bus Access	ILLBUS	BTRAP	00'0028$_H$	0A$_H$	I
Reserved			[2C$_H$ – 3C$_H$]	[0B$_H$ – 0F$_H$]	
Software Traps TRAP Instruction			Any [00'0000$_H$ – 00'01FC$_H$] in steps of 4$_H$	Any [00$_H$ – 7F$_H$]	Current CPU Priority

Note: A software reset is also executed in case of not allowed instruction access to the memory.

8.2 Combined Interrupt Sources

Some events in the INCA-D are indicated by means of a single interrupt output. Thus the interrupt request nodes irq(3), irq(4), irq(13) and irq(14) represent combined interrupt requests.

Since only one interrupt line is provided, the cause of an interrupt can be determined by reading the corresponding interrupt node status registers which belongs to the interrupt node. These nodes are described in the following sub-chapters.

The name of status register referring to node irq(x) is IRQxx_STA.

Note: After reading the interrupt request or interrupt status register, they will be cleared.

8.2.1 Interrupt Node 3 (USBINT)

The interrupt 3 comprises all interrupts that are related to the USB device itself.

The USB device interrupt request register DIRR contains the device specific interrupt flags of the USB module. These flags are set after the occurrence of special events. If a request flag has been set, it is automatically cleared after a read operation of the DIRR register.

The interrupts contained in the DIRR register can individually be masked in the DIER register. For further details refer to **Chapter 22**.

The structure is shown in figure 8-1.

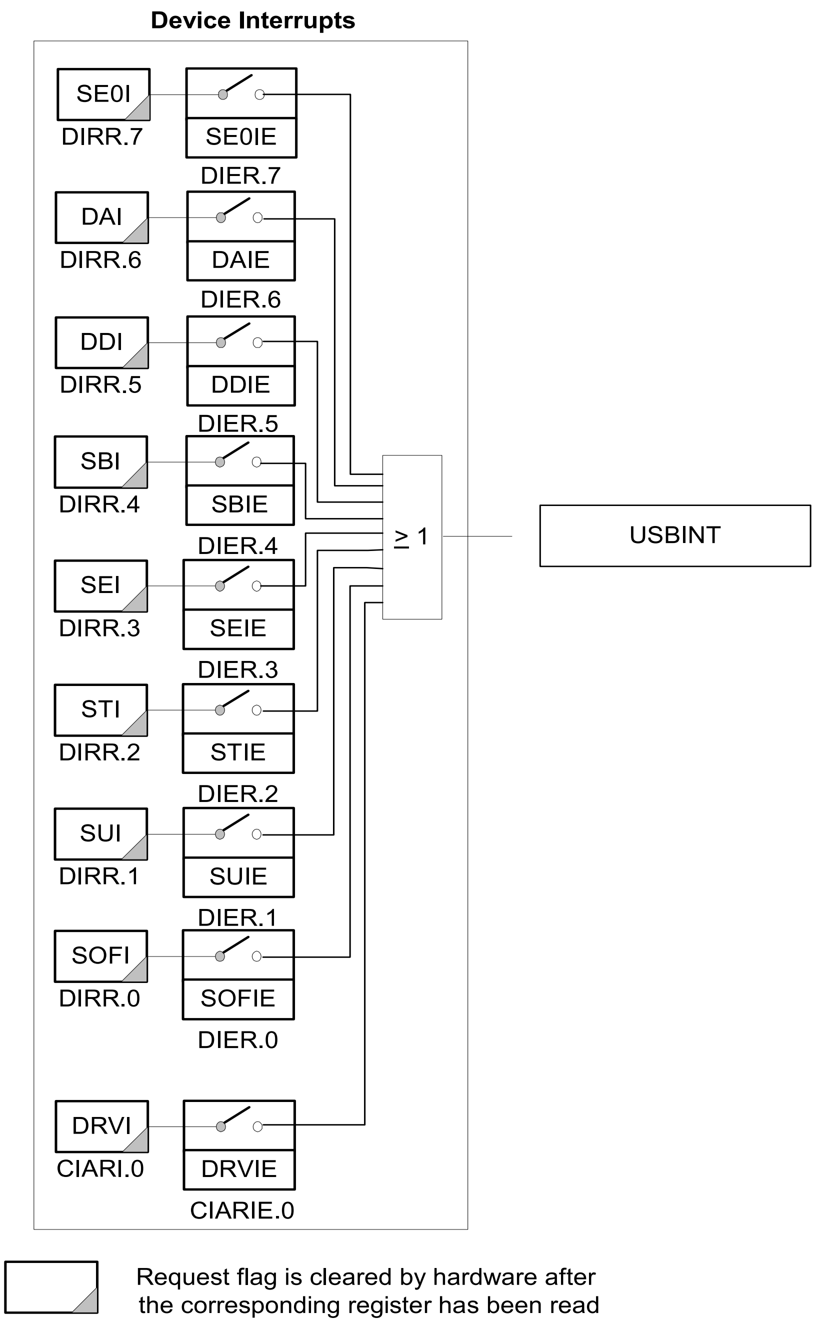


Figure 8-1 Combined interrupts for USBINT

Additionally, the Configuration, Interface and Alternate Setting Interrupt request Register (CIARI) sends an interrupt to the uC whenever the host programs multiple device configurations or interfaces.

8.2.2 Interrupt Node 4 (USBEPINT)

The interrupt node 4 handles all interrupts that are related to the USB endpoints EP0 and EP5-EP15. The different interrupt sources are handled by the EPIEn (Endpoint Interrupt Enable Register for Endpoint n), EPIRn ((Endpoint Interrupt Request Register for Endpoint n) and EPBCn register (Endpoint n Buffer Control Registerfor Endpoint n) of the USB module.

For further details refer to **Chapter 22**.

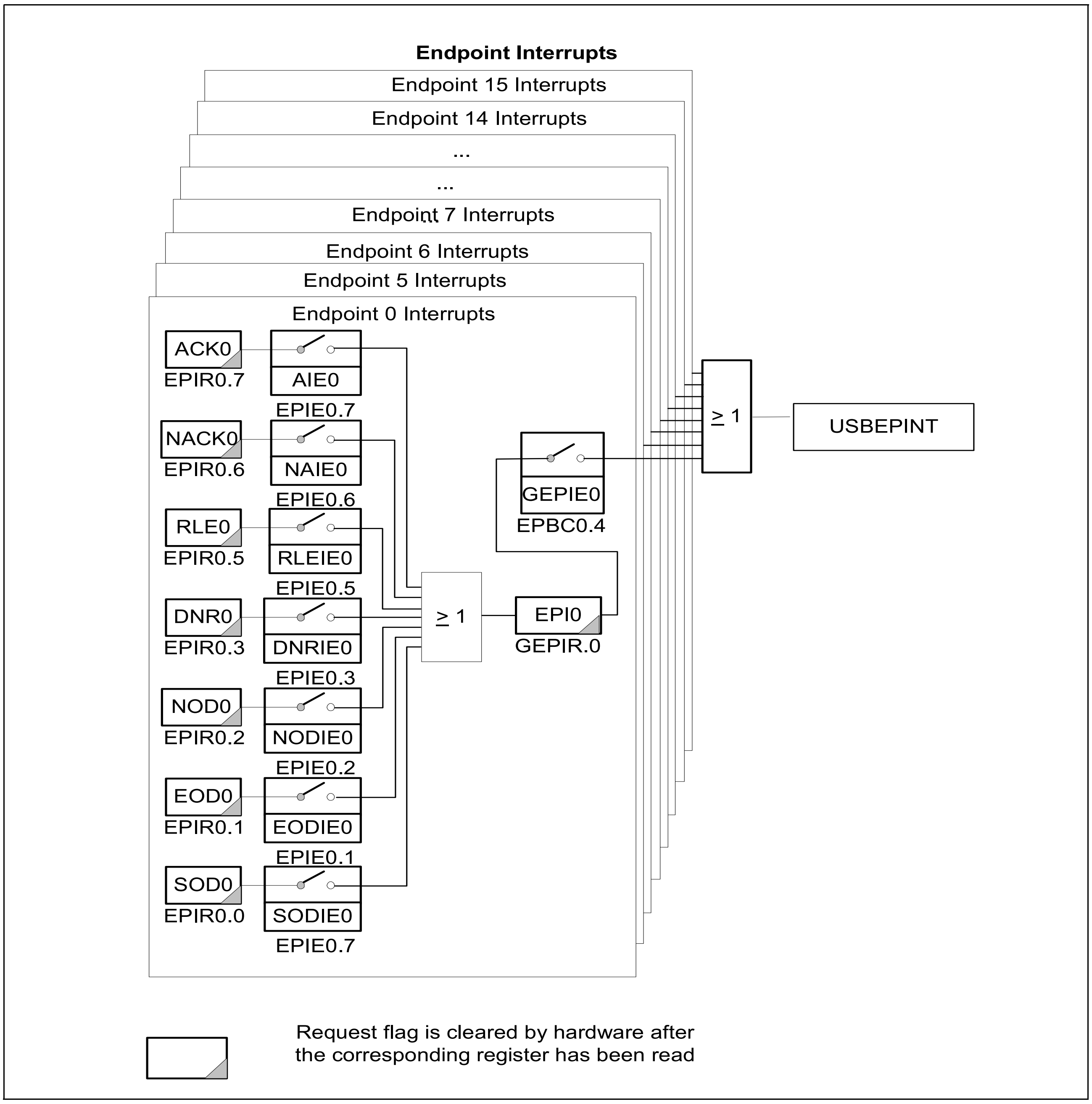


Figure 8-2 Combined interrupts for USBEPINT

8.2.3 Interrupt Node 13 (COMB1INT)

The interrupts of the Keypad scanner, the PIDD (parallel interface to the DSP), IOM-2 Handler, HDLC controller and the Line transceiver are combined to interrupt node COMB1INT, which belongs to interrupt node 13.

The status register IRQ13_STA contains the interrupt request bits of the different interrupt sources.

The mask register IRQ13_MSK can be used to selectively mask interrupts of the different sources.

The corrresponding registers are described in the dedicated chapters.

Table 8-3 Combined Interrupt Sources of node 13

Interrupt Sources	**Bit in Status Register**	**Source of Interrupt**
Keypad Scanner	KEYIR	*no further collection*
Parallel Interface to DSP	PIDDIR	*no further collection*
IOM Transfer Unit	ITRFRIR	*no further collection*
HDLC controller	HDLCIR	RME
HDLC controller	HDLCIR	RPF
HDLC controller	HDLCIR	RFO
HDLC controller	HDLCIR	XPR
HDLC controller	HDLCIR	XMR
HDLC controller	HDLCIR	XDU
Transceiver Level Detect	TRANIR	LD
Transceiver Info changed	TRANIR	RIC
CI Handler	CICIR	CIC0
CI Handler	CICIR	CIC1
Synchronous Transfer Unit	STIR	STIxy
Synchronous Transfer Unit	STIR	STOVxy

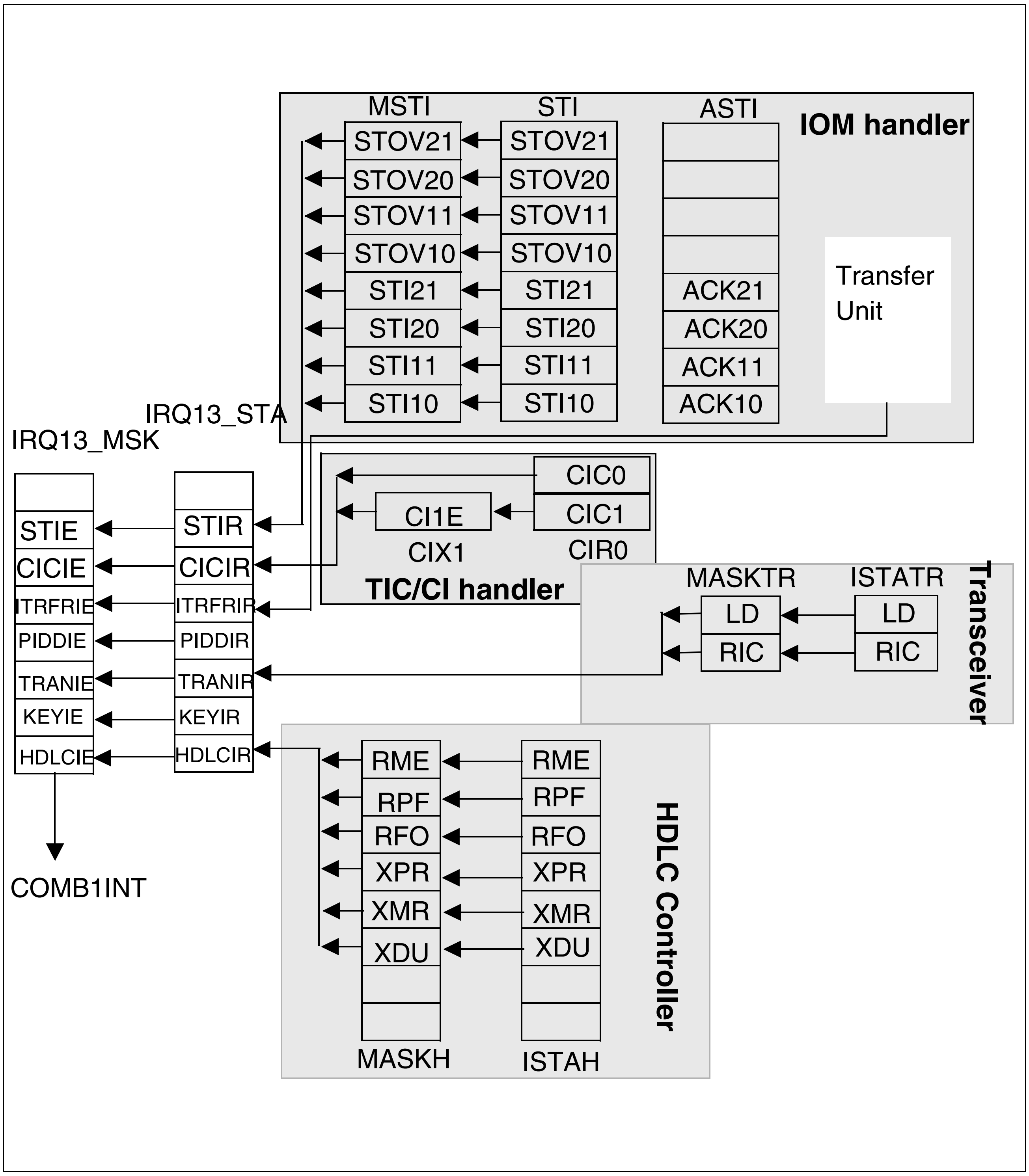


Figure 8-3 Combined Interrupts of node 13

IRQ13_STA (DF20)$_H$ **XBUS-SFR** **Reset Value:0000$_H$**

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
HDLCIR	KEYIR	TRANIR	PIDDIR	ITRFRIR	CICIR	STIR	0
rw	rw	rw	rw	rw	rw	rw	-

Bit	Value	Meaning
STIR, CICIR ITRFRIR PIDDIR TRANIR KEYIR HDLCIR	0	no INT request originating from corresponding registers (compare with **Figure 8-3**)
STIR, CICIR ITRFRIR PIDDIR TRANIR KEYIR HDLCIR	1	INT request from corresponding registers

IRQ13_MSK (DF22$_H$) **XBUS-SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8
0	0	0	0	0	0	0	0
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
HDLCI	KEYI	TRANI	PIDDI	ITRFRI	CICI	STI	0
rw	rw	rw	rw	rw	rw	rw	-

Bit	Value	Meaning
STI , CICI ITRFRI PIDDI TRANI KEYI HDLCI	0	Interrupt request of the dedicated source is not masked (compare with **Figure 8-3**)
STI , CICI ITRFRI PIDDI TRANI KEYI HDLCI	1	Interrupt request of the dedicated source is masked (compare with **Figure 8-3**)

8.2.4 Interrupt Node 14 (COMB2INT)

Different interrupt sources are combined to interrupt COMB2INT as listed in **Table 8-4**.

This interrupt belongs to interrupt node 14.

The status register IRQ14_STA contains the interrupt request bits of the different interrupt sources.

The mask register IRQ14_MSK can be used to selectively mask interrupts of the different sources.

Table 8-4 Combined Interrupt Sources of Node 14

Interrupt Sources	Bit in Status Register
GPT Timer 4	T4IR
GPT Timer 5	T5IR
GPT Timer 6	T6IR
GPT2 CAPREL	CRIR
Software Interrupt 0	SW0IR
Software Interrupt 1	SW1IR
Software Interrupt 2	SW2IR
ASC transmit	S0TIR
ASC error	S0EIR
SSC error	SSCEIR
I2C Protocol Event	ICINTPIR
External Interrupt 3	E3IR
External Interrupt 4	E4IR
External Interrupt 5	E5IR
External Interrupt 6	E6IR
Externall Interrupt 7	E7IR

IRQ14_STA (DF24$_H$) **XBUS-SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8
E7IR	E6IR	E5IR	E4IR	E3IR	ICINTPIR	SSCEIR	S0EIR
rw	rw	rw	rw	rw	rw	rw	rw

7	6	5	4	3	2	1	0
S0TIR	SW2IR	SW1IR	SW0IR	CRIR	T6IR	T5IR	T4IR
rw	rw	rw	rw	rw	rw	rw	rw

Bit	Value	Meaning
(all)	0	no INT request originating from corresponding source (compare with **Table 8-4**)
	1	INT request from corresponding source

IRQ14_MSK (DF26$_H$) **XBUS-SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8
E7I	E6I	E5I	E4I	E3I	ICINTPI	SSCEI	S0EI
rw	rw	rw	rw	rw	rw	rw	rw

7	6	5	4	3	2	1	0
S0TI	SW2I	SW1I	SW0I	CRI	T6I	T5I	T4I
rw	rw	rw	rw	rw	rw	rw	rw

Bit	Value	Meaning
(all)	0	interrupt requests from the corresponding sources are not masked (compare with **Table 8-4**)
	1	INT from corresponding source is masked

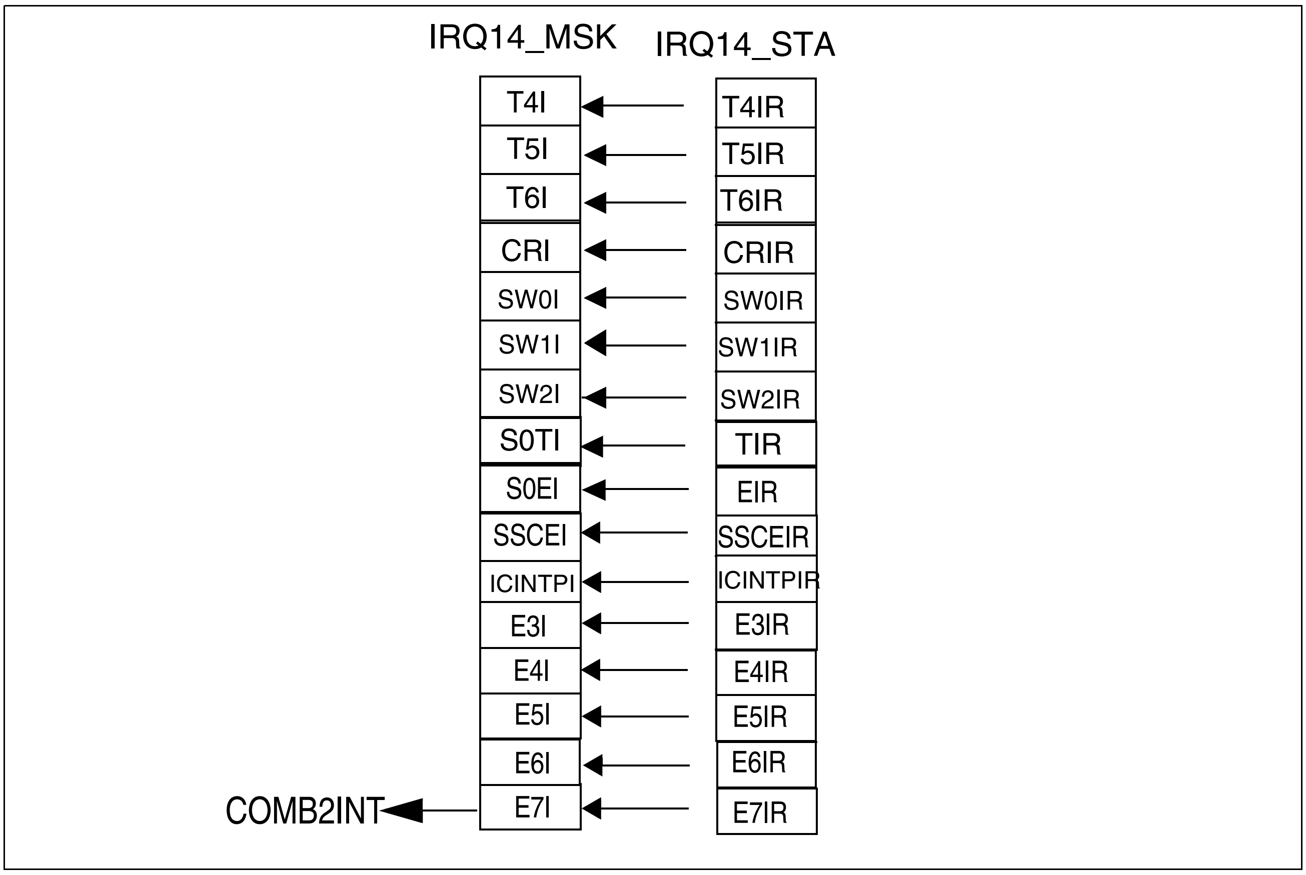


Figure 8-4 Combined Interrupts of node 14

8.2.5 Normal Interrupt Processing and PEC Service

During each instruction cycle one out of all sources which require PEC or interrupt processing is selected according to its interrupt priority. This priority of interrupts and PEC requests is programmable in two levels. Each requesting source can be assigned to a specific priority. A second level (called "group priority") allows to specify an internal order for simultaneous requests from a group of different sources on the same priority level. At the end of each instruction cycle the one source request with the highest current priority will be determined by the interrupt system. This request will then be serviced, if its priority is higher than the current CPU priority in register PSW.

8.2.6 Interrupt System Register Description

Interrupt processing is controlled globally by register PSW through a general interrupt enable bit (IEN) and the CPU priority field (ILVL). Additionally the different interrupt sources are controlled individually by their specific interrupt control registers (...IC). Thus, the acceptance of requests by the CPU is determined by both the individual interrupt control registers and the PSW. PEC services are controlled by the respective PECCx register and the source and destination pointers, which specify the task of the respective PEC service channel.

8.3 Interrupt Control Registers

All interrupt control registers are organized identically. The lower 8 bits of an interrupt control register contain the complete interrupt status information of the associated source, which is required during one round of prioritization, the upper 8 bits of the respective register are reserved. All interrupt control registers are bit-addressable and all bits can be read or written via software. This allows each interrupt source to be programmed or modified with just one instruction. When accessing interrupt control registers through instructions which operate on word data types, their upper 8 bits (15...8) will return zeros, when read, and will discard written data.

The layout of the Interrupt Control registers shown below applies to each xxIC register, where xx stands for the mnemonic for the respective source.

xxIC (yyyy$_H$ / XX$_H$) SFR Reset Value: xxxx xxxx 0000 0000

15	14	13	12	11	10	9	8
-	-	-	-	-	-	-	-

7	6	5	4	3	2	1	0
xxIR	xxIE	ILVL				GLVL	
rw	rw	rw				rw	

Bit	Function
GLVL	**Group Level** Defines the internal order for simultaneous requests of the same priority. 3: Highest group priority 0: Lowest group priority
ILVL	**Interrupt Priority Level** Defines the priority level for the arbitration of requests. F$_H$: Highest priority level 0$_H$: Lowest priority level

xxIE	**Interrupt Enable Control Bit** (individually enables/disables a specific source) ‘0’: Interrupt request is disabled ‘1’: Interrupt Request is enabled
xxIR	**Interrupt Request Flag** ‘0’: No request pending ‘1’: This source has raised an interrupt request

The Interrupt Request Flag is set by hardware whenever a service request from the respective source occurs. It is cleared automatically upon entry into the interrupt service routine or upon a PEC service. In the case of PEC service the Interrupt Request flag remains set, if the COUNT field in register PECCx of the selected PEC channel decrements to zero. This allows a normal CPU interrupt to respond to a completed PEC block transfer.

Note: Modifying the Interrupt Request flag via software causes the same effects as if it had been set or cleared by hardware.

Interrupt Priority Level and Group Level

The four bits of bit field ILVL specify the priority level of a service request for the arbitration of simultaneous requests. The priority increases with the numerical value of ILVL, so 0000_B is the lowest and 1111_B is the highest priority level.

When more than one interrupt request on a specific level gets active at the same time, the values in the respective bit fields GLVL are used for second level arbitration to select one request for being serviced. Again the group priority increases with the numerical value of GLVL, so 00_B is the lowest and 11_B is the highest group priority.

Note: All interrupt request sources that are enabled and programmed to the same priority level must always be programmed to different group priorities. Otherwise an incorrect interrupt vector will be generated.

Upon entry into the interrupt service routine, the priority level of the source that won the arbitration and who’s priority level is higher than the current CPU level, is copied into bit field ILVL of register PSW after pushing the old PSW contents on the stack.

The interrupt system of the INCA-D allows nesting of up to 15 interrupt service routines of different priority levels (level 0 cannot be arbitrated).

Interrupt requests that are programmed to priority levels 15 or 14 (ie, ILVL=$111X_B$) will be serviced by the PEC, unless the COUNT field of the associated PECC register contains zero. In this case the request will instead be serviced by normal interrupt processing. Interrupt requests that are programmed to priority levels 13 through 1 will always be serviced by normal interrupt processing.

Note: Priority level 0000_B is the default level of the CPU. Therefore a request on level 0 will never be serviced, because it can never interrupt the CPU. However, an enabled interrupt request on level 0000_B will terminate the INCA-D's Idle mode and reactivate the CPU.

For interrupt requests which are to be serviced by the PEC, the associated PEC channel number is derived from the respective ILVL (LSB) and GLVL (see figure below). So programming a source to priority level 15 (ILVL=1111_B) selects the PEC channel group 7...4, programming a source to priority level 14 (ILVL=1110_B) selects the PEC channel group 3...0. The actual PEC channel number is then determined by the group priority field GLVL.

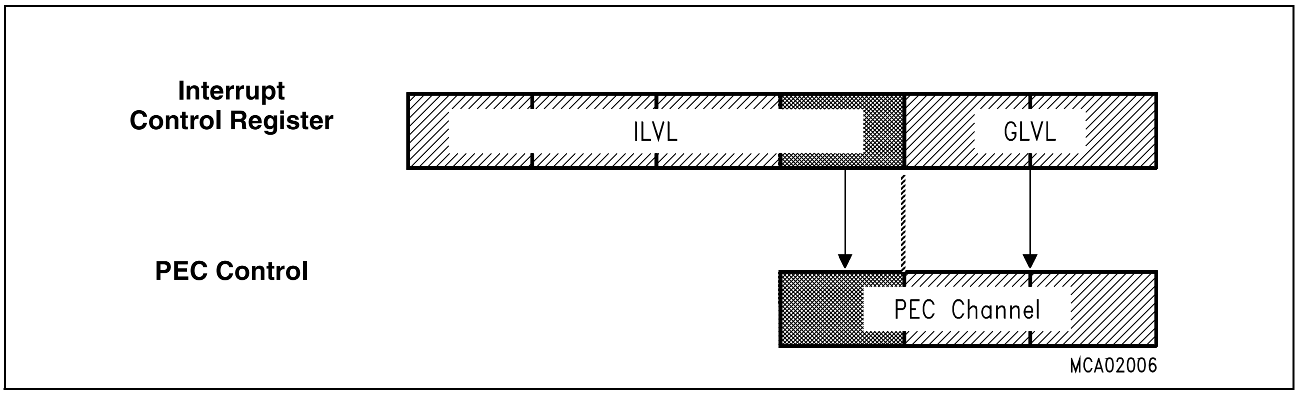


Figure 8-5 Priority Levels and PEC Channels

Simultaneous requests for PEC channels are prioritized according to the PEC channel number, where channel 0 has lowest and channel 7 has highest priority.

Note: All sources that request PEC service must be programmed to different PEC channels. Otherwise an incorrect PEC channel may be activated.

The table below shows in a few examples, which action is executed with a given programming of an interrupt control register.

Priority Level		**Type of Service**	
ILVL	**GLVL**	**COUNT = 00H**	**COUNT ≠ 00_H**
1 1 1 **1**	**1 1**	CPU interrupt, level 15, group priority 3	PEC service, channel 7
1 1 1 **1**	**1 0**	CPU interrupt, level 15, group priority 2	PEC service, channel 6
1 1 1 **0**	**1 0**	CPU interrupt, level 14, group priority 2	PEC service, channel 2
1 1 0 1	1 0	CPU interrupt, level 13, group priority 2	CPU interrupt, level 13, group priority 2

Priority Level		Type of Service	
ILVL	**GLVL**	**COUNT = 00H**	**COUNT ≠ 00_H**
0 0 0 1	1 1	CPU interrupt, level 1, group priority 3	CPU interrupt, level 1, group priority 3
0 0 0 1	0 0	CPU interrupt, level 1, group priority 0	CPU interrupt, level 1, group priority 0
0 0 0 0	X X	No service!	No service!

Note: All requests on levels 13...1 cannot initiate PEC transfers. They are always serviced by an interrupt service routine. No PECC register is associated and no COUNT field is checked.

Interrupt Control Functions in the PSW

The Processor Status Word (PSW) is functionally divided into 2 parts: the lower byte of the PSW basically represents the arithmetic status of the CPU, the upper byte of the PSW controls the interrupt system of the INCA-D and the arbitration mechanism for the external bus interface.

Note: Pipeline effects have to be considered when enabling/disabling interrupt requests via modifications of register PSW (see chapter "The Central Processing Unit").

PSW (FF10$_H$ / 88$_H$) SFR Reset Value: 00$_H$

15	14	13	12	11	10	9	8
ILVL				IEN	0	0	0
rw				rw	r	r	r

7	6	5	4	3	2	1	0
0	0	MULIP	E	Z	V	C	N
r	r	rw	rw	rw	rw	rw	rw

Bit	Function
N, C, V, Z, E, MULIP	**CPU status flags** (Described in section "The Central Processing Unit", page 53). Define the current status of the CPU (ALU, multiplication unit).
ILVL	**CPU Priority Level** Defines the current priority level for the CPU F$_H$: Highest priority level 0$_H$: Lowest priority level
IEN	**Interrupt Enable Control Bit** (globally enables/disables interrupt requests) '0': Interrupt requests are disabled '1': Interrupt requests are enabled

CPU Priority ILVL defines the current level for the operation of the CPU. This bit field reflects the priority level of the routine that is currently executed. Upon the entry into an interrupt service routine this bit field is updated with the priority level of the request that is being serviced. The PSW is saved on the system stack before. The CPU level determines the minimum interrupt priority level that will be serviced. Any request on the same or a lower level will not be acknowledged. The current CPU priority level may be adjusted via software to control which interrupt request sources will be acknowledged.

PEC transfers do not really interrupt the CPU, but rather "steal" a single cycle, so PEC services do not influence the ILVL field in the PSW.

Hardware traps switch the CPU level to maximum priority (ie. 15) so no interrupt or PEC requests will be acknowledged while an exception trap service routine is executed.

Note: The TRAP instruction does not change the CPU level, so software invoked trap service routines may be interrupted by higher requests.

Interrupt Enable bit IEN globally enables or disables PEC operation and the acceptance of interrupts by the CPU. When IEN is cleared, no interrupt requests are accepted by the CPU. When IEN is set to '1', all interrupt sources, which have been individually enabled by the interrupt enable bits in their associated control registers, are globally enabled.

Note: Traps are non-maskable and are therefore not affected by the IEN bit.

8.4 Operation of the PEC Channels

The INCA-D's Peripheral Event Controller (PEC) provides 8 PEC service channels, which move a single byte or word between two locations in the whole memory space. This is the fastest possible interrupt response and in many cases is sufficient to service the respective peripheral request (eg. serial channels, etc.). Each channel is controlled by a dedicated PEC Channel Counter/Control register (PECCx) and a pair of pointers for source (SRCPx) and destination (DSTPx) of the data transfer.

For further details regarding Extended PEC transfers refer to **Chapter 6.3**.

The PECC registers control the action that is performed by the respective PEC channel.

PECCx (FECy$_H$ / 6z$_H$, see table) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
					INC		BWT	COUNT							
-	-	-	-	-	rw		rw	rw							

Bit	Function
COUNT	PEC Transfer Count Counts PEC transfers and influences the channel’s action (see table below)
BWT	Byte / Word Transfer Selection 0: Transfer a Word 1: Transfer a Byte
INC	**Increment Control** (Modification of SRCPx or DSTPx) 0 0: Pointers are not modified 0 1: Increment DSTPx by 1 or 2 (BWT) 1 0: Increment SRCPx by 1 or 2 (BWT) 1 1: Reserved. Do not use this combination. (changed to 10 by hardware)

PEC Control Register Addresses

Register	Address	Reg. Space	Register	Address	Reg. Space
PECC0	FEC0$_H$ / 60$_H$	SFR	PECC4	FEC8$_H$ / 64$_H$	SFR
PECC1	FEC2$_H$ / 61$_H$	SFR	PECC5	FECA$_H$ / 65$_H$	SFR
PECC2	FEC4$_H$ / 62$_H$	SFR	PECC6	FECC$_H$ / 66$_H$	SFR
PECC3	FEC6$_H$ / 63$_H$	SFR	PECC7	FECE$_H$ / 67$_H$	SFR

Byte/Word Transfer bit BWT controls, if a byte or a word is moved during a PEC service cycle. This selection controls the transferred data size and the increment step for the modified pointer.

Increment Control Field INC controls, if one of the PEC pointers is incremented after the PEC transfer. It is not possible to increment both pointers, however. If the pointers are not modified (INC=’00’), the respective channel will always move data from the same source to the same destination.

Note: The reserved combination ‘11’ is changed to ‘10’ by hardware. However, it is not recommended to use this combination.

The PEC Transfer Count Field COUNT controls the action of a respective PEC channel, where the content of bit field COUNT at the time the request is activated selects the action. COUNT may allow a specified number of PEC transfers, unlimited transfers or no PEC service at all.

The table below summarizes, how the COUNT field itself, the interrupt requests flag IR and the PEC channel action depends on the previous content of COUNT.

Previous COUNT	Modified COUNT	IR after PEC service	Action of PEC Channel and Comments
FF_H	FF_H	‘0’	Move a Byte / Word Continuous transfer mode, ie. COUNT is not modified
FE_H..02_H	FD_H..01_H	‘0’	Move a Byte / Word and decrement COUNT
01_H	00_H	‘1’	Move a Byte / Word Leave request flag set, which triggers another request
00_H	00_H	(‘1’)	**No action!** Activate interrupt service routine rather than PEC channel.

The PEC transfer counter allows to service a specified number of requests by the respective PEC channel, and then (when COUNT reaches 00_H) activate the interrupt service routine, which is associated with the priority level. After each PEC transfer the COUNT field is decremented and the request flag is cleared to indicate that the request has been serviced.

Continuous transfers are selected by the value FF_H in bit field COUNT. In this case COUNT is not modified and the respective PEC channel services any request until it is disabled again.

When COUNT is decremented from 01_H to 00_H after a transfer, the request flag is not cleared, which generates another request from the same source. When COUNT already contains the value 00_H, the respective PEC channel remains idle and the associated interrupt service routine is activated instead. This allows to choose, if a level 15 or 14 request is to be serviced by the PEC or by the interrupt service routine.

Note: PEC transfers are only executed, if their priority level is higher than the CPU level, ie. only PEC channels 7...4 are processed, while the CPU executes on level 14. All interrupt request sources that are enabled and programmed for PEC service should use different channels. Otherwise only one transfer will be performed for all simultaneous requests. When COUNT is decremented to 00_H, and the CPU is to be interrupted, an incorrect interrupt vector will be generated.

The source and destination pointers specifiy the locations between which the data is to be moved. A pair of pointers (SRCPx and DSTPx) is associated with each of the 8 PEC channels. These pointers do not reside in specific SFRs, but are mapped into the internal RAM of the INCA-D just below the bit-addressable area (see figure below).

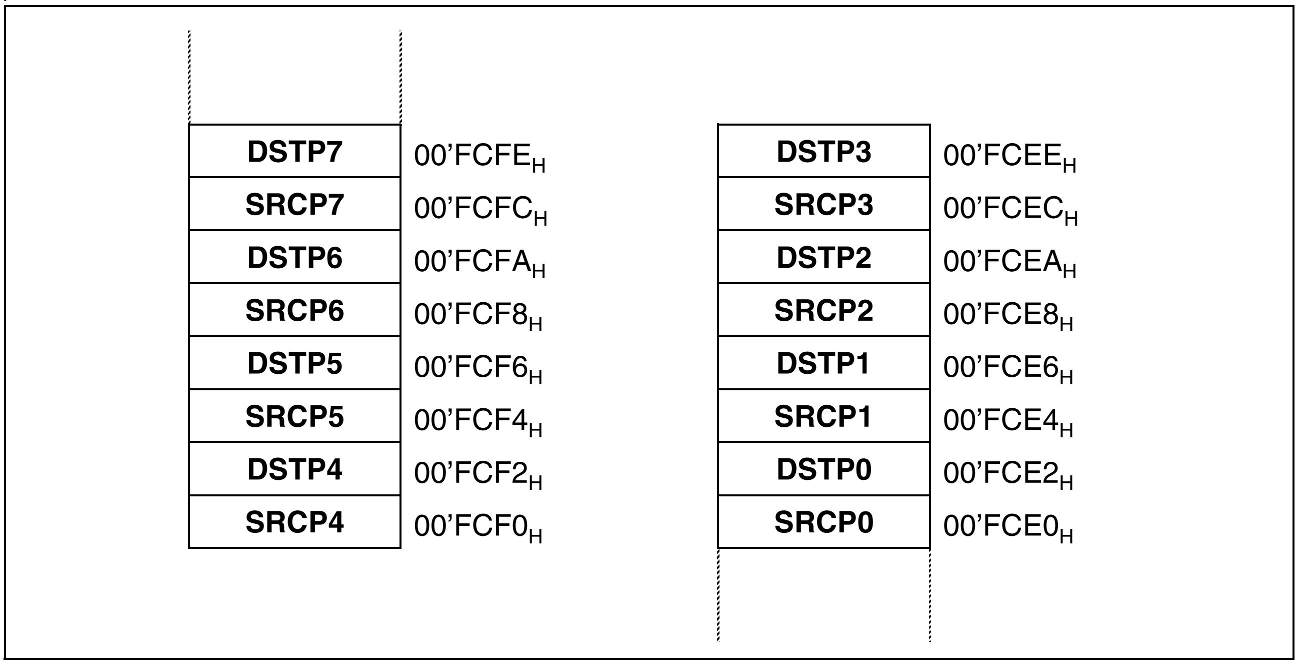


Figure 8-6 Mapping of PEC Pointers into the Internal RAM

The pointer locations for inactive PEC channels may be used for general data storage. Only the required pointers occupy RAM locations.

Note: If word data transfer is selected for a specific PEC channel (ie. BWT='0'), the respective source and destination pointers must both contain a valid word address which points to an even byte boundary. Otherwise the Illegal Word Access trap will be invoked, when this channel is used.

8.5 Prioritization of Interrupt and PEC Service Requests

Interrupt and PEC service requests from all sources can be enabled, so they are arbitrated and serviced (if they win), or they may be disabled, so their requests are disregarded and not serviced.

Enabling and disabling interrupt requests may be done via three mechanisms:

Control Bits allow to switch each individual source "ON" or "OFF", so it may generate a request or not. The control bits (xxIE) are located in the respective interrupt control registers. All interrupt requests may be enabled or disabled generally via bit IEN in register PSW. This control bit is the "main switch" that selects, if requests from any source are accepted or not.
For a specific request to be arbitrated the respective source's enable bit and the global enable bit must both be set.

The Priority Level automatically selects a certain group of interrupt requests that will be acknowledged, disclosing all other requests. The priority level of the source that won the arbitration is compared against the CPU's current level and the source is only serviced, if its level is higher than the current CPU level.

Changing the CPU level to a specific value via software blocks all requests on the same or a lower level. An interrupt source that is assigned to level 0 will be disabled and never be serviced.

The ATOMIC and EXTend instructions automatically disable all interrupt requests for the duration of the following 1...4 instructions. This is useful eg. for semaphore handling and does not require to re-enable the interrupt system after the unseparable instruction sequence.

Interrupt Class Management

An interrupt class covers a set of interrupt sources with the same importance, ie. the same priority from the system's viewpoint. Interrupts of the same class must not interrupt each other. The INCA-D supports this function with two features:

Classes with up to 4 members can be established by using the same interrupt priority (ILVL) and assigning a dedicated group level (GLVL) to each member. This functionality is built-in and handled automatically by the interrupt controller.

Classes with more than 4 members can be established by using a number of adjacent interrupt priorities (ILVL) and the respective group levels (4 per ILVL). Each interrupt service routine within this class sets the CPU level to the highest interrupt priority within the class. All requests from the same or any lower level are blocked now, ie. no request of this class will be accepted.

The example below establishes 3 interrupt classes which cover 2 or 3 interrupt priorities, depending on the number of members in a class. A level 6 interrupt disables all other sources in class 2 by changing the current CPU level to 8, which is the highest priority (ILVL) in class 2. Class 1 requests or PEC requests are still serviced in this case.

The 19 interrupt sources (excluding PEC requests) are so assigned to 3 classes of priority rather than to 7 different levels, as the hardware support would do.

Table 8-5 Software controlled Interrupt Classes (Example)

ILVL (Priority)	GLVL				Interpretation
	3	2	1	0	
15					PEC service on up to 8 channels
14					
13					
12	X	X	X	X	Interrupt Class 1 5 sources on 2 levels
11	X				
10					
9					

ILVL (Priority)	GLVL				Interpretation
	3	2	1	0	
8	X	X	X	X	Interrupt Class 2 9 sources on 3 levels
7	X	X	X	X	
6	X				
5	X	X	X	X	Interrupt Class 3 5 sources on 2 levels
4	X				
3					
2					
1					
0					No service!

8.6 Saving the Status during Interrupt Service

Before an interrupt request that has been arbitrated is actually serviced, the status of the current task is automatically saved on the system stack. The CPU status (PSW) is saved along with the location, where the execution of the interrupted task is to be resumed after returning from the service routine. This return location is specified through the Instruction Pointer (IP) and, in case of a segmented memory model, the Code Segment Pointer (CSP). Bit SGTDIS in register SYSCON controls, how the return location is stored.

The system stack receives the PSW first, followed by the IP (unsegmented) or followed by CSP and then IP (segmented mode). This optimizes the usage of the system stack, if segmentation is disabled.

The CPU priority field (ILVL in PSW) is updated with the priority of the interrupt request that is to be serviced, so the CPU now executes on the new level. If a multiplication or division was in progress at the time the interrupt request was acknowledged, bit MULIP in register PSW is set to ‘1’. In this case the return location that is saved on the stack is not the next instruction in the instruction flow, but rather the multiply or divide instruction itself, as this instruction has been interrupted and will be completed after returning from the service routine.

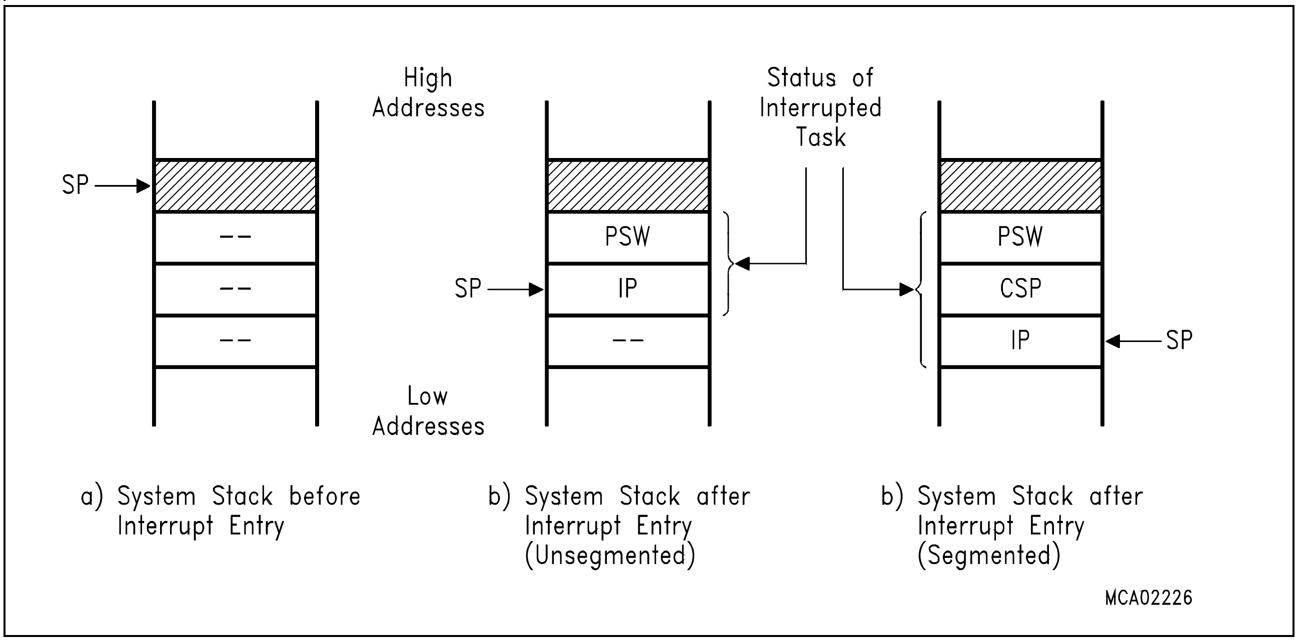


Figure 8-7 Task Status saved on the System Stack

The interrupt request flag of the source that is being serviced is cleared. The IP is loaded with the vector associated with the requesting source (the CSP is cleared in case of segmentation) and the first instruction of the service routine is fetched from the respective vector location, which is expected to branch to the service routine itself. The data page pointers and the context pointer are not affected.

When the interrupt service routine is left (RETI is executed), the status information is popped from the system stack in the reverse order, taking into account the value of bit SGTDIS.

Context Switching

An interrupt service routine usually saves all the registers it uses on the stack, and restores them before returning. The more registers a routine uses, the more time is wasted with saving and restoring. The INCA-D allows to switch the complete bank of CPU registers (GPRs) with a single instruction, so the service routine executes within its own, separate context.

The instruction “SCXT CP, #New_Bank” pushes the content of the context pointer (CP) on the system stack and loads CP with the immediate value “New_Bank”, which selects a new register bank. The service routine may now use its “own registers”. This register bank is preserved, when the service routine terminates, ie. its contents are available on the next call.
Before returning (RETI) the previous CP is simply POPped from the system stack, which returns the registers to the original bank.

Note: The first instruction following the SCXT instruction must not use a GPR.

Resources that are used by the interrupting program must eventually be saved and restored, eg. the DPPs and the registers of the MUL/DIV unit.

8.7 Interrupt Response Times

The interrupt response time defines the time from an interrupt request flag of an enabled interrupt source being set until the first instruction (I1) being fetched from the interrupt vector location. The basic interrupt response time for the INCA-D is 3 instruction cycles.

Pipeline Stage	Cycle 1	Cycle 2	Cycle 3	Cycle 4
FETCH	N	N + 1	N + 2	I1
DECODE	N - 1	N	TRAP (1)	TRAP (2)
EXECUTE	N - 2	N - 1	N	TRAP
WRITEBACK	N - 3	N - 2	N - 1	N

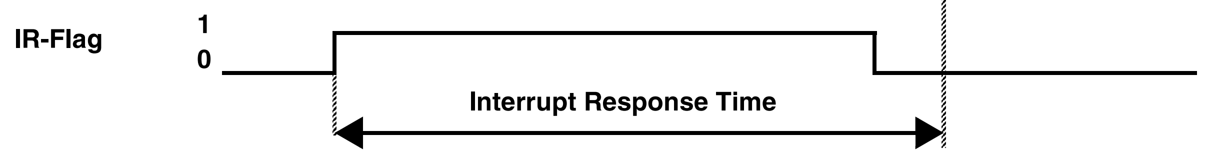


Figure 8-8 Pipeline Diagram for Interrupt Response Time

All instructions in the pipeline including instruction N (during which the interrupt request flag is set) are completed before entering the service routine. The actual execution time for these instructions (eg. waitstates) therefore influences the interrupt response time.

In the figure above the respective interrupt request flag is set in cycle 1 (fetching of instruction N). The indicated source wins the prioritization round (during cycle 2). In cycle 3 a TRAP instruction is injected into the decode stage of the pipeline, replacing instruction N+1 and clearing the source's interrupt request flag to '0'. Cycle 4 completes the injected TRAP instruction (save PSW, IP and CSP, if segmented mode) and fetches the first instruction (I1) from the respective vector location.

All instructions that entered the pipeline after setting of the interrupt request flag (N+1, N+2) will be executed after returning from the interrupt service routine.

The minimum interrupt response time is 5 states (10 TCL). This requires program execution from the internal code memory, no external operand read requests and setting the interrupt request flag during the last state of an instruction cycle. When the interrupt request flag is set during the first state of an instruction cycle, the minimum interrupt response time under these conditions is 6 state times (12 TCL).

The interrupt response time is increased by all delays of the instructions in the pipeline that are executed before entering the service routine (including N).

- When internal hold conditions between instruction pairs N-2/N-1 or N-1/N occur, or instruction N explicitly writes to the PSW or the SP, the minimum interrupt response time may be extended by 1 state time for each of these conditions.
- When instruction N reads an operand from the internal code memory, or when N is a call, return, trap, or MOV Rn, [Rm+ #data16] instruction, the minimum interrupt response time may additionally be extended by 2 state times during internal code memory program execution.
- In case instruction N reads the PSW and instruction N-1 has an effect on the condition flags, the interrupt response time may additionally be extended by 2 state times.

The worst case interrupt response time during internal code memory program execution adds to 12 state times (24 TCL).

Any reference to external locations increases the interrupt response time due to pipeline related access priorities. The following conditions have to be considered:

- Instruction fetch from an external location
- Operand read from an external location
- Result write-back to an external location

Depending on where the instructions, source and destination operands are located, there are a number of combinations. Note, however, that only access conflicts contribute to the delay.

A few examples illustrate these delays:

The worst case interrupt response time including external accesses will occur, when instructions N, N+1 and N+2 are executed out of external memory, instructions N-1 and N require external operand read accesses, instructions N-3 through N write back external operands, and the interrupt vector also points to an external location. In this case the interrupt response time is the time to perform 9 word bus accesses, because instruction I1 cannot be fetched via the external bus until all write, fetch and read requests of preceding instructions in the pipeline are terminated.

When the above example has the interrupt vector pointing into the internal code memory, the interrupt response time is 7 word bus accesses plus 2 states, because fetching of instruction I1 from internal code memory can start earlier.

When instructions N, N+1 and N+2 are executed out of external memory and the interrupt vector also points to an external location, but all operands for instructions N-3 through N are in internal memory, then the interrupt response time is the time to perform 3 word bus accesses.

When the above example has the interrupt vector pointing into the internal code memory, the interrupt response time is 1 word bus access plus 4 states.

After an interrupt service routine has been terminated by executing the RETI instruction, and if further interrupts are pending, the next interrupt service routine will not be entered until at least two instruction cycles have been executed of the program that was interrupted. In most cases two instructions will be executed during this time. Only one instruction will typically be executed, if the first instruction following the RETI instruction is a branch instruction (without cache hit), or if it reads an operand from internal code memory, or if it is executed out of the internal RAM.

Note: A bus access in this context includes all delays which can occur during an external bus cycle.

8.8 PEC Response Times

The PEC response time defines the time from an interrupt request flag of an enabled interrupt source being set until the PEC data transfer being started. The basic PEC response time for the INCA-D is 2 instruction cycles.

Pipeline Stage	Cycle 1	Cycle 2	Cycle 3	Cycle 4
FETCH	N	N + 1	N + 2	N + 2
DECODE	N - 1	N	PEC	N + 1
EXECUTE	N - 2	N - 1	N	PEC
WRITEBACK	N - 3	N - 2	N - 1	N

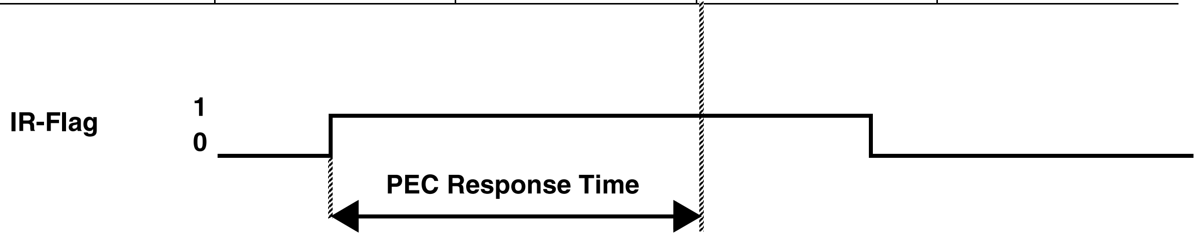


Figure 8-9 Pipeline Diagram for PEC Response Time

In **Figure 8-9** the respective interrupt request flag is set in cycle 1 (fetching of instruction N). The indicated source wins the prioritization round (during cycle 2). In cycle 3 a PEC transfer “instruction” is injected into the decode stage of the pipeline, suspending instruction N+1 and clearing the source's interrupt request flag to '0'. Cycle 4 completes the injected PEC transfer and resumes the execution of instruction N+1.

All instructions that entered the pipeline after setting of the interrupt request flag (N+1, N+2) will be executed after the PEC data transfer.

Note: When instruction N reads any of the PEC control registers PECC7...PECC0, while a PEC request wins the current round of prioritization, this round is repeated and the PEC data transfer is started one cycle later.

The minimum PEC response time is 3 states (6 TCL). This requires program execution from the internal code memory, no external operand read requests and setting the interrupt request flag during the last state of an instruction cycle. When the interrupt request flag is set during the first state of an instruction cycle, the minimum PEC response time under these conditions is 4 state times (8 TCL).

The PEC response time is increased by all delays of the instructions in the pipeline that are executed before starting the data transfer (including N).

- When internal hold conditions between instruction pairs N-2/N-1 or N-1/N occur, the minimum PEC response time may be extended by 1 state time for each of these conditions.
- When instruction N reads an operand from the internal code memory, or when N is a call, return, trap, or MOV Rn, [Rm+ #data16] instruction, the minimum PEC response time may additionally be extended by 2 state times during internal code memory program execution.
- In case instruction N reads the PSW and instruction N-1 has an effect on the condition flags, the PEC response time may additionally be extended by 2 state times.

The worst case PEC response time during internal code memory program execution adds to 9 state times (18 TCL).

Any reference to external locations increases the PEC response time due to pipeline related access priorities. The following conditions have to be considered:

- Instruction fetch from an external location
- Operand read from an external location
- Result write-back to an external location

Depending on where the instructions, source and destination operands are located, there are a number of combinations. Note, however, that only access conflicts contribute to the delay.

A few examples illustrate these delays:

The worst case interrupt response time including external accesses will occur, when instructions N and N+1 are executed out of external memory, instructions N-1 and N require external operand read accesses and instructions N-3, N-2 and N-1 write back external operands. In this case the PEC response time is the time to perform 7 word bus accesses.

When instructions N and N+1 are executed out of external memory, but all operands for instructions N-3 through N-1 are in internal memory, then the PEC response time is the time to perform 1 word bus access plus 2 state times.

Once a request for PEC service has been acknowledged by the CPU, the execution of the next instruction is delayed by 2 state times plus the additional time it might take to fetch the source operand from internal code memory or external memory and to write the destination operand over the external bus in an external program environment.

Note: A bus access in this context includes all delays which can occur during an external bus cycle.

8.9 External Interrupts

The INCA-D provides many possibilities to react on external asynchronous events by using dedicated pins or a number of I/O lines for interrupt input.

For three of these pins either a positive, a negative, or both a positive and a negative external transition can be selected to cause an interrupt or PEC service request (so called Fast External Interrupts). For five of these pins (External Interrupts 3-7 at Port 2) a rising edge, a falling edge or a detected high level on the interrupt request line cause an interrupt or PEC request.

Table 8-6 Pins to be used as External Interrupt Inputs

Port Pin	Original Function	Control Register
P2.0	Fast External Interrupt 0	EXICON
P2.1	Fast External Interrupt 1	EXICON
P2.2	Fast External Interrupt 2	EXICON
P2.3	External Interrupt 3	IRQ14_STA
P2.4	External Interrupt 4	IRQ14_STA
P2.5	External Interrupt 5	IRQ14_STA
P2.6	External Interrupt 6	IRQ14_STA
P2.7	External Interrupt 7	IRQ14_STA
P3.7	Auxiliary timer T2 input pin	T2CON
P3.5	Auxiliary timer T4 input pin	T4CON

Pins T2IN or T4IN can be used as external interrupt input pins when the associated auxiliary timer T2 or T4 in block GPT1 is configured for capture mode. This mode is selected by programming the mode control fields T2M or T4M in control registers T2CON or T4CON to 101_B. The active edge of the external input signal is determined by bit fields T2I or T4I. When these fields are programmed to $X01_B$, interrupt request flags T2IR or T4IR in registers T2IC or T4IC will be set on a positive external transition at pins T2IN or T4IN, respectively. When T2I or T4I are programmed to $X10_B$, then a negative external transition will set the corresponding request flag. When T2I or T4I are programmed to $X11_B$, both a positive and a negative transition will set the request flag. In all three cases, the contents of the core timer T3 will be captured into the auxiliary timer registers T2 or T4 based on the transition at pins T2IN or T4IN. When the interrupt enable bits T2IE or T4IE are set, a PEC request or an interrupt request for vector T2INT or T4INT will be generated.

Note: The non-maskable interrupt input pin $\overline{\text{NMI}}$ and the reset input $\overline{\text{RSTIN}}$ provide another possibility for the CPU to react on an external input signal. $\overline{\text{NMI}}$ and $\overline{\text{RSTIN}}$ are dedicated input pins, which cause hardware traps.

Regular External Interrupts at P2.3-7

The input pins that may be used for regular external interrupts are sampled every 16 TCL, ie. external events are scanned and detected in timeframes of 16 TCL. These five pins of Port 2 (P2.3...P2.7) can individually be programmed to determine the sensitivity, i.e. rising, falling or level sensitivity can be selected.

REXICON (00DF38$_H$) **Reset value: 0000$_H$**

15	14	13	12	11	10	9	8
-	-	-	-	-	-	RE7LS2	RE7LS1
						rw	rw

7	6	5	4	3	2	1	0
RE6LS2	RE6LS1	RE5LS2	RE5LS1	RE4LS2	RE4LS1	RE3LS2	RE3LS1
rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
RExLSn	**Regular External Interrupt x Level Sensitivity** 0 0: Interrupt on positive edge (rising) 0 1: Interrupt on negative edge (falling) 1 0: Interrupt on level detection, i.e. detected level is HIGH 1 1: Interrupt on level detection, i.e. detected level is HIGH

Fast External Interrupts

The INCA-D provides the capability to sample selected interrupt pins every 2 TCL, so external events are captured faster than with standard interrupt inputs.

Three pins of Port 2 (P2.2...P2.0) can individually be programmed to this fast interrupt mode, where also the trigger transition (rising, falling or both) can be selected. The External Interrupt Control register EXICON controls this feature for all 3 pins.

EXICON ($F1C0_H$ / $E0_H$) **ESFR** **Reset Value: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										EXI2ES		EXI1ES		EXI0ES	
-		-		-		-		-		rw		rw		rw	

Bit	Function
EXIxES	**External Interrupt x Edge Selection Field (x=7...0)** 0 0: Fast external interrupts disabled 0 1: Interrupt on positive edge (rising) 1 0: Interrupt on negative edge (falling) 1 1: Interrupt on any edge (rising or falling)

Note: The fast external interrupt inputs are sampled every 2 TCL. The interrupt request arbitration and processing, however, is executed every 8 TCL.

The interrupt control registers listed below (FEI2IC..FEI0IC) control the fast external interrupts of the INCA-D.

FEIxIC (See Table) SFR **Reset Value: - - 00_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								FEIx IR	FEIx IE	ILVL				GLVL	
-	-	-	-	-	-	-	-	rw	rw	rw				rw	

Note: Please refer to the general Interrupt Control Register description for an explanation of the control fields.

Table 8-7 Fast External Interrupt Control Register Addresses

Register	Address	External Interrupt
FEI0IC	$FF88_H$ / $C4_H$	FEX0IN
FEI1IC	$FF8A_H$ / $C5_H$	FEX1IN
FEI2IC	$FF8C_H$ / $C6_H$	FEX2IN

8.10 Trap Functions

Traps interrupt the current execution similar to standard interrupts. However, trap functions offer the possibility to bypass the interrupt system's prioritization process in cases where immediate system reaction is required. Trap functions are not maskable and always have priority over interrupt requests on any priority level.

The INCA-D provides two different kinds of trapping mechanisms. **Hardware traps** are triggered by events that occur during program execution (eg. illegal access or undefined opcode), **software traps** are initiated via an instruction within the current execution flow.

Software Traps

The TRAP instruction is used to cause a software call to an interrupt service routine. The trap number that is specified in the operand field of the trap instruction determines which vector location in the address range from $00'0000_H$ through $00'01FC_H$ will be branched to.

Executing a TRAP instruction causes a similar effect as if an interrupt at the same vector had occurred. PSW, CSP (in segmentation mode), and IP are pushed on the internal system stack and a jump is taken to the specified vector location. When segmentation is enabled and a trap is executed, the CSP for the trap service routine is set to code segment 0. No Interrupt Request flags are affected by the TRAP instruction. The interrupt service routine called by a TRAP instruction must be terminated with a RETI (return from interrupt) instruction to ensure correct operation.

Note: The CPU level in register PSW is not modified by the TRAP instruction, so the service routine is executed on the same priority level from which it was invoked. Therefore, the service routine entered by the TRAP instruction can be interrupted by other traps or higher priority interrupts, other than when triggered by a hardware trap.

Hardware Traps

Hardware traps are issued by faults or specific system states that occur during runtime of a program (not identified at assembly time). A hardware trap may also be triggered intentionally, eg. to emulate additional instructions by generating an Illegal Opcode trap. The INCA-D distinguishes eight different hardware trap functions. When a hardware trap condition has been detected, the CPU branches to the trap vector location for the respective trap condition. Depending on the trap condition, the instruction which caused the trap is either completed or cancelled (ie. it has no effect on the system state) before the trap handling routine is entered.

Hardware traps are non-maskable and always have priority over every other CPU activity. If several hardware trap conditions are detected within the same instruction cycle, the highest priority trap is serviced (see table in section “Interrupt System Structure”).

PSW, CSP (in segmentation mode), and IP are pushed on the internal system stack and the CPU level in register PSW is set to the highest possible priority level (ie. level 15), disabling all interrupts. The CSP is set to code segment zero, if segmentation is enabled. A trap service routine must be terminated with the RETI instruction.

The eight hardware trap functions of the INCA-D are divided into two classes:

Class A traps are

- external Non-Maskable Interrupt ($\overline{\text{NMI}}$)
- Stack Overflow
- Stack Underflow trap

These traps share the same trap priority, but have an individual vector address.

Class B traps are

- Undefined Opcode
- Protection Fault
- Illegal Word Operand Access
- Illegal Instruction Access
- Illegal External Bus Access Trap

These traps share the same trap priority, and the same vector address.

The bit-addressable Trap Flag Register (TFR) allows a trap service routine to identify the kind of trap which caused the exception. Each trap function is indicated by a separate request flag. When a hardware trap occurs, the corresponding request flag in register TFR is set to '1'.

TFR (FFAC$_H$ / D6$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
NMI	**STK OF**	**STK UF**	**OCD SF**	**-**	**-**	**-**	**-**	**UND OPC**	**-**	**-**	**-**	**PRT FLT**	**ILL OPA**	**ILL INA**	**ILL BUS**
rw	rw	rw	rw	-	-	-	-	rw	-	-	-	rw	rw	rw	rw

Bit	Function
ILLBUS	**Illegal External Bus Access Flag** An external access has been attempted with no external bus defined.
ILLINA	**Illegal Instruction Access Flag** A branch to an odd address has been attempted.
ILLOPA	**Illegal Word Operand Access Flag** A word operand access (read or write) to an odd address has been attempted.
PRTFLT	**Protection Fault Flag** A protected instruction with an illegal format has been detected.
UNDOPC	**Undefined Opcode Flag** The currently decoded instruction has no valid INCA-D opcode.
OCDSF	**OCDS Flag** An OCDS trap has been detected
STKUF	**Stack Underflow Flag** The current stack pointer value exceeds the content of register STKUN.
STKOF	**Stack Overflow Flag** The current stack pointer value falls below the content of register STKOV.
NMI	**Non Maskable Interrupt Flag** A negative transition (falling edge) has been detected on pin $\overline{\text{NMI}}$.

Note: The trap service routine must clear the respective trap flag, otherwise a new trap will be requested after exiting the service routine. Setting a trap request flag by software causes the same effects as if it had been set by hardware.

The reset functions (hardware, software, watchdog) may be regarded as a type of trap. Reset functions have the highest system priority (trap priority III).

Class A traps have the second highest priority (trap priority II), on the 3rd rank are class B traps, so a class A trap can interrupt a class B trap. If more than one class A trap occur at a time, they are prioritized internally, with the NMI trap on the highest and the stack underflow trap on the lowest priority.

All class B traps have the same trap priority (trap priority I). When several class B traps get active at a time, the corresponding flags in the TFR register are set and the trap service routine is entered. Since all class B traps have the same vector, the priority of service of simultaneously occurring class B traps is determined by software in the trap service routine.

A class A trap occurring during the execution of a class B trap service routine will be serviced immediately. During the execution of a class A trap service routine, however, any class B trap occurring will not be serviced until the class A trap service routine is exited with a RETI instruction. In this case, the occurrence of the class B trap condition is stored in the TFR register, but the IP value of the instruction which caused this trap is lost.

In the case where e.g. an Undefined Opcode trap (class B) occurs simultaneously with an NMI trap (class A), both the NMI and the UNDOPC flag is set, the IP of the instruction with the undefined opcode is pushed onto the system stack, but the NMI trap is executed. After return from the NMI service routine, the IP is popped from the stack and immediately pushed again because of the pending UNDOPC trap.

External NMI Trap

Whenever a high to low transition on the dedicated external $\overline{\text{NMI}}$ pin (Non-Maskable Interrupt) is detected, the NMI flag in register TFR is set and the CPU will enter the NMI trap routine. The IP value pushed on the system stack is the address of the instruction following the one after which normal processing was interrupted by the NMI trap.

Stack Overflow Trap

Whenever the stack pointer is decremented to a value which is less than the value in the stack overflow register STKOV, the STKOF flag in register TFR is set and the CPU will enter the stack overflow trap routine. Which IP value will be pushed onto the system stack depends on which operation caused the decrement of the SP. When an implicit decrement of the SP is made through a PUSH or CALL instruction, or upon interrupt or trap entry, the IP value pushed is the address of the following instruction. When the SP is decremented by a subtract instruction, the IP value pushed represents the address of the instruction after the instruction following the subtract instruction.

For recovery from stack overflow it must be ensured that there is enough excess space on the stack for saving the current system state (PSW, IP, in segmented mode also CSP) twice. Otherwise, a system reset should be generated.

Stack Underflow Trap

Whenever the stack pointer is incremented to a value which is greater than the value in the stack underflow register STKUN, the STKUF flag is set in register TFR and the CPU will enter the stack underflow trap routine. Again, which IP value will be pushed onto the system stack depends on which operation caused the increment of the SP. When an implicit increment of the SP is made through a POP or return instruction, the IP value pushed is the address of the following instruction. When the SP is incremented by an add instruction, the pushed IP value represents the address of the instruction after the instruction following the add instruction.

Undefined Opcode Trap

When the instruction currently decoded by the CPU does not contain a valid INCA-D opcode, the UNDOPC flag is set in register TFR and the CPU enters the undefined opcode trap routine. The IP value pushed onto the system stack is the address of the instruction that caused the trap.

This can be used to emulate unimplemented instructions. The trap service routine can examine the faulting instruction to decode operands for unimplemented opcodes based on the stacked IP. In order to resume processing, the stacked IP value must be incremented by the size of the undefined instruction, which is determined by the user, before a RETI instruction is executed.

Protection Fault Trap

Whenever one of the special protected instructions is executed where the opcode of that instruction is not repeated twice in the second word of the instruction and the byte following the opcode is not the complement of the opcode, the PRTFLT flag in register TFR is set and the CPU enters the protection fault trap routine. The protected instructions include DISWDT, EINIT, IDLE, PWRDN, SRST, and SRVWDT. The IP value pushed onto the system stack for the protection fault trap is the address of the instruction that caused the trap.

Illegal Word Operand Access Trap

Whenever a word operand read or write access is attempted to an odd byte address, the ILLOPA flag in register TFR is set and the CPU enters the illegal word operand access trap routine. The IP value pushed onto the system stack is the address of the instruction following the one which caused the trap.

Illegal Instruction Access Trap

Whenever a branch is made to an odd byte address, the ILLINA flag in register TFR is set and the CPU enters the illegal instruction access trap routine. The IP value pushed onto the system stack is the illegal odd target address of the branch instruction.

Illegal External Bus Access Trap

Whenever the CPU requests an external instruction fetch, data read or data write, and no external bus configuration has been specified, the ILLBUS flag in register TFR is set and the CPU enters the illegal bus access trap routine. The IP value pushed onto the system stack is the address of the instruction following the one which caused the trap.

9 Parallel Ports

The INCA-D features **Port 0** (inculdes 8 bit P0H and 8 bit P0L), **Port 1** (8 bit P1H and 8 bit P1L), **Port 2** (14 bit), **Port 3** (16 bit), **Port 4** (6 bit), **Port 6** (3 bit) and **Port 7** (10 bit).

The I/O ports are true bidirectional ports and may be used for general purpose Input/ Output controlled via software or may be used implicitly by INCA-D's integrated peripherals or the External Bus Controller.

All port lines are bit addressable, and all input/output lines are individually (bit-wise) programmable as inputs or outputs via direction registers.

Internal pull transistors are connected to the ports if the corresponding bits of register PxPUDEN are set to '1' . Either a pull down or a pull up transistors will be selected via register PxPUDSEL.

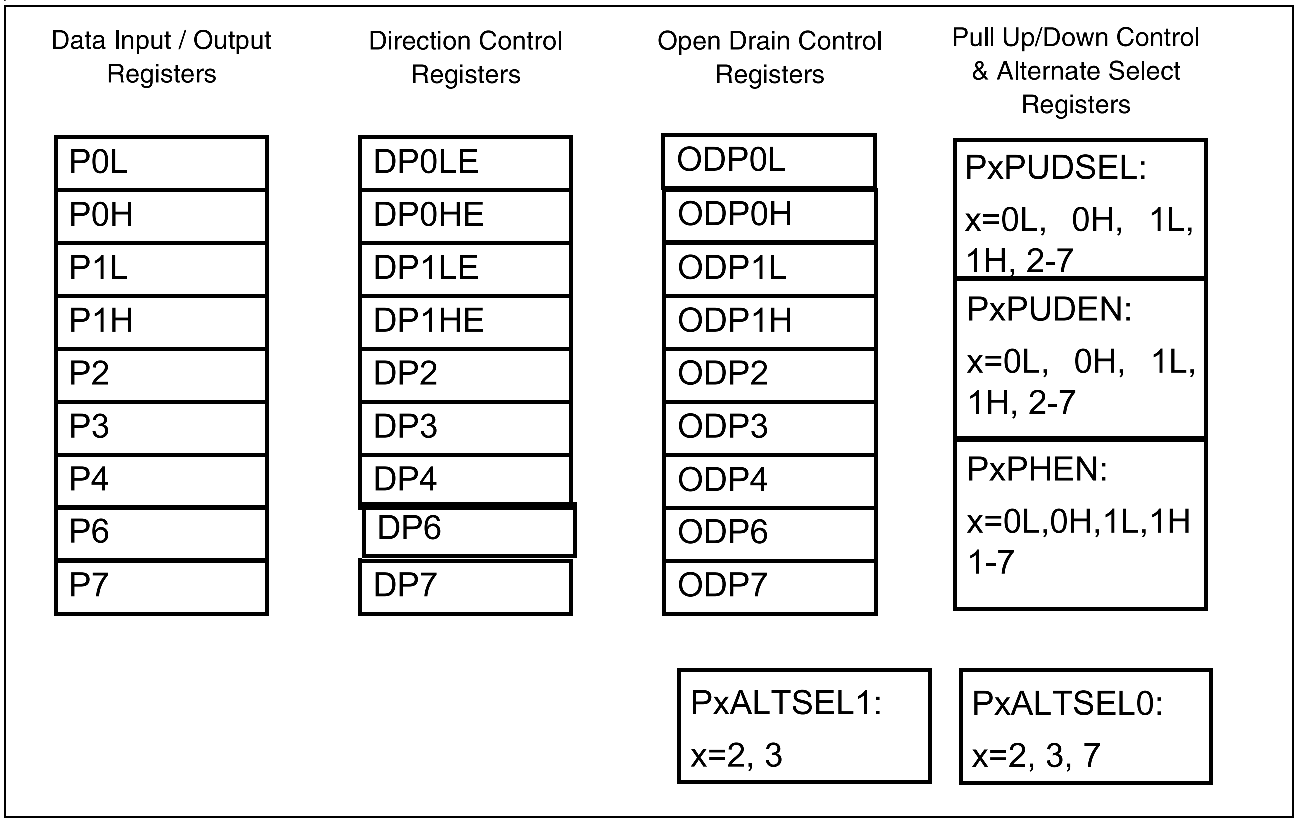
The logic level of a pin is clocked into the input latch once per state time, regardless whether the port is configured for input or output.

A write operation to a port pin configured as an input (DPx.y = '0') causes the value to be written into the port output latch, while a read operation returns the latched state of the pin itself. A read-modify-write operation reads the value of the pin, modifies it, and writes it back to the output latch.

Writing to a pin configured as an output (DPx.y='1') causes the output latch and the pin to have the written value, since the output buffer is enabled. Reading this pin returns the value of the output latch. A read-modify-write operation reads the value of the output latch, modifies it, and writes it back to the output latch, thus also modifying the level at the pin.

General Remark for all register descriptions:

Unused register bits always have an undefined reset value. The reset value for a whole register in hexadecimal notation does not apply to unused bits. Unused bits may be '0', '1', or '-'. Only if indicated with '-' , a bit can be written as '1' or '0'; in all other cases the predefined value must be written.



Data Input / Output Registers	Direction Control Registers	Open Drain Control Registers	Pull Up/Down Control & Alternate Select Registers
P0L	DP0LE	ODP0L	PxPUDSEL: x=0L, 0H, 1L, 1H, 2-7
P0H	DP0HE	ODP0H	
P1L	DP1LE	ODP1L	PxPUDEN: x=0L, 0H, 1L, 1H, 2-7
P1H	DP1HE	ODP1H	
P2	DP2	ODP2	
P3	DP3	ODP3	PxPHEN: x=0L,0H,1L,1H 1-7
P4	DP4	ODP4	
P6	DP6	ODP6	
P7	DP7	ODP7	
		PxALTSEL1: x=2, 3	PxALTSEL0: x=2, 3, 7

Figure 9-1 SFRs and Pins associated with the Parallel Ports

Output Driver Modes

In the INCA-D the ports provide Open Drain Control, which allows to switch the output driver of a port pin from a push/pull configuration to an open drain configuration. In push/pull mode a port output driver has an upper and a lower transistor, thus it can actively drive the line either to a high or a low level.

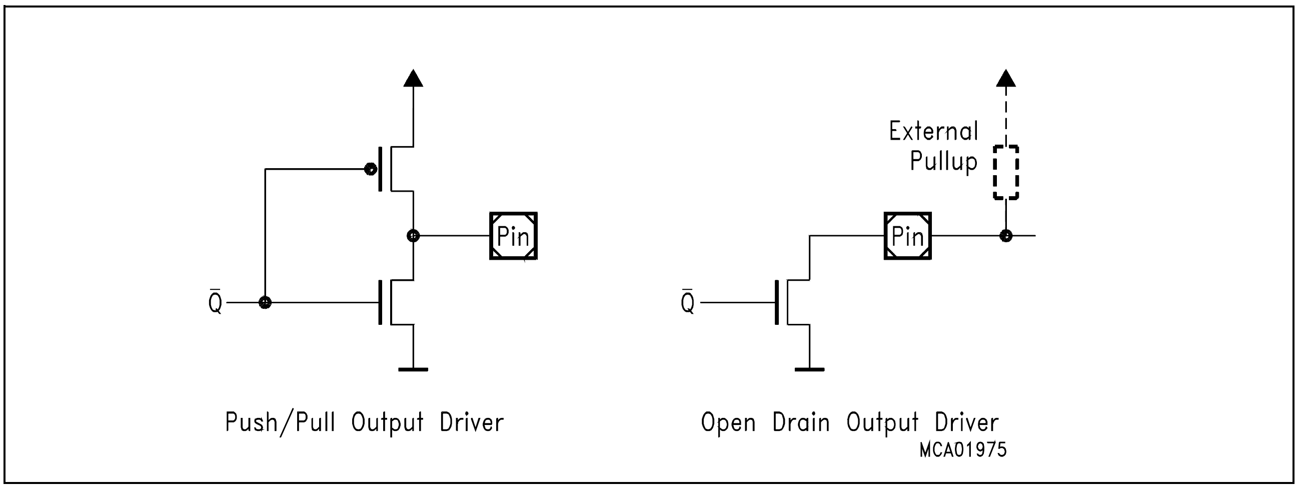


Figure 9-2 Output Drivers in Push/Pull Mode and in Open Drain Mode

In open drain mode the upper transistor is always switched off, and the output driver can only actively drive the line to a low level. When writing a ‘1’ to the port latch, the lower transistor is switched off and the output enters a high-impedance state. The high level can then be provided by using an internal pull up transisitor or by an external pull up device. With this feature, it is possible to connect several port pins together to a Wired-AND configuration, saving external glue logic and/or additional software overhead for enabling/disabling output signals.

This last feature is controlled through the respective Open Drain Control Registers ODPx. These registers allow the individual bit-wise selection of the open drain mode for each port line. If the respective control bit ODPx.y is ‘0’ (default after reset), the output driver is in the push/pull mode. If ODPx.y is ‘1’, the open drain configuration is selected. Note that all ODPx registers are located in the ESFR space.

The output driver is disabled in power down mode unless PxPHEN.y = ’1’.

Alternate Port Functions

Beside the use as general purpose I/Os, each port line has one or more programmable alternate input or output function associated.

Whether a port pin should operate as general purpose I/O or as alternate function, is determined by setting the corresonding bit in the PxALTSEL0 register, if the alternate function is not controlled by the corresponding peripheral itself.

If a port pin has a second alternate function which has to be selected per software, the bit of the PxALTSEL1 register has to be set accordingly.

Note: If two or more alternate functions are enabled concurrently, the behaviour is not predictable, but the device won’t be damaged.

On most of the port lines, the user software is responsible for setting the proper direction when using an alternate input or output function of a pin. This is done by setting or clearing the direction control bit DPx.y of the pin before enabling the alternate function. There are port lines, however, where the direction of the port line is switched automatically. For instance, in the multiplexed external bus modes of PORT0, the direction must be switched several times for an instruction fetch in order to output the addresses and to input the data. Obviously, this cannot be done through instructions. In these cases, the direction of the port line is switched automatically by hardware if the alternate function of such a pin is enabled.

Note: In this case, make sure DP0 ’ ’0’..

All port lines that are not used for these alternate functions may be used as general purpose I/O lines. When using port pins for general purpose output, the initial output value should be written to the port latch prior to enabling the output drivers, in order to avoid undesired transitions on the output pins. This applies to single pins as well as to pin groups (see examples below).

```
OUTPUT_ENABLE_SINGLE_PIN:
BSET    P4.0                    ;Initial output level is 'high'
BSET    DP4.0                   ;Switch on the output driver
OUTPUT_ENABLE_PIN_GROUP:
BFLDL   P4, #05H, #05H          ;Initial output level is 'high'
BFLDL   DP4, #05H, #05H         ;Switch on the output drivers
```

Note: When using several BSET pairs to control more pins of one port, these pairs must be separated by instructions, which do not reference the respective port (see "Particular Pipeline Effects" in chapter "The Central Processing Unit").

Each of these ports and the alternate input and output functions are described in detail in the following subsections. However, the port structure is similar as described in **Figure 9-3** for port 3.

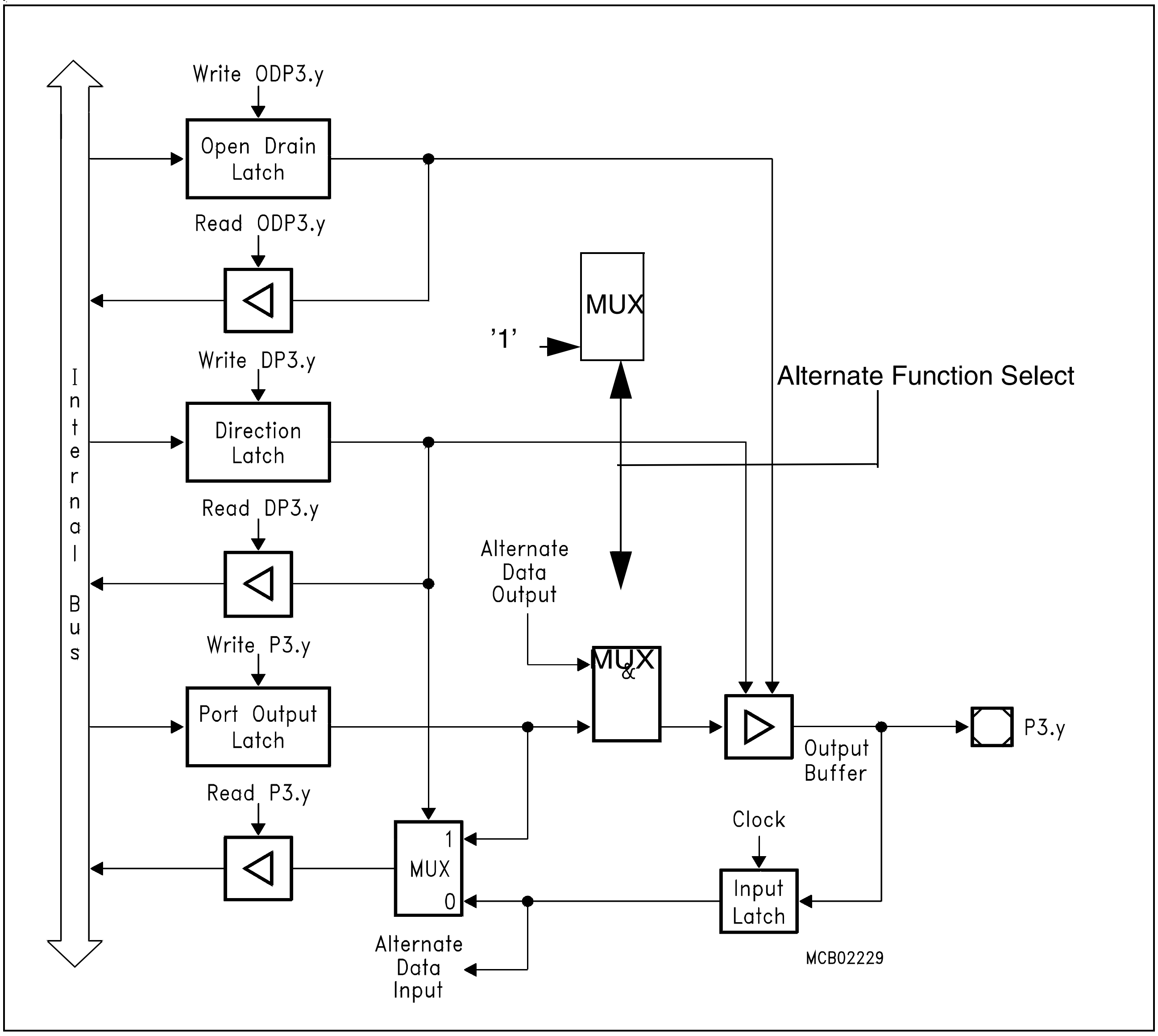


Figure 9-3 Port structure

9.1 PORT0

The two 8-bit ports P0H and P0L represent the higher and lower part of PORT0, respectively. Both halfs of PORT0 can be written (eg. via a PEC transfer) without effecting the other half.

If this port is used for general purpose IO, the direction of each line can be configured via the corresponding direction registers DP0H and DP0L.

Each port line of P0L and P0H can be switched into push/pull or open drain mode via the open drain control register ODP0H.

An internal pull transistor is connected to the pad if bits of register P0xPUDEN = ’1’, no matter whether the INCA-D is in normal operation mode or in power down mode. Either pulldown transistor or pullup transistor will be selected via P0xPUDSEL.

The output driver is disabled in power down mode unless P0xPHEN = ’1’.

After reset all bits of P0xPUDEN and P0xPUDSEL are set to ’1’ except P0LPUDSEL.6 and P0HPUDSEL.2.

While this feature allows to start without any external pull device, the configuration may be overwritten by external pull devices. In this case the internal pull resistors should be disabled by software after reset.

P0L (FF00$_H$ / 80$_H$) SFR **Reset Value: - - 00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0L.7	**P0L.6**	**P0L.5**	**P0L.4**	**P0L.3**	**P0L.2**	**P0L.1**	**P0L.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

P0H (FF02$_H$ / 81$_H$) SFR **Reset Value: - - 00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0H.7	**P0H.6**	**P0H.5**	**P0H.4**	**P0H.3**	**P0H.2**	**P0H.1**	**P0H.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	**Function**
P0X.y	**Port data register P0H or P0L bit y**

DP0L (F100$_H$ / 80$_H$) **ESFR** **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP0L .7	**DP0L .6**	**DP0L .5**	**DP0L .4**	**DP0L .3**	**DP0L .2**	**DP0L .1**	**DP0L .0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

DP0H (F102$_H$ / 81$_H$) **ESFR** **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP0H .7	**DP0H .6**	**DP0H .5**	**DP0H .4**	**DP0H .3**	**DP0H .2**	**DP0H .1**	**DP0H .0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	**Function**
DP0X.y	Port direction register DP0H or DP0L bit y DP0X.y = 0: Port line P0X.y is an input (high-impedance) DP0X.y = 1: Port line P0X.y is an output

ODP0L (FE20$_H$ / 10$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ODP0 L.7	**ODP0 L.6**	**ODP0 L.5**	**ODP0 L.4**	**ODP0 L.3**	**ODP0 L.2**	**ODP0 L.1**	**ODP0 L.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

ODP0H (FE22$_H$ / 11$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ODP0 H.7	**ODP0 H.6**	**ODP0 H.5**	**ODP0 H.4**	**ODP0 H.3**	**ODP0 H.2**	**ODP0 H.1**	**ODP0 H.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
ODP0X.y	**Port0XOpen Drain control register bit y** ODP0X.y = 0: Port line P0X.y output driver in push/pull mode ODP0X.y = 1: Port line P0X.y output driver in open drain mode

P0LPUDSEL (FE60$_H$ / 30$_H$) SFR **Reset Value: - -3F[1)]$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0L PUD SEL.7	**P0L PUD SEL.6**	**P0L PUD SEL.5**	**P0L PUD SEL.4**	**P0L PUD SEL.3**	**P0L PUD SEL.2**	**P0L PUD SEL.1**	**P0L PUD SEL.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

P0HPUDSEL (FE62$_H$ / 31$_H$) SFR **Reset Value: - - E3$_H$[1)]**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0H PUD SEL.7	**P0H PUD SEL.6**	**P0H PUD SEL.5**	**P0H PUD SEL.4**	**P0H PUD SEL.3**	**P0H PUD SEL.2**	**P0H PUD SEL.1**	**P0H PUD SEL.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P0xPUDSEL.y	Pulldown/Pullup Selection P0xPUDSEL.y = 0: internal programmable pulldown transistor is selected P0xPUDSEL.y = 1: internal programmable pullup transistor is selected

1) The reset value determines also the memory bus configuration after reset. For details refer to Chapter 24.8

P0LPUDEN (FE64$_H$ / 32$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0L PUD EN.7	**P0L PUD EN.6**	**P0L PUD EN.5**	**P0L PUD EN.4**	**P0L PUD EN.3**	**P0L PUD EN.2**	**P0L PUD EN.1**	**P0L PUD EN.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

P0HPUDEN (FE66$_H$ / 33$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0H PUD EN.7	**P0H PUD EN.6**	**P0H PUD EN.5**	**P0H PUD EN.4**	**P0H PUD EN.3**	**P0H PUD EN.2**	**P0H PUD EN.1**	**P0H PUD EN.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	**Function**
P0xPUDEN.y	Pulldown/Pullup Enable P0xPUDEN.y = 0: internal programmable pull transistor is disabled P0xPUDEN.y = 1: internal programmable pull transistor is enabled

P0LPHEN (FE68$_H$ / 34$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0L PHEN .7	**P0L PHEN .6**	**P0L PHEN .5**	**P0L PHEN .4**	**P0L PHEN .3**	**P0L PHEN .2**	**P0L PHEN .1**	**P0L PHEN .0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

P0HPHEN (FE6A$_H$ / 35$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P0H PHEN .7	**P0H PHEN .6**	**P0H PHEN .5**	**P0H PHEN .4**	**P0H PHEN .3**	**P0H PHEN .2**	**P0H PHEN .1**	**P0H PHEN .0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	**Function**
P0xPHEN.y	Output Driver Enable in Power Down Mode P0xPHEN.y = 0: output driver is disabled in power down mode P0xPHEN.y = 1: output driver is enabled in power down mode

9.1.1 Alternate Functions of PORT0

For external memory access PORT0 is used as data bus or address/data bus.
Note that an external 8-bit demultiplexed bus only uses P0L, while P0H is free for I/O (provided that no other bus mode is enabled).

PORT0 is also used to select the system startup configuration. During reset, PORT0 is configured to input, and each line is held high through an internal pullup device. Each line can now be individually pulled to a low level (see DC-level specifications in the respective Data Sheets) through an external pulldown device. A default configuration is selected when the respective PORT0 lines are at a high level. Through pulling individual lines to a low level, this default can be changed according to the needs of the applications.

The internal pullup devices are designed such that an external pulldown resistors (see specification) can be used to apply a correct low level. These external pulldown resistors can remain connected to the PORT0 pins also during normal operation, however, care has to be taken such that they do not disturb the normal function of PORT0 (this might be the case, for example, if the external resistor is too strong).

With the end of reset, the selected bus configuration will be written to the BUSCON0 register. The configuration of the high byte of PORT0, will be copied into the special register RP0H. This read-only register holds the selection for the number of chip selects and segment addresses. Software can read this register in order to react according to the selected configuration, if required.

Note: When the reset is terminated, the internal pullup devices must be switched off by Software using bit BCLR of register P0(L/H)PUDEN.x, and PORT0 will be switched to the appropriate operating mode.

During external accesses in multiplexed bus modes PORT0 first outputs the 16-bit intra-segment address as an alternate output function. PORT0 is then switched to high-impedance input mode to read the incoming instruction or data. In 8-bit data bus mode, two memory cycles are required for word accesses, the first for the low byte and the second for the high byte of the word. During write cycles PORT0 outputs the data byte or word after outputting the address.

During external accesses in demultiplexed bus modes PORT0 reads the incoming instruction or data word or outputs the data byte or word.

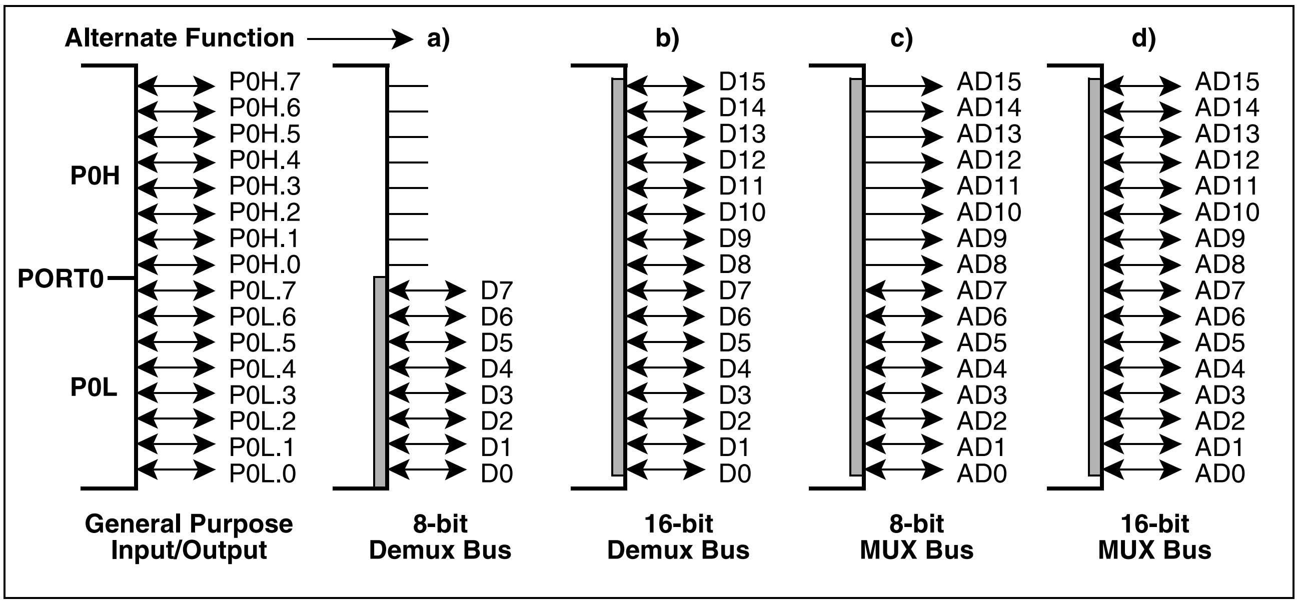

Alternate Function
a)
b)
c)
d)
P0H
PORT0
P0L
P0H.7
P0H.6
P0H.5
P0H.4
P0H.3
P0H.2
P0H.1
P0H.0
P0L.7
P0L.6
P0L.5
P0L.4
P0L.3
P0L.2
P0L.1
P0L.0
D7 D6 D5 D4 D3 D2 D1 D0
D15 D14 D13 D12 D11 D10 D9 D8 D7 D6 D5 D4 D3 D2 D1 D0
AD15 AD14 AD13 AD12 AD11 AD10 AD9 AD8 AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0
AD15 AD14 AD13 AD12 AD11 AD10 AD9 AD8 AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0
General Purpose Input/Output
8-bit Demux Bus
16-bit Demux Bus
8-bit MUX Bus
16-bit MUX Bus

Figure 9-4 PORT0 I/O and Alternate Functions

For external memory access the direction of the port pin and the loading of data into the port output latch are controlled by the bus controller hardware. The input of the port output latch is disconnected from the internal bus and is switched to the line labeled “Alternate Data Output” via a multiplexer. The alternate data can be the 16-bit intrasegment address or the 8/16-bit data information. The incoming data on PORT0 is read on the line “Alternate Data Input”. The user software should not write to the port output latch, otherwise unpredictable results may occur. **Figure 9-4** shows the structure of a PORT0 pin.

9.2 PORT1

The two 8-bit ports P1H and P1L represent the higher and lower part of PORT1, respectively. Both halfs of PORT1 can be written (eg. via a PEC transfer) without effecting the other half.

If this port is used for general purpose IO, the direction of each line can be configured via the corresponding direction registers DP1H and DP1L.

Each port line can be switched into push/pull or open drain mode via the open drain control register ODP1L and ODP1H.

An internal pull transistor is connected to the pad if bits of register P1xPUDEN.y = ’1’, no matter whether the INCA-D is in normal operation mode or in power down mode. Either pulldown transistor or pullup transistor will be selected via P1xPUDSEL.

The output driver is disabled in power down mode unless P1xPHEN = ’1’.

After reset, the bits of P1xPUDEN and P1xPUDSEL are set to ’1’.

P1L (FF04$_H$ / 82$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1L.7	**P1L.6**	**P1L.5**	**P1L.4**	**P1L.3**	**P1L.2**	**P1L.1**	**P1L.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

P1H (FF06$_H$ / 83$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1H.7	**P1H.6**	**P1H.5**	**P1H.4**	**P1H.3**	**P1H.2**	**P1H.1**	**P1H.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	**Function**
P1X.y	**Port data register P1H or P1L bit y**

DP1L (F104$_H$ / 82$_H$) **ESFR** **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP1L .7	**DP1L .6**	**DP1L .5**	**DP1L .4**	**DP1L .3**	**DP1L .2**	**DP1L .1**	**DP1L .0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

DP1H (F106$_H$ / 83$_H$) **ESFR** **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								DP1H .7	**DP1H .6**	**DP1H .5**	**DP1H .4**	**DP1H .3**	**DP1H .2**	**DP1H .1**	**DP1H .0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	**Function**
DP1X.y	**Port direction register DP1H or DP1L bit y** DP1X.y = 0: Port line P1X.y is an input (high-impedance) DP1X.y = 1: Port line P1X.y is an output

ODP1L (FE24$_H$ / 12$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ODP1 L.7	**ODP1 L.6**	**ODP1 L.5**	**ODP1 L.4**	**ODP1 L.3**	**ODP1 L.2**	**ODP1 L.1**	**ODP1 L.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

ODP1H (FE26$_H$ / 13$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								ODP1 H.7	**ODP1 H.6**	**ODP1 H.5**	**ODP1 H.4**	**ODP1 H.3**	**ODP1 H.2**	**ODP1 H.1**	**ODP1 H.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
ODP1x.y	**Port1x Open Drain control register bit y** ODP1x.y = 0: Port line P1x.y output driver in push/pull mode ODP1x.y = 1: Port line P1x.y output driver in open drain mode

P1LPUDSEL (FE6C$_H$ / 36$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1L PUD SEL.7	**P1L PUD SEL.6**	**P1L PUD SEL.5**	**P1L PUD SEL.4**	**P1L PUD SEL.3**	**P1L PUD SEL.2**	**P1L PUD SEL.1**	**P1L PUD SEL.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

P1HPUDSEL (FE6E$_H$ / 37$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1H PUD SEL.7	**P1H PUD SEL.6**	**P1H PUD SEL.5**	**P1H PUD SEL.4**	**P1H PUD SEL.3**	**P1H PUD SEL.2**	**P1H PUD SEL.1**	**P1H PUD SEL.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P1xPUDSEL.y	Pulldown/Pullup Selection P1xPUDSEL.y = 0: internal programmable pulldown transistor is selected P1xPUDSEL.y = 1: internal programmable pullup transistor is selected

P1LPUDEN (FE70$_H$ / 38$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1L PUD EN.7	**P1L PUD EN.6**	**P1L PUD EN.5**	**P1L PUD EN.4**	**P1L PUD EN.3**	**P1L PUD EN.2**	**P1L PUD EN.1**	**P1L PUD EN.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

P1HPUDEN (FE72$_H$ / 39$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1H PUD EN.7	**P1H PUD EN.6**	**P1H PUD EN.5**	**P1H PUD EN.4**	**P1H PUD EN.3**	**P1H PUD EN.2**	**P1H PUD EN.1**	**P1H PUD EN.0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P1xPUDEN.y	Pulldown/Pullup Enable P1xPUDEN.y = 0: internal programmable pull transistor is disabled P1xPUDEN.y = 1: internal programmable pull transistor is enabled

P1LPHEN (FE74$_H$ / 3A$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1L PHEN .7	**P1L PHEN .6**	**P1L PHEN .5**	**P1L PHEN .4**	**P1L PHEN .3**	**P1L PHEN .2**	**P1L PHEN .1**	**P1L PHEN .0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

P1HPHEN (FE76$_H$ / 3B$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								P1H PHEN .7	**P1H PHEN .6**	**P1H PHEN .5**	**P1H PHEN .4**	**P1H PHEN .3**	**P1H PHEN .2**	**P1H PHEN .1**	**P1H PHEN .0**
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P1xPHEN.y	Output Driver Enable in Power Down Mode P1xPHEN.y = 0: output driver is disabled in power down mode P1xPHEN.y = 1: output driver is enabled in power down mode

9.2.1 Alternate Functions of PORT1

When a demultiplexed external bus is enabled, PORT1 is used as address bus.
Note that demultiplexed bus modes use PORT1 as a 16-bit port. Otherwise all 16 port lines can be used for general purpose IO.

During external accesses in demultiplexed bus modes PORT1 outputs the 16-bit intra-segment address as an alternate output function.

During external accesses in multiplexed bus modes, when **no** BUSCON register selects a demultiplexed bus mode, PORT1 is not used and is available for general purpose IO.

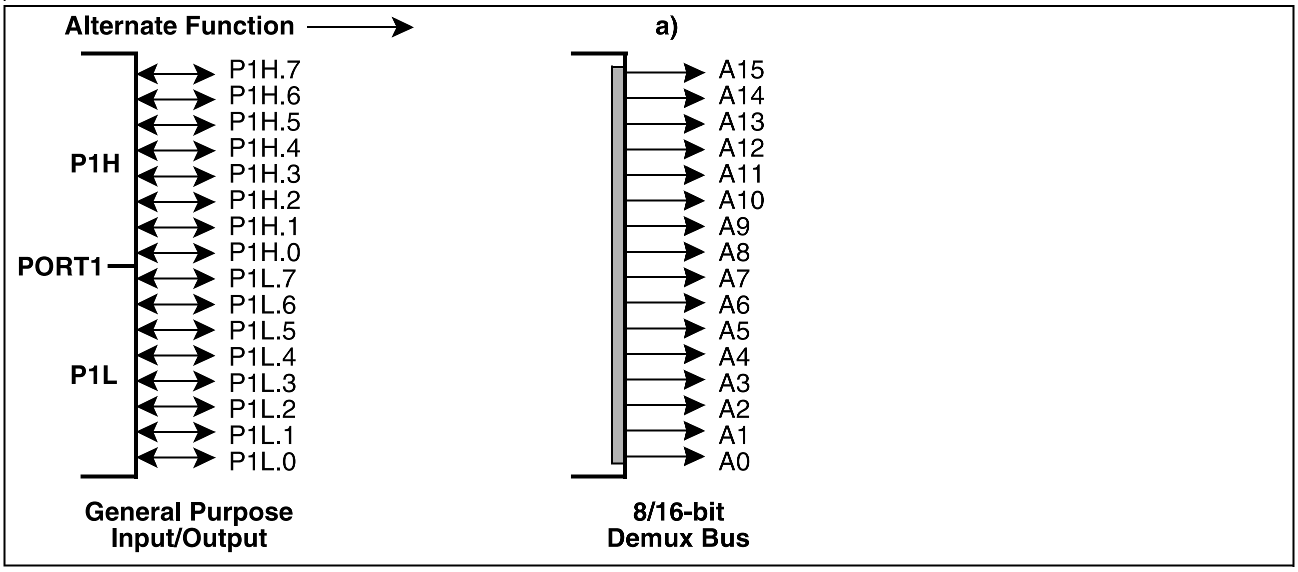


Figure 9-5 PORT1 I/O and Alternate Functions

For external memory access the direction of the port pin and the loading of data into the port output latch are controlled by the bus controller hardware. The input of the port output latch is disconnected from the internal bus and is switched to the line labeled “Alternate Data Output” via a multiplexer. The alternate data is the 16-bit intrasegment address. While an external bus mode is enabled, the user software should not write to the port output latch, otherwise unpredictable results may occur.

9.3 PORT2

The Port 2 of INCA-D is an 14-bit port. If Port 2 is used for general purpose I/O, the direction of each line can be configured via the corresponding direction register DP2. Each port line can be switched into push/pull or open drain mode via the open drain control register ODP2.

An internal pull transistor is connected to the pad if register P2PUDEN = ’1’, no matter whether the INCA-D is in normal operation mode or in power down mode. Either pulldown transistor or pullup transistor will be selected via P2PUDSEL.

The output driver is disabled in power down mode unless P2PHEN = ’1’.

After reset, bits of P2PUDEN and P2PUDSEL are set to ’1’.

P2 (FFC0$_H$ / E0$_H$) SFR **Reset Value: 00 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	P2.13	P2.12	P2.11	P2.10	P2.9	P2.8	P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P2.y	**Port data register P2 bit y**

DP2 (FFC2$_H$ / E1$_H$) SFR **Reset Value: 00 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	DP2 .13	DP2 .12	DP2 .11	DP2 .10	DP2 .9	DP2 .8	DP2 .7	DP2 .6	DP2 .5	DP2 .4	DP2 .3	DP2 .2	DP2 .1	DP2 .0
-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
DP2.y	**Port direction register DP2 bit y** DP2.y = 0: Port line P2.y is an input (high-impedance) DP2.y = 1: Port line P2.y is an output

ODP2 (F1C2$_H$ / E1$_H$) SFR **Reset Value00 00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	ODP2 .13	ODP2 .12	ODP2 .11	ODP2 .10	ODP2 .9	ODP2 .8	ODP2 .7	ODP2 .6	ODP2 .5	ODP2 .4	ODP2 .3	ODP2 .2	ODP2 .1	ODP2 .0
-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
ODP2.y	**Port 2 Open Drain control register bit y** ODP2.y = 0: Port line P2.y output driver in push/pull mode ODP2.y = 1: Port line P2.y output driver in open drain mode

P2PUDSEL (FE78$_H$ / 3C$_H$) SFR **Reset Value FFFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	-	P2 PUD SEL.13	P2 PUD SEL.12	P2 PUD SEL.11	P2 PUD SEL.10	P2 PUD SEL.9	P2 PUD SEL.8	P2 PUD SEL.7	P2 PUD SEL.6	P2 PUD SEL.5	P2 PUD SEL.4	P2 PUD SEL.3	P2 PUD SEL.2	P2 PUD SEL.1	P2 PUD SEL.0
-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P2PUDSEL.y	Pulldown/Pullup Selection P2PUDSEL.y = 0: internal programmable pulldown transistor is selected P2PUDSEL.y = 1: internal programmable pullup transistor is selected

P2PUDEN (FE7A$_H$ / 3D$_H$) SFR **Reset Value: FFFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	P2 PUD EN.13	P2 PUD EN.12	P2 PUD EN.11	P2 PUD EN.10	P2 PUD EN.9	P2 PUD EN.8	P2 PUD EN.7	P2 PUD EN.6	P2 PUD EN.5	P2 PUD EN.4	P2 PUD EN.3	P2 PUD EN.2	P2 PUD EN.1	P2 PUD EN.0
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P2PUDEN.y	Pulldown/Pullup Enable P2PUDEN.y = 0: internal programmable pull transistor is disabled P2PUDEN.y = 1: internal programmable pull transistor is enabled

P2PHEN (FE7C$_H$ / 3E$_H$) SFR **Reset Value: 00 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	P2 PHEN .13	P2 PHEN .12	P2 PHEN .11	P2 PHEN .10	P2 PHEN .9	P2 PHEN .8	P2 PHEN .7	P2 PHEN .6	P2 PHEN .5	P2 PHEN .4	P2 PHEN .3	P2 PHEN 2	P2 PHEN EN-	P2 PHEN .0
		rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P2PHEN.y	Output Driver Enable in Power Down Mode P2PHEN.y = 0: output driver is disabled in power down mode P2PHEN.y = 1: output driver is enabled in power down mode

P2ALTSEL0 (F122$_H$ / 91$_H$) ESFR **Reset Value: 00 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	P2 ALT SEL0. 13	P2 ALT SEL0. 12	P2 ALT SEL0. 11	P2 ALT SEL0. 10	P2 ALT SEL0.9	P2 ALT SEL0.8	P2 ALT SEL0.7	P2 ALT SEL0.6	P2 ALT SEL0.5	P2 ALT SEL0.4	P2 ALT SEL0.3	P2 ALT SEL0.2	P2 ALT SEL0.1	P2 ALT SEL0.0
-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function

P2ALTSEL0.y	Alternate Function 0 Selection
	P2ALTSEL0.y = 0: no alternate function 0 selected
	P2ALTSEL0.y = 1: alternate function 0 selected

P2ALTSEL1 (F124$_H$ / 92$_H$) ESFR **Reset Value : --00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	P2 ALT SEL1.7	P2 ALT SEL1.6	P2 ALT SEL1.5	P2 ALT SEL1.4	P2 ALT SEL1.3	P2 ALT SEL1.2	-	P2 ALT SEL1.0
-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	-	-

Bit	Function

P2ALTSEL1.y	Alternate Function 1Selection
	P2ALTSEL1.y = 0: no alternate function 1 selected
	P2ALTSEL1.y = 1: alternate function 1 selected

9.3.1 Alternate Functions of PORT2

The first 3 lines of Port 2 (P2.2..P2.0) serve as Fast External Interrupt inputs (EX2IN...EX0IN) while P2.0 and P2.1 serve also as *Serial Data Strobe* outputs . The remaining port lines can be used as GPI/O or to multiplex the connected LED field.

Table 9-1 summarizes the alternate functions of Port 2.

Table 9-1 Port 2 Alternate Functions

Port 2 Pin	Alternate Function	
P2.0	SDS1/FEX0IN	Serial Data Strobe 1 or Fast External Interrupt 0
P2.1	SDS2/FEX1IN	Serial Data Strobe 2 or Fast External Interrupt 1
P2.2	FEX2IN	Fast External Interrupt 2
P2.3	LEDMUX(0)	Output of LED multiplex unit
P2.4	LEDMUX(1)	Output of LED multiplex unit
P2.5	LEDMUX(2)	Output of LED multiplex unit
P2.6	LEDMUX(3)	Output of LED multiplex unit
P2.7	LEDMUX(4)	Output of LED multiplex unit
P2.8	LEDMUX(5)	Output of LED multiplex unit
P2.9	LEDMUX(6)	Output of LED multiplex unit
P2.10	LEDMUX(7)	Output of LED multiplex unit
P2.11	LEDMUX(8)	Output of LED multiplex unit
P2.12	LEDMUX(9)	Output of LED multiplex unit
P2.13	LEDMUX(10)	Output of LED multiplex unit

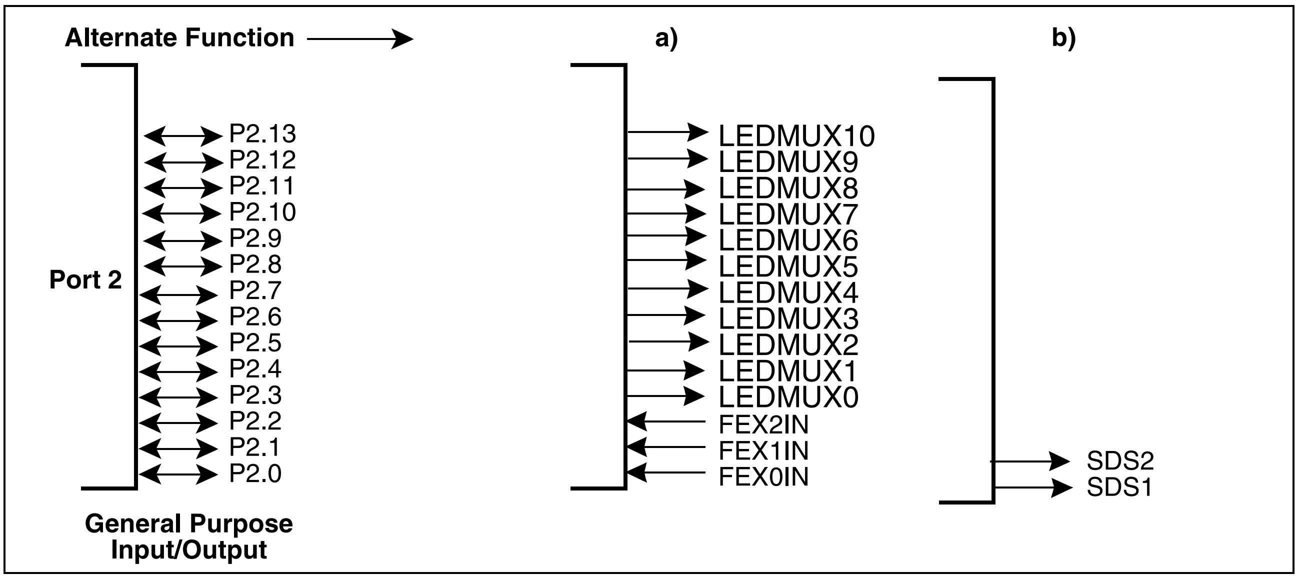


Figure 9-6 Port 2 I/O and Alternate Functions

9.4 PORT3

If this 16-bit port is used for general purpose I/O, the direction of each line can be configured via the corresponding direction register DP3. All port lines of P3 can be switched into push/pull or open drain mode via the open drain control register ODP3.

All port lines support internal pull transistors, which are connected to the pad if register P3PUDEN = ’1’, no matter whether the INCA-D is in normal operation mode or in power down mode. Either pulldown transistor or pullup transistor will be selected via P3PUDSEL.

The output driver is disabled in power down mode unless P3PHEN = ’1’. After reset, bits of P3PUDEN and P3PUDSEL are set to ’1’.

P3 (FFC4$_H$ / E2$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P3.15	**P3.14**	**P3.13**	**P3.12**	**P3.11**	**P3.10**	**P3.9**	**P3.8**	**P3.7**	**P3.6**	**P3.5**	**P3.4**	**P3.3**	**P3.2**	**P3.1**	**P3.0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P3.y	**Port data register P3 bit y**

DP3 (FFC6$_H$ / E3$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DP3 .15	**DP3. 14**	**DP3 .13**	**DP3 .12**	**DP3 .11**	**DP3 .10**	**DP3 .9**	**DP3 .8**	**DP3 .7**	**DP3 .6**	**DP3 .5**	**DP3 .4**	**DP3 .3**	**DP3 .2**	**DP3 .1**	**DP3 .0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
DP3.y	**Port direction register DP3 bit y** DP3.y = 0: Port line P3.y is an input (high-impedance) DP3.y = 1: Port line P3.y is an output

ODP3 (F1C6$_H$ / E3$_H$) **ESFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODP3 .15	**ODP3 .14**	**ODP3 .13**	**ODP3 .12**	**ODP3 .11**	**ODP3 .10**	**ODP3 .9**	**ODP3 .8**	**ODP3 .7**	**ODP3 .6**	**ODP3 .5**	**ODP3 .4**	**ODP3 .3**	**ODP3 .2**	**ODP3 .1**	**-**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
ODP3.y	**Port 3 Open Drain control register bit y** ODP3.y = 0: Port line P3.y output driver in push/pull mode ODP3.y = 1: Port line P3.y output driver in open drain mode

P3PUDSEL (FE7E$_H$ / 3F$_H$) SFR **Reset Value: FFFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P3PU DSEL. 15	P3PU DSEL. 14	P3PU DSEL. 13	P3PU DSEL. 12	P3PU DSEL. 11	P3PU DSEL. 10	P3PU DSEL. 9	P3PU DSEL. 8	P3PU DSEL. 7	P3PU DSEL. 6	P3PU DSEL. 5	P3PU DSEL. 4	P3PU DSEL. 3	P3PU DSEL. 2	P3PU DSEL. 1	-
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P3PUDSEL.y	Pulldown/Pullup Selection P3PUDSEL.y = 0: internal programmable pulldown transistor is selected P3PUDSEL.y = 1: internal programmable pullup transistor is selected

P3PUDEN (FE80$_H$ / 40$_H$) SFR **Reset Value: FF FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P3PU DEN. 15	P3PU DEN. 14	P3PU DEN. 13	P3PU DEN. 12	P3PU DEN. 11	P3PU DEN. 10	P3PU DEN. 9	P3PU DEN. 8	P3PU DEN. 7	P3PU DEN. 6	P3PU DEN. 5	P3PU DEN. 4	P3PU DEN. 3	P3PU DEN. 2	P3PU DEN. 1	-
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P3PUDEN.y	Pulldown/Pullup Enable P3PUDEN.y = 0: internal programmable pull transistor is disabled P3PUDEN.y = 1: internal programmable pull transistor is enabled

P3PHEN (FE82$_H$ / 41$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P3 PHEN .15	P3 PHEN .14	P3 PHEN .13	P3 PHEN .12	P3 PHEN .11	P3 PHEN .10	P3 PHEN .9	P3 PHEN .8	P3 PHEN .7	P3 PHEN .6	P3 PHEN .5	P3 PHEN .4	P3 PHEN .3	P3 PHEN .2	P3 PHEN .1	P3 PHEN .0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P3PHEN.y	Output Driver Enable in Power Down Mode P3PHEN.y = 0: output driver is disabled in power down mode P3PHEN.y = 1: output driver is enabled in power down mode

P3ALTSEL0 (F126$_H$ / 93$_H$) ESFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
P3 ALT SEL0. 15	P3 ALT SEL0. 14	P3 ALT SEL0. 13	P3 ALT SEL0. 12	P3 ALT SEL0. 11	P3 ALT SEL0	P3 ALT SEL0.9	P3 ALT SEL0.8	P3 ALT SEL0.7	P3 ALT SEL0.6	P3 ALT SEL0.5	P3 ALT SEL0.4	P3 ALT SEL0.3	P3 ALT SEL0.2	P3 ALT SEL0.1	P3 ALT SEL0.0
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function

P3ALTSEL0.y	Alternate Function 0 Selection
	P3ALTSEL0.y = 0: no alternate function 0 selected
	P3ALTSEL0.y = 1: alternate function 0 selected

P3ALTSEL1 (F128$_H$ / 94$_H$) ESFR **Reset Value: 00 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	P3 ALT SEL1. 13	-	-	-	P3 ALT SEL1.9	P3 ALT SEL1.8	-	-	-	-	-	P3 ALT SEL1.2	P3 ALT SEL1.1	P3 ALT SEL1.0
rw	-	rw	-	-	-	rw	rw	-	-	-	-	-	rw	rw	-

Bit	Function

P3ALTSEL1.y	Alternate Function 1 Selection
	P3ALTSEL.y = 0: no alternate function 1 selected
	P3ALTSEL.y = 1: alternate function 1 selected

9.4.1 Alternate Functions of PORT3

The pins of Port 3 serve for various functions. **Table 9-2** summarizes the alternate functions of Port 3.

Table 9-2 Alternate Functions of Port 3

Port 3 Pin	Alternate Function	
P3.0	MRST1/T5IN	SSC1 Master Receive / Slave Transmit / Timer 5 Input
P3.1	MTSR1/T4EUD	SSC1 Master Transm./ Slave Rec./Timer 4 External Up-Down Input
P3.2	SCLK1/T2EUD	SSC1 Shift Clock Input/Output / Timer 2 External up-down Input
P3.3	T3OUT	Timer 3 Toggle Output
P3.4	T3EUD	Timer 3 External Up/Down Control
P3.5	T4IN	Timer 4 Count Input
P3.6	T3IN	Timer 3 Count Input
P3.7	T2IN	Timer 2 Count Input
P3.8	MRST0/T6IN	SSC0 Master Receive / Slave Transmit / Timer 6 Input
P3.9	MTSR0/T5EUD	SSC0 Master Transm./ Slave Rec./Timer 5 External Up-Down Input
P3.10	TxD0	ASC Transmit Data Output
P3.11	RxD0	ASC Receive Data Input / Output
P3.12	$\overline{\text{BHE}}/\overline{\text{WRH}}$	Byte High Enable / Write High Output
P3.13	SCLK0/T6EUD	SSC0 Shift Clock Input/Output/ Timer 6 External up-down Input
P3.14	T6OUT	Timer 3 Toggle Output
P3.15	CAPIN	GPT2 Capture Input

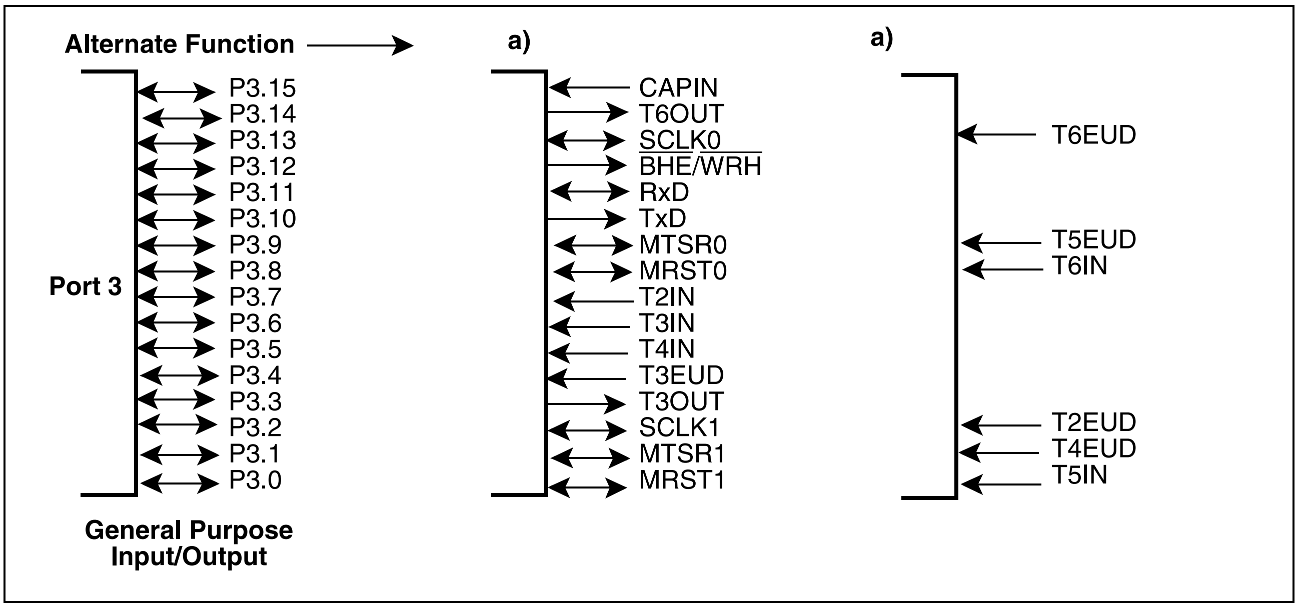
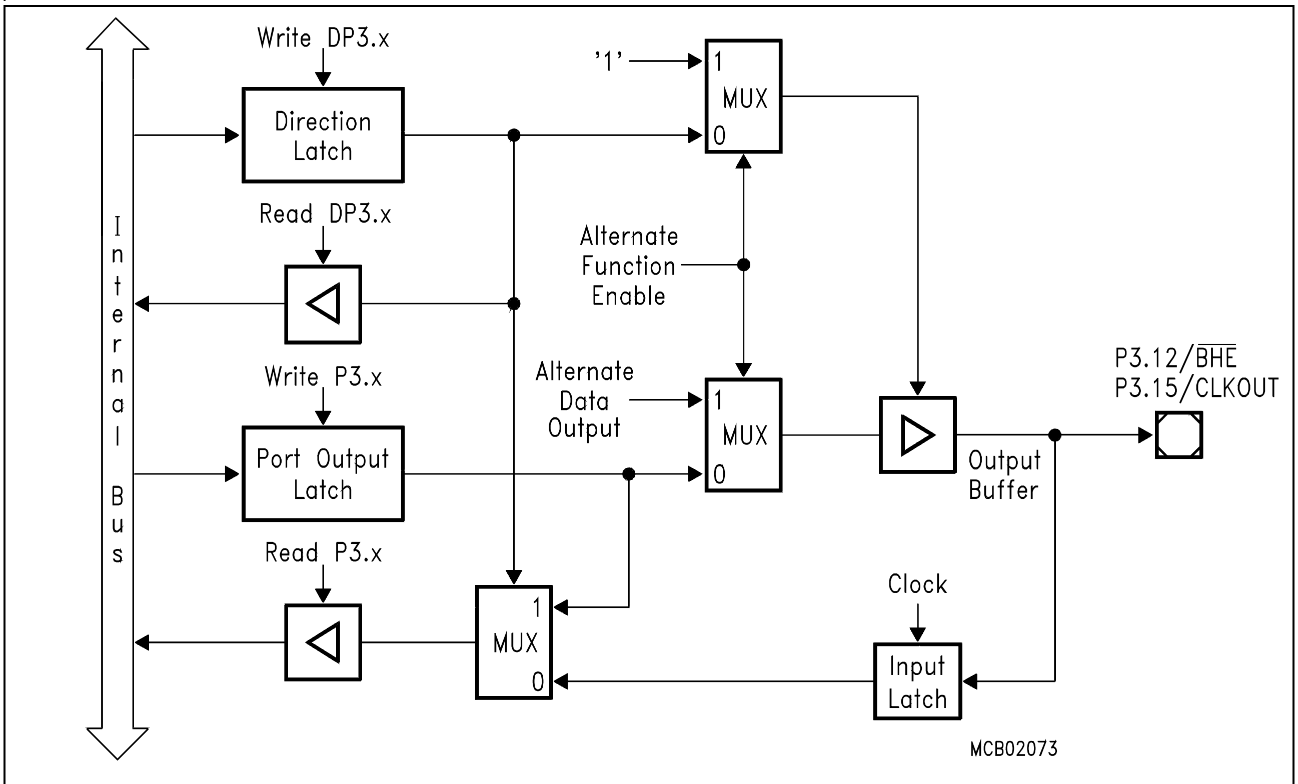


Figure 9-7 Port 3 I/O and Alternate Functions

When the on-chip peripheral associated with a Port 3 pin is configured to use the alternate input function, it reads the input latch, which represents the state of the pin, via the line labeled "Alternate Data Input".

When the on-chip peripheral associated with a Port 3 pin is configured to use both the alternate input and output function, the descriptions above apply to the respective curren

Pin P3.12 ($\overline{\text{BHE}}$/$\overline{\text{WRH}}$) is one more pin with an alternate output function. However, its structure is slightly different (see figure below), because after reset the $\overline{\text{BHE}}$ or $\overline{\text{WRH}}$ function must be used depending on the system startup configuration. In these cases there is no possibility to program any port latches before. Thus the appropriate alternate function is selected automatically. If $\overline{\text{BHE}}$/$\overline{\text{WRH}}$ is not used in the system, this pin can be used for general purpose I/O by disabling the alternate function (BYTDIS = '1' / WRCFG='0').



x = 15, 12

Figure 9-8 Block Diagram of Pins P3.15 (CLKOUT) and P3.12 ($\overline{\text{BHE}}$/$\overline{\text{WRH}}$)

Note: Enabling the $\overline{\text{BHE}}$ or $\overline{\text{WRH}}$ function automatically enables the P3.12 output driver. Setting bit DP3.12='1' is not required.

9.5 PORT4

If this 6-bit port is used for general purpose I/O, the direction of each line can be configured via the corresponding direction register DP4.

Each port line can be switched into push/pull or open drain mode via the open drain control register ODP4.

An internal pull transistor is connected to the pad if register P4PUDEN = ’1’, no matter whether the INCA-D is in normal operation mode or in power down mode. Either pulldown transistor or pullup transistor will be selected via P4PUDSEL.

The output driver is disabled in power down mode unless P4PHEN = ’1’.

After reset, all bits of P4PUDEN and P4PUDSEL are set to ’1’.

P4 (FFC8$_H$ / E4$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										P4.5	**P4.4**	**P4.3**	**P4.2**	**P4.1**	**P4.0**
-	-	-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw

Bit	Function
P4.y	**Port data register P4 bit y**

DP4 (FFCA$_H$ / E5$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										DP4.5	**DP4.4**	**DP4.3**	**DP4.2**	**DP4.1**	**DP4.0**
-	-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw

Bit	Function
DP4.y	**Port direction register DP4 bit y** DP4.y = 0: Port line P4.y is an input (high-impedance) DP4.y = 1: Port line P4.y is an output

ODP4 (F1CA$_H$ / E5$_H$) **ESFR** **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										ODP2 .5	**ODP2 .4**	**ODP2 .3**	**ODP2 .2**	**ODP2 .1**	**ODP2 .0**
-	-	-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw

Bit	Function
ODP4.y	**Port 4 Open Drain control register bit y** ODP4.y = 0: Port line P4.y output driver in push/pull mode ODP4.y = 1: Port line P4.y output driver in open drain mode

P4PUDSEL (FE84$_H$ / 42$_H$) SFR **Reset Value: - -FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										P4 PUD SEL.5	**P4 PUD SEL.4**	**P4 PUD SEL.3**	**P4 PUD SEL.2**	**P4 PUD SEL.1**	**P4 PUD SEL.0**
-	-	-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw

Bit	Function
P4PUDSEL.y	Pulldown/Pullup Selection P4PUDSEL.y = 0: internal programmable pulldown transistor is selected P4PUDSEL.y = 1: internal programmable pullup transistor is selected

P4PUDEN (FE86$_H$ / 43$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										P4 PUD EN.5	**P4 PUD EN.4**	**P4 PUD EN.3**	**P4 PUD EN.2**	**P4 PUD EN.1**	**P4 PUD EN.0**
-	-	-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw

Bit	Function
P4PUDEN.y	Pulldown/Pullup Enable P4PUDEN.y = 0: internal programmable pull transistor is disabled P4PUDEN.y = 1: internal programmable pull transistor is enabled

P4PHEN (FE88$_H$ / 44$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
										P4 PHEN .5	**P4 PHEN .4**	**P4 PHEN .3**	**P4 PHEN .2**	**P4 PHEN .1**	**P4 PHEN .0**
-	-	-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw	rw

Bit	Function
P4PHEN.y	Output Driver Enable in Power Down Mode P4PHEN.y = 0: output driver is disabled in power down mode P4PHEN.y = 1: output driver is enabled in power down mode

9.5.1 Alternate Functions of PORT4

During external bus cycles that use segmentation (ie. an address space above 64 KByte) a number of Port 4 pins may output the segment address lines. The number of pins that is used for segment address output determines the external address space which is directly accessible. The other pins of Port 4 (if any) may be used for general purpose I/O. If segment address lines are selected, the alternate function of Port 4 may be necessary to access eg. external memory directly after reset. For this reason Port 4 will be switched to its alternate function automatically.

The number of segment address lines is selected via PORT0 during reset. The selected value can be read from bitfield SALSEL in register RP0H (read only) eg. in order to check the configuration during run time.

Table 9-3 summarizes the alternate functions of Port 4 depending on the number of selected segment address lines (coded via bitfield SALSEL).

Table 9-3 Alternate Functions of Port 4

Port 4 Pin	**Std. Function** SALSEL=01 64 KB	**Altern. Function** SALSEL=11 256KB	**Altern. Function** SALSEL=00 1 MB	**Altern. Function** SALSEL=10 4 MB
P4.0	Gen. purpose I/O	Seg. Address A16	Seg. Address A16	Seg. Address A16
P4.1	Gen. purpose I/O	Seg. Address A17	Seg. Address A17	Seg. Address A17
P4.2	Gen. purpose I/O	Gen. purpose I/O	Seg. Address A18	Seg. Address A18
P4.3	Gen. purpose I/O	Gen. purpose I/O	Seg. Address A19	Seg. Address A19
P4.4	Gen. purpose I/O	Gen. purpose I/O	Gen. purpose I/O	Seg. Address A20
P4.5	Gen. purpose I/O	Gen. purpose I/O	Gen. purpose I/O	Seg. Address A21

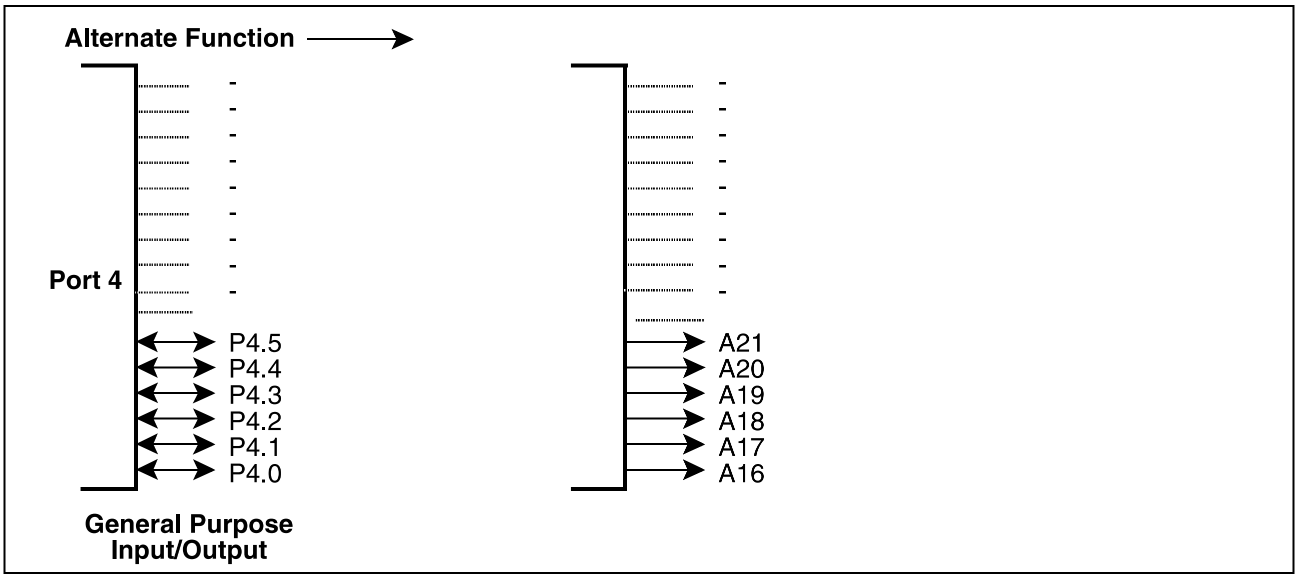


Figure 9-9 Port 4 I/O and Alternate Functions

9.6 PORT6

If this 3-bit port is used for general purpose I/O, the direction of each line can be configured via the corresponding direction register DP6. Each port line can be switched into push/pull or open drain mode via the open drain control register ODP6.

An internal pull transistor is connected to the pad if register P6PUDEN = ’1’, no matter whether the INCA-D is in normal operation mode or in power down mode. Either pulldown transistor or pullup transistor will be selected via P6PUDSEL.

The output driver is disabled in power down mode unless P6PHEN = ’1’. After reset, all bits of P6PUDEN and P6PUDSEL are set to ’1’.

P6 (FFCC$_H$ / E6$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													P6.2	**P6.1**	**P6.0**
-	-	-	-	-	-	-	-	-	-	-	-	-	rw	rw	rw

Bit	Function
P6.y	**Port data register P6 bit y**

DP6 (FFCE$_H$ / E7$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													DP6.2	**DP6.1**	**DP6.0**
-	-	-	-	-	-	-	-	-	-	-	-	-	rw	rw	rw

Bit	Function
DP6.y	**Port direction register DP6 bit y** DP6.y = 0: Port line P6.y is an input (high-impedance) DP6.y = 1: Port line P6.y is an output

ODP6 (F1CE$_H$ / E7$_H$) **ESFR** **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													ODP6 .2	**ODP6 .1**	**ODP6 .0**
-	-	-	-	-	-	-	-	-	-	-	-	-	rw	rw	rw

Bit	Function
ODP6.y	**Port 6 Open Drain control register bit y** ODP6.y = 0: Port line P6.y output driver in push/pull mode ODP6.y = 1: Port line P6.y output driver in open drain mode

P6PUDSEL (FE90$_H$ / 48$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													P6 PUD SEL.2	**P6 PUD SEL.1**	**P6 PUD SEL.0**
-	-	-	-	-	-	-	-	-	-	-	-	-	rw	rw	rw

Bit	Function
P6PUDSEL.y	Pulldown/Pullup Selection P6PUDSEL.y = 0: internal programmable pulldown transistor is selected P6PUDSEL.y = 1: internal programmable pullup transistor is selected

P6PUDEN (FE92$_H$ / 49$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													P6 PUD EN.2	**P6 PUD EN.1**	**P6 PUD EN.0**
-	-	-	-	-	-	-	-	-	-	-	-	-	rw	rw	rw

Bit	Function
P6PUDEN.y	Pulldown/Pullup Enable P6PUDEN.y = 0: internal programmable pull transistor is disabled P6PUDEN.y = 1: internal programmable pull transistor is enabled

P6PHEN (FE94$_H$ / 4A$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													P6 PHEN .2	**P6 PHEN .1**	**P6 PHEN .0**
-	-	-	-	-	-	-	-	-	-	-	-	-	rw	rw	rw

Bit	Function
P6PHEN.y	Output Driver Enable in Power Down Mode P6PHEN.y = 0: output driver is disabled in power down mode P6PHEN.y = 1: output driver is enabled in power down mode

P6ALTSEL0 (F12C$_H$ / 96$_H$) ESFR **Reset Value: - -00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
													P6 ALT SEL0.2	**P6 ALT SEL0.1**	**P6 ALT SEL0.0**
-	-	-	-	-	-	-	-	-	-	-	rw	rw	rw	rw	rw

Note: If the memory interface has been configured accordingly, the selection of $\overline{CS0}$ is done by the External Bus Controller automatically.

Bit	Function

P6ALTSEL0.y	Alternate Function 0 Selection
	P6ALTSEL.y = 0: no alternate function 0 selected
	P6ALTSELy = 1: alternate function 0 selected

9.6.1 Alternate Functions of PORT6

The three chip select signals (CS2..CS0) derived from the bus control registers (BUSCON2...BUSCON0) can be output on 3 pins of Port 6. The number of chip select signals is selected via PORT0 during reset. The selected value can be read from bitfield CSSEL in register RP0H (read only) eg. in order to check the configuration during run time.

Table 9-4 summarizes the alternate functions of Port 6 depending on the number of selected chip select lines (coded via bitfield CSSEL). For information about the coding of 0, 1, or 2 chip-select lines please refer to **Table 24-3** on page 24-603.

Table 9-4 Alternate Functions of Port 6

Port 6 Pin	Altern. Function
P6.0	Chip select $\overline{CS0}$
P6.1	Chip select $\overline{CS1}$
P6.2	Chip select $\overline{CS2}$

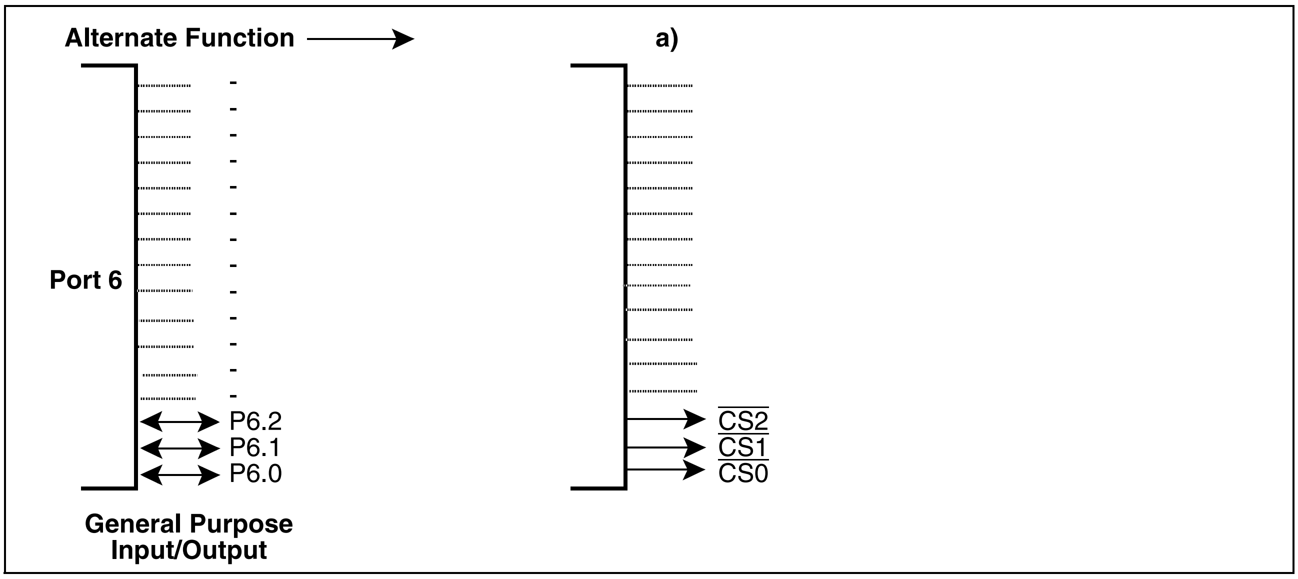


Figure 9-10 Port 6 I/O and Alternate Functions

9.7 PORT7

In the INCA-D Port 10 is an 8-bit general purpose I/O port. The direction of each line can be configured via the corresponding direction register DP7. Each port line can be switched into push/pull or open drain mode via the open drain control register ODP7.

An internal pull transistor is connected to the pad if register P7PUDEN = ’1’, no matter whether the INCA-D is in normal operation mode or in power down mode. Either pulldown transistor or pullup transistor will be selected via P7PUDSEL.

The output driver is disabled in power down mode unless P7PHEN = ’1’. After reset, all bits of P7PUDEN and P7PUDSEL are set to ’1’.

P7 (FFD0$_H$ / E8$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						P7.9	**P7.8**	**P7.7**	**P7.6**	**P7.5**	**P7.4**	**P7.3**	**P7.2**	**P7.1**	**P7.0**
-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P7.y	**Port data register P7 bit y**

DP7 (FFD2$_H$ / E9$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						DP7.9	**DP7.8**	**DP7.7**	**DP7.6**	**DP7.5**	**DP7.4**	**DP7.3**	**DP7.2**	**DP7.1**	**DP7.0**
-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
DP7.y	**Port direction register DP7 bit y** DP7.y = 0: Port line P7.y is an input (high-impedance) DP7.y = 1: Port line P7.y is an output

ODP7 (F1D2$_H$ / E9$_H$) **ESFR** **Reset Value: - - 00**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						ODP7 .9	**ODP7 .8**	**ODP7 .7**	**ODP7 .6**	**ODP7 .5**	**ODP7 .4**	**ODP7 .3**	**ODP7 .2**	**ODP7 .1**	**ODP7 .0**
-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
ODP7.y	**Port 7 Open Drain control register bit y** ODP7.y = 0: Port line P7.y output driver in push/pull mode ODP7.y = 1: Port line P7.y output driver in open drain mode

P7PUDSEL (FE96$_H$ / 4B$_H$) SFR **Reset Value: - -FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						P7 PUD SEL.9	P7 PUD SEL.8	P7 PUD SEL.7	P7 PUD SEL.6	P7 PUD SEL.5	P7 PUD SEL.4	P7 PUD SEL.3	P7 PUD SEL.2	P7 PUD SEL.1	P7 PUD SEL.0
-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P7PUDSEL.y	Pulldown/Pullup Selection P7PUDSEL.y = 0: internal programmable pulldown transistor is selected P7PUDSEL.y = 1: internal programmable pullup transistor is selected

P7PUDEN (FE98$_H$ / 4C$_H$) SFR **Reset Value: - - FF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						P7 PUD EN.9	P7 PUD EN.8	P7 PUD EN.7	P7 PUD EN.6	P7 PUD EN.5	P7 PUD EN.4	P7 PUD EN.3	P7 PUD EN.2	P7 PUD EN.1	P7 PUD EN.0
-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P7PUDEN.y	Pulldown/Pullup Enable P7PUDEN.y = 0: internal programmable pull transistor is disabled P7PUDEN.y = 1: internal programmable pull transistor is enabled

P7PHEN (FE9A$_H$ / 4D$_H$) SFR **Reset Value: - - 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						P7 PHEN .9	P7 PHEN .8	P7 PHEN .7	P7 PHEN .6	P7 PHEN .5	P7 PHEN .4	P7 PHEN .3	P7 PHEN .2	P7 PHEN .1	P7 PHEN .0
-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
P7PHEN.y	Output Driver Enable in Power Down Mode P7PHEN.y = 0: output driver is disabled in power down mode P7PHEN.y = 1: output driver is enabled in power down mode

P7ALTSEL0 (F12EA$_H$ / 97$_H$) ESFR **Reset Value: - -00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
						P7 ALT SEL0.9	P7 ALT SEL0.8	P7 ALT SEL0.7	P7 ALT SEL0.6	P7 ALT SEL0.5	P7 ALT SEL0.4	P7 ALT SEL0.3	P7 ALT SEL0.2	P7 ALT SEL0.1	P7 ALT SEL0.0
-	-	-	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function

P7ALTSEL0.y	Alternate Function 0 Selection
	P7ALTSEL.y = 0: no alternate function 0 selected
	P7ALTSELy = 1: alternate function 0 selected

9.7.1 Alternate Functions of PORT7

P7(9:0) can be used for key scan lines 0 to 9. **Table 9-5** summarizes the alternate functions of Port 7.

Table 9-5 Alternate Functions of Port 7

Port 7 Pin	Altern. Function
P7.0	Keyscan of line 0
P7.1	Keyscan of line 1
P7.2	Keyscan of line 2
P7.3	Keyscan of line 3
P7.4	Keyscan of line 4
P7.5	Keyscan of line 5
P7.6	Keyscan of line 6
P7.7	Keyscan of line 7
P7.8	Keyscan of line 8
P7.9	Keyscan of line 9

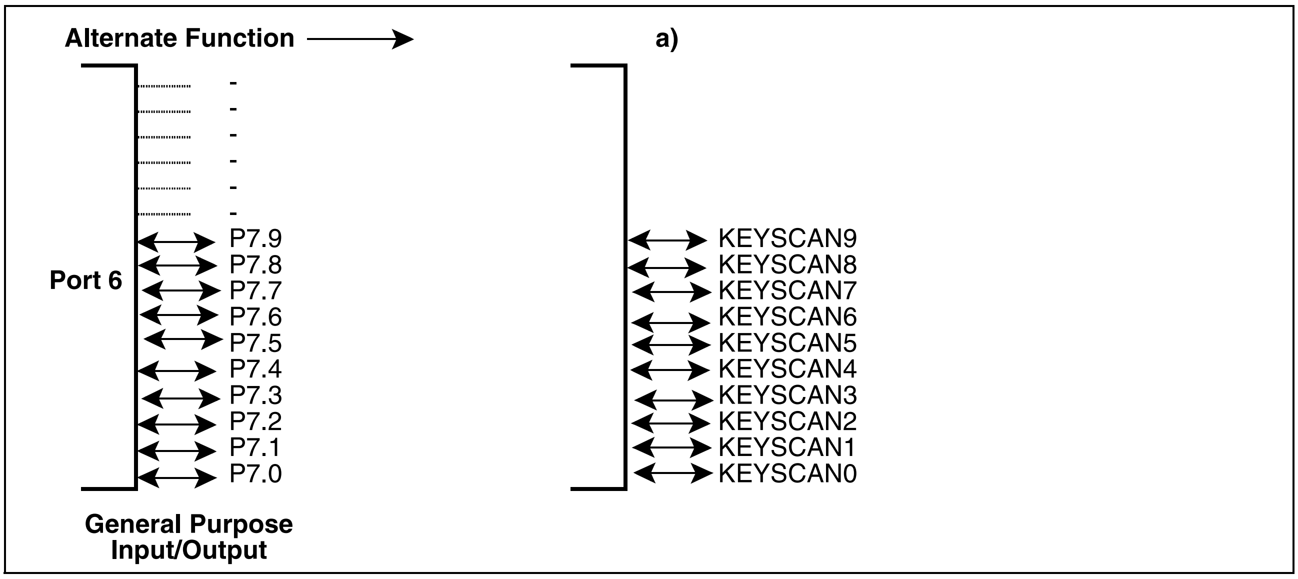


Figure 9-11 Port 7 I/O and Alternate Functions

10 Dedicated Pins

Most of the input/output or control signals of the functional the INCA-D are realized as alternate functions of pins of the parallel ports. There is, however, a number of signals that use separate pins, including the USB interface, the IOM-2 interface, the oscillator, special control signals and the power supply. **Table 10-1** summarizes the dedicated pins of the INCA-D.

Table 10-1 Dedicated Pins

Pin(s)	Function
VREF	Reference Voltage
BGREF	Bandgap Reference Voltage
MIP1/MIN1	Differential Mircrophone 1 Inputs
MIP2/MIN2	Differential Mircrophone 2 Inputs
MIP3/MIN3	Differential Mircrophone 3Inputs
HOP1/HON1	Differential earpiece 1 Outputs
HOP2/HON2	Differential earpiece 2 Output
LSP/LSN	Differential Loudspeaker Outputs
ALE	Address Latch Enable
$\overline{\text{RD}}$	External Read Strobe
$\overline{\text{WR}}$/$\overline{\text{WRL}}$	External Write/Write Low Strobe
LIA	Line Interface
LIB	Line Interface
$\overline{\text{NMI}}$	Non-Maskable Interrupt Input
$\overline{\text{RSTIN}}$	Reset Input
$\overline{\text{RSTOUT}}$	Reset Output
XTAL1, XTAL2	Oscillator Input/Output
DMNS, DPLS	USB
DU, DD, DCL, FSC	IOM-2 interface
BCL	IOM-2 bit clock
$\overline{\text{BRKIN}}$, $\overline{\text{BRKOUT}}$	OCDS Break In, Break out
TDI, TDO, TCK, TMS, TRST	JTAG Interface
TEST	Test Mode Enable (JTAG)
VDD, VSS	Power Supply and Ground (10 pins each)

The Reference Voltage VREF can be used for biasing external components. The DC voltage at VREF is the same as at the analog in-/output pins of the analog front-end (if enabled).

The Bandgap Reference Voltage BGREF (1.2 V) is used for internal references. The pin BGREF is necessary to connect a filter capacitor (approx. 22nF), which forms a RC-filter together with an on-chip resistor of about 300kΩ.

Symmetrical differential Microphone Inputs1,2 and 3 **MIN1/MIN2/MIN3 and MIP1/MIP2/ MIP3.**

Differential Handset earpiece output for 200 Ω transducers HOPx/HONx

Differential Loudspeaker output for 20 Ω loudspeaker LSP/LSN

The Address Latch Enable signal ALE controls external address latches that provide a stable address in multiplexed bus modes. **ALE is activated** for every external bus cycle independent of the selected bus mode, ie. it is also activated for bus cycles with a demultiplexed address bus. **ALE is not activated** for internal accesses, ie. the internal RAM and the special function registers.

The External Read Strobe $\overline{\textbf{RD}}$ controls the output drivers of external memory or peripherals when the INCA-D reads data from these external devices. During reset an internal pullup ensures an inactive (high) level on the $\overline{\text{RD}}$ output.

The External Write Strobe $\overline{\textbf{WR}}$/$\overline{\textbf{WRL}}$ controls the data transfer from the INCA-D to an external memory or peripheral device. This pin may either provide an general $\overline{\text{WR}}$ signal activated for both byte and word write accesses, or specifically control the low byte of an external 16-bit device ($\overline{\text{WRL}}$) together with the signal $\overline{\text{WRH}}$ (alternate function of P3.12/ $\overline{\text{BHE}}$). During reset an internal pullup ensures an inactive (high) level on the $\overline{\text{WR}}$/$\overline{\text{WRL}}$ output.

*Note: Whether **$\overline{\textbf{RD}}$** and **$\overline{\textbf{WR}}$/$\overline{\textbf{WRL}}$** remain idle during X-peripheral accesses depends on the value of bit VISIBLE of register SYSCON.*

The **line interface** consists of the two lines **LIa** and **LIb**.

The **IOM2-Interface** consists of the four lines **DCL, FSC, DD, and DU.** DCL is the 1.536MHz bit-clock (double-bit clock). FSC the 8kHz frame synchronization signal. DD and DU serve as serial data out-/input, they require external pull-up resistors. The double bit-clock at pin DCL can always be output as single-bit-clock (divided by 2) at pin **BCL**.

The **JTAG and OCDS Interface** offers an IEEE1149.1 compliant JTAG interface consisting of the lines **TCK**, **TDI**, **TDO**, **TMS**, **$\overline{\textbf{TRST}}$**. The JTAG TAP controller allows to use the JTAG interface for purposes of on-chip debugging. In this OCDS mode, additional lines **$\overline{\textbf{BRKIN}}$** and **$\overline{\textbf{BRKOUT}}$** are provided for controlling the behaviour of the CPU. The pin **TEST** is part of the JTAG interface and required for production test of the INCA-D. If unused, all pins of the JTAG/OCDS interface can remain unconnected.

The Non-Maskable Interrupt Input $\overline{\textbf{NMI}}$ allows to trigger a high priority trap via an external signal (eg. a power-fail signal). The $\overline{\text{NMI}}$ pin is sampled with every CPU clock cycle to detect transitions.

The Oscillator Input XTAL1 and Output XTAL2 connect the internal Pierce oscillator to the external crystal. An external clock signal may be fed to the input XTAL1, leaving XTAL2 open.

The **Universal Serial Bus** (USB) interface consists of the two lines **DMNS** and **DPLS**.

Events inside the **On Chip Debugging Interface** (OCDS) can be triggered by **BRKIN**. One of the possible actions after event detection is setting pin **BRKOUT**.

The Reset Input $\overline{\textbf{RSTIN}}$ allows to put the INCA-D into the well defined reset condition either at power-up or external events like a hardware failure or manual reset. The input voltage threshold of the $\overline{\text{RSTIN}}$ pin is raised compared to the standard pins in order to minimize the noise sensitivity of the reset input.

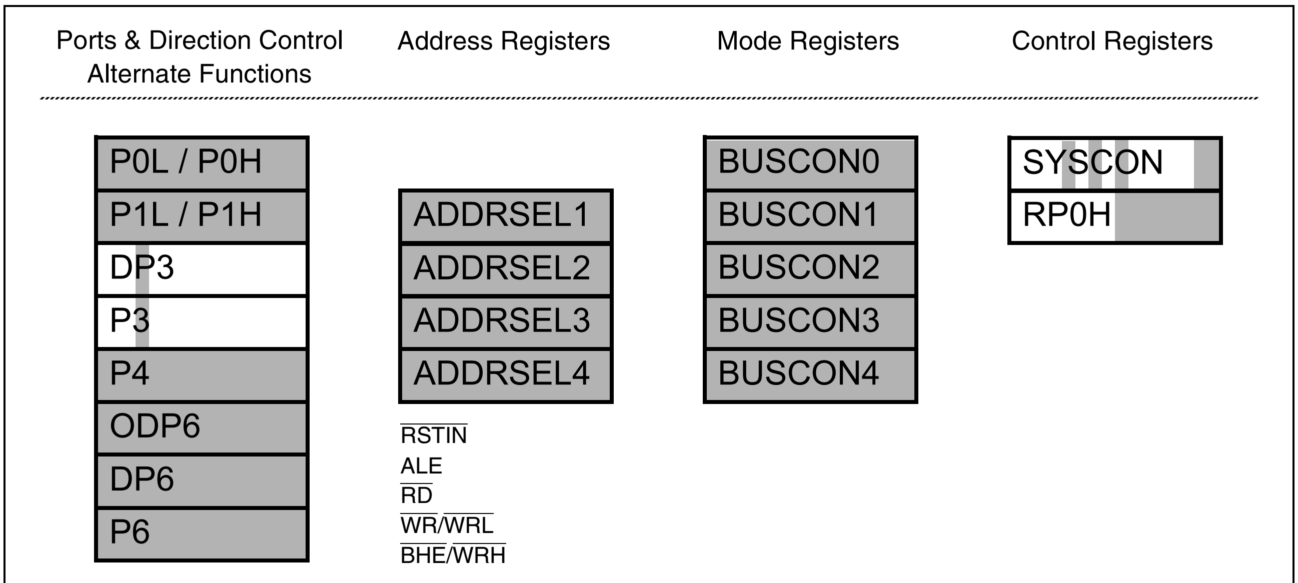
The Reset Output $\overline{\textbf{RSTOUT}}$ provides a special reset signal for external circuitry. $\overline{\text{RSTOUT}}$ is activated at the beginning of the reset sequence, triggered via $\overline{\text{RSTIN}}$, a watchdog timer overflow or by the SRST instruction. $\overline{\text{RSTOUT}}$ remains active (low) until the EINIT instruction is executed. This allows to initialize the controller before the external circuitry is activated.

The Power Supply pins provide the power supply for the digital logic of the INCA-D. The respective VCC/VSS pairs should be decoupled as close to the pins as possible. For best results it is recommended to implement two-level decoupling, eg. (the widely used) 100 nF in parallel with 30...40 pF capacitors which deliver the peak currents.

Note: All VDD pins and all VSS pins must be connected to the power supply and ground, respectively.

11 The External Bus Interface

The external bus interface allows to access external peripherals and additional volatile and non-volatile memory. The external bus interface provides a number of configurations, so it can be taylored to fit perfectly into a given application system.



P0L/P0H	PORT0 Data Registers	ADDRSELx	Address Range Select Register 1...4
P1L/P1H	PORT1 Data Registers	BUSCONx	Bus Mode Control Register 0...4
DP3	Port 3 Direction Control Register	SYSCON	System Control Register
P3	Port 3 Data Register	RP0H	Port P0H Reset Configuration Register
P4	Port 4 Data Register		
ODP6	Port 6 Open Drain Control Register		
DP6	Port 6 Direction Control Register		
P6	Port 6 Data Register		

Figure 11-1 SFRs and Port Pins Associated with the External Bus Interface

Accesses to external memory or peripherals are executed by the integrated External Bus Controller (EBC). The function of the EBC is controlled via the SYSCON register and the BUSCONx and ADDRSELx registers. The BUSCONx registers specify the external bus cycles in terms of address (mux/demux), data (16-bit/8-bit), chip selects and length (waitstates/ ALE / RW delay). These parameters are used for accesses within a specific address area which is defined via the corresponding register ADDRSELx.

The four pairs BUSCON1/ADDRSEL1...BUSCON4/ADDRSEL4 allow to define four independent “address windows”, while all external accesses outside these windows are controlled via register BUSCON0.

11.1 External Bus Modes

When the external bus interface is enabled (bit BUSACTx='1') and configured (bitfield BTYP), the INCA-D uses a subset of its port lines together with some control lines to build the external bus.

BTYP Encoding	External Data Bus Width	External Address Bus Mode
0 0	8-bit Data	Demultiplexed Addresses
0 1	8-bit Data	Multiplexed Addresses
1 0	16-bit Data	Demultiplexed Addresses
1 1	16-bit Data	Multiplexed Addresses

The bus configuration (BTYP) for the address windows (BUSCON4...BUSCON1) is selected via software typically during the initialization of the system.

The bus configuration (BTYP) for the default address range (BUSCON0) is selected via PORT0 during reset, otherwise BUSCON0 may be programmed via software just like the other BUSCON registers.

The address space of the INCA-D is divided into segments of 64 KByte each. The 16-bit intra-segment address is output on PORT0 for multiplexed bus modes or on PORT1 for demultiplexed bus modes. When segmentation is disabled, only one 64 KByte segment is available. Because of the occupied 8 KBytes of memory in segment 0, only 56 can be used and accessed. Otherwise additional address lines may be output on Port 4, and/or several chip select lines may be used to select different memory banks or peripherals. These functions are selected during reset via bitfields SALSEL and CSSEL of register RP0H, respectively.

Note: Bit SGTDIS of register SYSCON defines, if the CSP register is saved during interrupt entry (segmentation active) or not (segmentation disabled).

Multiplexed Bus Modes

In the multiplexed bus modes the 16-bit intra-segment address as well as the data use PORT0. The address is time-multiplexed with the data and has to be latched externally. The width of the required latch depends on the selected data bus width, ie. an 8-bit data bus requires a byte latch (the address bits A15...A8 on P0H do not change, while P0L multiplexes address and data), a 16-bit data bus requires a word latch (the least significant address line A0 is not relevant for word accesses). The upper address lines (A21...A16) are permanently output on Port 4 (if segmentation is enabled) and do not require latches.

The EBC initiates an external access by generating the Address Latch Enable signal (ALE) and then placing an address on the bus. The falling edge of ALE triggers an external latch to capture the address. After a period of time during which the address must have been latched externally, the address is removed from the bus. The EBC now activates the respective command signal ($\overline{RD}$, $\overline{WR}$, $\overline{WRL}$, $\overline{WRH}$). Data is driven onto the bus either by the EBC (for write cycles) or by the external memory/peripheral (for read cycles). After a period of time, which is determined by the access time of the memory/ peripheral, data become valid.

Read cycles: Input data is latched and the command signal is now deactivated. This causes the accessed device to remove its data from the bus which is then tri-stated again.

Write cycles: The command signal is now deactivated. The data remain valid on the bus until the next external bus cycle is started.

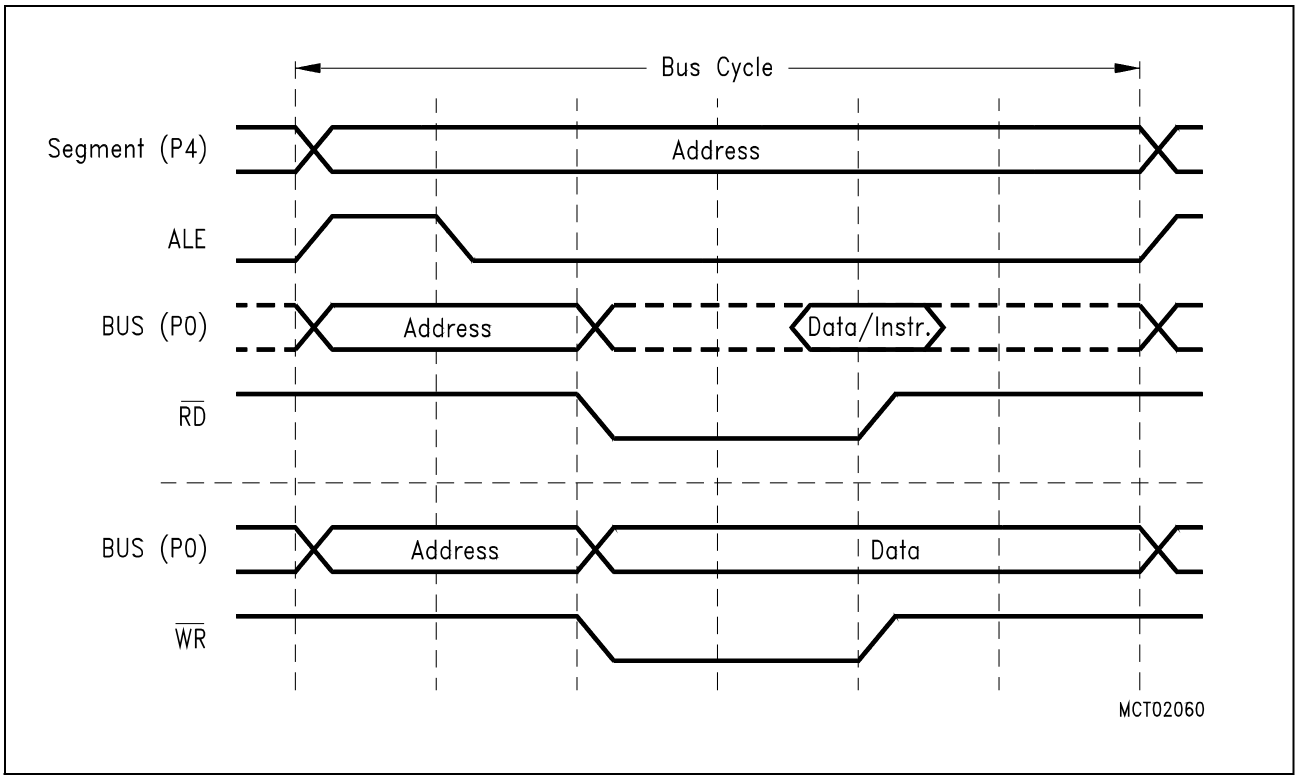


Figure 11-2 Multiplexed Bus Cycle

The bus cycle is described in number of TCL’s, where f_{CPU} = 1/(2 TCL).

Demultiplexed Bus Modes

In the demultiplexed bus modes the 16-bit intra-segment address is permanently output on PORT1, while the data uses PORT0 (16-bit data) or P0L (8-bit data).
The upper address lines are permanently output on Port 4 (if selected via SALSEL during reset). No address latches are required.

The EBC initiates an external access by placing an address on the address bus. After a programmable period of time the EBC activates the respective command signal ($\overline{RD}$, $\overline{WR}$, $\overline{WRL}$, $\overline{WRH}$). Data is driven onto the data bus either by the EBC (for write cycles) or by the external memory/peripheral (for read cycles). After a period of time, which is determined by the access time of the memory/peripheral, data become valid.

Read cycles: Input data is latched and the command signal is now deactivated. This causes the accessed device to remove its data from the data bus which is then tri-stated again.

Write cycles: The command signal is now deactivated. If a subsequent external bus cycle is required, the EBC places the respective address on the address bus. The data remain valid on the bus until the next external bus cycle is started.

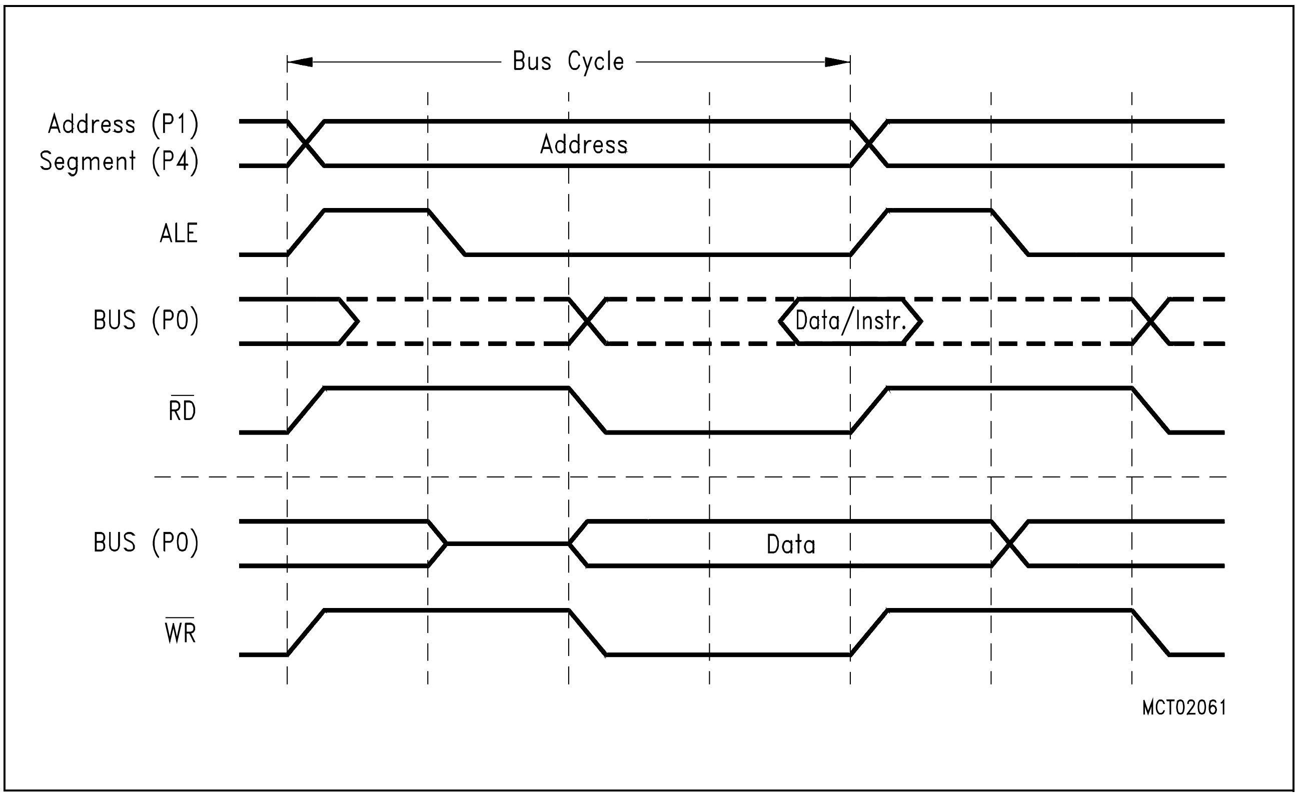


Figure 11-3 Demultiplexed Bus Cycle

The bus cycle is described in number of TCL's, where f_{CPU} = 1/(2 TCL).

Switching between the Bus Modes

The EBC allows to switch between different bus modes dynamically, ie. subsequent external bus cycles may be executed in different ways. Certain address areas may use multiplexed or demultiplexed buses or predefined waitstates.

A change of the external bus characteristics can be initiated in two different ways:

Reprogramming the BUSCON and/or ADDRSEL registers allows to either change the bus mode for a given address window, or change the size of an address window that uses a certain bus mode. Reprogramming allows to use a great number of different address windows (more than BUSCONs are available) on the expense of the overhead for changing the registers and keeping appropriate tables.

Switching between predefined address windows automatically selects the bus mode that is associated with the respective window. Predefined address windows allow to use different bus modes without any overhead, but restrict their number to the number of BUSCONs. However, as BUSCON0 controls all address areas, which are not covered by the other BUSCONs, this allows to have gaps between these windows, which use the bus mode of BUSCON0.

PORT1 will output the intra-segment address, when any of the BUSCON registers selects a demultiplexed bus mode, even if the current bus cycle uses a multiplexed bus mode. This allows to have an external address decoder connected to PORT1 only, while using it for all kinds of bus cycles.

Note: Never change the configuration for an address area that currently supplies the instruction stream. Due to the internal pipelining it is very difficult to determine the first instruction fetch that will use the new configuration. Only change the configuration for address areas that are not currently accessed. This applies to BUSCON registers as well as to ADDRSEL registers.

The usage of the BUSCON/ADDRSEL registers is controlled via the issued addresses. When an access (code fetch or data) is initiated, the respective generated physical address defines, if the access is made internally, uses one of the address windows defined by ADDRSEL4...1, or uses the default configuration in BUSCON0. After initializing the active registers, they are selected and evaluated automatically by interpreting the physical address. No additional switching or selecting is necessary during run time, except when more than the four address windows plus the default is to be used.

Switching from demultiplexed to multiplexed bus mode represents a special case. The bus cycle is started by activating ALE and driving the address to Port 4 and PORT1 as usual, if another BUSCON register selects a demultiplexed bus. However, in the multiplexed bus modes the address is also required on PORT0. In this special case the address on PORT0 is delayed by one CPU clock cycle, which delays the complete (multiplexed) bus cycle and extends the corresponding ALE signal (see figure below).

This extra time is required to allow the previously selected device (via demultiplexed bus) to release the data bus, which would be available in a demultiplexed bus cycle.

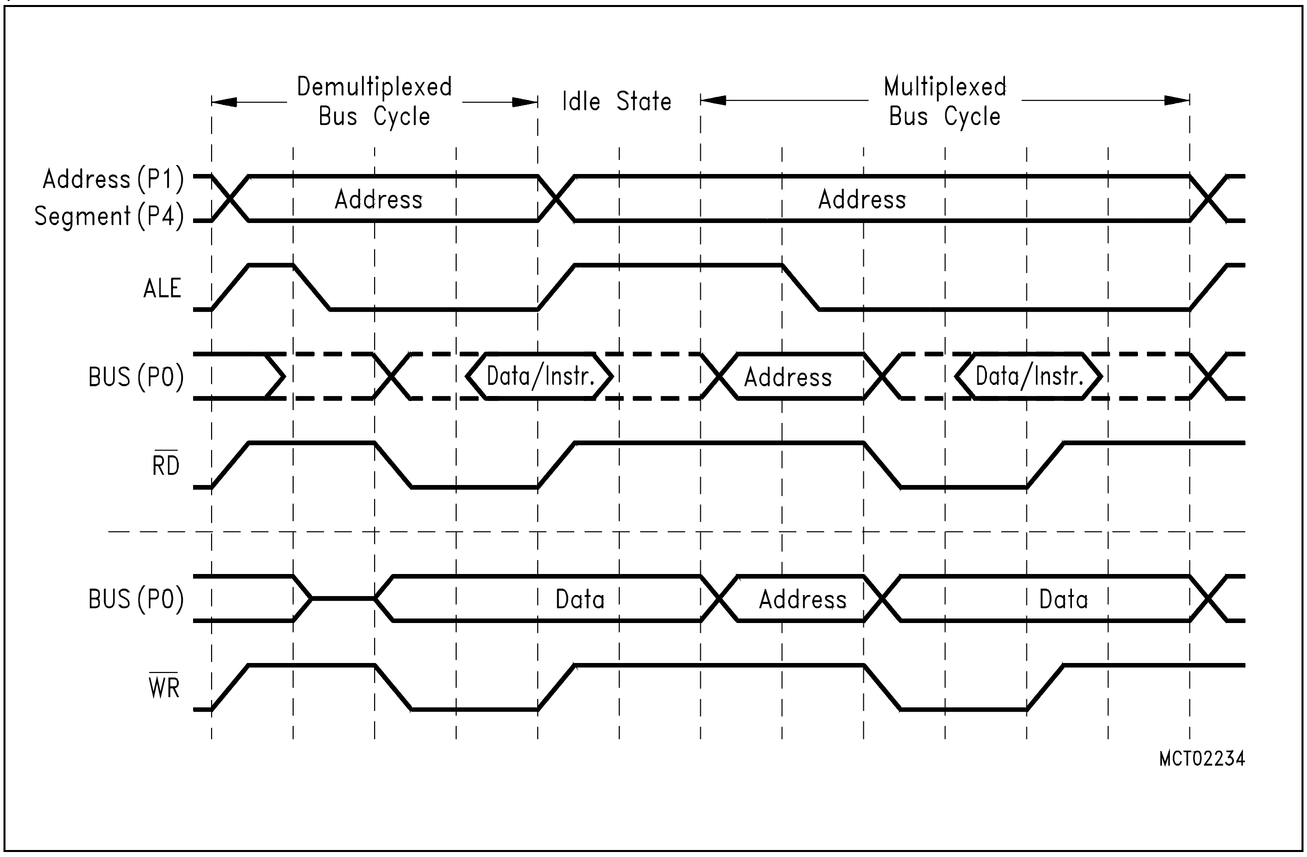


Figure 11-4 Switching from Demultiplexed to Multiplexed Bus Mode

The bus cycle is described in number of TCL's, where f_{CPU} = 1/(2 TCL).

External Data Bus Width

The EBC can operate on 8-bit or 16-bit wide external memory/peripherals. A 16-bit data bus uses PORT0, while an 8-bit data bus only uses P0L, the lower byte of PORT0. This saves on address latches, bus transceivers, bus routing and memory cost on the expense of transfer time. The EBC can control word accesses on an 8-bit data bus as well as byte accesses on a 16-bit data bus.

Word accesses on an 8-bit data bus are automatically split into two subsequent byte accesses, where the low byte is accessed first, then the high byte. The assembly of bytes to words and the disassembly of words into bytes is handled by the EBC and is transparent to the CPU and the programmer.

Byte accesses on a 16-bit data bus require that the upper and lower half of the memory can be accessed individually. In this case the upper byte is selected with the $\overline{\text{BHE}}$ signal, while the lower byte is selected with the A0 signal. So the two bytes of the memory can be enabled independent from each other, or together when accessing words.

When writing bytes to an external 16-bit device, which has a single $\overline{\text{CS}}$ input, but two $\overline{\text{WR}}$ enable inputs (for the two bytes), the EBC can directly generate these two write control signals. This saves the external combination of the $\overline{\text{WR}}$ signal with A0 or $\overline{\text{BHE}}$. In this case pin $\overline{\text{WR}}$ serves as $\overline{\text{WRL}}$ (write low byte) and pin $\overline{\text{BHE}}$ serves as $\overline{\text{WRH}}$ (write high byte). Bit WRCFG in register SYSCON selects the operating mode for pins $\overline{\text{WR}}$ and BHE. The respective byte will be written on both data bus halfs.

When reading bytes from an external 16-bit device, whole words may be read and the INCA-D automatically selects the byte to be input and discards the other. However, care must be taken when reading devices that change state when being read, like FIFOs, interrupt status registers, etc. In this case individual bytes should be selected using $\overline{\text{BHE}}$ and A0.

Bus Mode	Transfer Rate (Speed factor for byte/word/dword access)	System Requirements	Free I/O Lines
8-bit Multiplexed	Very low (1.5 / 3 / 6)	Low (8-bit latch, byte bus)	P1H, P1L
8-bit Demultipl.	Low (1 / 2 / 4)	Very low (no latch, byte bus)	P0H
16-bit Multiplexed	High (1.5 / 1.5 / 3)	High (16-bit latch, word bus)	P1H, P1L
16-bit Demultipl.	Very high (1 / 1 / 2)	Low (no latch, word bus)	---

Note: PORT1 gets available for general purpose I/O, when none of the BUSCON registers selects a demultiplexed bus mode.

Disable/Enable Control for Pin $\overline{\text{BHE}}$ (BYTDIS)

Bit BYTDIS is provided for controlling the active low Byte High Enable ($\overline{\text{BHE}}$) pin. The function of the $\overline{\text{BHE}}$ pin is enabled, if the BYTDIS bit contains a '0'. Otherwise, it is disabled and the pin can be used as standard I/O pin. The $\overline{\text{BHE}}$ pin is implicitly used by the External Bus Controller to select one of two byte-organized memory chips, which are connected to the INCA-D via a word-wide external data bus. After reset the $\overline{\text{BHE}}$ function is automatically enabled (BYTDIS = '0'), if a 16-bit data bus is selected during reset, otherwise it is disabled (BYTDIS='1'). It may be disabled, if byte access to 16-bit memory is not required, and the $\overline{\text{BHE}}$ signal is not used.

Segment Address Generation

During external accesses the EBC generates a (programmable) number of address lines on Port 4, which extend the 16-bit address output on PORT0 or PORT1, and so increase the accessible address space. The number of segment address lines is selected during reset and coded in bit field SALSEL in register RP0H (see table below and Table 24-3, "Code definitions of startup configurations," on page 603).

Table 11-1 Coding of SALSEL

SALSEL	Segment Address Lines	Directly accessible Address Space
11	Two: A17...A16	256 KByte
10	Six: A21...A16	4 MByte (Maximum)

Table 11-1 Coding of SALSEL

SALSEL	Segment Address Lines	Directly accessible Address Space
01	None	64 KByte (Minimum)
00	Four: A19...A16	1 MByte (default)

Note: The total accessible address space may be increased by accessing several banks which are distinguished by individual chip select signals.

$\overline{\text{CS}}$ Signal Generation

During external accesses the EBC can generate a (programmable) number of $\overline{\text{CS}}$ lines on Port 6, which allow to directly select external peripherals or memory banks without requiring an external decoder. The number of $\overline{\text{CS}}$ lines is selected during reset and coded in bits CSSEL in register RP0H (see table below and Table 24-3, "Code definitions of startup configurations," on page 603).

Table 11-2 Coding of CSSEL

CSSEL (P0H.2:1)	Chip Select Lines	
1 1	Reserved	
1 0	None	pins P6.0 and P6.1usable as GP I/Os
0 1	$\overline{\text{CS1}}$ and $\overline{\text{CS0}}$ availabe	two CS lines at pins P6.0 and P6.1 (default)
0 0	Reserved	

When CSSEL=01 has been selected, the chip select signal $\overline{\text{CS1}}$ has to be enabled manually by activating the corresponding alternate function of port pin P6.1 (see **Chapter 9.6.1**). Otherwise only the chip select signal $\overline{\text{CS0}}$ is enabled. The activation of $\overline{\text{CS0}}$ (and the implicit selection of the alternate function of P6.0) is done automativcally by the External Bus Controller.

The $\overline{\text{CSx}}$ outputs are associated with the BUSCONx registers and are driven active (low) for any access within the address area defined for the respective BUSCON register. For any access outside this defined address area the respective $\overline{\text{CSx}}$ signal will go inactive (high). At the beginning of each external bus cycle the corresponding valid $\overline{\text{CS}}$ signal is determined and activated. All other $\overline{\text{CS}}$ lines are deactivated (driven high) at the same time.

Note: The $\overline{\text{CSx}}$ signals will not be updated for an access to any internal address area (ie. when no external bus cycle is started), even if this area is covered by the respective ADDRSELx register. An access to an on-chip X-Peripheral deactivates all external $\overline{\text{CS}}$ signals.

Upon accesses to address windows without a selected $\overline{CS}$ line all selected $\overline{CS}$ lines are deactivated.

The chip select signals allow to be operated in four different modes, which are selected via bits CSWENx and CSRENx in the respective BUSCONx register.

CSWENx	CSRENx	Chip Select Mode
0	0	Address Chip Select (Default after Reset, mode for $\overline{CS0}$)
0	1	Read Chip Select
1	0	Write Chip Select
1	1	Read/Write Chip Select

Address Chip Select signals remain active until an access to another address window. An address chip select becomes active with the falling edge of ALE and becomes inactive with the falling edge of ALE of an external bus cycle that accesses a different address area. No spikes will be generated on the chip select lines.

Read or Write Chip Select signals remain active only as long as the associated control signal ($\overline{RD}$ or $\overline{WR}$) is active. This also includes the programmable read/write delay. Read chip select is only activated for read cycles, write chip select is only activated for write cycles, read/write chip select is activated for both read and write cycles (write cycles are assumed, if any of the signals $\overline{WRH}$ or $\overline{WRL}$ gets active). These modes save external glue logic, when accessing external devices like latches or drivers that only provide a single enable input.

Note: $\overline{CS0}$ provides an address chip select directly after reset when the first instruction is fetched.

Segment Address versus Chip Select

The external bus interface of the INCA-D supports many configurations for the external memory. By increasing the number of segment address lines the INCA-D can address a linear address space of 256 KByte, 1 MByte or 4 MByte. This allows to implement a large sequential memory area, and also allows to access a great number of external devices, using an external decoder. By increasing the number of $\overline{CS}$ lines the INCA-D can access memory banks or peripherals without external glue logic. These two features may be combined to optimize the overall system performance. Enabling 4 segment address lines and 2 chip select lines eg. allows to access two memory banks of 1 MByte each. So the available address space is 2 MByte (without glue logic).

Note: Bit SGTDIS of register SYSCON defines, if the CSP register is saved during interrupt entry (segmentation active) or not (segmentation disabled).

11.2 Programmable Bus Characteristics

Important timing characteristics of the external bus interface have been made user programmable to allow to adapt it to a wide range of different external bus and memory configurations with different types of memories and/or peripherals.

The following parameters of an external bus cycle are programmable:

- **ALE Control** defines the ALE signal length and the address hold time after its falling edge
- **Memory Cycle Time** (extendable with 1...15 waitstates) defines the allowable access time
- **Memory Tri-State Time** (extendable with 1 waitstate) defines the time for a data driver to float
- **Read/Write Delay Time** defines when a command is activated after the falling edge of ALE

Note: Internal accesses are executed with maximum speed and therefore are not programmable.
External acceses use the slowest possible bus cycle after reset. The bus cycle timing may then be optimized by the initialization software.

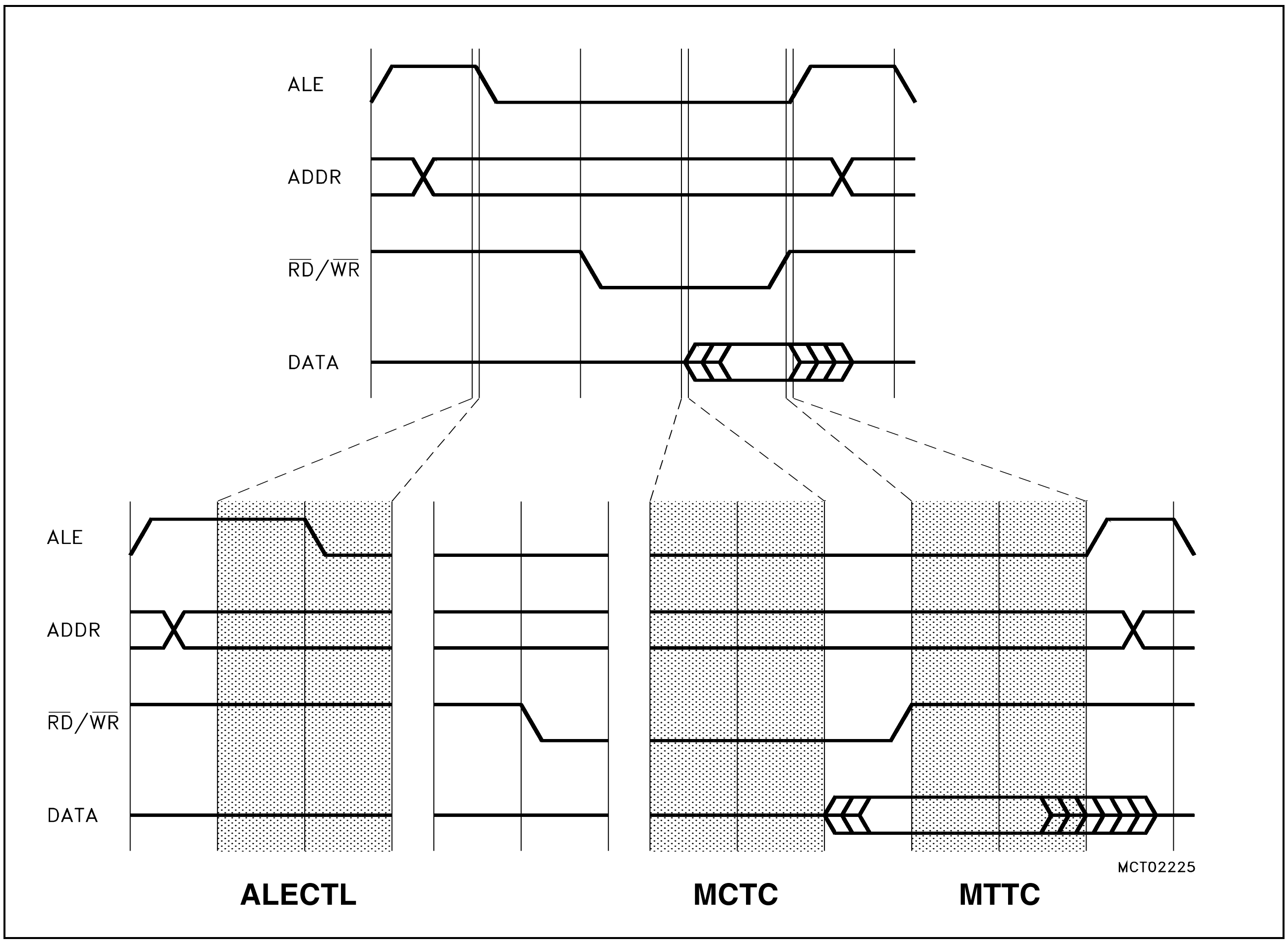


Figure 11-5 Programmable External Bus Cycle

ALE Length Control

The length of the ALE signal and the address hold time after its falling edge are controlled by the ALECTLx bits in the BUSCON registers. When bit ALECTL is set to ‘1’, external bus cycles accessing the respective address window will have their ALE signal prolonged by half a CPU clock. Also the address hold time after the falling edge of ALE (on a multiplexed bus) will be prolonged by half a CPU clock, so the data transfer within a bus cycle refers to the same CLKOUT edges as usual (ie. the data transfer is delayed by one CPU clock). This allows more time for the address to be latched.

Note: ALECTL0 is ‘1’ after reset to select the slowest possible bus cycle, the other ALECTLx are ‘0’ after reset.

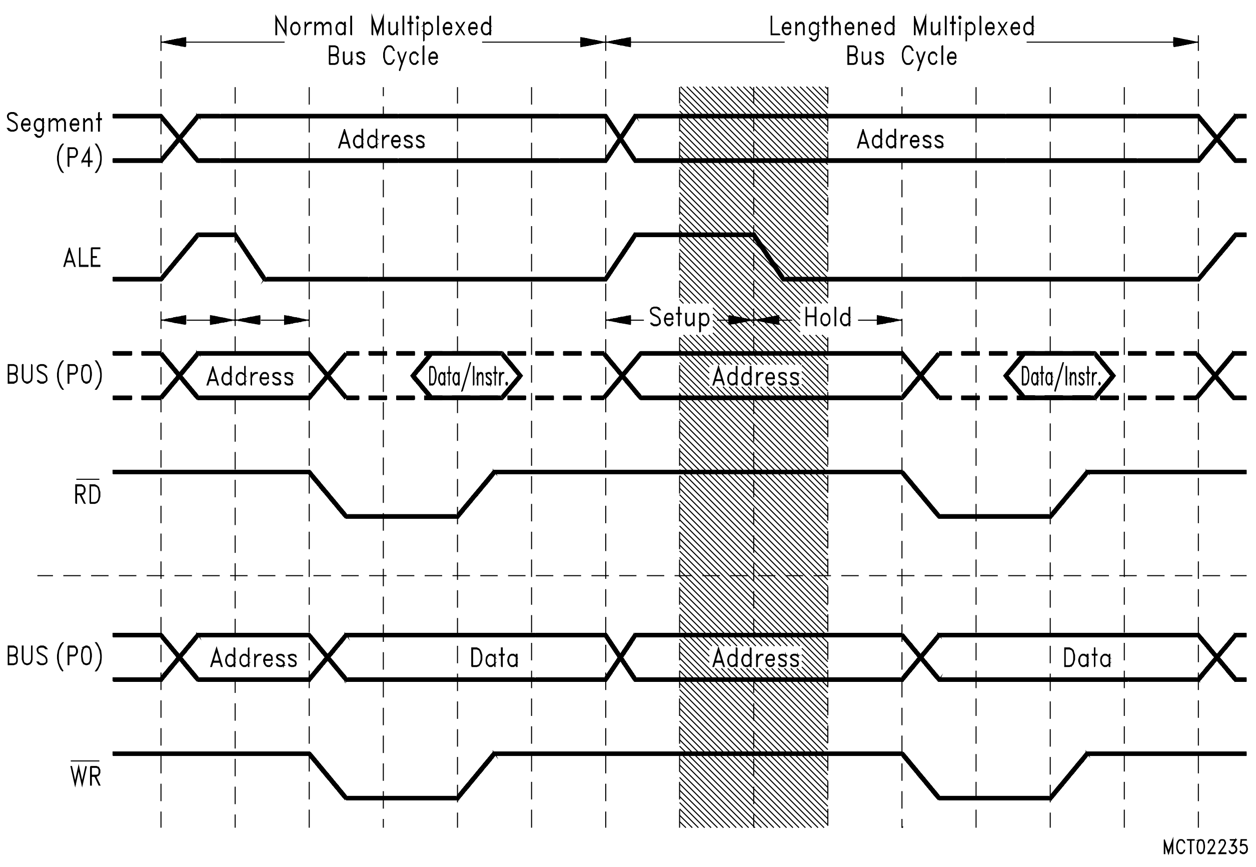


Figure 11-6 ALE Length Control

Programmable Memory Cycle Time

The INCA-D allows the user to adjust the controller's external bus cycles to the access time of the respective memory or peripheral. This access time is the total time required to move the data to the destination. It represents the period of time during which the controller's signals do not change.

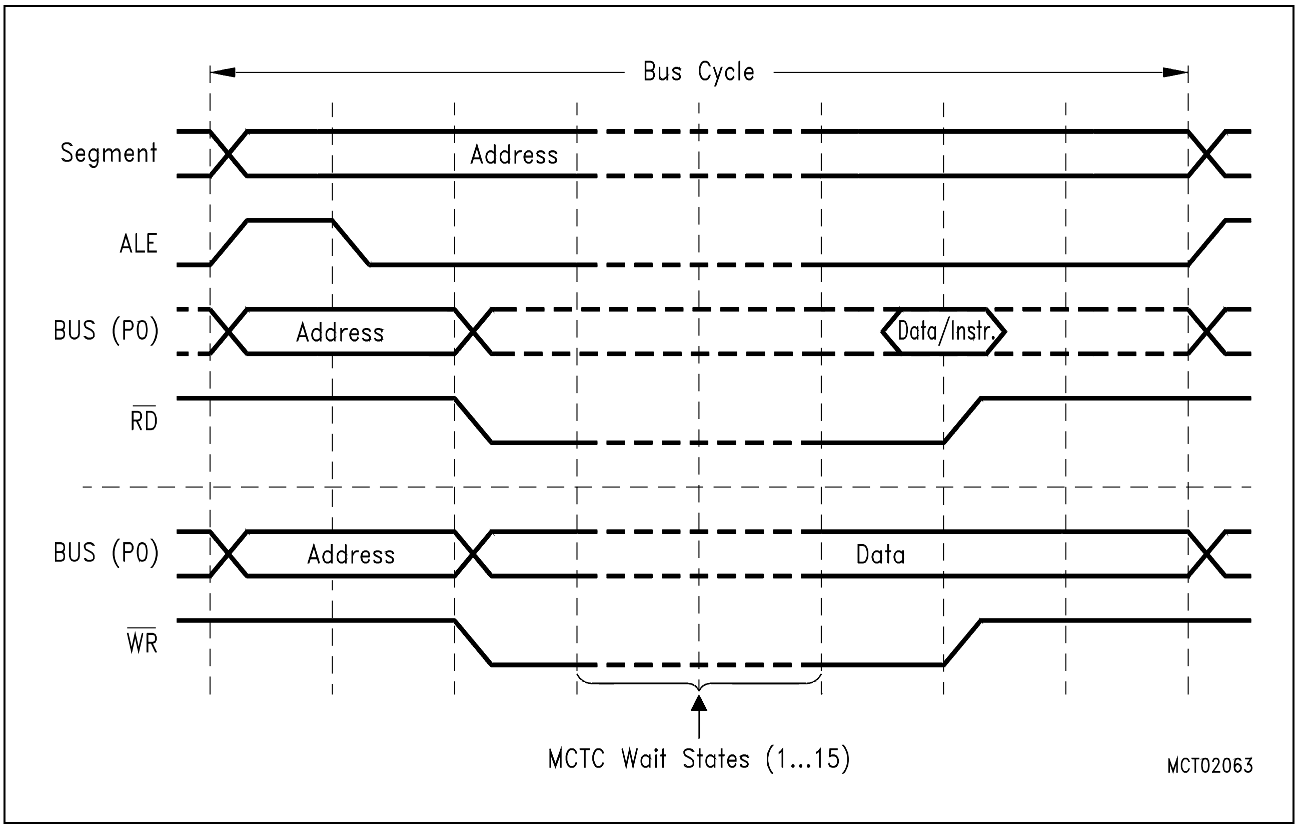


Figure 11-7 Memory Cycle Time

The external bus cycles of the INCA-D can be extended for a memory or peripheral, which cannot keep pace with the controller's maximum speed, by introducing wait states during the access (see figure above). During these memory cycle time wait states, the CPU is idle, if this access is required for the execution of the current instruction.

The memory cycle time wait states can be programmed in increments of one CPU clock within a range from 0 to 15 (default after reset) via the MCTC fields of the BUSCON registers. 15-<MCTC> waitstates will be inserted.

Programmable Memory Tri-State Time

The INCA-D allows the user to adjust the time between two subsequent external accesses to account for the tri-state time of the external device. The tri-state time defines, when the external device has released the bus after deactivation of the read command ($\overline{\text{RD}}$).

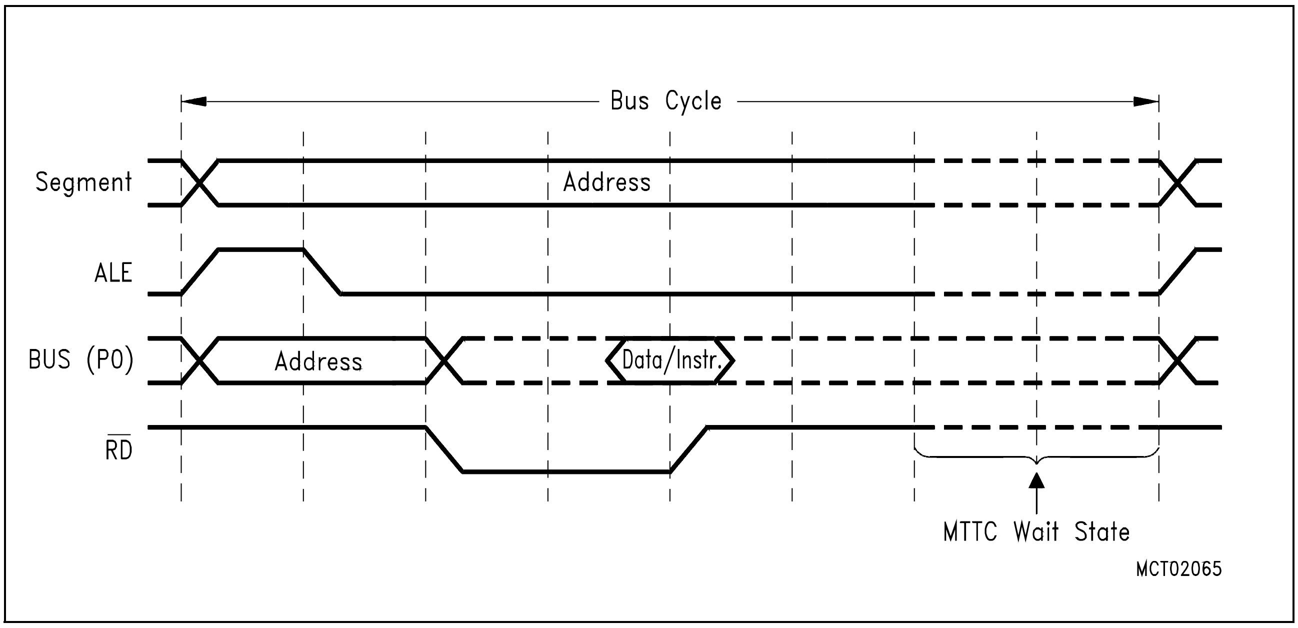


Figure 11-8 Memory Tri-State Time

The output of the next address on the external bus can be delayed for a memory or peripheral, which needs more time to switch off its bus drivers, by introducing a wait state after the previous bus cycle (see figure above). During this memory tri-state time wait state, the CPU is not idle, so CPU operations will only be slowed down if a subsequent external instruction or data fetch operation is required during the next instruction cycle.

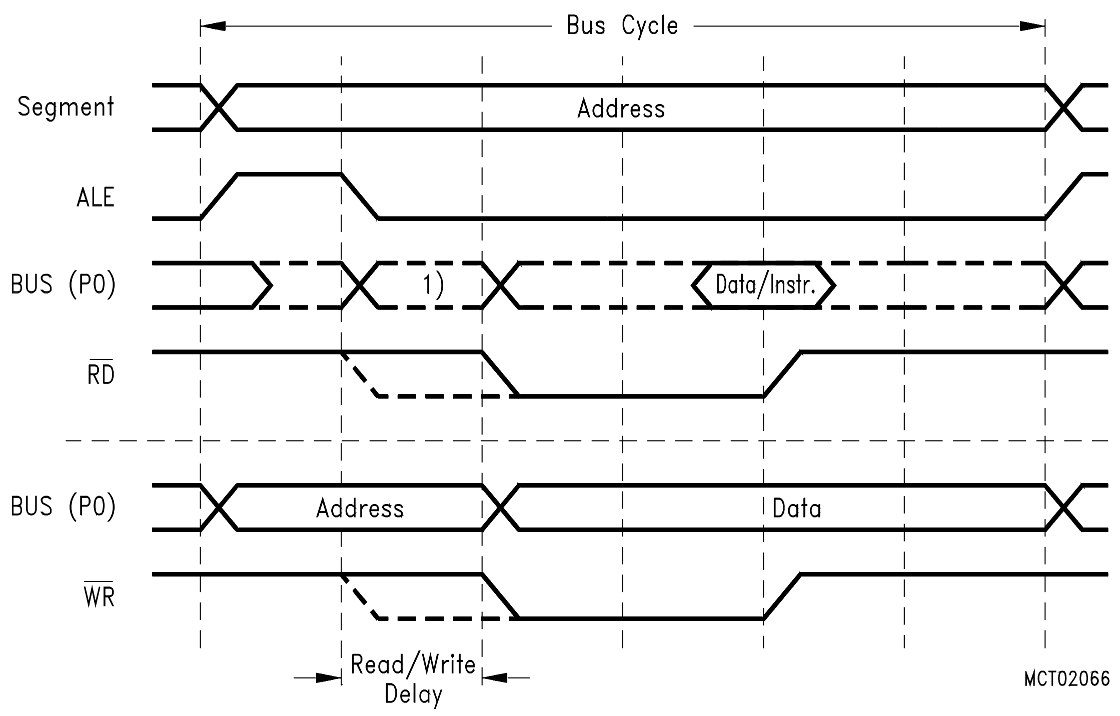
The memory tri-state time waitstate requires one CPU clock (28 ns at f_{CPU} = 36 MHz) and is controlled via the MTTCx bits of the BUSCON registers. A waitstate will be inserted, if bit MTTCx is '0' (default after reset).

Note: External bus cycles in multiplexed bus modes implicitly add one tri-state time waitstate in addition to the programmable MTTC waitstate.

Read/Write Signal Delay

The INCA-D allows the user to adjust the timing of the read and write commands to account for timing requirements of external peripherals. The read/write delay controls the time between the falling edge of ALE and the falling edge of the command. Without read/write delay the falling edges of ALE and command(s) are coincident (except for propagation delays). With the delay enabled, the command(s) become active half a CPU clock after the falling edge of ALE.

The read/write delay does not extend the memory cycle time, and does not slow down the controller in general. In multiplexed bus modes, however, the data drivers of an external device may conflict with the INCA-D's address, when the early $\overline{\text{RD}}$ signal is used. Therefore multiplexed bus cycles should always be programmed with read/write delay.



1) The data drivers from the previous bus cycle should be disabled when the $\overline{RD}$ signal becomes active.

Figure 11-9 Read/Write Delay

The read/write delay is controlled via the RWDCx bits in the BUSCON registers. The command(s) will be delayed, if bit RWDCx is ‘0’ (default after reset).

11.3 Controlling the External Bus Controller

A set of registers controls the functions of the EBC. General features like the usage of interface pins ($\overline{WR}$, $\overline{BHE}$) and segmentation are controlled via register SYSCON. The properties of a bus cycle like chip select mode, length of ALE, external bus mode, read/write delay and waitstates are controlled via registers BUSCON4...BUSCON0. Four of these registers (BUSCON4...BUSCON1) have an address select register (ADDRSEL4...ADDRSEL1) associated with them, which allows to specify up to four address areas and the individual bus characteristics within these areas. All accesses that are not covered by these four areas are then controlled via BUSCON0.

This allows to use memory components or peripherals with different interfaces within the same system, while optimizing accesses to each of them.

SYSCON (FF12$_H$ / 89$_H$) **SFR-b** **Reset Value: EA84$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
STKSZ			**0**	**SGT DIS**	**0**	**BYT DIS**	**CLK EN**	**WR CFG**	**CS CFG**	**0**	**0**	**0**	**XPEN**	**VISI BLE**	**0**
rw			r	rw	r	rw	rw	rw	rw	r	r	r	rw	rw	r

Bit	Function
VISIBLE	**Visible Mode Control** 0: Accesses to XBUS peripherals are done internally 1: XBUS peripheral accesses are made visible on the external pins
XPEN	**XBUS Peripheral Enable Bit** 0: Accesses to the on-chip X-Peripherals and their functions are disabled 1: The on-chip X-Peripherals are enabled and can be accessed
CSCFG	**Chip Select Configuration Control** 0: Latched $\overline{\text{CS}}$ mode. The $\overline{\text{CS}}$ signals are latched internally and driven to the (enabled) port pins synchronously. 1: Unlatched $\overline{\text{CS}}$ mode. The $\overline{\text{CS}}$ signals are directly derived from the address and driven to the (enabled) port pins.
WRCFG	**Write Configuration Control** (Set according to pin P0H.0 during reset) 0: Pins $\overline{\text{WR}}$ and $\overline{\text{BHE}}$ retain their normal function 1: Pin $\overline{\text{WR}}$ acts as $\overline{\text{WRL}}$, pin $\overline{\text{BHE}}$ acts as $\overline{\text{WRH}}$
CLKEN	**System Clock Output Enable** (CLKOUT [1]) 0: CLKOUT disabled 1: CLKOUT enabled;
BYTDIS	**Disable/Enable Control for Pin $\overline{\text{BHE}}$** (Set according to data bus width) 0: Pin $\overline{\text{BHE}}$ enabled 1: Pin $\overline{\text{BHE}}$ disabled, pin may be used for general purpose I/O
SGTDIS	**Segmentation Disable/Enable Control** '0': Segmentation enabled (CSP is saved/restored during interrupt entry/exit) '1': Segmentation disabled (Only IP is saved/restored)
STKSZ	**System Stack Size** Selects the size of the system stack (in the internal RAM) from 32 to 1024 words

[1] The implemented pad type for CLKOUT supports only an open-drain output driver

Note: Register SYSCON cannot be changed after execution of the EINIT instruction.

Note: Only exception: bit VISIBLE can also be changed by the on-chip debug support with DPEC access.

The layout of the five BUSCON registers is identical. Registers BUSCON4...BUSCON1, which control the selected address windows, are completely under software control, while register BUSCON0, which eg. is also used for the very first code access after reset, is partly controlled by hardware, ie. it is initialized via PORT0 during the reset sequence. This hardware control allows to define an appropriate external bus for systems, where no internal program memory is provided.

BUSCON0 (FF0C$_H$ / 86$_H$) SFR **Reset Value: 0680$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN0	**CSR EN0**	**0**	**0**	**0**	**BUS ACT0**	**ALE CTL0**	**0**	**BTYP**		**MTT C0**	**RWD C0**	**MCTC**			
rw	rw	r	r	r	rw	rw	r	rw		rw	rw	rw			

BUSCON1 (FF14$_H$ / 8A$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN1	**CSR EN1**	**0**	**0**	**0**	**BUS ACT1**	**ALE CTL1**	**0**	**BTYP**		**MTT C1**	**RWD C1**	**MCTC**			
rw	rw	r	r	r	rw	rw	r	rw		rw	rw	rw			

BUSCON2 (FF16$_H$ / 8B$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN2	**CSR EN2**	**0**	**0**	**0**	**BUS ACT2**	**ALE CTL2**	**0**	**BTYP**		**MTT C2**	**RWD C2**	**MCTC**			
rw	rw	-	r	-	rw	rw	-	rw		rw	rw	rw			

BUSCON3 (FF18$_H$ / 8C$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN3	**CSR EN3**	**0**	**0**	**0**	**BUS ACT3**	**ALE CTL3**	**0**	**BTYP**		**MTT C3**	**RWD C3**	**MCTC**			
rw	rw	r	r	r	rw	rw	r	rw		rw	rw	rw			

BUSCON4 (FF1A$_H$ / 8D$_H$) SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CSW EN4	**CSR EN4**	**0**	**0**	**0**	**BUS ACT4**	**ALE CTL4**	**0**	**BTYP**		**MTT C4**	**RWD C4**	**MCTC**			
rw	rw	r	r	r	rw	rw	r	rw		rw	rw	rw			

Note: As default the bits BUSACT0 and ALECTL0 are set ('1') and bit field BTYP is loaded with the bus configuration selected via PORT0.

Bit	Function
MCTC	**Memory Cycle Time Control** (Number of memory cycle time wait states) 0 0 0 0 : 15 waitstates (Number = 15 - <MCTC>) . . . 1 1 1 1 : No waitstates
RWDCx	**Read/Write Delay Control for BUSCONx** ‘0’: With read/write delay: activate command 1 TCL after falling edge of ALE ‘1’: No read/write delay: activate command with falling edge of ALE
MTTCx	**Memory Tristate Time Control** ‘0’: 1 waitstate ‘1’: No waitstate
BTYP	**External Bus Configuration** 0 0 : 8-bit Demultiplexed Bus 0 1 : 8-bit Multiplexed Bus 1 0 : 16-bit DemultiplexedBus 1 1 : 16-bit Multiplexed Bus **Note:** For BUSCON0 BTYP is defined via PORT0 during reset.
ALECTLx	**ALE Lengthening Control** ‘0’: Normal ALE signal ‘1’: Lengthened ALE signal
BUSACTx	**Bus Active Control** ‘0’: External bus disabled ‘1’: External bus enabled (within the respective address window, see ADDRSEL)
CSRENx	**Read Chip Select Enable** ‘0’: The $\overline{\text{CS}}$ signal is independent of the read command ($\overline{\text{RD}}$) ‘1’: The $\overline{\text{CS}}$ signal is generated for the duration of the read command
CSWENx	**Write Chip Select Enable** ‘0’: The $\overline{\text{CS}}$ signal is independent of the write command ($\overline{\text{WR}}$,$\overline{\text{WRL}}$,$\overline{\text{WRH}}$) ‘1’: The $\overline{\text{CS}}$ signal is generated for the duration of the write command

ADDRSEL1 ($FE18_H$ / $0C_H$) SFR **Reset Value: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGSAD												**RGSZ**			
rw												rw			

ADDRSEL2 ($FE1A_H$ / $0D_H$) SFR **Reset Value: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGSAD												**RGSZ**			
rw												rw			

ADDRSEL3($FE1C_H$ / $0E_H$) SFR **Reset Value: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGSAD												**RGSZ**			
rw												rw			

ADDRSEL4 ($FE1E_H$ / $0F_H$) SFR **Reset Value: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RGSAD												**RGSZ**			
rw												rw			

Bit	Function
RGSZ	**Range Size Selection** Defines the size of the address area controlled by the respective BUSCONx/ ADDRSELx register pair. See table below.
RGSAD	**Range Start Address** Defines the upper bits of the start address (A23..A12) of the internal addressable address space. Only address lines A21..A0 can be accessed externally. The possible start addresses for A21..A12 are summarized in the table below.

Note: The 2 chip selects can be used to select a memory window of 4 MByte each. In this case the RGSAD bit 14 has to programmed accordingly.

Note: There is no register ADDRSEL0, as register BUSCON0 controls all external accesses outside the four address windows of BUSCON4...BUSCON1 within the complete address space.

Definition of Address Areas

The four register pairs BUSCON4/ADDRSEL4...BUSCON1/ADDRSEL1 allow to define 4 separate address areas within the address space of the INCA-D. Within each of these address areas external accesses can be controlled by one of the four different bus modes, independent of each other and of the bus mode specified in register BUSCON0. Each ADDRSELx register in a way cuts out an address window, within which the parameters in register BUSCONx are used to control external accesses. The range start address of such a window defines the upper address bits, which are not used within the address window of the specified size (see table below). For a given window size only those upper address bits of the start address are used (marked “R”), which are not implicitly used for addresses inside the window. The lower bits of the start address (marked “x”) are disregarded.

Bit field RGSZ	Resulting Window Size	Relevant Bits (R) of Start Address (A21...A12)
0 0 0 0	4 KByte	R R R R R R R R R R
0 0 0 1	8 KByte	R R R R R R R R R x
0 0 1 0	16 KByte	R R R R R R R R x x
0 0 1 1	32 KByte	R R R R R R R x x x
0 1 0 0	64 KByte	R R R R R R x x x x
0 1 0 1	128 KByte	R R R R R x x x x x
0 1 1 0	256 KByte	R R R R x x x x x x
0 1 1 1	512 KByte	R R R x x x x x x x
1 0 0 0	1 MByte	R R x x x x x x x x
1 0 0 1	2 MByte	R x x x x x x x x x
1 0 1 0	4 MByte	x x x x x x x x x x
1 0 1 1	Reserved	
1 1 x x	Reserved.	

Address Window Arbitration

For each access the EBC compares the current address with all address select registers (ADDRSELx and XADRSx). This comparison is done in four levels.

Priority 1: The XADRSx registers are evaluated first. A match with one of these registers directs the access to the respective X-Peripheral using the corresponding XBCONx register and ignoring all other ADDRSELx registers.

Priority 2: Registers ADDRSEL2 and ADDRSEL4 are evaluated before ADDRSEL1 and ADDRSEL3, respectively. A match with one of these registers directs the access to the respective external area using the corresponding BUSCONx register and ignoring registers ADDRSEL1/3 (see figure below).

Priority 3: A match with registers ADDRSEL1 or ADDRSEL3 directs the access to the respective external area using the corresponding XBCONx register.

Priority 4: If there is no match with any XADRSx or ADDRSELx register the access to the external bus uses register BUSCON0.

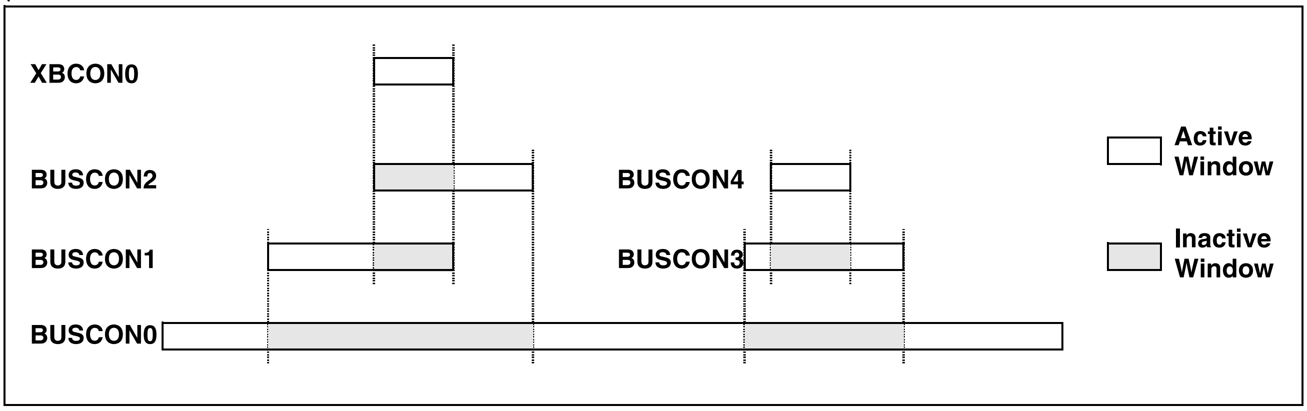


Figure 11-10 Address Window Arbitration

Note: Only the indicated overlaps are defined. All other overlaps lead to erroneous bus cycles. Eg. ADDRSEL4 may not overlap ADDRSEL2 or ADDRSEL1. The hardwired XADRSx registers are defined non-overlapping.

For a description of the regsiter **RP0H** which holds the reset configuration (number of chip select lines, address range etc.) please refer to chapter "System Startup Configuration" on page 600.

Precautions and Hints

• The external bus interface is enabled as long as at least one of the BUSCON registers has its BUSACT bit set.

• PORT1 will output the intra-segment address as long as at least one of the BUSCON registers selects a demultiplexed external bus, even for multiplexed bus cycles.

• Not all address areas defined via registers ADDRSELx may overlap each other. The operation of the EBC will be unpredictable in such a case. See chapter „Address Window Arbitration".

• The address areas defined via registers ADDRSELx may overlap internal address areas. Internal accesses will be executed in this case.

• For any access to an internal address area the EBC will remain inactive (see EBC Idle State).

11.4 EBC Idle State

When the external bus interface is enabled, but no external access is currently executed, the EBC is idle. As long as only internal resources (from an architecture point of view) like IRAM, GPRs or SFRs, etc. are used the external bus interface does not change (see table below).

Accesses to on-chip X-Peripherals are also controlled by the EBC. However, even though an X-Peripheral appears like an external peripheral to the controller, the respective accesses do not generate valid external bus cycles.

Due to timing constraints address and write data of an XBUS cycle are reflected on the external bus interface (see table below). The „address“ mentioned above includes PORT1, Port 4, $\overline{BHE}$ and ALE which also pulses for an XBUS cycle. The external $\overline{CS}$ signals on Port 6 are driven inactive (high) because the EBC switches to an internal $\overline{XCS}$ signal.

The **external control signals** ($\overline{RD}$ and $\overline{WR}$ or $\overline{WRL}/\overline{WRH}$ if enabled) don’t remain inactive during BUS accesses. When internal XRAM addresses are accessed, the control signals show the same behavior as for external accesses.

Table 11-3 Status of the external bus interface during EBC idle state:

Pins	Internal accesses only	XBUS accesses
PORT0	Tristated (floating)	Tristated (floating) for read accesses XBUS write data for write accesses
PORT1	Last used external address (if used for the bus interface)	Last used XBUS address (if used for the bus interface)
Port 4	Last used external segment address (on selected pins)	Last used XBUS segment address (on selected pins)
Port 6	Active external $\overline{CS}$ signal corresponding to last used address	Inactive (high) for selected $\overline{CS}$ signals
$\overline{BHE}$	Level corresponding to last external access	Level corresponding to last XBUS access
ALE	Inactive (low)	Pulses as defined for X-Peripheral
$\overline{RD}$	Inactive (high)	Level corresponding to last XBUS access
$\overline{WR}/\overline{WRL}$	Inactive (high)	Level corresponding to last XBUS access
$\overline{WRH}$	Inactive (high)	Level corresponding to last XBUS access

12 The Watchdog Timer (WDT)

To allow recovery from software or hardware failure, the INCA-D provides a Watchdog Timer. If the software fails to service this timer before an overflow occurs, an internal reset sequence will be initiated. This internal reset will also pull the $\overline{\text{RSTOUT}}$ pin low, which also resets the peripheral hardware, which might be the cause for the malfunction. When the watchdog timer is enabled and the software has been designed to service it regularly before it overflows, the watchdog timer will supervise the program execution, as it only will overflow if the program does not progress properly. The watchdog timer will also time out, if a software error was due to hardware related failures. This prevents the controller from malfunctioning for longer than a user-specified time.

The watchdog timer provides two registers: a read-only timer register that contains the current count, and a control register for initialization.

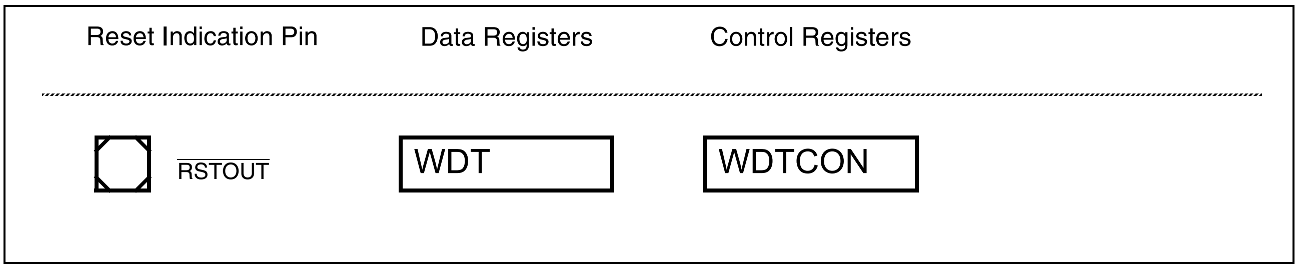


Figure 12-1 SFRs and Port Pins associated with the Watchdog Timer

The watchdog timer is a 16-bit up counter which can be clocked with the CPU clock (f_{CPU}) either divided by 2 or divided by 128. This 16-bit timer is realized as two concatenated 8-bit timers (see figure below). The upper 8 bits of the watchdog timer can be preset to a user-programmable value via a watchdog service access in order to vary the watchdog expire time. The lower 8 bits are reset on each service access.

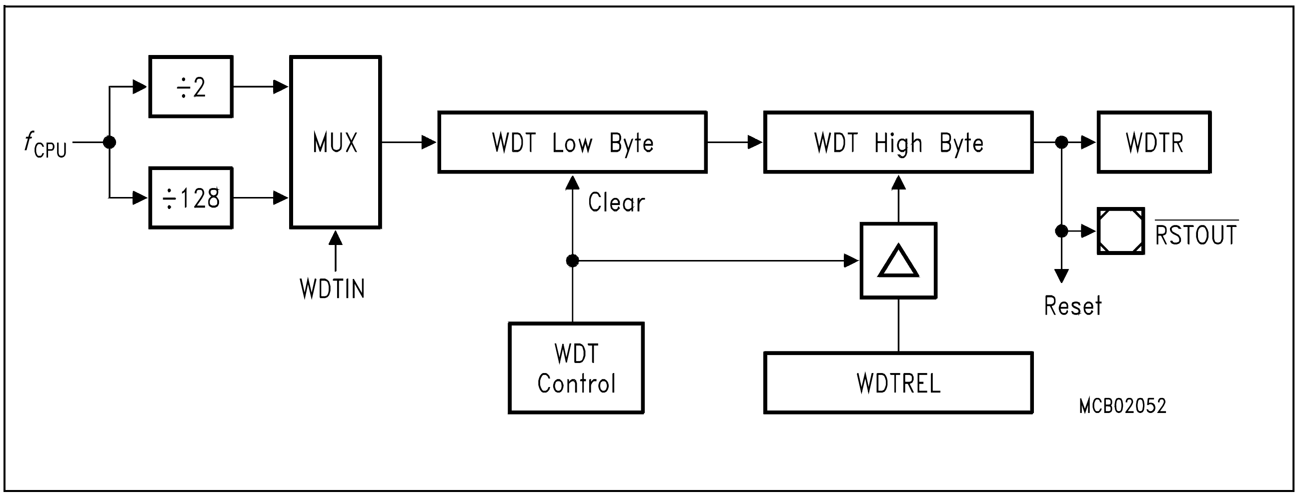


Figure 12-2 Watchdog Timer Block Diagram

12.1 Operation of the Watchdog Timer

The current count value of the Watchdog Timer is contained in the Watchdog Timer Register WDT, which is a non-bitaddressable read-only register. The operation of the Watchdog Timer is controlled by its bitaddressable Watchdog Timer Control Register WDTCON. This register specifies the reload value for the high byte of the timer, selects the input clock prescaling factor and provides a flag that indicates a watchdog timer overflow.

WDTCON (FFAE$_H$ / D7$_H$) SFR-b Reset Value: 003C$_H$

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WDTREL								**0**	**0**	**1**	**LHWR**	**SHWR**	**SWR**	**WDTR**	**WDTIN**
rw								r	r	r	r	r	r	r	rw

The reset value depends on the Reset Source.

Bit	Function
WDTIN	Watchdog Timer Input Frequency Selection ‘0’: Input frequency is f_{CPU} / 2 ‘1’: Input frequency is f_{CPU} / 128
WDTR[1)]	Watchdog Timer Reset Indication Flag Set by the watchdog timer on an overflow. Cleared by the SRVWDT instruction.
SWR	Software Reset Set by the command SRST
SHWR	Short Hardware Reset Set by the Input RSTIN
LHWR	Long Hardware Reset Set by the Input RSTIN or by Undervoltage Detection
WDTREL	Watchdog Timer Reload Value (for the high byte of WDT)

1) More than one reset source may be visible. After EINIT all reset bits are cleared

After any software reset, external hardware reset (see note), or watchdog timer reset, the watchdog timer is enabled and starts counting up from 0000$_H$ with the frequency f_{CPU}/2. The input frequency may be switched to f_{CPU}/128 by setting bit WDTIN. The watchdog timer can be disabled via the instruction DISWDT (Disable Watchdog Timer). Instruction DISWDT is a protected 32-bit instruction which will ONLY be executed during the time between a reset and execution of either the EINIT (End of Initialization) or the SRVWDT (Service Watchdog Timer) instruction. Either one of these instructions disables the execution of DISWDT.

When the watchdog timer is not disabled via instruction DISWDT, it will continue counting up, even during Idle Mode. If it is not serviced via the instruction SRVWDT by the time the count reaches $\mathrm{FFFF_H}$ the watchdog timer will overflow and cause an internal reset. This reset will pull the external reset indication pin $\overline{\mathrm{RSTOUT}}$ low. It differs from a software or external hardware reset in that bit WDTR (Watchdog Timer Reset Indication Flag) of register WDTCON will be set. A hardware reset or the SRVWDT instruction will clear this bit. Bit WDTR can be examined by software in order to determine the cause of the reset.

A watchdog reset will also complete a running external bus cycle before starting the internal reset sequence

Note: After a hardware reset that activates the Bootstrap Loader the watchdog timer will be disabled.

To prevent the watchdog timer from overflowing, it must be serviced periodically by the user software. The watchdog timer is serviced with the instruction SRVWDT, which is a protected 32-bit instruction. Servicing the watchdog timer clears the low byte and reloads the high byte of the watchdog timer register WDT with the preset value from bitfield WDTREL which is the high byte of register WDTCON. Servicing the watchdog timer will also reset bit WDTR. After being serviced the watchdog timer continues counting up from the value (<WDTREL> * 2^8). Instruction SRVWDT has been encoded in such a way that the chance of unintentionally servicing the watchdog timer (eg. by fetching and executing a bit pattern from a wrong location) is minimized. When instruction SRVWDT does not match the format for protected instructions the Protection Fault Trap will be entered, rather than the instruction be executed.

The time period for an overflow of the watchdog timer is programmable in two ways:

• **the input frequency** to the watchdog timer can be selected via bit WDTIN in register WDTCON to be either $f_{CPU}/2$ or $f_{CPU}/128$.
• **the reload value** WDTREL for the high byte of WDT can be programmed in register WDTCON.

The period P_{WDT} between servicing the watchdog timer and the next overflow can therefore be determined by the following formula:

$$P_{WDT} = \frac{2^{(1 + \langle WDTIN \rangle * 6)} * (2^{16} - \langle WDTREL \rangle * 2^8)}{f_{CPU}}$$

Note: For safety reasons, the user is advised to rewrite WDTCON each time before the watchdog timer is serviced

13 The Bootstrap Loader

The built-in bootstrap loader of the INCA-D provides a mechanism to load the startup program, which is executed after reset, via the serial interface.

The bootstrap loader moves code/data into the internal RAM, but it is also possible to transfer data via the serial interface into an external RAM using a second level loader routine. It may be used to provide lookup tables or may provide "core-code", ie. a set of general purpose subroutines, eg. for I/O operations, number crunching, system initialization, etc.

The boot strap loader program resides inside the internal ROM. It is used to download an executable code into the internal memory via *ASC*.

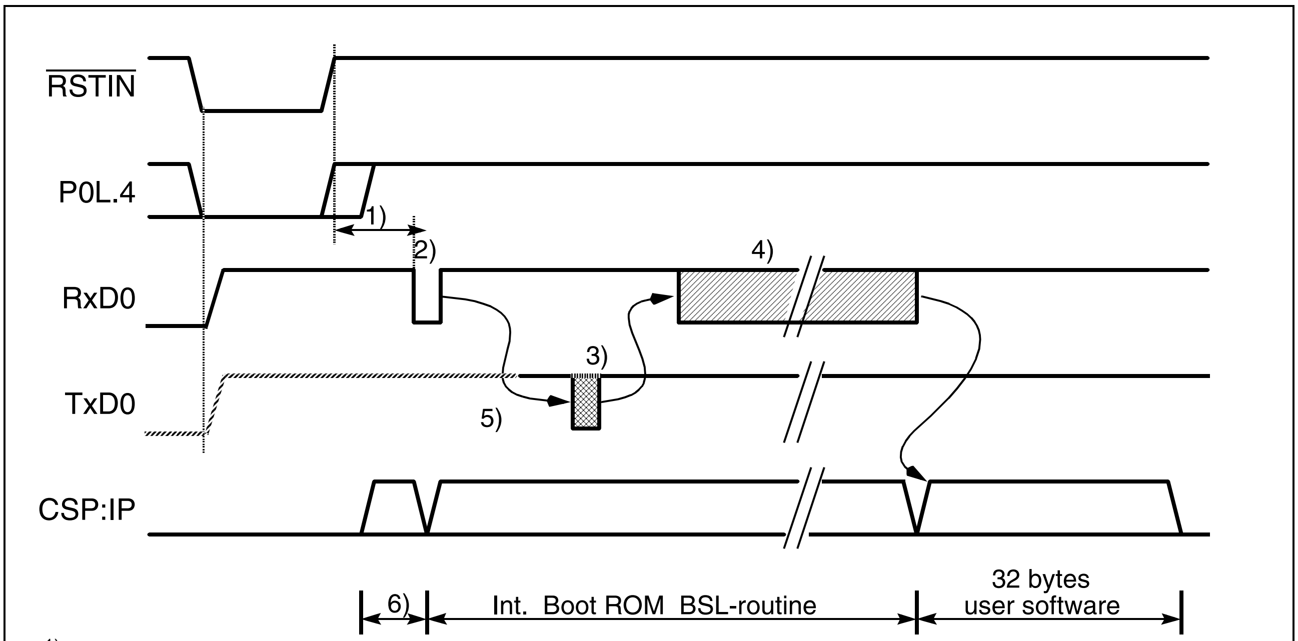
The ASC bootstrap loader is similar to the loader used in the C167 microprocessor family. However, there are some major differences:

- The baudrate error is minimized because the reload value for the baudrate generator as well as the fractional divider (register S0FDV) are modified according to the measured time of the received zero byte.
- The user program is started by a jump to its start address instead of a hardware or software reset.

13.1 Activation of ASC Bootstrap Loader

The ASC bootstrap loader in ROM is activated by a hardware reset (pin $\overline{\text{RSTIN}}$) under the following conditions:

- Port 0.4 is pulled low by an external pull-down resistor.
- Port 0.3 is left open.
- Pin TEST is connected to VSS



[1] BSL initialization time, > 2µs @ f_{CPU} = 20 MHz.
[2] Zero byte (1 start bit, eight '0' data bits, 1 stop bit), sent by host.
[3] Identification byte, sent by INCA-D.
[4] 32 bytes of code / data, sent by host.
[5] Caution: TxD0 is only driven a certain time after reception of the zero byte.
[6] Internal Boot ROM.

Figure 13-1 Bootstrap Loader Sequence

After reset, the microcontroller expects the serial reception of a zero byte (8 data bits = 00_H, one stop bit, no parity) from a host at pin RXD (P3.11). The time is measured by timer T6, which runs at maximum speed. The result T6 depends on the baud rate BR and the CPU clock frequency f_{CPU}.

$$T6 = 9 * f_{CPU} / (4*BR) \qquad (1)$$

Factor 9 results from zero byte duration including start bit, factor 4 from T6 prescaler.

Using T6, the bootstrap loader software calculates the optimum combination of

- FDV = value of the Fractional Divider Register S0FDV
- BG = Reload value for the Baud rate Timer / Reload Register S0BG

For this purpose, the software modifies FDV in a loop and calculates BG.

As described in **Chapter 15**, the baud rate is given by:

$$BR = FDV * f_{CPU} / (512*16*(BG+1)) \qquad (2)$$

Thus, BG can be calculated from equation 1 and 2 in the following way:

$$BG = (T6 * FDV * 4) / (9*512*16) - 1 = (T6 * FDV - 18432) / 18432 \qquad (3)$$

To get a rounded integer result, 18432/2 is added to numerator. So the software uses actually the following equation:

$$BG = (T6 * FDV - 9216) / 18432 \quad (4)$$

The remainder of the division will be stored. The absolute error (in T6 units) introduced by this division is:

$$Error_0 = |T6 - T6(\text{due to BG calculation})| \quad (5)$$

$$= |18432 * BG + Remainder + 9216) / FDV - 18432 * (BG + 1) / FDV|$$

$$= |Remainder - 9216| / FDV$$

To calculate the error with higher resolution and to get a rounded result, the numerator is shifted left 10 bits, and FDV/2 is added. So the software uses actually the following equation:

$$Error= ((|Remainder - 9216| << 10) + (FDV >>1))/ FDV \quad (6)$$

The software searches for the combination of FDV and BG that produces the minimum error. FDV is modified only from 256 to 512, because values 1 to 255 would not produce better results. If the optimum FDV value is 512, it will be changed to 0 according to the ASC description.

- Then the serial port is initialized in the following way:
- 8 data bits, one stop bit, no parity bit,
- fractional divider and baud rate generator enabled,
- calculated values for S0FDV and S0BG,
- TXD output set to 1 and enabled.

An identification byte $D5_H$ is sent back to the host via pin TXD (P3.10).

*Note: The range of timer T6 (max. 65535 increments) determines the lowest possible baud rate: BR > (9 * f_{CPU}) / (4* 65535)*

Note: At very high baud rates, the accuracy is limited by the restrictions for frequency division via S0FDV and S0BG. Additionally, the quantization error of timer T6 and signal distortion effects (e.g., different on/off delays) must be taken into account.

13.2 Loading the Startup Code

At the next step, the ASC bootstrap loader goes into a loop expecting 32 bytes to be received. This loop does not have an exit on time-out condition, so the program will wait forever unless 32 bytes are received. The received bytes are stored sequentially in the internal RAM beginning at address 00’$FA40_H$ and ending at address 00’$FA5F_H$.

After the reception of the 32 bytes, the ASC bootstrap loader automatically performs a jump to location 00’$FA40_H$, and the loaded program will be executed. Starting at this point, the program is no longer defined by the ROM code.

In most cases 32 bytes are not sufficient for a complete loader program. Therefore this code will normally be used as a pre-loader to copy the main loader program to internal RAM with start address 00’FA60$_H$. The maximum byte address of internal RAM is 00’FDFF$_H$.

After the reception of the main loader, the 32-byte pre-loader performs a jump to the start address of this program.

13.3 Transfer of User Program

After the boot strap loader execution, the system has the configuration in table 13-1:

Table 13-1 Register Content After Boot Strap Loader Execution

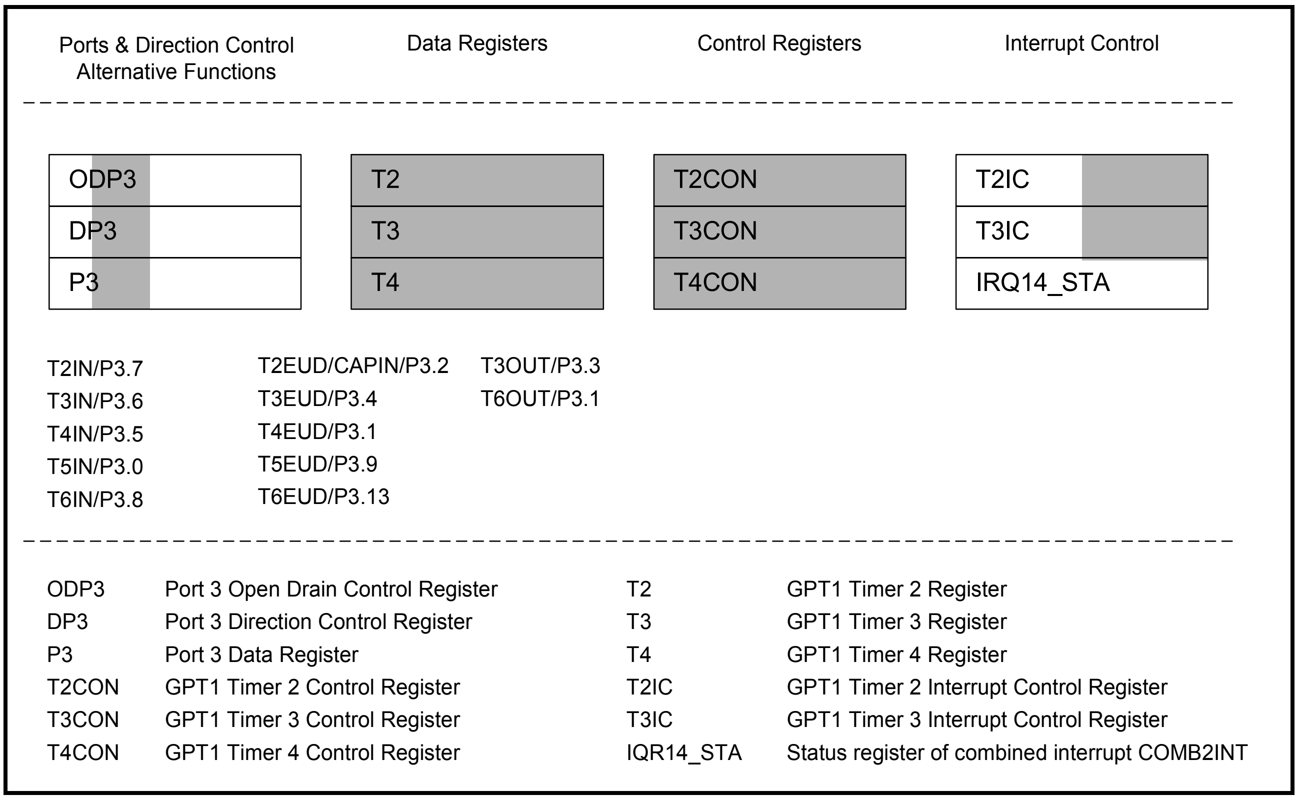
Peripheral Unit	State After Bootstrap Loader Execution
Watchdog Timer	disabled
CP Register	FA00$_H$
SYSCON Register	0E00$_H$
BUSCON0 Register	0XX0$_H$ according to bus configuration during reset
XADRS1 Register	0DF0$_H$
XBCON1 Register	04BF$_H$
SP Register	FA40$_H$
STKUN Register	FA40$_H$
STKOV Register	FA20$_H$
S0CON Register	8011$_H$
S0DV Register	Depending on zero byte evaluation
S0BG Register	Reload value depending on zero byte evaluation
S0TBIR in S0TBIC Register	1
TxD/P3.10	1
DP3.10	1

Note: During the entire bootstrap loading sequence, all interrupts and error resets must remain disabled.

Note: The user program can not be started via a hardware or software reset.

An address range from 00'0000$_H$ to 00'7FFF$_H$ (32 KByte) is reserved for internal accesses. Code fetches from this area will be made to the boot ROM. Data fetches from this area will return undefined values because there are no valid internal ROM data. The maximum stack size is 32 words. If necessary, the user program can modify the above configuration.

14 General Purpose Timer Unit



ODP3	Port 3 Open Drain Control Register	T2	GPT1 Timer 2 Register
DP3	Port 3 Direction Control Register	T3	GPT1 Timer 3 Register
P3	Port 3 Data Register	T4	GPT1 Timer 4 Register
T2CON	GPT1 Timer 2 Control Register	T2IC	GPT1 Timer 2 Interrupt Control Register
T3CON	GPT1 Timer 3 Control Register	T3IC	GPT1 Timer 3 Interrupt Control Register
T4CON	GPT1 Timer 4 Control Register	IQR14_STA	Status register of combined interrupt COMB2INT

Figure 14-1 SFR associated with GPT

The General Purpose Timer Unit (GPT) represents very flexible multifunctional timer structures which may be used for timing, event counting, pulse width measurement, pulse generation, frequency multiplication, and other purposes.

In the INCA-D, there are following alternate function pins available: T2IN, T2EUD/CAPIN, T3IN, T3EUD, T3OUT, T4IN, T4EUD/T6OUT, T5IN, T5EUD, T6IN, and T6EUD. These pins are part of the alternate functions of Port 3; please refer to Table 9-2, “Alternate Functions of Port 3,” on page 144.

The GPT incorporate five 16-bit timers that are grouped into the two timer blocks GPT1 and GPT2. Each timer in each block may operate independently in a number of different modes such as gated timer or counter mode, or may be concatenated with another timer of the same block.

Block 1 contains 3 timers/counters with a maximum resolution of $f_{Timer}/4$. The auxiliary timers of GPT1 may optionally be configured as reload or capture registers for the core timer.

Block 2 contains 2 timers/counters with a maximum resolution of $f_{Timer}/2$. An additional CAPREL register supports capture and reload operation with extended functionality.

The following enumeration summarizes all features to be supported:

- Timer Block 1:
  - $f_{Timer}/4$ maximum resolution.
  - 3 independent timers/counters.
  - Timers/counters can be concatenated.
  - 4 operating modes (timer, gated timer, counter, incremental).
  - Separate interrupt nodes.

- Timer Block 2:
  - $f_{Timer}/2$ maximum resolution.
  - 2 independent timers/counters.
  - Timers/counters can be concatenated.
  - 3 operating modes (timer, gated timer, counter).
  - Extended capture/reload functions via 16-bit Capture/Reload register CAPREL.
  - Separate interrupt nodes.

14.1 Functional Description of Timer Block 1

All three timers of block 1 (T2, T3, T4) can run in 4 basic modes, which are timer, gated timer, counter and incremental interface mode, and all timers can either count up or down.

Each timer has an input line (TxIN) associated with it which serves as the gate control in gated timer mode, or as the count input in counter mode. The count direction (Up / Down) may be programmed via software or may be dynamically altered by a signal at an external control input line. An overflow/underflow of core timer T3 is indicated by the output toggle latch T3OTL whose state may be output on related line T3OUT and on line T3OFL. The auxiliary timers T2 and T4 may additionally be concatenated with the core timer, or used as capture or reload registers for the core timer. Concatenation of T3 with other timers is provided through line T3OFL.

The current contents of each timer can be read or modified by the CPU by accessing the corresponding timer registers T2, T3, or T4, which are located in the non-bitaddressable SFR space. When any of the timer registers is written to by the CPU in the state immediately before a timer increment, decrement, reload, or capture is to be performed, the CPU write operation has priority in order to guarantee correct results.

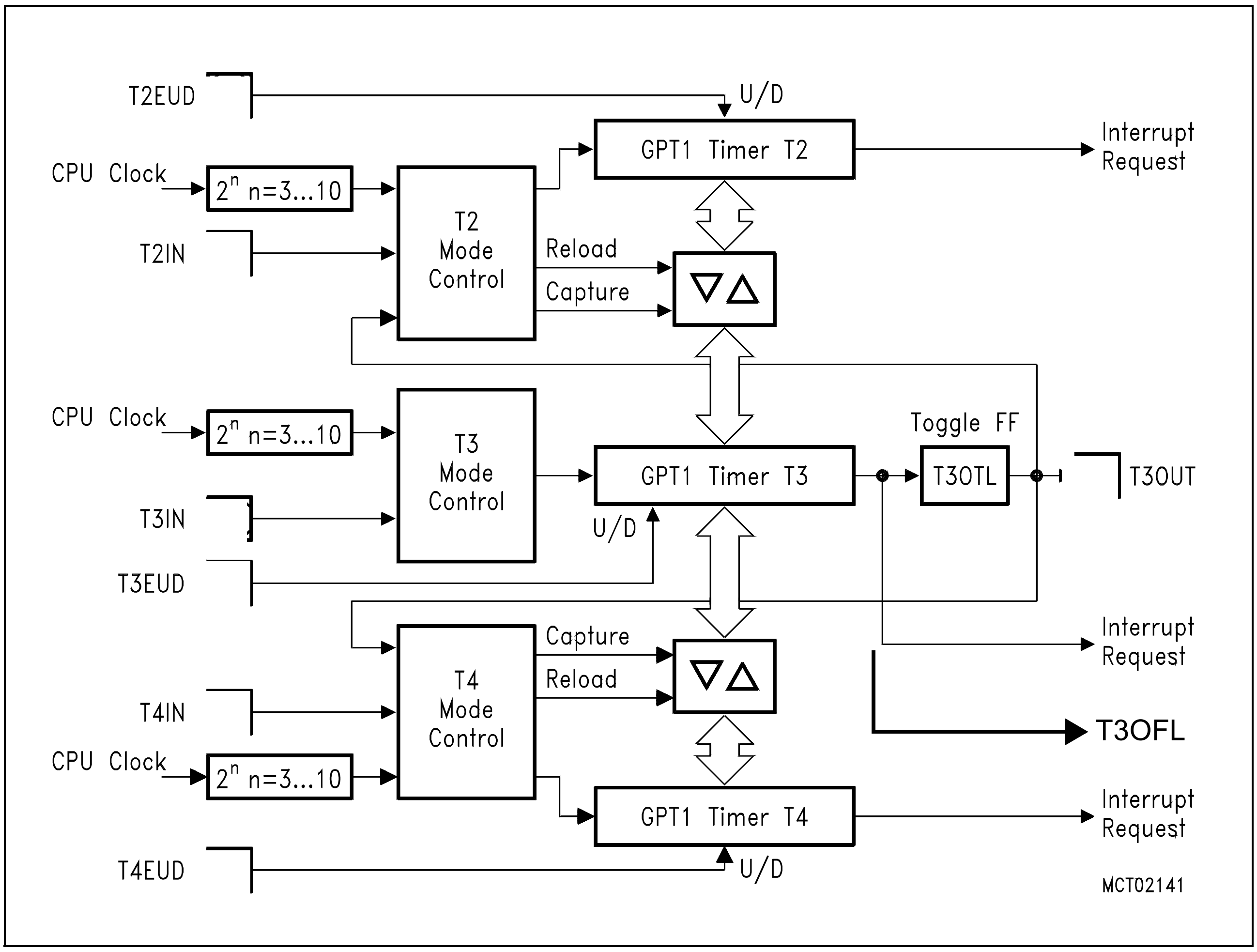


Figure 14-2 Structure of Timer Block 1

14.1.1 Core Timer T3

The operation of the core timer T3 is controlled by its bitaddressable control register T3CON.

Run Control

The timer can be started or stopped by software through bit T3R (Timer T3 Run Bit). Setting bit T3R to '1' will start the timer, clearing T3R stops the timer.

In gated timer mode, the timer will only run if T3R = '1' and the gate is active (high or low, as programmed).

Note: When bit T2RC/T4RC in timer control register T2CON/T4CON is set to '1', T3R will also control (start and stop) auxiliary timer T2/T4.

Count Direction Control

The count direction of the core timer can be controlled either by software or by the external input line T3EUD (Timer T3 External Up/Down Control Input). These options are selected by bits T3UD and T3UDE in control register T3CON. When the up/down control is done by software (bit T3UDE = ‘0’), the count direction can be altered by setting or clearing bit T3UD. When T3UDE = ‘1’, line T3EUD is selected to be the controlling source of the count direction. However, bit T3UD can still be used to reverse the actual count direction, as shown in the table below. If T3UD = ‘0’ and line T3EUD shows a low level, the timer is counting up. With a high level at T3EUD the timer is counting down. If T3UD = ‘1’, a high level at line T3EUD specifies counting up, and a low level specifies counting down. The count direction can be changed regardless of whether the timer is running or not.

When line T3EUD is used as external count direction control input, its associated port pin must be configured as input.

Table 14-1 GPT1 Core Timer T3 Count Direction Control

Line TxEUD	Bit TxUDE	Bit TxUD	Count Direction
X	0	0	Count Up
X	0	1	Count Down
0	1	0	Count Up
1	1	0	Count Down
0	1	1	Count Down
1	1	1	Count Up

Note: The direction control works the same for core timer T3 and for auxiliary timers T2 and T4. Therefore the lines and bits are named Tx...

Timer 3 Overflow/Underflow Monitoring

An overflow or underflow of timer T3 will clock the overflow toggle latch T3OTL in control register T3CON. T3OTL can also be set or reset by software. Bit T3OE (Overflow/ Underflow Output Enable) in register T3CON enables the state of T3OTL to be monitored via an external line T3OUT. If this line is configured as output, T3OTL can be used to control external HW.

In addition, T3OTL can be used in conjunction with the timer over/underflows as an input for the counter function or as a trigger source for the reload function of the auxiliary timers T2 and T4. For this purpose, the state of T3OTL does not have to be available at any port pin, because an internal connection is provided for this option.

An overflow or underflow of timer T3 can also be used to clock other timers. For this purpose, there is the special output line T3OFL.

Timer 3 in Timer Mode

Timer mode for the core timer T3 is selected by setting bit field T3M in register T3CON to ‘000_B’.

In this mode, T3 is clocked with the module clock f_{Timer} divided by a programmable prescaler, which is controlled by bit field T3I and bit FM1. The input frequency f_{T3} for timer T3 and its resolution r_{T3} are scaled linearly with lower module clock frequencies, as can be seen from the following formula:

$$f_{T3} = \frac{f_{Timer}}{8 * 2^{<T3I - FM1>}} \qquad r_{T3}\,[ms] = \frac{8 * 2^{<T3I - FM1>}}{f_{Timer}\,[MHz]}$$

Table 14-2 Example for Timer 3 Frequencies and Resolutions

f_{Timer} [MHz]	**T3I**	**FM1**	f_{T3} [KHz]	r_{T3} [ms]
24	7	0	23.44	42.67
24	0	1	6000.0	0.17
36	0	0	4500.0	0.22
36	4	0	281.25	3.55
36	7	1	70.31	14.22

This formula also applies to the Gated Timer Mode of T3 and to the auxiliary timers T2 and T4 in timer and gated timer mode.

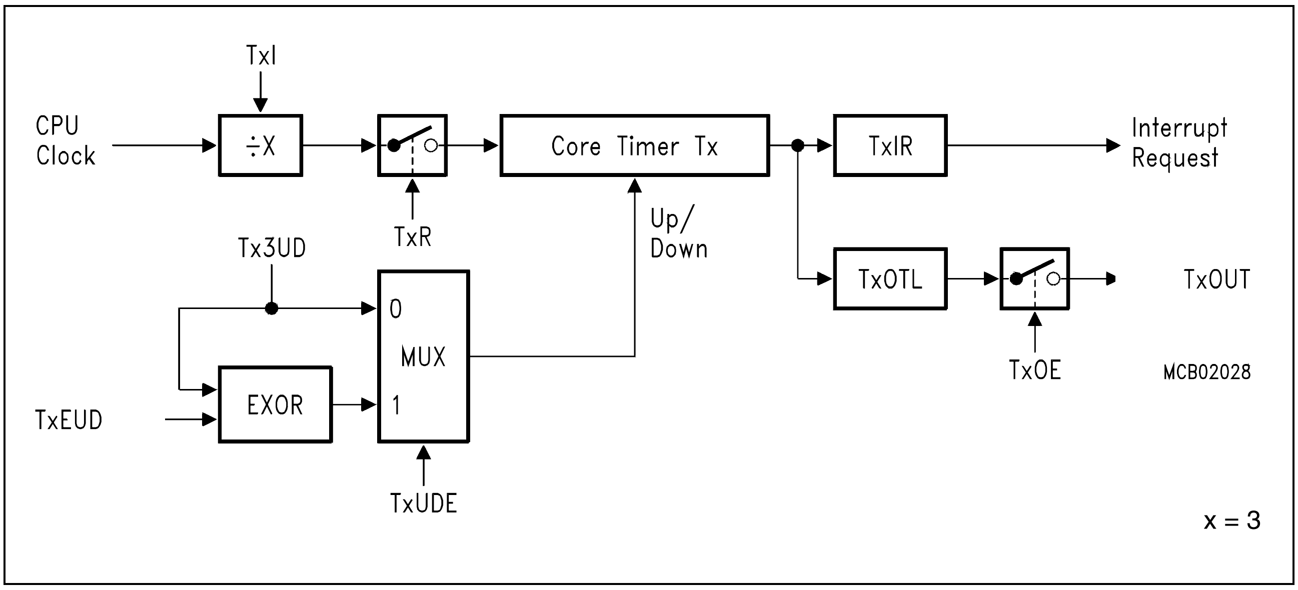


Figure 14-3 Block Diagram of Core Timer T3 in Timer Mode

Timer 3 in Gated Timer Mode

Gated timer mode for the core timer T3 is selected by setting bit field T3M in register T3CON to ‘010_B’ or ‘011_B’.

Bit T3M.0 (T3CON.3) selects the active level of the gate input. In gated timer mode the same options for the input frequency as for the timer mode are available. However, the input clock to the timer in this mode is gated by the external input line T3IN (Timer T3 External Input), which is an alternate function of P3.6.

To enable this operation pin P3.6/T3IN must be configured as input, ie. direction control bit DP3.6 must contain ’0’.

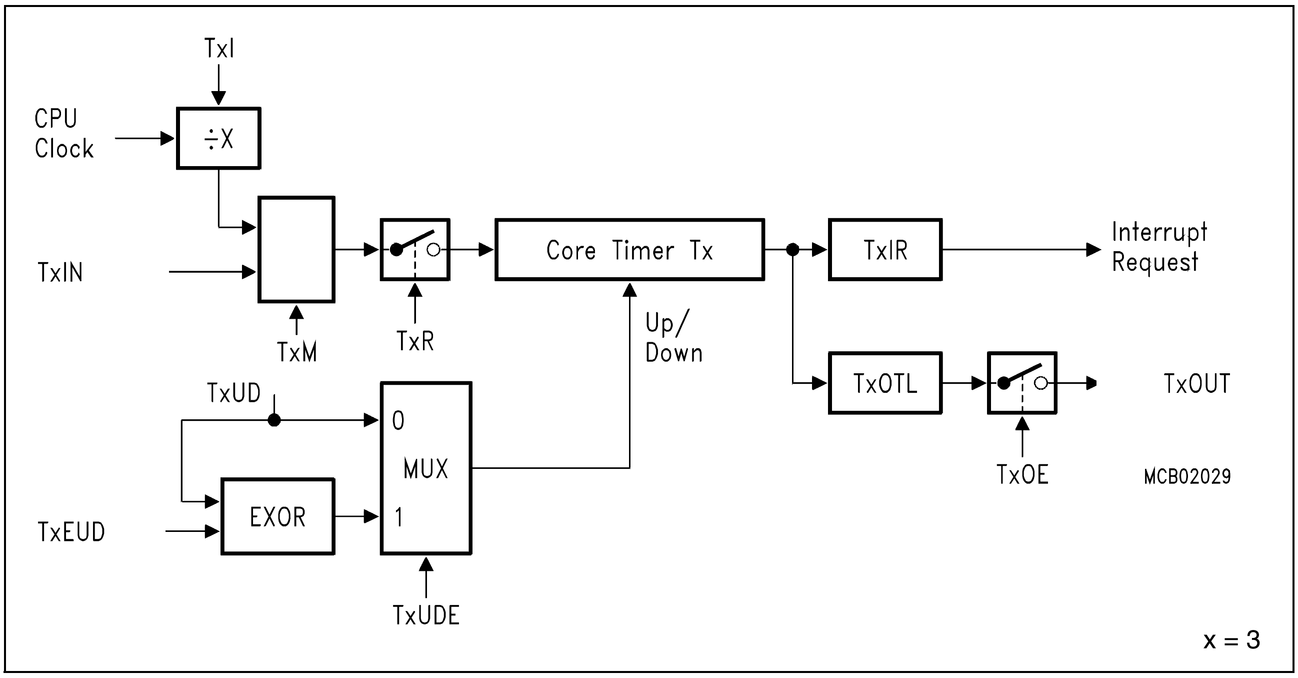


Figure 14-4 Block Diagram of Core Timer T3 in Gated Timer Mode

If T3M = ‘010_B’, the timer is enabled when T3IN shows a low level. A high level at this line stops the timer. If T3M = ‘011_B’, line T3IN must have a high level in order to enable the timer. In addition, the timer can be turned on or off by software using bit T3R. The timer will only run, if T3R = ‘1’ and the gate is active. It will stop, if either T3R = ‘0’ or the gate is inactive.

Note: A transition of the gate signal at line T3IN does not cause an interrupt request.

Timer 3 in Counter Mode

Counter mode for the core timer T3 is selected by setting bit field T3M in register T3CON to ‘001_B’. In counter mode timer T3 is clocked by a transition at the external input pin T3IN, which is an alternate function of P3.6. The event causing an increment or decrement of the timer can be a positive, a negative, or both a positive and a negative transition at this line. Bit field T3I in control register T3CON selects the triggering transition (see **Table 14-3** below).

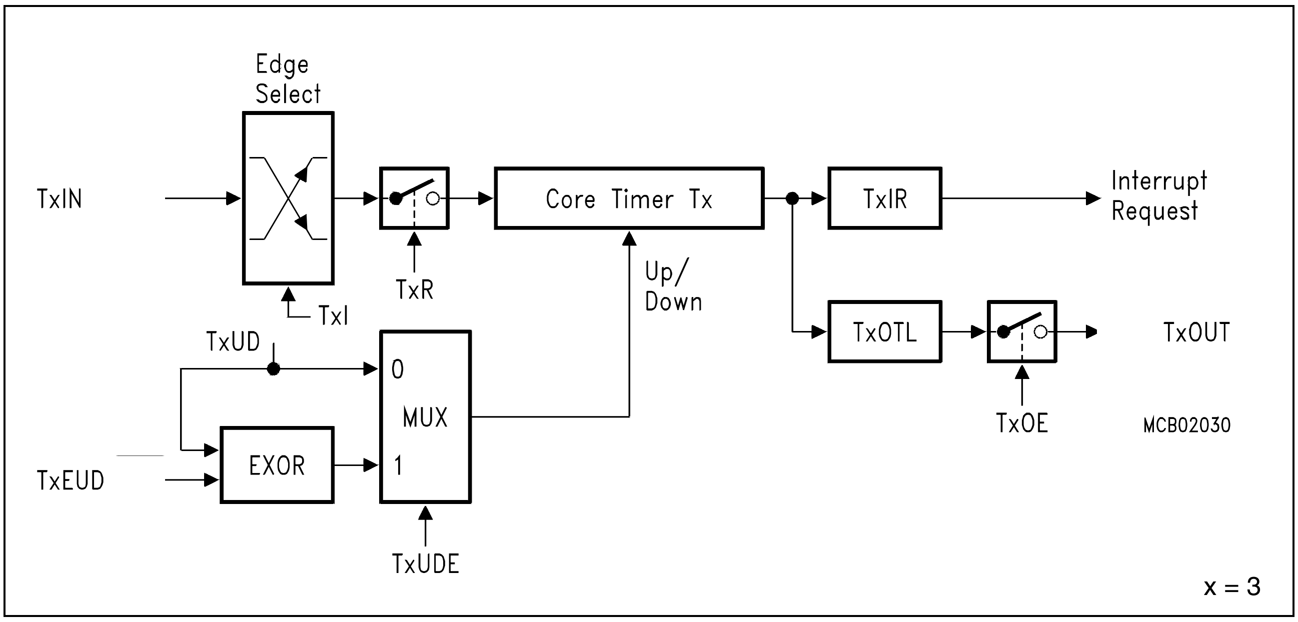


Figure 14-5 Block Diagram of Core Timer T3 in Counter Mode

Table 14-3 Core Timer T3 (Counter Mode) Input Edge Selection

T3I	Triggering Edge for Counter Increment / Decrement
0 0 0	None. Counter T3 is disabled
0 0 1	Positive transition (rising edge) on T3IN
0 1 0	Negative transition (falling edge) on T3IN
0 1 1	Any transition (rising or falling edge) on T3IN
1 X X	Reserved. Do not use this combination

For counter operation, a port pin P3.6/T3IN must be configured as input. The maximum input frequency which is allowed in counter mode is $f_{Timer}/8$ (FM1 = '1'). To ensure that a transition of the count input signal which is applied to T3IN is correctly recognized, its level should be held high or low for at least 4 f_{Timer} cycles (FM1 = '1') before it changes.

Timer 3 in Incremental Interface Mode

Incremental Interface mode for the core timer T3 is selected by setting bit field T3M in register T3CON to '110_B' or '111_B'. In incremental interface mode pin P3.6/T3IN (configured as timer input T3IN) and pin P3.5/T3EUD (configured as timer input) are used to interface to an incremental encoder.

Note: In the INCA-D, the T3EUD timer input is connected to P3.5. In this case, Timer 4 input T4IN can be used only by Software.

T3 is clocked by each transition on one or both of the external input lines which gives 2-fold or 4-fold resolution of the encoder input.

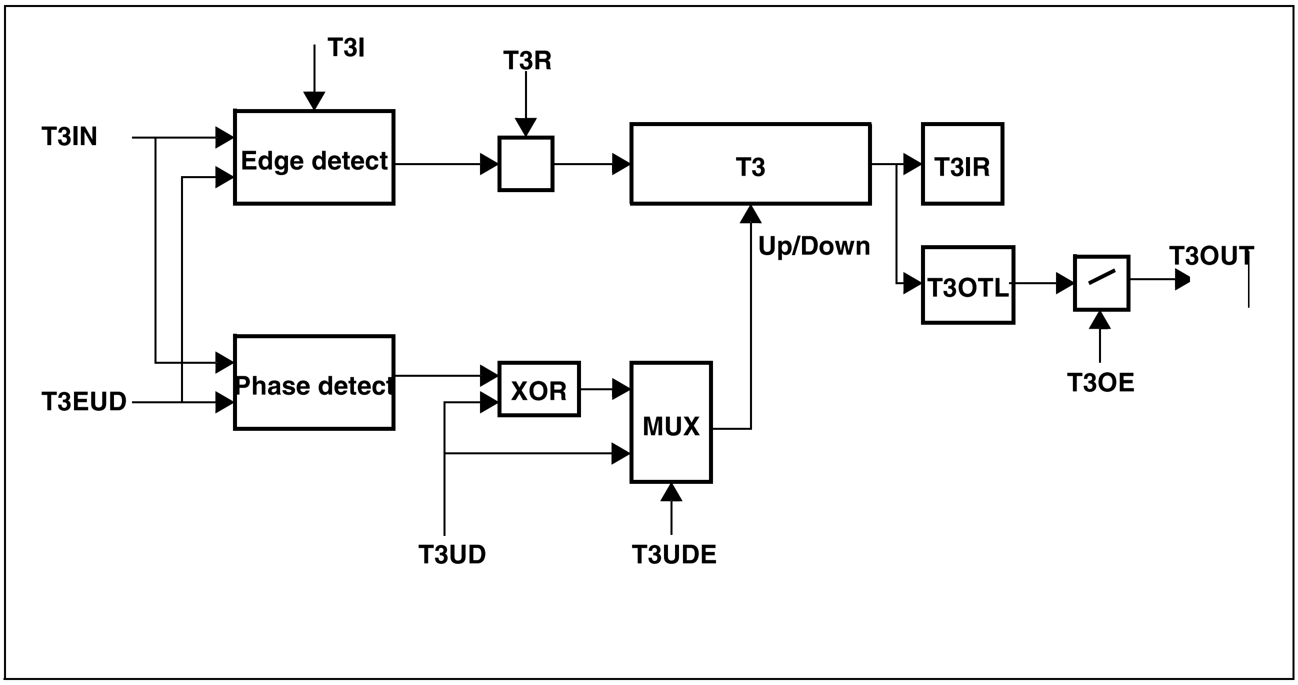


Figure 14-6 Block Diagram of Core Timer T3 in Incremental Interface Mode

Bit field T3I in control register T3CON selects the triggering transitions (see table below). In this mode the sequence of the transitions of the two input signals is evaluated and generates count pulses as well as the direction signal. Depending on the chosen Incremental Intrerface Mode, Rotation detection ‘110_B’ or Edge Detection ‘111_B’, an interrupt can be generated. This interrupt is only generated if it’s enabled by setting bit T3IREN in register T3CON. For the Rotation detection an interrupt will be generated each time the count direction of timer 3 changes. For the Edge detection an interrupt will be generated each time a count action for timer 3 occurs. Count direction, changes in the count direction and count requests are monitored through the status bits T3RDIR, T3CHDIR and T3EDGE in register T3CON. T3 is modified automatically according to the speed and the direction of the incremental encoder. Therefore, the contents of timer T3 always represents the encoder’s current position.

Table 14-4 Core Timer T3 (Incremental Interface Mode) Input Edge Selection

T3I	Triggering Edge for Counter Increment / Decrement
0 0 0	None. Counter T3 stops.
0 0 1	Any transition (rising or falling edge) on T3IN.
0 1 0	Any transition (rising or falling edge) on T3EUD.

Table 14-4 Core Timer T3 (Incremental Interface Mode) Input Edge Selection

T3I	Triggering Edge for Counter Increment / Decrement
0 1 1	Any transition (rising or falling edge) on any T3 input (T3IN or T3EUD).
1 X X	Reserved. Do not use this combination

The incremental encoder can be connected directly to the microcontroller without external interface logic. In a standard system, however, comparators will be employed to convert the encoder's differential outputs (e.g. A, $\overline{A}$) to digital signals (e.g. A). This greatly increases noise immunity.

Note: The third encoder output T0, which indicates the mechanical zero position, may be connected to an external interrupt input and trigger a reset of timer T3.

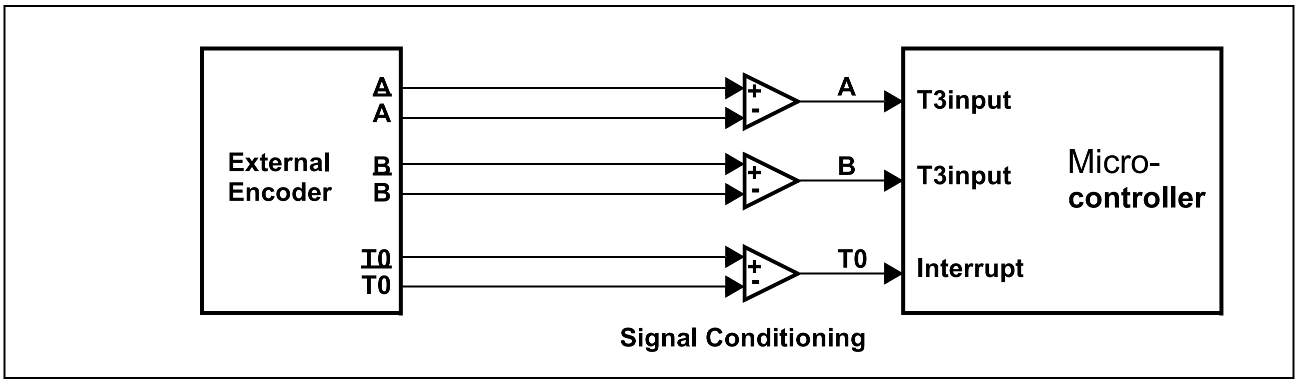


Figure 14-7 Interfacing the Encoder to the Microcontroller

For incremental interface operation the following conditions must be met:

- Bitfield T3M must be '110_B' or '111_B'.
- Pins associated to lines T3IN and T3EUD must be configured as input.
- Bit T3UDE must be '1' to enable automatic direction control.

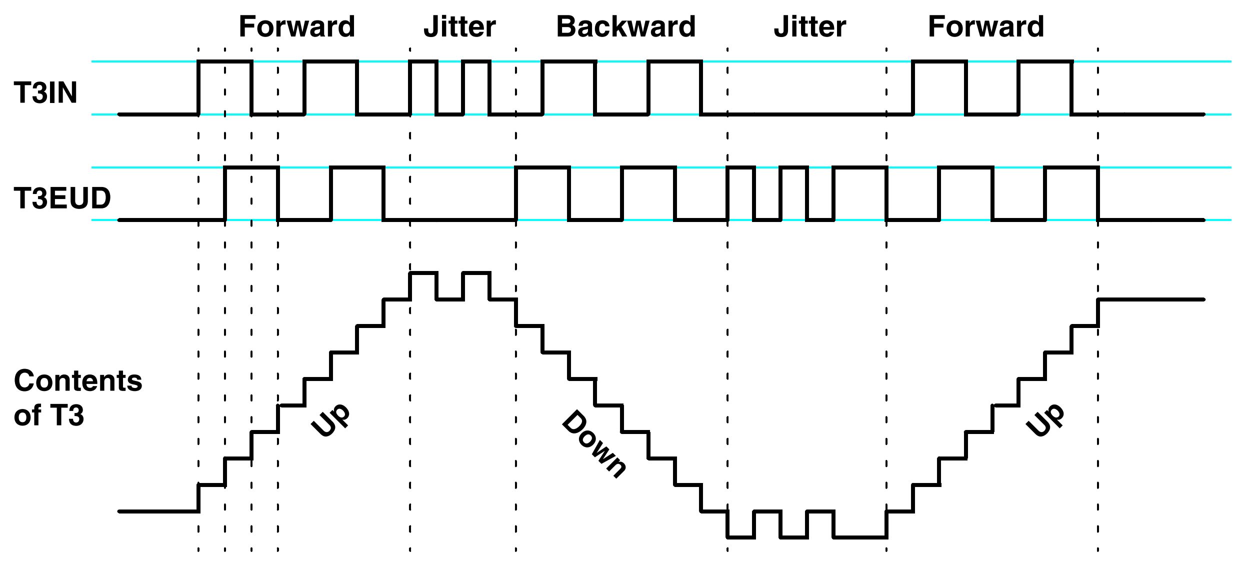
The maximum input frequency which is allowed in incremental interface mode is $f_{Timer}/8$ (FM = 1). To ensure that a transition of any input signal is correctly recognized, its level should be held high or low for at least 4 f_{Timer} cycles (FM = 1) before it changes.

In Incremental Interface Mode the count direction is automatically derived from the sequence in which the input signals change, which corresponds to the rotation direction of the connected sensor. The table below summarizes the possible combinations.

The figures below give examples of T3's operation, visualizing count signal generation and direction control. It also shows how input jitter is compensated which might occur if the sensor rests near to one of its switching points.

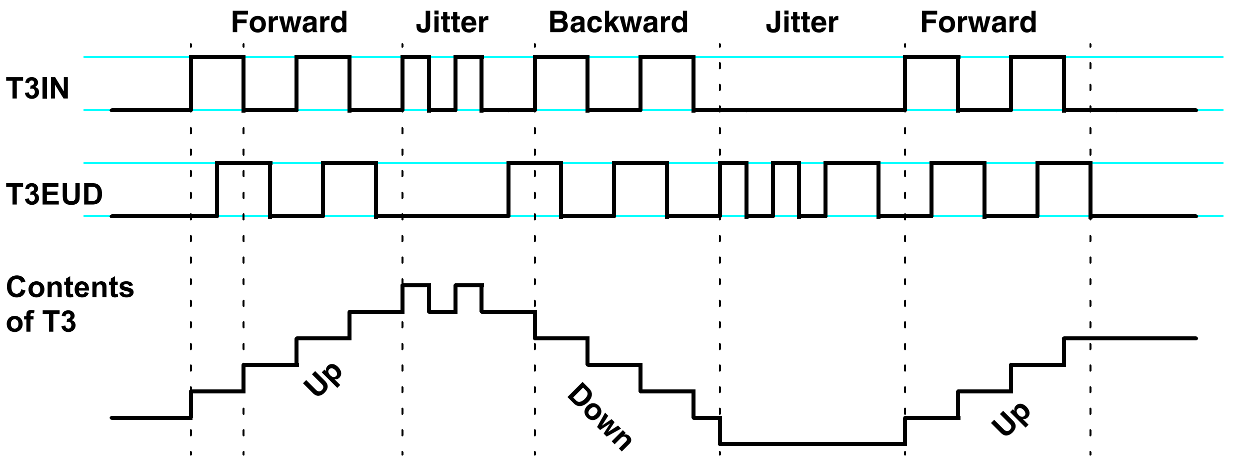
Table 14-5 Core Timer T3 (Incremental Interface Mode) Count Direction

Level on respective other input	T3IN Input		T3EUD Input	
	Rising ↗	Falling ↘	Rising ↗	Falling ↘
High	Down	Up	Up	Down
Low	Up	Down	Down	Up



Note: This example shows the timer behavior assuming that T3 counts upon any transition on any input, i.e. T3I = '011_B'.

Figure 14-8 Evaluation of the Incremental Encoder Signals



Note: This example shows the timer behavior assuming that T3 counts upon any transition on input T3IN, i.e. T3I = ’001_B’.

Figure 14-9 Evaluation of the Incremental Encoder Signals

Note: Timer T3 operating in incremental interface mode automatically provides information on the sensor’s current position. Dynamic information (speed, acceleration, deceleration) may be obtained by measuring the incoming signal periods.

14.1.2 Auxiliary Timers T2 and T4

Both auxiliary timers T2 and T4 have exactly the same functionality. They can be configured for timer, gated timer, counter, or incremental interface mode with the same options for the timer frequencies and the count signal as the core timer T3. In addition to these 4 counting modes, the auxiliary timers can be concatenated with the core timer, or they may be used as reload or capture registers in conjunction with the core timer.

The individual configuration for timers T2 and T4 is determined by their bitaddressable control registers T2CON and T4CON, which are both organized identically. Note that functions which are present in all 3 timers of timer block 1 are controlled in the same bit positions and in the same manner in each of the specific control registers.

Run control for auxiliary timers T2 and T4 can be handled by the associated Run Control Bit T2R, T4R in register T2CON/T4CON. Alternatively, a remote control option (T2RC, T4RC = ’1’) may be enabled to start and stop T2/T4 via the run bit T3R of core timer T3.

Timers T2 and T4 in Timer Mode or Gated Timer Mode

When the auxiliary timers T2 and T4 are programmed to timer mode or gated timer mode, their operation is the same as described for the core timer T3. The descriptions, figures and tables apply accordingly with two exceptions:

- There is no TxOUT output line for T2 and T4.
- Overflow/Underflow Monitoring is not supported (no output toggle latch).

Timers T2 and T4 in Counter Mode

In counter mode timers T2 and T4 can be clocked either by a transition at the respective external input line TxIN, or by a transition of timer T3's output toggle latch T3OTL.

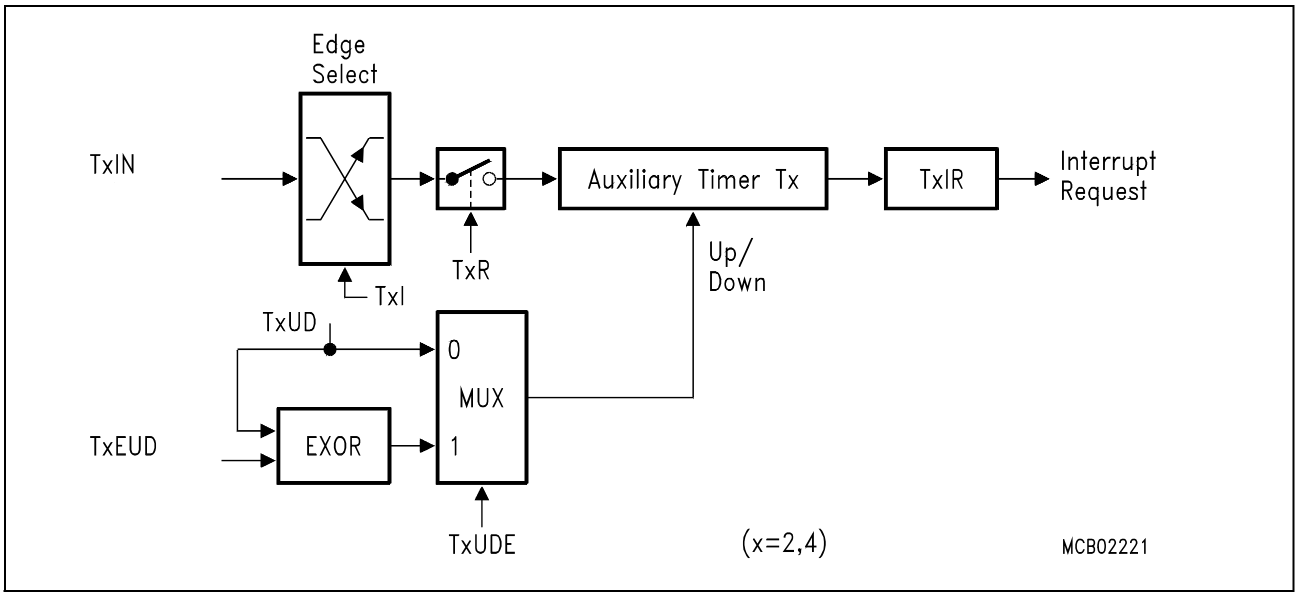


Figure 14-10 Block Diagram of an Auxiliary Timer in Counter Mode

The event causing an increment or decrement of a timer can be a positive, a negative, or both a positive and a negative transition at either the respective input line, or at the output toggle latch T3OTL.

Bit field TxI in the respective control register TxCON selects the triggering transition (see table below).

Table 14-6 Auxiliary Timer (Counter Mode) Input Edge Selection

T2I / T4I	Triggering Edge for Counter Increment / Decrement
X 0 0	None. Counter Tx is disabled
0 0 1	Positive transition (rising edge) on TxIN
0 1 0	Negative transition (falling edge) on TxIN
0 1 1	Any transition (rising or falling edge) on TxIN
1 0 1	Positive transition (rising edge) of output toggle latch T3OTL
1 1 0	Negative transition (falling edge) of output toggle latch T3OTL
1 1 1	Any transition (rising or falling edge) of output toggle latch T3OTL

Note: Only state transitions of T3OTL which are caused by the overflows/underflows of T3 will trigger the counter function of T2/T4. Modifications of T3OTL via software will NOT trigger the counter function of T2/T4.

The maximum input frequency which is allowed in counter mode is $f_{\text{Timer}}/8$ (FM1 = ’1’). To ensure that a transition of the count input signal which is applied to TxIN is correctly recognized, its level should be held for at least 4 f_{Timer} cycles (FM1 = ’1’) before it changes.

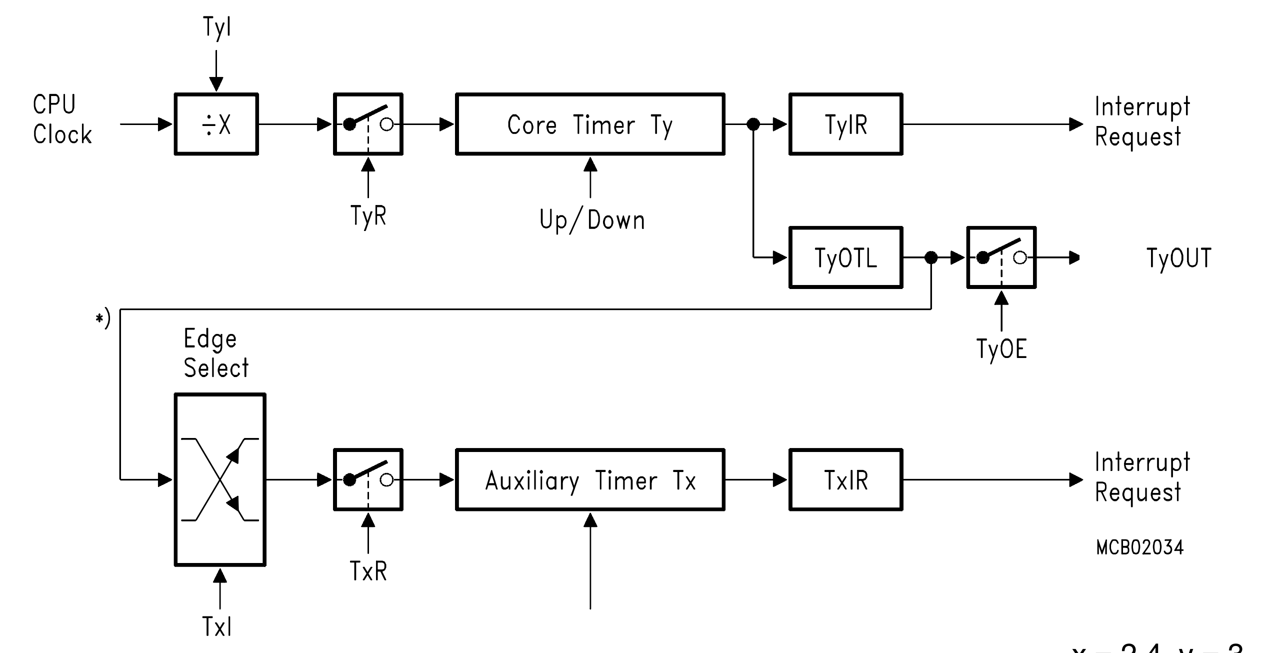
Timer Concatenation

Using the output toggle latch T3OTL as a clock source for an auxiliary timer in counter mode concatenates the core timer T3 with the respective auxiliary timer. Depending on which transition of T3OTL is selected to clock the auxiliary timer, this concatenation forms a 32-bit or a 33-bit timer/counter.

- **32-bit Timer/Counter**: If both a positive and a negative transition of T3OTL is used to clock the auxiliary timer, this timer is clocked on every overflow/underflow of the core timer T3. Thus, the two timers form a 32-bit timer.
- **33-bit Timer/Counter**: If either a positive or a negative transition of T3OTL is selected to clock the auxiliary timer, this timer is clocked on every second overflow/underflow of the core timer T3. This configuration forms a 33-bit timer (16-bit core timer+T3OTL+16-bit auxiliary timer).

The count directions of the two concatenated timers are not required to be the same. This offers a wide variety of different configurations.

T3 can operate in timer, gated timer or counter mode in this case.



x = 2,4 y = 3

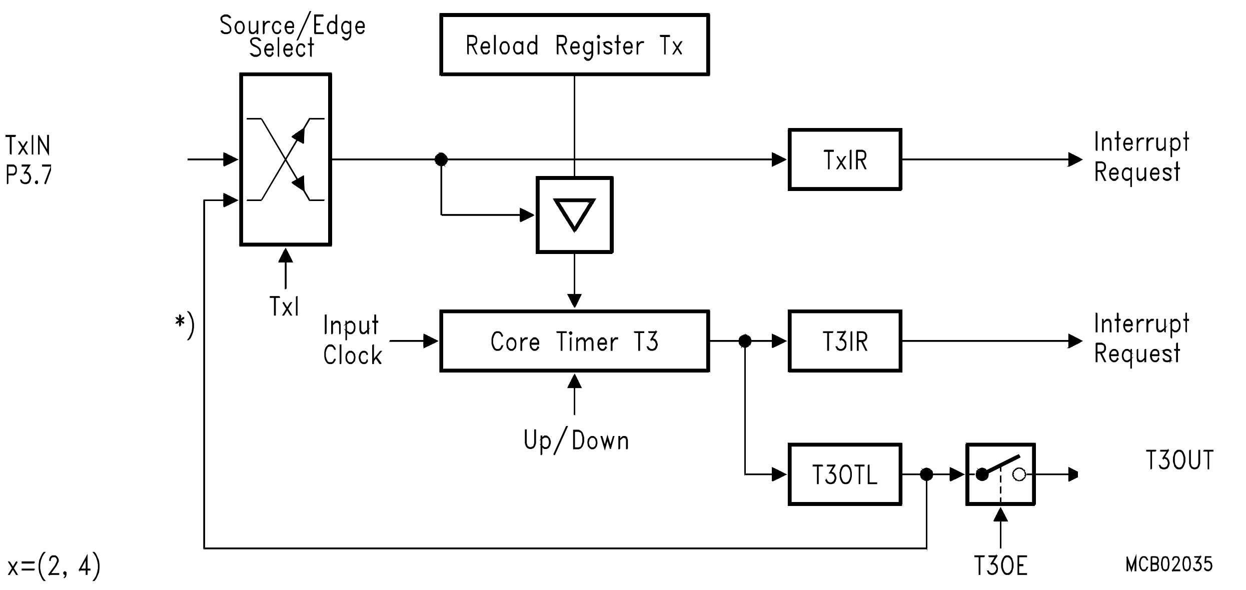
Note: Line '*' only affected by over/underflows of T3, but NOT by software modifications of T3OTL.

Figure 14-11 Concatenation of Core Timer T3 and an Auxiliary Timer

Auxiliary Timer in Reload Mode

Reload mode for the auxiliary timers T2 and T4 is selected by setting bit field TxM in the respective register TxCON to '100_B'. In reload mode the core timer T3 is reloaded with the contents of an auxiliary timer register, triggered by one of two different signals. The trigger signal is selected the same way as the clock source for counter mode (see table above), i.e. a transition of the auxiliary timer's input or the output toggle latch T3OTL may trigger the reload.

Note: When programmed for reload mode, the respective auxiliary timer (T2 or T4) stops independent of its run flag T2R or T4R.



Note: Line '*' only affected by over/underflows of T3, but NOT by software modifications of T3OTL.

Figure 14-12 GPT1 Auxiliary Timer in Reload Mode

Upon a trigger signal T3 is loaded with the contents of the respective timer register (T2 or T4) and the interrupt request flag (T2IR or T4IR) is set.

Note: When a T3OTL transition is selected for the trigger signal, also the interrupt request flag T3IR will be set upon a trigger, indicating T3's overflow or underflow. Modifications of T3OTL via software will NOT trigger the counter function of T2/T4.

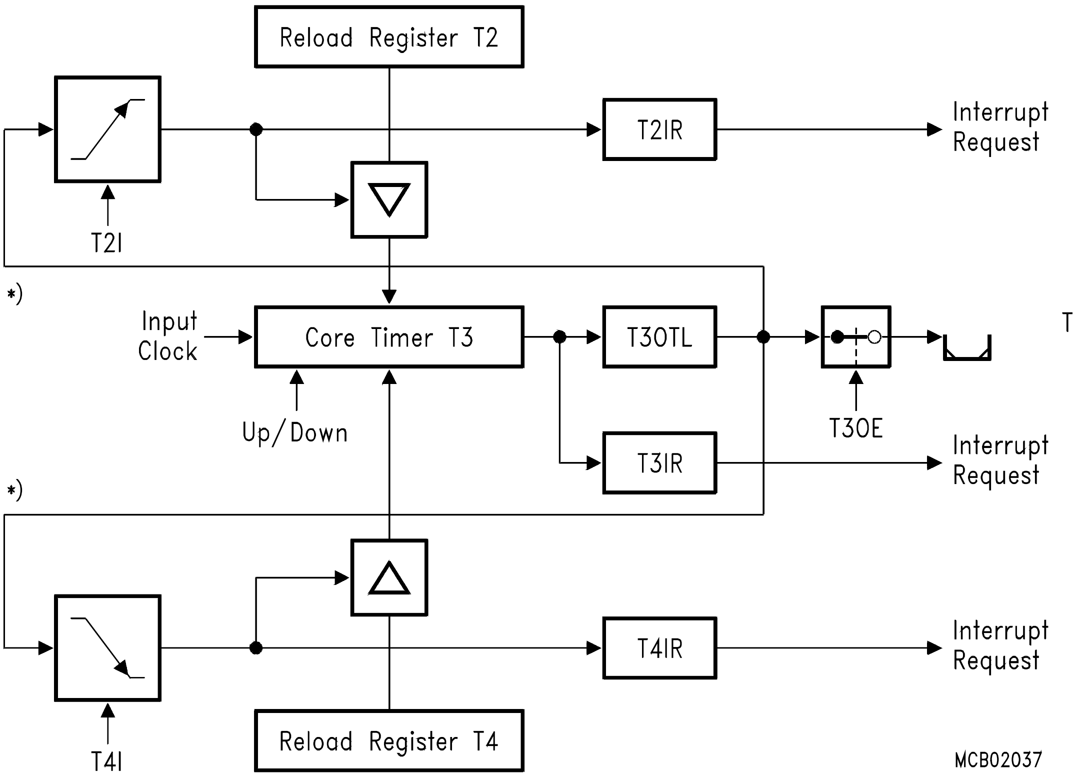
The reload mode triggered by T3OTL can be used in a number of different configurations. Depending on the selected active transition the following functions can be performed:

- If both a positive and a negative transition of T3OTL is selected to trigger a reload, the core timer will be reloaded with the contents of the auxiliary timer each time it overflows or underflows. This is the standard reload mode (reload on overflow/underflow).
- If either a positive or a negative transition of T3OTL is selected to trigger a reload, the core timer will be reloaded with the contents of the auxiliary timer on every second overflow or underflow.
- Using this "single-transition" mode for both auxiliary timers allows to perform very flexible pulse width modulation (PWM). One of the auxiliary timers is programmed to reload the core timer on a positive transition of T3OTL, the other is programmed for a reload on a negative transition of T3OTL. With this combination the core timer is alternately reloaded from the two auxiliary timers.

The figure below shows an example for the generation of a PWM signal using the alternate reload mechanism. T2 defines the high time of the PWM signal (reloaded on positive transitions) and T4 defines the low time of the PWM signal (reloaded on negative transitions). The PWM signal can be output on line T3OUT if the control bit T3OE is set to ‘1’. With this method the high and low time of the PWM signal can be varied in a wide range.

Note: The output toggle latch T3OTL is accessible via software and may be changed, if required, to modify the PWM signal. However, this will NOT trigger the reloading of T3.

Note: An associated port pin linked to line T3OUT should be configured as output.



Note: Lines ’*’ only affected by over/underflows of T3, but NOT by software modifications of T3OTL.

Figure 14-13 GPT1 Timer Reload Configuration for PWM Generation

Note: Although it is possible, it should be avoided to select the same reload trigger event for both auxiliary timers. In this case both reload registers would try to load the core timer at the same time. If this combination is selected, T2 is disregarded and the contents of T4 is reloaded.

Auxiliary Timer in Capture Mode

Capture mode for the auxiliary timers T2 and T4 is selected by setting bit field TxM in the respective register TxCON to ‘101_B’. In capture mode the contents of the core timer are latched into an auxiliary timer register in response to a signal transition at the respective auxiliary timer's external input line TxIN. The capture trigger signal can be a positive, a negative, or both a positive and a negative transition.

The two least significant bits of bit field TxI are used to select the active transition (see table in the counter mode section), while the most significant bit TxI.2 is irrelevant for capture mode. It is recommended to keep this bit cleared (TxI.2 = ‘0’).

Note: When programmed for capture mode, the respective auxiliary timer (T2 or T4) stops independent of its run flag T2R or T4R.

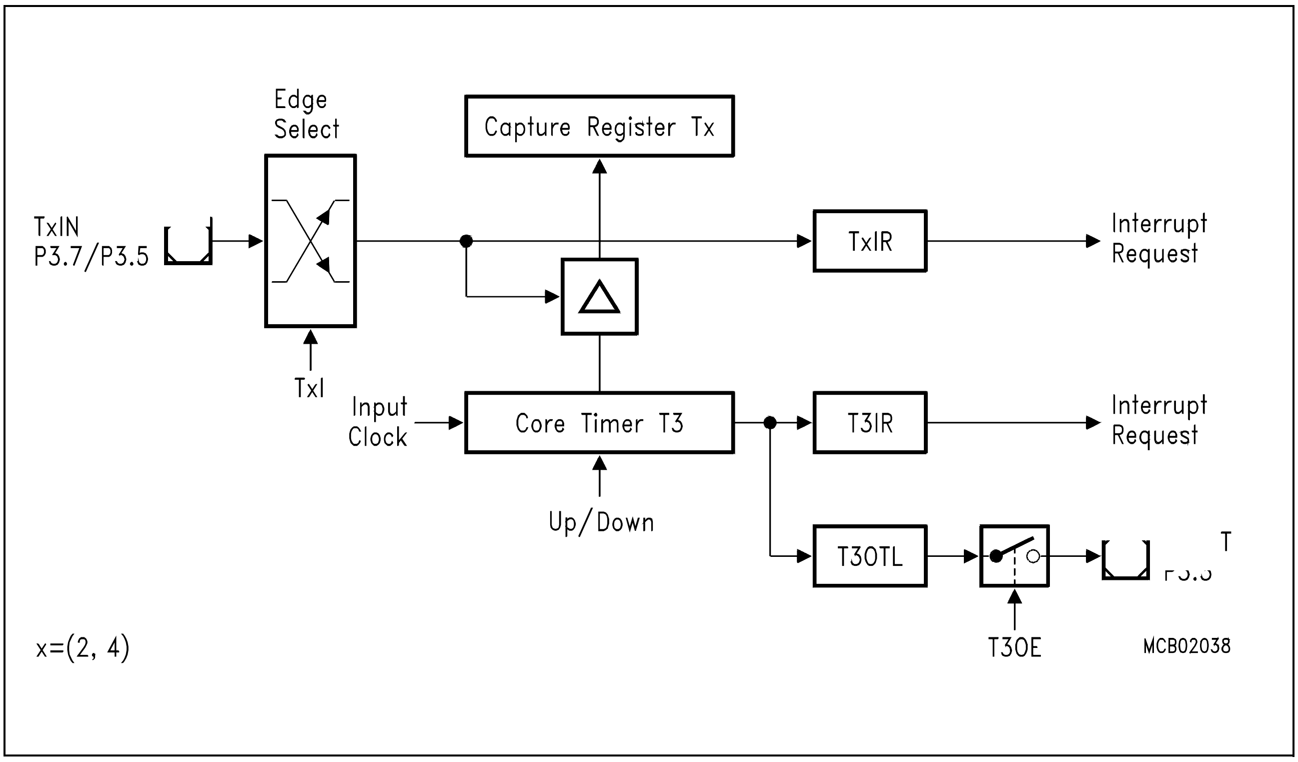


Figure 14-14 Auxiliary Timer of Timer Block 1 in Capture Mode

Upon a trigger (selected transition) at the corresponding input line TxIN the contents of the core timer are loaded into the auxiliary timer register and the associated interrupt request flag TxIR will be set.

Note: The direction control for T2IN and for T4IN must be set to 'Input', and the level of the capture trigger signal should be held high or low for at least 4 f_{Timer} (FM1 = ’1’) cycles before it changes to ensure correct edge detection.

14.2 Functional Description of Timer Block 2

Timer block 2 includes the two timers T5 (referred to as the auxiliary timer) and T6 (referred to as the core timer), and the 16-bit capture/reload register CAPREL.

Each timer has an input line (TxIN) associated with it which serves as the gate control in gated timer mode, or as the count input in counter mode. The count direction (Up / Down) may be programmed via software or may be dynamically altered by a signal at an external control input line. An overflow/underflow of core timer T6 is indicated by the output toggle latch T6OTL whose state may be output on related line T6OUT and on line T6OFL. The auxiliary timer T6 may be reloaded with the contents of CAPREL.

The toggle bit also supports the concatenation of T6 with auxiliary timer T5, while concatenation of T6 with other timers is provided through line T6OFL. Triggered by an external signal, the contents of T5 can be captured into register CAPREL, and T5 may optionally be cleared. Both timer T6 and T5 can count up or down, and the current timer value can be read or modified by the CPU in the non-bitaddressable SFRs T5 and T6.

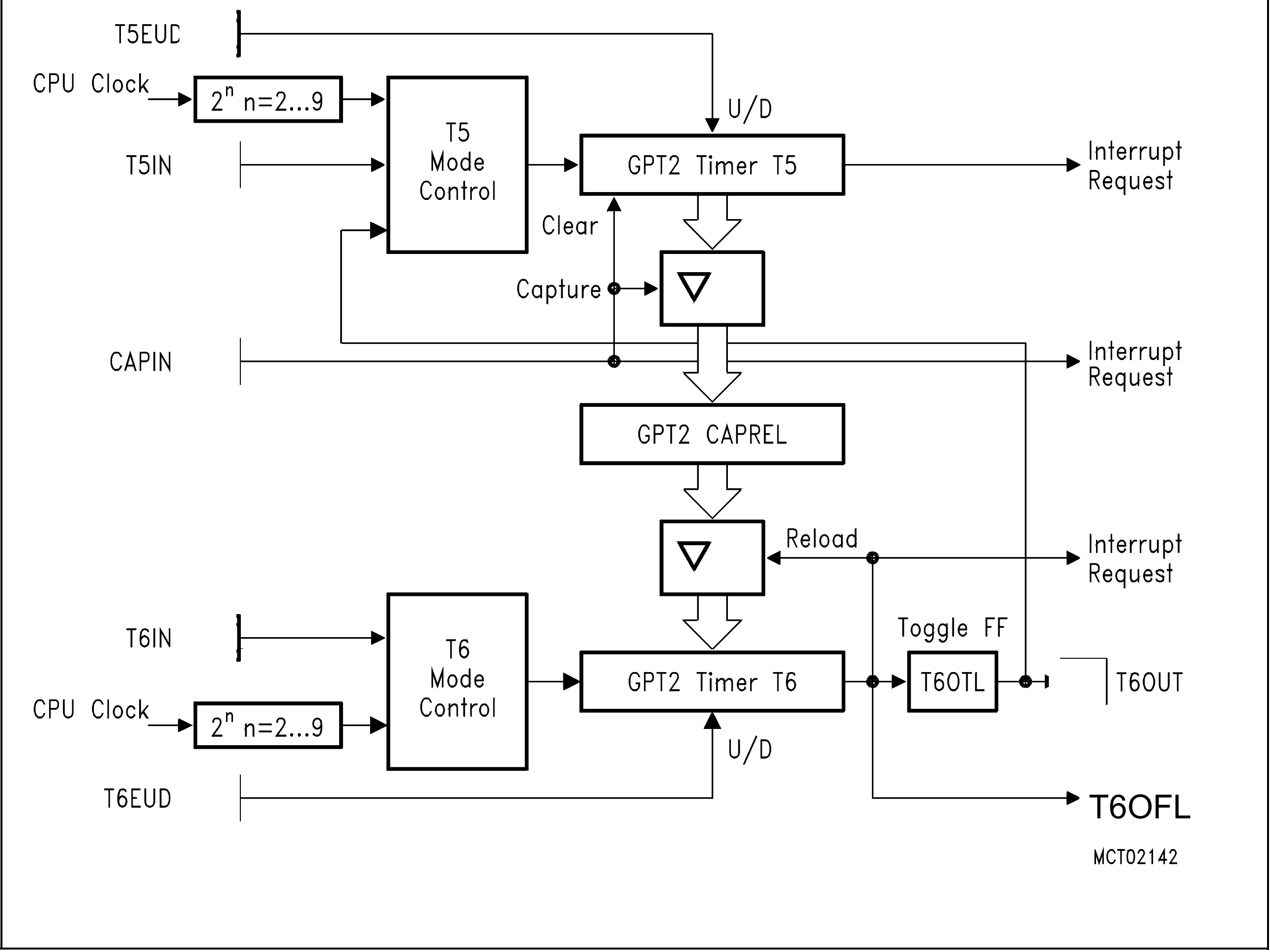


Figure 14-15 Structure of Timer Block 2

14.2.1 Core Timer T6

The operation of the core timer T6 is controlled by its bitaddressable control register T6CON.

Timer 6 Run Bit

The timer can be started or stopped by software through bit T6R (Timer T6 Run Bit). Setting bit T6R to ‘1’ will start the timer, clearing T6R stops the timer.

In gated timer mode, the timer will only run if T6R = ‘1’ and the gate is active (high or low, as programmed).

Note: When bit T5RC = ’1’ bit T6R will also control (start and stop) auxiliary timer T5.

Count Direction Control

The count direction of the core timer can be controlled either by software or by the external input line T6EUD (Timer T6 External Up/Down Control Input). These options are selected by bits T6UD and T6UDE in control register T6CON. When the up/down control is done by software (bit T6UDE = ‘0’), the count direction can be altered by setting or clearing bit T6UD. When T6UDE = ‘1’, line T6EUD is selected to be the controlling source of the count direction. However, bit T6UD can still be used to reverse the actual count direction, as shown in the table below. If T6UD = ‘0’ and line T6EUD shows a low level, the timer is counting up. With a high level at T6EUD the timer is counting down. If T6UD = ‘1’, a high level at line T6EUD specifies counting up, and a low level specifies counting down. The count direction can be changed regardless of whether the timer is running or not.

Table 14-7 Core Timer T6 Count Direction Control

Line TxEUD	Bit TxUDE	Bit TxUD	Count Direction
X	0	0	Count Up
X	0	1	Count Down
0	1	0	Count Up
1	1	0	Count Down
0	1	1	Count Down
1	1	1	Count Up

Note: The direction control works the same for core timer T6 and for auxiliary timer T5. Therefore the lines and bits are named Tx...

Timer 6 Overflow/Underflow Monitoring

An overflow or underflow of timer T6 will clock the toggle latch T6OTL in control register T6CON. T6OTL can also be set or reset by software. Bit T6OE (Overflow/Underflow Output Enable) in register T6CON enables the state of T6OTL to be monitored via the external output line T6OUT. An associated port pin must be configured as output.

In addition, T6OTL can be used in conjunction with the timer over/underflows as an input for the counter function of the auxiliary timer T5. For this purpose, the state of T6OTL does not have to be available at line T6OUT, because an internal connection is provided for this option.

An overflow or underflow of timer T6 can also be used to clock other timers. For this purpose, there is the special output line T6OFL.

Timer 6 in Timer Mode

Timer mode for the core timer T6 is selected by setting bit field T6M in register T6CON to '000_B'. In this mode, T6 is clocked with the module clock divided by a programmable prescaler, which is selected by bit field T6I. The input frequency f_{T6} for timer T6 and its resolution r_{T6} are scaled linearly with lower clock frequencies f_{Timer}, as can be seen from the following formula:

$$f_{T6} = \frac{f_{Timer}}{4 * 2^{<T6I\text{ - }FM2>}} \qquad r_{T6}\,[\mu s] = \frac{4 * 2^{<T6I\text{ - }FM2>}}{f_{Timer}\,[MHz]}$$

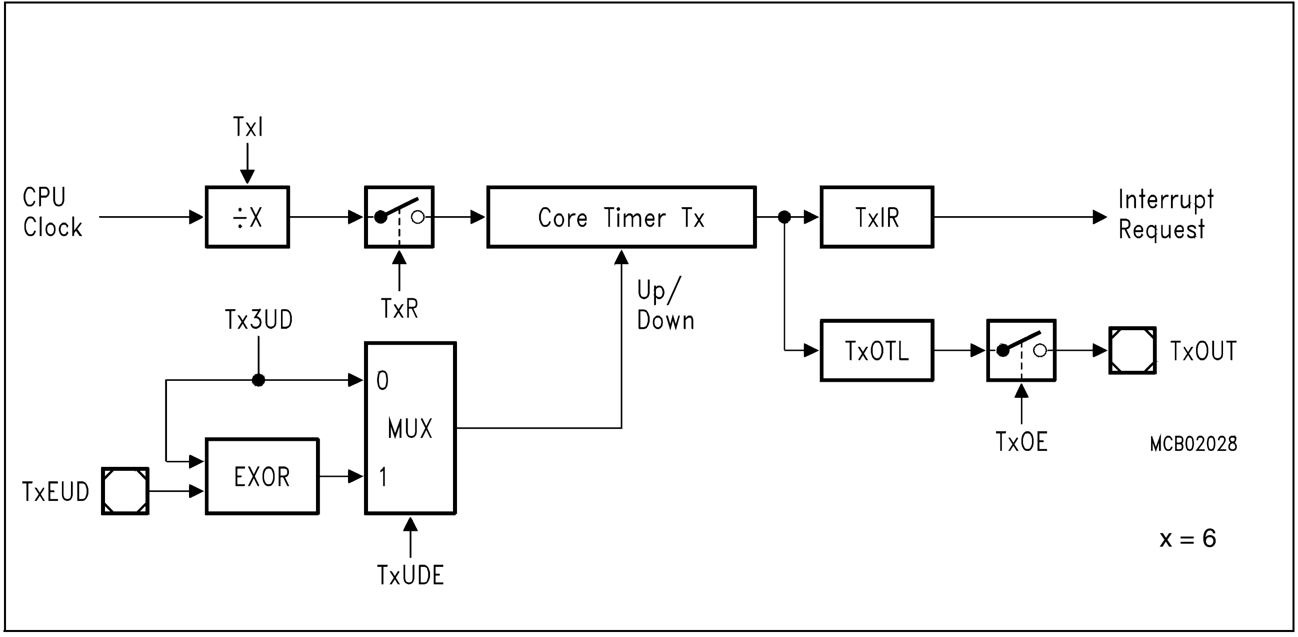


Figure 14-16 Block Diagram of Core Timer T6 in Timer Mode

Timer 6 in Gated Timer Mode

Gated timer mode for the core timer T6 is selected by setting bit field T6M in register T6CON to ‘010_B’ or ‘011_B’. Bit T6M.0 (T6CON.3) selects the active level of the gate input. In gated timer mode the same options for the input frequency as for the timer mode are available. However, the input clock to the timer in this mode is gated by the external input line T6IN (Timer T6 External Input).

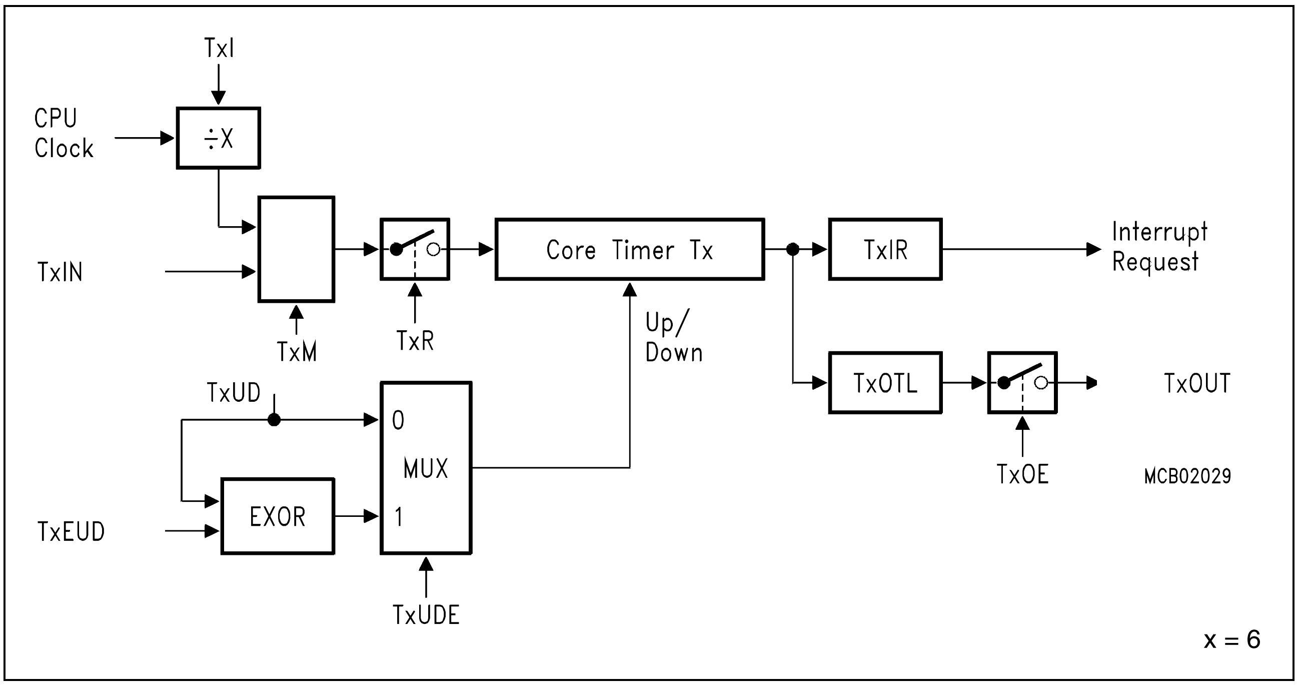


Figure 14-17 Block Diagram of Core Timer T6 in Gated Timer Mode

If T6M.0 = ‘0’ the timer is enabled when T6IN shows a low level. A high level at this line stops the timer. If T6M.0 = ‘1’ line T6IN must have a high level in order to enable the timer. In addition, the timer can be turned on or off by software using bit T6R. The timer will only run, if T6R = ‘1’ and the gate is active. It will stop, if either T6R = ‘0’ or the gate is inactive.

Note: A transition of the gate signal at line T6IN does not cause an interrupt request.

Timer 6 in Counter Mode

Counter mode for the core timer T6 is selected by setting bit field T6M in register T6CON to ‘001_B’. In counter mode timer T6 is clocked by a transition at the external input line T6IN. The event causing an increment or decrement of the timer can be a positive, a negative, or both a positive and a negative transition at this line. Bit field T6I in control register T6CON selects the triggering transition (see table below).

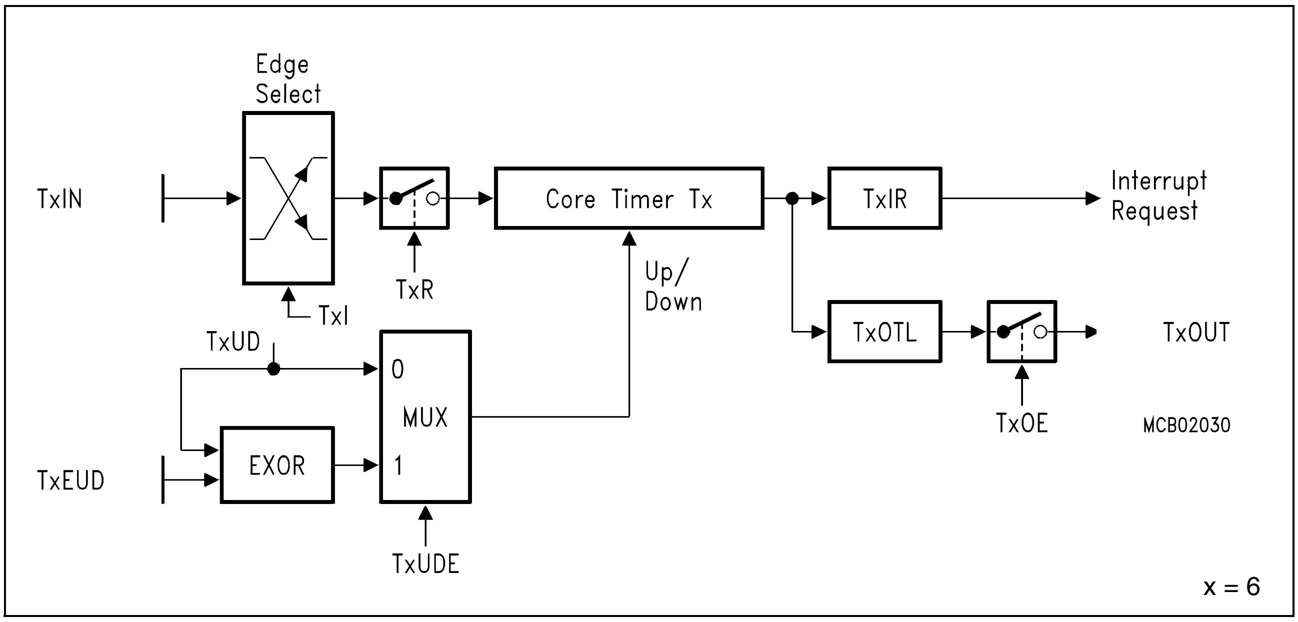


Figure 14-18 Block Diagram of Core Timer T6 in Counter Mode

Table 14-8 Core Timer T6 (Counter Mode) Input Edge Selection

T6I	Triggering Edge for Counter Increment / Decrement
0 0 0	None. Counter T6 is disabled
0 0 1	Positive transition (rising edge) on T6IN
0 1 0	Negative transition (falling edge) on T6IN
0 1 1	Any transition (rising or falling edge) on T6IN
1 X X	Reserved. Do not use this combination

The maximum input frequency which is allowed in counter mode is $f_{\mathrm{Timer}}/4$ (FM2 = ’1’). To ensure that a transition of the count input signal which is applied to T6IN is correctly recognized, its level should be held high or low for at least 2 f_{Timer} cycles (FM2 = ’1’) before it changes.

14.2.2 Auxiliary Timer T5

The auxiliary timer T5 can be configured for timer, gated timer, or counter mode with the same options for the timer frequencies and the count signal as the core timer T6. In addition to these 3 counting modes, the auxiliary timer can be concatenated with the core timer.

The individual configuration for timer T5 is determined by its bitaddressable control register T5CON. Note that functions which are present in both timers of timer block 2 are controlled in the same bit positions and in the same manner in each of the specific control registers.

Run control for auxiliary timer T5 can be handled by the associated Run Control Bit T5R in register T5CON. Alternatively, a remote control option (T5RC = ’1’) may be enabled to start and stop T5 via the run bit T6R of core timer T6.

Note: The auxiliary timer has no overflow/underflow toggle latch. Therefore, an output line for Overflow/Underflow Monitoring is not provided.

Count Direction Control for Auxiliary Timer

The count direction of the auxiliary timer can be controlled in the same way as for the core timer T6. The description and the table apply accordingly.

Timer T5 in Timer Mode or Gated Timer Mode

When the auxiliary timer T5 is programmed to timer mode or gated timer mode, its operation is the same as described for the core timer T6. The descriptions, figures and tables apply accordingly with two exceptions:

- There is no TxOUT line for T5.
- Overflow/Underflow Monitoring is not supported (no output toggle latch).

Timer T5 in Counter Mode

Counter mode for the auxiliary timer T5 is selected by setting bit field T5M in register T5CON to ‘001_B’. In counter mode timer T5, can be clocked by a transition of timer T6’s output toggle latch T6OTL only.

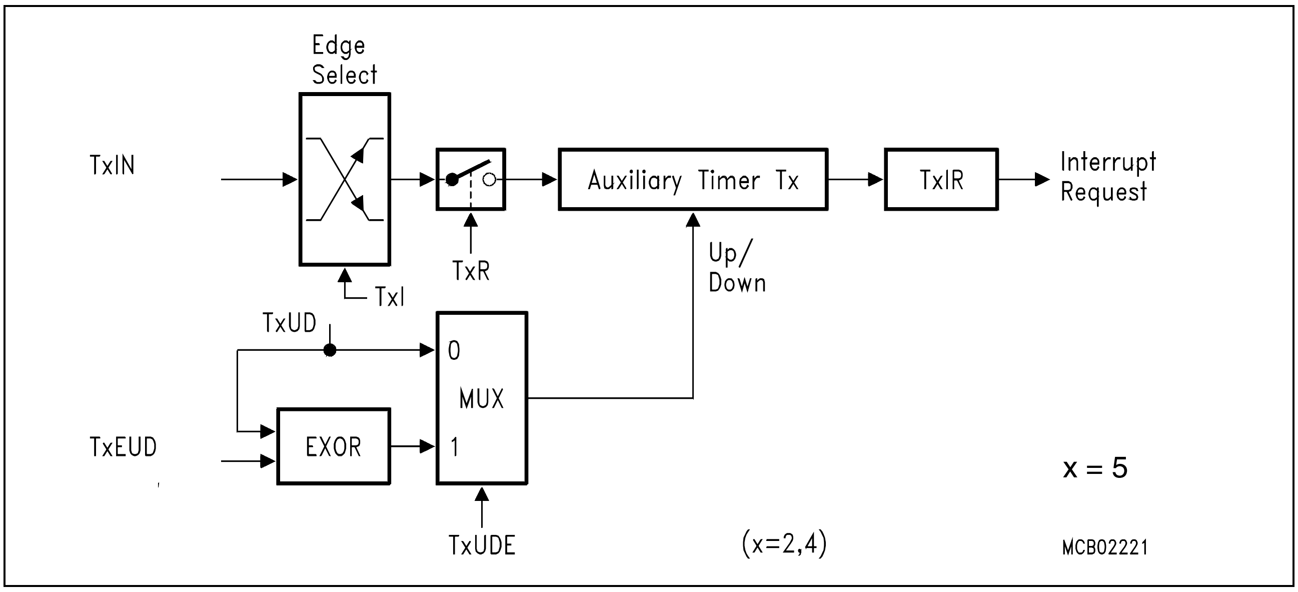


Figure 14-19 Block Diagram of Auxiliary Timer T5 in Counter Mode

The event causing an increment or decrement of the timer can be a positive, a negative, or both a positive and a negative transition at either the input line, or at the toggle latch T6OTL.

Bit field T5P in control register T5CON selects the triggering transition (see table below).

Table 14-9 Auxiliary Timer (Counter Mode) Input Edge Selection

T5P	Triggering Edge for Counter Increment / Decrement
X 0 0	None. Counter T5 is disabled
0 0 1	Positive transition (rising edge) on T5IN
0 1 0	Negative transition (falling edge) on T5IN
0 1 1	Any transition (rising or falling edge) on T5IN
1 0 1	Positive transition (rising edge) of output toggle latch T6OTL
1 1 0	Negative transition (falling edge) of output toggle latch T6OTL
1 1 1	Any transition (rising or falling edge) of output toggle latch T6OTL

Note: Only state transitions of T6OTL which are caused by the overflows/underflows of T6 will trigger the counter function of T5. Modifications of T6OTL via software will **NOT** trigger the counter function of T5.

The maximum input frequency which is allowed in counter mode is $f_{\mathrm{Timer}}/4$ (FM2 = ’1’). To ensure that a transition of the count input signal which is applied to T5IN is correctly recognized, its level should be held high or low for at least 2 f_{Timer} cycles (FM2 = ’1’) before it changes.

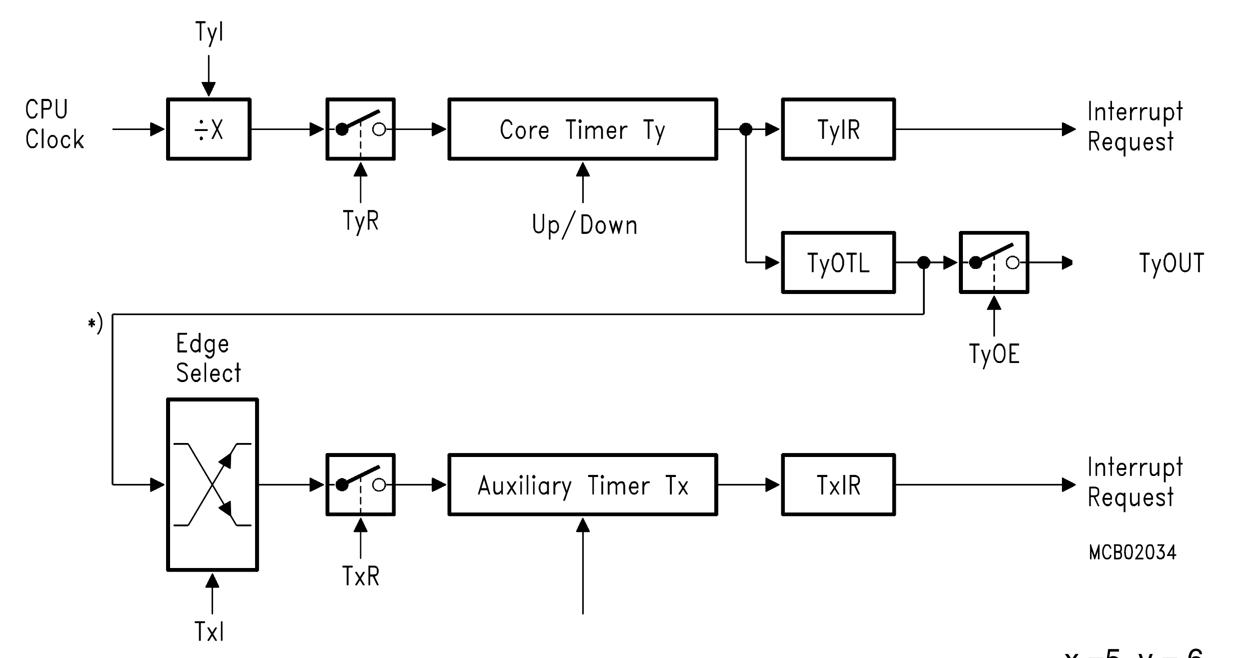
14.2.3 Timer Concatenation

Using the toggle bit T6OTL as a clock source for the auxiliary timer in counter mode concatenates the core timer T6 with the auxiliary timer. Depending on which transition of T6OTL is selected to clock the auxiliary timer, this concatenation forms a 32-bit or a 33-bit timer / counter.

- 32-bit Timer/Counter: If both a positive and a negative transition of T6OTL is used to clock the auxiliary timer, this timer is clocked on every overflow/underflow of the core timer T6. Thus, the two timers form a 32-bit timer.
- 33-bit Timer/Counter: If either a positive or a negative transition of T6OTL is selected to clock the auxiliary timer, this timer is clocked on every second overflow/underflow of the core timer T6. This configuration forms a 33-bit timer (16-bit core timer+T6OTL+16-bit auxiliary timer).

The count directions of the two concatenated timers are not required to be the same. This offers a wide variety of different configurations.

T6 can operate in timer, gated timer or counter mode in this case.



Note: Line '*' only affected by over/underflows of T6, but NOT by software modifications of T6OTL.

Figure 14-20 Concatenation of Core Timer T6 and Auxiliary Timer T5

Capture/Reload Register CAPREL in Capture Mode

This 16-bit register can be used as a capture register for the auxiliary timer T5. This mode is selected by setting bit T5SC = '1' in control register T5CON. Bit CT3 selects the external input line CAPIN or the input lines of timer T3 as the source for a capture trigger. Either a positive, a negative, or both a positive and a negative transition at line CAPIN can be selected to trigger the capture function, or transitions on input T3IN or input T3EUD or both inputs T3IN and T3EUD. The active edge is controlled by bit field CI in register T5CON.

The maximum input frequency for the capture trigger signal at CAPIN is $f_{Timer}/2$ (FM2 = '1'). To ensure that a transition of the capture trigger signal is correctly recognized, its level should be held for at least 2 f_{Timer} cycles (FM2 = '1') before it changes.

When the timer T3 capture trigger is enabled (CT3 = '1') register CAPREL captures the contents of T5 upon transitions of the selected input(s). These values can be used to measure T3's input signals. This is useful e.g. when T3 operates in incremental interface mode, in order to derive dynamic information (speed acceleration) from the input signals.

When a selected transition at the external input line CAPIN is detected, the contents of the auxiliary timer T5 are latched into register CAPREL, and interrupt request flag CRIR is set. With the same event, timer T5 can be cleared to 0000_H.

This option is controlled by bit T5CLR in register T5CON. If T5CLR = ‘0’, the contents of timer T5 is not affected by a capture. If T5CLR = ‘1’, timer T5 is cleared after the current timer value has been latched into register CAPREL.

Note: Bit T5SC only controls whether a capture is performed or not. If T5SC = ‘0’, the input line CAPIN can still be used to clear timer T5 or as an external interrupt input. This interrupt is controlled by the CAPREL interrupt control register CRIC.

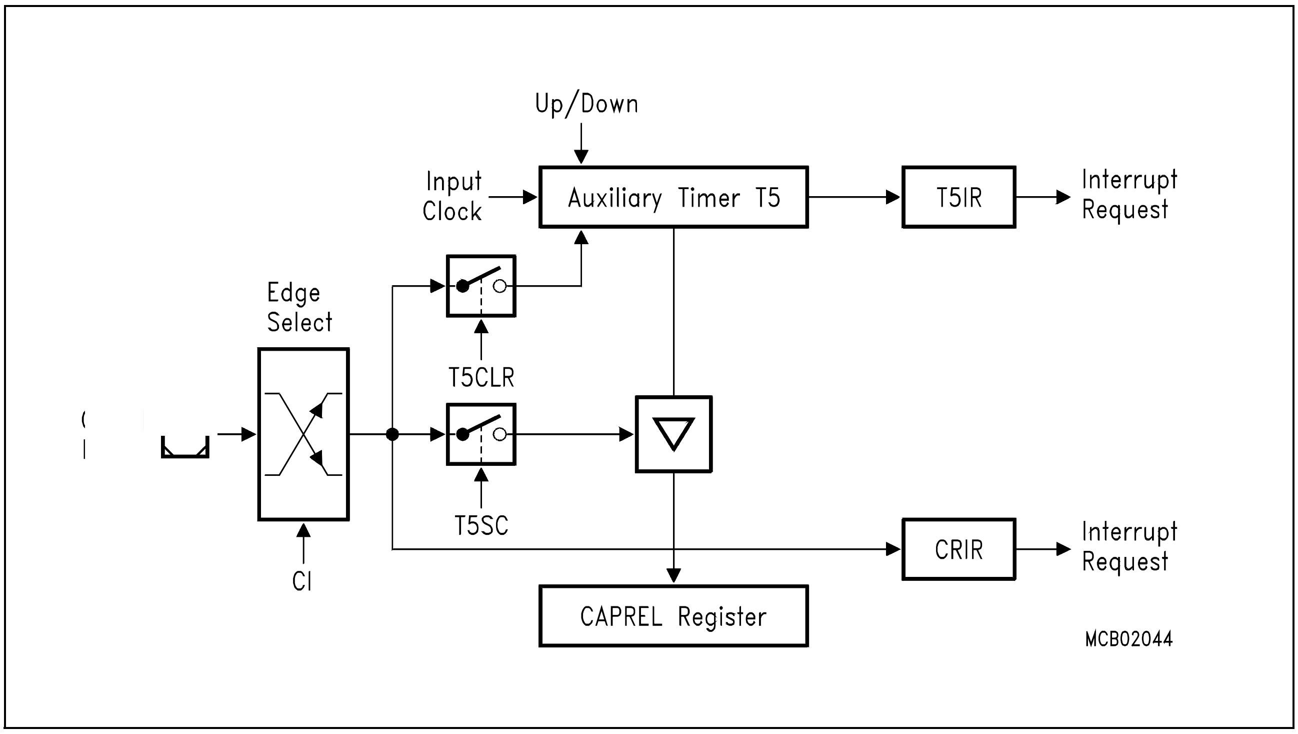


Figure 14-21 Timer Block 2 Register CAPREL in Capture Mode

Timer Block 2 Capture/Reload Register CAPREL in Reload Mode

This 16-bit register can be used as a reload register for the core timer T6. This mode is selected by setting bit T6SR = ‘1’ in register T6CON. The event causing a reload in this mode is an overflow or underflow of the core timer T6.

When timer T6 overflows from $FFFF_H$ to 0000_H (when counting up) or when it underflows from 0000_H to $FFFF_H$ (when counting down), the value stored in register CAPREL is loaded into timer T6. This will not set the interrupt request flag CRIR associated with the CAPREL register. However, interrupt request flag T6IR will be set indicating the overflow/underflow of T6.

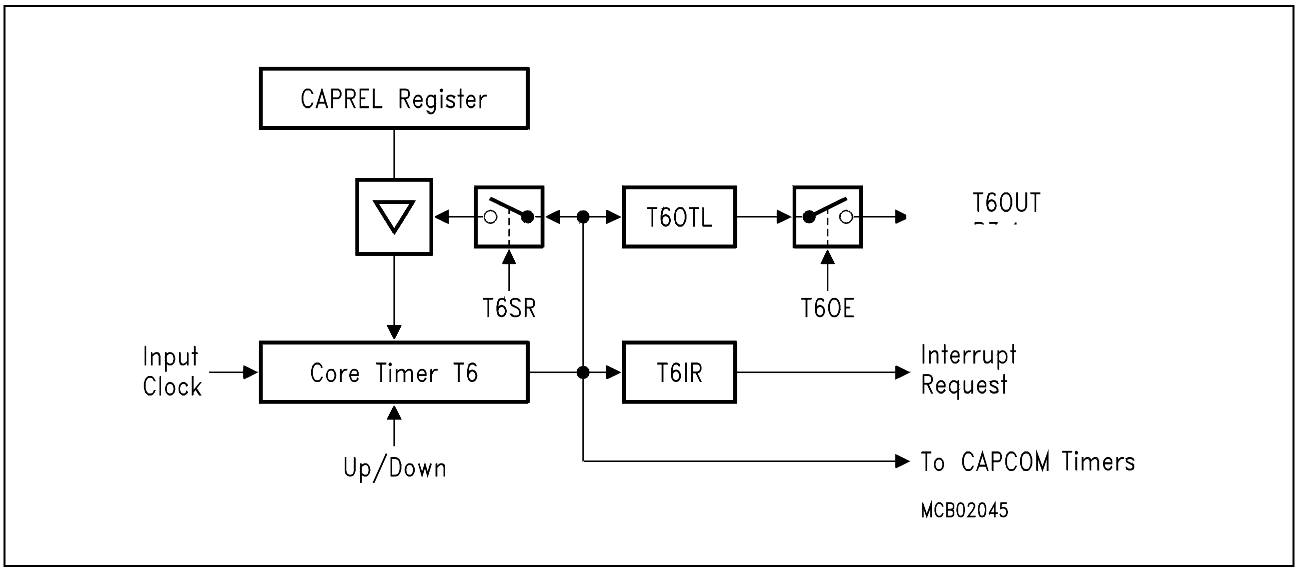


Figure 14-22 Timer Block 2 Register CAPREL in Reload Mode

Timer Block 2 Capture/Reload Register CAPREL in Capture-And-Reload Mode

Since the reload function and the capture function of register CAPREL can be enabled individually by bits T5SC and T6SR, the two functions can be enabled simultaneously by setting both bits. This feature can be used to generate an output frequency that is a multiple of the input frequency.

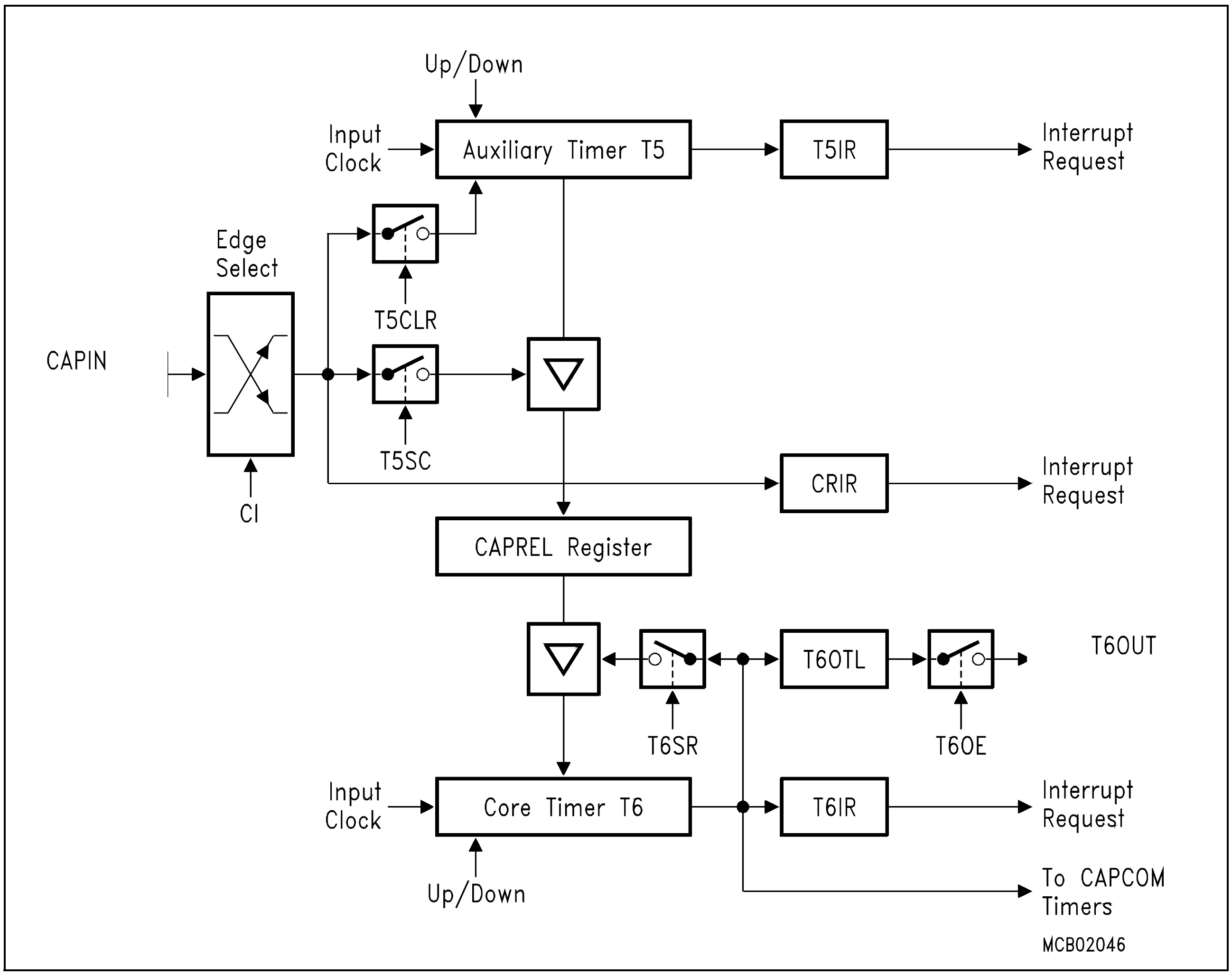


Figure 14-23 Timer Block 2 Register CAPREL in Capture-And-Reload Mode

This combined mode can be used to detect consecutive external events which may occur aperiodically, but where a finer resolution, that means, more 'ticks' within the time between two external events is required.

For this purpose, the time between the external events is measured using timer T5 and the CAPREL register. Timer T5 runs in timer mode counting up with a frequency of e.g. $f_{Timer}/32$. The external events are applied to line CAPIN. When an external event occurs, the timer T5 contents are latched into register CAPREL, and timer T5 is cleared (T5CLR = ‘1’). Thus, register CAPREL always contains the correct time between two events, measured in timer T5 increments. Timer T6, which runs in timer mode counting down with a frequency of e.g. $f_{Timer}/4$, uses the value in register CAPREL to perform a reload on underflow. This means, the value in register CAPREL represents the time between two underflows of timer T6, now measured in timer T6 increments. Since timer T6 runs 8 times faster than timer T5, it will underflow 8 times within the time between two external events. Thus, the underflow signal of timer T6 generates 8 'ticks'. Upon each underflow, the interrupt request flag T6IR will be set and bit T6OTL will be toggled. The state of T6OTL may be output on line T6OUT. This signal has 8 times more transitions than the signal which is applied to line CAPIN.

A certain deviation of the output frequency is generated by the fact that timer T5 will count actual time units (e.g. T5 running at 1 MHz will capture the value $64_H/100_D$ for a 10 KHz input signal) while T6OTL will only toggle upon an underflow of T6 (i.e. the transition from 0000_H to $FFFF_H$). In the above mentioned example T6 would count down from 64_H so the underflow would occur after 101 T6 timing ticks. The actual output frequency then is 79.2 KHz instead of the expected 80 KHz.
This can be solved by activating the Capture Correction (CC = '1'). If capture correction is active the content of T5 is decremented by 1 before being captured. The described deviation is eliminated (in the example T5 would capture $63_H/99_D$ and the output frequency is 80 KHz).

The underflow signal of timer T6 can furthermore be used to clock one ore more of the timers of the CAPCOM units, which gives the user the possibility to set compare events based on a finer resolution than that of the external events. This connection is accomplished via signal T6OFL.

14.2.4 Programming the GPT Unit

Several steps have to be done for getting a timer into operation. **Figure 14-24** presents a block diagram of the software tasks controlling the GPT unit.

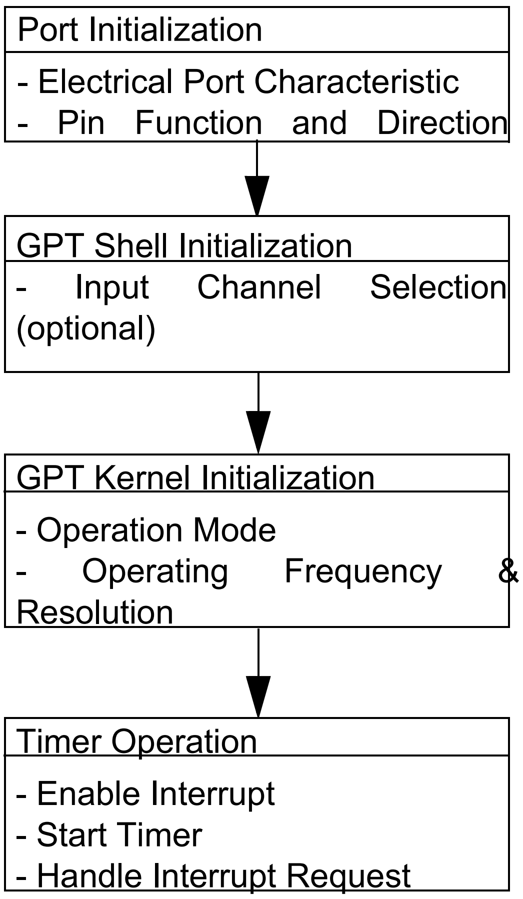

Port Initialization
- Electrical Port Characteristic
- Pin Function and Direction
GPT Shell Initialization
- Input Channel Selection (optional)
GPT Kernel Initialization
- Operation Mode
- Operating Frequency & Resolution
Timer Operation
- Enable Interrupt
- Start Timer
- Handle Interrupt Request

Figure 14-24 Block Diagram of Software Tasks controlling a Timer

14.3 GPT Registers

All available registers are summarized in the overview table 14-10.

Table 14-10 GPT12 Registers

Register Name	Register Description	Physical/8bit Address	b/p[1]	Reset Value
GPTCLC	GPT Clock Control Register	$FE4C/26_H$		0000_H
T2CON	Timer 2 Control Register	$FF40/A0_H$	b	0000_H
T3CON	Timer 3 Control Register	$FF42/A1_H$	b	0000_H
T4CON	Timer 4 Control Register	$FF44/A2_H$	b	0000_H
T5CON	Timer 5 Control Register	$FF46/A3_H$	b	0000_H
T6CON	Timer 6 Control Register	$FF48/A4_H$	b	0000_H
CAPREL	Capture/Reload Register	$FE4A/25_H$		0000_H
T2	Timer 2 Register	$FE40/20_H$		0000_H
T3	Timer 3 Register	$FE42/21_H$		0000_H
T4	Timer 4 Register	$FE44/22_H$		0000_H
T5	Timer 5 Register	$FE46/23_H$		0000_H
T6	Timer 6 Register	$FE48/24_H$		0000_H

[1] **b**: **b**it addressable / **p**: bit **p**rotected

The clock control register is described in chapter **23.5**.

Function Control Registers

The operating mode of the core timer T3 is configured and controlled via its bitaddressable control register T3CON.

Timer 3 Control Register

T3CON ($FF42_H$ / $A1_H$) **SFR-b** **Reset Value: 0000_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T3 IREN	T3 RDIR	T3CH DIR	T3 EDG E	FM1	T3 OTL	T3OE	T3 UDE	T3UD	T3R	T3M			T3I		

Field	Bits	Type	Value	Description
T3I	[2:0]	rw		**Timer 3 Input Parameter Selection** Timer mode see **Table 14-11** for encoding Gated Timer see **Table 14-11** for encoding Counter mode see **Table 14-12** for encoding Incremental Interface mode see **Table 14-13** for encoding
T3M	[5:3]	rw	 0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	**Timer 3 Mode Control** Timer Mode Counter Mode Gated Timer with Gate active low Gated Timer with Gate active high ***Reserved***. Do not use this combination! ***Reserved***. Do not use this combination! Incremental Interface Mode (Rotation detection) Incremental Interface Mode (Edge detection)
T3R	6	rw	 0 1	**Timer 3 Run Bit** Timer / Counter 3 stops Timer / Counter 3 runs
T3UD	7	rw	 0 1	**Timer 3 Up / Down Control** (when T3UDE = ’0) Counting ’Up’ Counting ’Down’
T3UDE	8	rw	 0 1	**Timer 3 External Up/Down Enable** Counting direction is internally controlled by SW Counting direction is externally controlled by line T3EUD
T3OE	9	rw	 0 1	**Overflow/Underflow Output Enable** T3 overflow/underflow can not be externally monitored T3 overflow/underflow may be externally monitored via T3OUT
T3OTL	10	rw	 0 / 1	**Timer 3 Output Toggle Latch** Toggles on each overflow / underflow of T3. Can be set or reset by software.
FM1	11	rw	 0 1	**Fast Mode for Timer Block 1** The maximum input frequency for Timer 2/3/4 is f_{Timer} / 8. The maximum input frequency for Timer 2/3/4 is f_{Timer} / 4.

Field	Bits	Type	Value	Description
T3EDGE	12	rw		**Timer 3 Edge Detection** The bit is set on each successful edge detection. The bit has to be reset by SW.
			0	No count edge was detected
			1	A count edge was detected
T3CHDIR	13	rw		**Timer 3 Count Direction Change** The bit is set on a change of the countdirection of timer 3. The bit has to be reset by SW.
			0	No change in count direction was detected
			1	A change in count direction was detected
T3RDIR	14	r		**Timer 3 Rotation Direction**
			0	Timer 3 counts up.
			1	Timer 3 counts down.
T3IREN	15	rw		**Timer 3 Interrupt Enable**
			0	Interrupt generation for T3CHDIR and T3EDGE is disabled.
			1	Interrupt generation for T3CHDIR and T3EDGE is enabled.

Table 14-11 Timer 3 Input Parameter Selection for Timer mode and Gated mode

T3I	Prescaler for f_{Timer} (FM1 = 0)	Prescaler for f_{Timer} (FM1 = 1)
000	8	4
001	16	8
010	32	16
011	64	32
100	128	64
101	256	128
110	512	256
111	1014	512

Table 14-12 Timer 3 Input Parameter Selection for Counter mode

T3I	Triggering Edge for Counter Update
000	None. Counter T3 is disabled
001	Positive transition (raising edge) on T3IN
010	Negative transition (falling edge) on T3IN
011	Any transition (raising or falling edge) on T3IN
1XX	***Reserved***. Do not use this combination!

Table 14-13 Timer 3 Input Parameter Selection for Incremental Interface mode

T3I	Triggering Edge for Counter Update
000	None. Counter T3 stops
001	Any transition (raising or falling edge) on T3IN
010	Any transition (raising or falling edge) on T3EUD
011	Any transition (raising or falling edge) on T3IN or T3EUD
1XX	***Reserved***. Do not use this combination!

Timer 2/4 Control Register

T2CON (FF40$_H$ / A0$_H$) **SFR-b** **Reset Value: 0000$_H$**

T4CON (FF44$_H$ / A2$_H$) **SFR-b** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Tx IREN	**Tx RDIR**	**TxCH DIR**	**Tx EDG E**	**0**		**TxRC**	**Tx UDE**	**TxUD**	**TxR**	**TxM**			**TxI**		

Field	Bits	Type	Value	Description
TxI	[2:0]	rw		**Timer x Input Parameter Selection** Timer mode see **Table 14-14** for encoding Gated Timer see **Table 14-14** for encoding Counter mode see **Table 14-15** for encoding Incremental Interface mode see **Table 14-16** for encoding
TxM	[5:3]	rw	 0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	**Timer x Mode Control** (Basic Operating Mode) Timer Mode Counter Mode Gated Timer with Gate active low Gated Timer with Gate active high Reload Mode Capture Mode Incremental Interface Mode (Rotation detection) Incermental Interface Mode (Edge detection)
TxR	6	rw	 0 1	**Timer x Run Bit** Timer / Counter x stops Timer / Counter x runs

Field	Bits	Type	Value	Description
TxUD	7	rw	 0 1	**Timer x Up / Down Control** **(when TxUDE = ’0)** Counting ’Up’ Counting ’Down’
TxUDE	8	rw	 0 1	**Timer x External Up/Down Enable** Counting direction is internally controlled by SW Counting direction is externally controlled by line TxEUD
TxRC	9	rw	 0 1	**Timer x Remote Control** Timer / Counter x is controlled by its own run bit TxR Timer / Counter x is controlled by the run bit of core timer 3
0	[11:10]	r		**reserved** for future use; reading returns 0; writing to these bit positions has no effect.
TxEDGE	12	rw	 0 1	**Timer x Edge Detection** The bit is set on each successful edge detection. The bit has to be reset by SW. No count edge was detected A count edge was detected
TxCHDIR	13	rw	 0 1	**Timer x Count Direction Change** The bit is set on a change of the countdirection of timer x. The bit has to be reset by SW. No change in count direction was detected A change in count direction was detected
TxRDIR	14	r	 0 1	**Timer x Rotation Direction** Timer x counts up. Timer x counts down.
TxIREN	15	rw	 0 1	**Timer x Interrupt Enable** Interrupt generation for TxCHDIR and TxEDGE is disabled. Interrupt generation for TxCHDIR and TxEDGE is enabled.

Table 14-14 Timer x Input Parameter Selection for Timer mode and Gated mode

TxI	Prescaler for f_{Timer} (FM1 = 0)	Prescaler for f_{Timer} (FM1 = 1)
000	8	4
001	16	8
010	32	16
011	64	32
100	128	64
101	256	128
110	512	256
111	1014	512

Table 14-15 Timer x Input Parameter Selection for Counter mode

TxI	Triggering Edge for Counter Update
X00	None. Counter Tx is disabled
001	reserved
010	reserved
011	reserved
101	Positive transition (rising edge) of output toggle latch T3OTL
110	Negative transition (falling edge) of output toggle latch T3OTL
111	Any transition (rising or falling edge) of output toggle latch T3OTL

Table 14-16 Timer x Input Parameter Selection for Incremental Interface mode

TxI	Triggering Edge for Counter Update
00	None. Counter Tx stops
001	reserved
010	reserved
011	reserved
101	Positive transition (rising edge) of output toggle latch T3OTL
110	Negative transition (falling edge) of output toggle latch T3OTL
111	Any transition (rising or falling edge) of output toggle latch T3OTL

Timer 6 Control Register

T6CON (FF48$_H$ / A4$_H$) **SFR-b** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T6SR	**T6 CLR**	**0**		**FM2**	**T6 OTL**	**T6OE**	**T6 UDE**	**T6UD**	**T6R**	**T6M**			**T6I**		

Field	Bits	Type	Value	Description
T6I	[2:0]	rw		**Timer 6 Input Parameter Selection** Timer mode see **Table 14-17** for encoding Gated Timer see **Table 14-17** for encoding Counter mode see **Table 14-18** for encoding
T6M	[5:3]	rw	 0 0 0 0 0 1 0 1 0 0 1 1 1 x x	**Timer 6 Mode Control** (Basic Operating Mode) Timer Mode Counter Mode Gated Timer with Gate active low Gated Timer with Gate active high ***Reserved***. Do not use this combination!
T6R	6	rw	 0 1	**Timer 6 Run Bit** Timer / Counter 6 stops Timer / Counter 6 runs
T6UD	7	rw	 0 1	**Timer 6 Up / Down Control (when T6UDE = ’0)** Counting ’Up’ Counting ’Down’
T6UDE	8	rw	 0 1	**Timer 6 External Up/Down Enable** Counting direction is internally controlled by SW Counting direction is externally controlled by line T6EUD
T6OE	9	rw	 0 1	**Overflow/Underflow Output Enable** T6 overflow/underflow can not be externally monitored T6 overflow/underflow may be externally monitored via T6OUT
T6OTL	10	rw	 0 / 1	**Timer 6 Output Toggle Latch** Toggles on each overflow / underflow of T6. Can be set or reset by software.

Field	Bits	Type	Value	Description
FM2	11	rw	 0 1	**Fast Mode for Timer Block 2** The maximum input frequency for Timer 5/6 is f_{Timer} / 4. The maximum input frequency for Timer 5/6 is f_{Timer} / 2.
0	[13:12]	r		**reserved** for future use; reading returns 0; writing to these bit positions has no effect.
T6CLR	14	rw	 0 1	**Timer 6 Clear Bit** Timer 6 is not cleared on a capture event Timer 6 is cleared on a capture event
T6SR	15	rw	 0 1	**Timer 6 Reload Mode Enable** Reload from register CAPREL Disabled Reload from register CAPREL Enabled

Table 14-17 Timer 6 Input Parameter Selection for Timer mode and Gated mode

T6I	Prescaler for f_{Timer} (FM2 = 0)	Prescaler for f_{Timer} (FM2 = 1)
000	4	2
001	8	4
010	16	8
011	32	16
100	64	32
101	128	64
110	256	128
111	512	256

Table 14-18 Timer 6 Input Parameter Selection for Counter mode

T6I	Triggering Edge for Counter Update
000	None. Counter T6 is disabled
001	Positive transition (raising edge) on T6IN
010	Negative transition (falling edge) on T6IN
011	Any transition (raising or falling edge) on T6IN
1XX	***Reserved***. Do not use this combination!

Timer 5 Control Register

T5CON (FF46$_H$ / A3$_H$) **SFR-b** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
T5SC	**T5 CLR**	**CI**		**CC**	**CT3**	**T5RC**	**T5 UDE**	**T5UD**	**T5R**	**0**	**T5M**		**T5I**		

Field	Bits	Type	Value	Description
T5I	[2:0]	rw		**Timer 5 Input Parameter Selection** Timer mode see **Table 14-19** for encoding Gated Timer see **Table 14-19** for encoding Counter mode see **Table 14-20** for encoding
T5M	[4:3]	rw	 0 0 0 1 1 0 1 1	**Timer 5 Mode Control** (Basic Operating Mode) Timer Mode Counter Mode Gated Timer with Gate active low Gated Timer with Gate active high
0	5	r		**reserved** for future use; reading returns 0; writing to these bit positions has no effect.
T5R	6	rw	 0 1	**Timer 5 Run Bit** Timer / Counter 5 stops Timer / Counter 5 runs
T5UD	7	rw	 0 1	**Timer 5 Up / Down Control** **(when T5UDE = ’0)** Counting ’Up’ Counting ’Down’
T5UDE	8	rw	 0 1	**Timer 5 External Up/Down Enable** Counting direction is internally controlled by SW Counting direction is externally controlled by line T5EUD
T5RC	9	rw	 0 1	**Timer 5 Remote Control** Timer / Counter 5 is controlled by its own run bit T5R Timer / Counter 5 is controlled by the run bit of core timer 6 (T6R)
CT3	10	rw	 0 1	**Timer 3 Capture Trigger Enable** Capture trigger from line CAPIN Capture trigger from T3 input lines
CC	11	rw	 0 1	**Capture Correction** T5 is just captured T5 is decremented by 1 before being captured

Field	Bits	Type	Value	Description
CI	[13:12]	rw	 0 0 0 1 1 0 1 1	**Register CAPREL Capture Trigger Selection** (depending on bit CT3) Capture disabled Positive transition (rising edge) on CAPIN or any transition on T3IN Negative transition (falling edge) on CAPIN or any transition on T3EUD Any transition (rising or falling edge) on CAPIN or any transition on T3IN or T3EUD
T5CLR	14	rw	 0 1	**Timer 5 Clear Bit** Timer 5 not cleared on a capture Timer 5 is cleared on a capture
T5SC	15	rw	 0 1	**Timer 5 Capture Mode Enable** Capture into register CAPREL Disabled Capture into register CAPREL Enabled

Table 14-19 Timer 5 Input Parameter Selection for Timer mode and Gated mode

T5I	Prescaler for f_{Timer} (FM2 = 0)	Prescaler for f_{Timer} (FM2 = 1)
000	4	2
001	8	4
010	16	8
011	32	16
100	64	32
101	128	64
110	256	128
111	512	256

Table 14-20 Timer 5 Input Parameter Selection for Counter mode

T5I	Triggering Edge for Counter Update
X00	None. Counter T5 is disabled
001	reserved
010	reserved

Table 14-20 Timer 5 Input Parameter Selection for Counter mode (cont'd)

T5I	Triggering Edge for Counter Update
011	reserved
101	Positive transition (rising edge) of output toggle latch T6OTL
110	Negative transition (falling edge) of output toggle latch T6OTL
111	Any transition (rising or falling edge) of output toggle latch T6OTL

Timer and Reload registers

T2 (FF40$_H$)	**SFR-b**	**Reset Value: 0000$_H$**
T3 (FF42$_H$)	**SFR-b**	**Reset Value: 0000$_H$**
T4 (FF44$_H$)	**SFR-b**	**Reset Value: 0000$_H$**
T5 (FF46$_H$)	**SFR-b**	**Reset Value: 0000$_H$**
T6 (FF48$_H$)	**SFR-b**	**Reset Value: 0000$_H$**
CAPREL (FF4A$_H$)	**SFR-b**	**Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Timer/Reload Value															

15 The Asynchronous / Synchr. Serial Interface

15.1 Functional Description

The ASC supports full-duplex asynchronous communication up to 1.5 MBaud and half-duplex synchronous communication up to 3 MBaud (@ 24 MHz CPU clock) . In synchronous mode, data are transmitted or received synchronous to a shift clock which is generated by CPU. In asynchronous mode, 8- or 9-bit data transfer, parity generation, and the number of stop bits can be selected.

15.1.1 Features

Full duplex asynchronous operating modes

- 8- or 9-bit data frames, LSB first
- Parity bit generation/checking
- One or two stop bits
- Baudrate from 1.5 MBaud to 0.3552 Baud (@24 MHz CPU clock)
- Multiprocessor mode for automatic address/data byte detection
- Loop-back capability
- Support for IrDA data transmission/reception up to max 115.2 KBaud

Half-duplex 8-bit synchronous operating mode

- Baudrate from 3 MBaud to 305.76 Baud (@ 24 MHz CPU clock)
- Double buffered transmitter/receiver

Interrupt generation

- on a transmitter buffer empty condition
- on a transmit last bit of a frame condition
- on a receiver buffer full condition
- on an error condition (frame, parity, overrun error)

15.1.2 Overview

Figure 15-1 shows a block diagram of the ASC with its operating modes (asynchronous and synchronous mode.).

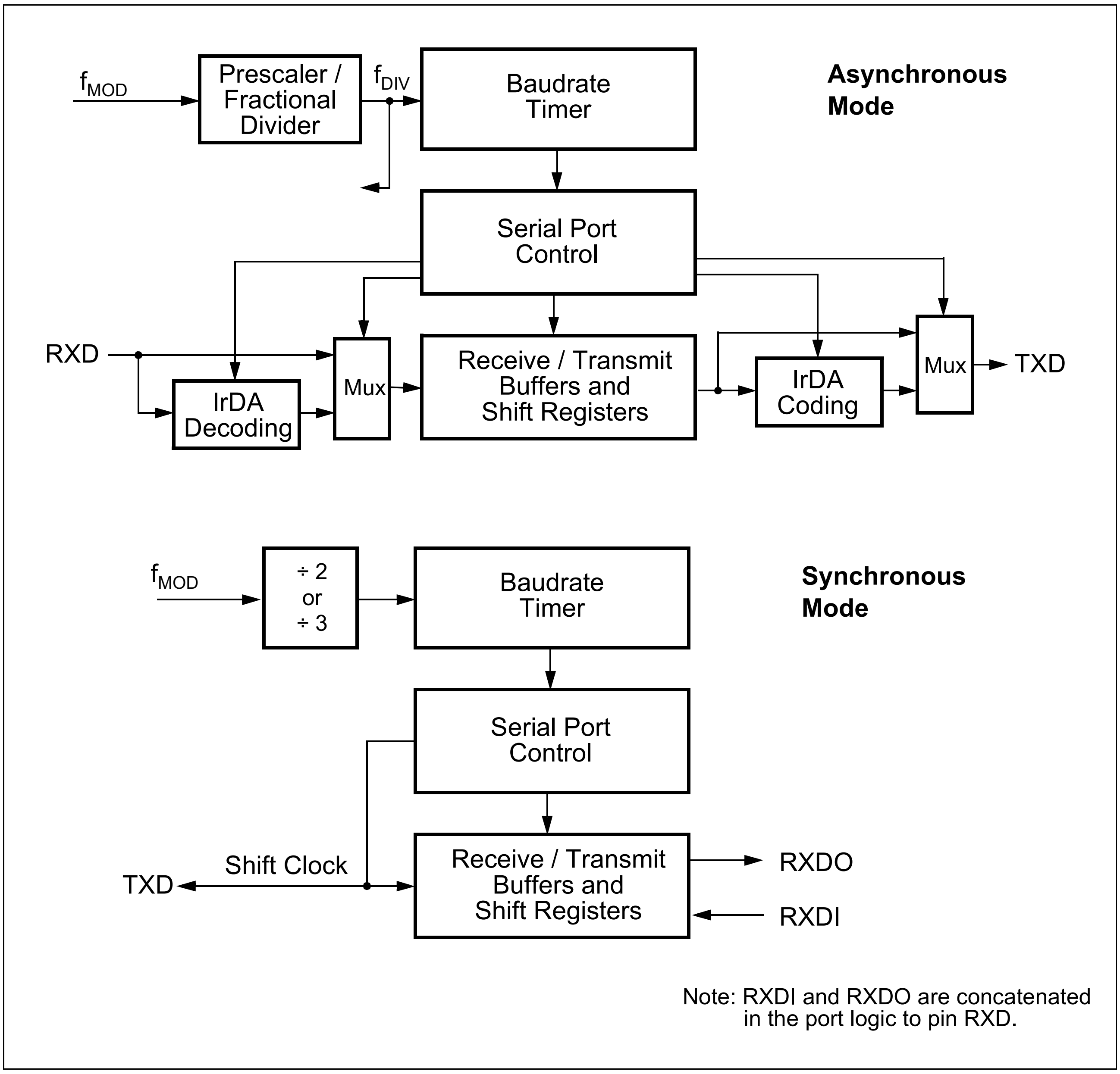


Figure 15-1 Block Diagram of the ASC

15.1.3 Register Description

The **ASC_P** registers can be basically divided into three types of registers as shown in **Figure 15-2**.

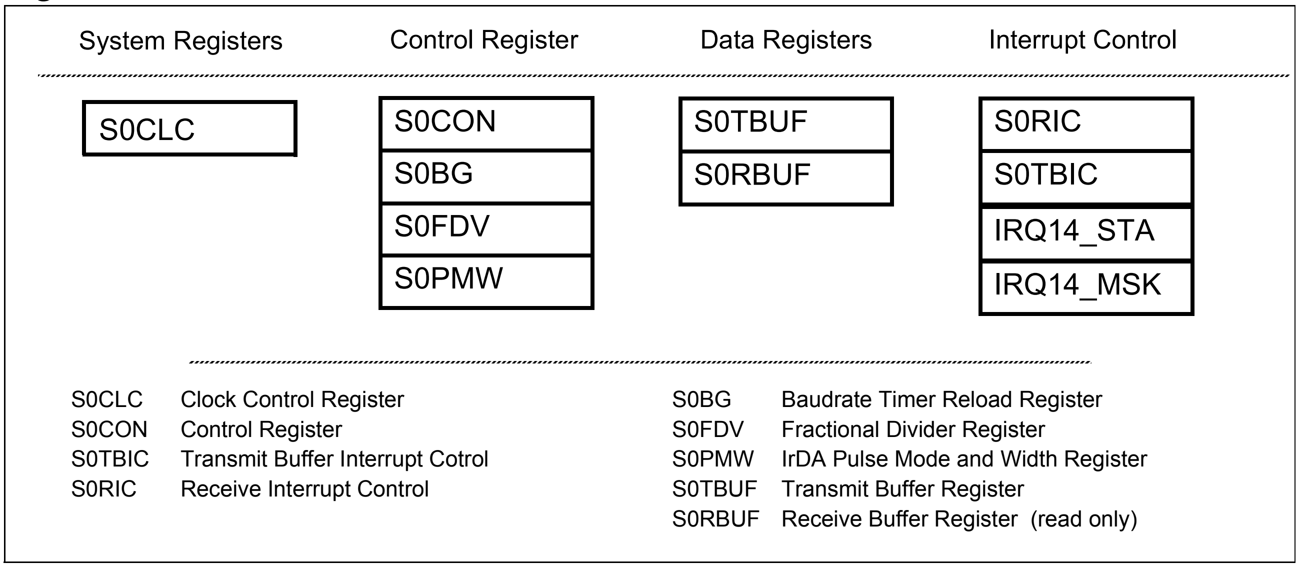

System Registers
Control Register
Data Registers
Interrupt Control
S0CLC
S0CON
S0BG
S0FDV
S0PMW
S0TBUF
S0RBUF
S0RIC
S0TBIC
IRQ14_STA
IRQ14_MSK
S0CLC Clock Control Register
S0CON Control Register
S0TBIC Transmit Buffer Interrupt Cotrol
S0RIC Receive Interrupt Control
S0BG Baudrate Timer Reload Register
S0FDV Fractional Divider Register
S0PMW IrDA Pulse Mode and Width Register
S0TBUF Transmit Buffer Register
S0RBUF Receive Buffer Register (read only)

Figure 15-2 SFRs associated with ASC_P1

Table 15-1 ASC_P Register Summary

Name	Address	Reset Value	Type [1]	Description
16-Bit Register Mapping				
S0CLC	$FFBA_H$	0000_H	rw	Clock Control Register [2]
S0CON	$FFB0_H$	0000_H	rwh	Control Register
S0BG	$FEB4_H$	0000_H	rw	Baudrate Timer Reload Register
S0FDV	$FEB6_H$	0000_H	rw	Fractional Divider Register
S0PMW	$FEAA_H$	0000_H	rw	IrDA Pulse Mode and Width Register
S0TBUF	$FEB0_H$	0000_H	rw	Transmit Buffer Register
S0RBUF	$FEB2_H$	0000_H	r	Receive Buffer Register
S0TBIC	$F19C_H$	0000_H	rw	Transmit Buffer Interrupt Control
S0RIC	$FF6E_H$	0000_H	rw	Receive Interrupt Control
IRQ14_STA	$DF24_H$	0000_H		Combined Interrupt 14 Status Reg.
IRQ14_MSK	$DF26_H$	0000_H		Combined Interrupt 14 Mask Reg.

[1] r: read only; w: write only; rw: read- and writeable; rwh: like rw, but SFR/bit is also affected by hardware.

[2] The **ASC_P** Clock Control Register CLC is physically located in the Bus Peripheral Interface. The table above defines only its register address not its content.

The serial operating modes of the ASC module are controlled by its control register S0CON. This register contains control bits for mode and error check selection, and status flags for error identification.

S0CON
Control Register

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LB	BRS	ODD	FDE	OE	FE	PE	OEN	FEN	PEN/ RXDI	REN	ST	M		

Field	Bits	Type	Value	Description
M	2-0	rw		**Mode Selection**
			0 0 0	8-bit data synchronous operation
			0 0 1	8-bit data async. operation
			0 1 0	IrDA mode, 8-bit data async. operation
			0 1 1	7-bit data + parity async. operation
			1 0 0	9-bit data async. operation
			1 0 1	8-bit data + wake up bit async. operation
			1 1 0	Reserved. Do not use this combination!
			1 1 1	8-bit data + parity async. operation
STP	3	rw		**Number of Stop Bit Selection**
			0	One stop bit
			1	Two stop bits
REN	4	rwh		**Receiver Enable Control**
			0	Receiver disabled
			1	Receiver enabled Bit is reset by hardware after reception of byte in synchronous mode.
PEN / RXDI	5	rw		**Parity Check Enable / IrDA Input Inverter Enable** All asynchronous modes without IrDA mode :
			0	Ignore parity errors
			1	Check parity errors Only in IrDA mode (M=010) :
			0	RXD input is not inverted
			1	RXD input is inverted
FEN	6	rw		**Framing Check Enable** (async. operation only)
			0	Ignore framing errors
			1	Check framing errors
OEN	7	rw		**Overrun Check Enable**
			0	Ignore overrun errors
			1	Check overrun errors
PE	8	rwh		**Parity Error Flag** Set by hardware on a parity error (PEN='1'). Must be reset by software.
FE	9	rwh		**Framing Error Flag** Set by hardware on a framing error (FEN='1'). Must be reset by software.

Field	Bits	Type	Value	Description
OE	10	rwh		**Overrun Error Flag** Set by hardware on an overrun error (OEN='1'). Must be reset by software.
FDE	11	rw	 0 1	**Fractional Divider Enable** Fractional divider disabled Fractional divider is enabled and used as prescaler for baudrate timer (bit BRS is don't care)
ODD	12	rw	 0 1	**Parity Selection** Even parity selected (parity bit set on odd number of '1's in data) Odd parity selected (parity bit set on even number of '1's in data)
BRS	13	rw	 0 1	**Baudrate Selection** Baudrate timer prescaler divide-by-2 selected Baudrate timer prescaler divide-by-3 selected BRS is don't care if FDE=1 (fractional divider enabled)
LB	14	rw	 0 1	**Loopback Mode Enable** Loopback mode disabled Loopback mode enabled
R	15	rw	 0 1	**Baudrate Generator Run Control** Baudrate generator disabled (**ASC_P** inactive) Baudrate generator enabled BG should only be written if R='0'.
–	31-16	0	all	reserved

Note: Serial data transmission or reception is only possible when the run bit CON_R is set to '1'. Otherwise the serial interface is idle.
Do not program the mode control field COM_M to one of the reserved combinations to avoid unpredictable behaviour of the serial interface.

The baudrate timer reload register S0BG of the ASC module contains the 13-bit reload value for the baudrate timer in asynchronous and sychronous mode.

Baudrate Timer/Reload Register

S0BG (FEB4$_H$ / 5A$_H$) **SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**0**	**0**	**BR_VALUE**												

Field	Bits	Type	Value	Description
BR_VALUE	12-0	rw	all	**Baudrate Timer/Reload Register Value** Reading BG returns the 13-bit content of the baudrate timer (bits 15....13 return 0); writing BG loads the baudrate timer reload register (bits 15....13 are don't care). BG should only be written if CON_R='0'.
–	15-13	0	all	reserved

The fractional divider register S0FDV of the ASC module contains the 9-bit divider value for the fractional divider (asynchronous mode only). It is also used for reference clock generation of the autobaud detection unit.

Fractional Divider Register
S0FDV (FEB6$_H$ / 5B$_H$) **SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**0**	**0**	**0**	**0**	**0**	**0**	**FD_VALUE**								

Field	Bits	Type	Value	Description
FD_VALUE	8-0	rw	all	**Fractional Divider Register Value** FDV contains the 9-bit value n of the fractional divider which defines the fractional divider ratio: n/512 n=0-511). With n=0, the fractional divider is switched off (input=output frequency, $f_{DIV} = f_{MOD}$, see **Figure 15-11**).
–	15-9	0	all	reserved

The transmitter buffer register S0TBUF of the ASC module contains the transmit data value in asynchronous and synchronous modes.

Transmitter Buffer Register

S0TBUF (FEB0$_H$ / 58$_H$) **SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**0**	**0**	**0**	**0**	**0**	**0**	**TD_VALUE**								

Field	Bits	Type	Value	Description
TD_VALUE	8-0	rw	all	**Transmit Data Register Value** TBUF contains the data to be transmitted in asynchronous and synchronous operating mode of the ASC. Data transmission is double buffered, Therefore, a new value can be written to TBUF before the transmission of the previous value is complete.
–	15-9	0	all	reserved

The receiver buffer register S0RBUF of the ASC module contains the receive data value in asynchronous and synchronous modes.

S0RBUF
Transmitter Buffer Register

S0RBUF (FEB2$_H$ / 59$_H$) **SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**0**	**0**	**0**	**0**	**0**	**0**	**RD_VALUE**								

Field	Bits	Type	Value	Description
RD_VALUE	8-0	rw	all	**Receive Data Register Value** S0RBUF contains the reveived data bits and, depending on the selected mode, the parity bit in asynchronous and synchronous operating mode of the ASC. In asynchronous operating mode with M=011 (7-bit data + parity) the received parity bit is written into RD7. In asynchronous operating mode with M=111 (8-bit data + parity) the received parity bit is written into RD8.
–	15-9	0	all	reserved

The IrDA pulse mode and width register S0PMW of the ASC module contains the 8-bit IrDA pulse width value and the IrDA pulse width mode select bit. This register is only required in the IrDA operating mode..

IrDA Pulse Mode/Width Register
S0PMW (FEAA$_H$ / 55$_H$) **SFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	**0**	**0**	**0**	**0**	**0**	**0**	**IRPW**	**PW_VALUE**							

Field	Bits	Type	Value	Description
PW_VALUE	7-0	rw	all	**IrDA Pulse Width Value** PW_VALUE is the 8-bit value n, which defines the variable pulse width of an IrDA pulse. Depending on the **ASC_P** input frequency f_{MOD}, this value can be used to adjust the IrDA pulse width to value which is not equal 3/16 bit time (e.g. 1.6 µs).
IRPW	8	rw	 0 1	**IrDA Pulse Width Mode Control** IrDA pulse width is 3/16 of the bit time IrDA pulse width is defined by PW_VALUE
–	15-9	0	all	reserved

15.1.4 General Operation

Parity, framing, and overrun error detection is provided to increase the reliability of data transfers. Transmission and reception of data is double-buffered. For multiprocessor communication, a mechanism to distinguish address from data bytes is included. Testing is supported by a loop-back option. A 13-bit baudrate timer with a versatile input clock divider circuitry provides the ASC with the serial clock signal.
A transmission is started by writing to the Transmit Buffer register S0TBUF. Only the number of data bits which is determined by the selected operating mode will actually be transmitted, ie. bits written to positions 9 through 15 of register S0TBUF are always insignificant.
Data transmission is double-buffered, so a new character may be written to the transmit buffer register, before the transmission of the previous character is complete. This allows the transmission of characters back-to-back without gaps.
Data reception is enabled by the Receiver Enable Bit CON_REN. After reception of a character has been completed, the received data and, if provided by the selected operating mode, the received parity bit can be read from the (read-only) Receive Buffer register S0RBUF. Bits in the upper half of S0RBUF which are not valid in the selected operating mode will be read as zeros.
Data reception is double-buffered, so that reception of a second character may already begin before the previously received character has been read out of the receive buffer register. In all modes, receive buffer overrun error detection can be selected through bit CON_OEN. When enabled, the overrun error status flag CON_OE and the error interrupt request line EIR will be acitvated when the receive buffer register has not been read by the time reception of a second character is complete. The previously received character in the receive buffer is overwritten.
The Loop-Back option (selected by bit CON_LB) allows the data currently being transmitted to be received simultaneously in the receive buffer. This may be used to test serial communication routines at an early stage without having to provide an external network. In loop-back mode the alternate input/output function of port pins is not required.

Note: Serial data transmission or reception is only possible when the Baudrate Generator Run Bit CON_R is set to '1'. Otherwise the serial interface is idle.
Do not program the mode control field COM_M to one of the reserved combinations to avoid unpredictable behaviour of the serial interface

15.1.5 Asynchronous Operation

Asynchronous mode supports full-duplex communication, where both transmitter and receiver use the same data frame format and the same baudrate. Data is transmitted on pin P3.10/TXD and received on pin P3.11/RXD.
IrDA data transmission/reception is supported up to 115.2 KBit/s. **Figure 15-3** shows the block diagram of the **ASC_P**3 when operating in asynchronous mode.

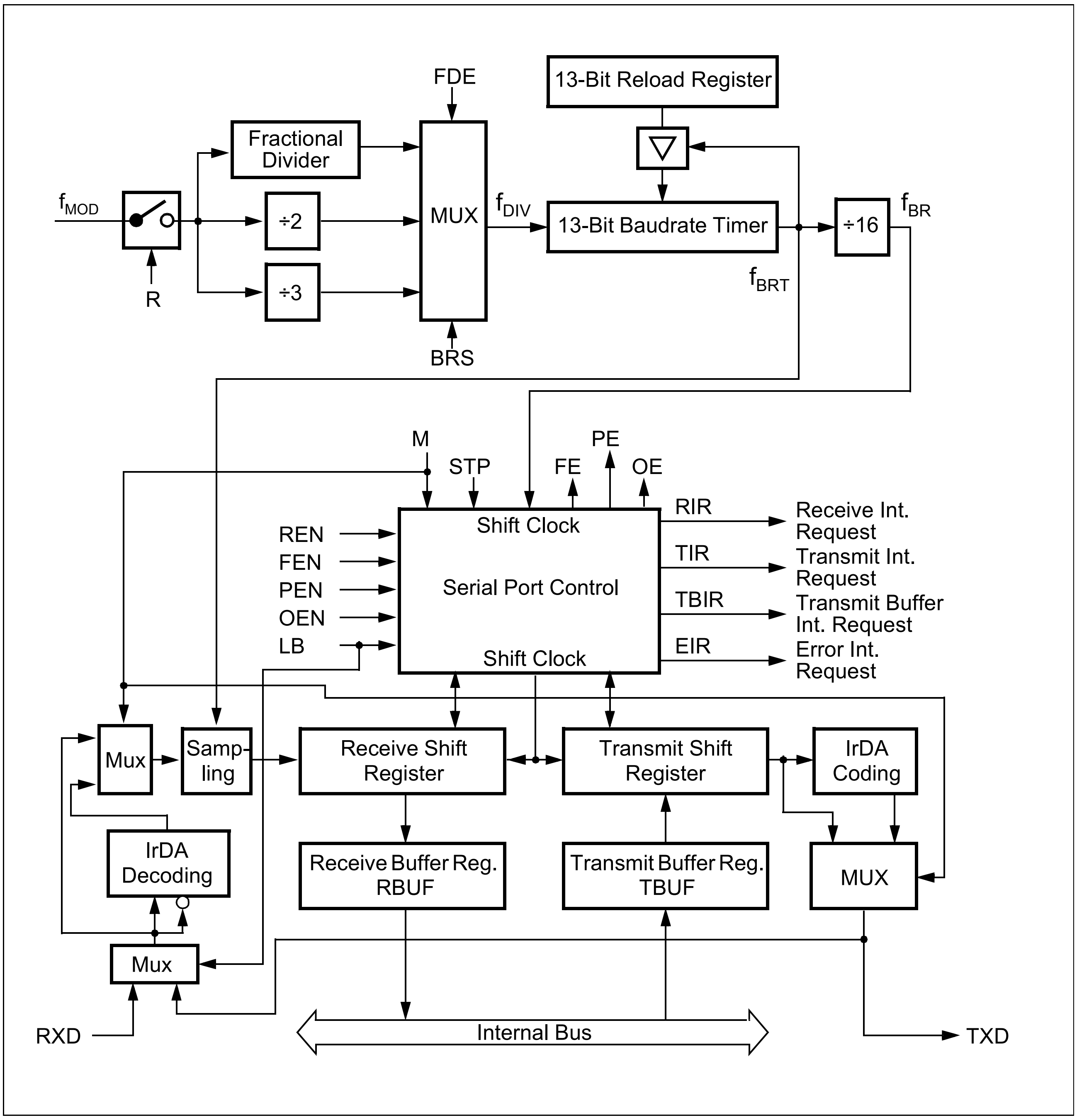


Figure 15-3 Asynchronous Mode of Serial Channel ASC_P3

15.1.5.1 Asynchronous Data Frames

8-Bit Data Frames

8-bit data frames either consist of 8 data bits D7...D0 (S0CON_M='001_B'), or of 7 data bits D6...D0 plus an automatically generated parity bit (S0CON_M='011_B'). Parity may be odd or even, depending on bit CON_ODD. An even parity bit will be set, if the modulo-

2-sum of the 7 data bits is ‘1’. An odd parity bit will be cleared in this case. Parity checking is enabled via bit CON_PEN (always OFF in 8-bit data mode). The parity error flag CON_PE will be set along with the error interrupt request flag, if a wrong parity bit is received. The parity bit itself will be stored in bit RBUF.7.

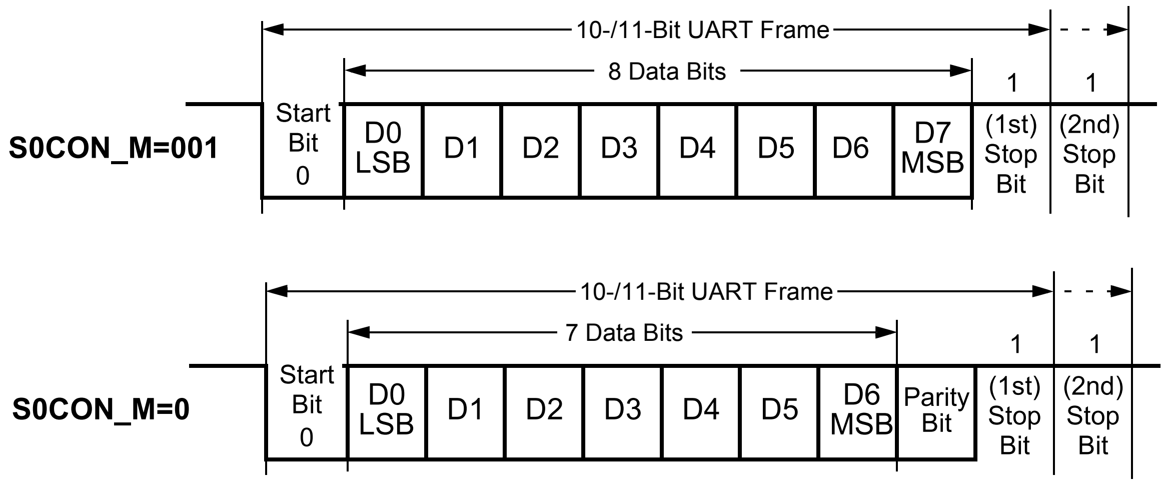


Figure 15-4 Asynchronous 8-Bit Frames

9-Bit Data Frames

9-bit data frames either consist of 9 data bits D8...D0 (S0CON_M=’100$_B$’), of 8 data bits D7...D0 plus an automatically generated parity bit (S0CON_M=’111$_B$’) or of 8 data bits D7...D0 plus wake-up bit (CON_M=’101$_B$’). Parity may be odd or even, depending on bit CON_ODD. An even parity bit will be set, if the modulo-2-sum of the 8 data bits is ‘1’. An odd parity bit will be cleared in this case. Parity checking is enabled via bit CON_PEN (always OFF in 9-bit data and wake-up mode). The parity error flag CON_PE will be set along with the error interrupt request flag, if a wrong parity bit is received. The parity bit itself will be stored in bit RBUF.8.

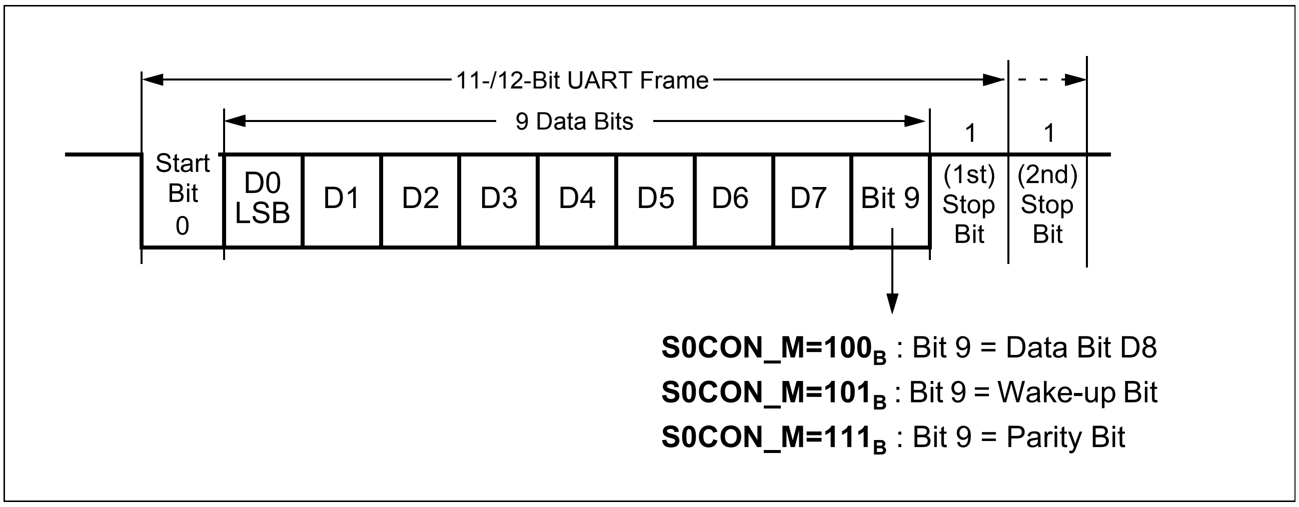

11-/12-Bit UART Frame
9 Data Bits
1
1
Start Bit 0
D0 LSB
D1
D2
D3
D4
D5
D6
D7
Bit 9
(1st) Stop Bit
(2nd) Stop Bit
S0CON_M=100B : Bit 9 = Data Bit D8
S0CON_M=101B : Bit 9 = Wake-up Bit
S0CON_M=111B : Bit 9 = Parity Bit

Figure 15-5 Asynchronous 9-Bit Frames

In wake-up mode received frames are only transferred to the receive buffer register, if the 9th bit (the wake-up bit) is ‘1’. If this bit is ‘0’, no receive interrupt request will be activated and no data will be transferred.
This feature may be used to control communication in multi-processor system:
When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the additional 9th bit is a '1' for an address byte and a '0' for a data byte, so no slave will be interrupted by a data 'byte'. An address 'byte' will interrupt all slaves (operating in 8-bit data + wake-up bit mode), so each slave can examine the 8 LSBs of the received character (the address). The addressed slave will switch to 9-bit data mode (eg. by clearing bit CON_M.0), which enables it to also receive the data bytes that will be coming (having the wake-up bit cleared). The slaves that were not being addressed remain in 8-bit data + wake-up bit mode, ignoring the following data bytes.

IrDA Frames

The modulation schemes of IrDA is based on standard asynchronous data transmission frames. The asynchronous data format in IrDA mode (S0CON_M=010_B) is defined as follows :
1 start bit / 8 data bits / 1 stop bit

The coding/decoding of/to the asynchronous data frames is shown in **Figure 15-6**. In general, during the IrDA transmissions UART frames are encoded into IR frames and vice cersa. A low level on the IR frame indicates a “LED off“ state. A high level on the IR frame indicates a “LED on“ state.
For a “0“ bit in the UART frame a high pulse is generated. For a “1“ bit in the UART frame no pulse is generated. The high pulse starts in the middle of a bit cell and has a fixed

width of 3/16 of the bit time. The **ASC_P** also allows to program the length of the IrDA high pulse. Further, the polarity of the received IrDA pulse cane be inverted in IrAD mode.

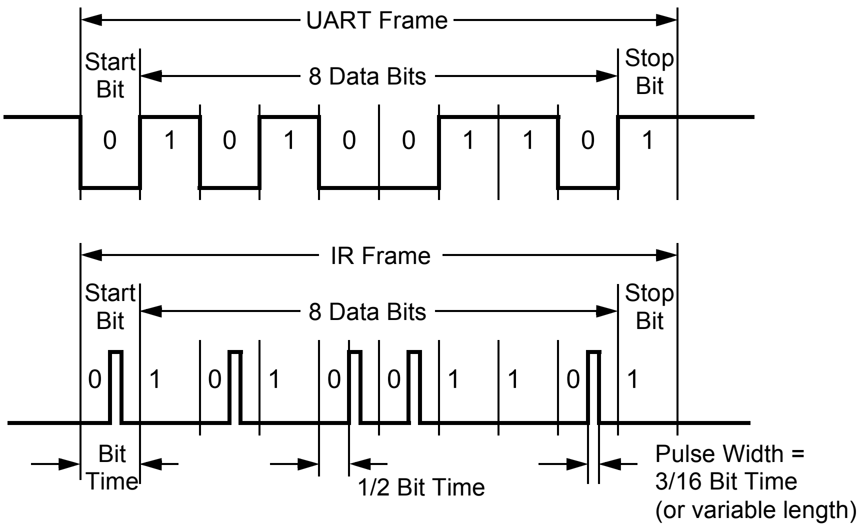


Figure 15-6 IrDA Frame Encoding/Decoding

15.1.5.2 Asynchronous Transmission

Asynchronous transmission begins at the next overflow of the divide-by-16 baudrate timer (transition of the baudrate clock f_{BR}), if bit S0CON_R must be set and data has been loaded into S0TBUF. The transmitted data frame consists of three basic elements:

- the start bit
- the data field (8 or 9 bits, LSB first, including a parity bit, if selected)
- the delimiter (1 or 2 stop bits)

Data transmission is double buffered. When the transmitter is idle, the transmit data loaded into S0TBUF is immediately moved to the transmit shift register thus freeing S0TBUF for the next data to be sent. This is indicated by the transmit buffer interrupt request line S0TBIR being activated. S0TBUF may now be loaded with the next data, while transmission of the previous one is still going on.

The transmit interrupt request line S0TIR will be activated before the last bit of a frame is transmitted, ie. before the first or the second stop bit is shifted out of the transmit shift register.

The transmitter output pin TXD must be configured for alternate data output'.

15.1.5.3 Asynchronous Reception

Asynchronous reception is initiated by a falling edge (1-to-0 transition) on pin RXD, provided that bits S0CON_R and S0CON_REN are set. The receive data input pin RXD is sampled at 16 times the rate of the selected baudrate. A majority decision of the 7th, 8th and 9th sample determines the effective bit value. This avoids erroneous results that may be caused by noise.
If the detected value is not a '0' when the start bit is sampled, the receive circuit is reset and waits for the next 1-to-0 transition at pin RXD. If the start bit proves valid, the receive circuit continues sampling and shifts the incoming data frame into the receive shift register.
When the last stop bit has been received, the content of the receive shift register is transferred to the receive data buffer register S0RBUF. Simultaneously, the receive interrupt request line RIR is activated after the 9th sample in the last stop bit time slot (as programmed), regardless whether valid stop bits have been received or not. The receive circuit then waits for the next start bit (1-to-0 transition) at the receive data input pin.
The receiver input pin RXD must be configured for input.
Asynchronous reception is stopped by clearing bit S0CON_REN. A currently received frame is completed including the generation of the receive interrupt request and an error interrupt request, if appropriate. Start bits that follow this frame will not be recognized.

Note: In wake-up mode received frames are only transferred to the receive buffer register, if the 9th bit (the wake-up bit) is '1'. If this bit is '0', no receive interrupt request will be activated and no data will be transferred.

15.1.5.4 IrDA Mode

The duration of the IrDA pulse is normally 3/16 of a bit period. The IrDA standard also allows the pulse duration being independent of the baudrate or bit period. In this case the transmitted pulse has always the width corresponding to the 3/16 pulse width at 115.2 kBaud which is 1.627 µs. Both, bit period dependend or fixed IrDA pulse width generation can be selected. The IrDA pulse width mode is selected by bit PMW_IRPW.
In case of fixed IrDA pulse width generation, the lower 8 bits in register PMW are used to adapt the IrDA pulse width to a fixed value of e.g. 1.627 µs. The fixed IrDA pulse width is generated by a programmable timer as shown in **Figure 15-7**.

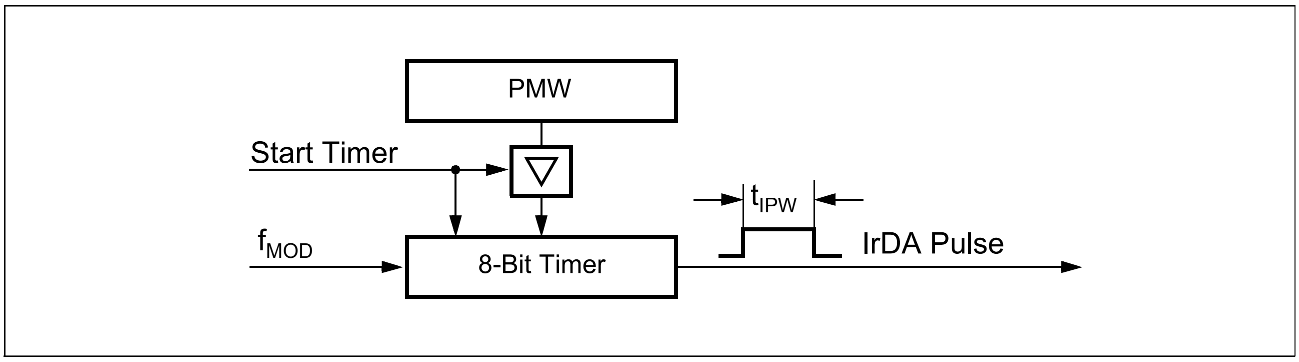


Figure 15-7 Fixed IrDA Pulse Generation

The IrDA pulse width can be calculated according the formulas given in **Table 15-2**.

Table 15-2 Formulas for the IrDA Pulse Width Calculation

PMW	PMW_IRPW	Formulas	
1 ... 255	0	$t_{IPW} = \frac{3}{16 \text{ x Baudrate}}$	$t_{IPW\ min} = \frac{(PMW >> 1)}{f_{MOD}}$
	1	$t_{IPW} = \frac{PMW}{f_{MOD}}$	

The name PMW in the formulas of **Table 15-2** represents the content of the reload register PMW (PW_VALUE), taken as unsigned 8-bit integer.
The content of PMW further defines the minimum IrDA pulse width ($t_{IPW\ min}$) which is still recognized during a receive operation as a valid IrDA pulse. This function is independent of the selected IrDA pulse width mode (fixed or variable) which is defined by bit PMW_IRPW. The minimum IrDA pulse width is calculated by a shift right operation of PMW bit 7-0 by one bit divided by the module clock f_{MOD}.

Note: If PMW_IRPW=0 (fixed IrDA pulse width), PW_VALUE must be a value which assures that $t_{IPW} > t_{IPW\ min}$.

Table 15-3 gives two examples for typical frequencies of f_{MOD}.

Table 15-3 IrDA Pulse Width Adaption to 1.627 µs

f_{MOD}	PMW	t_{IPW}	Error	$t_{IPW\ min}$
13 MHz	21	1.615 µs	- 0.12 %	0.77 µs
25 MHz	41	1.64 µs	+ 0.8 %	0.8 µs

15.1.5.5 RXD/TXD Data Path Selection in Asynchronous Modes

Figure 15-8 shows the asynchronous mode data paths.

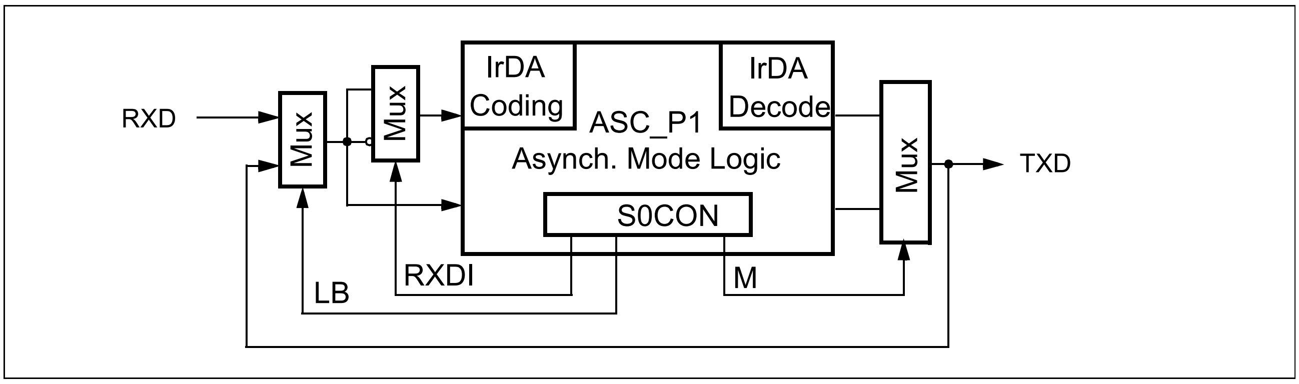


Figure 15-8 RXD/TXD Data Path in Asynchronous Modes (ASC_P1)

15.1.6 Synchronous Operation

Synchronous mode supports half-duplex communication, basically for simple I/O expansion via shift registers. Data is transmitted and received via pin RXD while pin TXD outputs the shift clock. These signals are alternate functions of port pins. Synchronous mode is selected with S0CON_M='000_B'.
8 data bits are transmitted or received synchronous to a shift clock generated by the internal baudrate generator. The shift clock is only active as long as data bits are transmitted or received.
The lines RXDI and RXDO must be concatenated in the port logic to pin RXD.

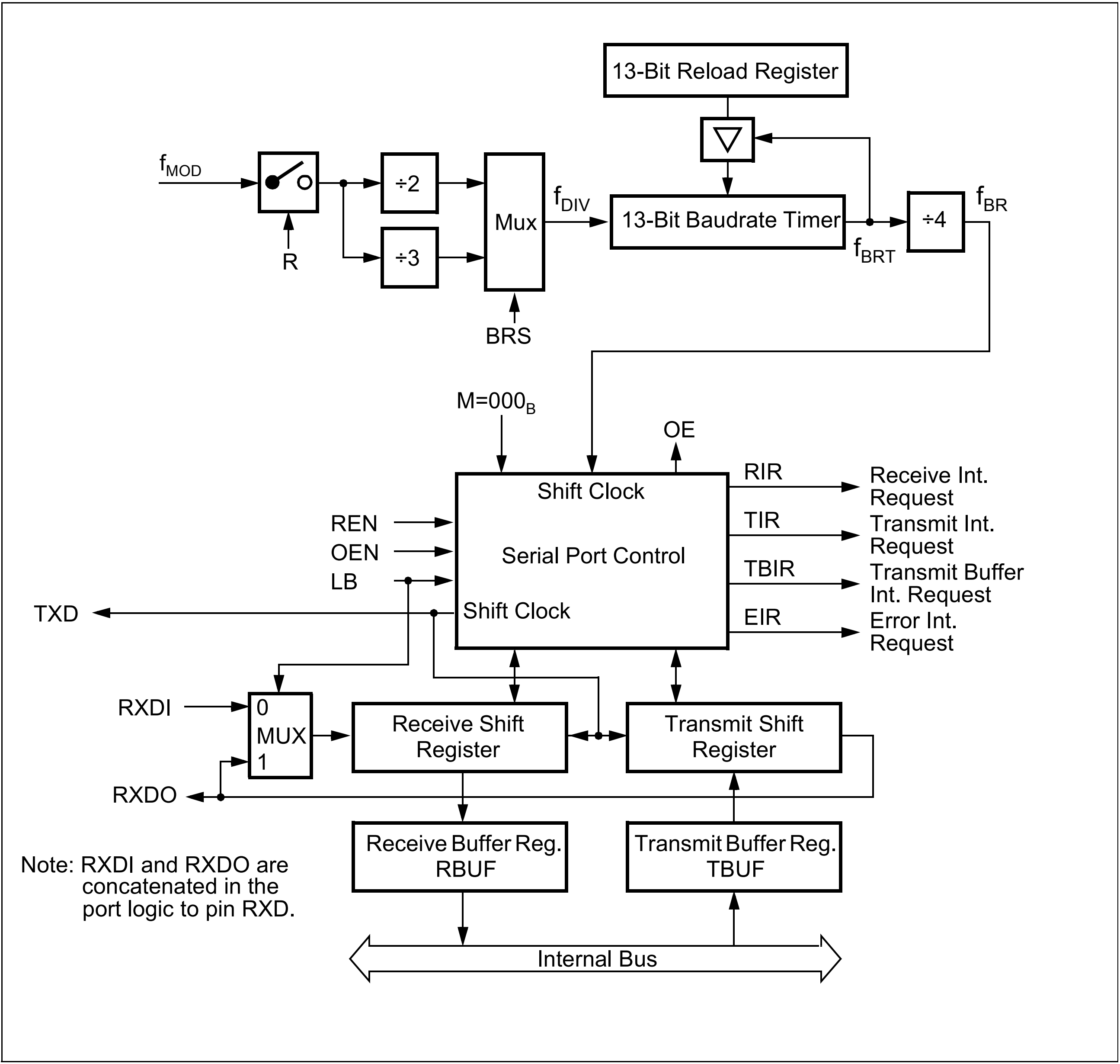


Figure 15-9 Synchronous Mode of Serial Channel ASC_P

15.1.6.1 Synchronous Transmission

Synchronous transmission begins within 4 state times after data has been loaded into S0TBUF provided that S0CON_R is set and S0CON_REN='0' (half-duplex, no reception). Exception : in loopback mode (bit S0CON_LB set), S0CON_REN must be set for reception of the transmitted byte. Data transmission is double buffered. When the transmitter is idle, the transmit data loaded into S0TBUF is immediately moved to the transmit shift register thus freeing S0TBUF for the next data to be sent. This is indicated by the transmit buffer interrupt request line TBIR being activated. S0TBUF may now be loaded with the next data, while transmission of the previous one is still going on. The data bits are transmitted synchronous with the shift clock. After the bit time for the 8th data bit, both TXD and RXD will go high, the transmit interrupt request line TIR is activated, and serial data transmission stops.

Pin P3.10/TXD must be configured for alternate data output in order to provide the shift clock. Pin P3.11/RXD must also be configured for output during transmission.

15.1.6.2 Synchronous Reception

Synchronous reception is initiated by setting bit S0CON_REN='1'. If bit S0CON_R=1, the data applied at RXD is clocked into the receive shift register synchronous to the clock which is output at pin TXD. After the 8th bit has been shifted in, the content of the receive shift register is transferred to the receive data buffer RBUF, the receive interrupt request line RIR is activated, the receiver enable bit S0CON_REN is reset, and serial data reception stops.

Pin P3.10/TXD must be configured for alternate data output in order to provide the shift clock. Pin P3.11/RXD must be configured as alternate data input.

Synchronous reception is stopped by clearing bit S0CON_REN. A currently received byte is completed including the generation of the receive interrupt request and an error interrupt request, if appropriate. Writing to the transmit buffer register while a reception is in progress has no effect on reception and will not start a transmission.

If a previously received byte has not been read out of the receive buffer register at the time the reception of the next byte is complete, both the error interrupt request line EIR and the overrun error status flag S0CON_OE will be activated/set, provided the overrun check has been enabled by bit S0CON_OEN.

15.1.6.3 Synchronous Timing

Figure 15-10 shows timing diagrams of the ASC synchronous mode data reception and data transmission. In idle state the shift clock is at high level. With the beginning of a synchronous transmission of a data byte the data is shifted out at RXD with the falling edge of the shift clock. If a data byte is received through RXD data is latched with the rising edge of the shift clock.

Between two consecutive receive or transmit data bytes one shift clock cycle (f_{BR}) delay is inserted.

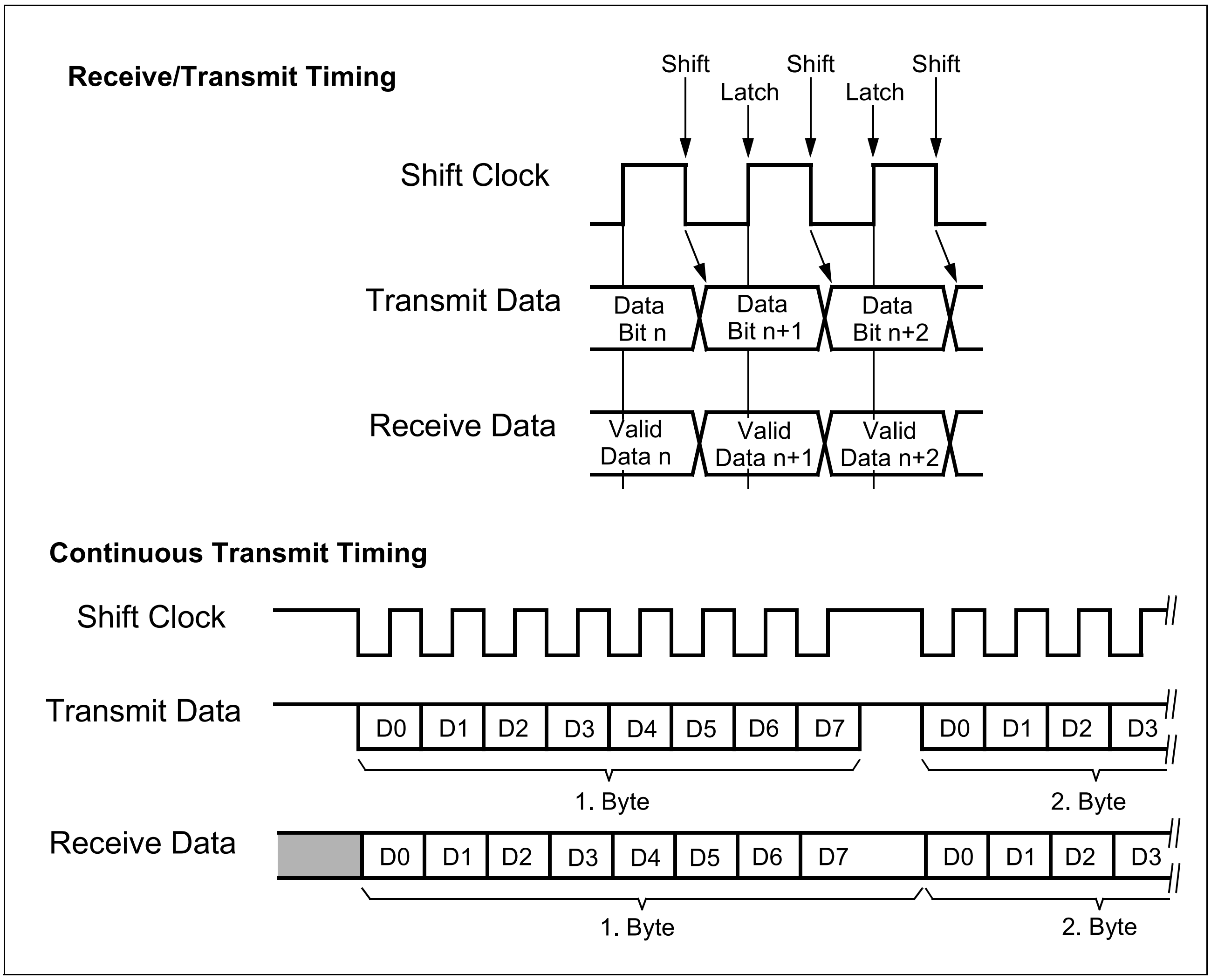


Figure 15-10 ASC_P3 Synchronous Mode Waveforms

15.1.7 Baudrate Generation

The serial channel ASC has its own dedicated 13-bit baudrate generator with 13-bit reload capability, allowing baudrate generation independent of the GPT timers.

The baudrate generator is clocked with a clock (f_{DIV}) which is derived via a prescaler from the ASC input clock f_{MOD}, e.g. 36 MHz. The baudrate timer is counting downwards and can be started or stopped through the baudrate generator run bit S0CON_R. Each underflow of the timer provides one clock pulse to the serial channel. The timer is reloaded with the value stored in its 13-bit reload register each time it underflows. The resulting clock f_{BRT} is again divided by a factor for the baudrate clock (± 16 in asynchronous modes and ± 4 in synchronous mode). The prescaler is selected by the bits S0CON_BRS and S0CON_FDE. In the asynchronous operating modes, additionally to the two fixed dividers a fractional divider prescaler unit is available which allows to select prescaler divider ratios of n/512 with n=0-511. Therefore, the baudrate of ASC is determined by the module clock, the content of S0FDV, the reload value of S0BG and the operating mode (asynchronous or synchronous).

Register S0BG is the dual-function Baudrate Generator/Reload register. Reading BG returns the content of the timer BR_VALUE (bits 15...13 return zero), while writing to S0BG always updates the reload register (bits 15...13 are insiginificant).

An auto-reload of the timer with the content of the reload register is performed each time S0CON_BG is written to. However, if S0CON_R='0' at the time the write operation to BG is performed, the timer will not be reloaded until the first instruction cycle after S0CON_R='1'. For a clean baudrate initialization S0BG should only be written if S0CON_R='0'. If S0BG is written with S0CON_R='1', an unpredicted behaviour of the ASC may occur during running transmit or receive operations.

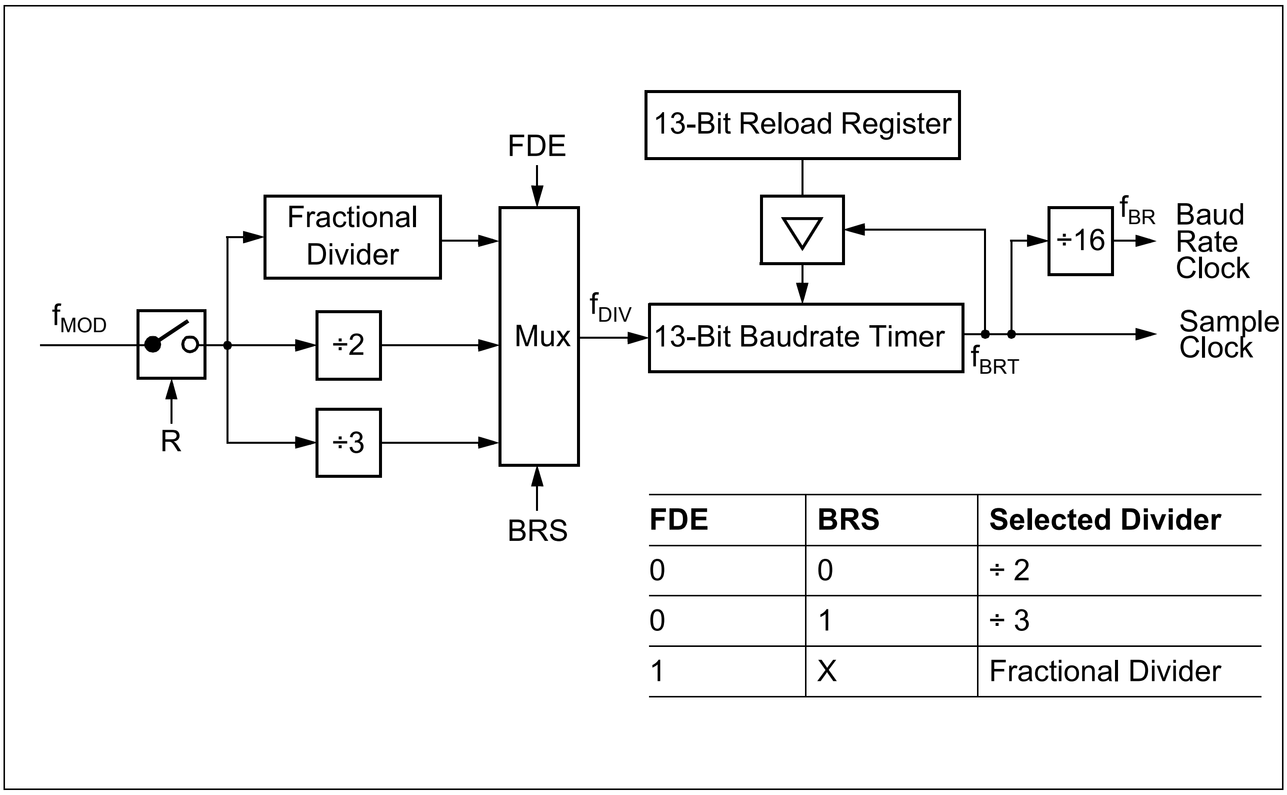
15.1.7.1 Baudrates in Asynchronous Mode

For asynchronous operation, the baudrate generator provides a clock f_{BRT} with 16 times the rate of the established baudrate. Every received bit is sampled at the 7th, 8th and 9th cycle of this clock. The clock divider circuitry, which generates the input clock for the 13-bit baudrate timer, is extended by a fractional divider circuitry, which allows the adjustment of more accurate baudrates and the extension of the baudrate range.

The baudrate of the baudrate generator depends on the following input clock, bits and register values :

- Input clock f_{MOD}
- Selection of the baudrate timer input clock f_{DIV} by bits S0CON_FDE and S0CON_BRS
- If bit S0CON_FDE=1 (fractional divider) : value of register S0CON_FDV
- value of the 13-bit reload register S0BG

The output clock of the baudrate timer with the reload register is the sample clock in the asynchronous modes of the ASC. For baudrate calculations, this baudrate clock f_{BR} is derived from the sample clock f_{DIV} by a division by 16.



FDE	BRS	Selected Divider
0	0	÷ 2
0	1	÷ 3
1	X	Fractional Divider

Figure 15-11 ASC Baudrate Generator Circuitry in Asynchronous Modes

Using the fixed Input Clock Divider

The baudrate for asynchronous operation of serial channel ASC when using the fixed input clock divider ratios (S0CON_FDE=0) and the required reload value for a given baudrate can be determined by the following formulas :

Table 15-4 Asynchronous Baudrate Formulas using the Fixed Input Clock Dividers

FDE	BRS	BG	Formula
0	0	0 ... 8191	$\text{Baudrate} = \frac{f_{MOD}}{32 \times (BG+1)}$ $BG = \frac{f_{MOD}}{32 \times \text{Baudrate}} - 1$
	1		$\text{Baudrate} = \frac{f_{MOD}}{48 \times (BG+1)}$ $BG = \frac{f_{MOD}}{48 \times \text{Baudrate}} - 1$

BG represents the contents of the reload register S0BG (BR_VALUE), taken as unsigned 13-bit integer.

The maximum baudrate that can be achieved for the asynchronous modes when using the two fixed clock dividers and a module clock of 24 MHz is 0.5 MBaud. **Table 15-5** below lists various commonly used baudrates together with the required reload values and the deviation errors compared to the intended baudrate.

Table 15-5 Typical Asynchronous Baudrates using the Fixed Input Clock Dividers

Baudrate		BRS = ‘0’, f_{MOD} = 24 MHz		BRS = ‘1’, f_{MOD} = 24 MHz	
		Deviation Error	Reload Value	Deviation Error	Reload Value
19.2	KBaud	+ 0.16 %	0026_H	+ 0.16 %	0019_H
9600	Baud	+ 0.16 %	$004C_H$	+ 0.16 %	0033_H

Note: S0CON_FDE must be 0 to achieve the baudrates in the table above. The deviation errors given in the table above are rounded. Using a baudrate crystal will provide correct baudrates without deviation errors.

Using the Fractional Divider

When the fractional divider is selected, the input clock f_{DIV} for the baudrate timer is derived from the module clock f_{MOD} by a programmable divider. If S0CON_FDE=1, the fractional divider is activated, It divides f_{MOD} by a fraction of n/512 for any value of n from 0 to 511. If n=0, the divider ratio is 1 which means that $f_{DIV}=f_{MOD}$. In general, the fractional divider allows to program the baudrrate with a much better accuracy than with the two fixed prescaler divider stages.

Table 15-6 Asynchronous Baudrate Formulas using the Fractional Input Clock Divider

FDE	BRS	BG	FDV	Formula
1	X	1 ... 8191	1 ... 511	$\text{Baudrate} = \frac{FDV}{512} \times \frac{f_{MOD}}{16 \times (BG+1)}$
			0	$\text{Baudrate} = \frac{f_{MOD}}{16 \times (BG+1)}$

BG represents the content of the reload register S0BG (BR_VALUE), taken as unsigned 13-bit integer. FDV represents the content of the fractional divider register S0FDV (FD_VALUE) taken as unsigned 9-bit integer. For example, typical asynchronous baudrates are shown in **Table 15-7**.

Using the fractional divider and a module clock of 24 MHz (equal to the INCA-D CPU clock) the available baudrate range is 1 MBaud down to 0.5364 Baud.

.

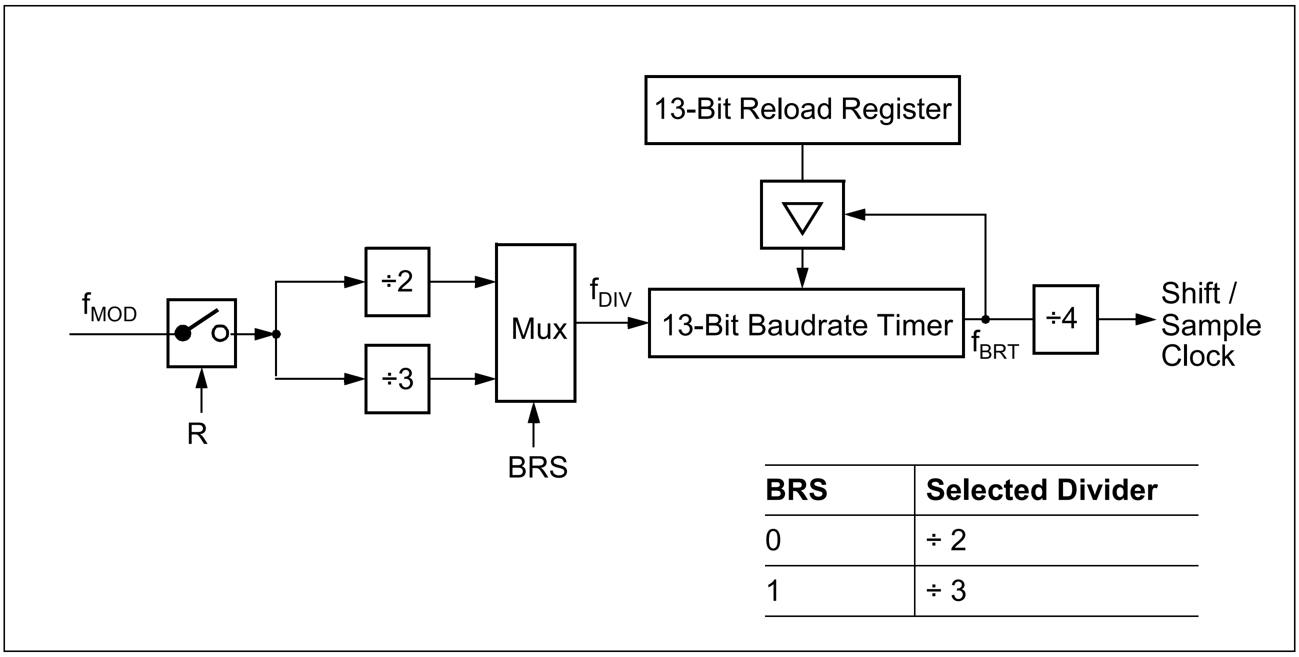
Table 15-7 Typical Asynchronous Baudrates using the Fractional Input Clock Divider

f_{MOD}	Desired Baudrate	BG	FDV	Resulting Baudrate	Deviation
	57.6 kBaud	20	413	57617	< 0.03 %
	38.4 kBaud	27	367	38.400 kBaud	0 %
	19.2 kBaud	55	367	19.200 kBaud	0 %
	min. Baudrate	8191	1	0.53644 Baud	0 %

*Note: The **ApNote AP2423** provides a program 'ASC.EXE' which allows to calculate values for the S0FDV and S0BG registers depending on f_{MOD}, the requested baudrate, and the maximum deviation. Please contact your Infineon Technologies representative.*

15.1.7.2 Baudrates in Synchronous Mode

For synchronous operation, the baudrate generator provides a clock with 4 times the rate of the established baudrate.(see **Figure 15-12**).



BRS	Selected Divider
0	÷ 2
1	÷ 3

Figure 15-12 ASC Baudrate Generator Circuitry in Synchronous Mode

The baudrate for synchronous operation of serial channel ASC can be determined by the formulas as shown in **Table 15-8**.

Table 15-8 Synchronous Baudrate Formulas

BRS	BG	Formula
0	0 ... 8191	Baudrate = $\frac{f_{MOD}}{8 \times (BG+1)}$ BG = $\frac{f_{MOD}}{8 \times Baudrate} - 1$
1		Baudrate = $\frac{f_{MOD}}{12 \times (BG+1)}$ BG = $\frac{f_{MOD}}{12 \times Baudrate} - 1$

BG represents the content of the reload register S0BR (BR_VALUE), taken as unsigned 13-bit integers.

The maximum baudrate that can be achieved in synchronous mode when using a module clock of 36 MHz is 4.5 MBaud.

15.1.8 Hardware Error Detection Capabilities

To improve the safety of serial data exchange, the serial channel ASC provides an error interrupt request flag, which indicates the presence of an error, and three (selectable) error status flags in register S0CON, which indicate which error has been detected during reception. Upon completion of a reception, the error interrupt request line S0EIR will be activated simultaneously with the receive interrupt request line S0RIR, if one or more of the following conditions are met :

- the framing error detection enable bit S0CON_FEN is set and any of the expected stop bits is not high, the framing error flag S0CON_FE is set, indicating that the error interrupt request is due to a framing error (Asynchronous mode only).
- If the parity error detection enable bit S0CON_PEN is set in the modes where a parity bit is received, and the parity check on the received data bits proves false, the parity error flag S0CON_PE is set, indicating that the error interrupt request is due to a parity error (Asynchronous mode only).
- If the overrun error detection enable bit S0CON_OEN is set and the last character received was not read out of the receive buffer by software or DMA transfer at the time the reception of a new frame is complete, the overrun error flag S0CON_OE is set indicating that the error interrupt request is due to an overrun error (Asynchronous and synchronous mode).

15.1.9 Interrupts

There are four different interrupts associated with the serial channel ASC. **S0TIR** indicates a transmit interrupt, **S0TBIR** indicates a transmit buffer interrupt, **S0IR** indicates a receive interrupt and **S0EIR** indicates an error interrupt of the serial channel.

The cause of an error interrupt request (framing, parity, overrun error) can be identified by the error status flags FE, PE, and OE which are located in the control register S0CON.

For normal operation (ie. besides the error interrupt) the ASC provides three interrupt requests to control data exchange via the serial channel:

- **S0TBIR** is activated when data is moved from S0TBUF to the transmit register
- **S0TIR** is activated before the last bit of an async. frame is transmitted, or after the last bit of a synchronous frame has been transmitted
- **S0RIR** is activated when the received frame is moved to S0RBUF.

S0TBIR and S0RIR are connected to dedicated interrupt nodes of the CPU, whereas SEIR and STIR are part of the combined interrupt node COMB2INT (refer to "Interrupt System Structure" on page 8-85).

The transmitter is serviced by two interrupt handlers. This provides advantages for the servicing software.

For single transfers it is sufficient to use the transmitter interrupt (S0TIR), which indicates that the previously loadad data has been transmitted, except for the last bit of an asynchronous frame.)

For multiple back-to-back transfers it is necessary to load the next piece of data at last until the time the last bit of the previous frame has been transmitted. In asynchronous mode this leaves just one bit-time for the handler to respond to the transmitter interrupt request, in synchronous mode it is impossible at all.

Using the transmit buffer interrupt (S0TBIR) to reload transmit data gives the time to transmit a complete frame for the service routine, as TBUF may be reload while the previous data is still being transmitted.

16 The High-Speed Synchronous Serial Interfaces

The INCA-D comprises two High-Speed Synchronous Serial Interfaces SSC0 and SSC1. These interfaces provide flexible high-speed serial communication between the INCA-D and other microcontrollers, microprocessors or external peripherals.

The SSCs support full-duplex and half-duplex synchronous communication up to 12.5 MBaud. The serial clock signals can be generated by the SSCs itself (master mode) or be received from an external master (slave mode). Data width, shift direction, clock polarity and phase are programmable. This allows communication with SPI-compatible devices. Transmission and reception of data is double-buffered. A 16-bit baud rate generator provides each of the the SSC with a separate serial clock signal.

The high-speed synchronous serial interfaces can be configured in a very flexible way, so they can be used with other synchronous serial interfaces (eg. the ASC in synchronous mode), serve for master/slave or multimaster interconnections or operate compatible with the popular SPI interface. So they can be used to communicate with shift registers (IO expansion), peripherals (eg. EEPROMs etc.) or other controllers (networking). The SSCs support half-duplex and full-duplex communication. Data is transmitted or received for SSC0 on pins MTSR0/P3.9 (Master Transmit / Slave Receive) and MRST0/P3.8 (Master Receive / Slave Transmit) and for SSC1 on the pins MTSR1/P3.1 and MRST1/P3.0 respectively. The clock signals are output or input on pin SCLK0/P3.13 or SCLK1/P3.2 respectively. These pins are alternate functions of Port 3 pins.

Figure 16-1 and **Figure 16-2** give an overview about the used registers to control the two SSCs.

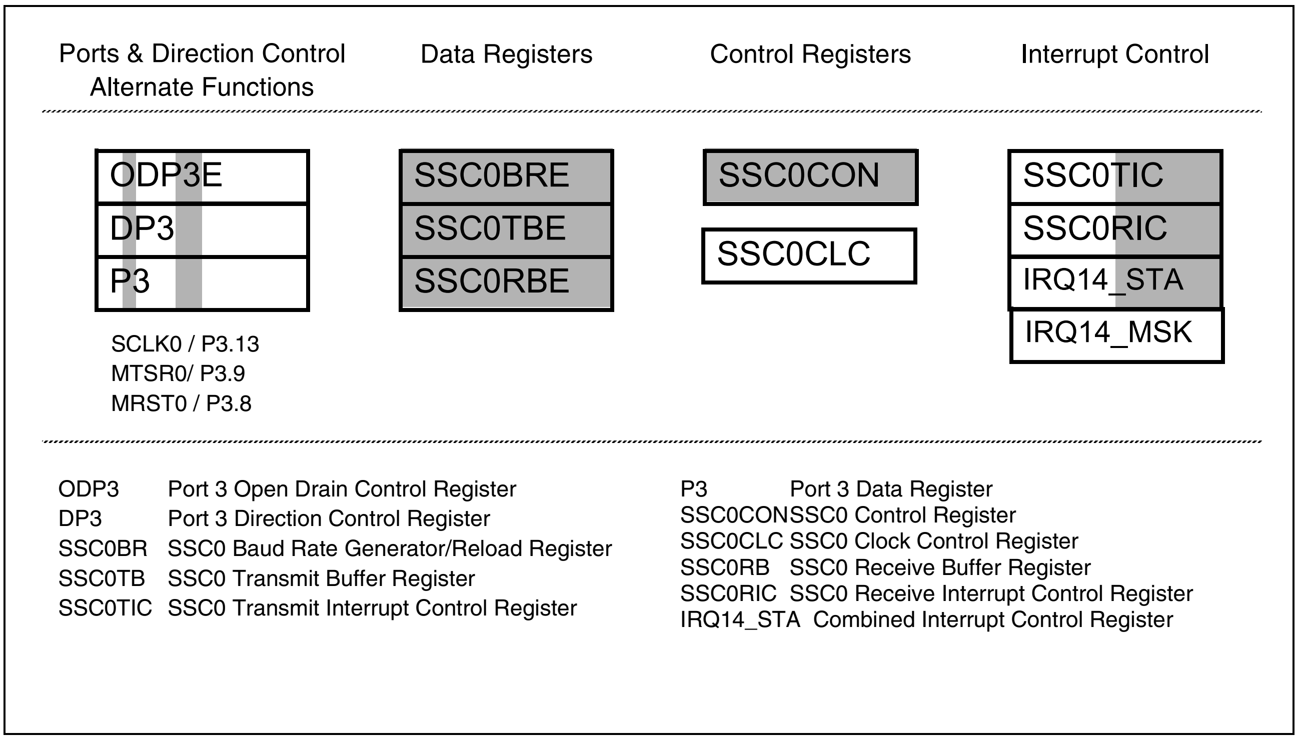

Ports & Direction Control
Alternate Functions
Data Registers
Control Registers
Interrupt Control
ODP3E
DP3
P3
SSC0BRE
SSC0TBE
SSC0RBE
SSC0CON
SSC0CLC
SSC0TIC
SSC0RIC
IRQ14_STA
IRQ14_MSK
SCLK0 / P3.13
MTSR0/ P3.9
MRST0 / P3.8
ODP3 Port 3 Open Drain Control Register
DP3 Port 3 Direction Control Register
SSC0BR SSC0 Baud Rate Generator/Reload Register
SSC0TB SSC0 Transmit Buffer Register
SSC0TIC SSC0 Transmit Interrupt Control Register
P3 Port 3 Data Register
SSC0CONSSC0 Control Register
SSC0CLC SSC0 Clock Control Register
SSC0RB SSC0 Receive Buffer Register
SSC0RIC SSC0 Receive Interrupt Control Register
IRQ14_STA Combined Interrupt Control Register

Figure 16-1 SFRs and Port Pins associated with the SSC0

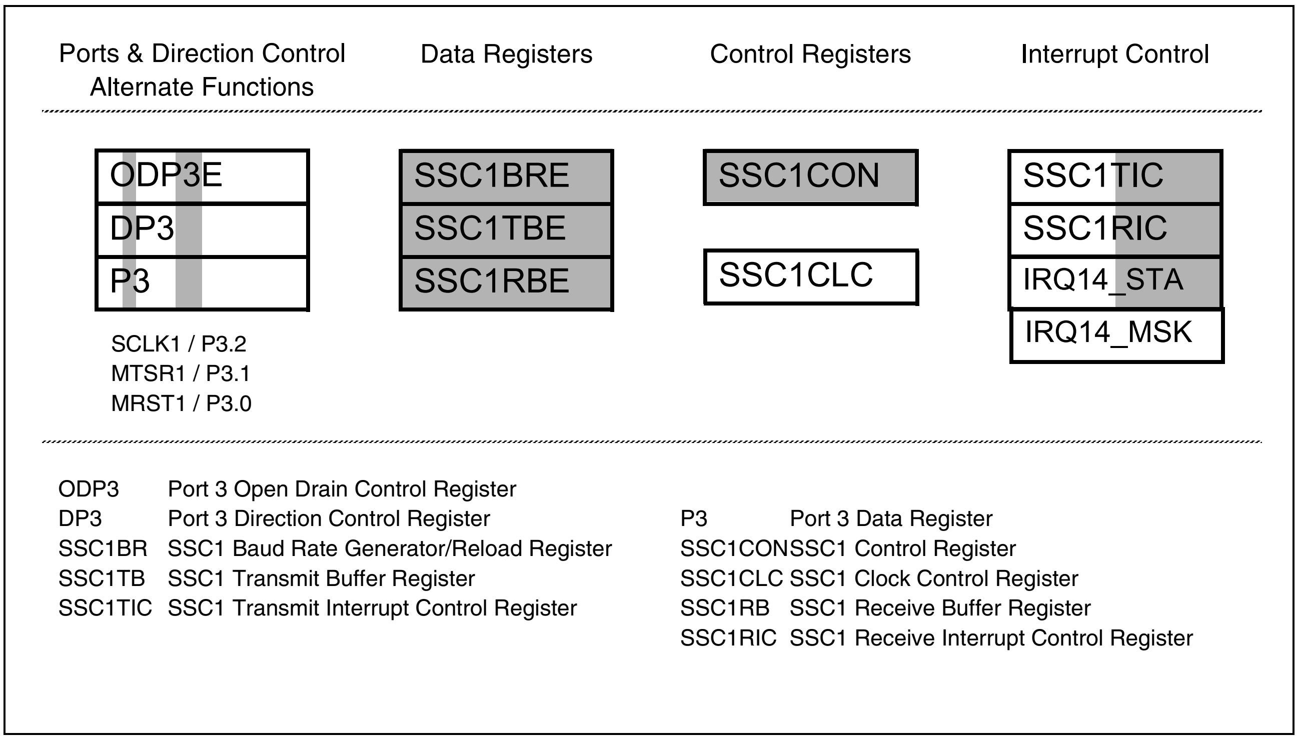

Ports & Direction Control
Alternate Functions
Data Registers
Control Registers
Interrupt Control
ODP3E
DP3
P3
SSC1BRE
SSC1TBE
SSC1RBE
SSC1CON
SSC1CLC
SSC1TIC
SSC1RIC
IRQ14_STA
IRQ14_MSK
SCLK1 / P3.2
MTSR1 / P3.1
MRST1 / P3.0
ODP3 Port 3 Open Drain Control Register
DP3 Port 3 Direction Control Register
SSC1BR SSC1 Baud Rate Generator/Reload Register
SSC1TB SSC1 Transmit Buffer Register
SSC1TIC SSC1 Transmit Interrupt Control Register
P3 Port 3 Data Register
SSC1CONSSC1 Control Register
SSC1CLC SSC1 Clock Control Register
SSC1RB SSC1 Receive Buffer Register
SSC1RIC SSC1 Receive Interrupt Control Register

Figure 16-2 SFRs and Port Pins associated with the SSC1

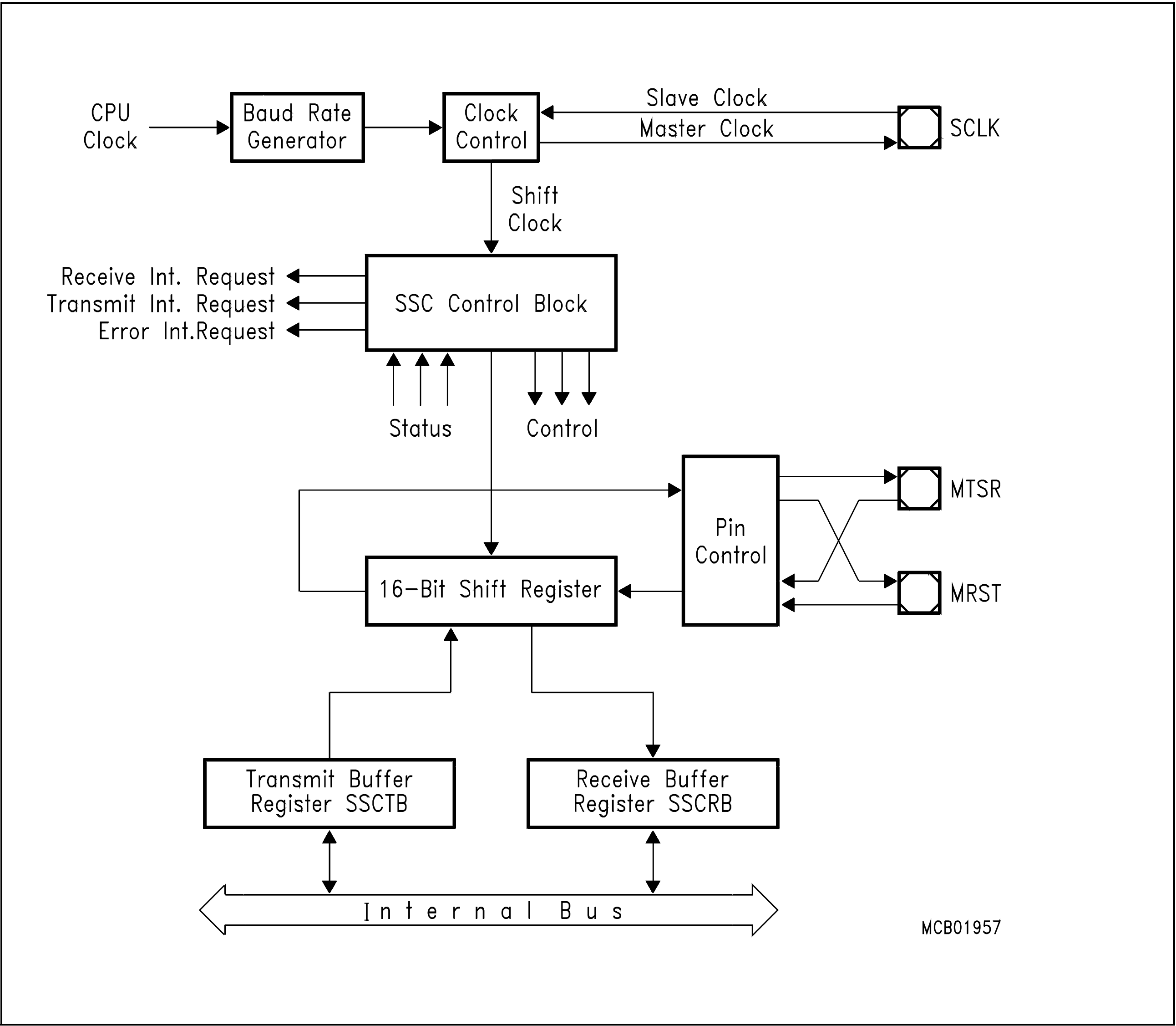


Figure 16-3 Synchronous Serial Channel SSC Block Diagram (SSC0 and SSC1)

The **clock control register** SSC0CLC is located at F0B6$_H$ and SS1CLC can be found at F058$_H$ (see Chapter 23.5)

The operating mode of the serial channels SSC0 and SSC1 is controlled by their bit-addressable control register SSC0CON or SSC1CON respectively. The registers serve for two purposes:

Note: The following descriptions are valid for SSC0 as well as for SSC1.

• during programming (SSC0 disabled by SSCEN0='0') they provide access to a set of control bits,
• during operation (SSC0 enabled by SSC0EN='1') it provides access to a set of status

flags.
Register SSC0CON is shown below in each of the two modes.

Because the register descriptions are valid for SSC0 and SSC1, the corresponding bits are generally denoted by 'x'. This means 'x' is the placeholder for '1' and '2' respectively

SSC0CON (FFB2$_H$ / D9$_H$) **SSC1CON (FF5E$_H$ / AF$_H$)** SFRs **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSCx EN=0	**SSCx MS**	-	**SSCx AREN**	**SSCx BEN**	**SSCx PEN**	**SSCx REN**	**SSCx TEN**	-	**SSCx PO**	**SSCx PH**	**SSCx HB**	**SSCxBM**			
rw	rw	-	rw	rw	rw	rw	rw	-	rw	rw	rw	rw			

Bit	Function (Programming Mode, SSCxEN = '0')
SSCxBM	**SSCx Data Width Selection** 0 : Reserved. Do not use this combination. 1...15 : Transfer Data Width is 2...16 bit (<SSCxBM>+1)
SSCxHB	**SSCx Heading Control Bit** 0 : Transmit/Receive LSB First 1 : Transmit/Receive MSB First
SSCxPH	**SSCx Clock Phase Control Bit** 0 : Shift transmit data on the leading clock edge, latch on trailing edge 1 : Latch receive data on leading clock edge, shift on trailing edge
SSCxPO	**SSCx Clock Polarity Control Bit** 0 : Idle clock line is low, leading clock edge is low-to-high transition 1 : Idle clock line is high, leading clock edge is high-to-low transition
SSCxTEN	**SSCx Transmit Error Enable Bit** 0 : Ignore transmit errors 1 : Check transmit errors
SSCxREN	**SSCx Receive Error Enable Bit** 0 : Ignore receive errors 1 : Check receive errors
SSCxPEN	**SSCx Phase Error Enable Bit** 0 : Ignore phase errors 1 : Check phase errors
SSCxBEN	**SSCx Baudrate Error Enable Bit** 0 : Ignore baudrate errors 1 : Check baudrate errors
SSCxAREN	**SSCx Automatic Reset Enable Bit** 0 : No additional action upon a baudrate error 1 : The SSCx is automatically reset upon a baudrate error

Bit	Function (Programming Mode, SSCxEN = '0')
SSCxMS	**SSCx Master Select Bit** 0 : Slave Mode. Operate on shift clock received via SCLK. 1 : Master Mode. Generate shift clock and output it via SCLK.
SSCxEN	**SSCx Enable Bit = '0'** Transmission and reception disabled. Access to control bits.

SSC0CON (FFB2$_H$ / D9$_H$) **SSC1CON (FF5E$_H$ / AF$_H$)** SFRs **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSCx EN=1	SSCx MS	-	SSCx BSY	SSCx BE	SSCx PE	SSCx RE	SSCx TE	-	-	-	-	SSCxBC			
rw	rw	-	r	rw	rw	rw	rw	-	-	-	-	r			

Bit	Function (Operating Mode, SSCxEN = '1')
SSCxBC	**SSCx Bit Count Field** Shift counter is updated with every shifted bit. **Do not write to!!!**
SSCxTE	**SSCx Transmit Error Flag** 1 : Transfer starts with the slave's transmit buffer not being updated
SSCxRE	**SSCx Receive Error Flag** 1 : Reception completed before the receive buffer was read
SSCxPE	**SSCx Phase Error Flag** 1 : Received data changes around sampling clock edge
SSCxBE	**SSCx Baudrate Error Flag** 1 : More than factor 2 or 0.5 between Slave's actual and expected baudrate
SSCxBSY	**SSCx Busy Flag** Set while a transfer is in progress. **Do not write to!!!**
SSCxMS	**SSCx Master Select Bit** 0 : Slave Mode. Operate on shift clock received via SCLK. 1 : Master Mode. Generate shift clock and output it via SCLK.
SSCxEN	**SSCx Enable Bit = '1'** Transmission and reception enabled. Access to status flags and M/S control.

Note: The target of an access to SSCxCON (control bits or flags) is determined by the state of SSCxEN prior to the access, ie. writing C057$_H$ to SSCxCON in programming mode (SSCxEN='0') will initialize the SSCx (SSCxEN was '0') and then turn it on (SSCxEN='1'). When writing to SSCxCON, make sure that reserved locations receive zeros.

The SSCx baud rate timer reload register SSCxBR containthe 16-bit reload value for the baud rate timer.

SSC0BR (F0B4$_H$ / 5A$_H$) **SSC1BR (F05E$_H$ / 2F$_H$)** **ESFRs** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSCx RL15	**SSCx RL14**	**SSCx RL13**	**SSCx RL12**	**SSCx RL11**	**SSxC RL10**	**SSCx RL9**	**SSCx RL8**	**SSCx RL7**	**SSCx RL6**	**SSCx RL5**	**SSCx RL4**	**SSCx RL3**	**SSCx RL2**	**SSCx RL1**	**SSCx RL0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
SSCxRL15-0	**Baud Rate Timer/Reload Register Value** Reading SSCxxBG returns the 16-bit content of the baud rate timer. Writing SSCxxBG loads the baud rate timer reload register.

The SSCx transmitter buffer register SSCxTB contain the transmit data value.

SSC0TB (F0B0$_H$ / 58$_H$) **SSC1TB (F05A$_H$ / 2D$_H$)** **ESFR** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSCx TD15	**SSCx TD14**	**SSCx TD13**	**SSCx TD12**	**SSCx TD11**	**SSCx TD10**	**SSCx TD9**	**SSCx TD8**	**SSCx TD7**	**SSCx TD6**	**SSCx TD5**	**SSCx TD4**	**SSCx TD3**	**SSCx TD2**	**SSCx TD1**	**SSCx TD0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
SSCxxTD15-0	**Transmit Data Register Value** SSCxTB contains the data to be transmitted. Unselected bits of SSCxTB are ignored during transmission.

The SSCx receiver buffer register SSCxRB contains the receive data value.

SSC0RB (F0B2$_H$ / 59$_H$) **SSC1RB (F05C$_H$ / 2E$_H$)** **ESFRs** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SSCx RD15	**SSCx RD14**	**SSCx RD13**	**SSCx RD12**	**SSCx RD11**	**SSCx RD10**	**SSCx RD9**	**SSCx RD8**	**SSCx RD7**	**SSCx RD6**	**SSCx RD5**	**SSCx RD4**	**SSCx RD3**	**SSCx RD2**	**SSCx RD1**	**SSCx RD0**
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Function
SSCxRD7-0	**Receive Data Register Value** SSCxRB contains the reveived data bits. Unselected bits of SSC0RB will be not valid and should be ignored

The shift registers of SSC0 and SSC1 are connected to both the transmit pin and the receive pin via the pin control logic (see block diagram). Transmission and reception of serial data is synchronized and takes place at the same time, ie. the same number of transmitted bits is also received. Transmit data is written into the Transmit Buffer SSCxTB. It is moved to the shift register as soon as this is empty. An SSC-master (SSCxMS='1') immediately begins transmitting, while an SSC-slave (SSCxMS='0') will wait for an active shift clock. When the transfer starts, the busy flag SSCxBSY is set and a transmit interrupt request (SSCxTIR) will be generated to indicate that SSCxTB may be reloaded again. When the programmed number of bits (2...16) has been transferred, the contents of the shift register are moved to the Receive Buffer SSCxRB and a receive interrupt request (SSCxRIR) will be generated. If no further transfer is to take place (SSCxTB is empty), SSCxBSY will be cleared at the same time. Software should not modify SSCxBSY, as this flag is hardware controlled.

The transfer of serial data bits can be programmed in many respects:

- the data width can be chosen from 2 bits to 16 bits
- transfer may start with the LSB or the MSB
- the shift clock may be idle low or idle high
- data bits may be shifted with the leading or trailing edge of the clock signal
- the baudrate may be set from 274.7 Baud up to 9 MBd (@ 36 MHz CPU clock)
- the shift clock can be generated (master) or received (slave)

This allows the adaptation of the SSCs to a wide range of applications, where serial data transfer is required.

The Data Width Selection supports the transfer of frames of any length, from 2-bit "characters" up to 16-bit "characters". Starting with the LSB (SSCxHB='0') allows communication eg. with ASC devices in synchronous mode (C166 family) or 8051 like serial interfaces. Starting with the MSB (SSCHB='1') allows operation compatible with the SPI interface.
Regardless which data width is selected and whether the MSB or the LSB is transmitted first, the transfer data is always right aligned in registers SSCxTB and SSCxRB, with the LSB of the transfer data in bit 0 of these registers. The data bits are rearranged for transfer by the internal shift register logic. The unselected bits of SSCxTB are ignored, the unselected bits of SSCxRB will be not valid and should be ignored by the receiver service routine.

The Clock Control allows the adaptation of transmit and receive behaviour of the SSCx to a variety of serial interfaces. A specific clock edge (rising or falling) is used to shift out transmit data, while the other clock edge is used to latch in receive data. Bit SSCxPH selects the leading edge or the trailing edge for each function. Bit SSCxPO selects the level of the clock line in the idle state. So for an idle-high clock the leading edge is a falling one, a 1-to-0 transition. The figure below is a summary.

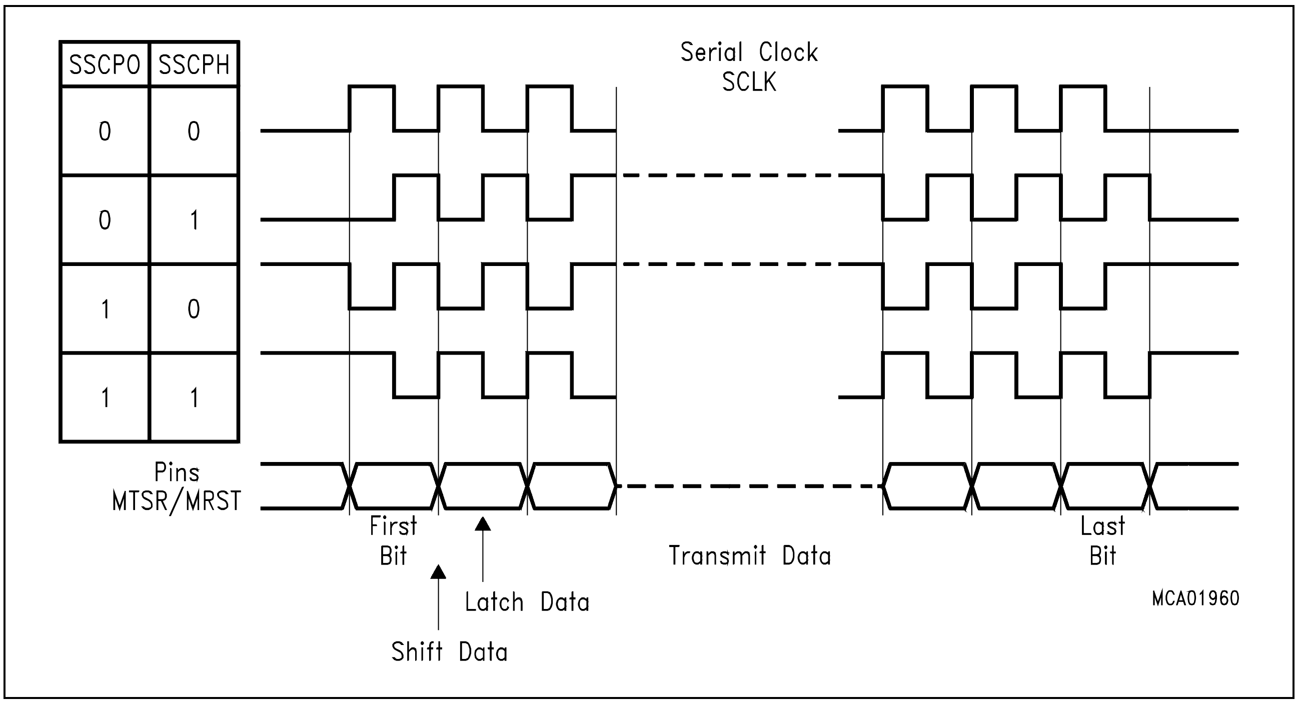


Figure 16-4 Serial Clock Phase and Polarity Options

16.1 Full-Duplex Operation

The different devices are connected through three lines. The definition of these lines is always determined by the master: The line connected to the master's data output pin MTSRx is the transmit line, the receive line is connected to its data input line MRSTx, and the clock line is connected to pin SCLKx. Only the device selected for master operation generates and outputs the serial clock on pin SCLKx. All slaves receive this clock, so their pin SCLKx must be switched to input mode. The output of the master's shift register is connected to the external transmit line, which in turn is connected to the slaves' shift register input. The output of the slaves' shift register is connected to the external receive line in order to enable the master to receive the data shifted out of the slave. The external connections are hard-wired, the function and direction of these pins is determined by the master or slave operation of the individual device.

Note: The shift direction shown in the figure applies for MSB-first operation as well as for LSB-first operation.

When initializing the devices in this configuration, select one device for master operation (SSCxMS='1'), all others must be programmed for slave operation (SSCxMS='0'). Initialization includes the operating mode of the device's SSCx and also the function of the respective port lines (see "Port Control").

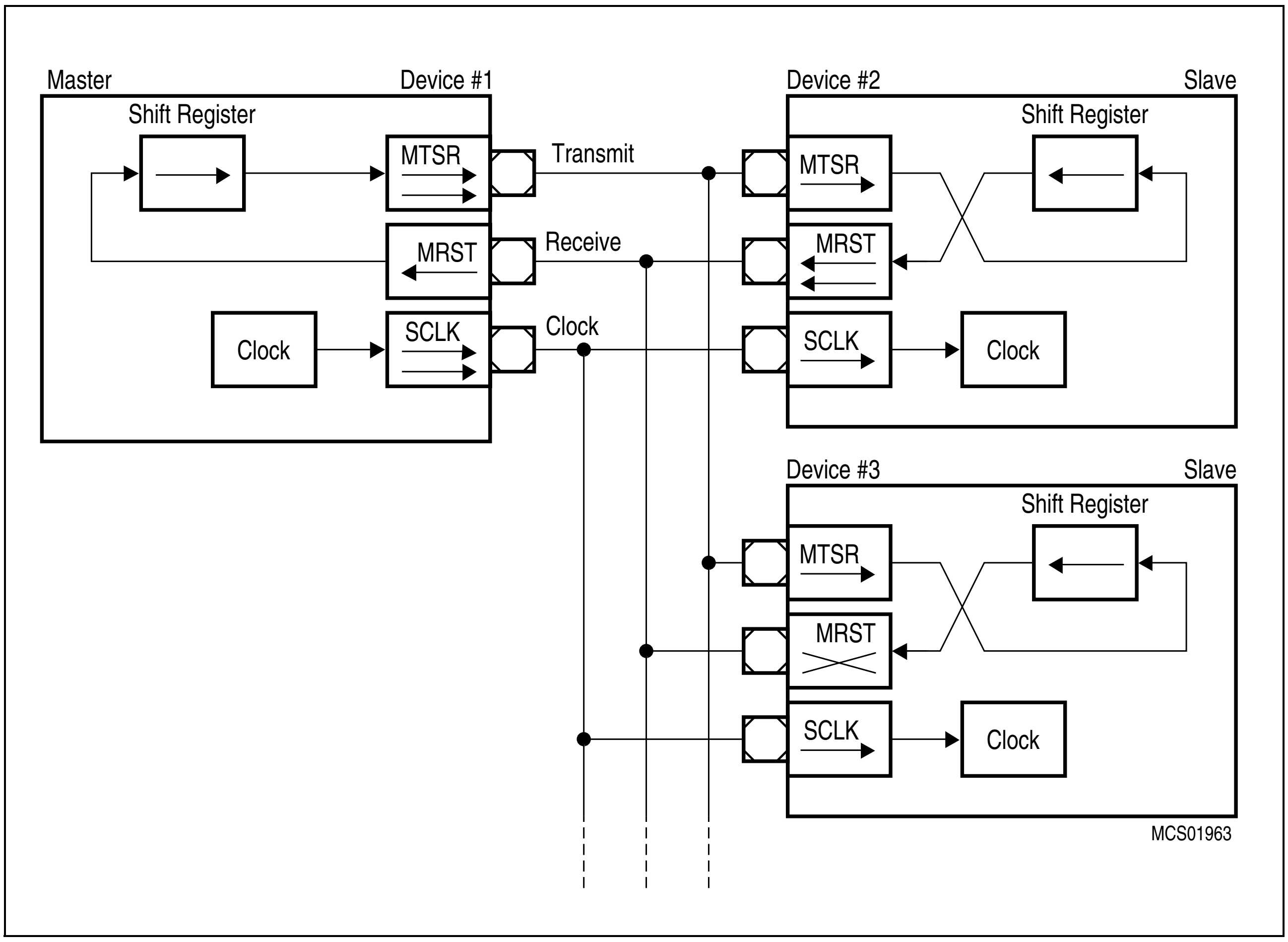


Figure 16-5 Full Duplex Configuration (valid for SSC0 and SSC1)

The data output pins MRSTx of all slave devices are connected together onto the one receive line in this configuration. During a transfer each slave shifts out data from its shift register. There are two ways to avoid collisions on the receive line due to different slave data:

Only one slave drives the line, ie. enables the driver of its MRSTx pin. All the other slaves have to program their MRSTx pins to input. So only one slave can put its data onto the master's receive line. Only receiving of data from the master is possible. The master selects the slave device from which it expects data either by separate select lines, or by sending a special command to this slave. The selected slave then switches its MRSTx line to output, until it gets a deselection signal or command.

The slaves use open drain output on MRSTx. This forms a Wired-AND connection. The receive line needs an external pullup in this case. Corruption of the data on the receive line sent by the selected slave is avoided, when all slaves which are not selected for transmission to the master only send ones (‘1’). Since this high level is not actively driven onto the line, but only held through the pullup device, the selected slave can pull this line actively to a low level when transmitting a zero bit. The master selects the slave device from which it expects data either by separate select lines, or by sending a special command to this slave.

After performing all necessary initializations of the SSCx, the serial interfaces can be enabled. For a master device, the alternate clock line will now go to its programmed polarity. The alternate data line will go to either '0' or '1', until the first transfer will start. After a transfer the alternate data line will always remain at the logic level of the last transmitted data bit.

When the serial interfaces are enabled, the master device can initiate the first data transfer by writing the transmit data into register SSCxTB. This value is copied into the shift register (which is assumed to be empty at this time), and the selected first bit of the transmit data will be placed onto the MTSRx line on the next clock from the baudrate generator (transmission only starts, if SSCxEN='1'). Depending on the selected clock phase, also a clock pulse will be generated on the SCLKx line. With the opposite clock edge the master at the same time latches and shifts in the data detected at its input line MRSTx. This "exchanges" the transmit data with the receive data. Since the clock line is connected to all slaves, their shift registers will be shifted synchronously with the master's shift register, shifting out the data contained in the registers, and shifting in the data detected at the input line. After the preprogrammed number of clock pulses (via the data width selection) the data transmitted by the master is contained in all slaves' shift registers, while the master's shift register holds the data of the selected slave. In the master and all slaves the content of the shift register is copied into the receive buffer SSCxRB and the receive interrupt flag SSCxRIR is set.

A slave device will immediately output the selected first bit (MSB or LSB of the transfer data) at pin MRSTx, when the content of the transmit buffer is copied into the slave's shift register. It will not wait for the next clock from the baudrate generator, as the master does. The reason for this is that, depending on the selected clock phase, the first clock edge generated by the master may be already used to clock in the first data bit. So the slave's first data bit must already be valid at this time.

*Note: On the SSCx always a transmission **and** a reception takes place at the same time, regardless whether valid data has been transmitted or received. This is different eg. from asynchronous reception on ASC.*

The initialization of the SCLKx pin on the master requires some attention in order to avoid undesired clock transitions, which may disturb the other receivers. The state of the internal alternate output lines is '1' as long as the SSCx is disabled. This alternate output signal is ANDed with the respective port line output latch. Enabling the SSCx with an idle-low clock (SSCxPO='0') will drive the alternate data output and (via the AND) the port pin SCLKx immediately low. To avoid this, use the following sequence:

- select the clock idle level (SSCxPO=’x’)
- load the port output latch with the desired clock idle level
- switch the pin to output
- enable the SSC (SSCxEN=’1’)
- if SSCxPO=’0’: enable alternate data output

The same mechanism as for selecting a slave for transmission (separate select lines or special commands) may also be used to move the role of the master to another device in the network. In this case the previous master and the future master (previous slave) will have to toggle their operating mode (SSCxMS) and the direction of their port pins (see description above).

16.2 Half Duplex Operation

In a half duplex configuration only one data line is necessary for both receiving **and** transmitting of data. The data exchange line is connected to both pins MTSRx and MRSxT of each device, the clock line is connected to the SCLKx pin.

The master device controls the data transfer by generating the shift clock, while the slave devices receive it. Due to the fact that all transmit and receive pins are connected to the one data exchange line, serial data may be moved between arbitrary stations.

Similar to full duplex mode there are **two ways to avoid collisions** on the data exchange line:

- only the transmitting device may enable its transmit pin driver
- the non-transmitting devices use open drain output and only send ones.

Since the data inputs and outputs are connected together, a transmitting device will clock in its own data at the input pin (MRSTx for a master device, MTSRx for a slave). By these means any corruptions on the common data exchange line are detected, where the received data is not equal to the transmitted data.

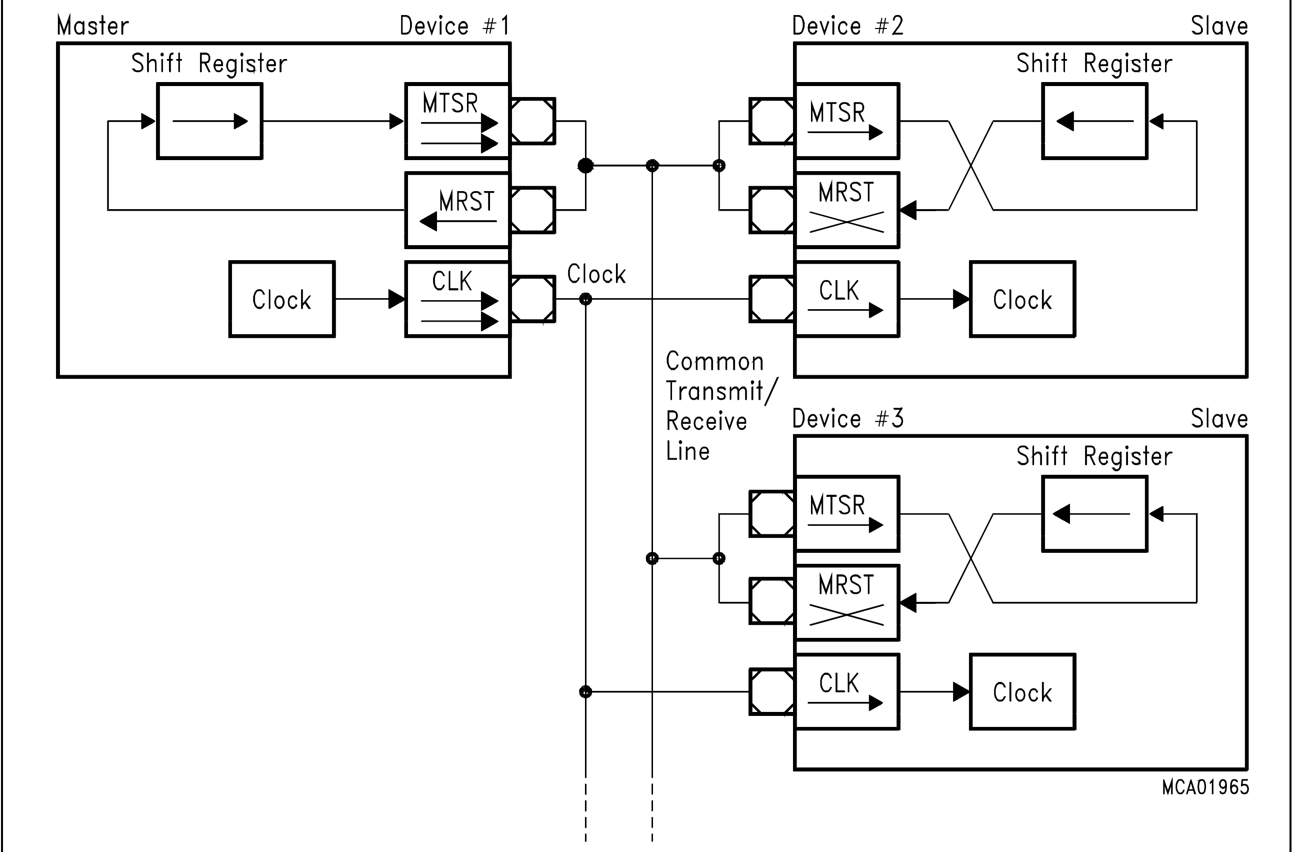


Figure 16-6 xHalf Duplex Configuration (valid for SSC0 and SSC1)

Continuous Transfers

When the transmit interrupt request flag is set, it indicates that the transmit buffer SSCxTB is empty and ready to be loaded with the next transmit data. If SSCxTB has been reloaded by the time the current transmission is finished, the data is immediately transferred to the shift register and the next transmission will start without any additional delay. On the data line there is no gap between the two successive frames. Eg. two byte transfers would look the same as one word transfer. This feature can be used to interface with devices which can operate with or require more than 16 data bits per transfer. It is just a matter of software, how long a total data frame length can be. This option can also be used eg. to interface to byte-wide and word-wide devices on the same serial bus.

Note: Of course, this can only happen in multiples of the selected basic data width, since it would require disabling/enabling of the SSCx to reprogram the basic data width on-the-fly.

Port Control

The SSCx uses three pins of Port 3 to communicate with the external world.

The operation of these pins depends on the selected operating mode (master or slave). In order to enable the alternate output functions of these pins instead of the general purpose I/O operation, the respective port latches have to be set to '1', since the port latch outputs and the alternate output lines are ANDed. When an alternate data output line is not used (function disabled), it is held at a high level, allowing I/O operations via the port latch. The direction of the port lines depends on the operating mode. The SSCx will automatically use the correct alternate input or output line of the ports when switching modes. The direction of the pins, however, must be programmed by the user, as shown in the tables. Using the open drain output feature helps to avoid bus contention problems and reduces the need for hardwired hand-shaking or slave select lines. In this case it is not always necessary to switch the direction of a port pin.

16.3 Baud Rate Generation

The serial channel SSCx has its own dedicated 16-bit baud rate generator with 16-bit reload capability, allowing baud rate generation independent from the timers.

The baud rate generator is clocked with the CPU clock divided by 2 ($f_{CPU}/2$). The timer is counting downwards and can be started or stopped through the global enable bit SSCxEN in register SSCxCON. Register SSCxBR is the dual-function Baud Rate Generator/Reload register. Reading SSCxBR, while the SSCx is enabled, returns the content of the timer. Reading SSCxBR, while the SSCx is disabled, returns the programmed reload value. In this mode the desired reload value can be written to SSCxBR.

Note: Never write to SSCxBR, while the SSCx is enabled.

The formulas below calculate either the resulting baud rate for a given reload value, or the required reload value for a given baudrate:

$$B_{SSC} = \frac{f_{CPU}}{2 * (\text{<SSCBR>} + 1)} \qquad \text{SSCBR} = \left(\frac{f_{CPU}}{2 * \text{Baudrate}_{SSC}} \right) - 1$$

<SSCxBR> represents the content of the reload register, taken as unsigned 16-bit integer.

The maximum baud rate that can be achieved when using a CPU clock of 36 MHz is 18 MBaud in SSCx Master Mode (<SSCxBR>= '0_d'), while in SSCx Slave Mode the maximum baud rate is 9 MBaud (<SSCxBR>= '1_d'; - <SSCxBR>='0_d' is not allowed in

Slave Mode -). The minimum baud rate is 274.66 Baud (<SSCxBR> = ’$\mathrm{FFFF_H}$’ = ’65535_D’).

16.4 Error Detection Mechanisms

The SSCx is able to detect four different error conditions. Receive Error and Phase Error are detected in all modes, while Transmit Error and Baudrate Error only apply to slave mode. When an error is detected, the respective error flag is set. When the corresponding Error Enable Bit is set, also an error interrupt request will be generated by setting SSCxEIR (see figure below). The error interrupt handler may then check the error flags to determine the cause of the error interrupt. The error flags are not reset automatically (like SSCxEIR), but rather must be cleared by software after servicing. This allows servicing of some error conditions via interrupt, while the others may be polled by software.

Note: The error interrupt handler must clear the associated (enabled) errorflag(s) to prevent repeated interrupt requests.

A **Receive Error** (Master or Slave mode) is detected, when a new data frame is completely received, but the previous data was not read out of the receive buffer register SSCxRB. This condition sets the error flag SSCxRE and, when enabled via SSCxREN, the error interrupt request flag SSCxEIR. The old data in the receive buffer SSCxRB will be overwritten with the new value and is unretrievably lost.

A **Phase Error** (Master or Slave mode) is detected, when the incoming data at pin MRST (master mode) or MTSR (slave mode), sampled with the same frequency as the CPU clock, changes between one sample before and two samples after the latching edge of the clock signal (see “Clock Control”). This condition sets the error flag SSCxPE and, when enabled via SSCxPEN, the error interrupt request flag SSCxEIR.

A **Baud Rate Error** (Slave mode) is detected, when the incoming clock signal deviates from the programmed baud rate by more than 100%, ie. it either is more than double or less than half the expected baud rate. This condition sets the error flag SSCxBE and, when enabled via SSCxBEN, the error interrupt request flag SSCxEIR. Using this error detection capability requires that the slave's baud rate generator is programmed to the same baud rate as the master device. This feature detects false additional, or missing pulses on the clock line (within a certain frame).

Note: If this error condition occurs and bit SSCxAREN=’1’, an automatic reset of the SSCx will be performed in case of this error. This is done to reinitialize the SSCx, if too few or too many clock pulses have been detected.

A **Transmit Error** (Slave mode) is detected, when a transfer was initiated by the master (shift clock gets active), but the transmit buffer SSCxTB of the slave was not updated since the last transfer. This condition sets the error flag SSCxTE and, when enabled via SSCxTEN, the error interrupt request flag SSCxEIR. If a transfer starts while the transmit buffer is not updated, the slave will shift out the 'old' contents of the shift register, which normally is the data received during the last transfer.
This may lead to the corruption of the data on the transmit/receive line in half-duplex mode (open drain configuration), if this slave is not selected for transmission. This mode requires that slaves not selected for transmission only shift out ones, ie. their transmit buffers must be loaded with 'FFFF$_H$' prior to any transfer.

Note: A slave with push/pull output drivers, which is not selected for transmission, will normally have its output drivers switched. However, in order to avoid possible conflicts or misinterpretations, it is recommended to always load the slave's transmit buffer prior to any transfer.

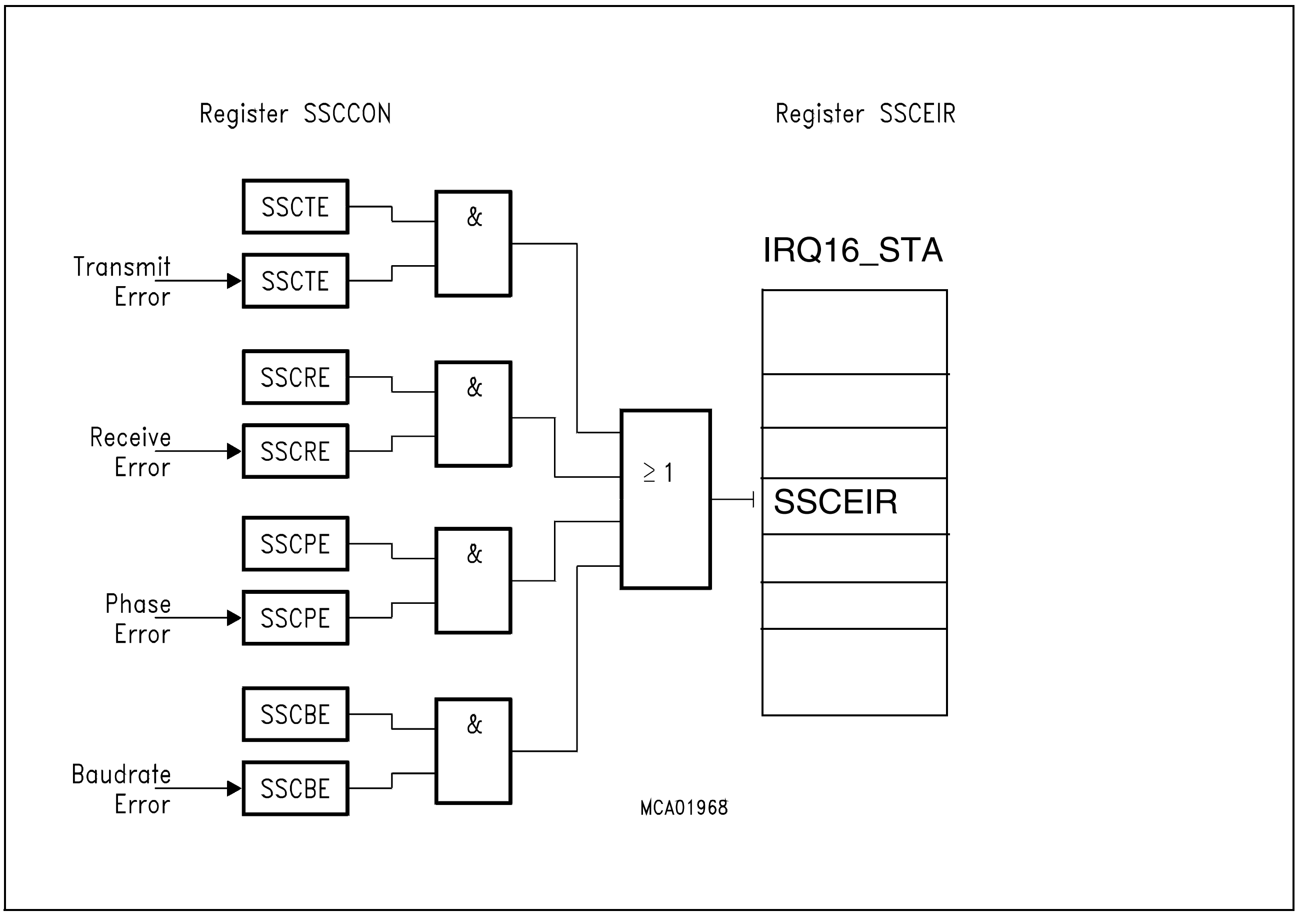


Figure 16-7 Error Interrupt Control (valid for SSC0 and SSC1)

16.5 SSCx Interrupt Control

The SSC0 and SSC1 (SSCx) can generate three different interrupts: SSCxTINT (transmit), SSCxRINT (receive), and SSCxEIR (error). SSCxTINT and SSCxRINT are connected to dedicated interrupt nodes. SSCxEIR is handeled by the combined interrupt COMB2INT (register **IRQ14_STA**). For a description of the interrupt structure refer to chapter "Interrupt System Structure" on page 8-85.

Register SSCxTIC controls the transmit interrupt and register SSCxRIC controls the receive interrupt. The cause of an error interrupt request (receive, phase, baudrate, transmit error) can be identified by the error status flags in control register SSCxCON.

Note: Please refer to the general Interrupt Control Register description for an explanation of the control fields (chapter "Interrupt Control Registers" on page 8-99)

17 IOM-2 Handler, TIC/CI Handler and HDLC Controller

17.1 IOM2 Handler

One task of the IOM handler is to establish a communication path from DSP to the line interface and vice versa.

Beside that function, it handles also the data transfer between line transveiver, HDLC controller and TIC/CI channel handler. Additionally, it provides CPU access to all IOM-2 timeslots via the four controller data access (CDA) register and the IOM-2 Data Transfer Unit.

The PCM data of the controller data access (CDA) registers and the IOM-2 Data Transfer Unit can be configured by programming the corresponding time slot and data port selection registers.

The IOM-2 handler provides access to the following blocks:

- DSP
- Transceiver
- C/I channel handler (C/I0,C/I1)
- TIC bus handler and
- HDLC controller

The following **Figure 17-1** shows the architecture of the IOM-2 handler.

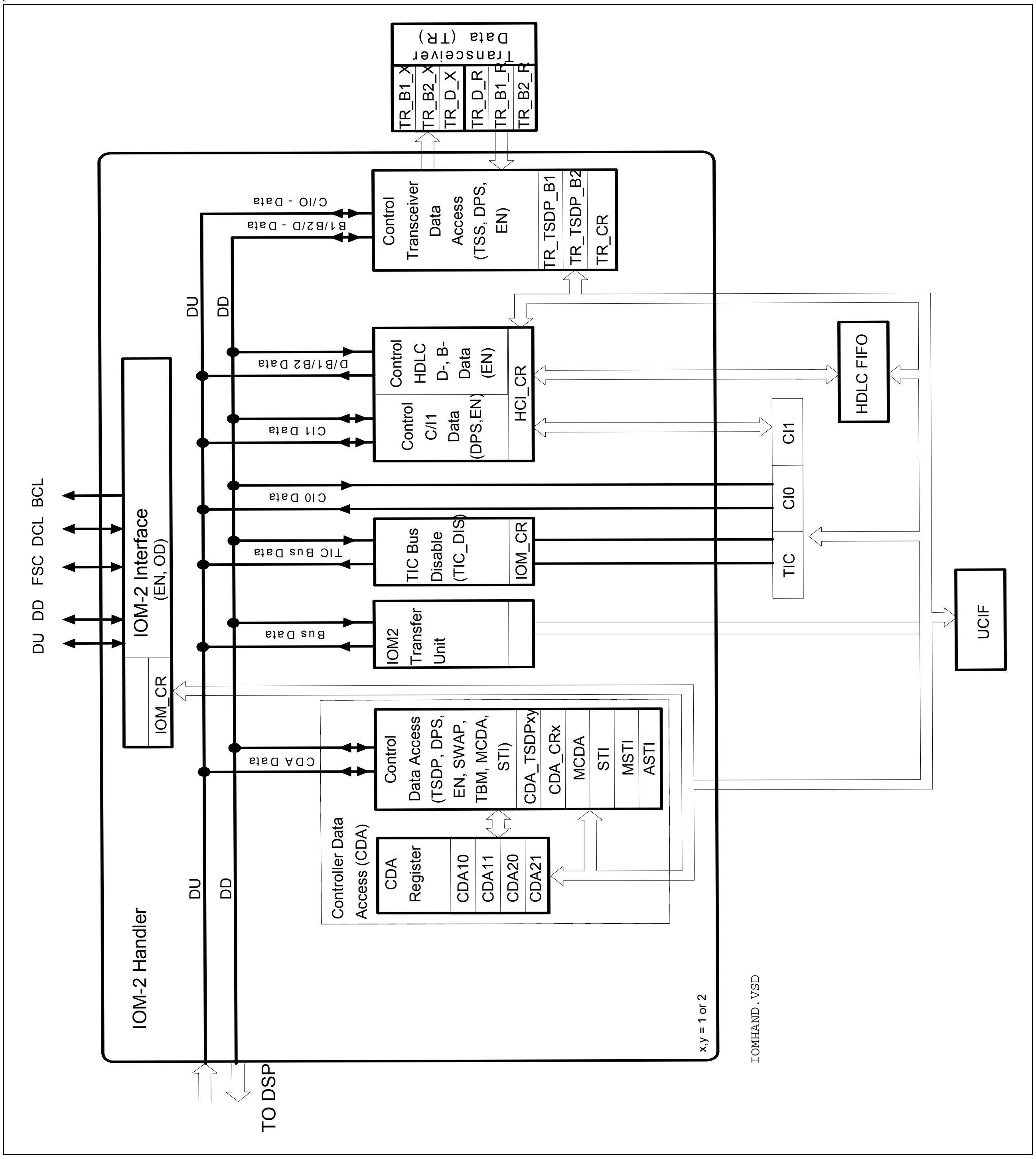


Figure 17-1 Architecture of the IOM-2 Handler

17.1.1 IOM-2 Interface

The IOM-2 interface consists of four lines: FSC, DCL, DD and DU. The rising edge of FSC indicates the start of an IOM-2 frame. The FSC signal is generated by the receive DPLL (Adjust Unit) which synchronizes to the received line frame. The DCL and the BCL output clock signals synchronize the data transfer on both data lines. The DCL is twice the bit rate, the BCL output rate is equal to the bit rate. The bits are shifted out with the rising edge of the first DCL clock cycle and sampled at the falling edge of the second clock cycle.

The FSC signal is an 8 kHz frame sync signal. With DCL = 1.536 MHz, 3 channels of 4 timeslots are available.

The frame structure on the IOM-2 data ports (DU,DD) is shown in figure 17-2.

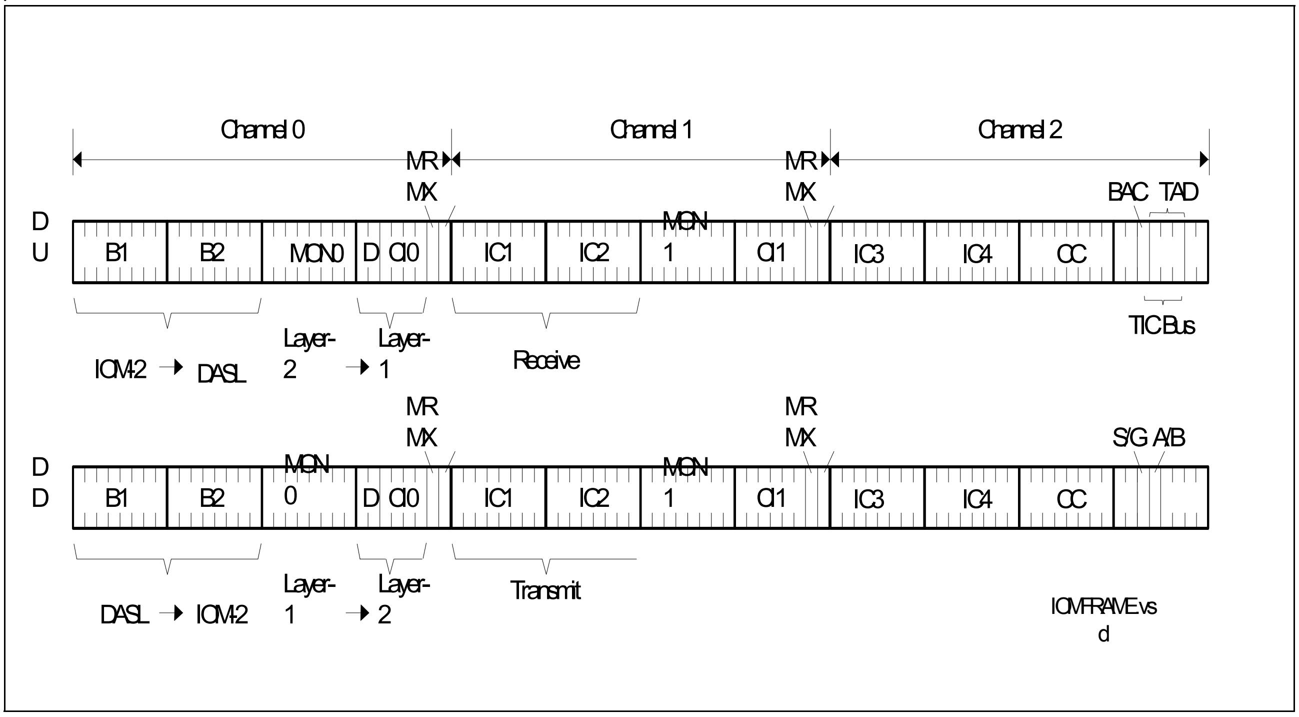


Figure 17-2 IOM®-2 Frame Structure

- Channel 0 contains 144-kbit/s of user and signaling data (2B + D), a MONITOR programming channel (MON0) and a command/indication channel (CI0) for control and programming of the layer-1 transceiver.
- Channel 1 contains two 64-kbit/s intercommunication channels (IC) plus a MONITOR and command/indicate channel (MON1, CI1) to program transfer data to other IOM-2 devices.
- Channel 2 is used for the TIC-bus access. Additionally channel 2 supports further IC3, IC4 and CC channel.

17.1.2 Controller Data Access (CDA)

With its four controller data access registers (CDA10, CDA11, CDA20, CDA21) the IOM-2 handler provides one solution for the CPU to access the IOM-2 time slots.The functional unit CDA (controller data access) allows with its control and configuration registers

- looping of up to four independent PCM channels from DU to DD or vice versa over the four CDA registers
- shifting of two independent PCM channels to another two independent PCM channels on both data ports (DU, DD). Between reading and writing the data can be manipulated (processed with an algorithm) by the microcontroller. If this is not the case a switching function is performed.
- monitoring of up to four time slots on the IOM-2 interface simultaneously
- CPU read and write access to each PCM timeslot

The access principle which is identical for the two channel register pairs CDA10/11 and CDA20/21 is illustrated in figure 17-3. The index variables x,y used in the following description can be 1 or 2 for x and 0 or 1 for y. The prefix 'CDA_' from the register names has been omitted for simplification.

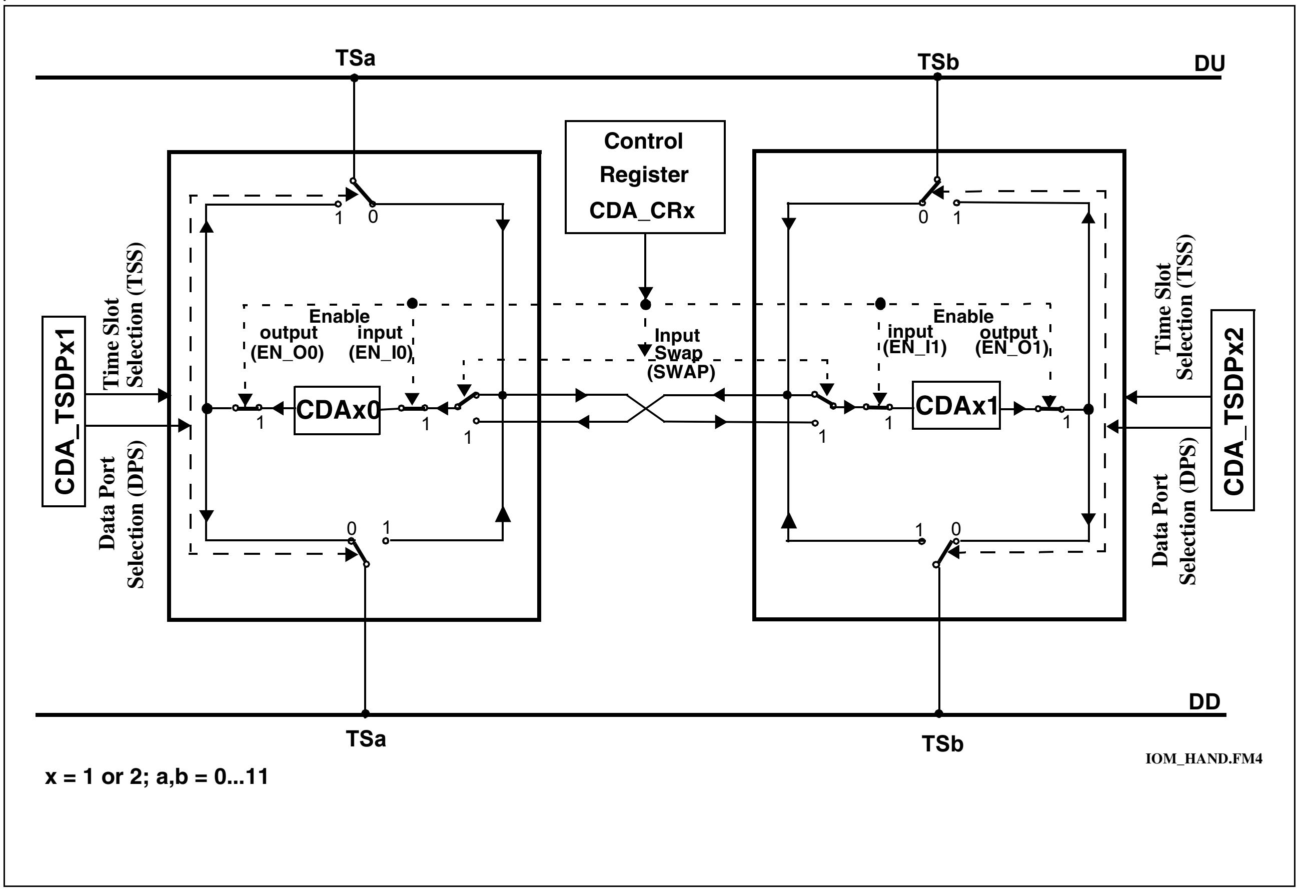


Figure 17-3 Access principle to CDA registers

To each of the four CDAxy data registers a TSDPxy register is assigned by which the time slot and the data port can be determined. With the TSS (Time Slot Selection) bits a time slot can be selected. With the DPS (Data Port Selection) bit the output of the CDAxy register can be assigned to DU or DD respectively. The time slot and data port for the output of CDAxy is always defined by its own TSDPxy register. The input of CDAxy depends on the SWAP bit in the control registers CRx.

If the SWAP bit = ’0’ (swap is disabled) the time slot and data port for the input and output of the CDAxy register is defined by its own TSDPxy register.
If the SWAP bit = ’1’ (swap is enabled) the input port and timeslot of the CDAx0 is defined by the TSDP register of CDAx1 and the input port and timeslot of CDAx1 is defined by the TSDP register of CDAx0. The input definition for time slot and data port CDAx0 are thus swapped to CDAx1 and for CDAx1 swapped to CDAx0. The output timeslots are not affected by SWAP.

The input and output of every CDAxy register can be enabled or disabled by setting the corresponding EN (-able) bit in the control register CDAx_CR. If the input of a register is disabled the output value in the register is retained

17.1.2.1 Looping and Shifting Data

Figure 17-4 gives examples for typical configurations with the above explained control and configuration possibilities with the bits TSS, DPS, EN and SWAP in the registers TSDPxy or CDAx_CR:

a) looping IOM-2 time slot data from DU to DD or vice versa (SWAP = 0)

b) shifting data from TSa to TSb and TSc to TSd in both transmission directions (SWAP = 1)

c) switching data from TSa to TSb and looping from DU to DD or switching TSc to TSd and looping from DD to DU respectively

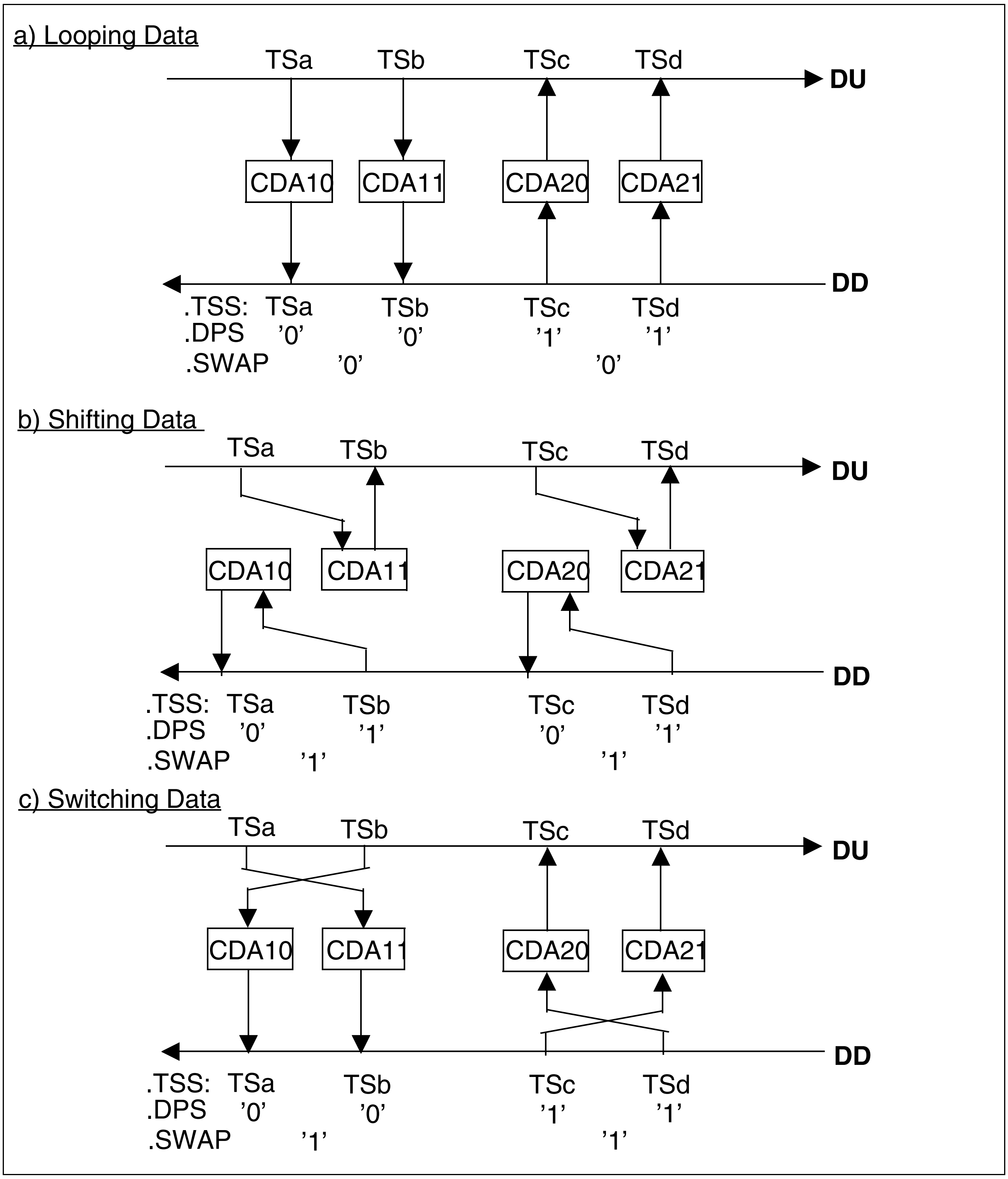


Figure 17-4 Examples for Data Access via CDAxy Registers
a) Looping Data
b) Shifting (Switching) Data
c) Shifting and Looping Data

Figure 17-5 shows the timing of looping TSa from DU to DD via CDAxy register. TSa is read in the CDAxy register from DU and is written one frame later on DD.

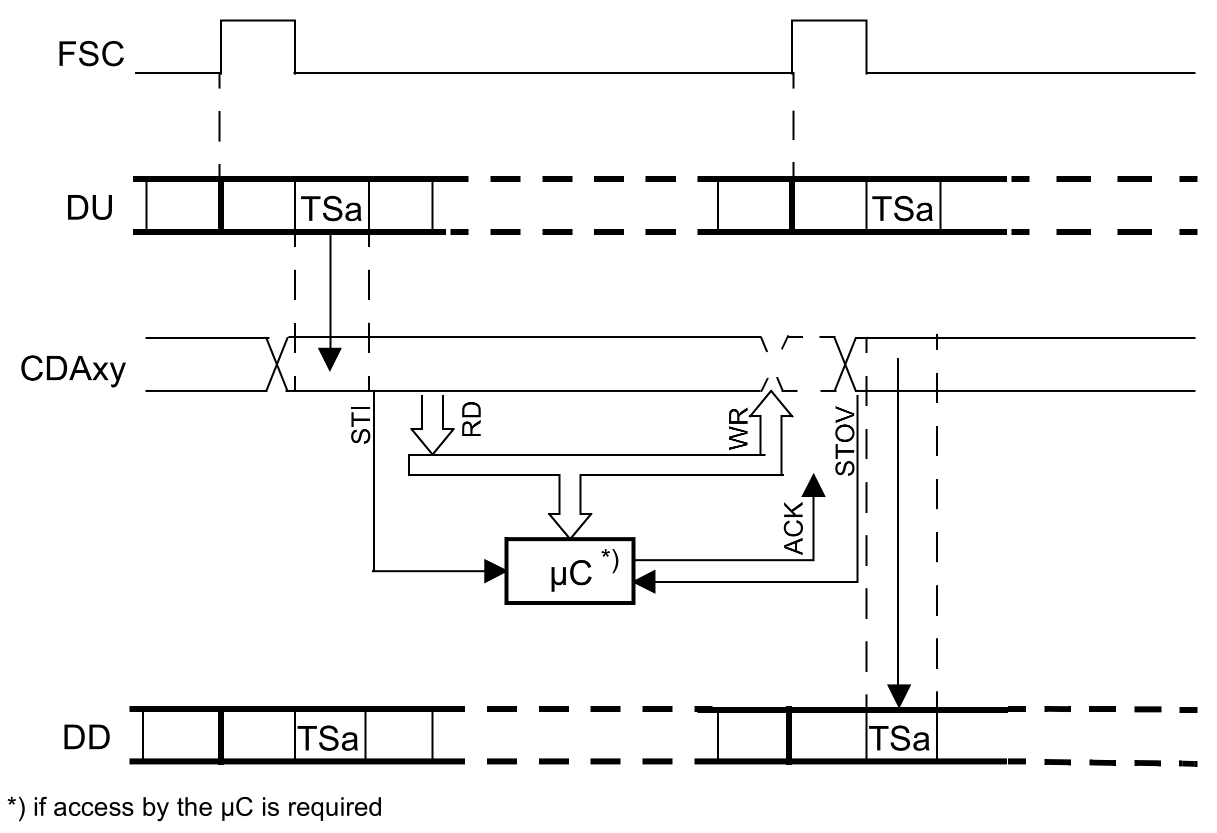
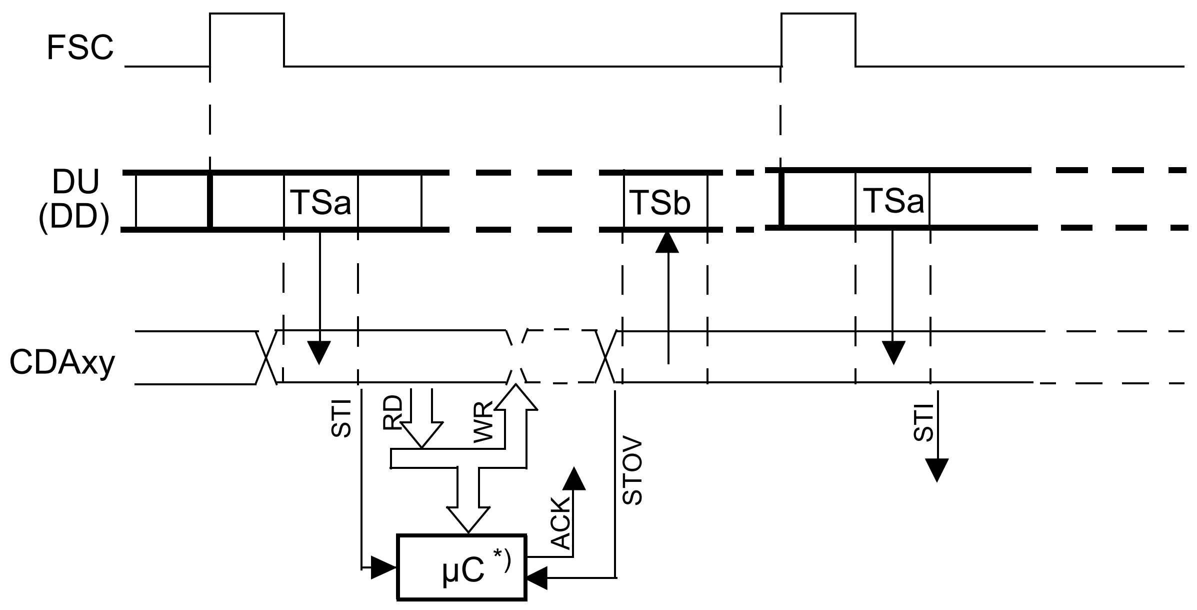


Figure 17-5 Data Access when Looping TSa from DU to DD

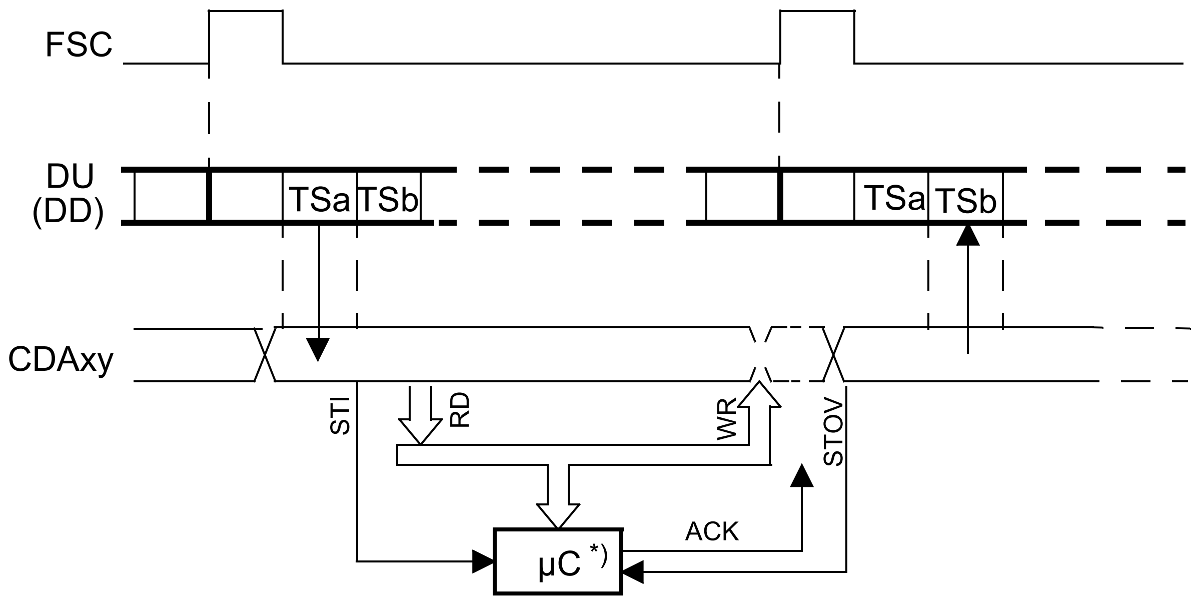
Figure 17-6 shows the timing of shifting data from TSa to TSb on DU(DD) asuming an example with 12 timeslots per FSC frame. In **Figure 17-6a)** shifting is done in one frame because TSa and TSb didn't succeed directly to one another (a = 0...9 and b ≥ a+2). In **figure 17-6b)** shifting is done from one frame to the following frame. This is the case when the time slots succeed one other (b = a+1) or b is smaller than a (b < a).

At looping and shifting the data can be accessed by the controller between the synchronous transfer interrupt (STI) and the status overflow interrupt (STOV). STI and STOV are explained in the section 'Synchronous Transfer'. If there is no controller intervention the looping and shifting is done autonomously.

a) Shifting TSa → TSb within one frame
(a,b: 0...11 and b ≥ a+2)



b) Shifting TSa → TSb in the next frame
(a,b: 0...11 and (b = a+1 or b <a)



*) if access by the µC is required

Figure 17-6 Data Access when Shifting TSa to TSb on DU (DD)

17.1.3 Monitoring Data

Figure 17-7 gives an example for monitoring of two IOM-2 time slots each on DU or DD simultaneously. For monitoring on DU and/or DD the channel registers with even numbers (CDA10, CDA20) are assigned to time slots with even numbers TS(2n) and the channel registers with odd numbers (CDA11, CDA21) are assigned to time slots with odd numbers TS(2n+1). The user has to take care of this restriction by programming the appropriate time slots.

This mode is only valid if two blocks (e.g. transceiver and HDLC controller) are programmed to these timeslots and communicating via IFA.

However, if only one block is programmed to this timeslot the timeslots for CDAx0 and CDAx1 can be programmed completely independently.

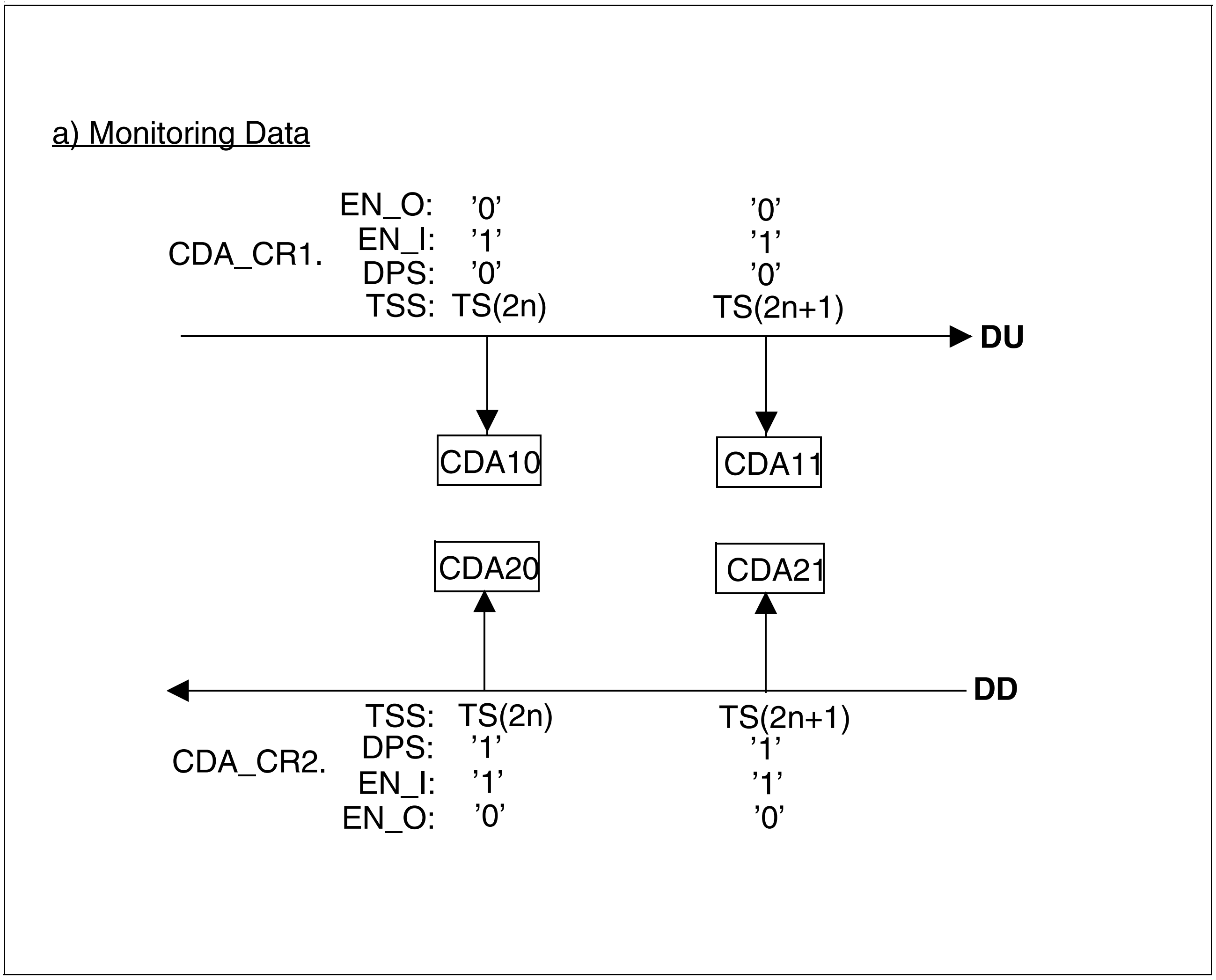


Figure 17-7 Example for Monitoring Data

Monitoring TIC Bus

Monitoring the TIC bus (TS11) is handled as a special case. The TIC bus can be monitored with the registers CDAx0 by setting the EN_TBM (Enable TIC Bus Monitoring) bit in the control registers CRx. The TSDPx0 must be set to 08_h for monitoring from DU or 88_h for monitoring from DD respectively. By this it is possible to monitor the TIC bus (TS11) and the odd numbered D-channel (TS3) simultaneously on DU and DD.

17.1.4 Synchronous Transfer

While looping, shifting and switching the data can be accessed by the controller between the synchronous transfer interrupt (STI) and the synchronous transfer overflow interrupt (STOV).

The CPU access to the CDAxy registers can be synchronized by means of four programmable synchronous transfer interrupts (STIxy) and synchronous transfer overflow interrupts (STOVxy) in the STI register.
Depending on the DPS bit in the corresponding CDA_TSDPxy register the STIxy is generated two (for DPS='0') or one (for DPS='1') BCL clock after the selected time slot (CDA_TSDPxy.TSS). One BCL clock is equivalent to two DCL clocks.

In the following description the index xy_0 and xy_1 are used to refer to two different interrupt pairs (STI/STOV) out of the four CDA interrupt pairs (STI10/STOV10, STI11/STOV11, STI20/STOV20, STI21/STOV21).

An $STOVxy_0$ is related to its $STIxy_0$ and is only generated if $STIxy_0$ is enabled and not acknowledged. However, if $STIxy_0$ is masked, the STOVxy0 is generated for any other STIxy1 which is enabled and not acknowledged.

Table 17-1 gives some examples for that. It is asumed that an STOV interrupt is only generated because an STI interrupt was not acknowledged before.

In example 1 only the $STIxy_0$ is enabled and thus $STIxy_0$ is only generated. If no STI is enabled, no interrupt will be generated even if STOV is enabled (example 2).
In example 3 $STIxy_0$ is enabled and generated and the corresponding $STOVxy_0$ is disabled. $STIxy_1$ is disabled but its $STOVxy_1$ is enabled, and therefore $STOVxy_1$ is generated due to $STIxy_0$. In example 4 additionally the corresponding $STOVxy_0$ is enabled, so $STOVxy_0$ and $STOVxy_1$ are both generated due to $STIxy_0$.
In example 5 additionally the $STIxy_1$ is enabled with the result that $STOVxy_0$ is only generated due to $STIxy_0$ and $STOVxy_1$ is only generated due to $STIxy_1$.

Compared to the previous example $STOVxy_0$ is disabled in example 6, so $STOVxy_0$ is not generated and $STOVxy_1$ is only generated for $STIxy_1$ but not for $STIxy_0$.

Compared to example 5 in example 7 a third $STOVxy_2$ is enabled and thus STOVxy2 is generated additionally for both $STIxy_0$ and $STIxy_1$.

Table 17-1 Examples for Synchronous Transfer Interrupts

Enabled Interrupts (Register MSTI)		Generated Interrupts (Register STI)		
STI	**STOV**	**STI**	**STOV**	
xy_0	-	xy_0	-	Example 1
-	xy_0	-	-	Example 2
xy_0	xy_1	xy_0	xy_1	Example 3
xy_0	xy_0 ; xy_1	xy_0	xy_0 ; xy_1	Example 4
xy_0 ; xy_1	xy_0 ; xy_1	xy_0 xy_1	xy_0 xy_1	Example 5
xy_0 ; xy_1	xy_1	xy_0 xy_1	- xy_1	Example 6
xy_0 ; xy_1	xy_0 ; xy_1 ; xy_2	xy_0 xy_1	xy_0 ; xy_2 xy_1 ; xy_2	Example 7

An STOV interrupt is not generated if all stimulating STI interrupts are acknowledged.

An STIxy must be acknowledged by setting the ACKxy bit in the ASTI register one BCL clock (for DPS='0') or zero BCL clocks (for DPS='1') before the time slot which is selected for the appropriate STIxy. The interrupt must be acknowledged 1 BCL clock cylce earlier than described above due to internal delays.

The interrupt structure of the synchronous transfer is shown in **figure 17-8**.

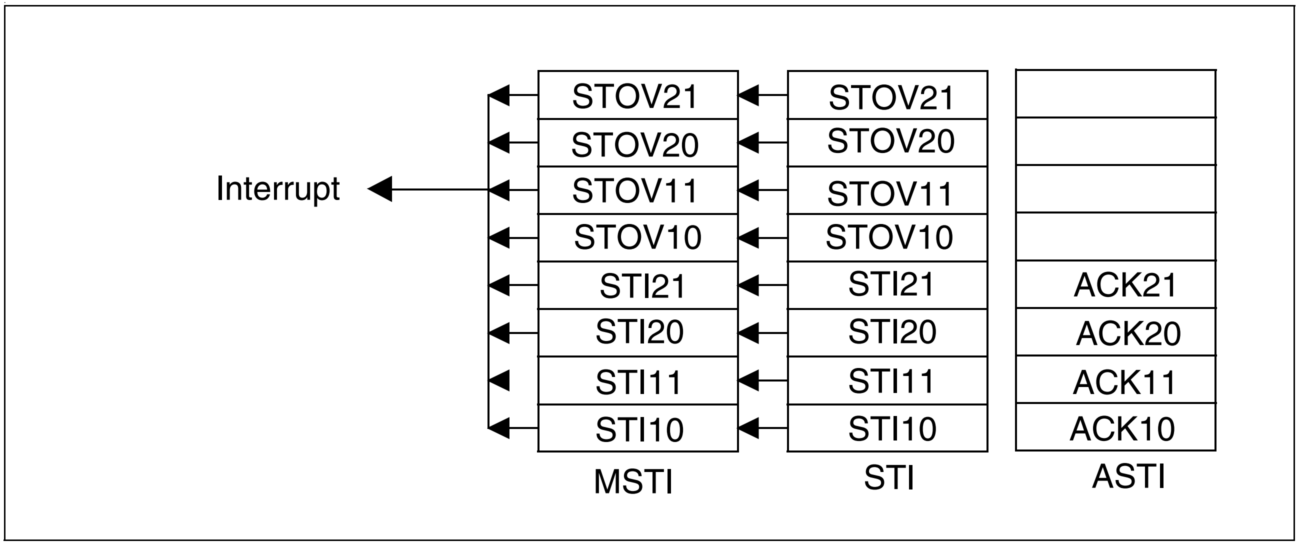


Figure 17-8 Interrupt Structure of the Synchronous Data Transfer

Figure 17-9 shows some examples based on the timeslot structure. In figure a) shows at which point in time an STI and STOV interrrupt is generated for a specific timeslot. Figure b) is identical to example 3 above, figure c) corresponds to example 5 and figure d) shows example 4.

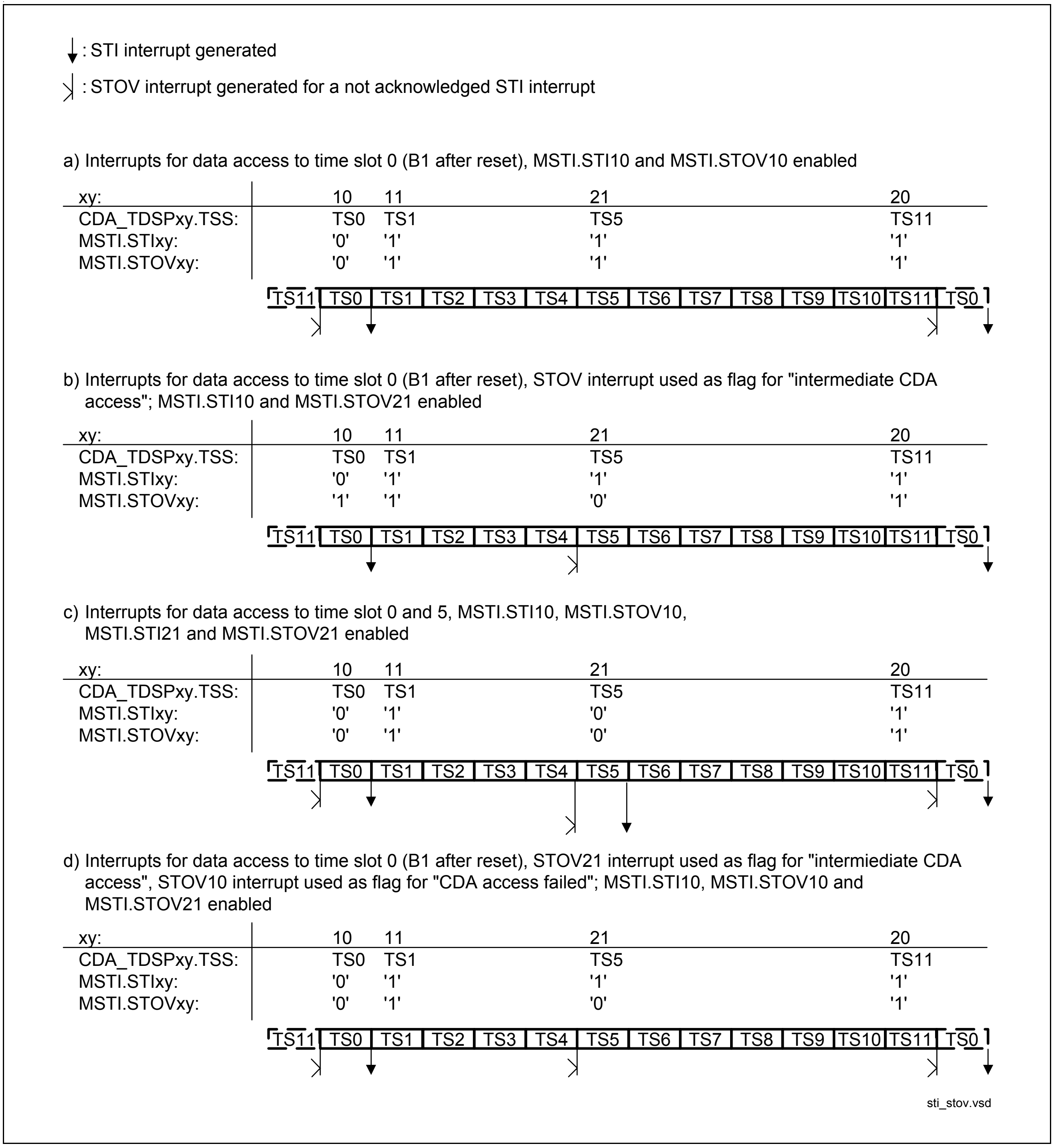

: STI interrupt generated
: STOV interrupt generated for a not acknowledged STI interrupt
a) Interrupts for data access to time slot 0 (B1 after reset), MSTI.STI10 and MSTI.STOV10 enabled
xy: 10 11 21 20
CDA_TDSPxy.TSS: TS0 TS1 TS5 TS11
MSTI.STIxy: '0' '1' '1' '1'
MSTI.STOVxy: '0' '1' '1' '1'
TS11 TS0 TS1 TS2 TS3 TS4 TS5 TS6 TS7 TS8 TS9 TS10 TS11 TS0
b) Interrupts for data access to time slot 0 (B1 after reset), STOV interrupt used as flag for "intermediate CDA access"; MSTI.STI10 and MSTI.STOV21 enabled
xy: 10 11 21 20
CDA_TDSPxy.TSS: TS0 TS1 TS5 TS11
MSTI.STIxy: '0' '1' '1' '1'
MSTI.STOVxy: '1' '1' '0' '1'
TS11 TS0 TS1 TS2 TS3 TS4 TS5 TS6 TS7 TS8 TS9 TS10 TS11 TS0
c) Interrupts for data access to time slot 0 and 5, MSTI.STI10, MSTI.STOV10, MSTI.STI21 and MSTI.STOV21 enabled
xy: 10 11 21 20
CDA_TDSPxy.TSS: TS0 TS1 TS5 TS11
MSTI.STIxy: '0' '1' '0' '1'
MSTI.STOVxy: '0' '1' '0' '1'
TS11 TS0 TS1 TS2 TS3 TS4 TS5 TS6 TS7 TS8 TS9 TS10 TS11 TS0
d) Interrupts for data access to time slot 0 (B1 after reset), STOV21 interrupt used as flag for "intermiediate CDA access", STOV10 interrupt used as flag for "CDA access failed"; MSTI.STI10, MSTI.STOV10 and MSTI.STOV21 enabled
xy: 10 11 21 20
CDA_TDSPxy.TSS: TS0 TS1 TS5 TS11
MSTI.STIxy: '0' '1' '1' '1'
MSTI.STOVxy: '0' '1' '0' '1'
TS11 TS0 TS1 TS2 TS3 TS4 TS5 TS6 TS7 TS8 TS9 TS10 TS11 TS0
sti_stov.vsd

Figure 17-9 Examples for the Synchronous Transfer Interrupt Control with one enabled STIxy

17.1.5 Data transfer from and to the IOM-2 bus

In addition to the CDA registers, a mechanism of data transfer between the IOM-2 bus and specific memory locations has been implemented. This mechanism allows simultaneous access to all 12 time slots of an IOM-2 frame whereas up to 8 bytes (ie. contents of 8 time slots) can be moved..For up- and downstream transfers, mask register define which octets of the IOM-2 frame should be transferred, e.g. B1, B2, IC2 and IC4, or to which octets data should be written respectively.

Thus the transfer direction (read/write from/to the IOM bus) and the IOM port (DU or DD) have to be determined.

The data of selected timeslots is then moved using PEC (Peripheral Event Controller) transfers or by regular interrupt mechanisms offered by the CPU.

The IOM-2 transfer unit consists of 8 mask register to exploit the available 8 PEC channels. The transfers itself can be globally en- or disabled by setting the bits in the transfer control register ITR_CR accordingly.

17.1.5.1 Data Access on IOM-2

The IOM transfer control allows the transfer of data between memory and IOM timeslots using PEC channels. Therefore, an interrupt generation logic is provided to generate up to eight interrupts. These interrupts may be connected to a PEC channel to trigger the data transfer.

A mask register ITR_MSKx is provided for each interrupt. It contains a bit map of the twelve IOM-timeslots plus control bits to indicate the IOM data line DU or DD (ITR_MSKx.PRT) and the access direction (ITR_MSKx.DIR, RD from IOM, WR to IOM).

The software has to ensure that only one interrupt is generated per direction line and accessed timeslot.

Note: There is no logic coordinating simultaneous access from different interrupts at the same timeslots. The result is unpredictable.

If all current mask bits are ´0‘, a reload by a FSC pulse is enabled. As long as at least one bit is set, reload by FSC is blocked.

The transfer interrupt generation is independent from the IOM access. The data of one complete IOM frame is stored in a buffer containing 12 bytes. To allow a response time of close to 125µs, two buffers are available per line and direction. One buffer is used to fill in the data of the current IOM-2 frame, while the other one holds the data that can be accessed by the microcontroller as described in figure **Figure 17-11**.

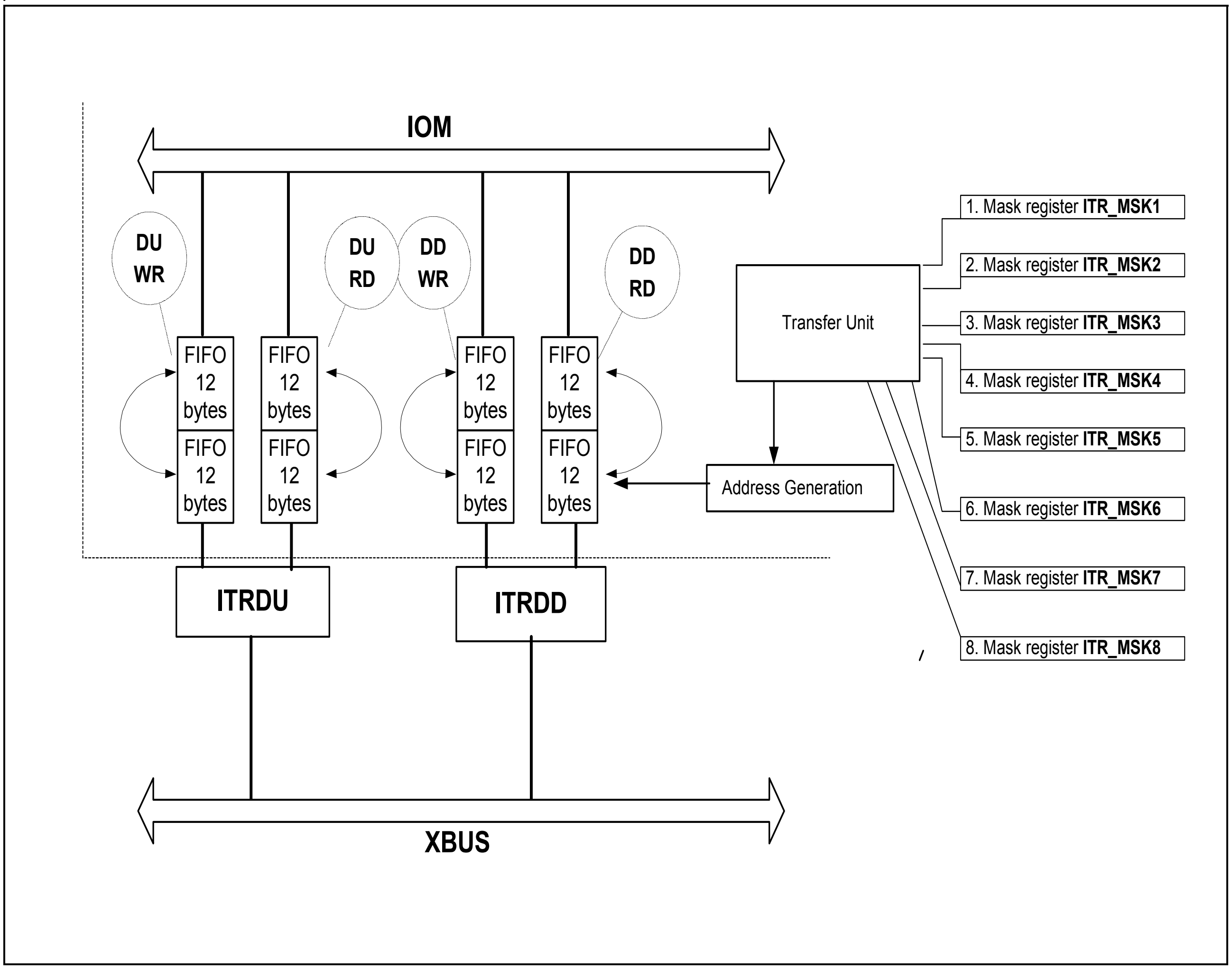

IOM
DU WR
DU RD
DD WR
DD RD
FIFO 12 bytes
Transfer Unit
Address Generation
1. Mask register ITR_MSK1
2. Mask register ITR_MSK2
3. Mask register ITR_MSK3
4. Mask register ITR_MSK4
5. Mask register ITR_MSK5
6. Mask register ITR_MSK6
7. Mask register ITR_MSK7
8. Mask register ITR_MSK8
ITRDU
ITRDD
XBUS

Figure 17-10 Data buffer to store IOM timeslot information

Thus the data of one IOM frame are buffered and remain available for 125µs. The transfer interrupts are generated using the CPU clock signal. To ensure a proper interrupt generation the CPU frequency must be at least 2.5 MHz. Otherwise the behavior is unpredictable.

The generation of a transfer interrupt depends on the actual bit value of the enabled mask register. Therefore the contents of enabled mask registers is copied into temporary shift register whenever a rising edge on FSC is detected and the remaining contents of all temporary shift register is '00_H'. This logic is required to ensure correct memory alignment in case of more than one timeslot being transferred. Otherwise in case of spikes on FSC, the frame structure in memory may be destroyed.

The transfer interrupts are generated for each bit of the temporary shift register. If all bits for one timeslot are zero, then no transfer interrupt occurs. Depending on the number and kind of selected mask bits, the transfer interrupts are generated and the logic will wait until all accesses to the data register have been occurred. Only after all accesses occurred, the mask bits will be shifted with the next clock cycle.

In case the COUNT field of a particular PEC channel control register (PECCx), which is decremented by one after every transfer, reaches ’00_H’ (COUNT defines implicitely the size of the buffer where the transferred data has been stored), the interrupt request is not cleared as it’s done when COUNT is not ’00_H’. Thus the corresponding regular interrupt service routine is called to reinitialize the buffer.

This means also that no PEC transfers with lower interrupt priority will occur if the PEC channel with higher priority reaches COUNT = 0.

The software has to ensure that servicing all used IOM data transfer interrupts (up to eight) has to be done within 100µs.

In case all pending interrupts (i.e. PEC transfers) are not completed before the next frame starts (within 100µs), data access to/from the next frame does not take place and the interrupt request bit ITRFRIR of the control register ITR_CR is set. No data is written to and no data is taken from the next frame.

An example is shown in **Figure 17-12**. ITR_MSK1 is set to receive data from DD at timeslots B1, B2 and IC1 (in figure indicated with "Mask 1"). ITR_MSK2 is set to transmit data to DU at TS B1. ITR_MSK3 is set to transmit data to DU at TS B2 and ITR_MSK4 is set to transmit data to DU at TS IC1. After FSC, the mask bits of all ITR_MSKx registers are checked. Each 1 will generate an interrupt of the corresponding interrupt line IOMTRAxINT.

After the interrupt has been serviced by PEC transfers, the address for the buffer registers are incremented and the mask bits are shifted, so that the bits for the next timeslot are evaluated. Again, if at least one bit was set to 1, the corresponding interrupt is generated and the access to the data register ITRDU or ITRDD has to occur.

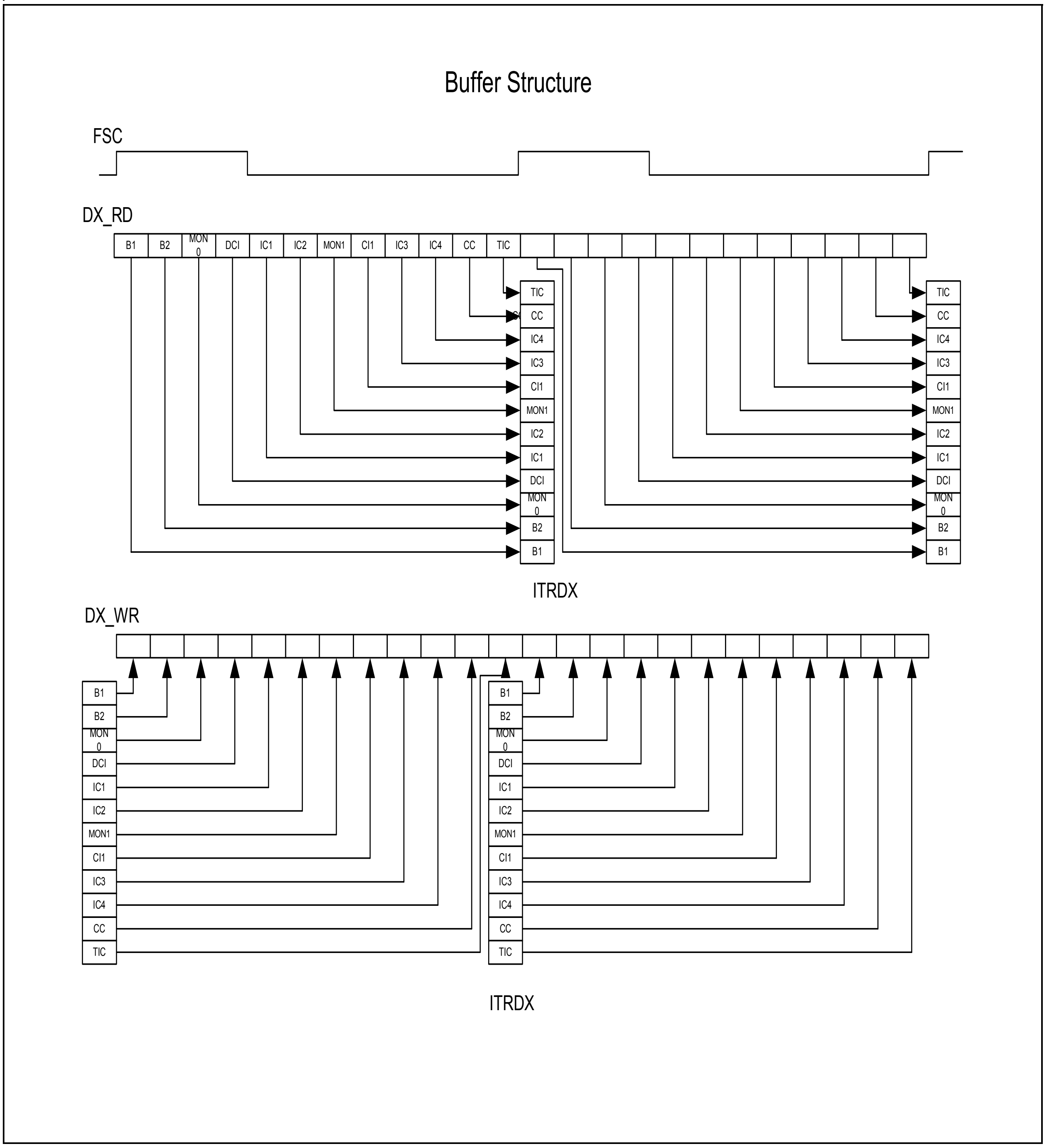


Figure 17-11 Buffer Structure to store the contents of an IOM-2 frame

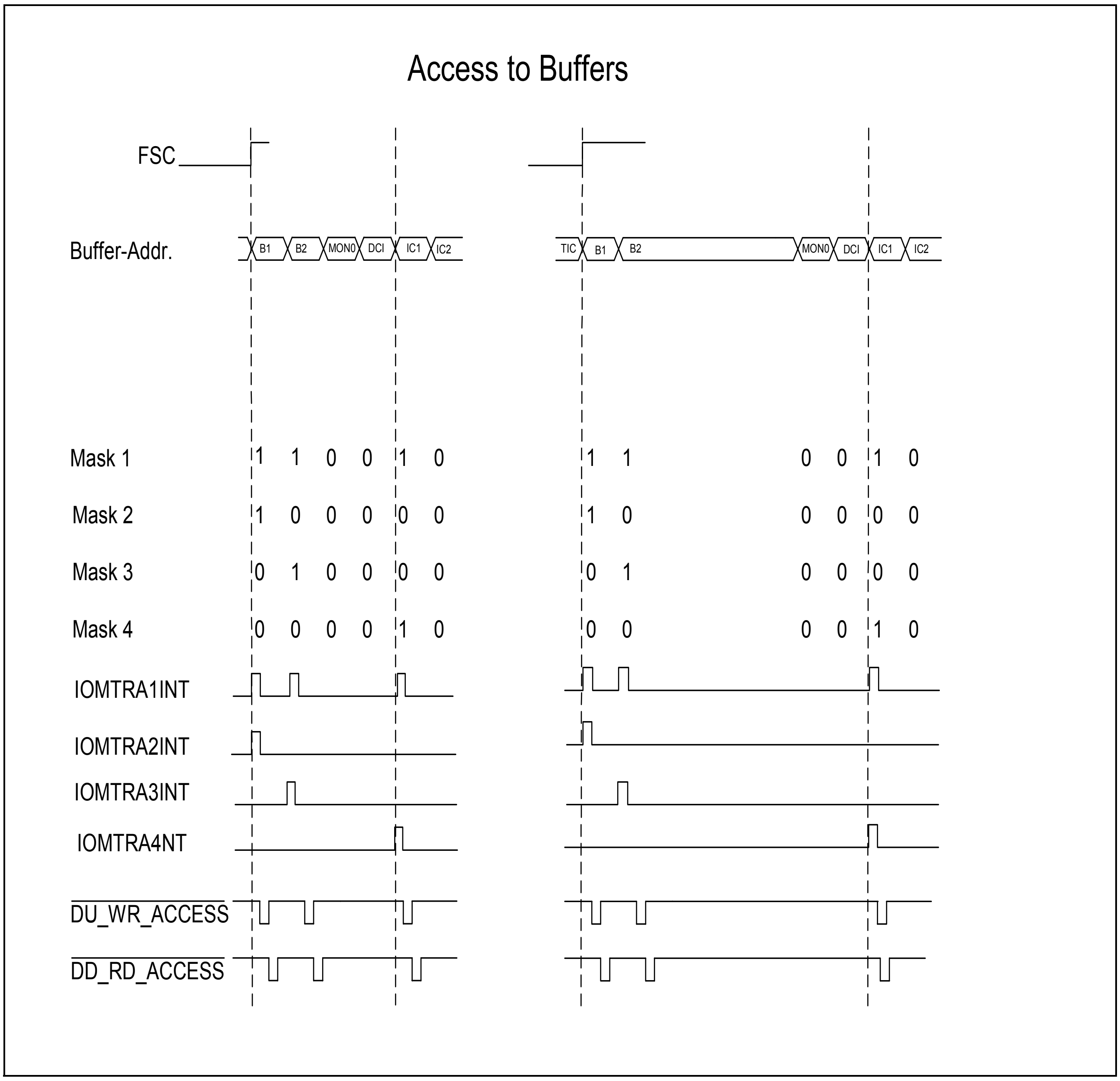


Figure 17-12 Example for the generation of interrupts to trigger PEC transfers

17.1.6 Compare Feature for transfer unit 0

The first of the eight transfer units allows to specify data bytes that won't be transferred to memory. For this purpose the selected byte of one of the buffers is compared to bit field ITRICV. No transfer interrupt will be generated if ITRICV is identical to the value in the selected byte. The comparison with the contents of ITRICV is enabled by setting the bit CED in register ITR_MSK0. This feature allows to define an idle code that does not generate interrupts.

17.1.7 Communication between DSP and IOM-2

To communicate with the DSP a set of four 8 bit timeslot register inside the IOM-2 handler have implicitely to be used, whereas two of them belong to DSP channel 1 and two of them belong to DSP channel 2.

The data from IOM, which should be carried on one of these registers is defined by the 'codec time slot and data port selection registers' CO_TSDPx as described in table 17-2.

Table 17-2 Mapping of DSP channels to 'codec selection register'

Codec Selection Register	DSP channel	16 bit Data Width
CO_TSDP10	1	Low Byte
CO_TSDP11	1	High Byte
CO_TSDP20	2	Low Byte
CO_TSDP21	2	High Byte

If DSP channel 1 issues 8 bit data, the corresponding time slot on IOM will be defined by CO_TSDP10. (DSP channel 2 accordingly)

If the DSP writes 16 bit data to its channel 1, the high byte is controlled by CO_TSDP11 and the low-byte is controlled by CO_TSDP10. (DSP channel 2 accordingly)

Note: Because after reset CO_DSP10 refers to the B1-IOM slot and CO_DSP11 refers to B2, you'll find a 16 bit value without register change as (B1=low-byte / B2=high-byte) on IOM.

Particulary for DSP channel 2 pls. note the following:

Without "Channel Split"

CO_TSDP20 = channel 2, 8 bit

CO_TSDP20 + CO_TSDP21 = channel 2, 16 bit

With "Channel Split"

CO_TSDP20 = channel 3

CO_TSDP21 = channel 2

17.1.8 Serial Data Strobe Signal and strobed Data Clock

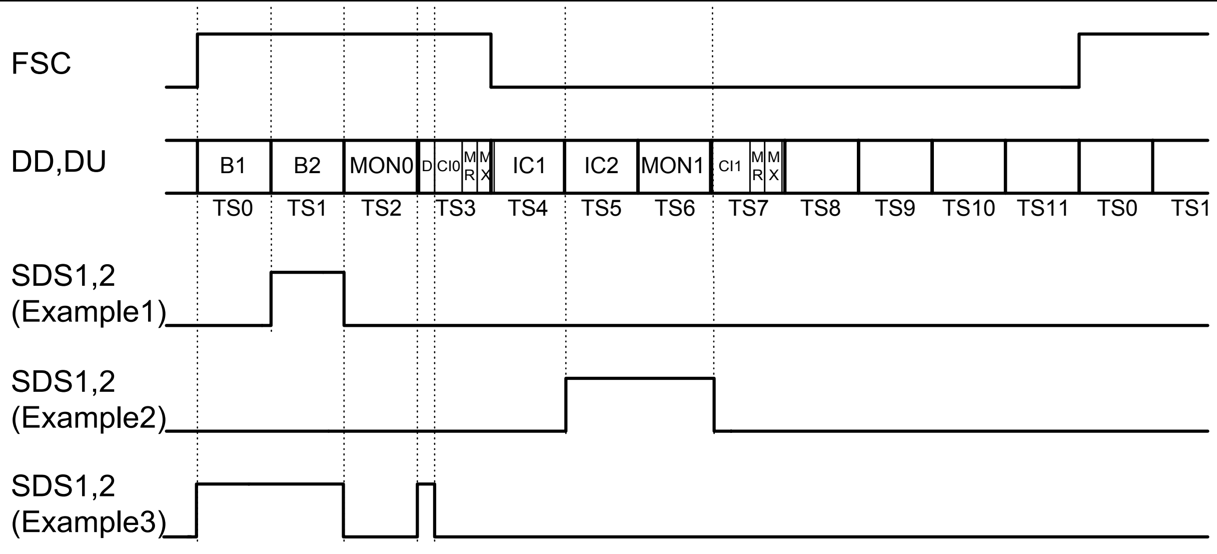
For time slot oriented standard devices connected to the IOM-2 interface the IFA provides two independent data strobe signals SDS1 and SDS2. Instead of a data strobe signal a strobed IOM bit clock can be provided on pin SDS1 and SDS2.

17.1.8.1 Serial Data Strobe Signal

The two strobe signals can be generated with every 8-kHz frame and are controlled by the registers SDS1/2_CR. By programming the TSS bits and three enable bits (ENS_TSS, ENS_TSS+1, ENS_TSS+3) a data strobe can be generated for the IOM-2 time slots TS, TS+1 and TS+3 and any combination of them.

The data strobes for TS and TS+1 are always 8 bits long (bit7 to bit0) whereas the data strobe for TS+3 is always 2 bits long (bit7, bit6).

Figure 17-13 shows three examples for the generation of a strobe signal. In example 1 the SDS is active during channel B2 on IOM-2 whereas in the second example during IC2 and MON1. The third example shows a strobe signal for 2B+D channels which is used e.g. at an IDSL (144kbit/s) transmission.



Example 1:	TSS	= '0_H'
	ENS_TSS	= '0'
	ENS_TSS+1	= '1'
	ENS_TSS+3	= '0'
Example 2:	TSS	= '5_H'
	ENS_TSS	= '1'
	ENS_TSS+1	= '1'
	ENS_TSS+3	= '0'
Example 3:	TSS	= '0_H'
	ENS_TSS	= '1'
	ENS_TSS+1	= '1'
	ENS_TSS+3	= '1'

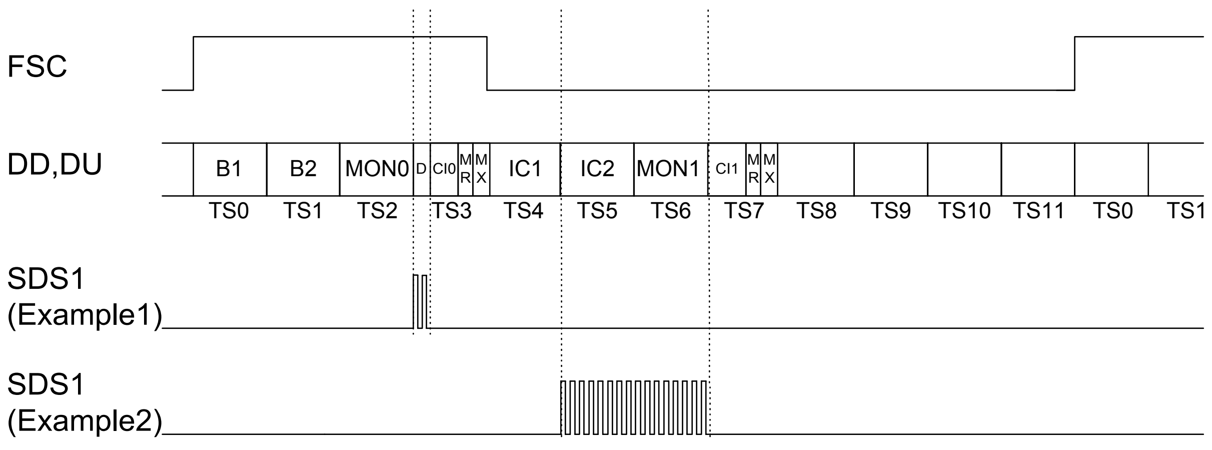
strobe.vsd

For all examples SDS_CONF.SDS1/2_BCL must be set to “0”.

Figure 17-13 Data Strobe Signal

17.1.8.2 Strobed IOM Bit Clock

The strobed IOM bit clock is active during the programmed window. Outside the programmed window a ’0’ is driven. Two examples are shown in **Figure 17-14**.



Setting of SDS1_CR:

Example 1:	TSS	= '0_H'
	ENS_TSS	= '0'
	ENS_TSS+1	= '0'
	ENS_TSS+3	= '1'
Example 2:	TSS	= '5_H'
	ENS_TSS	= '1'
	ENS_TSS+1	= '1'
	ENS_TSS+3	= '0'

bcl_strobed.vsd

For all examples SDS_CONF.SDS1_BCL must be set to “1”.

Figure 17-14 Strobed IOM Bit Clock. Register SDS_CONF programmed to 01_H

The strobed bit clock can be enabled in SDS_CONF.SDS1/2_BCL.

17.1.9 Collision

If data is written via CDA to a timeslot simultaneously with another block (e.g. HDLC controller) then the combined data value (“0” wins) is written on IOM-2 data line.

17.2 TIC/CI Handler

17.2.1 Overview

The Command/Indication channel carries real-time status information for the transceiver on the IOM-2 bus. The C/I handler realizes the C/I channel access on the IBUS from the UBUS and vice versa. It’s function is controlled by register DCI_CR/DCIC_CR at address offset 53_H

17.2.2 C/I channel

C/I Channel 0

One C/I channel (called C/I0) conveys the commands and indications between the layer-1 and the layer-2 parts of the device. C/I0 channel access may be arbitrated via the TIC bus access protocol. In this case the arbitration is done in IOM channel 2.

The C/I0 channel is accessed via register CIR0 (in receive direction) and register CIX0 (in transmit direction). The C/I0 code is four bits long.

In the receive direction, the code from layer-1 is continuously monitored, with an interrupt being generated anytime a change occurs and bit CIR0.CIC0 being set. A new code must be found in two consecutive IOM frames to be considered valid and to trigger a C/I code change interrupt status (double last look criterion).

In the transmit direction, the code written to CIX0 is continuously transmitted in C/I0.

C/I Channel 1

A second C/I channel (called C/I1) can be used to convey real time status information between the device and various non-layer-1 peripheral devices. The C/I1 channel consists of four or six bits in each direction. The width can be changed from 4bit to 6bit by setting bit CIX1.CICW.

In 4-bit mode 6-bits are written whereby the higher 2 bits must be set to “1” and 6-bits are read whereby only the 4 LSBs are used for comparison and interrupt generation (i.e. the higher two bits are ignored).

The C/I1 channel is accessed via registers CIR1 and CIX1. A change in the received C/I1 code is indicated by an interrupt status without double last look criterion.

17.2.3 CIC Interrupt Logic

A CIC interrupt may originate

- from a change in received C/I channel 0 code (CIC0)

or

- from a change in received C/I channel 1 code (CIC 1).

The two corresponding status bits CIC0 and CIC1 are read in CIR0 register. CIC1 can be individually disabled by clearing the enable bit CI1E in the CIX1 register. In this case the occurrence of a code change in CIR1 will not be displayed by CIC1 until the corresponding enable bit has been set to one. Bits CIC0 and CIC1 are cleared by a read of CIR0.

An interrupt status is indicated every time a valid new code is loaded in CIR0 or CIR1. The CIR0 is buffered with a FIFO size of two. If a second code change occurs in the received C/I channel 0 before the first one has been read, immediately after reading of CIR0 a new interrupt will be generated and the new code will be stored in CIR0. If several consecutive codes are detected, only the first and the last code is obtained at the first and second register read, respectively. For CIR1 no FIFO is available. The actual code of the received C/I channel 1 is always stored in CIR1.

Please refer to **Figure 8-3** for information about the further processing of the CIC interrupt.

The direction of the CI0 and CI1 handler are determined by the IOM frame arbiter IFA and programmable with register DCI_CR (address offset 53_H).

17.2.4 TIC Bus Handler - Functional Description

The TIC bus is implemented to organize the access to the layer-1 functions and to the D-channel from up to 7 D-channels HDLC controllers (see **Figure 17-15**). The arbitration mechanism must be activated by setting the MODEH.DIM2-0(2:0) signals to 00x.

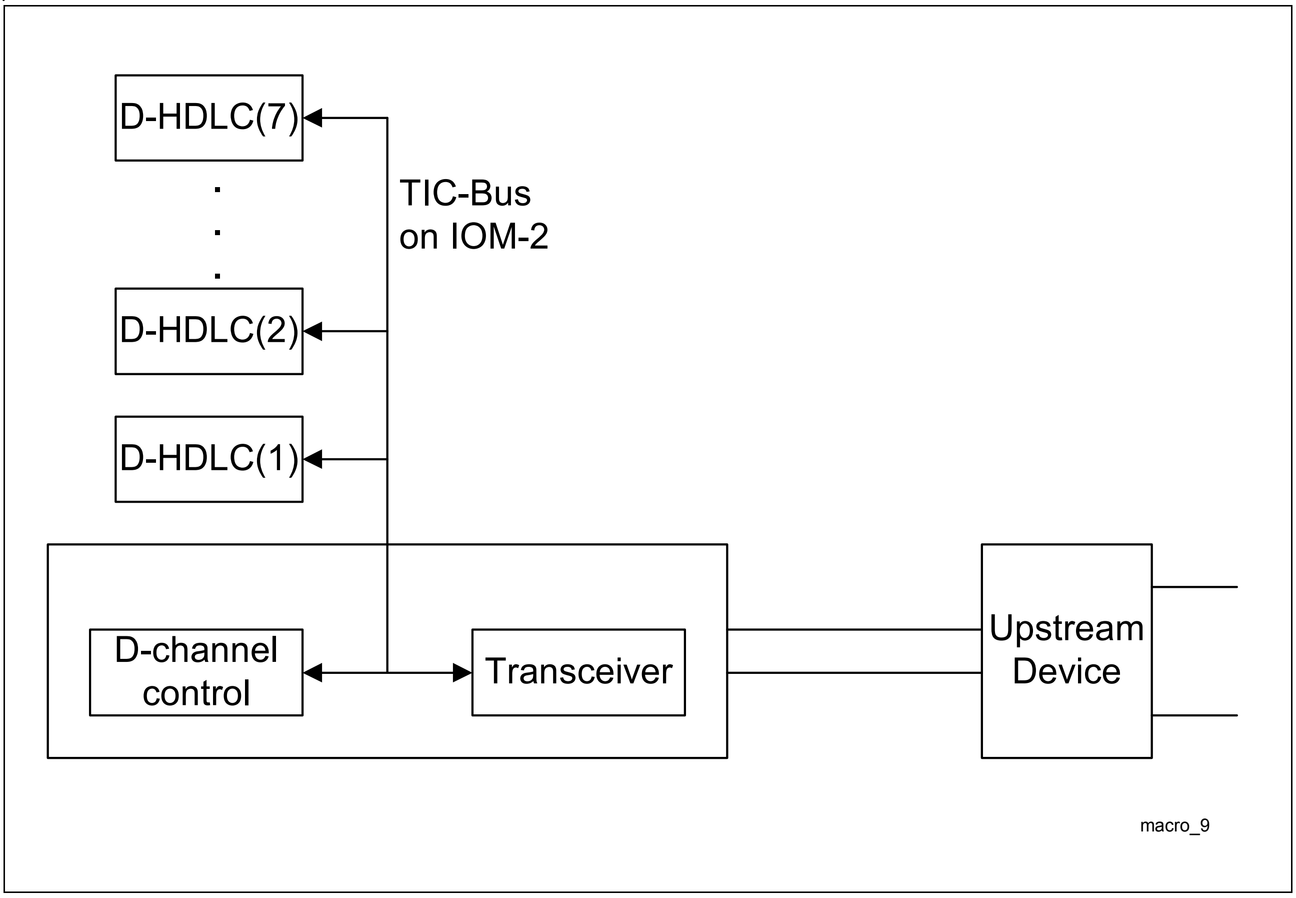


Figure 17-15 Applications of TIC Bus in IOM-2 Bus Configuration

The arbitration mechanism is implemented in the last octet in IOM channel 2 of the IOM-2 interface. An access request to the TIC bus may either be generated by software or by the D-channel HDLC controller (transmission of an HDLC frame in the D-channel). A software access request to the bus is effected by setting the BAC bit (CIX0 register) to ’1’.

In case of an access request, the Bus Accessed-bit BAC (bit 5 of last octet of CH2 on DU) is checked for the status 'bus free', which is indicated by a logical '1'. If the bus is free, the individual TIC bus address TAD programmed in the CIX0 register (CIX0:TBA2-0) is transmitted. While being transmitted the TIC bus address TAD is compared bit by bit with the value read back from DU. If a sent bit set to '1' is read back as '0' because of the access of another D-channel source with a lower TAD, the TIC bus access is immediately withdrawn, i.e. the remaining TAD bits are not transmitted. The TIC bus is occupied by the device which sends and reads back its address error-free. If more than one device attempt to seize the bus simultaneously, the one with the lowest address values wins. This one will set BAC=0 on TIC bus and start D-channel transmission.

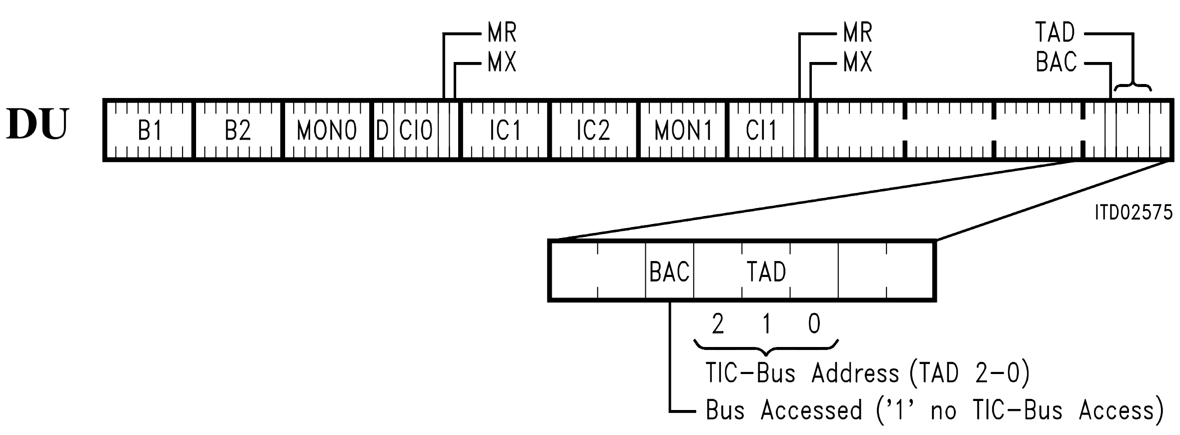

DU
B1
B2
MON0
D
CI0
MR
MX
IC1
IC2
MON1
CI1
MR
MX
TAD
BAC
ITD02575
BAC
TAD
2 1 0
TIC–Bus Address (TAD 2–0)
Bus Accessed ('1' no TIC–Bus Access)

Figure 1 Structure of Last Octet of Ch2 on DU

When the TIC bus is seized, the bus is identified to other devices as occupied via the DU Ch2 Bus Accessed-bit state '0' until the access request is withdrawn. After a successful bus access, the device is automatically set into a lower priority class, that is, a new bus access cannot be performed until the status "bus free" is indicated in two successive frames.

If none of the devices connected to the IOM interface request access to the D and C/I channels, the TIC bus address 7 will be present. The device with this address will therefore have access, by default, to the D and C/I channels.

Note: Bit BAC (CIX0 register) should be reset by the µP when access to the C/I channels is no more requested, to grant other devices access to the D and C/I channels.

17.3 HDLC Controller

17.3.1 Overview

The HDLC controller handles layer-2 functions of the D- channel protocol (LAPD) or B-channel protocols. It can access the D or B-channels or any combination of them e.g. 18 bit IDSL data (2B+D) via the IBUS.

It performs the framing functions used in HDLC based communication: flag generation/recognition, bit stuffing, CRC check and address recognition.

One 64 byte FIFO for the receive and one for the transmit direction are available. They are implemented as cyclic buffers. The transceiver reads and writes data sequentially with constant data rate whereas the data transfer between FIFO and UBUS interface uses a block oriented protocol with variable block sizes.

17.3.2 Message Transfer Modes

The HDLC controller can be programmed to operate in various modes, which are different in the treatment of the HDLC frame in receive direction. Thus the receive data flow and the address recognition features can be programmed in a flexible way to satisfy different system requirements.

The structure of a LAPD two-byte address is shown below.

High Address Byte			Low Address Byte	
SAPI1, 2, SAPG	C/R	0	TEI 1, 2, TEIG	EA

For the address recognition the HDLC controller contains four programmable registers for individual SAPI and TEI values (SAP1, 2 and TEI1, 2), plus two fixed values for the “group” SAPI (SAPG = ’FE’ or ’FC’) and TEI (TEIG = ’FF’).

The received C/R bit is excluded from the address comparison. EA is the address field extension bit which is set to ’1’ for LAPD protocol. There are 5 different operating modes which can be selected via the mode selection bits MDS2-0 in the MODEH register.

Non-Auto Mode (MDS2-0 = ’01x’)

Characteristics: Full address recognition with one-byte (MDS = ’010’) or two-byte (MDS = ’011’) address comparison

All frames with valid addresses are accepted and the bytes following the address are transferred to the UBUS interface via RFIFO.

Transparent Mode 0 (MDS2-0 = ’110’).

Characteristics: no address recognition

Every received frame is stored in RFIFO (first byte after opening flag to CRC field). Additional information can be read from RSTA.

Transparent Mode 1 (MDS2-0 = '111').

Characteristics: SAPI recognition

A comparison is performed on the first byte after the opening flag with SAP1, SAP2 and "group" SAPI (FE_H/FC_H). In the case of a match, all following bytes are stored in RFIFO. Additional information can be read from RSTA.

Transparent Mode 2 (MDS2-0 = '101').

Characteristics: TEI recognition

A comparison is performed only on the second byte after the opening flag, with TEI1, TEI2 and group TEI (FF_H). In case of a match the rest of the frame is stored in the RFIFO. Additional information can be read from RSTA.

Extended Transparent Mode (MDS2-0 = '100' and SRA = '0').

Characteristics: fully transparent

In extended transparent mode fully transparent data transmission/reception without HDLC framing is performed i.e. without FLAG generation/recognition, CRC generation/ check, bitstuffing mechanism. This allows user specific protocol variations.

The transmitter sends the data out of the FIFO without manipulation. Transmission is always IOM-frame aligned and byte aligned, i.e. transmission starts in the first selected channel (e.g. B1, B2, D) of the next IOM-frame.

The FIFO indications and commands are the same as in other modes.

If the microcontroller sets XTF & XME the transmitter responds with an XPR interrupt after sending the last byte, then it returns to its idle state (sending continuous '1').

If the cts (clear to send) signal is set to "0" during transmission the transparent HDLC controller responds always with an XMR (transmit message repeat) interrupt and stops transmission.

If the microcontroller fails to respond to an XPR interrupt in time and the transmitter runs out of data then it will assert an XDU (transmit data underrun) interrupt.

The reception is IOM-frame aligned and byte aligned, like transmission, i.e. reception starts in the first selected channel (B1, B2, D on IOM-2) of the next IOM frame. The FIFO indications and commands are the same as in other modes.

All incoming data bytes are stored in the RFIFO. If the FIFO is full an RFO interrupt is asserted (EXMR.SRA = '0').

Note: In the extended transparent mode the EXMR register has to be set to 'xxx00000'

Enhanced Extended Transparent Mode (MDS2-0 = ’100’ and SRA = ’1’).

If exmr.sra=1 then the receive data flow is controlled by start and stop characters. The value of SAP1 is interpreted as start and TEI1 as stop (similar to the HDLC address recognition). Each character bit which is used for comparison can be masked by a binary “1” (i.e. it is not used for comparison) in SAP2 and TEI2, respectively.

The start character recognition works non-aligned, i.e. reception starts as soon as a start character is detected anywhere in the serial data stream. After this start pattern has been recognized the following data bits are byte aligned. As opposed to the start character, the stop character is byte aligned.

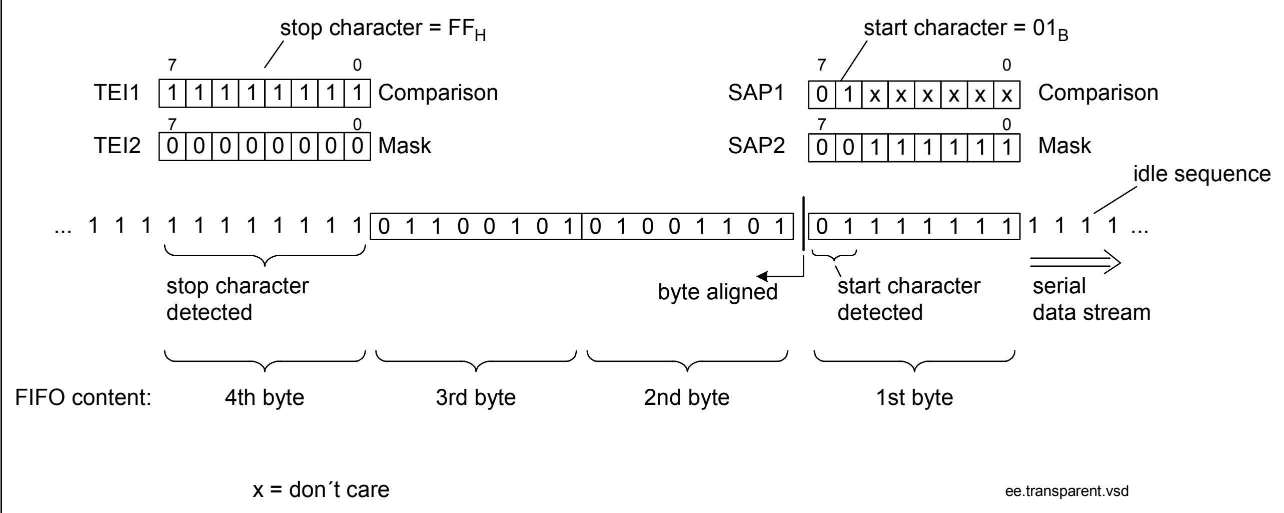


Figure 17-16 Example

Note: It should be noted that the pattern of the stop character must not be contained in the serial data stream.

Received bytes, including start and stop characters are stored in the receive fifo. In case of several consecutive stop characters only the first is stored in the fifo. It is marked as message end, so the uP gets an RME interrupt when the block containing the last character is ready for readout in the fifo.

17.3.3 Data Reception

17.3.3.1 General Description

The 64-byte cyclic RFIFO buffer has variable FIFO block sizes (thresholds) of 4, 8, 16 or 32 bytes, which can be selected by setting the corresponding RFBS bits in the EXMR register. The variable block size allows an optimized HDLC processing concerning frame length, I/O throughput and interrupt load.
The transfer protocol between HDLC FIFO and CPU is block orientated with the CPU as master. The control of the data transfer between the CPU and the HDLC controller is handled via interrupts (HDLC controller → CPU) and commands (CPU → HDLC controller).

There are three different interrupt indications in the ISTAH register concerned with the reception of data:

RPF (**R**eceive **P**ool **F**ull) interrupt, indicating that a data block of the selected length (EXMR.RFBS) can be read from RFIFO. The message which is currently received exceeds the block size so further blocks will be received to complete the message.

RME (**R**eceive **M**essage **E**nd) interrupt, indicating that the reception of one message is completed, i.e. either

a short message is received
(message length ≤ the defined block size (EXMR.RFBS)) or

the last part of a long message is received
(message length > the defined block size (EXMR.RFBS))
and is stored in the RFIFO.

RFO (**R**eceive **F**rame **O**verflow) interrupt, indicating that a complete frame could not be stored in RFIFO and is therefore lost as the RFIFO is occupied. This occurs if the host fails to respond quickly enough to RPF/RME interrupts since previous data was not read by the host.

There are two control commands that are used with the reception of data:

RMC (**R**eceive **M**essage **C**omplete) command, telling the HDLC controller that a data block has been read from the RFIFO and the corresponding FIFO space can be released for new receive data.

RRES (**R**eceiver **R**eset) command, resetting the HDLC receiver and clearing the receive FIFO of any data (e.g. used before start of reception). It has to be used after a change of the message transfer mode. RRES does not clear pending interrupt indications of the receiver, but have to be be cleared by reading these interrutps.

The significant interrupts and commands are underlined as only these are usually used during a normal reception sequence.

The following description of the receive FIFO operation is illustrated in **figure 17-17** for a RFIFO block size (threshold) of 16 and 32 bytes .

The RFIFO requests service from the microcontroller by setting a bit in the ISTAH register, which causes an interrupt (RPF, RME, RFO). The microcontroller then reads status information (RBCH,RBCL), data from the RFIFO and may change the RFIFO block size (EXMR.RFBS). A block transfer is completed by the microcontroller via a receive message complete (CMDR.RMC) command. This causes the space of the transferred bytes being released for new data and in case the frame was complete (RME) the reset of the receive byte counter RBC (RBCH,RBCL).
The total length of the frame is contained in the RBCH and RBCL registers which contain a 12 bit number (RBC11...0), so frames up to 4095 byte length can be counted. If a frame is longer than 4095 bytes, the RBCH.OV (overflow) bit will be set. The least significant bits of RBCL contain the number of valid bytes in the last data block indicated by RME (length of last data block ≤ selected block size). **Table 17-3** gives an example for a FIFO size of 64 byte and shows which RBC bits contain the number of bytes in the last data block or number of complete data blocks, respectively. If the number of bytes in the last data block is ’0’ the length of the last received block is equal to the block size.

Table 17-3 Receive Byte Count with RBC11...0 in the RBCH and RBCL registers

EXMR.RFBS bits	Selected block size	Number of	
		complete data blocks in	bytes in the last data block in
’00’	32 byte	RBC11...5	RBC4...0
’01’	16 byte	RBC11...4	RBC3...0
’10’	8 byte	RBC11...3	RBC2...0
’11’	4 byte	RBC11...2	RBC1...0

In this example the transfer block size (EXMR.RFBS) is 32 bytes. If it is necessary to react to an incoming frame within the first few bytes the microcontroller can set the RFIFO block size to a smaller value. Each time a CMDR.RMC or CMDR.RRES command is issued, the RFIFO access controller sets its block size to the value specified in EXMR.RFBS, so the microcontroller has to write the new value for RFBS before the RMC command. When setting an initial value for RFBS before the first HDLC activities, a RRES command must be issued afterwards.

The RFIFO can hold any number of frames fitting in the 64 bytes. At the end of a frame the RSTA byte is always appended.

All generated interrupts are inserted together with all additional information into a wait line to be individually passed to the host. For example if several data blocks have been received to be read by the host and the host acknowledges the current block, a new RPF or RME interrupt from the wait line is immediately generated to indicate new data.

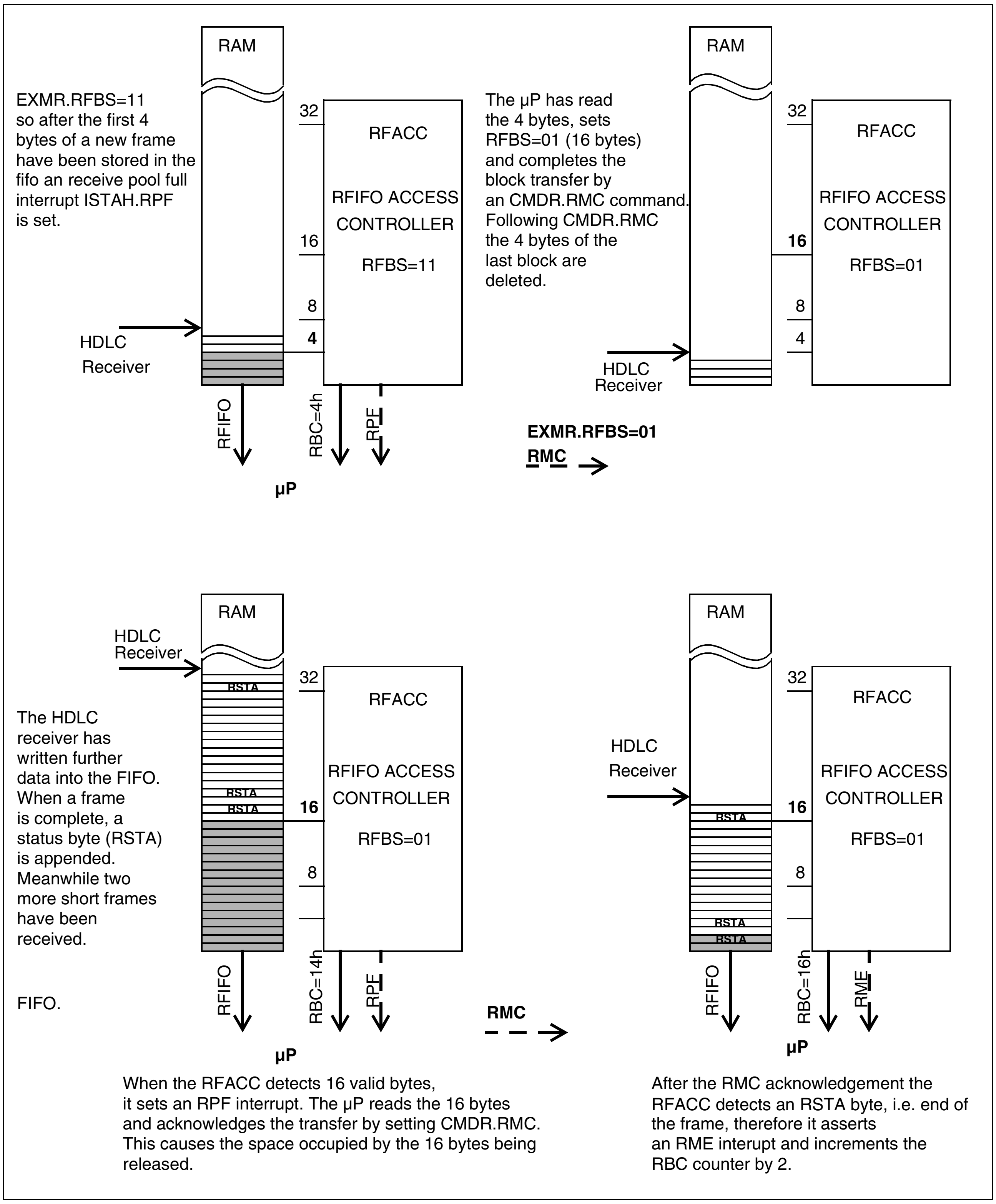


Figure 17-17 RFIFO Operation

17.3.3.2 Possible Error Conditions during Reception of Frames

If parts of a frame get lost because the receive FIFO is full, the Receive Data Overflow (RDO) bit in the RSTA byte will be set. If a complete frame is lost, i.e. if the FIFO is full when a new frame is received, the receiver will assert a Receive Frame Overflow (RFO) interrupt.

The microcontroller sees a cyclic buffer, i.e. if it tries to read more data than available, it reads the same data block again and again. On the other hand, if it doesn’t read all data, they are deleted after the RMC command.

If the microcontroller reads data without a prior RME or RPF interrupt, the content of the RFIFO would not be corrupted, but new data is only transferred to the host as long as new valid data is available in the RFIFO, otherwise the last data is read again and again.

17.3.3.3 Data Reception Procedure

The general procedures for a data reception sequence are outlined in the flow diagram in **figure 17-18**.

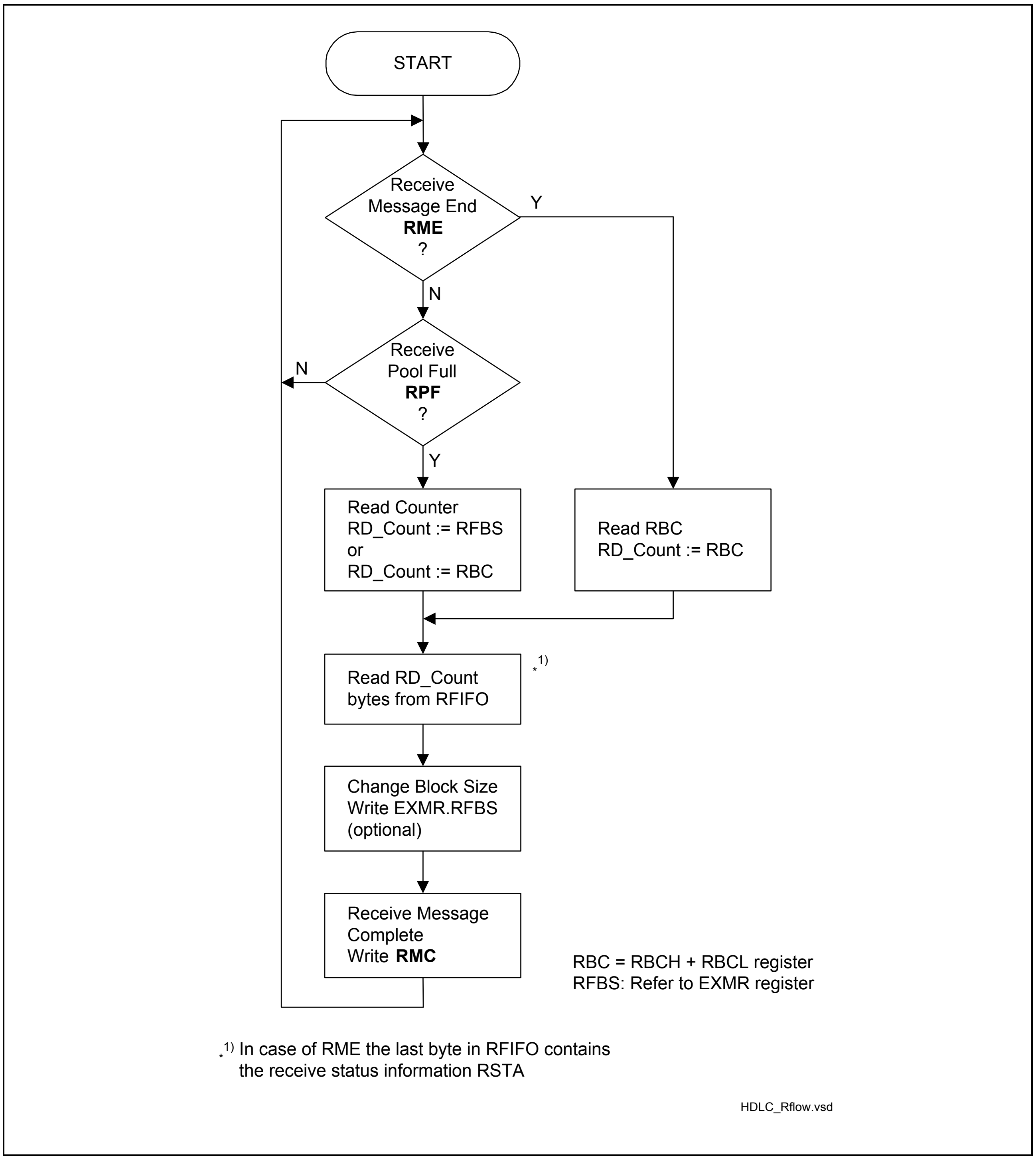

START
Receive Message End RME ?
Y
N
Receive Pool Full RPF ?
N
Y
Read Counter RD_Count := RFBS or RD_Count := RBC
Read RBC RD_Count := RBC
Read RD_Count bytes from RFIFO
*1)
Change Block Size Write EXMR.RFBS (optional)
Receive Message Complete Write RMC
RBC = RBCH + RBCL register
RFBS: Refer to EXMR register
*1) In case of RME the last byte in RFIFO contains the receive status information RSTA
HDLC_Rflow.vsd

Figure 17-18 Data Reception Procedures

Figure 17-19 gives an example of an interrupt controlled reception sequence, supposed that a long frame (68 byte) followed by two short frames (12 byte each) is received. The FIFO threshold (block size) is set to 32 byte in this example:

After 32 bytes of frame 1 have been received an RPF interrupt is generated to indicate that a data block can be read from the RFIFO.

The host reads the first data block from RFIFO and acknowledges the reception by RMC. Meanwhile the second data block is received and stored in RFIFO.

The second 32 byte block is indicated by RPF which is read and acknowledged by the host as described before.

The reception of the remaining 4 bytes plus RSTA are indicated by RME (i.e. the receive status in RSTA register is always appended to the end of a frame).

The host gets the number of received bytes (COUNT = 5) from RBCL/RBCH and reads out the RFIFO and optionally the status register RSTA. The frame is acknowledged by RMC.

The second frame is received and indicated by RME interrupt.

The host gets the number of bytes (COUNT = 13) from RBCL/RBCH and reads out the RFIFO and optionally status register. The RFIFO is acknowledged by RMC.

The third frame is transferred in the same way.

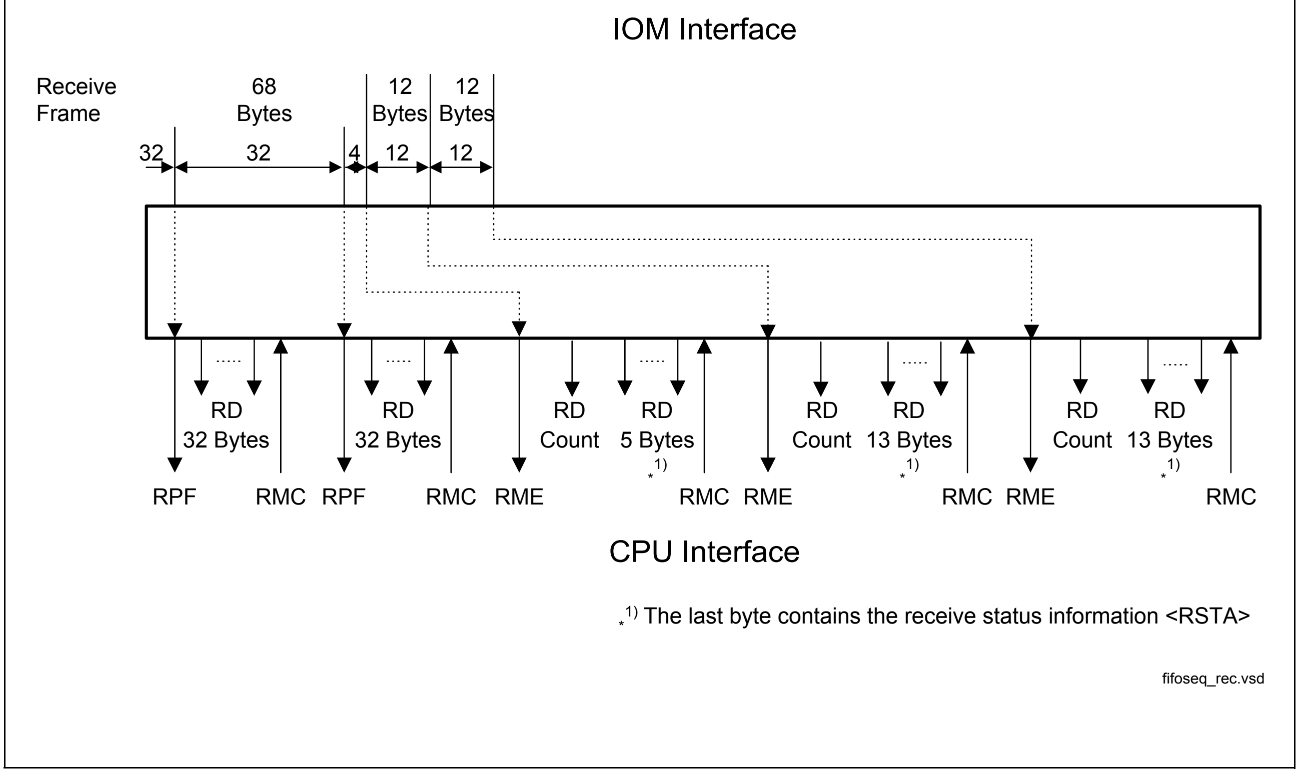
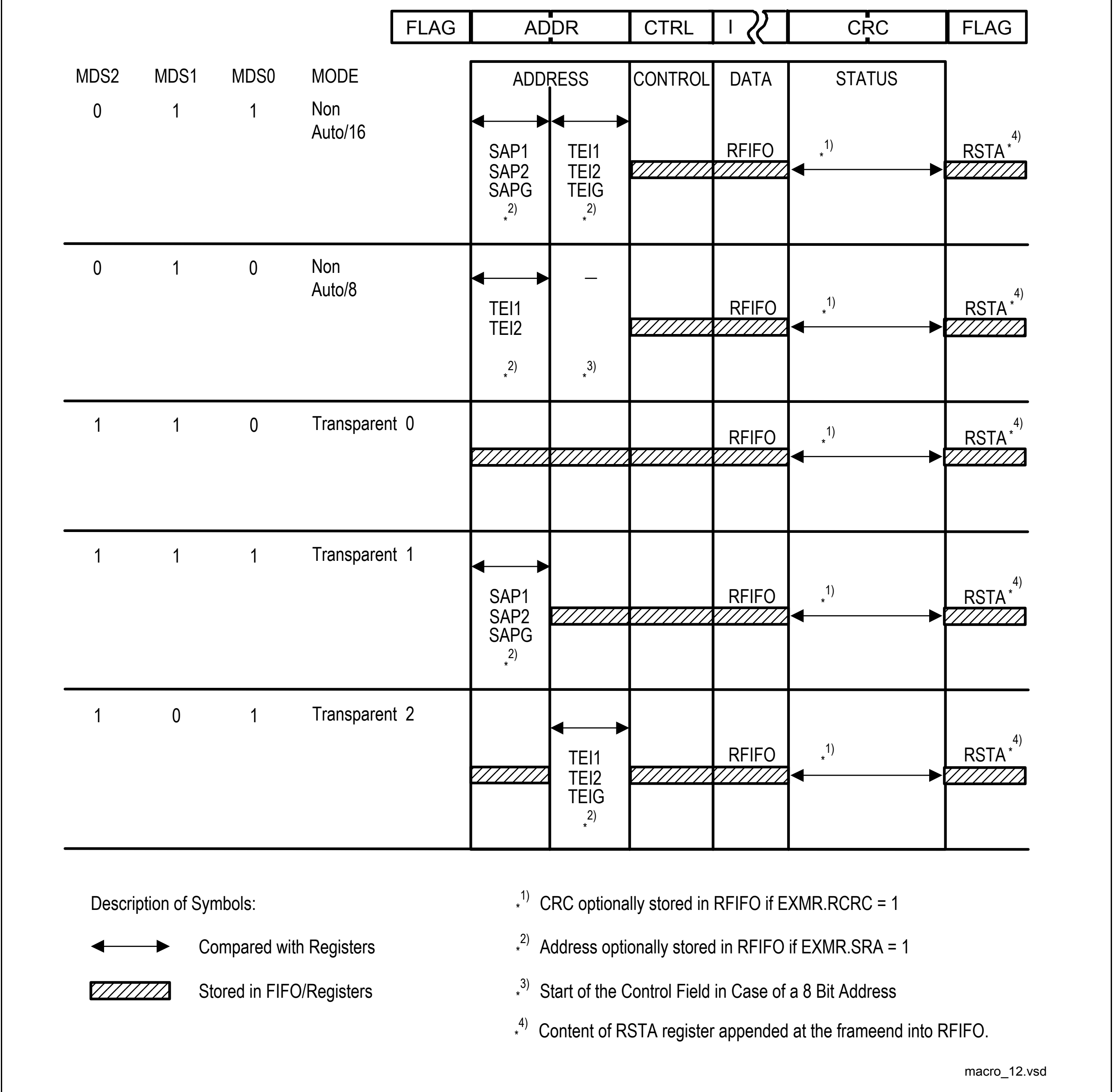


Figure 17-19 Reception Sequence Example

17.3.4 Receive Frame Structure

The management of the received HDLC frames as affected by the different operating modes is shown in **Figure 17-20**.

Figure 17-20 Receive Data Flow



Note: The figure shows all modes except the extended transparent mode as this mode uses no typical frame structure or address recognition. Data is transferred purely transparent.

17.3.5 Data Transmission

17.3.5.1 General Description

The 64-byte cyclic XFIFO buffer has variable FIFO block sizes (thresholds) of 16 or 32 bytes, selectable by the XFBS bit in the EXMR register.

There are three different interrupt indications in the ISTAH register concerned with the transmission of data:

XPR (**T**ransmit **P**ool **R**eady) interrupt, indicating that a data block of up to 16 or 32 byte (block size selected via EXMR:XFBS) can be written to the XFIFO.
An XPR interrupt is generated either

after an XRES (Transmitter Reset) command (which is issued for example for frame abort) or

when a data block from the XFIFO is transmitted and the corresponding FIFO space is released to accept further data from the host.

XDU (**T**ransmit **D**ata **U**nderrun) interrupt, indicating that the transmission of the current frame has been aborted (seven consecutive ’1’s are transmitted) as the XFIFO holds no further transmit data. This occurs if the host fails to respond to an XPR interrupt quickly enough.

XMR (**T**ransmit **M**essage **R**epeat) interrupt, indicating that the transmission of the complete last frame has to be repeated as the cts signal was set to “0” (stop) while the first data bytes have already been overwritten with new data. So the XFIFO does not hold the first data bytes of the frame. (The HDLC transmitter is stopped if cts=0).

The occurence of an XDU or XMR interrupt clears the XFIFO and an XPR interrupt is issued together with an XDU or XMR interrupt, respectively. Data cannot be written to the XFIFO as long as an XDU/XMR interrupt is pending.

Three different control commands are used for transmission of data:

XTF (**T**ransmit **T**ransparent **F**rame) command, telling the HDLC controller that up to 16 or 32 byte (according to selected block size) have been written to the XFIFO and should be transmitted. A start flag is generated automatically.

XME (**T**ransmit **M**essage **E**nd) command, telling the HDLC controller that the last data block written to the XFIFO completes the corresponding frame and should be transmitted. This implies that according to the selected mode a frame end (CRC + closing flag) is generated and appended to the frame.

XRES (**T**ransmitter **R**eset) command, resetting the HDLC transmitter and clearing the transmit FIFO of any data. After an XRES command the transmitter always sends an abort sequence, i.e. this command can be used to abort a transmission. XRES does not clear pending interrupt indications of the transmitter, but have to be be cleared by reading these interrutps.

Optionally two additional status conditions can be read by the host:

XDOV (**T**ransmit **D**ata **O**verflow), indicating that the data block size has been exceeded, i.e. more than 16 or 32 byte were entered and data was overwritten.

XFW (**T**ransmit **F**IFO **W**rite Enable), indicating that data can be written to the XFIFO. This status flag may be polled instead of or in addition to XPR.

The significant interrupts and commands are underlined as only these are usually used during a normal transmission sequence.

The XFIFO requests service from the microcontroller by setting a bit in the ISTAH register, which causes an interrupt (XPR, XDU, XMR). The microcontroller can then read the status register STAR (XFW, XDOV), write data in the FIFO and it may optionally change the transmit FIFO block size (EXMR.XFBS) if required.
The instant of the initiation of a transmit pool ready (XPR) interrupt after different transmit control commands is listed in **table 17-4**.

Table 17-4 XPR Interrupt (availability of the XFIFO) after XTF, XME Commands

CMDR.	Transmit pool ready (XPR) interrupt initiated...
XTF	as soon as the selected buffer size in the FIFO is available.
XTF & XME	after the successful transmission of the closing flag. The transmitter sends always an abort sequence.
XME	as soon as the selected buffer size in the FIFO is available, two consecutive frames share flags (endflag = startflag of next frame).

When setting XME the transmitter appends the CRC and the endflag at the end of the frame. When XTF & XME has been set, the XFIFO is locked until successful transmission of the current frame, so a consecutive XPR interrupt also indicates successful transmission of the frame whereas after XME or XTF the XPR interrupt is asserted as soon as there is space for one data block in the XFIFO.

The transfer block size is 32 bytes by default, but sometimes, if the microcontroller has a high computational load, it is useful to increase the maximum reaction time for an XPR interrupt. The maximum reaction time is:
t_{max} = (XFIFO size - XFBS) / data transmission rate
As an example with a 64 byte FIFO, a selected block size of 16 bytes means that an XPR interrupt is indicated when there are still 48 bytes (64 bytes - 16 bytes) to be transmitted. With a 32 bytes block size the XPR is initiated when there are still 32 bytes (64 bytes - 32 bytes), i.e. the maximum reaction time for the smaller block size is 50 % higher with

the trade-off of a doubled interrupt load. A selected block size of 32 or 16 bytes respectively always indicates the available space in the XFIFO. So any number of bytes smaller than the selected XFBS may be stored in the FIFO during one “write block“ access cycle.

Similar to RFBS for the receive FIFO, a new setting of XFBS takes effect after the next XTF,XME or XRES command. XRES resets the XFIFO.

The XFIFO can hold any number of frames fitting in the 64 (M) bytes.

17.3.5.2 Possible Error Conditions during Transmission of Frames

If the transmitter sees an empty FIFO, i.e. if the microcontroller does not react quickly enough to an XPR interrupt, an XDU (transmit data underrun) interrupt will be generated. If the HDLC channel becomes unavailable during transmission the transmitter tries to repeat the current frame as specified in the LAPD protocol. This is impossible after the first data block has been sent (16 or 32 bytes), in this case an XMR transmit message repeat interrupt is set and the microcontroller has to send the whole frame again.

Both XMR and XDU interrupts cause a reset of the XFIFO. The XFIFO is locked while an XMR or XDU interrupt is pending, i.e. all write actions of the microcontroller will be ignored as long as the microcontroller has not read the ISTAH register with the set XDU, XMR interrupts.

If the microcontroller writes more data than allowed (16 or 32 bytes) , then the data in the XFIFO will be corrupted and the STAR.XDOV bit is set. If this happens, the microcontroller has to abort the transmission by CMDR.XRES and start new.

17.3.5.3 Data Transmission Procedure

The general procedures for a data transmission sequence are outlined in the flow diagram in **figure 17-21**.

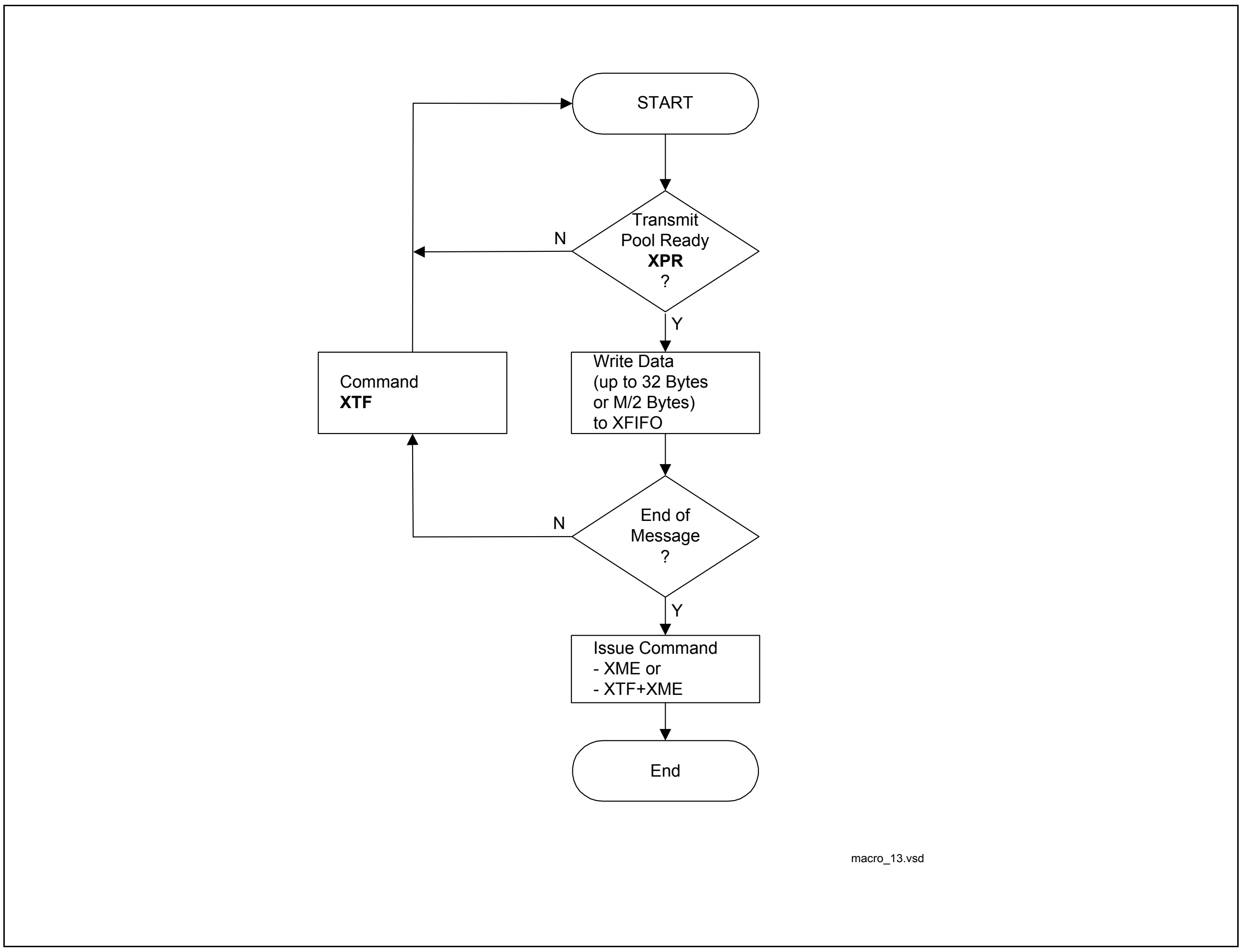


Figure 17-21 Data Transmission Procedure

The following description gives an example with FIFO size of 64 byte for the transmission of a 76 byte frame with a selected block size of 32 byte:

The host writes 32 bytes to the XFIFO, issues an XTF command and waits for an XPR interrupt in order to continue with entering data.

The HDLC controller immediately issues an XPR interrupt (as remaining XFIFO space is not used) and starts transmission.

Due to the XPR interrupt the host writes the next 32 bytes to the XFIFO, followed by the XTF command, and waits for XPR.

As soon as the last byte of the first block is transmitted, the HDLC controller issues an XPR interrupt (XFIFO space of first data block is free again) and continues transmitting the second block.

The host writes the remaining 12 bytes of the frame to the XFIFO and issues the XTF command together with XME to indicate that this is the end of frame.

After the last byte of the frame has been transmitted the HDLC controller releases an XPR interrupt and the host may proceed with transmission of a new frame.

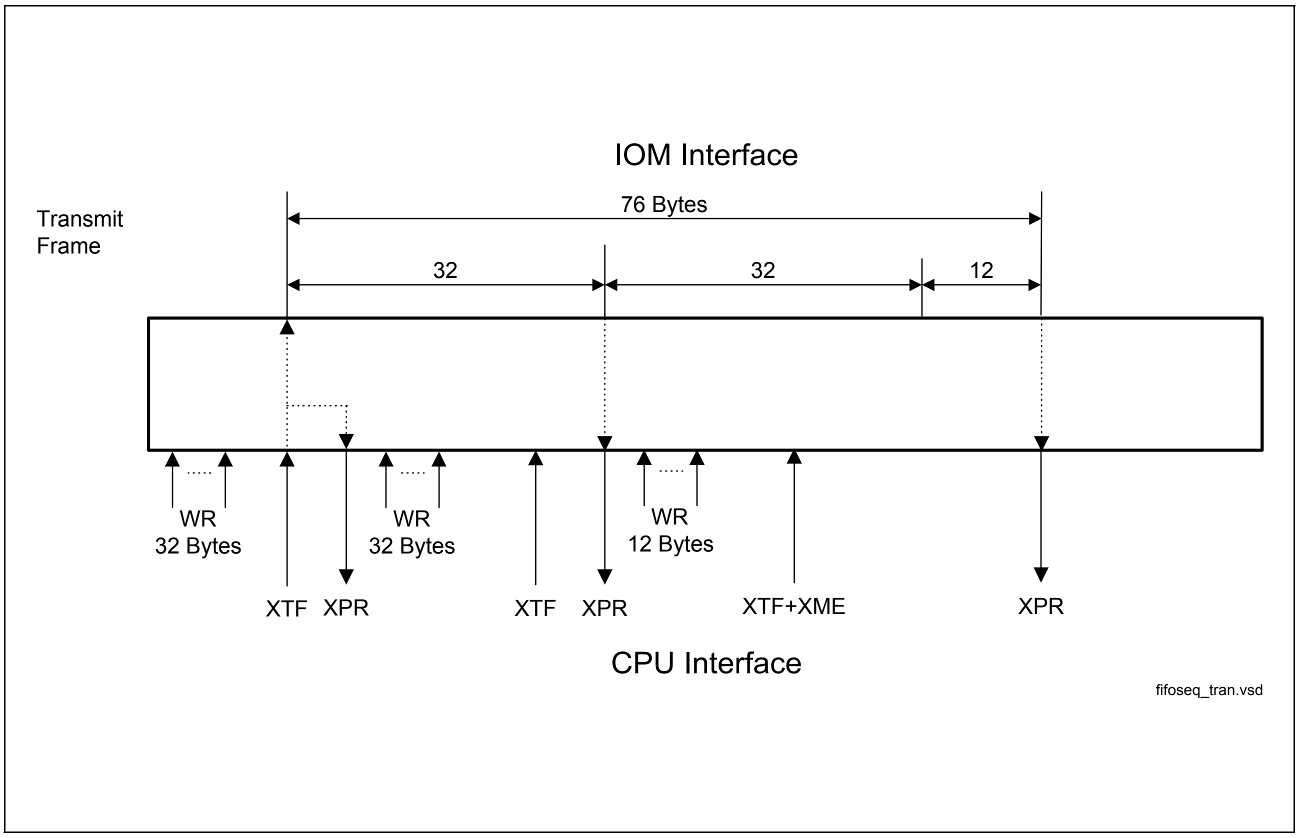


Figure 17-22 Transmission Sequence, Example

17.3.5.4 Transmit Frame Structure

The transmission of transparent frames (XTF command) is shown in **figure 17-23**

For transparent frames, the whole frame including address and control field must be written to the XFIFO. The host configures whether the CRC is generated and appended to the frame (default) or not (selected in EXMR.XCRC).

Furthermore, the host selects the interframe time fill signal which is transmitted between HDLC frames (EXMR:ITF). One option is to send continuous flags ('01111110') or an idle sequence (continuous '1's are transmitted), which is used if D-channel access handling (collision resolution on the S bus) is required for example. Reprogramming of ITF takes effect only after the transmission of the current frame has been completed or after an XRES command.

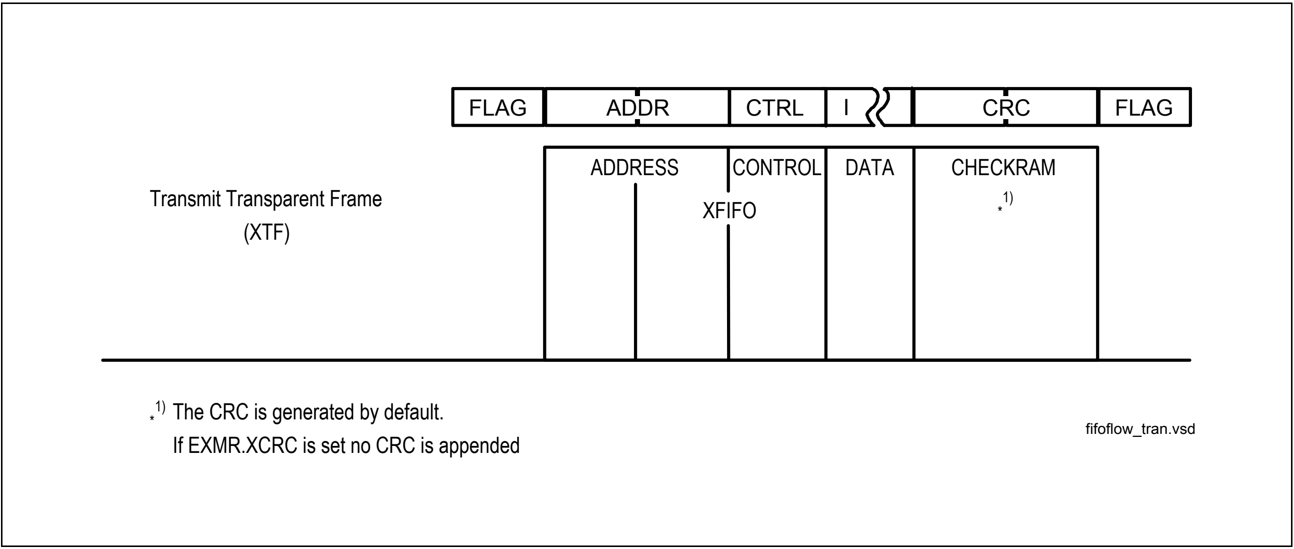

FLAG
ADDR
CTRL
I
CRC
FLAG
ADDRESS
CONTROL
DATA
CHECKRAM
XFIFO
*1)
Transmit Transparent Frame
(XTF)
*1) The CRC is generated by default.
If EXMR.XCRC is set no CRC is appended
fifoflow_tran.vsd

Figure 17-23 Transmit Data Flow

17.3.6 Extended Transparent Mode

This non-HDLC mode is selected by setting MODE2...0 to ’100’. In extended transparent mode fully transparent data transmission/reception without HDLC framing is performed i.e. without FLAG generation/recognition, CRC generation/check, bitstuffing mechanism. This allows user specific protocol variations.

17.3.6.1 Transmitter

The transmitter sends the data out of the FIFO without manipulation. Transmission is always IOM-frame aligned and byte aligned, i.e. transmission starts in the first selected channel (e.g. B1, B2, D) of the next IOM-frame.

The FIFO indications and commands are the same as in other modes.

If the microcontroller sets XTF & XME the transmitter responds with an XPR interrupt after sending the last byte, then it returns to its idle state (sending continuous ‘1’).

If the cts (clear to send) signal is set to “0” during transmission the transparent HDLC controller responds always with an XMR (transmit message repeat) interrupt and stops transmission.

If the CPU fails to respond to an XPR interrupt in time and the transmitter runs out of data then it will assert an XDU (transmit data underrun) interrupt.

17.3.6.2 Receiver

The reception is IOM-frame aligned and byte aligned, like transmission, i.e. reception starts in the first selected channel (B1, B2, D on IOM-2) of the next IOM frame. The FIFO indications and commands are the same as in other modes.

All incoming data bytes are stored in the RFIFO. If the FIFO is full an RFO interrupt is asserted (EXMR.SRA = ’0’).

Note: In the extended transparent mode the EXMR register has to be set to ’xxx00000’

17.4 HDLC Controller Interrupts

All interrupt sources from the ISTAH register are combined (ORed) to a single HDLC controller interrupt signal hint. Each of the interrupt sources can individually be masked in the MASKH register. A masked interrupt is not indicated in the ISTAH register but remains internally stored and pending until the interrupt is unmasked and read by the host.

The HDLC controller interrupts XDU and XMR have special impact on the internal functions. E.g. the transmitter of the HDLC controller is locked if a data underrun condition occurs and the ISTAH.XDU is not read (the interrupt can only be read if unmasked), same applies for XMR.

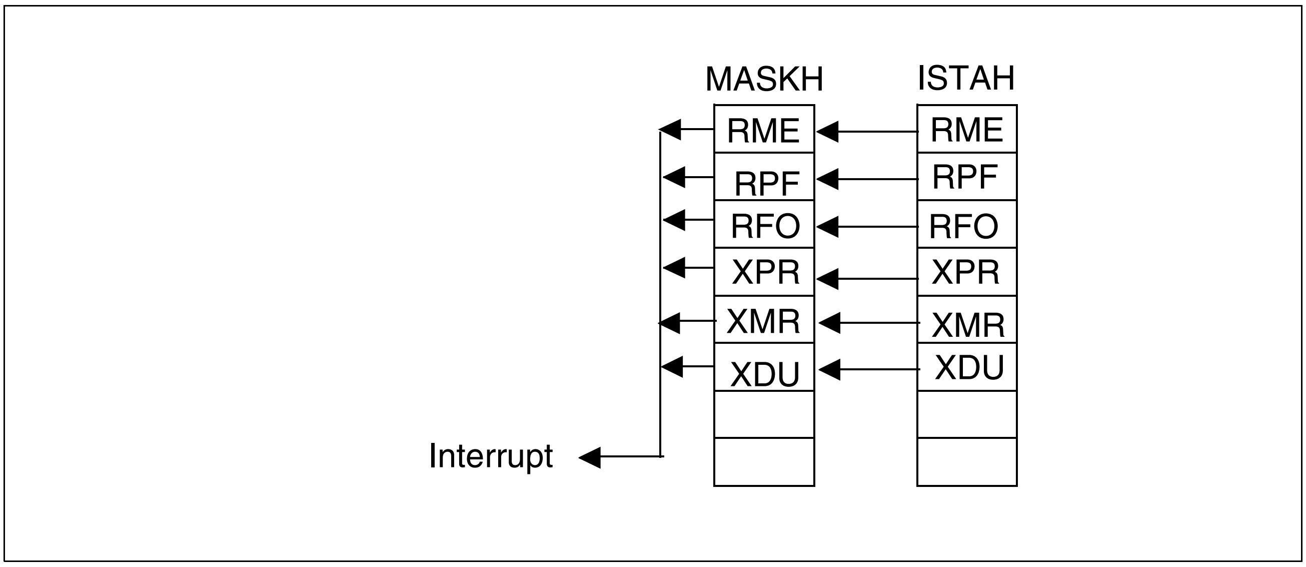


Figure 17-24 Interrupt Status Registers of the HDLC Controller

17.5 Register Description

The register mapping is shown in **Figure 17-25.** All listed addresses refer to XBUS-SFR's (see **Figure 3-1** and **Figure 25-1**). The corresponding addresses are described as address offsets to the base address of 00'DE00'$_H$.

The register SRES with the reset bit for HDLC and IOM2 Handler is described in Chapter 24.5.

Note: Unused register bits always have an undefined reset value. The reset value for a whole register in hexadecimal notation does not apply to unused bits. Unused bits may be '0', '1', or '-'. Only if indicated with '-' , a bit can be written as '1' or '0'; in all other cases the predefined value must be written.

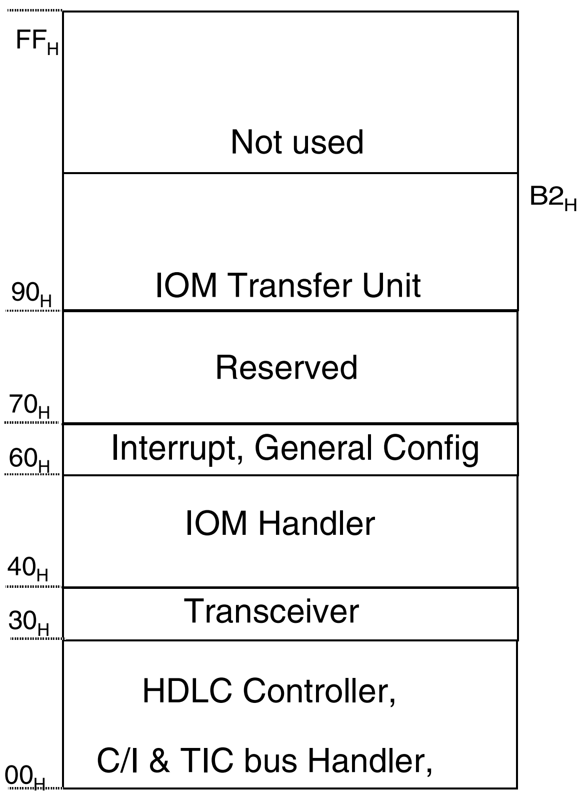


Figure 17-25 Address Offsets and Register Mapping

HDLC Control Registers, CI Handler

Name	7	6	5	4	3	2	1	0	ADDR	R/W	RES
RFIFO	D-Channel Receive FIFO								00_H-$1F_H$	R	
XFIFO	D-Channel Transmit FIFO								00_H-$1F_H$	W	
ISTAH	RME	RPF	RFO	XPR	XMR	XDU	0	0	20_H	R	10_H
MASKH	RME	RPF	RFO	XPR	XMR	XDU	0	0	20_H	W	FC_H
STAR	XDOV	XFW	0	0	RACI	0	XACI	0	21_H	R	40_H
CMDR	RMC	RRES	0	0	XTF	0	XME	XRES	21_H	W	00_H
MODEH	MDS2	MDS1	MDS0	0	RAC	DIM2	DIM1	DIM0	22_H	R/W	$C0_H$
EXMR	XFBS	RFBS		SRA	XCRC	RCRC	0	ITF	23_H	R/W	00_H
SAP1	SAPI1						0	MHA	25_H	W	FC_H
SAP2	SAPI2						0	MLA	26_H	W	FC_H
RBCL	RBC7							RBC0	26_H	R	00_H
RBCH	0	0	0	OV	RBC11			RBC8	27_H	R	00_H
TEI1	TEI1							EA	27_H	W	FF_H
TEI2	TEI2							EA	28_H	W	FF_H
RSTA	VFR	RDO	CRC	RAB	SA1	SA0	C/R	TA	28_H	R	$0F_H$
TMH	0	0	0	0	0	0	0	TLP	29_H	R/W	00_H
	Reserved								$2A_H$-$2D_H$		
CIR0	CODR0				CIC0	CIC1	S/G	BAS	$2E_H$	R	$F3_H$
CIX0	CODX0				TBA2	TBA1	TBA0	BAC	$2E_H$	W	FE_H
CIR1	CODR1						0	0	$2F_H$	R	FC_H
CIX1	CODX1						CICW	CI1E	$2F_H$	W	FE_H

IOM Handler (Timeslot , Data Port Selection, CDA Data and CDA Control Register)

Name	7	6	5	4	3	2	1	0	ADDR	R/W	RES
CDA10	Controller Data Access Register (CH10)								40_H	R/W	FF_H
CDA11	Controller Data Access Register (CH11)								41_H	R/W	FF_H
CDA20	Controller Data Access Register (CH20)								42_H	R/W	FF_H
CDA21	Controller Data Access Register (CH21)								43_H	R/W	FF_H
CDA_ TSDP10	DPS	0	0	0	TSS				44_H	R/W	00_H
CDA_ TSDP11	DPS	0	0	0	TSS				45_H	R/W	01_H
CDA_ TSDP20	DPS	0	0	0	TSS				46_H	R/W	80_H
CDA_ TSDP21	DPS	0	0	0	TSS				47_H	R/W	81_H
CO_ TSDP10	DPS	0	0	0	TSS				48_H	R/W	80_H
CO_ TSDP11	DPS	0	0	0	TSS				49_H	R/W	81_H
CO_ TSDP20	DPS	0	0	0	TSS				$4A_H$	R/W	81_H
CO_ TSDP21	DPS	0	0	0	TSS				$4B_H$	R/W	85_H
TR_ TSDP_B1	DPS	0	0	0	TSS				$4C_H$	R/W	00_H
TR_ TSDP_B2	DPS	0	0	0	TSS				$4D_H$	R/W	01_H
CDA1_ CR	0	0	EN_ TBM	EN_I1	EN_I0	EN_O1	EN_O0	SWAP	$4E_H$	R/W	00_H

CDA2_ CR	0	0	EN_ TBM	EN_I1	EN_I0	EN_O1	EN_O0	SWAP	$4F_H$	R/W	00_H

IOM Handler (Control Registers, Synchronous Transfer Interrupt Control),

Name	7	6	5	4	3	2	1	0	ADDR	R/W	RES
TR_CR (CI_CS=0)	EN_ D	EN_ B2R	EN_ B1R	EN_ B2X	EN_ B1X	0	0	0	50_H	R/W	$F8_H$
BCHA_CR	1	0	0	EN_ BC2	EN_ BC1	0	0	0	51		80
BCHB_CR	1	0	0	EN_ BC2	EN_ BC1	0	0	1	52		81
DCI_CR (CI_CS=0)	DPS_ CI1	EN_ CI1	EN_ D	0	0	0	0	0	53_H	R/W	$A0_H$
DCIC_CR (CI_CS=1)	DPS_ CI0	EN_ CI0	DPS_ D	0	0	0	0	0	53_H	R/W	$E0_H$
SDS1_CR	ENS_ TSS	ENS_ TSS+1	ENS_ TSS+3	0	TSS				55_H	R/W	00_H
SDS2_CR	ENS_ TSS	ENS_ TSS+1	ENS_ TSS+3	0	TSS				56_H	R/W	00_H
IOM_CR	SPU	DIS_ AW	CI_CS	TIC_ DIS	EN_ BCL	CLKM	DIS_ OD	DIS_ IOM	57_H	R/W	08_H
STI	STOV 21	STOV 20	STOV 11	STOV 10	STI 21	STI 20	STI 11	STI 10	58_H	R	00_H
ASTI	0	0	0	0	ACK 21	ACK 20	ACK 11	ACK 10	58_H	W	00_H
MSTI	STOV 21	STOV 20	STOV 11	STOV 10	STI 21	STI 20	STI 11	STI 10	59_H	R/W	FF_H
SDS_ CONF	0	0	0	0	0	0	SDS2 _ BCL	SDS1 _ BCL	$5A_H$	R/W	00_H

MCDA	MCDA21	MCDA20	MCDA11	MCDA10	$5B_H$	R	FF_H

.

Interrupt, General Configuration Registers

NAME	7	6	5	4	3	2	1	0	ADDR	R/W	RES
ISTA	0	0	ST	CIC	0	TRAN	0	ICD	60_H	R	00_H
MASK	1	1	ST	CIC	1	TRAN	1	ICD	60_H	W	FF_H
MODE1	0	0	0	0	0	CFS	0	0	62_H	R/W	00_H
SRES	RES_CI	0	0	0	RES_DCH	RES_IOM	RES_TR	0	64_H	W	00_H
	reserved								65_H-$6F_H$		

.x

IOM Handler (Buffer for Transfer Unit x = ITR),

Name	7	6	5	4	3	2	1	0	ADDR	R/W	RES
ITRDU									90_H	R/W	00_H
ITRDD									91_H	R/W	00_H
ITRICV									92_H	R/W	FF_H

IOM Handler (Transfer Unit)

Name																			
ITR_CR	ITR FR IR								EN 7	EN 6	EN 5	EN 4	EN 3	EN 2	EN 1	EN 0	94_H	R/W	
ITR_MSK_0	DIR	PRT	CED	0	TS 11	TS 10	TS 9	TS 8	TS 7	TS 6	TS 5	TS 4	TS 3	TS 2	TS 1	TS 0	96_H	R/W	
ITR_MSK_1	DIR	PRT	0	0	TS 11	TS 10	TS 9	TS 8	TS 7	TS 6	TS 5	TS 4	TS 3	TS 2	TS 1	TS 0	98_H	R/W	

ITR_MSK_2	DIR	PRT	0	0	TS11	TS10	TS9	TS8	TS7	TS6	TS5	TS4	TS3	TS2	TS1	TS0	$9A_H$	R/W
ITR_MSK_3	DIR	PRT	0	0	TS11	TS10	TS9	TS8	TS7	TS6	TS5	TS4	TS3	TS2	TS1	TS0	$9C_H$	R/W
ITR_MSK_4	DIR	PRT	0	0	TS11	TS10	TS9	TS8	TS7	TS6	TS5	TS4	TS3	TS2	TS1	TS0	$9E_H$	R/W
ITR_MSK_5	DIR	PRT	0	0	TS11	TS10	TS9	TS8	TS7	TS6	TS5	TS4	TS3	TS2	TS1	TS0	$A0_H$	R/W
ITR_MSK_6	DIR	PRT	0	0	TS11	TS10	TS9	TS8	TS7	TS6	TS5	TS4	TS3	TS2	TS1	TS0	$A2_H$	R/W
ITR_MSK_7	DIR	PRT	0	0	TS11	TS10	TS9	TS8	TS7	TS6	TS5	TS4	TS3	TS2	TS1	TS0	$A4_H$	R/W

.

IOM Handler Waitstates

Name								
IWSR	RE	*reserved*	ASA-WS	G-WS	HF-WS	$B0_H$	R/W	$FFFF_H$

17.5.1 HDLC Control and C/I Registers

17.5.1.1 RFIFO - Receive FIFO

	7 … 0	
RFIFO	Receive data	RD (00_H-$1F_H$)

A read access to any address within the range 00h-1Fh gives access to the "current" FIFO location selected by an internal pointer which is automatically incremented after each read access. This allows for the use of efficient "move string" type commands by the microcontroller.

The RFIFO contains up to 64 bytes of received data.

After an ISTAH.RPF interrupt, a complete data block is available. The block size can be 4, 8, 16, 32 bytes depending on the EXMR.RFBS setting.

After an ISTAH.RME interrupt, the number of received bytes can be obtained by reading the RBCL/RBCH registers.

17.5.1.2 XFIFO - Transmit FIFO

	7 … 0	
XFIFO	Transmit data	WR (00_H-$1F_H$)

A write access to any address within the range 00-$1F_H$ gives access to the "current" FIFO location selected by an internal pointer which is automatically incremented after each write access. This allows the use of efficient "move string" type commands by the microcontroller.

Depending on EXMR.XFBS up to 16 bytes of transmit data can be written to the XFIFO following an ISTAH.XPR interrupt.

17.5.1.3 ISTAH - Interrupt Status Register HDLC

Value after reset: 10_H

	7							0	
ISTAH	RME	RPF	RFO	XPR	XMR	XDU	0	0	RD (20_H)

Refer also to Chapter 8.2.3 for a description of interrupt handling.

RME ... Receive Message End

One complete frame of length less than or equal to the defined block size (EXMR.RFBS) or the last part of a frame of length greater than the defined block size has been received. The contents are available in the RFIFO. The message length and additional information may be obtained from RBCH and RBCL and the RSTA register.

RPF ... Receive Pool Full

A data block of a frame longer than the defined block size (EXMR.RFBS) has been received and is available in the RFIFO. The frame is not yet complete.

RFO ... Receive Frame Overflow

The received data of a frame could not be stored, because the RFIFO is occupied. The whole message is lost.
This interrupt can be used for statistical purposes and indicates that the microcontroller does not respond quickly enough to an RPF or RME interrupt (ISTAH).

XPR ... Transmit Pool Ready

A data block of up to the defined block size (EXMR.XFBS) can be written to the XFIFO. An XPR interrupt will be generated in the following cases:

- after an XTF or XME command as soon as the 16 or 32 respectively bytes in the XFIFO are available and the frame is not yet complete
- after an XTF together with an XME command is issued, when the whole frame has been transmitted

XMR ... Transmit Message Repeat

The transmission of the last frame has to be repeated because a collision has been detected after the 16th/32th data byte of a transmit frame.

XDU ... Transmit Data Underrun

The current transmission of a frame is aborted by transmitting seven ’1’s because the XFIFO holds no further data. This interrupt occurs whenever the microcontroller has failed to respond to an XPR interrupt (ISTAH register) quickly enough, after having initiated a transmission and the message to be transmitted is not yet complete.

17.5.1.4 MASKH - Mask Register HDLC

Value after reset: FC_H

	7							0	
MASKH	RME	RPF	RFO	XPR	XMR	XDU	0	0	WR (20_H)

Each interrupt source in the ISTAH register can be selectively masked by setting to ’1’ the corresponding bit in MASK. Masked interrupt status bits are not indicated when ISTAH is read. Instead, they remain internally stored and pending, until the mask bit is reset to ’0’.

17.5.1.5 STAR - Status Register

Value after reset: 40_H

	7							0	
STAR	XDOV	XFW	0	0	RACI	0	XACI	0	RD (21_H)

XDOV ... Transmit Data Overflow

More than 16/32 bytes have been written in one pool of the XFIFO, i.e. data has been overwritten.

XFW ... Transmit FIFO Write Enable

Data can be written in the XFIFO. This bit may be polled instead of (or in addition to) using the XPR interrupt.

RACI ... Receiver Active Indication

The HDLC receiver is active when RACI = ’1’. This bit may be polled. The RACI bit is set active after a begin flag has been received and is reset after receiving an abort sequence.

XACI ... Transmitter Active Indication

The HDLC-transmitter is active when XACI = ’1’. This bit may be polled. The XACI-bit is active when an XTF-command is issued and the frame has not been completely transmitted.

17.5.1.6 CMDR - Command Register

Value after reset: 00_H

	7							0	
CMDR	RMC	RRES	0	0	XTF	0	XME	XRES	WR (21_H)

RMC ... Receive Message Complete

Reaction to RPF (Receive Pool Full) or RME (Receive Message End) interrupt. By setting this bit, the microcontroller confirms that it has fetched the data, and indicates that the corresponding space in the RFIFO may be released.

RRES ... Receiver Reset

HDLC receiver is reset, the RFIFO is cleared of any data.

XTF ... Transmit Transparent Frame

After having written up to 16 bytes (EXMR.XFBS) in the XFIFO, the microcontroller initiates the transmission of a transparent frame by setting this bit to '1'. Except in the extended transparent mode the opening flag is automatically added to the message.

XME ... Transmit Message End

By setting this bit to '1' the microcontroller indicates that the data block written last in the XFIFO completes the corresponding frame. Except in the extended transparent mode the transmission is terminated by appending the CRC and the closing flag sequence to the data.

XRES ... Transmitter Reset

HDLC transmitter is reset and the XFIFO is cleared of any data. This command can be used by the microcontroller to abort a frame currently in transmission.

Note: After an XPR interrupt further data has to be written to the XFIFO and the appropriate Transmit Command (XTF) has to be written to the CMDR register again to continue transmission, when the current frame is not yet complete (see also XPR in ISTAH).
During frame transmission, the 0-bit insertion according to the HDLC bit-stuffing mechanism is done automatically except in the extended mode.

17.5.1.7 MODEH - Mode Register

Value after reset: $C0_H$

	7							0	
MODEH	MDS2	MDS1	MDS0	0	RAC	DIM2	DIM1	DIM0	RD/WR (22_H)

MDS2-0 **... Mode Select**

Determines the message transfer mode of the HDLC controller, as follows:

MDS2-0	Mode	Number of Address Bytes	Address Comparison 1.Byte	2.Byte	Remark
0 0 0	Reserved				
0 0 1	Reserved				
0 1 0	Non-Auto mode	1	TEI1,TEI2	–	One-byte address compare.
0 1 1	Non-Auto mode	2	SAP1,SAP2,SAPG	TEI1,TEI2,TEIG	Two-byte address compare.
1 0 0	Extended transparent mode				
1 1 0	Transparent mode 0	–	–	–	No address compare. All frames accepted.
1 1 1	Transparent mode 1	> 1	SAP1,SAP2,SAPG	–	High-byte address compare.
1 0 1	Transparent mode 2	> 1	–	TEI1,TEI2,TEIG	Low-byte address compare.

Note: SAP1, SAP2: two programmable address values for the first received address byte (in the case of an address field longer than 1 byte);
SAPG = fixed value FC / FE_H.
TEI1, TEI2: two programmable address values for the second (or the only, in the case of a one-byte address) received address byte; TEIG = fixed value FF_H

*Two different methods of the high byte and/or low byte address comparision can be selected by setting SAP1.MHA and/or SAP2.MLA (see also description of these bits in **chapter 17.5.1.9** or **17.5.1.11** respectively)*

RAC **... Receiver Active**

The HDLC receiver is activated when this bit is set to ’1’. If it is ’0’ the HDLC data is not evaluated in the receiver.

DIM2-0 **... Digital Interface Modes**

These bits define the characteristics of the IOM Data Ports (DU, DD). The DIM0 bit enables/disables the collission detection. The DIM1 bit enables/disables the TIC bus access. The effect of the individual DIM bits is summarized in **table 17-5**.

Table 17-5 IOM®-2 Terminal Modes

DIM2	DIM1	DIM0	Characteristics
0	x	0	Transparent D-channel, the collission detection is disabled
0	x	1	Stop/go bit evaluated for D-channel access handling
0	0	x	Last octet of IOM channel 2 used for TIC bus access
0	1	x	TIC bus access is disabled
1	x	x	Reserved

17.5.1.8 EXMR- Extended Mode Register

Value after reset: 00_H

	7						0	
EXMR	XFBS	RFBS	SRA	XCRC	RCRC	0	ITF	RD/WR (23_H)

XFBS **... Transmit FIFO Block Size**

0: Block size for the transmit FIFO data is16 byte
1: Block size for the transmit FIFO data is 8 byte

Note: A change of XFBS will take effect after a transmitter command (CMDR.XME, CMDR.XRES, CMDR.XTF) has been written

RFBS **Receive FIFO Block Size**

RFBS Bit6	RFBS Bit5	Block Size Receive FIFO
0	0	32 byte
0	1	16 byte
1	0	8 byte
1	1	4 byte

Note: A change of RFBS will take effect after a receiver command (CMDR.RMC, CMDR.RRES,) has been written

SRA **… Store Receive Address**

0: Receive Address is not stored in the RFIFO
1: Receive Address is stored in the RFIFO

XCRC **… Transmit CRC**

0: CRC is transmitted
1: CRC is not transmitted

RCRC **… Receive CRC**

0: CRC is not stored in the RFIFO
1: CRC is stored in the RFIFO

ITF **… Interframe Time Fill**

Selects the inter-frame time fill signal which is transmitted between HDLC-frames.

0: Idle (continuous ’1’)
1: Flags (sequence of patterns: ‘0111 1110’)

Note: ITF must be set to ’0’ for power down mode. In applications with D-channel access handling (collision resolution), the only possible inter-frame time fill is idle (continuous ’1’)*. Otherwise the D-channel on the line interface can not be accessed*

17.5.1.9 SAP1 - SAPI1 Register

Value after reset: FC_H

	7		0	
SAP1	SAPI1	0	MHA	WR (25_H)

SAPI1 **... SAPI1 value**

Value of the first programmable Service Access Point Identifier (SAPI) according to the ISDN LAPD protocol.

MHA **... Mask High Address**

0: The SAPI address of an incomming frame is compared with SAP1, SAP2, SAPG
1: The SAPI address of an incomming frame is compared with SAP1 and SAPG. SAP1 can be masked with SAP2 thereby bitpositions of SAP1 are not compared if they are set to '1' in SAP2.

17.5.1.10 RBCL - Receive Frame Byte Count Low

Value after reset: 00_H

	7	0	
RBCL	RBC7	RBC0	RD (26_H)

RBC7-0 **... Receive Byte Count**

Eight least significant bits of the total number of bytes in a received message.

17.5.1.11 SAP2 - SAPI2 Register

Value after reset: FC_H

	7		0	
SAP2	SAPI2	0	MLA	WR (26_H)

SAPI2 **... SAPI2 value**

Value of the second programmable Service Access Point Identifier (SAPI) according to the ISDN LAPD-protocol.

MLA **... Mask Low Address**

0:	The TEI address of an incomming frame is compared with TEI1, TEI2, TEIG
1:	The TEI address of an incomming frame is compared with TEI1 andTEIG. TEI1 can be masked with TEI2 thereby bitpositions of TEI1 are not compared if they are set to '1' in TEI2

17.5.1.12 RBCH - Receive Frame Byte Count High

Value after reset: 00_H.

	7					0	
RBCH	0	0	0	OV	RBC11	RBC8	RD (27_H)

OV **... Overflow**

A '1' in this bit position indicates a message longer than $(2^{12} - 1) = 4095$ bytes .

RBC11-8 **... Receive Byte Count**

Four most significant bits of the total number of bytes in a received message.

Note: Normally RBCH and RBCL should be read by the microcontroller after an RME-interrupt in order to determine the number of bytes to be read from the RFIFO, and the total message length. The contents of the registers are valid only after an RME or RPF interrupt, and remain so until the frame is acknowledged via the RMC bit or RRES.

17.5.1.13 TEI1 - TEI1 Register 1

Value after reset: FF_H

	7	0	
TEI1	TEI1	EA	WR (27_H)

TEI1 ... Terminal Endpoint Identifier

In all message transfer modes except in transparent modes 0, 1 and extended transparent mode, TEI1 is used for address recognition. In the case of a two-byte address field, it contains the value of the first programmable Terminal Endpoint Identifier according to the ISDN LAPD-protocol.

In non-auto-modes with one-byte address field, TEI1 is a command address, according to X.25 LAPB.

EA **... Address field Extension bit**

This bit is set to '1' according to HDLC/LAPD.

17.5.1.14 RSTA - Receive Status Register

Value after reset: $0F_H$

	7							0	
RSTA	VFR	RDO	CRC	RAB	SA1	SA0	C/R	TA	RD (28_H)

VFR **... Valid Frame**

Determines whether a valid frame has been received.
The frame is valid (1) or invalid (0).
A frame is invalid when there is not a multiple of 8 bits between flag and frame end (flag, abort).

RDO **... Receive Data Overflow**

If RDO=1, at least one byte of the frame has been lost, because it could not be stored in RFIFO.

CRC **... CRC Check**

The CRC is correct (1) or incorrect (0).

RAB **... Receive Message Aborted**

The receive message was aborted by the remote station (1), i.e. a sequence of seven 1's was detected before a closing flag.

SA1-0 **... SAPI Address Identification**
TA **... TEI Address Identification**

SA1-0 are significant in non-auto-mode with a two-byte address field, as well as in transparent mode 3. TA is significant in all modes except in transparent modes 0 and 1.

Two programmable SAPI values (SAP1, SAP2) plus a fixed group SAPI (SAPG of value FC/FE_H), and two programmable TEI values (TEI1, TEI2) plus a fixed group TEI (TEIG of value FF_H), are available for address comparison.

The result of the address comparison is given by SA1-0 and TA, as follows:

				Address Match with	
	SA1	**SA0**	**TA**	**1st Byte**	**2nd Byte**
Number of Address Bytes = 1	x	x	0	TEI2	-
	x	x	1	TEI1	-
Number of address Bytes=2	0	0	0	SAP2	TEIG
	0	0	1	SAP2	TEI2
	0	1	0	SAPG	TEIG
	0	1	1	SAPG	TEI1 or TEI2
	1	0	0	SAP1	TEIG
	1	0	1	SAP1	TEI1
	1	1	x	reserved	

C/R **... Command/Response**

The C/R bit contains the C/R bit of the received frame (Bit1 in the SAPI address)

Note: The contents of RSTA corresponds to the last received HDLC frame; it is duplicated into RFIFO for every frame (last byte of frame)

Note: If SAP1 and SAP2 contains identical values, the combination 001 will be omitted.

17.5.1.15 TEI2 - TEI2 Register

Value after reset: FF_H

	7	0	
TEI2	TEI2	EA	WR (28_H)

TEI2 **... Terminal Endpoint Identifier**

In all message transfer modes except in transparent modes 0, 1 and extended transparent mode, TEI2 is used for address recognition. In the case of a two-byte address field, it contains the value of the second programmable Terminal Endpoint Identifier according of the ISDN LAPD-protocol.
In non-auto-modes with one-byte address field, TEI2 is a response address, according to X.25 LAPD.

EA **... Address field Extension bit**

This bit is to be set to ’1’ according to HDLC/LAPD.

17.5.1.16 TMH -Test Mode Register HDLC

Value after reset: 00_H

	7							0	
TMH	0	0	0	0	0	0	0	TLP	RD/WR (29_H)

TLP **... Test Loop**

The TX path of layer-2 is internally connected with the RX path of layer-2. Data coming from the layer 1 controller will not be forwarded to the layer 2 controller.

Bit 7:1 have always be programmed to ’0’

17.5.1.17 CIR0 - Command/Indication Receive 0

Value after reset: $F3_H$

	7					0	
CIR0	CODR0		CIC0	CIC1	S/G	BAS	RD ($2E_H$)

CODR0 **... C/I Code 0 Receive**

Value of the received Command/Indication code. A C/I-code is loaded in CODR0 only after being the same in two consecutive IOM-frames and the previous code has been read from CIR0.

CIC0 **... C/I Code 0 Change**

A change in the received Command/Indication code has been recognized. This bit is set only when a new code is detected in two consecutive IOM-frames. It is reset by a read of CIR0.

CIC1 **... C/I Code 1 Change**

A change in the received Command/Indication code in IOM-channel 1 has been recognized. This bit is set when a new code is detected in one IOM-frame. It is reset by a read of CIR0.

S/G **... Stop/Go Bit Monitoring**

Indicates the availability of the D-channel on the line interface.

1: Stop
0: Go

BAS **... Bus Access Status**

Indicates the state of the TIC-bus:

0: The INCA-D itself occupies the D- and C/I-channel
1: Another device occupies the D- and C/I-channel

Note: The CODR0 bits are updated every time a new C/I-code is detected in two consecutive IOM-frames. If several consecutive valid new codes are detected and CIR0 is not read, only the first and the last C/I code is made available in CIR0 at the first and second read of that register, respectively.

17.5.1.18 CIX0 - Command/Indication Transmit 0

Value after reset: FE_H

	7				0	
CIX0	CODX0	TBA2	TBA1	TBA0	BAC	WR ($2E_H$)

CODX0 **... C/I-Code 0 Transmit**

Code to be transmitted in the C/I-channel 0.

TBA2-0 **... TIC Bus Address**

Defines the individual address for the INCA-D on the IOM bus.

This address is used to access the C/I- and D-channel on the IOM interface.

Note: If only one device is liable to transmit in the C/I- and D-channels of the IOM it should always be given the address value ’7’.

BAC **... Bus Access Control**

Only valid if the TIC-bus feature is enabled (MODE:DIM2-0).

If this bit is set, the INCA-D will try to access the TIC-bus to occupy the C/I-channel even if no D-channel frame has to be transmitted. It should be reset when the access has been completed to grant a similar access to other devices transmitting in that IOM-channel.

Note: If the TIC-bus address (TBA2-0) is programmed to ’7’ and is not blocked by another device the INCA-D writes its C/I0 code to IOM continuously.

17.5.1.19 CIR1 - Command/Indication Receive 1

Value after reset: FC_H

	7		0	
CIR1	CODR1	0	0	RD ($2F_H$)

CODR1 **... C/I-Code 1 Receive**

Value of the received Command/Indication code.

17.5.1.20 CIX1 - Command/Indication Transmit 1

Value after reset: FE_H

	7		0	
CIX1	CODX1	CICW	CI1E	WR ($2F_H$)

CODX1 **... C/I-Code 1 Transmit**

Bits 7-2 of C/I-channel 1

CICW **... C/I-Channel Width**

CICW selects between a 4 bit (’0’) and 6 bit (’1’) C/I1 channel width

CI1E **... C/I-channel 1 interrupt enable**

Interrupt generation ISTA.CIC of CIR0.CIC1 is enabled (1) or masked (0).

17.5.2 IOM Handler (CDA) Register

17.5.2.1 CDAxy - Controller Data Access Register xy

Value after reset: See table below

	7 … 0	
CDAxy	Controller Data Access Register	RD/WR (40_H-43_H)

Data register CDAxy which can be accessed from the controller.

Register	Value after Reset	Register Address
CDA10	FF_H	40_H
CDA11	FF_H	41_H
CDA20	FF_H	42_H
CDA21	FF_H	43_H

17.5.2.2 XXX_TSDPxy - Time Slot and Data Port Selection for CHxy

Vaule after reset: See table below

	7				0	
XXX_ TSDPxy	DPS	0	0	0	TSS	RD/WR (44_H-$4D_H$)

Register	Value after Reset	Register Address
CDA_TSDP10	00_H (= output on B1-DD)	44_H
CDA_TSDP11	01_H (= output on B2-DD)	45_H
CDA_TSDP20	80_H (= output on B1-DU)	46_H
CDA_TSDP21	81_H (= output on B2-DU)	47_H
CO_TSDP10	80_H (= output on B1-DU)	48_H
CO_TSDP11	81_H (= output on B2-DU)	49_H
CO_TSDP20	81_H (= output on B2-DU)	$4A_H$
CO_TSDP21	85_H (= output on IC2-DU)	$4B_H$

TR_TSDP_B1	00_H (= output on B1-DD)	$4C_H$
TR_TSDP_B2	01_H (= output on B2-DD)	$4D_H$

This register determines the time slots and the data ports on the IOM-2 Interface for the data channels xy of the functional units XXX (Controller Data Access (CDA), Codec (CO) and Transceiver (TR)).

DPS ... Data Port Selection

0: The data channel xy of the functional unit XXX is output on DD.
The data channel xy of the functional unit XXX is input from DU.
1: The data channel xy of the functional unit XXX is output on DU.
The data channel xy of the functional unit XXX is input from DD.

Note: For the CDA (controller data access) data the input is determined by the CDA_CRx.SWAP bit. If SWAP = '0' the input for the CDAxy data is vice versa to the output setting for CDAxy. If the SWAP = '1' the input from CDAx0 is vice versa to the output setting of CDAx1 and the input from CDAx1 is vice versa to the output setting of CDAx0. See controller data access description)

TSS ... Timeslot Selection

Selects one of the 12 timeslots from 0...11 on the IOM-2 interface for the data channels.

17.5.2.3 CDAx_CR - Control Register Controller Data Access CH1x

Value after reset: See table below

	7							0	
CDAx_ CR	0	0	EN_ TBM	EN_I1	EN_I0	EN_O1	EN_O0	SWAP	RD/WR ($4E_H$-$4F_H$)

Register	Value after Reset	Register Address
CDA1_CR	00_H	$4E_H$
CDA2_CR	00_H	$4F_H$

EN_TBM ... Enable TIC Bus Monitoring

0: The TIC bus monitoring is disabled
1: The TIC bus monitoring with the CDAx0 register is enabled. The TSDPx0 register must be set to 08_H for monitoring from DU or 88_H for monitoring from DD respectively.

EN_I1, EN_I0 ... Enable Input CDAx0, CDAx1

0: The input of the CDAx0, CDAx1 register is disabled
1: The input of the CDAx0, CDAx1 register is enabled

EN_O1, EN_O0 ... Enable Output CDAx0, CDAx1

0: The output of the CDAx0, CDAx1 register is disabled
1: The output of the CDAx0, CDAx1 register is enabled

SWAP ... Swap Inputs

0: The time slot and data port for the input of the CDAxy register is defined by its own TSDPxy register. The data port for the CDAxy input is vice versa to the output setting for CDAxy.
1: The input (time slot and data port) of the CDAx0 is defined by the TSDP register of CDAx1 and the input of CDAx1 is defined by the TSDP register of CDAx0. The data port for the CDAx0 input is vice versa to the output setting for CDAx1. The data port for the CDAx1 input is vice versa to the output setting for CDAx0. The input definition for time slot and data port CDAx0 are thus swapped to CDAx1 and for CDAx1 to CDAx0. The outputs are not affected by the SWAP bit.

17.5.3 IOM Handler (Control register)

17.5.3.1 TR_CR - Control Register Transceiver Data

Value after reset: $F8_H$

	7							0	
TR_CR	EN_ D	EN_ B2R	EN_ B1R	EN_ B2X	EN_ B1X	0	0	0	RD/WR (50_H)

EN_D	**... Enable D-Channel Data**
EN_B2R	**... Enable B2 Data received from IOM**
EN_B1R	**... Enable B1 Data received from IOM**
EN_B2X	**... Enable B2 Data to be transmitted to IOM**
EN_B1X	**... Enable B1 Data to be transmitted to IOM**

0: The transceiver data _xxx is disabled
1: The transceiver data _xxx is enabled

17.5.3.2 BCHx_CR - Control Register B-Channel Data

	7							0	
BCHx_CR	0	0	0	EN_ BC2	EN_ BC1	0	0	0/1	RD/WR (51,52)

Register	Register Address	Value after Reset
BCHA_CR	11_H	80_H
BCHB_CR	12_H	81_H

The registers CO_TSDPxy (see 17.5.2.2) select the IOM-2 timeslots and data channel .

The registers BCHx_CR are used to enable each of the selected timeslots according to the table below.

BCHA_CR.ENBC1	CO_TSDP11
BCHA_CR.ENBC2	CO_TSDP12
BCHB_CR.ENBC1	CO_TSDP21
BCHB_CR.ENBC2	CO_TSDP22

EN_BCx ... Enable Timelsot (8-bit) and data channel

0: Corresponding timeslot and channel can not be accessed.

1: Corresponding timeslot and channel can be accessed.

17.5.3.3 DCI_CR - Control Register for HDLC and CI1 Data

Value after reset: $A0_{H;}$

	7							0	
DCI_CR	DPS_ CI1	EN_ CI1	EN_ D	0	0	0	0	0	RD/WR (53_H)

DPS_CI1 ... Data Port Selection CI1 Data

0: The CI1 data is output on DD and input from DU
1: The CI1 data is output on DU and input from DD

EN_CI1 ... Enable CI1 Data

0:The CI1 data access is disabled
1: The CI1 data access is enabled

EN_D ... Enable D-Channel Data

0:The HDLC data access is disabled
1: The HDLC data access is enabled

17.5.3.4 DCIC_CR - Control Register for HDLC and CI1 Data

Value after reset: Software Controlled modes: $E0_{H;}$ TR: 40_H

.

	7							0	
DCIC_CR	DPS_ CI0	EN_ CI0	DPS_ D	0	0	0	0	0	RD/WR (53_H)

DPS_CI0 ... Data Port Selection CI0 Data

0: The CI0 data is output on DD and input from DU
1: The CI0 data is output on DU and input from DD

EN_CI0 ... Enable CI0 Data0:
0: CI0 data access is disabled
1: CI0 data access is enabled

DPS_D ... Data Port Selection for HDLC Controller

0: The D data is output on DD and input from DU
1: The D data is output on DU and input from DD

Note: This bit toggles the direction for the D-Channel HDLC Controller; the TIC Handler is unaffected

17.5.3.5 SDSx_CR - Control Register Serial Data Strobe x

Value after reset: 00_H

	7				0	
SDSx_CR	ENS_ TSS	ENS_ TSS+1	ENS_ TSS+3	0	TSS	RD/WR ($55\text{-}56_H$)

Register	Value after Reset	Register Address
SDS1_CR	00_H	55_H
SDS2_CR	00_H	$56F_H$

Note: ENS_TSS ... Enable Serial Data Strobe of timeslot TS
ENS_TSS+1 ... Enable Serial Data Strobe of timeslot TS+1

0: The serial data strobe or bit clock on SDSx for TS, TS+1 is disabled
1: The serial data strobe or bit clock on SDSx for TS, TS+1 is enabled

ENS_TSS+3 ... Enable Serial Data Strobe of timeslot TS+3 (D-Channel)

0: The serial data strobe or bit clock on SDSx for the D-channel (bit7, 6) of TS+3 is disabled
1: The serial data strobe or bit clock on SDSx for the D-channel (bit7, 6) of TS+3 is enabled

TSS ... Timeslot Selection

Selects one of 12 timeslots on the IOM-2 interface (with respect to FSC) during which SDSx is active. The data strobe signal allows standard data devices to access a programmable channel.

17.5.3.6 IOM_CR - Control Register IOM Data

Value after reset: : 01_H;

	7							0	
IOM_CR	SPU	0	CI_CS	TIC_ DIS	EN_ BCL	CLKM	DIS_ OD	DIS_ IOM	RD/WR (57_H)

SPU **... Software Power UP**

0: The DU line is normally used for transmitting data
1: Setting this bit to ’1’ will pull the DU line to low. This will enforce connected layer 1 devices to deliver IOM-clocking.

After a subsequent CIC-interrupt (C/I-code change; ISTA) and reception of the C/I-code ”PU” (Power Up indication in TE-mode) the microcontroller writes an AR or TIM command as C/I-code in the CIX0-register, resets the SPU bit and wait for the following CIC-interrupt.

CI_CS ... C/I Channel Selection

The channel selection for D-channel and C/I-channel is done in the channel select bits CS2-0 of register TR_CR (for the transceiver) and DCI_CR (for the D-channel controller and C/I-channel controller).

0: A write access to CS2-0 has effect on the configuration of D- and C/I-channel, whereas a read access deliveres the D-channel configuration only.

1: A write access to CS2-0 has effect on the configuration the C/I-channel only and a read access deliveres the C/I-channel configuration only.

TIC_DIS **... TIC Bus Disable**

0: The last octet of the last IOM time slot (TS 11) is treated as TIC bus
1: The last octet of the last IOM time slot is treated as any other timeslot. No TIC-bus handling possible.

Note: to use the build-in TIC-bus handler, it must be enabled additionally in MODEH

EN_BCL **... Enable Bit Clock BCL**

0: The BCL clock is disabled
1: The BCL clock is enabled

CLKM **... Clock Mode**

If the transceiver is disabled (DIS_TR = ’1’) the DCL from the IOM-2 interface is an input. With

0: A double clock per bit is expected
1: A single clock per bit is expected

DIS_OD ... Open Drain

0: IOM outputs are open drain driver
1: IOM outputs are push pull driver

DIS_IOM ... Disable IOM

DIS_IOM should be set to '1' if external devices connected to the IOM interface should be "disconnected" e.g. for power saving purposes or for not disturbing the internal IOM connection between layer 1 and layer 2. However, the internal operation between transceiver, B-channel and D-channel controller is independent of the DIS_IOM bit.

0: The IOM interface is enabled
1: The IOM interface is disabled (high impedance)

17.5.3.7 STI - Synchronous Transfer Interrupt

Value after reset: 00_H

	7							0	
STI	STOV 21	STOV 20	STOV 11	STOV 10	STI 21	STI 20	STI 11	STI 10	RD (58_H)

For all interrupts in the STI register following logical states are applied:

0: Interrupt is not acitvated
1: Interrupt is acitvated

STOVxy ... Synchronous Transfer Overflow Interrupt

Enabled STOV interrupts for a certain STIxy interrupt are generated when the STIxy has not been acknowledged in time via the ACKxy bit in the ASTI register. This must be one (for DPS='0') or zero (for DPS='1') BCL clocks before the time slot which is selected for the STOV.

STIxy ... Synchronous Transfer Interrupt

Depending on the DPS bit in the corresponding TSDPxy register the Synchronous Transfer Interrupt STIxy is generated two (for DPS='0') or one (for DPS='1') BCL clock after the selected time slot (TSDPxy.TSS).

Note: ST0Vxy and ACKxy are useful for synchronizing microcontroller accesses and receive/transmit operations. One BCL clock is equivalent to two DCL clocks.

17.5.3.8 ASTI - Acknowledge Synchronous Transfer Interrupt

Value after reset: 00_H

	7							0	
ASTI	0	0	0	0	ACK 21	ACK 20	ACK 11	ACK 10	WR (58_H)

ACKxy ... Acknowledge Synchronous Transfer Interrupt

After a STIxy interrupt the microcontroller has to acknowledge the interrupt by setting the corresponding ACKxy bit.

0: No activity is initiated
1: Sets the acknowledge bit ACKxy for a STIxy interrupt

17.5.3.9 MSTI - Mask Synchronous Transfer Interrupt

Value after reset: FF_H

	7							0	
MSTI	STOV 21	STOV 20	STOV 11	STOV 10	STI 21	STI 20	STI 11	STI 10	RD/WR (59_H)

For the MSTI register following logical states are applied:

0: Interrupt is not masked
1: Interrupt is masked

STOVxy ... Synchronous Transfer Overflow for STIxy

By masking the STOV bits the number and time of the STOV interrupts for a certain enabled STIxy interrupt can be controlled. For an enabled STIxy the own STOVxy is generated when the STOVxy is enabled (MSTI.STIxy and MSTI.STOVxy = ’0’). Additionally all other STOV interrupts of which the corresponding STI is disabled (MSTI.STI = ’1’ and MSTI.STOV = ’0’) are generated.

STIxy **... Synchronous Transfer Interrupt xy**

The STIxy interrupts can be masked by setting the corresponding mask bit to '1'. For a masked STIxy no STOV interrupt is generated.

17.5.3.10 SDS_CONF - Configuration Register for Serial Data Strobes

Value after reset: 00_H

	7							0	
SDS_ CONF	0	0	0	0	0	0	SDS2_ BCL	SDS1_ BCL	RD/WR ($5A_H$)

SDSx_BCL **... Enable IOM Bit Clock for SDSx**

0: The serial data strobe is generated in the programmed timeslot (see **Chapter 17.5.3.5**).
1: The IOM bit clock is generated in the programmed timeslot.

17.5.3.11 MCDA - Monitoring CDA Bits

Value after reset: FF_H

	7							0	
MCDA	MCDA21		MCDA20		MCDA11		MCDA10		RD ($5B_H$)
	Bit7	Bit6	Bit7	Bit6	Bit7	Bit6	Bit7	Bit6	

MCDAxy **... Monitoring CDAxy Bits**

Bit 7 and Bit 6 of the CDAxy registers are mapped into the MCDA register.

This can be used for monitoring the D-channel bits on DU and DD and the 'Echo bits' on the TIC bus with the same register

17.6 Interrupt and General Configuration

17.6.1 ISTA - Interrupt Status Register

Value after reset: 00_H

	7							0	
ISTA	0	0	ST	CIC	0	TRAN	0	ICD	RD (60)

For all interrupts in the ISTA register following logical states are applied:

0: Interrupt is not acitvated
1: Interrupt is acitvated

ICD ... HDLC Interrupt from D-channel

An interrupt originated from the HDLC controller of D-channel has been recognized.

ST ... Synchronous Transfer

This interrupt is generated to enable the microcontroller to lock on to the IOM timing for synchronous transfers. The source can be read from the STI register.

CIC ... C/I Channel Change

A change in C/I channel 0 or C/I channel 1 has been recognized. The actual value can be read from CIR0 or CIR1.

TRAN ... Transceiver Interrupt

An interrupt originated in the transceiver interrupt status register (ISTATR) has been recognized.

Note: A read of the ISTA register clears none of the interrupts. They are only cleared by reading the corresponding status register.

17.6.2 MASK - Mask Register

Value after reset: FF_H

	7							0	
MASK	1	1	ST	CIC	1	TRAN	1	ICD	WR (60)

For the MASK register following logical states are applied:

0: Interrupt is enabled
1: Interrupt is disabled

Each interrupt source in the ISTA register can selectively be masked/disabled by setting the corresponding bit in MASK to ’1’. Masked interrupt status bits are not indicated when ISTA is read. Instead, they remain internally stored and pending, until the mask bit is reset to ’0’.

Note: In the event of a C/I channel change, CIC is set in ISTA even if the corresponding mask bit in MASK is set, but no interrupt is generated.

17.6.3 MODE1 - Mode1 Register

Value after reset: 00_H

	7							0	
MODE1	0	0	0	0	0	CFS	0	0	RD/WR (62)

CFS ... Configuration Select

This bit determines clock relations and recovery on S/T and IOM interfaces.

0: The IOM interface clock and frame signals are always active, "Power Down" state included.
The states "Power Down" and "Power Up" are thus functionally identical except for the indication: PD = 1111 and PU = 0111.
With the C/I command Timing (TIM) the microcontroller can enforce the "Power Up" state and with C/I command Deactivation Indication (DI) the "Power Down" state is reached again.
However, it is also possible to activate the line interface directly with the C/I command Activate Request (AR 8/10/L) without the TIM command.

1: The IOM interface clock and frame signals are normally inactive ("Power Down").
For activating the IOM-2 clocks the "Power Up" state can be induced by software (IOM_CR.SPU) or by resetting CFS again.
After that theline-interface can be activated with the C/I command Activate Request (AR 8/10/L). The "Power Down" state can be reached again with the C/I command Deactivation Indication (DI).

Note: After reset the IOM interface is always active. To reach the "Power Down" state the CFS-bit has to be set.

17.6.4 ID - Identification Register

Value after reset: 01_H

	7			0	
ID	0	0	DESIGN		RD (64)

DESIGN ... Design Number

The design number allows to identify different hardware designs of the INCA-D by software.

01_H: Version 1.1

(all other codes reserved)

17.6.5 SRES - Software Reset Register

Value after reset: 00_H

	7							0	
SRES	RES_ CI	0	0	0	RES_ DCH	RES_ IOM	RES_ TR	0	WR (64)

RES_xx ... Reset Functional Block xx

In addition to the regular software and hardware resets, a reset can be activated on the functional block C/I-handler, D-channel, IOM handler, DASL-transceiver.

Setting one of these bits to “1” causes the corresponding block to be reset for a duration of 4 BCL clock cycles. The bits are automatically reset to “0” again.

17.6.6 IOM Handler (Transfer Units)

17.6.6.1 ITRDU - IOM Transfer Unit, Read/Write Register DU

Value after reset: 00_H

	7							0	
ITRDU									RD/WR (90_H)

This register holds the 8bit value from the IOM bus DU line. Compare with Section 17.1.5.

17.6.6.2 ITRDD - IOM Transfer Unit, Read/Write Register DD

Value after reset: 00_H

	7							0	
ITRDD									RD/WR (91_H)

This register holds the 8bit value from the IOM bus DD line. Compare with Section 17.1.5.

17.6.6.3 ITRICV - IOM Transfer Unit, Idle Code Value for transfer unit 0

Value after reset: FF_H

	7							0	
ITRICV									RD/WR (92_H)

This register holds the bit pattern to which the received data of transfer unit 0 is being compared, if ITR_MSK0.CED=1. For further desrciption refer to Section 17.1.6.

17.6.6.4 ITR_CR - IOM Transfer Control Register Register

ITR_CR (94_H) SFR **Reset Value: 00 00_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ITR FRIR	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**EN7**	**EN6**	**EN5**	**EN4**	**EN3**	**EN2**	**EN1**	**EN0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

ENx: Enable Transfer

1: The transfer is enabled.

0: The transfer is disabled

ITRFRIR: Interrupt Transfer Unit Frame Interrupt request

1: No new IOM frame has been loaded, interrupt requested

0: No interrupt requested

17.6.6.5 ITR_MSK0 - IOM Transfer Time Slot and Data Port Selection Register 0

ITR_MSK0 (96_H) SFR **Reset Value: 00 00_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR	**PRT**	**CED**	**0**	**TS11**	**TS10**	**TS9**	**TS8**	**TS7**	**TS6**	**Ts5**	**TS4**	**TS3**	**TS2**	**TS1**	**TS0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

This register determines the time slot and the data port on the IOM-2 interface for the data access from or to the memory for transfer channel 0. The register bank start address is ’00 $E09A_H$’

TSx ... Time Slot Selection

0:The data octet in timeslot x of the IOM-2 bus should not be accessed.
1:The data octet in timeslot x of the IOM-2 bus should be accessed.

’Accessed’ means, data were read when transfers from IOM to memory are selected (DIR=’1) or data have to be written to the IOM-2 bus when transfers from memory to IOM are selected (DIR=’0).

CED... Compare Enable

0:The comparison of the actual IOM value with contents of register ITRICV is disabled
1:The Comparison of the actual IOM value with contents of register ITRICV is enabled

PRT... Port Selection

0:Transfers refer to the data upstream line (DU) .
1:Transfers refersto the data downstream line (DD)

DIR ... Direction Selection

0:Selected data have to be written to selected port (DD/DU)
1:Selected data have to be read from selected port (DD/DU)

17.6.6.6 ITR_MSKx - IOM Transfer Time Slot and Data Port Selection Register 1 - 7

ITR_MSKx (98_H) SFR **Reset Value: 00 00_H**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR	**PRT**	**0**	**0**	**TS11**	**TS10**	**TS9**	**TS8**	**TS7**	**TS6**	**Ts5**	**TS4**	**TS3**	**TS2**	**TS1**	**TS0**
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

These registers determine the time slots and the data ports on the IOM-2 interface for the data access from or to the memory for the transfer channels 1 - 7. The register bank start address is '00 $E09A_H$'

TSx ... Time Slot Selection

0:The data octet in timeslot x of the IOM-2 bus should not be accessed.
1:The data octet in timeslot x of the IOM-2 bus should be accessed.

'Accessed' means, data were read when transfers from IOM to memory are selected (DIR='1) or data have to be written to the IOM-2 bus when transfers from memory to IOM are selected (DIR='0).

PRT... Port Selection

0:Transfers refer to the data upstream line (DU) .
1:Transfers refersto the data downstream line (DD)

DIR ... Direction Selection

0:Selected data have to be written to selected port (DD/DU)
1:Selected data have to be read from selected port (DD/DU)

17.6.6.7 IWSR - IOM waitstates register

IWSR ($B0_H$) **XBUS**-SFR **Reset Value: $FFFF_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RE	***reserved***			**ASA-WS**				**G-WS**				**HF-WS**			
rw	-	-	-	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

The number of waitstates for some of the IOM handler modules has to be programmed by this register.

HF-WS ... Wait states for HDLC FIFOs

Wait state register to access the dual port RAM in the HDLC FIFO

G-WS ... General Wait states

Wait states required for all registers inside the IOM handler except the HDLC FIFOs

ASA-WS ... Wait states for All-Slot-Access block (IOM transfer unit)

1111 : 15 waitstates
...
...
0000 : No waitstates

RE ... Ready Enable

0:Ready bit will not be asserted

Note: AFTER setting RE to '0' the READY in the XBCON2 register has to be disabled

1:Ready bit will be asserted

*Note: F**or CPU frequencies between 25 - 32 MHz there should be 1 wait state to access the HDLC FIFOs. This means IWSR = 8001$_H$.***

18 Line Transceiver

18.1 Wiring Configurations

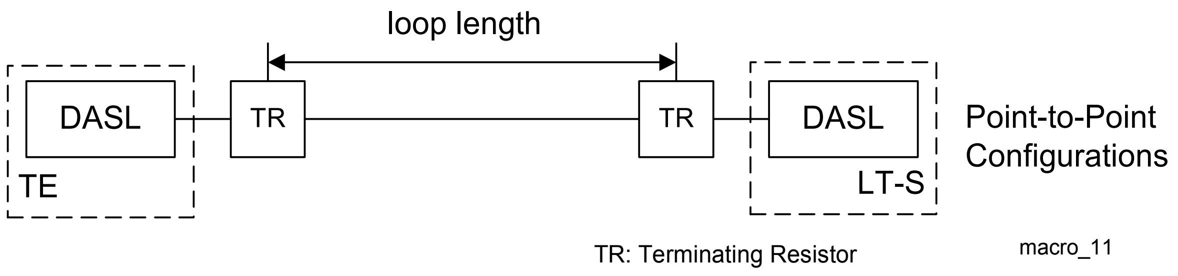


Figure 2 Wiring Configurations forDASL Interface

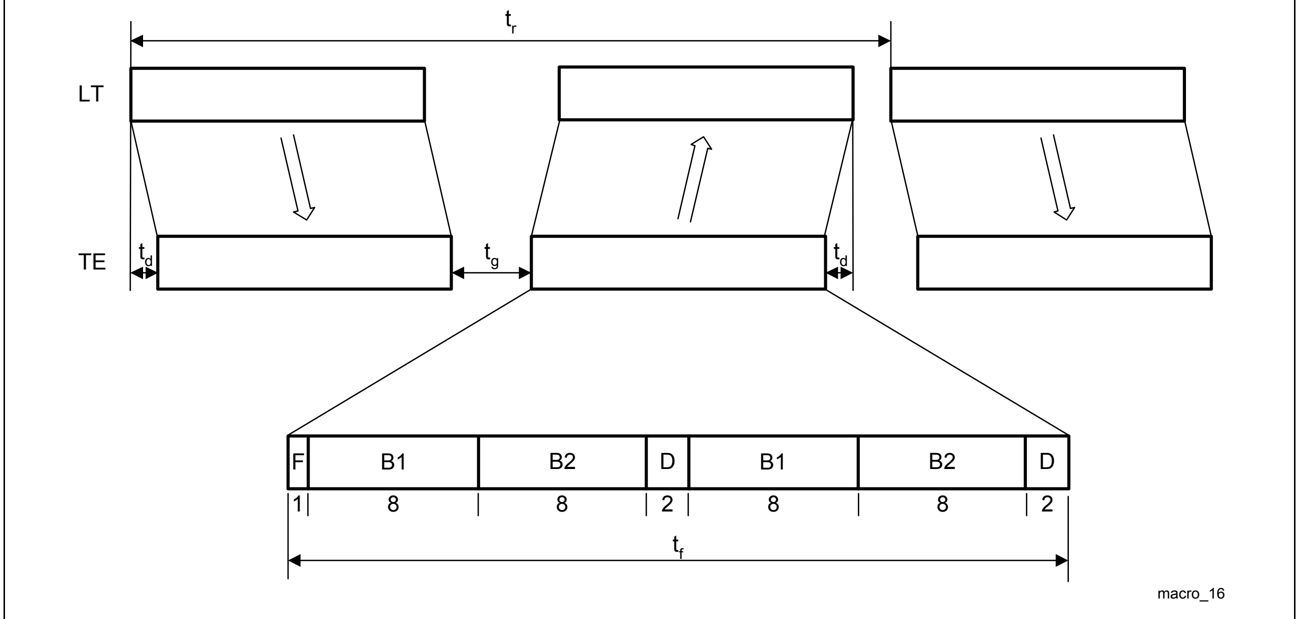
18.2 Line Coding, Frame Structure of the DASL interfacee

Burst Frame

Figure 18-1 demonstrates the general principles of the line interface communication scheme. A frame transmitted by the exchange (LT) is received by the terminal equipment (TE) after a line propagation delay. The terminal equipment waits the minimum guard time (t_g = 15.625 µs) while the line clears. It then transmits a frame to the exchange. The exchange will begin a transmission every 250 µs (known as the burst repetition period). Within a burst, the data rate is 384 kbit/s.

One frame contains the framing bit (F) and the user channels (2B + D). It can readily be seen that in the 250-µs burst repetition period, 4 D-bits, 16 B1-bits and 16 B2-bits are transferred in each direction. This gives an effective full duplex data rate of 16 kbit/s for the D-channel and 64 kbit/s for each B-channel.

AMI-coding is used for the line interface. Logical ‘0s’ correspond to a neutral level, logical ‘1s’ are coded as alternating positive and negative pulses (i.e. with a 100 % pulse width).



t_r: repetition period	(96 bits, 250 µs)
t_d: line delay	
t_g: guard time	(6 bits, 15.625 µs)
t_f: frame size	(37 bits, 96.35 µs)

Figure 18-1 Line Interface Structure

18.3 Data Transfer and Delay between IBUS and DASL Interface

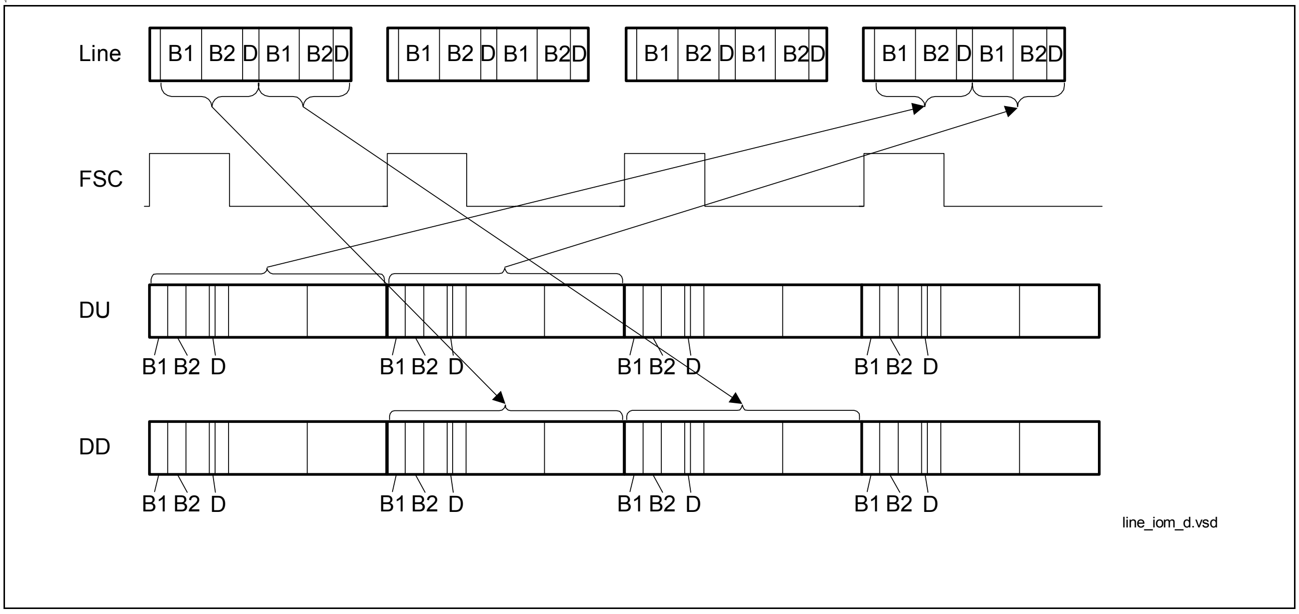


Figure 18-2 Data Delay between IBUS and Line Interface

The IBUS-interface D- and B-channels are used to convey the 16-kbit/s signalling channel and the two 64-kbit/s user channels in both directions. However, the transceiver only transfers the data transparently in the activated state (incl. analog loop activated) while the data are set to ‘1’ in any non activated state.

18.4 Control of Layer 1 / State Machine

The layer-1 activation/ deactivation can be controlled by an internal statemachine via the IBUS C/I0 channel or by software via the UBUS interface directly. In the default state the internal layer-1 statemachine of the INCA-D is used.

By setting the L1SW bit in the TR_CONF0 register the internal statemachine can be disabled and the layer-1 transmit commands, which are normally generated by the internal statemachine can be written directly into the TR_CMD register or the received status read out from the TR_STA register, respectively. The INCA-D layer-1 control flow is shown in **figure 18-3**.

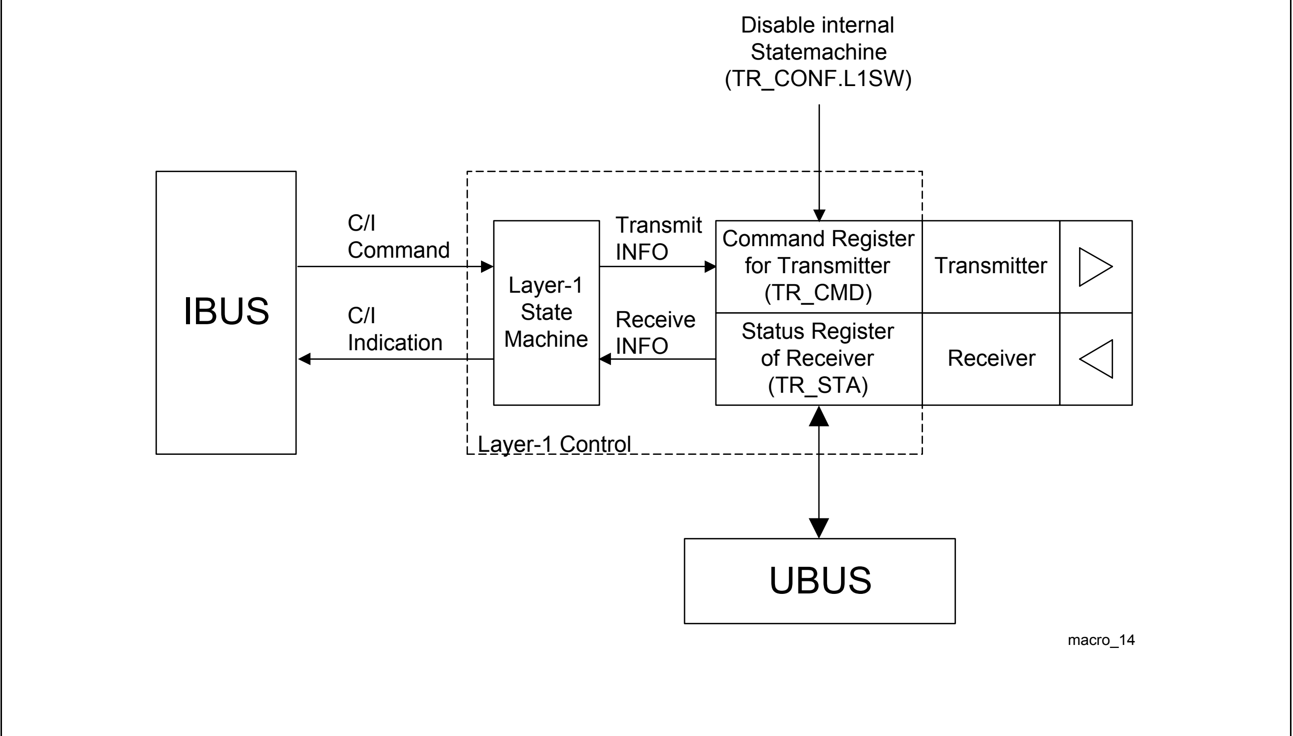

Disable internal Statemachine (TR_CONF.L1SW)
IBUS
C/I Command
C/I Indication
Layer-1 State Machine
Transmit INFO
Receive INFO
Command Register for Transmitter (TR_CMD)
Status Register of Receiver (TR_STA)
Transmitter
Receiver
Layer-1 Control
UBUS
macro_14

Figure 18-3 Layer-1 Control

In the following sections the layer-1 control by the INCA-D statemachine will be described.

The layer-1 functions are controlled by commands issued via the C/I channel 0 of the IBUS interface to the layer-1 to trigger certain procedures, such as activation/ deactivation, switching of test loops and transmission of special pulse patterns.

Responses from layer-1 are obtained by indications via the C/I channel 0 of the IBUS interface.

The statemachine includes all information relevant to the user. The state diagram notation is given in **Figure 18-4**.

The information contained in the state diagrams are:

state name

- Signal received from the line interface (INFO)
- Signal transmitted to the line interface (INFO)
- C/I code received (commands)
- C/I code transmitted (indications)
- transition criteria

The transition criteria are grouped into:

- C/I commands
- Signals received from the line interface (INFOs)
- Reset

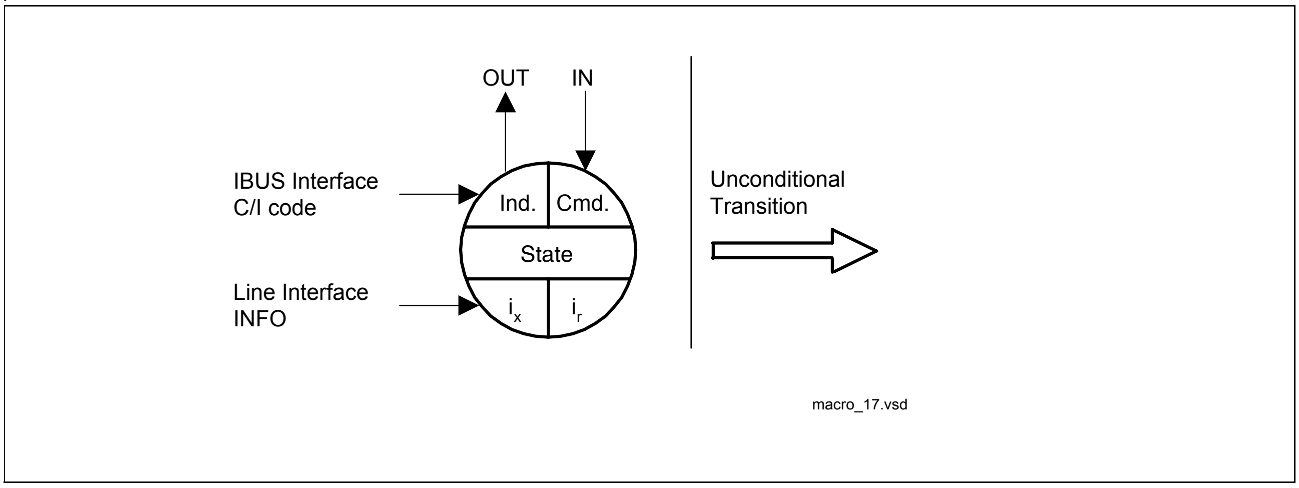


Figure 18-4 State Diagram Notation

As can be seen from the transition criteria, combinations of multiple conditions are possible as well. A "∗" stands for a logical AND combination. And a "+" indicates a logical OR combination.

The sections following the state diagram contain detailed information on all states and signals used.

Test Signals

- Send Single Pulses (SSP)

One pulse with a width of one bit period per frame with alternating polarity.

- Send Continuous Pulses (SCP)

Continuous pulses with a pulse width of one bit period.

External Layer-1 Statemachine

Instead of using the integrated layer-1 statemachine it is also possible to implement the layer-1 statemachine completely in software.

The internal layer-1 statemachine can be disabled by setting the L1SW bit in the TR_CONF0 register to '1'.

The transmitter is completely under control of the microcontroller via register TR_CMD.

The status of the receiver is stored in register TR_STA and has to be evaluated by the microcontroller. This register is updated continuously. If not masked a RIC interrupt is generated by any change of the register contents. The interrupt is cleared after a read access to this register.

Reset States

An active signal on the reset pin $\overline{\text{RSTIN}}$ brings the transceiver state machine to the reset state. The function of this reset event is identical to the C/I code RES concerning the state machine.

C/I Codes in Reset State

In the reset state the C/I code 0001 (RES) is valid. This state is entered either with a hardware reset ($\overline{\text{RSTIN}}$) or with the C/I code RES.

18.4.1 State Machine

Figure 18-5 shows the state transition diagram of the INCA-D state machine.

18.4.1.1 States

Reset, Pending Deactivation

State after reset or deactivation from the line interface by info 0. Note that no activation from the terminal side is possible starting from this state. A 'DI'-command has to be issued to enter the state 'Deactivated'.

Deactivated State

The lineinterface is deactivated and the clocks are deactivated 500 µs after entering this state and receiveing info 0 if the pin oscgo is set to "0". Activation is possible from the line interface and from the IBUS interface.

Power-Up

The line interface is deactivated (info 0 on the line) and the clocks are running.

Pending Activation

Upon the command Activation Request (AR) the INCA-D transmits the 2-kHz info 1w towards the network, waiting for info 2.

Level Detect (Resynchronisation)

During the first period of receiving info 2 or under severe disturbances on the line the receiver recognizes the receipt of a signal but is not (yet) synchronized.

Synchronized

The receiver is synchronized and detects info 2. It continues the activation procedure by transmission of info 1.

Activated

The receiver is synchronized and detects info 4. It concludes the activation procedure by transmission of info 3. All user channels are now conveyed transparently.

Analog Loop 3 Pending

Upon the C/I-command Activation Request Loop (ARL) the INCA-D loops back the transmitter to the receiver and activates by transmission of info 1. The receiver is not yet synchronized.

Analog Loop 3 Synchronized

After synchronization the transmitter continues by transmitting info 3.

Analog Loop 3 Activated

After recognition of the looped back info 3 the channels are looped back transparently.

Test Mode i

Test mode initiated by SCP-, SSP-commands.

Reset state

A C/I command reset (RES) forces the INCA-D to an idle state where INFO 0 is transmitted. Thus activation from the NT is not possible. Clocks are still supplied.

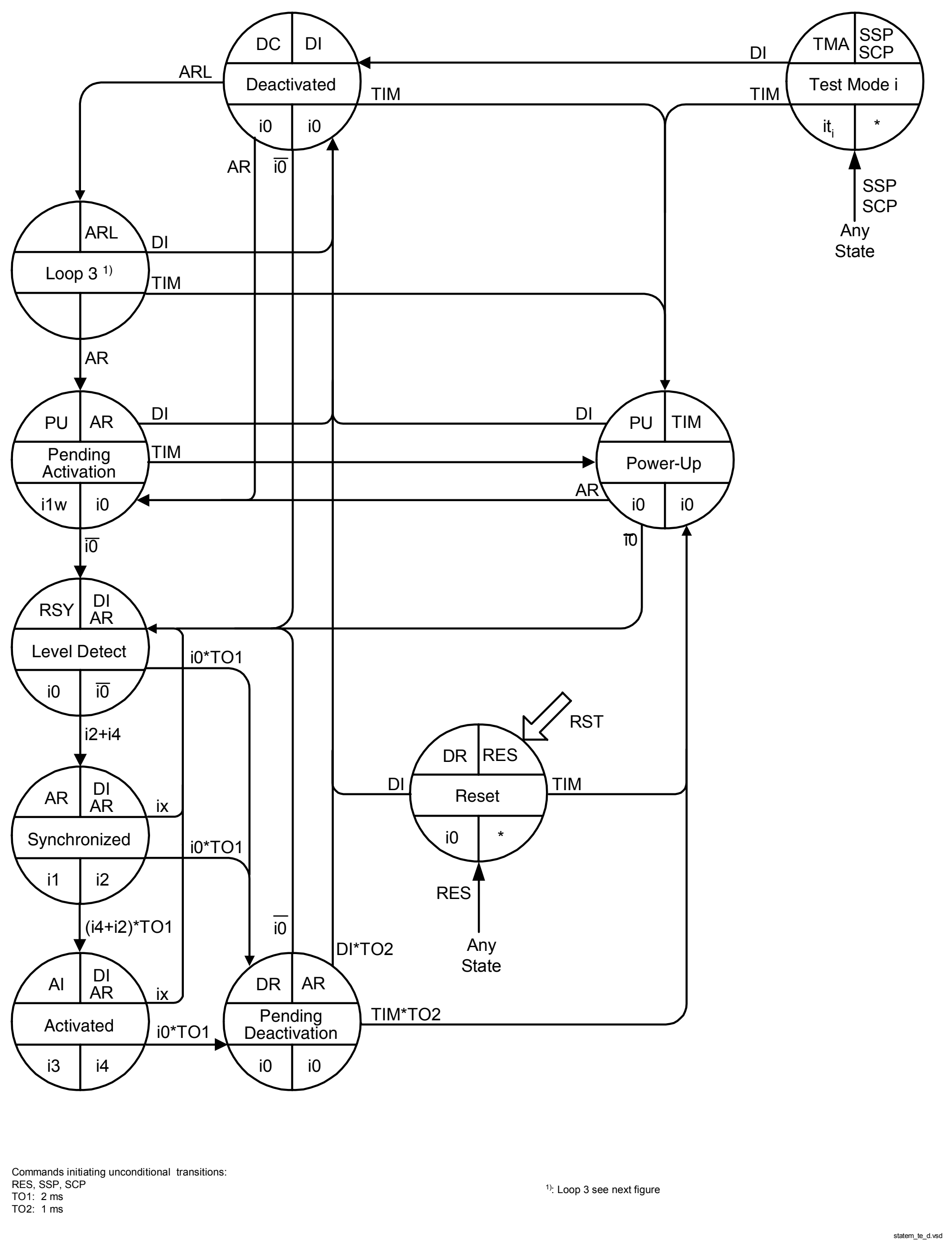


Figure 18-5 State Transition Diagram

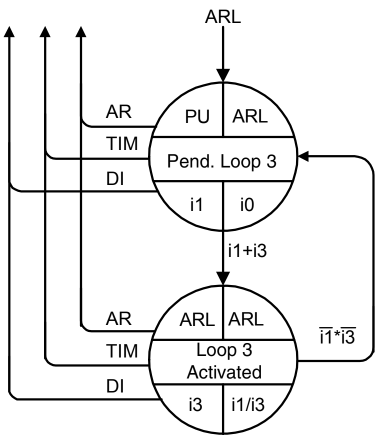


Figure 18-6 State Transition Diagram of the Loop 3 State

For a state description and C/I codes see UP interface, except Loop 2 and all corresponding C/I codes are not used in DASL mode.

18.4.1.2 C/I Commands

Command (Upstream)	Abbr.	Code	Remarks
Timing	TIM	0000	Layer-2 device requires clocks to be activated
Reset	RES	0001	Statemachine reset
Send Single Pulses	SSP	0010	
Send Continuous Pulses	SCP	0011	
Activate Request	AR	1000	
Activate Request Loop 3	ARL	1001	Local analog loop
Deactivation Indication	DI	1111	

18.4.1.3 C/I Indications

Indication (Downstream)	Abbr.	Code	Remarks
Deactivation Request	DR	0000	
Slip Detected	SLD	0011	
Power-Up	PU	0111	
Test Mode Acknowledge	TMA	0010	Acknowledge for both SSP and SCP
Resynchronization	RSY	0100	Receiver not synchronous
Activation Request	AR	1000	Receiver synchronized
Activation Request Loop 3	ARL	1001	Local loop synchronized
Activation Indication	AI	1100	
Activation Indication Loop 3	AIL	1101	Local loop activated
Deactivation Confirmation	DC	1111	Line- and if MODE1.CFS = ’1’ also clocks are powered down

18.4.1.4 Receive Infos on the Line (Downstream)

Name	Abbr.	Description
Info 0	i0	No signal on the line
Info 2	i2	4-kHz burst signal F=’1’ B1-, B2- and D- channels are scrambled ’1’s.
Info 4	i4	4-kHz burst signal F=’1’ B1-, B2- and D- channels are scrambled data.
Info X	ix	Any signal except info 2 or info 4

18.4.1.5 Transmit Infos on the Line (Upstream)

Name	Abbr.	Description
Info 0	i0	No signal on the line
Info 1w	i1w	Asynchronous wake signal 2-kHz burst rate F=’1’ B1-, B2- and D- channels are scrambled ’1’s.
Info 1	i1	4-kHz burst signal F=’1’ B1-, B2- and D- channels are scrambled ’1’s.
Info 3	i3	4-kHz burst signal F=’1’ B1-, B2- and D- channels are scrambled data.
Test Info 1	it_1	SCP
Test Info 2	it_2	SSP

18.4.2 C/I Codes

The following table summarizes the C/I codes. The code description can be found in the respective chapter above.

Code				DASL	
				Cmd	Ind
0	0	0	0	TIM	DR
0	0	0	1	RES	–
0	0	1	0	SSP	TMA
0	0	1	1	SCP	–
0	1	0	0	–	RSY
0	1	0	1	–	–
0	1	1	0	–	–
0	1	1	1	–	PU
1	0	0	0	AR	AR
1	0	0	1	ARL	ARL
1	0	1	0	–	–
1	0	1	1	–	–
1	1	0	0	–	AI
1	1	0	1	–	AIL
1	1	1	0	–	–
1	1	1	1	DI	DC

Table 18-1 C/I codes for DASL interface

18.5 Level Detection and Power Down

If TR_CONF0.LDD is set to ’0’ an activation initiated from the line side will have the consequence that the clock signals are provided automatically. If TR_CONF0.LDD is set to ’1’ the microcontroller has to take care of an interrupt caused by the level detect circuit (ISTATR.LD).

18.6 Transceiver Enable/Disable

The layer-1 part of the INCA-D can be enabled/disabled with the two bits TR_CONF0.DIS_TR and TR_CONF2.DIS_TX.
By default all layer-1 functions should be enabled (DIS_TR = ’0’, DIS_TX = ’0’). If DIS_TX=’1’ the transmit buffers are disabled. The receiver will monitor for incoming data in this configuration.
If the transceiver is disabled (DIS_TR = ’1’, DIS_TX = don’t care) all layer-1 functions are

disabled including the level detection circuit of the receiver. In this case the power consumption of the layer-1 is reduced to a minimum.

18.7 Test Functions

The INCA-D provides several test and diagnostic functions for the transceiver:

18.7.1 Line Transceiver Test

Two test loops allow the local or the remote test of the transceiver function.

- The local loop (test loop 3) which is activated by a C/I command ARL loops the transmit data of the transmitter to its receiver. The information of the IBUS input B- and D-channels is looped back to the output B- and D-channels.
- The remote loop (test loop 2) is activated by TR_CONF2.RLP. The data received from the line interface is looped back to the line interface. The D-channel information received from the line card is transparently forwarded to the output IBUS D-channel. The output B-channel information on IBUS is fixed to ‘FF’$_H$ while test loop 2 is active.

18.7.2 Test Signals on the Line Interface

Two kinds of test signals may be sent by the INCA-D:

- The single pulses are of alternating polarity at a frequency of the fundamental mode of 2 kHz (one pulse per frame). The corresponding C/I command is SSP (Send single pulses).
- The continuous pulses are pulses of alternating polarity at a frequency of the fundamental mode of 96 kHz. The corresponding C/I command is SCP (Send continuous pulses).

The test signals are AMI-coded.

18.8 Receive PLL (DPLL-Adjust Unit)

The receive DPLL adjusts to the center of every “1”-bit which follows on a “0”-bit. To avoid jitter adjustment is only done after three consecutive violations of the bit center condition in the same direction. This function can be disabled by TR_CONF1.RPLL_INTD.

The receive PLL adjusts in steps of a half or full oscillator period which is selected by the TR_CONF1.RPLL_ADJ.

18.9 Register Description

The listed addresses are actually address offeset to to the base address of 00’DE00’$_H$.

Note: Unused register bits always have an undefined reset value. The reset value for a whole register in hexadecimal notation does not apply to unused bits. Unused bits may be ’0’, ’1’, or ’-’. Only if indicated with ’-’ , a bit can be written as ’1’ or ’0’; in all other cases the predefined value must be written.

Note: The register SRES with the reset bit for the transceiver is described in Section 24.5, page 24-596.

Transceiver Configuration Registers

NAME	7	6	5	4	3	2	1	0	ADDR	R/W	RES
TR_ CONF0	DIS_ TR	0	0	0	L1SW	0	0	LDD	30_H	R/W	00_H
TR_ CONF1	RPLL_ INTD	1	1	0	0	1	0	0	31_H	R/W	64_H
TR_ CONF2	DIS_ TX	0	0	0	0	0	0	0	32_H	R/W	00_H
TR_STA	RINF		0	RDS	0	FSYN	0	0	33_H	R	00_H
TR_CMD	XINF			0	0	PD	LP_A	0	34_H	R/W	00_H
	reserved								35_H		
	reserved								36_H		
	reserved								37_H		
ISTATR	0	-	-	-	LD	RIC	0	0	38_H	R	00
MASKTR	0	1	1	1	LD	RIC	1	1	39_H	R/W	7F

18.9.1 TR_CONF0 - Transceiver Configuration Register

Value after reset: 00_H

	7							0	
TR_ CONF0	DIS_ TR	0	0	0	L1SW	0	0	LDD	RD/WR (30_H)

DIS_TR ... Disable Transceiver

0: All layer-1 functions are enabled.
1: All layer-1 functions are disabled. The HDLC controller can still operate via IOM-2. DCL and FSC pins become input (default after reset)

L1SW ... Enable Layer 1 State Machine in Software

0: Layer 1 state machine of the INCA-D is used
1: Layer 1 state machine is disabled. The functionality can be realized in software. The commands can be written in register TR_CMD and the status read from TR_STA.

LDD ... Level Detection Discard

0: Clock generation after detection of any signal on the line in the power down state
1: No clock generation after detection of any signal on the line in the power down state

Note: If an interrupt is generated by the internal level detect circuitry, the CPU has to set this bit to '0' for an activation of the line interface.

18.9.2 TR_CONF1 - Receiver Configuration Register

Value after reset: 64_H

	7							0	
TR_ CONF1	RPLL_ INTD	1	1	0	0	1	0	0	RD/WR (31_H)

RPLL_INTD ... Receive PLL Integrator Disable

0: The integrator function of the receive PLL is enabled
1: The integrator function of the receive PLL is disabled

18.9.3 TR_CONF2 - Transmitter Configuration Register

Value after reset: 00_H

	7							0	
TR_ CONF2	DIS_ TX	0	0	0	0	0	0	0	RD/WR (32_H)

DIS_TX **... Disable Line Driver**

The transmitter of the line transceiver can be disabled or enabled by setting DIS_TX. This can be used to make the analog loop (Loop3) transparent (DIS_TX = ’0’) or not (DIS_TX = ’1’).

0: Transmitter is enabled
1: Transmitter is disabled

18.9.4 TR_STA - Transceiver Status Register

Value after reset: 00_H

	7							0	
TR_ STA	RINF		0	RDS	0	FSYN	0	0	RD (33_H)

RINF **... Receiver INFO**

00: Received INFO 0
01: Received any signal except INFO 2 or INFO 4
10: Received INFO 2
11: Received INFO 4

RDS **... Running Digital Sum**

0: No code violation beyond F- or M-bit
1: At least one code violation beyond F- or M-bit received

FSYN **... Frame Synchronization State**

0: The line receiver has not synchronized or has lost synchronization to the framing bit F
1: The line receiver has synchronized to the framing bit F

18.9.5 TR_CMD - Transceiver Command Register

Value after reset: 00_H

	7					0	
TR_ CMD	XINF	0	0	PD	LP_A	0	RD/WR (34_H)

Normally the signals in this register are generated by the layer 1 state machine. If the internal layer 1 state machine is disabled (bit L1SW in TR_CONF = '1') this register can be written by the CPU.

XINF ... Transmit INFO

000: Transmit INFO 0
001: Transmit INFO 1W
010: Transmit INFO 1
011: Transmit INFO 3
100: Send continous 192 kHz pulses (Test Mode 2)
101: Send single 4 kHz pulses (Test Mode 1)
11x: reserved

PD ... Power Down

0: Transceiver in operational mode
1: Transceiver in power down mode. From the analog part only the level detector is active. Additionally no clocks are provided and the complete digital part of the transceiver is inactive if the CFS bit is set to '1'.

LP_A ... Loop Analog

The setting of this bit corresponds to the C/I command ARL.

0: Analog loop is open
1: Analog loop is closed

18.9.6 ISTATR - Interrupt Status Register Transceiver

Value after reset: 00_H

	7							0	
ISTATR	0	-	-	-	LD	RIC	0	0	RD (38_H)

For all interrupts in the ISTATR register following logical states are defined:

0: Interrupt is not activated
1: Interrupt is activated

LD ... Level Detection

Any receive signal has been detected on the line

RIC ... Receiver INFO Change

Any bit of register TR_STA has changed. This bit is reset by reading this register

18.9.7 MASKTR - Mask Transceiver Interrupt

Value after reset: $7F_H$

	7							0	
MASKTR	0	1	1	1	LD	RIC	1	1	RD/WR (39_H)

0: The transceiver interrupts LD and RIC are enabled

1: The transceiver interrupts LD and RIC are disabled

19 Analog Front End

The Analog Front End section of the INCA-D is the interface between the analog transducers and the digital signal processor.

A block diagram of the AFE is shown in figure 19-1.

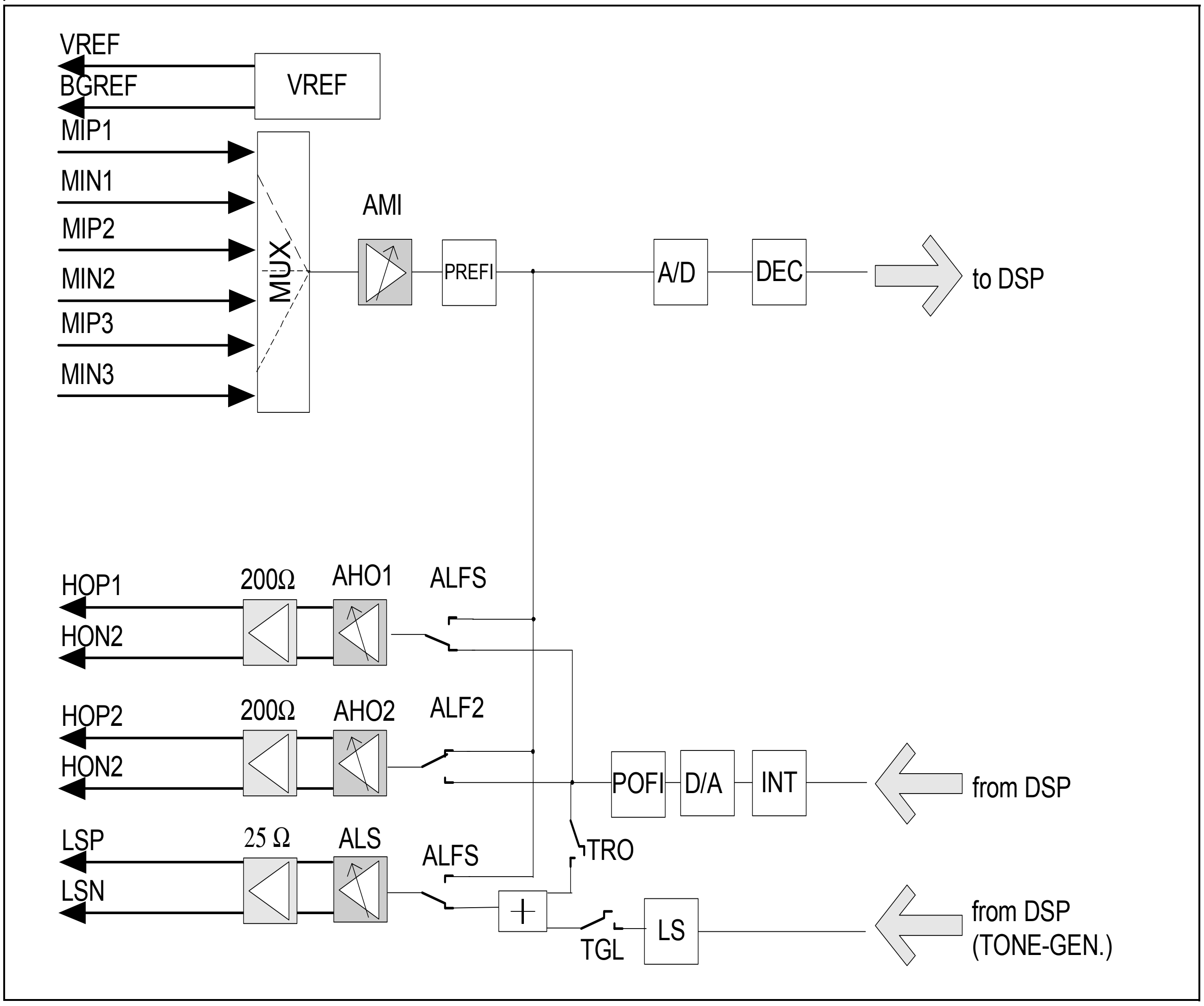


Figure 19-1 Analog Front End

19.1 Analog Front End (AFE) Description

In the transmit direction the AFE function is to amplify the transducer input signals (handset microphone, headset microphone and stand-alone microphone) and to convert them into digital signals. In the AFE receive section the incoming digital signal is converted to an analog signal which is output to an earpiece of an handset or headset and/or a loudspeaker.

Three differential inputs (MIP1/MIN1, MIP2/MIN2 and MIP3/MIN3) can be connected to the amplifier AMI via an analog input multiplexer (ATCR.AIMX). The programmable amplifier AMI (ATCR.MIC) provides a coarse gain adjustment range from 0...42dB in 6dB steps.

The maximum value of the programmable gain adjustment of the microphone amplifier with specified transmission characteristics (see Electrical Characteristics) is 36 dB for the differential input. Fine gain adjustment is performed in the digital domain via the programmable gain adjustment stage IG1 (see DSP section).

The fully differential outputs HOP1/HON1 or HOP2/HON2 respectively, connect the amplifier AHO1 or AHO2 to a handset or headset earpiece. The outputs support loads of 200 Ω.

Differential output LSP/LSN is provided for use with a 20 or 25 Ω loudspeaker. The driver capability for the loudspeaker is 100 or 80mW respectively (sine wave).

The programmable amplifiers AHO1, AHO2 and ALS (ARCR.HOC, ARCR.LSC) provide a coarse gain adjustment range from 11.5dB...-21.5dB (ALS) or 2.5dB...-21.5dB (AHO1, AHO2) respectively. The step size is for all three amplifiers 3dB. Fine gain adjustment is performed in the digital domain via the programmable adjustment stage IG2.

Each output of the differential amplifiers AHO1, AHO2 and ALS can be enabled separately (ACR.EHOP, EHON, ELSP, ELSN). The "loudhearing" mode is active when the conversation happens via the handset (with HOP1/HON1 and MIP1/MIN1 being active for example) and when additionally the loudspeaker is used with LSP/LSN being active. If loudhearing is active, the override-enable bit OVRE should be set. While the loudspeaker amplifier ALS amplifies with LSC normally, the ALS amplifies with OVLS if the speech detector SDR of the echo suppression unit detects speech on the line side. This technique allows to use a lower amplification of the ALS when speech is received in order to prevent acoustic echos.

The bandgap reference voltage is low-pass filtered via a capacity connected to pin BGREF.

The internal and external reference voltages are derived from this filtered bandgap reference voltage providing a good noise performance.

A square wave signal from the tone generator can be output directly to the loudspeaker amplifier. To output this signal to the loudspeaker, TRL must be set.

If the DSP does not have enough computational power left to provide the analog frontend with the appropriate data, the bit COV in the STATUS register is set (see Chapter 20.3 for details on the computational costs). The bit COV will be cleared by a write access to register AFECTL.

The registers for programming the analog frontend are shown in table 19-1.

Table 19-1 AFE Registers

Register	# of Bits	Name	Comment
AFECTL	1	EN	AFE enable
STATUS	1	COV	Codec overflow
ACR	1	TRO	Tone Ringing Only
AFECTL	1	TGL	1 bit Tone Generator output link
ACR	1	ALF2	Analog loop via AHO2
ACR	1	ALFS	Analog loop via AHO1 and ALS
ACR	1	EBIA	Enable bias (must be enabled first)
ACR	1	EADC	Enable A/D enable
ACR	1	EDAC	Enable D/A enable
ACR	1	EHOP2	Enable HOP2 amplifier
ACR	1	EHON2	Enable HON2 amplifier
ACR	1	EHOP1	Enable HOP1 amplifier
ACR	1	EHON1	Enable HON1 amplifier
ACR	1	ELSP	Enable LSP amplifier
ACR	1	ELSN	Enable LSN amplifier
ATCR	4	MIC	AMI amplifier control
ATCR	1	OVRE	Override enable (for loudhearing)
ATCR	2	AIMX	Analog input multiplexer
ARCR	4	HOC2	AHO2 amplifier control
ARCR	4	LSC	ALS amplifier control (OVRE = 0 or double talk)
ARCR	4	HOC1	AHO1 amplifier control
ARCR	4	OVLS	ALS amplifier control (OVRE = 1 and double talk)

19.2 External circuitry

The bandgap reference voltage input pin (BGREF) has to be connected to an external capacity of about 33 nF as described in **Figure 19-2**.

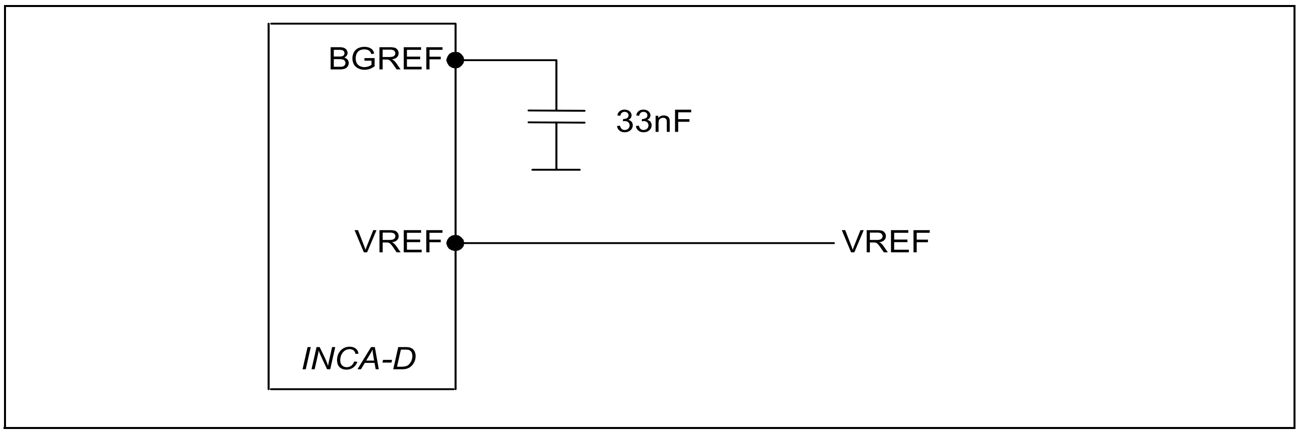


Figure 19-2 Capacity at BGREF

If the used microphones should be used in a 'single ended application', ie. one of the microphone pins is connected to GND, the following external circuitry is recommended, where one of INCA-D's differential microphone input pins is set to the VREF DC potential according to **Figure 19-3**.

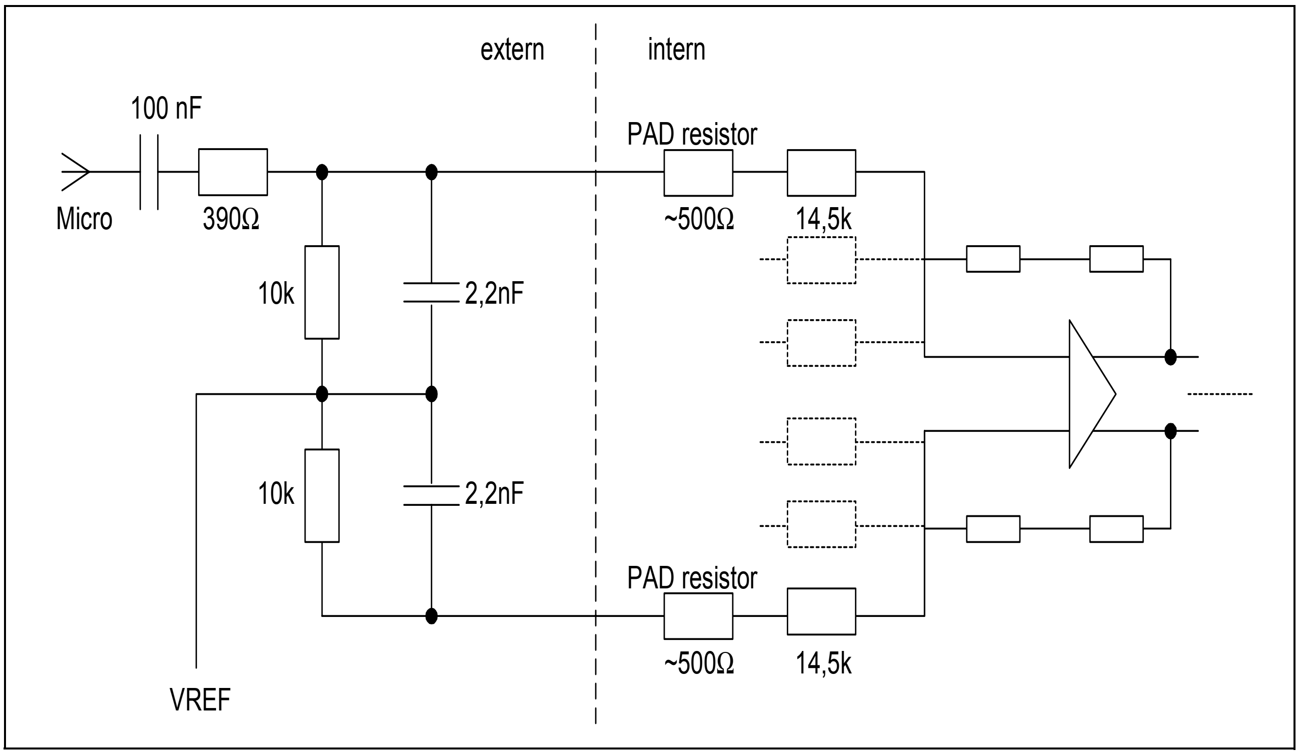


Figure 19-3 Single ended microphone connection

The dotted lines in **Figure 19-3** correspond to the remaining differential input lines.

If the noise level of the VREF ouput signal should be further improved, an RC lowpass filter can optionally be added as described in **Figure 19-4**.

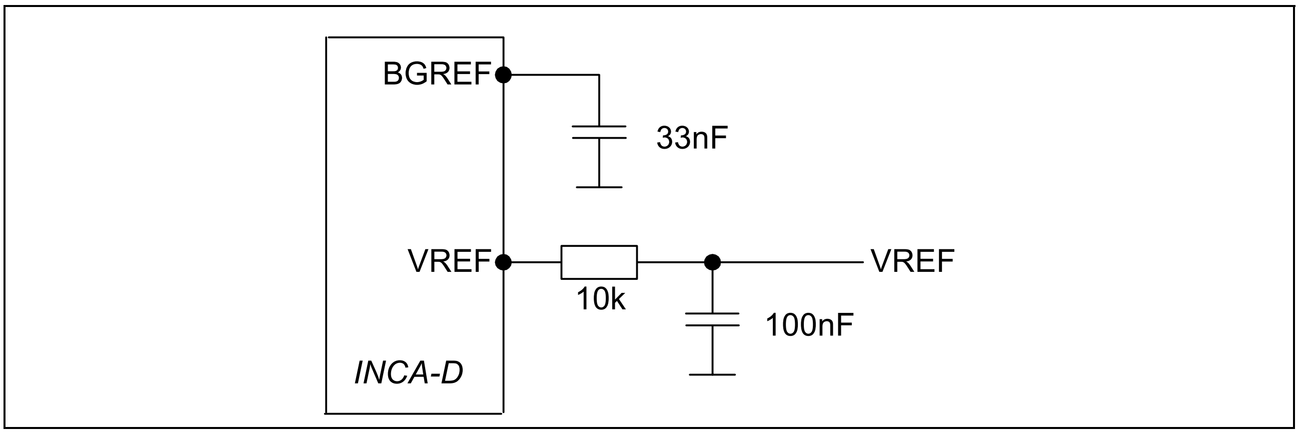


Figure 19-4 Improvement of VREF quality

20 Digital Signal Processor

20.1 Functional Units

The DSP of the INCA-D performs all signal processing tasks of the device. Each signal processing task is solved by a functional unit called module. The modules can be combined in any order with almost no restrictions to perform the given tasks.

Beside the Analog Front End the DSP offers two types of interfaces. First, the Data Interface (CDI) that handles the data accesses from and to the different time slots of the IOM-2 bus (see also **Chapter 17.1.7**). Secondly, the Parallel Interface to the DSP Domain (PIDD) which is actually the interface between CPU and DSP.

Figure 20-1 shows the modules available within the DSP of the INCA-D. They can be connected to each other or to external components through the 3 codec channels of the CDI or the microcontroller parallel interface.

Generally, all modules should be enabled only, **after** all parameters have been programmed.

For example, if the tone generator shoud be used for ringing, program all parameters of the tone generator after power up and if you want to start the ringer, enable the tone generator.

There are some exceptions, which are defined by the internal use of the parameters. This means that certain parameters can be changed during normal operation, without the necessity to disable the module first. For example in the speakerphone, the gains and attenuation SATT1.ATT can be setup on the fly.

Note: The (virtual)DSP registers keep their programmed values after the DSP has been set to power-down and subsequently to power-up, if no specific reset value has been defined for them. Otherwise, the defined reset value will be set, because after power-down the reset (initialization) routine is invoked.

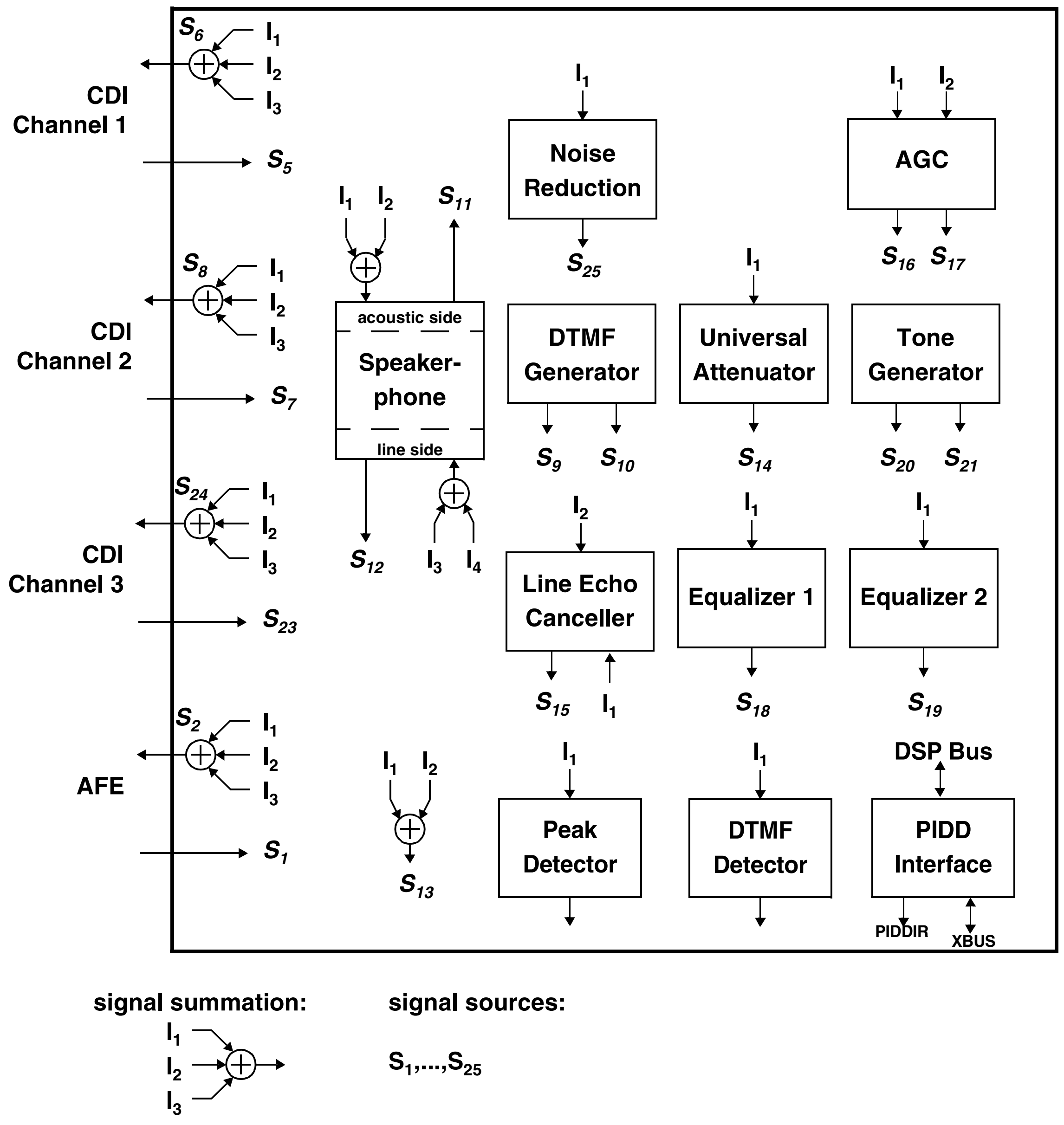


Figure 20-1 General Structure of the DSP

Each unit has one or more signal inputs (denoted by I). Most units have at least one signal output (denoted by S). Any input I can be connected to any signal output S. In addition to the signals shown in figure 20-1 there is also the signal S_0 (silence), which is useful at signal summation points. **Table 20-1** lists the available signals within the INCA-D according to their reference points. The 5-bit code for each signal in **Table 20-1** is the binary representation of the signal number.

Table 20-1 Signal Summary

Signal	Description
S_0	Silence
S_1	Handset-/ External Microphone input
S_2	Loudspeaker/Handset output
S_5	CDI input, Channel 1
S_6	CDI output, Channel 1
S_7	CDI input, Channel 2
S_8	CDI output, Channel 2
S_9	DTMF generator output
S_{10}	DTMF generator auxiliary output
S_{11}	Speakerphone output (acoustic side)
S_{12}	Speakerphone output (line side)
S_{13}	Universal Summation Output
S_{14}	Universal attenuator output
S_{15}	Line echo canceller output
S_{16}	Automatic gain control output (after gain stage)
S_{17}	Automatic gain control output (before gain stage)
S_{18}	Equalizer 1 output
S_{19}	Equalizer 2 output
S_{20}	Tone generator output 1
S_{21}	Tone generator output 2
S_{22}	reserved
S_{23}	CDI input, Channel 3
S_{24}	CDI output, Channel 3
S_{25}	NR output

The following sections describe the functional units in detail.

Figure 20-2 gives an example of how the units may be combined when a hands-free phone conversation is in progress. Units that are not needed are not shown. Unused inputs are connected to S_0 which provides silence (denoted as 0). The equalizers are used to improve the quality of the microphone and the loudspeaker.

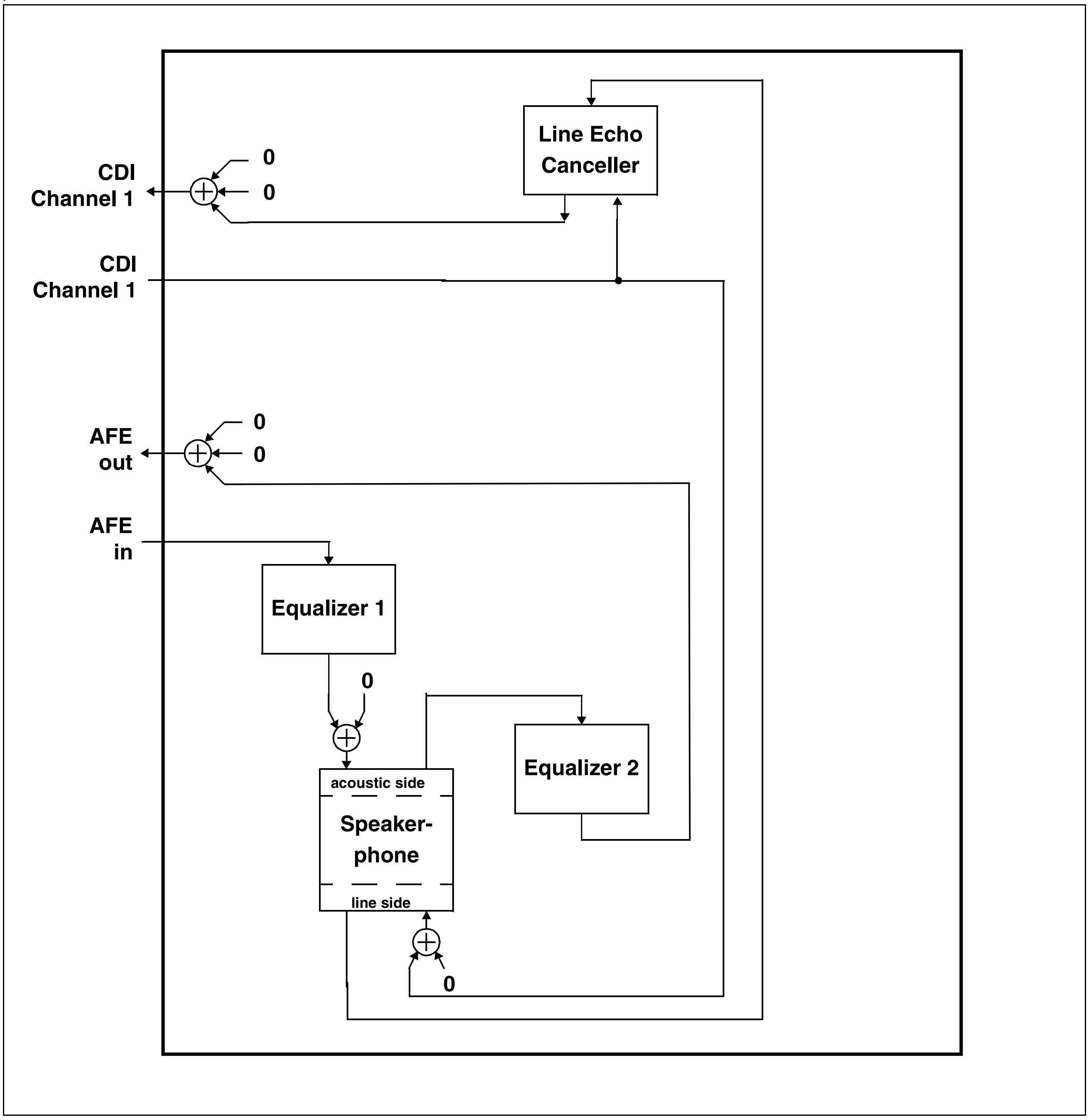


Figure 20-2 Functional Units - Speakerphone

20.1.1 Full Duplex Speakerphone

The speakerphone unit is attached to four signals (microphone, loudspeaker, line out and line in). The two input signals (microphone, line in) are preceded by a signal summation point.

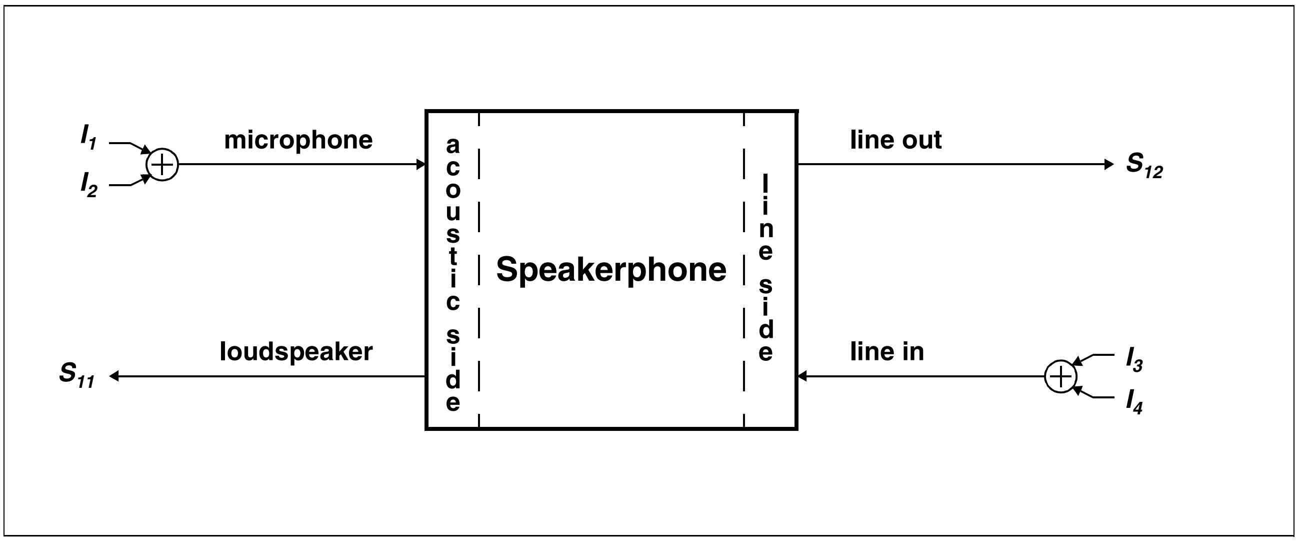


Figure 20-3 Speakerphone - Signal Connections

Internally, this unit can be divided into an echo cancellation unit and an echo suppression unit (figure 20-4). The echo cancellation unit provides the attenuation G_c while the echo suppression unit provides the attenuation G_s. The total attenuation ATT of the speakerphone is therefore ATT=G_C+G_S.

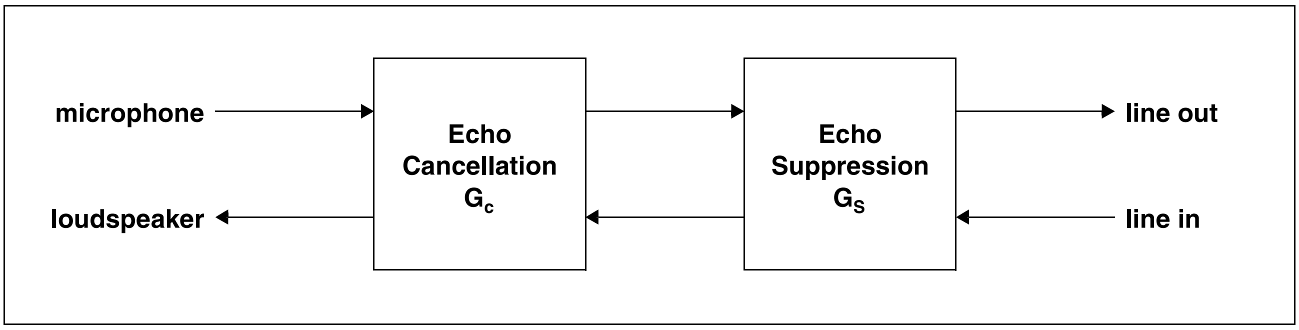


Figure 20-4 Speakerphone - Block Diagram

The echo cancellation unit estimates that part of the signal at the microphone that originates from the loudspeaker. This part is then subtracted from the signal at the microphone. This technique allows a full-duplex speakerphone. Furthermore, the echo cancellation unit has a built-in noise reduction unit for better sound quality at the line side.

The echo suppression unit attenuates the receive or transmit path dependent on what path is active. Without the echo cancellation unit and by using a high attenuation of the echo suppression unit, the echo suppression unit provides a half-duplex speakerphone. If the echo cancellation unit is active but cannot provide all of the required attenuation itself, the echo suppression unit can be used to provide additional attenuation.Then, the echo cancellation unit reports its current attenuation G_c to the echo suppression unit so that the echo suppression unit can adapt its attenuation G_s accordingly.

20.1.1.1 Echo Cancellation Unit

The echo cancellation unit has two operating modes: fullband and subband mode. Table 20-2 shows the basic differences of the two modes. The two modi are described in the sequel.

Table 20-2 Echo Cancellation Modes

	fullband mode	**subband mode**
max. G_c	20 dB	30 dB
echo length	16-96 ms	>50-159 ms
delay	1-2 ms	35 ms

The selection between the modes is performed with the parameter QU as summarized in table 20-3. The different modi are described in the sequel.

Table 20-3 AEC Mode (QU) Encoding

QU_2	**QU_1**	**QU_0**	**Acoustic Echo Cancellation Modes**
0	0	0	Echo cancellation disabled (half duplex)
0	0	1	reserved
0	1	0	Fullband mode one (similar to PSB 2170 Version 1.1)
0	1	1	Fullband mode two
1	0	0	Subband, reduced filter length (car kit application)
1	0	1	Subband, analog line mode
1	1	0	Subband, ISDN mode
1	1	1	Subband, enhanced mode

20.1.1.2 Echo Cancellation (Fullband Mode)

A simplified block diagram of the fullband echo cancellation unit is shown in figure 20-5.

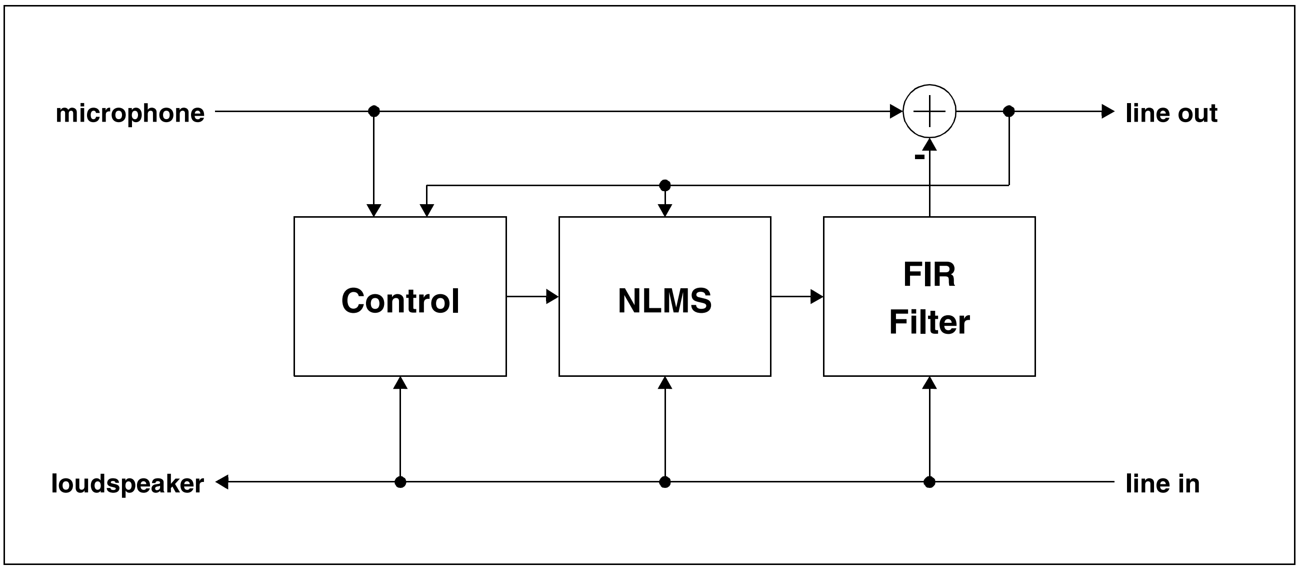


Figure 20-5 Echo Cancellation Unit (Fullband Mode) - Block Diagram

The echo cancellation unit consists of a finite impulse response filter (FIR) that models the expected acoustic echo, an NLMS based adaptation unit and a control unit. The expected echo is subtracted from the actual input signal from the microphone. If the model is exact and the echo does not exceed the length of the filter, then the echo can be cancelled completely. However, even if this ideal state can be achieved for one given moment, the acoustic echo usually changes over the time. Therefore the NLMS unit continuously adapts the coefficients of the FIR filter. This adaptation process is steered by the control unit. As an example, the adaptation is inhibited as long as double talk is detected by the control unit. Furthermore the control unit informs the echo suppression unit about the achieved echo return loss. The length of the FIR can be programmed by the parameters FBLEN. Furthermore, the number of FIR parameters that are adapted can be programmed by the parameter FBADA.

With the parameter QU, the echo cancellation unit can be set into two different fullband modes (section 20.1.1.1). The fullband mode 1 is similar to the mode provided in the PSB 2170 Version 1.1. In this mode, the parameter FBADA has no effect and the noise reduction cannot be enabled. The delay added in this mode is 0.25 ms. In the fullband mode two, the delay is shorter than 2 ms. If this delay is not acceptable, the delay can be reduced to less than 1 ms by setting the bit ERD. The performance of the fullband mode may suffer in this reduced mode but please note that enabling the noise reduction will add some additional delay (section 20.1.2).

In order to detect double talk, the remaining speech signal after the echo cancellation (signal after summation point in figure 20-5) is compared to the signal expected by the echo cancellation unit (after the signal summation point in figure 20-5). The difference of the energy of these signals is considered. If this energy is greater than allowed by the parameter AECDTM, double talk is detected.

When double talk is detected, the attenuation reported by the echo cancellation unit to the echo suppression unit is increased. This is because the echo is less noticeable anyway and any possible signal distortion from the echo suppression is reduced. The parameter AECDTR determines the maximal increase of the reported attenuation during double talk. The rate how fast the reported attenuation increase gets active after double talk detection or inactive after the end of double talk is determined by the parameters AECDTI and AECDTD, respectively. This is illustrated in figure 20-6 where the dashed line shows the actual attenuation while the solid line shows the reported attenuation.

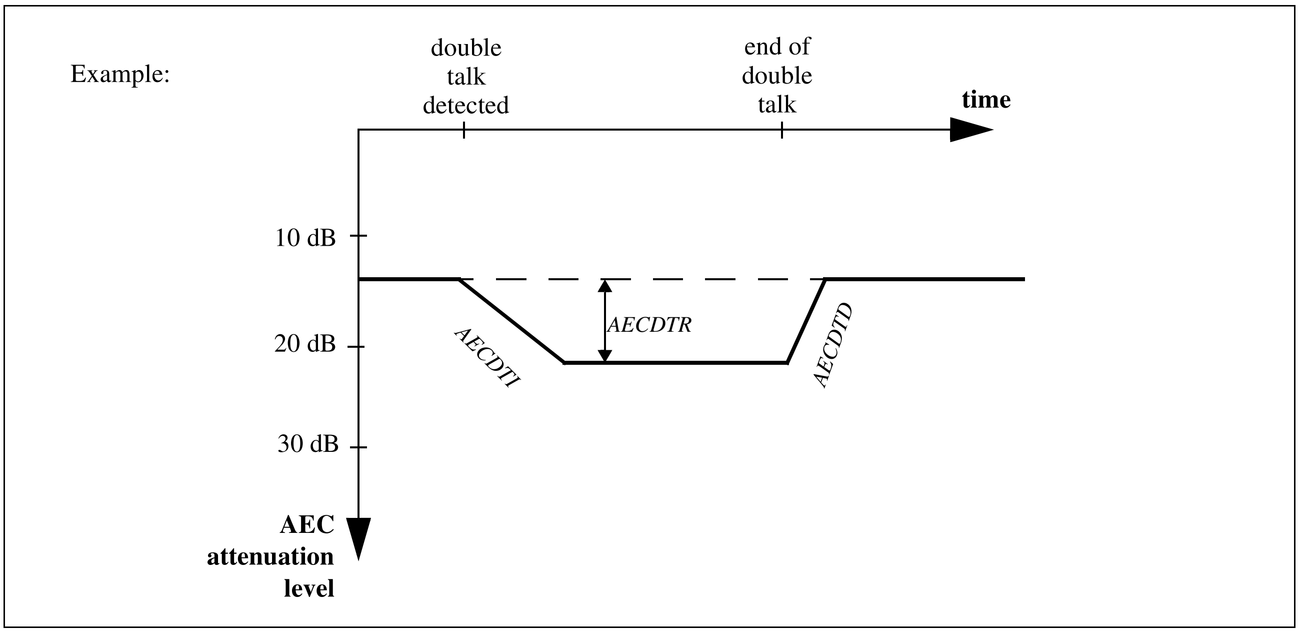


Figure 20-6 Echo Cancellation Unit - Double Talk Reduction

The attenuation of the echo suppression unit may be insufficient when the echo cancellation unit is not yet well adapted. In case of a significant change of the characteristics of the acoustics, the attenuation reported by the echo cancellation unit may be too high until it discovers that it has to adapt itself again. If, in addition, double talk reduction is in effect, then the echo suppression unit might not attenuate enough to avoid echoes. Therefore a maximal echo attenuation reported by the echo cancellation unit to the echo suppression unit can be programmed by the parameter AECLIM.

Table 20-4 shows the registers associated with the echo cancellation unit in fullband mode.

Table 20-4 Echo Cancellation Unit Registers

Register	# of Bits	Name	Comment
SCTL	3	QU	Determine AEC modes (see table 20-5)
SCTL	1	ERD	Enable reduced delay for new fullband mode.
SAELEN	10	FBLEN	Length of FIR filter.
SAEAW	9	FBADA	Length of adaption window.
SAEDTR	15	AECATT	AEC attenuation reduction during double-talk.
SAEEL	15	AECLIM	Maximum attenuation reported by the AEC unit.
SAEDTL	15	AECDTM	Minimum energy to detect double talk.
SAEDTI	15	AECDTI	How fast the attenuation reduction gets incremented.
SAEDTD	15	AECDTD	How fast the attenuation reduction gets decremented.

The length of the FIR filter for the echo compensation can be varied from 63 up to

- 639 taps (~80 ms) if fullband mode one is used
- 768 taps (~96 ms) if fullband mode two is used

The length of the adaptation window can be varied from 64 to 256 taps (8 ms to 32 ms).

20.1.1.3 Echo Cancellation (Subband Mode)

A simplified block diagram of the subband echo cancellation unit is shown in figure 20-7. The signal coming from the microphone and the signal from the line side are analyzed into several subbands. Then for each subband, the block diagram of figure 20-7 is identical to the block diagram of the fullband mode. After the echo cancellation of each subband, the subbands are synthesized. Finally and additionally to the fullband mode, in the subband mode an optional Wiener filter is provided.

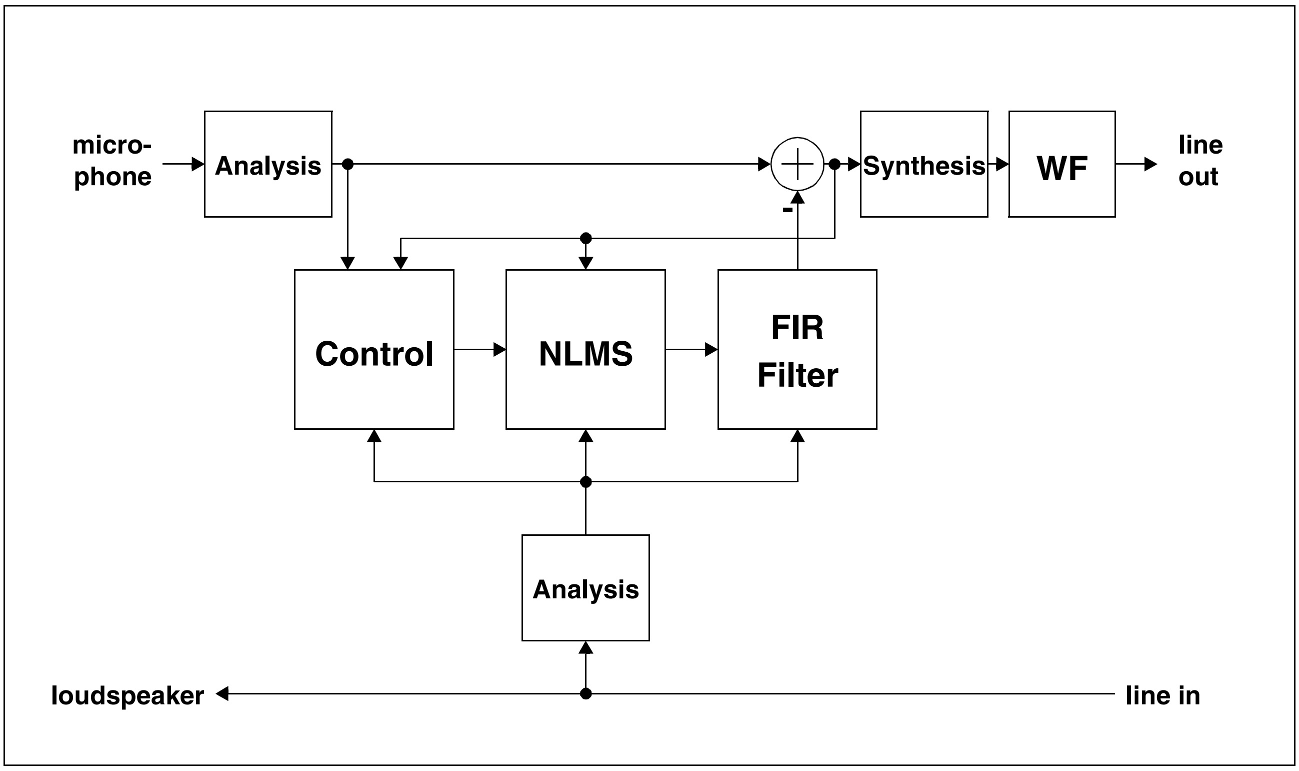


Figure 20-7 Echo Cancellation Unit (Subband Mode) - Block Diagram

The subband mode can be enabled in five different submodes. These submodes offer a trade-off between the maximum echo length and the functional units that can be run simultaneously. All units that cannot be run simultaneously must be disabled before the subband echo cancellation unit can be enabled. After the subband echo cancellation unit is disabled, the parameters for the affected units must be rewritten by the microcontroller.

For the optional Wiener filter, the maximum attenuation can be programmed with the parameter WFATT. If the Wiener filter is enabled, it is only active while there is no speech detected on the near side (microphone).

As shown in figure 20-7 the total attenuation provided by the speakerphone consists of the attenuation G_C (provided by the echo cancellation unit) and G_S (provided by the echo suppression unit). In subband mode the attenuation G_C is further split into G_A (provided by the adaptive filter) and G_W (provided by the Wiener filter).

If G_A already exceeds WFATT due to good adaptation then the Wiener filter is deactivated and G_C=G_A. Otherwise WFATT limits the attenuation G_W of the Wiener filter such that G_C=G_A+G_W never exceeds WFATT.

Table 20-5 shows the registers associated with the subband echo cancellation unit. The parameters AECATT, AECLIM, AECDTM, AECDTI and AECDTD have the same meaning as in fullband mode. Note that the reported echo loss and parameter AECLIM cover the complete echo cancellation unit including the Wiener filter.

Table 20-5 Subband Mode Registers

Register	# of Bits	Name	Comment
SCTL	3	QU	Determine AEC modes
SCTL	1	EWF	Wiener filter enable (subband only)
SAEWFL	15	WFATT	Wiener filter attenuation limit
SAEDTR	15	AECATT	AEC attenuation reduction during double-talk
SAEEL	15	AECLIM	Upper limit of the attenuation of the AEC
SAEDTL	15	AECDTM	Minimum energy to detect double talk
SAEDTI	15	AECDTI	How fast the attenuation gets incremented.
SAEDTD	15	AECDTD	How fast the attenuation gets decremented.

As shown in table 20-5, with the control bits QU different modes of the acoustic echo cancellation unit can be selected. In subband mode, four different modes are provided. With increasing number of QU, the considered echo length increases and so does the computational effort.

20.1.2 Noise Reduction

The INCA-D offers a noise reduction block built in the echo cancellation unit. This noise reduction suppresses noise before the signal is output to the line side. This makes conversation more pleasant for the person at the line side.

In fullband mode, the noise reduction can only be enabled if the new fullband mode is used. The noise reduction is performed after the echo cancellation and can thus be considered as an additional block after the signal summation point in figure 20-5. Note that the noise reduction adds about 1 ms delay to the signal. If the bit SCTL:ERD is set, then the delay for echo cancellation and noise reduction is still below 2 ms.

In subband mode, the noise suppression is performed for each subband. Figure 20-8 shows how the noise reduction block (called NR) matches in the echo cancellation unit. The noise reduction adds no delay to the signal.

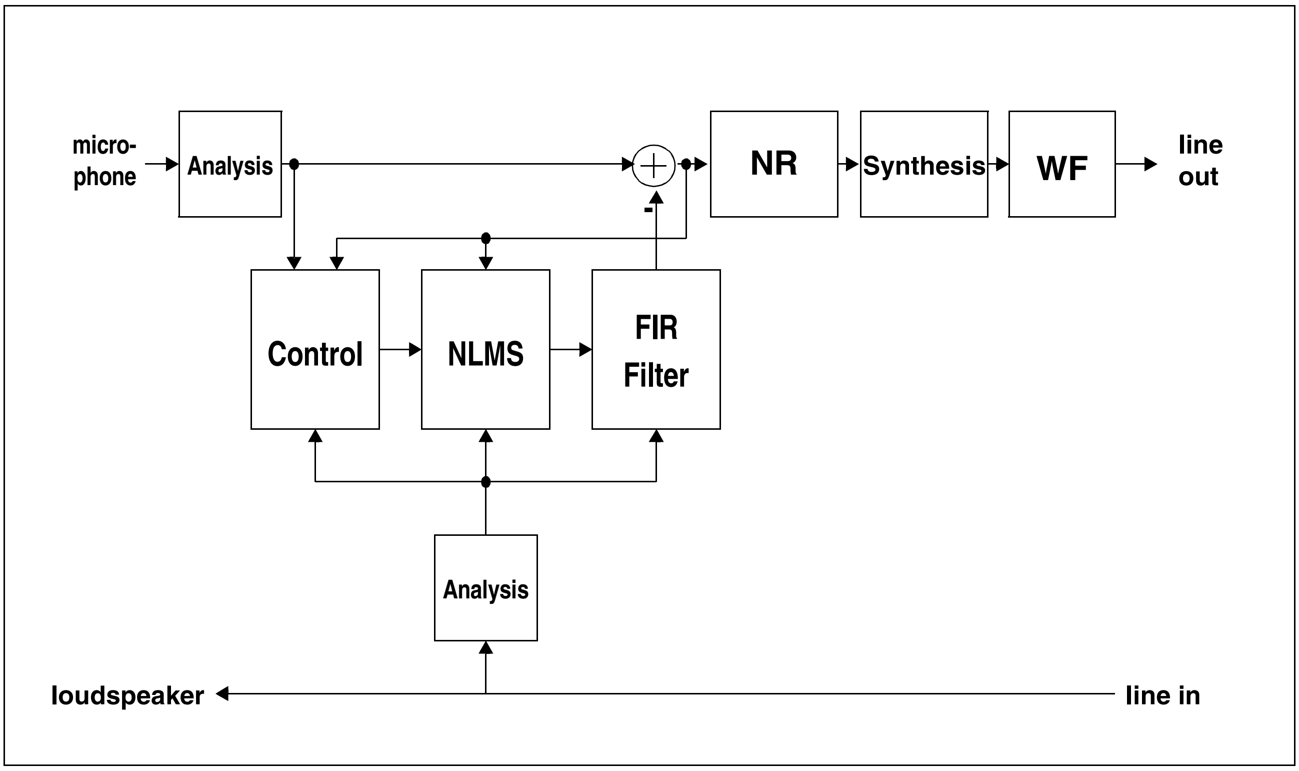


Figure 20-8 Echo Cancellation with Noise Reduction (Subband Mode)

The parameter NRATT determines the maximum attenuation provided by the noise reduction unit for frequencies with noise. Note that NRATT determines a upper limit. With decreasing level of the background noise, the attenuation of the noise reduction unit decreases as well.

In subband mode, if the Wiener filter is enabled and if no speech is detected at the microphone, the Wiener filter provides attenuation of the signal from the microphone. If noise is present at the microphone and thus the noise reduction unit changes the signal coming from the microphone, then the additional attenuation of the Wiener filter can cause modulations audible at the line side. Therefore, a coupling of the noise reduction unit and the Wiener filter is provided. In case no noise is present at the near end, the Wiener filter attenuates as programmed with the parameter SWATT:WFATT. In case significant noise is present at the near end, the Wiener filter only attenuates so much that the attenuation of the noise reduction plus the attenuation of the Wiener filter never exceeds NRATT. This is controlled with two parameters:

The parameter NRUP determines the noise energy at which the coupling between the noise reduction and the Wiener filter gets active. The parameter NRLOW determines the noise energy at which the coupling gets inactive. This value should be lower than NRUP in order to avoid frequent activation and deactivation, when the noise energy is around the level of NRUP.

Table 20-6 Noise Reduction Registers

Register	# of Bits	Name	Comment
SCTL	1	NR	Enable noise reduction
SNRATT	15	NRATT	Maximal attenuation of frequencies with noise.
SNRLNL	15	NRLOW	Lower limit to deactivate coupling.
SNRUNL	15	NRUP	Upper limit to activate coupling.

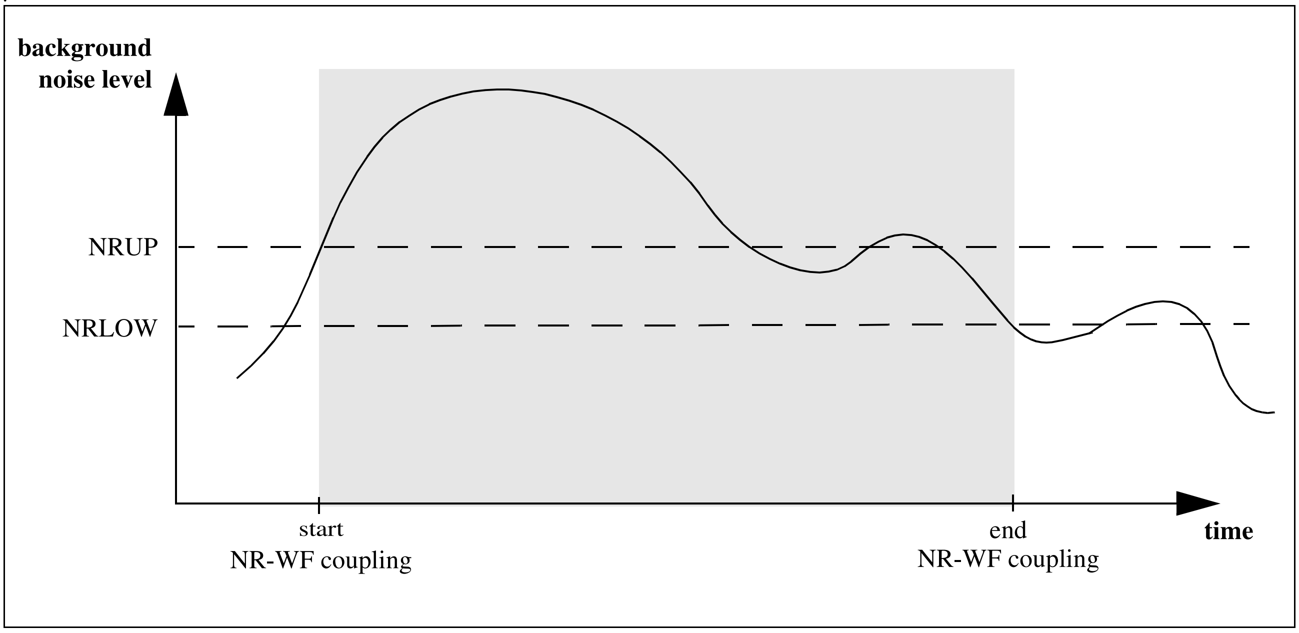


Figure 20-9 Coupling NR with WF

20.1.3 Echo Suppression

The echo suppression unit can be in one of three states:

- transmit state
- receive state
- idle state

In transmit state the microphone signal drives the line output while the line input is attenuated. In receive state the loudspeaker signal is driven by the line input while the microphone signal is attenuated. In idle state both signal paths are active with evenly distributed attenuation.

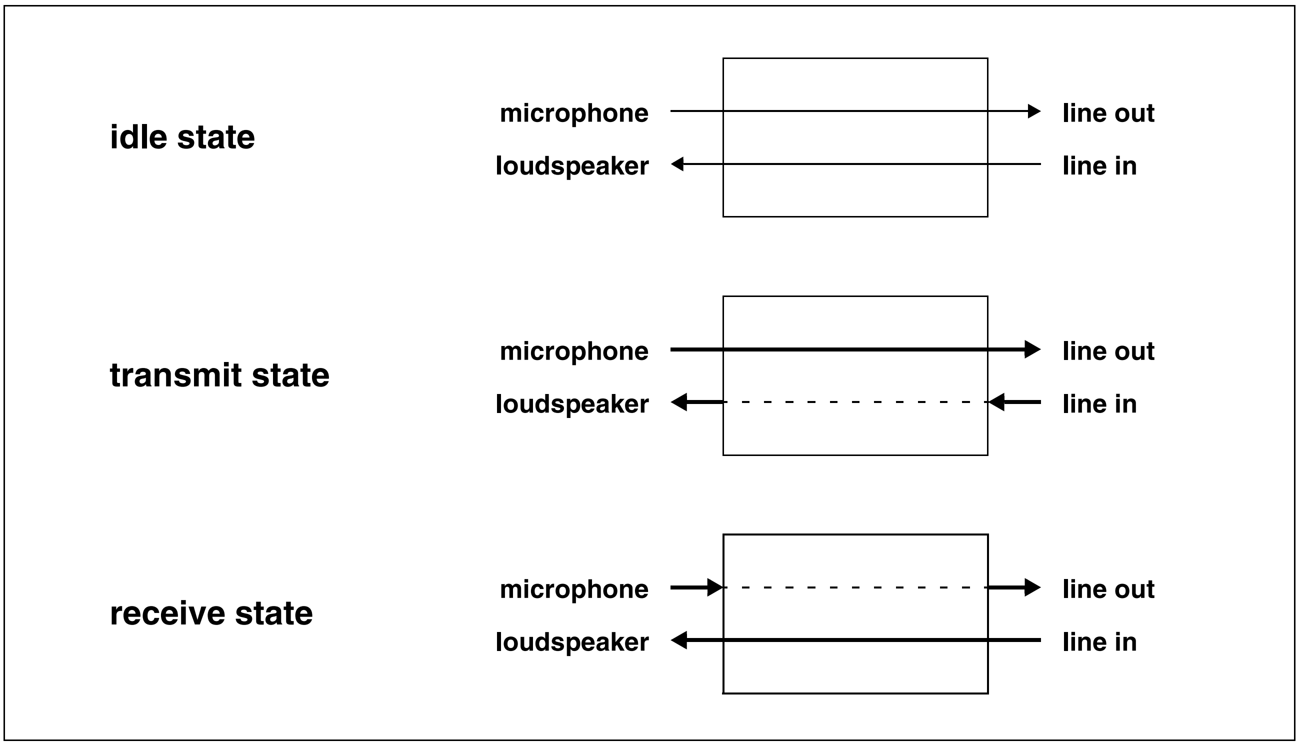


Figure 20-10 Echo Suppression Unit - States of Operation

Figure 20-11 shows the signal flow graph of the echo suppression unit in more detail.

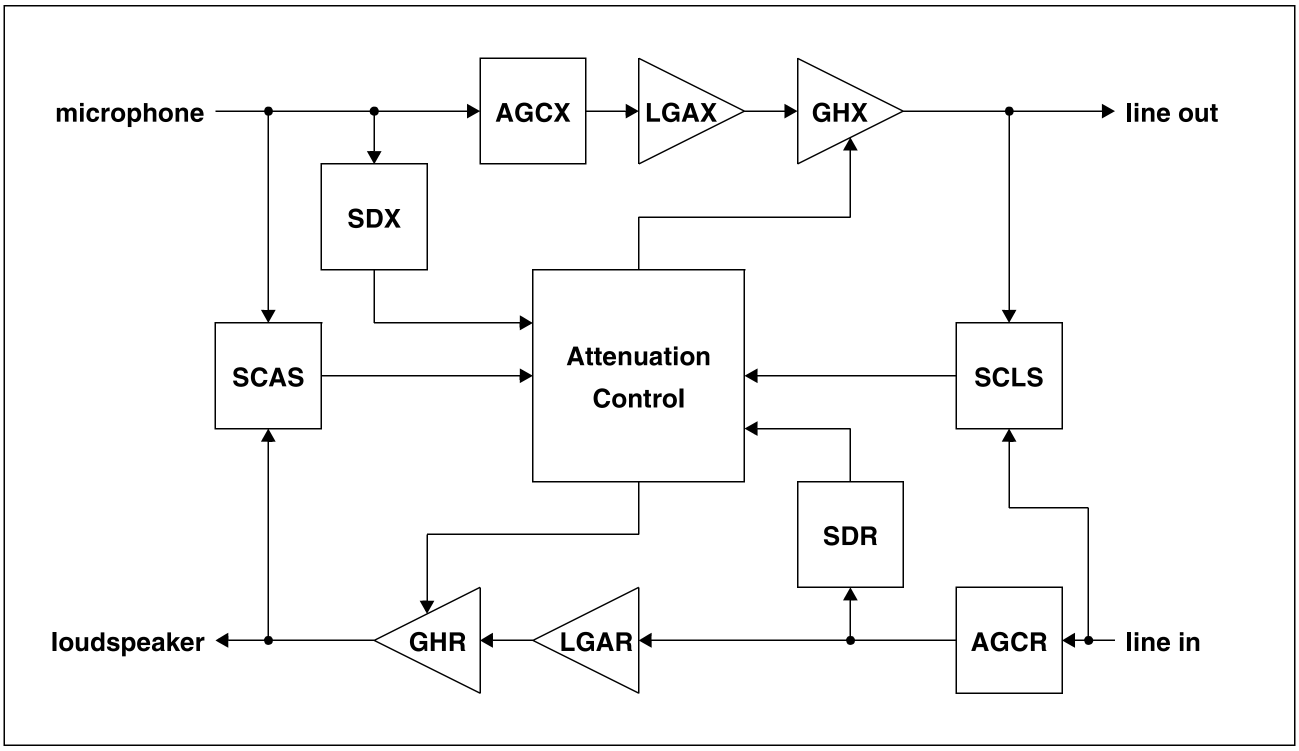


Figure 20-11 Echo Suppression Unit - Block Diagram

The Attenuation Control performs the switching between the three possible states by using the attenuation stages GHX and GHR. Actually, state switching is controlled by the speech comparators SCAS and SCLS and by the speech detectors SDX and SDR. The gain control units AGCX, AGCR, LGAX, and LGAR are used to achieve proper signal levels for each state.

All blocks are programmable. Thus, the telephone set can be optimized and adjusted to the particular geometrical and acoustical environment. The following sections discuss the blocks of the echo suppression unit in detail.

20.1.3.1 Speech Detector

For each signal source a speech detector (SDX, SDR) is available. The speech detectors are identical but can be programmed individually. Figure 20-12 shows the signal flow graph of a speech detector.

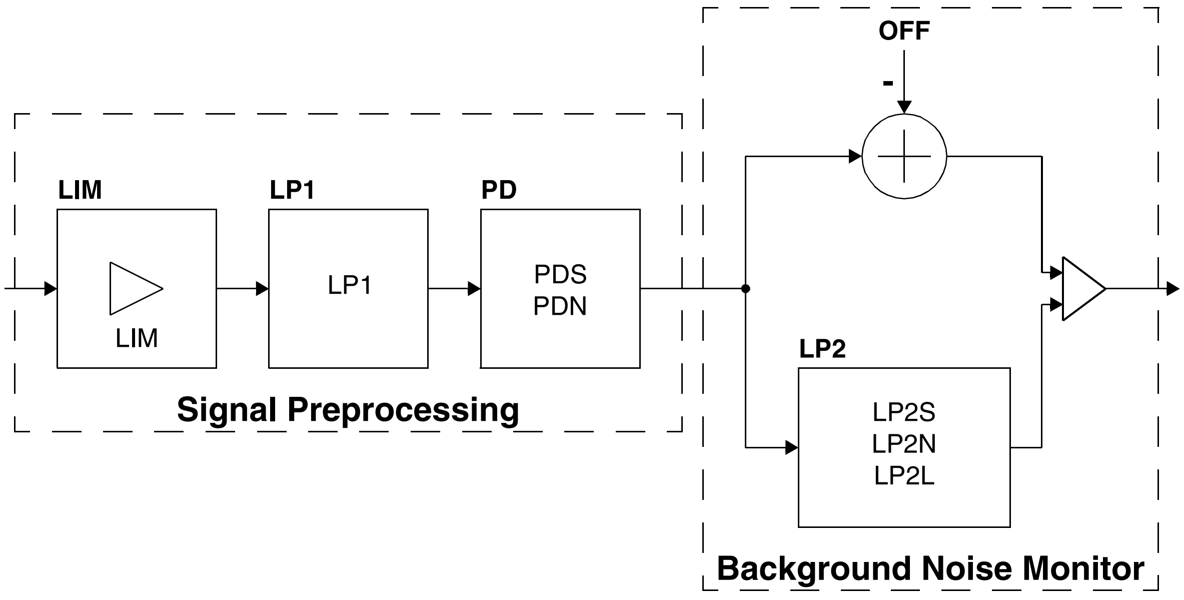


Figure 20-12 Speech Detector - Block Diagram

The first three units (LIM, LP1, PD) are used for preprocessing the signal while the actual speech detection is performed by the background noise monitor.

Background Noise Monitor

The tasks of the noise monitor are to differentiate voice signals from background noise, even if it exceeds the voice level, and to recognize voice signals without any delay. Therefore the Background Noise Monitor consists of the Low-Pass Filter 2 (LP2) and the offset in two separate branches. Basically it works on the burst-characteristic of the speech: voice signals consist of short peaks with high power (bursts). In contrast, background noise can be regarded approximately stationary from its average power.

Low-Pass Filter 2 provides different time constants for noise (non-detected speech) and speech. It determines the average of the noise reference level. In case of background noise the level at the output of LP2 is approximately the level of the input. As in the other branch an additional offset OFF is added to the signal, the comparator signals noise. At speech bursts the digital signals arriving at the comparator via the offset branch change faster than those via the LP2-branch. If the difference exceeds the offset OFF, the comparator signals speech. Therefore the output of the background noise monitor is a digital signal indicating speech (1) or noise (0).

A small fade constant (LP2N) enables fast settling of LP2 to the average noise level after the end of speech recognition. However, a too small time constant for LP2N can cause rapid charging to such a high level that after recognizing speech the danger of an unwanted switching back to noise exists. It is recommended to choose a large rising constant (LP2S) so that speech itself charges the LP2 very slowly. Generally, it is not recommended to choose an infinite LP2S because then approaching the noise level is disabled. During continuous speech or tones the LP2 will be charged until the limitation LP2L is reached. Then the value of LP2 is frozen until a break discharges the LP2. This limitation permits transmission of continuous tones and “music on hold”.

The offset stage represents the estimated difference between the speech signal and averaged noise.

Signal Preprocessing

As described in the preceding chapter, the background noise monitor is able to discriminate between speech and noise. In very short speech pauses e.g. between two words, however, it changes immediately to non-speech, which is equal to noise. Therefore a peak detection is required in front of the Noise Monitor.

The main task of the Peak Detector (PD) is to bridge the very short speech pauses during a monolog so that this time constant has to be long. Furthermore, the speech bursts are stored so that a sure speech detection is guaranteed. But if no speech is recognized the noise low-pass LP2 must be charged faster to the average noise level. In addition, the noise edges are to be smoothed. Therefore two time constants are necessary. As the peak detector is very sensitive to spikes, the low-pass LP1 filters the incoming signal containing noise in a way that main spikes are eliminated. Due to the programmable time constant it is possible to refuse high-energy sibilants and noise edges.

To compress the speech signals in their amplitudes and to ease the detection of speech, the signals have to be companded logarithmically. Hereby, the speech detector should not be influenced by the system noise which is always present but should discriminate between speech and background noise. The limitation of the logarithmic amplifier can be programmed via the parameter LIM. LIM is related to the maximum PCM level. A signal exceeding the limitation defined by LIM is getting amplified logarithmically, while very smooth system noise below is neglected. It should be the level of the minimum system noise which is always existing; in the transmit path the noise generated by the telephone circuitry itself and in receive direction the level of the first bit which is stable without any speech signal at the receive path. Table 20-7 shows the parameters for the speech detector.

Table 20-7 Speech Detector Parameters

Parameter	# of bytes	Range	Comment
LIM	1	0 to -95 dB	Limitation of log. amplifier
OFF	1	0 to 95 dB	Level offset up to detected noise
PDS	1	1 to 2000 ms	Peak decrement PD (speech)
PDN	1	1 to 2000 ms	Peak decrement PD (noise)
LP1	1	1 to 2000 ms	Time constant LP1
LP2S	1	2 to 250 s	Time constant LP2 (speech)
LP2N	1	1 to 2000 ms	Time constant LP2 (noise)
LP2L	1	0 to 95 dB	Maximum value of LP2

The input signal of the speech detector can be connected to either the input signal of the echo suppression unit (as shown for SDX in **Figure 20-10**) or the output of the associated AGC (as shown for SDR in **Figure 20-10**).

20.1.3.2 Speech Comparators (SC)

The echo suppression unit has two identical speech comparators (SCAS, SCLS). Each comparator can be programmed individually to accommodate the different system characteristics of the acoustic interface and the line interface. As SCAS and SCLS are identical, the following description holds for both SCAS and SCLS.

The SC has two input signals SX and SR, which map to microphone/loudspeaker for SCAS and line in/line out for SCLS.

The speech comparator decides whether the signal coming in on SR is only an echo from the signal outgoing on SX or a real speech activity. The result is then interpreted by the Attenuation Control of figure 20-11. In general, the SC works according to the following equation:

$$\text{if} \quad SX > SR + V \text{ then switch state}$$

Therefore, SCAS controls the switching to transmit state and SCLS controls the switching to receive state. Switching is done only if SX exceeds SR by at least the expected acoustic level enhancement V. This level enhancement is divided into two parts: G and GD. A block diagram of the SC is shown in figure 20-13.

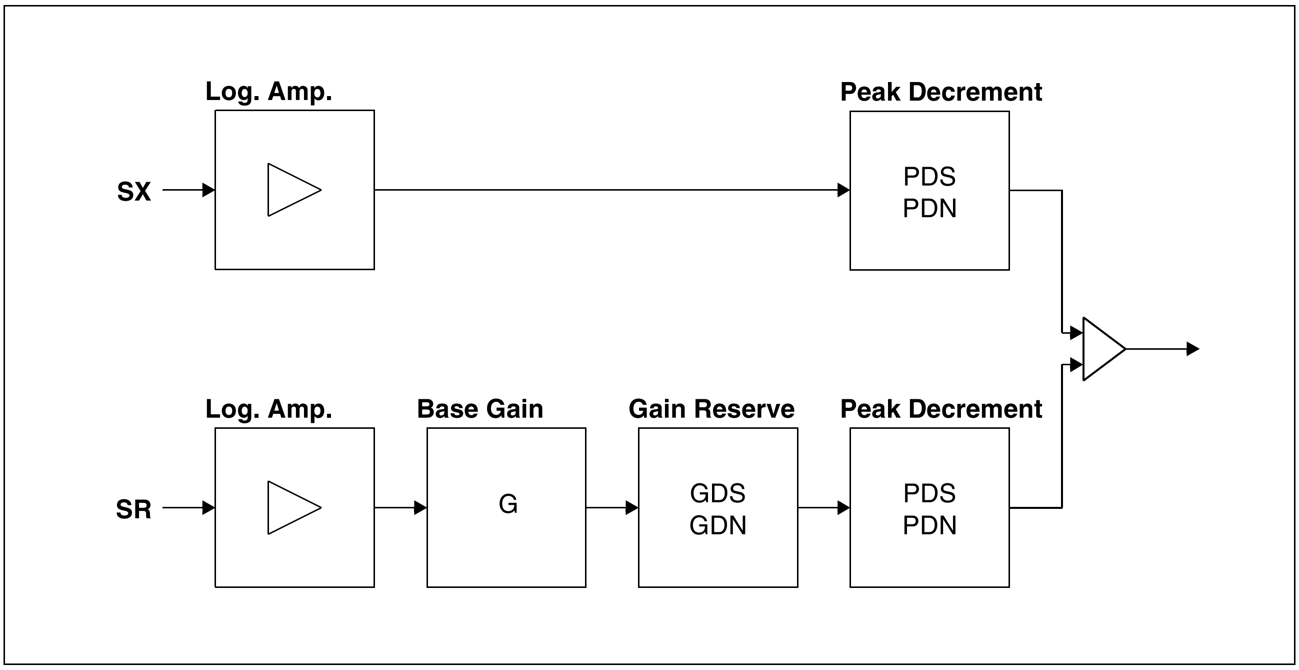


Figure 20-13 Speech Comparator - Block Diagram

At both inputs, logarithmic amplifiers compress the signal range. Hence, only logarithmic levels are on both paths and after the signals have been processed, logarithmic levels on both paths are compared.

The main task of the comparator is to control the echo. The internal coupling due to the direct sound and mechanical resonances is covered by G. The external coupling, mainly caused by the acoustic feedback, is controlled by GD/PD. An example for direct sound (1) and acoustic feedback (2) is illustrated in figure 20-14.

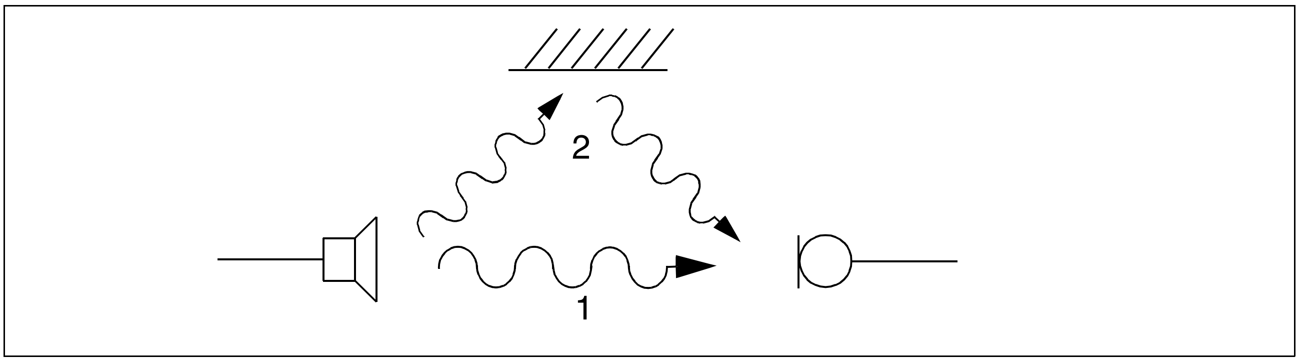


Figure 20-14 Speech Comparator - Acoustic Echoes

The base gain G corresponds to the terminal couplings of the complete telephone. Thus, G is the measured or calculated level enhancement between the receive and the transmit inputs of the SC.

To control the acoustic feedback two parameters are necessary: GD represents the actual reserve on the measured G. Together with the Peak Decrement (PD), the echo behavior at the acoustic side is modeled: After speech has ended there is a short time during which hard couplings through the mechanics and resonances and the direct echo are present. Till the end of that time (Δt), the level enhancement V must be at least equal to G to prevent clipping caused by these internal couplings. After that time (Δt), only the acoustic feedback is present. This coupling, however, is reduced by air attenuation. For this in general the longer the delay, the smaller the echo being valid. This echo behavior is taken care of by the decrement rate PD.

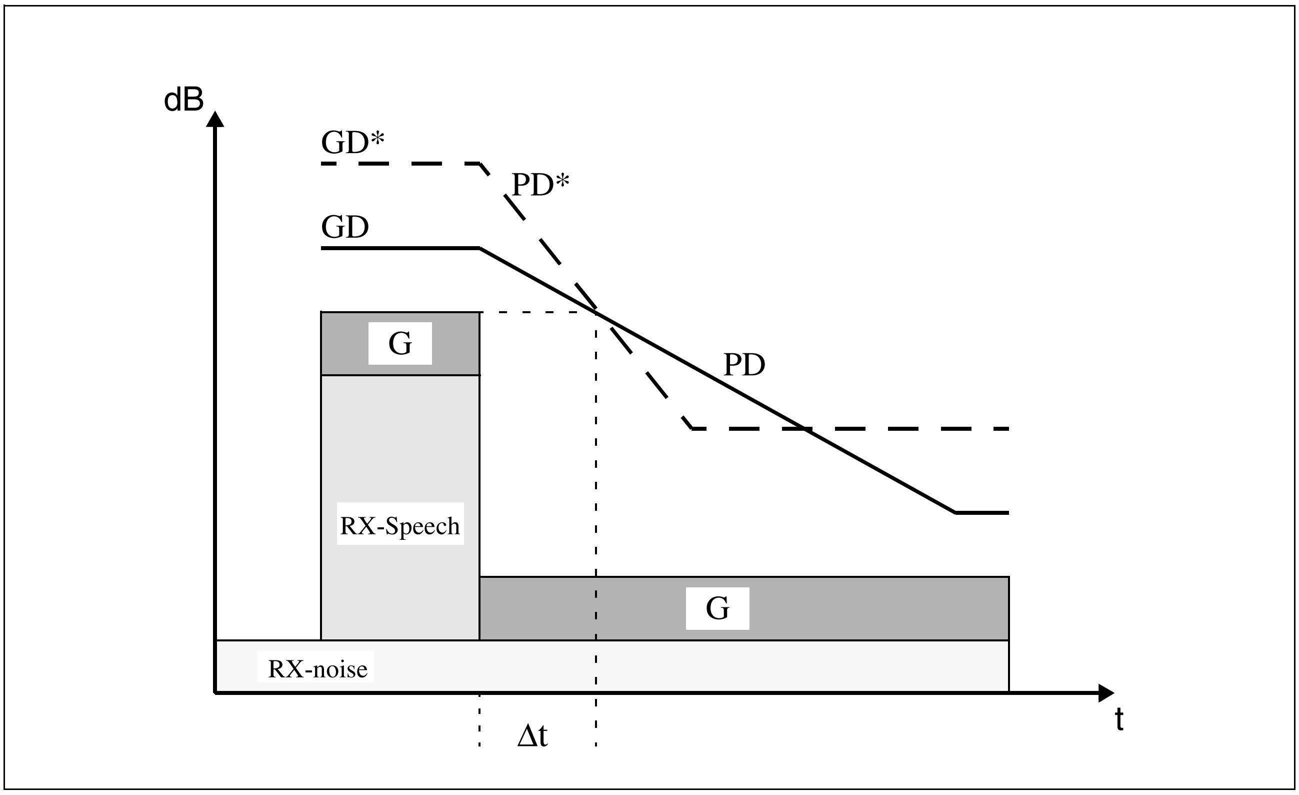


Figure 20-15 Speech Comparator - Interdependence of Parameters

According to figure 20-15, a compromise between the reserve GD and the decrement PD has to be made: a smaller reserve (GD) above the level enhancement G requires a longer time to decrease (PD). It is easy to overshout the other side but the intercommunication is harder because after the end of the speech, the level of the estimated echo has to be exceeded. In contrary, with a higher reserve (GD*) it is harder to overshout continuous speech or tones, but it enables a faster intercommunication because of a stronger decrement (PD*).

Two pairs of coefficients, GDS/PDS when speech is detected, and GDN/PDN in case of noise, offer a different echo handling for speech and non-speech. With speech, even if very strong resonances are present, the performance will not be worsened by the high GDS needed. Only when speech is detected, a high reserve prevents clipping.

The time ET [ms] after speech ends, the parameters of the comparator are switched to the “noise” values. If both sets of the parameters are equal, ET has no effect.

Table 20-8 Speech Comparator Parameters

Parameter	# of bytes	Range	Comment
G	1	– 48 to + 48 dB	Base Gain
GDS	1	0 to 48 dB	Gain Reserve (Speech)
PDS	1	0.025 to 6 dB/ms	Peak Decrement (Speech)
GDN	1	0 to 48 dB	Gain Reserve (Noise)
PDN	1	0.025 to 6 dB/ms	Peak Decrement (Noise)
ET	1	0 to 992 ms	Time to Switch from speech to noise parameters

20.1.3.3 Attenuation Control

The attenuation control unit performs state switching by controlling the attenuation stages GHX and GHR. In receive state, the attenuation G is completely switched to GHX. In transmit state, the attenuation G is completely switched to GHR. In idle state, both GHX and GHR attenuate by G/2. State switching depends on the signals of one speech comparator and the corresponding speech detector.

The attenuation G is programmable. The attenuation G actually provided by the attenuation stages GHR and GHX is the attenuation determined by the parameter ATT minus the attenuation reported by the echo cancellation unit ($G = ATT - G_C$).

Additional (fixed) attenuation on the transmit and receive path is also influenced by the automatic gain control stages AGCX and AGCR, respectively.

While each state is associated with the programmed attenuation, the time T_{SW} it takes to reach the steady-state attenuation after a state switch can be programmed. The time T_{SW} depends on a programmable decay rate SW and the current attenuation G by the formula $T_{sw} = SW \times G$.

If the current state is either transmit or receive and no speech on either side has been detected for time T_{TW} then the idle state is entered. To smoothen the transition, the attenuation is incremented (decremented) by DS until the evenly distribution G/2 for both GHX and GHR is reached.

Table 20-9 summarizes the parameters for the attenuation unit.

Table 20-9 Attenuation Control Parameters

Parameter	# of bytes	Range	Comment
TW	1	16 ms to 4 s	T_{TW} to return to idle state
ATT	1	0 to 95 dB	Attenuation for GHX and GHR
DS	1	0.6 to 680 ms/dB	Decay Speed (to idle state)
SW	1	0.0052 to 10 ms/dB	Decay Rate (used for T_{SW})

Note: In addition, attenuation is also influenced by the Automatic Gain Control stages (AGCX, AGCR) in order to keep the total loop attenuation constant.

Note: By programming parameter DS to 0xFF idle mode is disabled and the speakerphone will remain in the last state. This parameter must be changed before enabling the speakerphone.

20.1.3.4 Echo Suppression Status Output

The INCA-D can report the current state of the echo suppression unit to ease the optimization of the parameter set of the echo suppression unit. The information is available in register SPSCTL. The bits SPS0 and SPS1 are coded according to **Table 20-10**.

Table 20-10 SPS Encoding

SPS_0	SPS_1	Echo Suppression Unit State
0	0	no echo suppression operation
0	1	receive
1	0	transmit
1	1	idle

20.1.3.5 Automatic Gain Control

The echo suppression unit has two identical automatic gain control units AGCX and AGCR both referred to as AGC in this section.

Whether the automatic gain control AGC amplifies or attenuates depends on whether the signal level is above or below the threshold level defined by parameter COM. The threshold is relative to the maximum PCM-value and thus negative. The parameters AG_GAIN and AG_ATT determine the amplification and attenuation, respectively. The bold line in figure 20-16 gives an example for the steady-state output level of the AGC as a function of the input level.

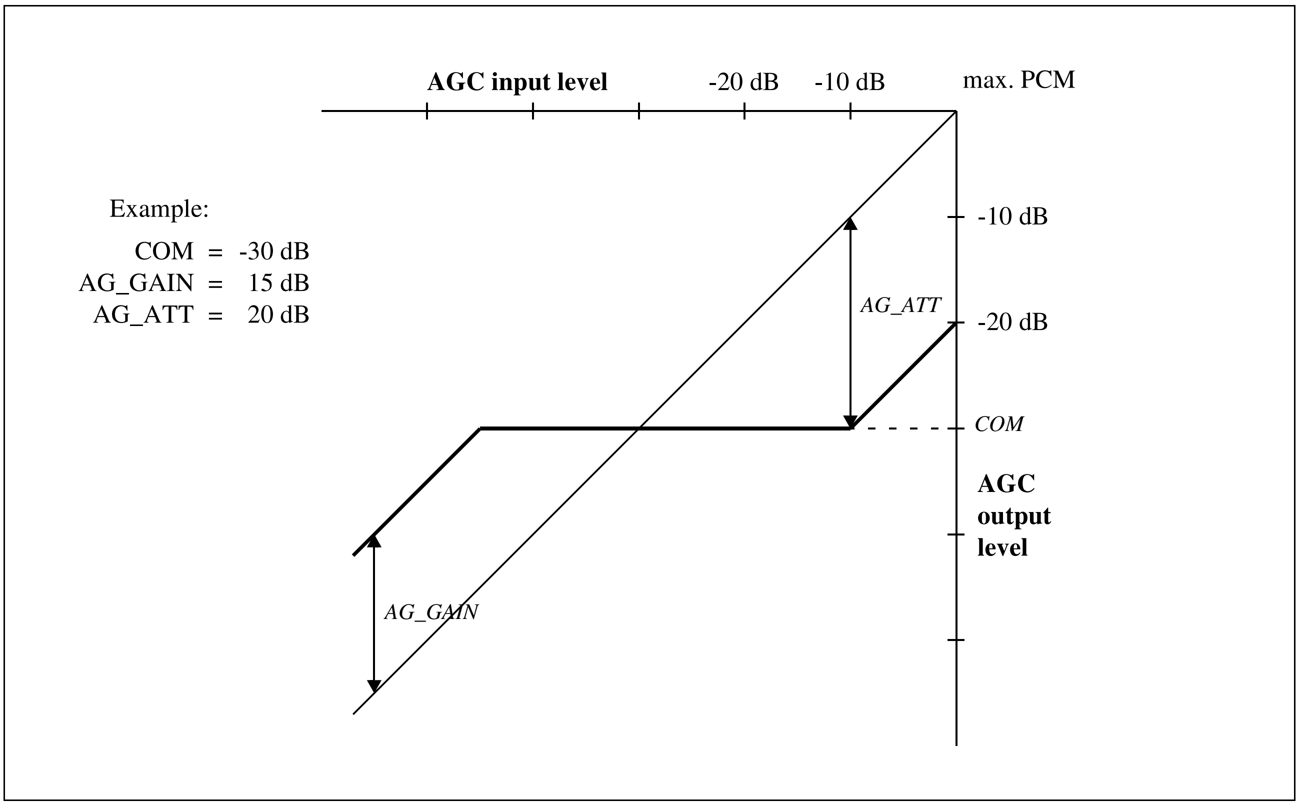


Figure 20-16 Echo Suppression Unit - Automatic Gain Control

For reasons of physiological acceptance, the AGC gain is automatically reduced in case of continuous background noise (e.g. by ventilators). The reduction is programmed via the NOIS parameter. When the noise level exceeds the threshold determined by NOIS, the amplification will be reduced by the same amount the noise level is above the threshold.

The regulation speed is controlled by SPEEDH for signal amplitudes above the threshold and SPEEDL for amplitudes below. Usually SPEEDH will be chosen to be at least 10 times faster than SPEEDL. An additional low pass with time constant LP is provided to avoid an immediate response of the AGC to very short signal bursts. The time constant of the low pass should not be selected longer than 4 ms in order to avoid unstable behavior.

If the speech detector SDX detects noise or the receive path is active, AGCX freezes its current attenuation and the last gain setting is used. Regulation starts with this value as soon as SDX detects speech and the receive path is inactive. Likewise, if SDR detects noise or the transmit path is active, AGCR freezes its current attenuation and the last gain setting is used. Regulation starts with this value as soon as SDR detects speech and the transmit path is inactive.

The current gain/attenuation of the AGC can be read at any time (AG_CUR). When the AGC has been disabled, the initial gain used immediately after enabling the AGC can be programmed. Table 20-13 shows the parameters of the AGC.

Table 20-11 Automatic Gain Control Parameters

Parameter	# of Bytes	Range	Comment
AG_INIT	1	-95 dB to 95dB	Initial AGC gain/attenuation
COM	1	0 to – 95 dB	Compare level rel. to max. PCM-value
AG_ATT	1	0 to -95 dB	Attenuation range
AG_GAIN	1	0 to 48 dB	Gain range
AG_CUR	1	-95 dB to 95 dB	Current gain/attenuation
SPEEDL	1	0.25 to 62.5 dB/s	Change rate for lower levels
SPEEDH	1	0.25 to 62.5 dB/s	Change rate for higher levels
NOIS	1	0 to – 95 dB	Threshold for AGC-reduction by background noise
LPA	1	0.025 to 16 ms	AGC low pass time constant

Note: There are two sets of parameters, one for AGCX and one for AGCR.

Note: By setting AG_GAIN to 0 dB a limitation function can be realized with the AGC.

20.1.3.6 Fixed Gain

Each signal path features an additional amplifier (LGAX, LGAR) that can be set to a fixed gain. These amplifiers should be used for the basic amplification in order to avoid saturation in the preceding stages. Table 20-12 shows the only parameter of this stage.

Table 20-12 Fixed Gain Parameters

Parameter	# of Bytes	Range	Comment
LGAX	1	-24 dB to 24 dB	always active
LGAR	1	-36 dB to 12 dB	always active

20.1.3.7 Mode Control

Table 20-13 shows the main registers used to determine the signal sources and the mode.

Table 20-13 Speakerphone Registers

Register	# of Bits	Name	Comment
SCTL	1	ENS	Echo suppression unit enable
SCTL	3	QU	Select Quality Mode (half-duplex/full-/subband)
SCTL	1	MD	Speakerphone or loudhearing mode
SCTL	1	AGX	AGCX enable
SCTL	1	AGR	AGCR enable
SSRC1	5	I1	Input signal 1 (microphone)
SSRC1	5	I2	Input signal 2 (microphone)
SSRC2	5	I3	Input signal 3 (line in)
SSRC2	5	I4	Input signal 4 (line in)
SCTL	1	SDX	Signal source of SDX
SCTL	1	SDR	Signal source of SDR

20.1.4 Line Echo Cancellation Unit

The DSP contains an adaptive line echo cancellation (LEC) unit for the cancellation of near end echoes. In normal and superior mode line echoes up to 4 ms are considered. The extended mode works basically like the superior mode, but considers line echoes of up to 8 ms.

The line echo cancellation unit is especially useful in front of the DTMF detector. A general block diagram is shown in figure 20-17.

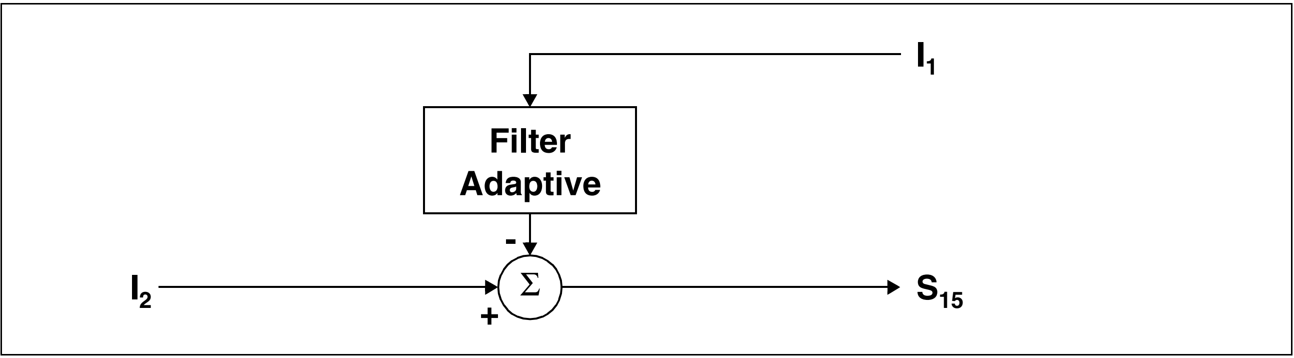


Figure 20-17 Line Echo Cancellation Unit - Block Diagram

Input I_2 is usually connected to the line input while input I_1 is connected to the outgoing signal.

In normal mode the adaption process is controlled by the three parameters MIN, ATT and MGN. Adaption takes place only if both of the following conditions hold:

1. $I1 > MIN$
2. $I1 - I2 - ATT + MGN > 0$

With the first condition, adaption to small signals can be avoided. The second condition avoids adaption during double talk. The parameter ATT represents the echo loss provided by external circuitry. The adaption stops if the power of the received signal (I2) exceeds the power of the expected signal (I1-ATT) by more than the margin MGN.

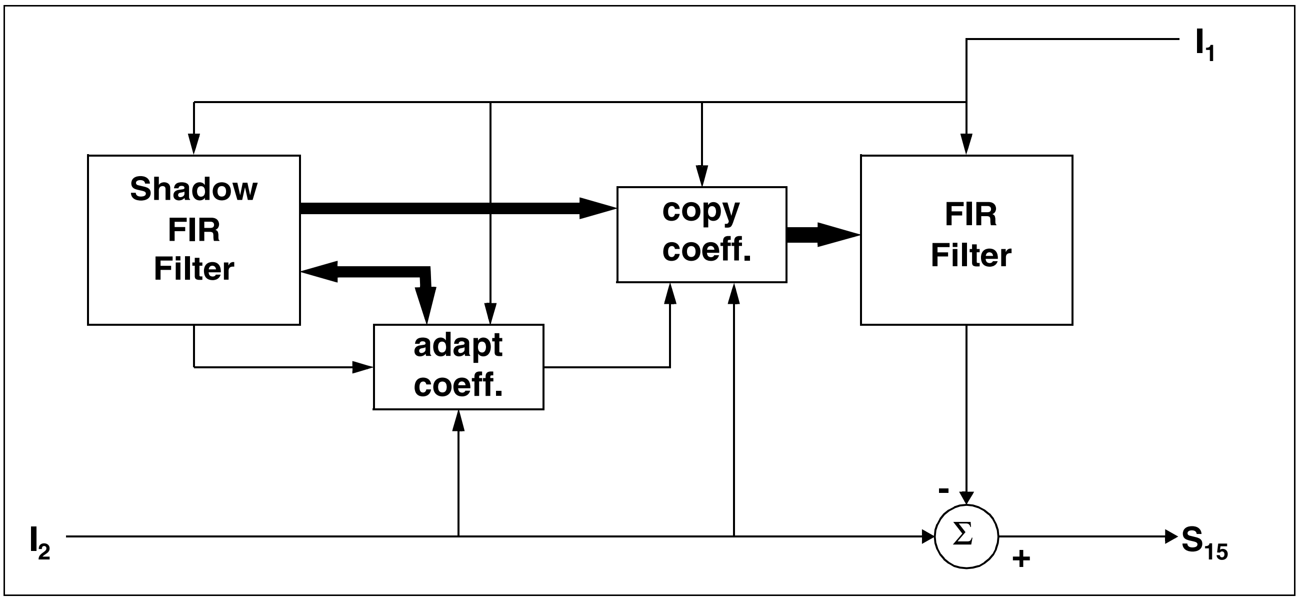


Figure 20-18 Line Echo Cancellation Unit - Superior Mode with Shadow FIR

The basic idea of the superior mode is shown in figure **20-18**. The shadow FIR filter on the left hand side gets its coefficients adapted similarly to the adaptive filter of the Line Echo Canceller in normal mode. For cancelling the line echo, however, the FIR filter on the right hand side is used. When the quality of this FIR filter is excelled by the quality of the shadow FIR filter, the coefficients of the shadow FIR filter are copied to the FIR on the right hand side. More formally, the coefficients of the shadow FIR filter are adapted if similar to normal mode, the following two conditions hold:

1. $I1 > MIN$
2. $I1 - I2 - ATT > 0$

In this case, ATT is already the difference between external echo loss and margin ($ATT_{superior}$ = ATT_{normal} - MGM_{normal}) so that the condition is actually the same as for normal mode. The parameter ATT should be adjusted accordingly. Note that ATT can now be negative.

The coefficients are copied from the shadow FIR filter to the actually used FIR filter if

1. currently the adaption of the shadow FIR filter is in progress

and at least one of the following two conditions holds:.

2. $ATTS - ATTA > MGN$
   The attenuation of the shadow FIR filter ATTS is better than the attenuation of the actually used FIR filter ATTA by a margin MGN. Note that parameter MGN has now a different meaning than in normal mode
3. $TTS(t) > \max(ATTS(t-1), \ldots, ATTS($last time condition 2 has been valid$))$
   The current attenuation ATTS of the shadow FIR is better than at any time since last update according to condition 2.

Table 20-14 shows the registers associated with the line echo canceller.

Table 20-14 Line Echo Cancellation Unit Registers

Register	# of Bits	Name	Comment	Relevant Mode
LECCTL	1	EN	Line echo canceller enable	all
LECCTL	2	CM	Select normal mode, superior mode or extended mode	
LECCTL	1	AS	Adaptation stop	all
LECCTL	1	SP	Adaptation speed	extended
LECCTL	5	I2	Input signal selection for I_2	all
LECCTL	5	I1	Input signal selection for I_1	all
LECLEV	15	MIN	Minimal power for signal I_1	all
LECATT	15	ATT	Externally provided attenuation (I_1 to I_2)	all
LECMGN	15	MGN	Margin	all

The adaptation of the coefficients can be stopped by setting bit AS in register LECCTL. This holds for all modes of the Line Echo Canceller. Furthermore for superior mode, also the copying of the coefficients from the shadow FIR is disabled.

20.1.5 DTMF Detector

The DSP contains a DTMF detector that recognizes the sixteen standard DTMF tones (for performance characteristics refer to **Chapter 26.11**). Figure 20-19 shows a block diagram of the DTMF detector.The results of the detector are available in the status and a dedicated result register.

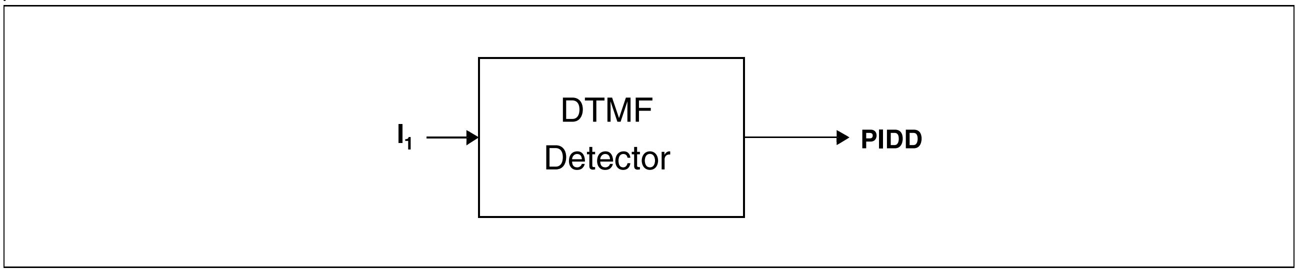


Figure 20-19 DTMF Detector - Block Diagram

Table 20-15 to 20-17 show the associated registers.

Table 20-15 DTMF Detector Control Register

Register	# of Bits	Name	Comment
DDCTL	1	EN	DTMF detector enable
DDCTL	5	I1	Input signal selection

As soon as a valid DTMF tone is recognized, the status word and the DTMF tone code are updated (table 20-16).

Table 20-16 DTMF Detector Results

Register	# of Bits	Name	Comment
PIDDSTAT	1	DTV	DTMF code valid
DDCTL	5	DTC	DTMF tone code

DTV is set when a DTMF tone is currently recognized and cleared when no DTMF tone is recognized or the detector is disabled. The code for the DTMF tone is provided in register DDCTL. DTC is valid when DTV is set and until the next incoming DTMF tone. The registers DDTW and DDLEV contain the parameters for detection (table 20-17) .

Table 20-17 DTMF Detector Parameters

Register	# of Bits	Name	Comment
DDTW	15	TWIST	Twist for DTMF recognition
DDLEV	6	MIN	Minimum signal level to detect DTMF tones

20.1.6 DTMF Generator

The DTMF generator can generate single or dual tones with programmable frequency and gain. This unit is primarily used to generate the common DTMF tones but can also be used for signalling or other user defined tones. A block diagram is shown in figure 20-20.

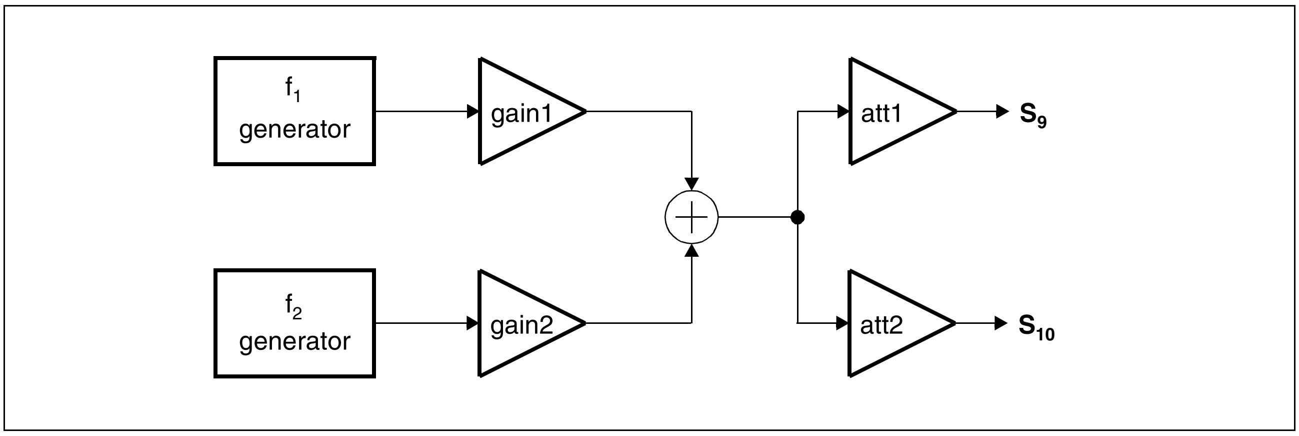


Figure 20-20 DTMF Generator - Block Diagram

Both generators and amplifiers are identical. There are two modes for programming the generators, cooked mode and raw mode. In cooked mode, DTMF tones are generated by programming a single 4 bit code. In raw mode, the frequency of each generator/ amplifier can be programmed individually by a separate register. The unit has two outputs which provide the same signal but with individually programmable attenuation. Table 20-18 shows the parameters of this unit.

Table 20-18 DTMF Generator Registers

Register	# of Bits	Name	Comment
DGCTL	1	EN	Enable for generators
DGCTL	1	MD	Mode (cooked/raw)
DGCTL	4	DTC	DTMF code (cooked mode)
DGF1	15	FRQ1	Frequency of generator 1
DGF2	15	FRQ2	Frequency of generator 2
DGL	7	LEV1	Level of signal for generator 1
DGL	7	LEV2	Level of signal for generator 2
DGATT	8	ATT1	Attenuation of S_9
DGATT	8	ATT2	Attenuation of S_{10}

Note: DGF1 and DGF2 are undefined when cooked mode is used and must not be written.

20.1.7 Interface to the Analog Front End

There is one interface to the analog interface as shown in **Figure 20-21**.

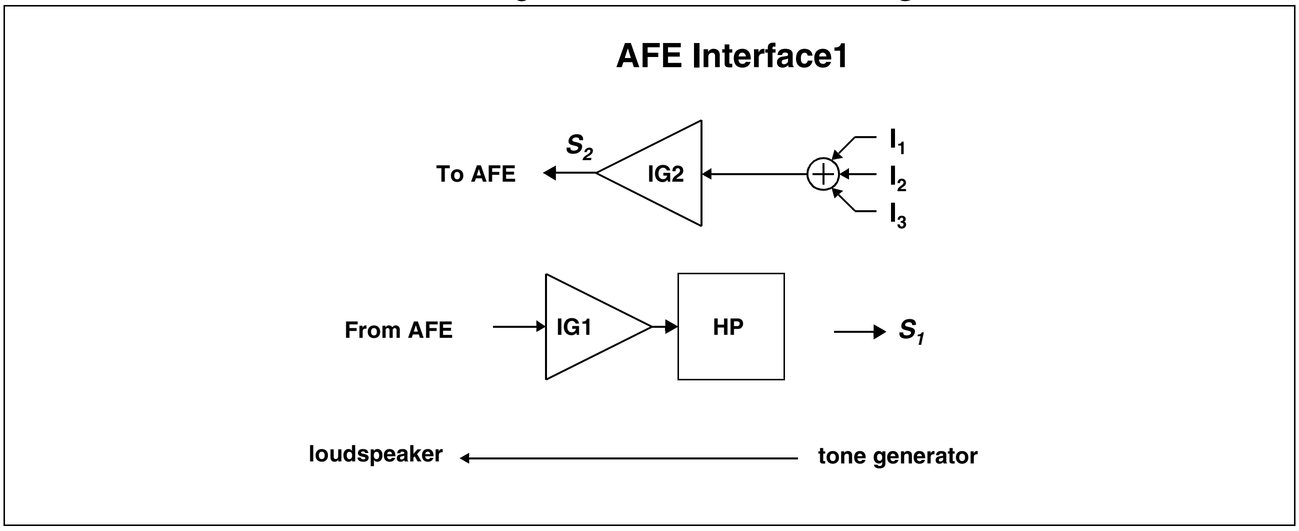


Figure 20-21 Analog Frontend Interface - Block Diagram

For each direction, an amplifier is provided for level adjustment. The signal coming from the analog front end is passed through a high-pass (HP). With the parameter HP, the high-pass can be configured either to suppress the DC part of its input or to be a "real" high-pass with the -3 dB corner frequency at 180 Hz. Furthermore, up to three signals can be mixed in order to generate the outgoing signals (S_2). Table 20-19 shows the associated registers.

Table 20-19 Analog Frontend Interface Registers

Register	# of Bits	Name	Comment
IFG1	15	IG1	Gain for IG1
IFG2	15	IG2	Gain for IG2
IFS1	1	HP	High-pass for S_1
IFS1	5	I1	Input signal 1 for IG2
IFS1	5	I2	Input signal 2 for IG2
IFS1	5	I3	Input signal 3 for IG2

20.1.8 Universal Attenuator

The DSP contains an universal attenuator that can be connected to any signal (e.g. for side-tone gain in ISDN applications).

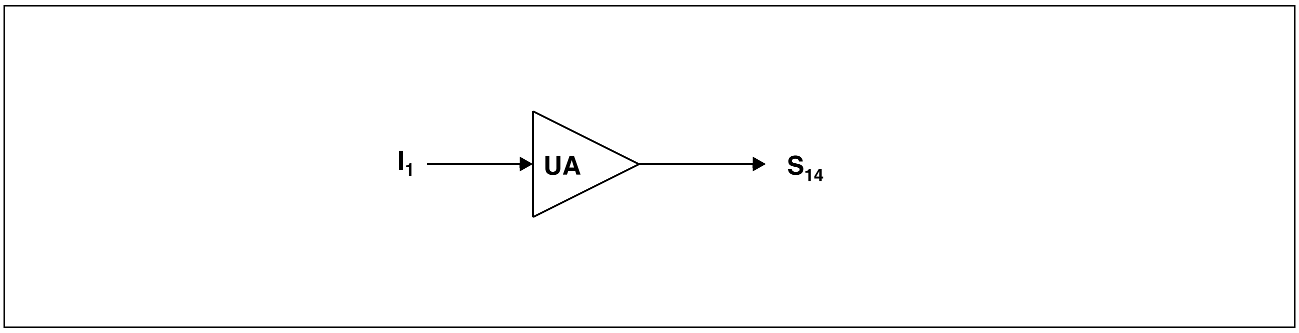


Figure 20-22 Universal Attenuator - Block Diagram

Table below shows the associated register.

Table 20-20 Universal Attenuator Registers

Register	# of Bits	Name	Comment
UA	8	ATT	Attenuation for UA
UA	5	I1	Input signal for UA

20.1.9 Automatic Gain Control Unit

In addition to the universal attenuator with programmable but fixed gain the DSP contains an amplifier with automatic gain control (AGC). The AGC is preceded by a signal summation point for two input signals. One of the input signals can be attenuated.

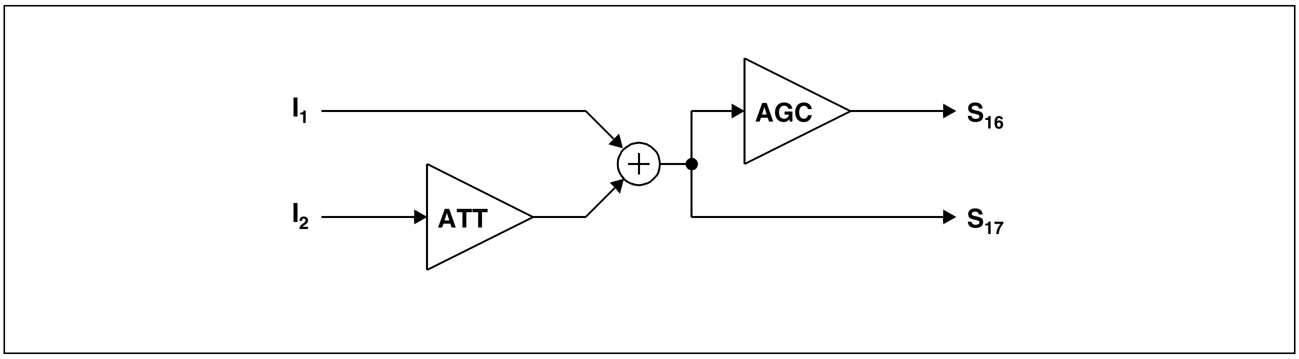


Figure 20-23 Automatic Gain Control Unit - Block Diagram

The operation of the AGC is similar to AGCX (ACCR) of the speakerphone. The differences are as follows:

- No NOIS parameter
- Separate enable/disable control
- Slightly different coefficient format

Furthermore the AGC contains a comparator that starts and stops the gain regulation. The signal after the summation point (S17) is filtered by a peak detector with time constant DEC for decay. Then the signal is compared to a programmable limit LIM. Regulation takes only place when the filtered signal exceeds the limit.

Table 20-21 shows the associated registers.

Table 20-21 Automatic Gain Control Registers

Register	# of Bits	Name	Comment
AGCCTL	1	EN	Enable
AGCCTL	5	I1	Input signal 1 for AGC
AGCCTL	5	I2	Input signal 2 for AGC
AGCATT	15	ATT	Attenuation for I_2
AGC1	8	AG_INIT	Initial AGC gain/attenuation
AGC1	8	COM	Compare level relative to max. PCM-value
AGC2	8	SPEEDL	Change rate for lower levels
AGC2	8	SPEEDH	Change rate for higher level
AGC3	7	AG_ATT	Attenuation range
AGC3	8	AG_GAIN	Gain range
AGC4	7	DEC	Peak detector time constant
AGC4	8	LIM	Comparator minimal signal level
AGC5	7	LP	AGC low pass time constant

20.1.10 Noise Reduction Unit

Additionally to the noise reduction (NR) unit built in the speakerphone, another noise reduction unit is available. It can be useful for reducing the noise coming in the receive path if uncomfortable noise comes from the line side.

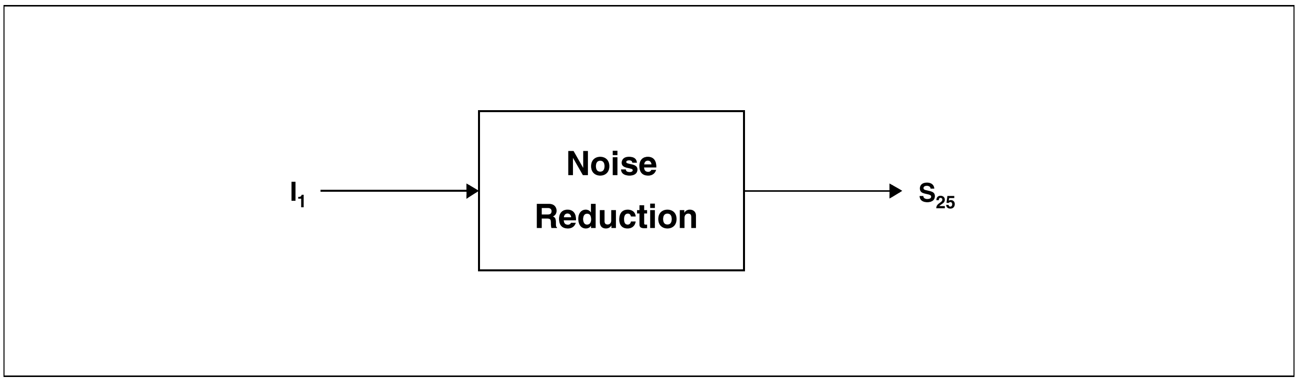


Figure 20-24 Noise Reduction - Block Diagram

The noise reduction unit attenuates frequencies with a great ratio of the noise energy level to the level of the speech signal. The maximal attenuation of the noise reduction unit can be programmed by the parameter NRATT.

There are some restrictions for the simultaneous selection of the speakerphone (refer to **Chapter 20.3**). Different trade-offs can be selected with the parameter MD.

Note: The Noise Reduction unit should be directly connected to the Line-In signal. Otherwise it can cause acoustic distortions.

Table 20-22 Peak Detector Registers

Register	# of Bits	Name	Comment
NRCTL	1	EN	Noise Reduction Unit Enable
NRCTL	2	MD	Select restriction of speakerphone
NRCTL	5	I1	Input signal selection
NRATT	15	NRATT	Maximal attenuation of the noise reduction unit.

20.1.11 Equalizer

The DSP contains two identical equalizers which can be programmed individually. Each equalizer can be inserted into any signal path. The main application for the equalizer is the adaptation to the frequency characteristics of the microphone, transducer or loudspeaker.

Each equalizer consists of an IIR filter followed by an FIR filter as shown in **Figure 20-25**.

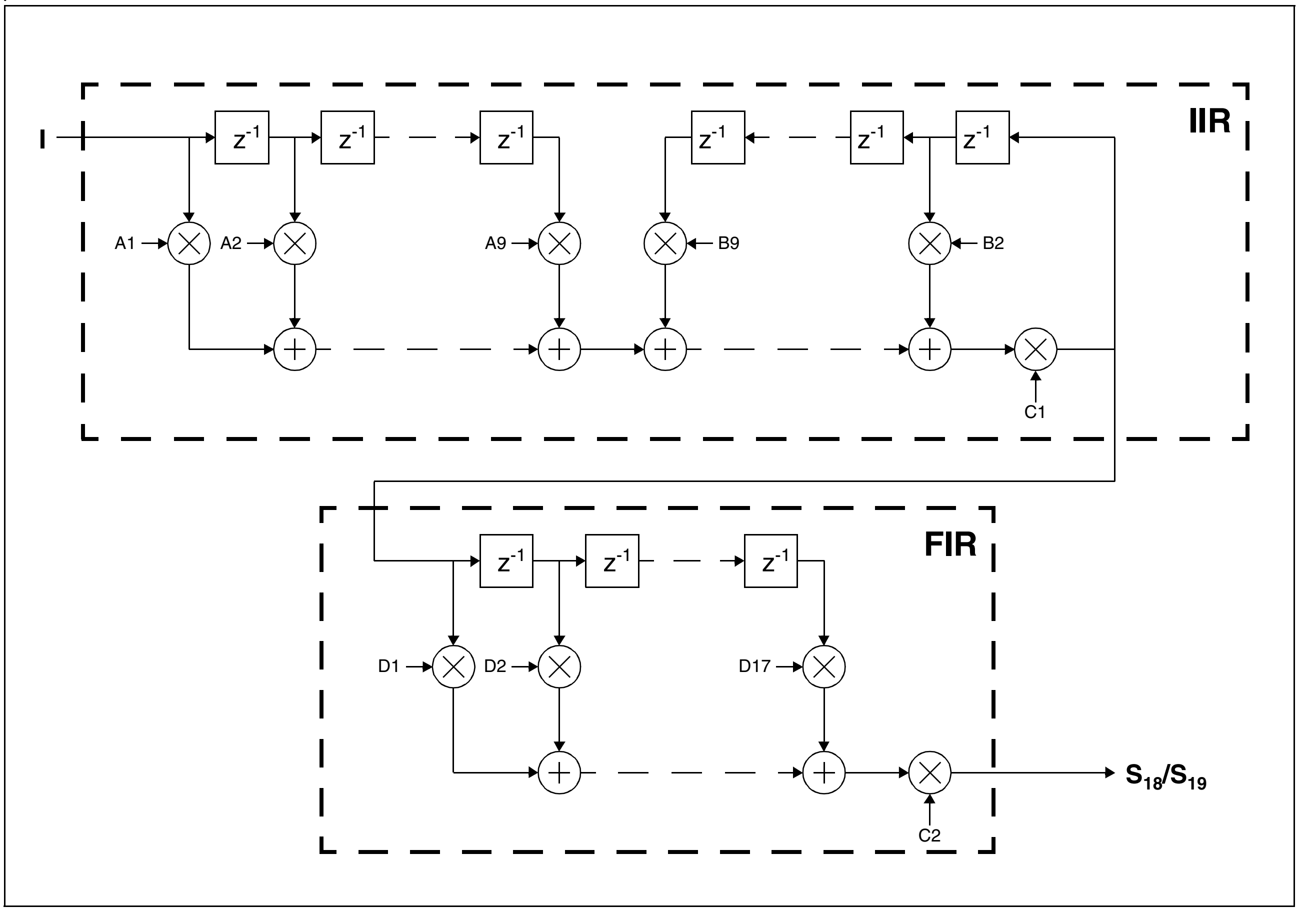


Figure 20-25 Equalizer - Block Diagram

The coefficients A_1-A_9, B_2-B_9 and C_1 belong to the IIR filter, the coefficients D_1-D_{17} and C_2 belong to the FIR filter. Table 20-23 shows the registers associated with the first equalizer (S_{18}). The second equalizer (S_{19}) is programmed by the registers FCFCTL2 and FCFCOF2, respectively

Table 20-23 Equalizer Registers

Register	# of Bits	Name	Comment
FCFCTL1	1	EN	Enable
FCFCTL1	5	I1	Input signal for equalizer
FCFCTL1	6	ADR	Filter coefficient address
FCFCOF1	16		Filter coefficient data

Due to the multitude of coefficients the INCA-D uses an indirect addressing scheme for reading or writing an individual coefficient. The address of the coefficient is given by ADR and the actual value is read or written to register FCFCOF1 or FCFCOF2 respectively.

In order to ease programming the DSP automatically increments the address ADR after each access to FCFCOF1 or FCFCOF2 respectively.

Note: Any access to an out-of-range address automatically resets FCFCTL1:ADR.

20.1.12 Tone Generator

The DSP contains an universal tone generator which can be used for tone alerting, call progress tones or other audible feedback tones. **Figure 20-26** shows a block diagram of this unit.

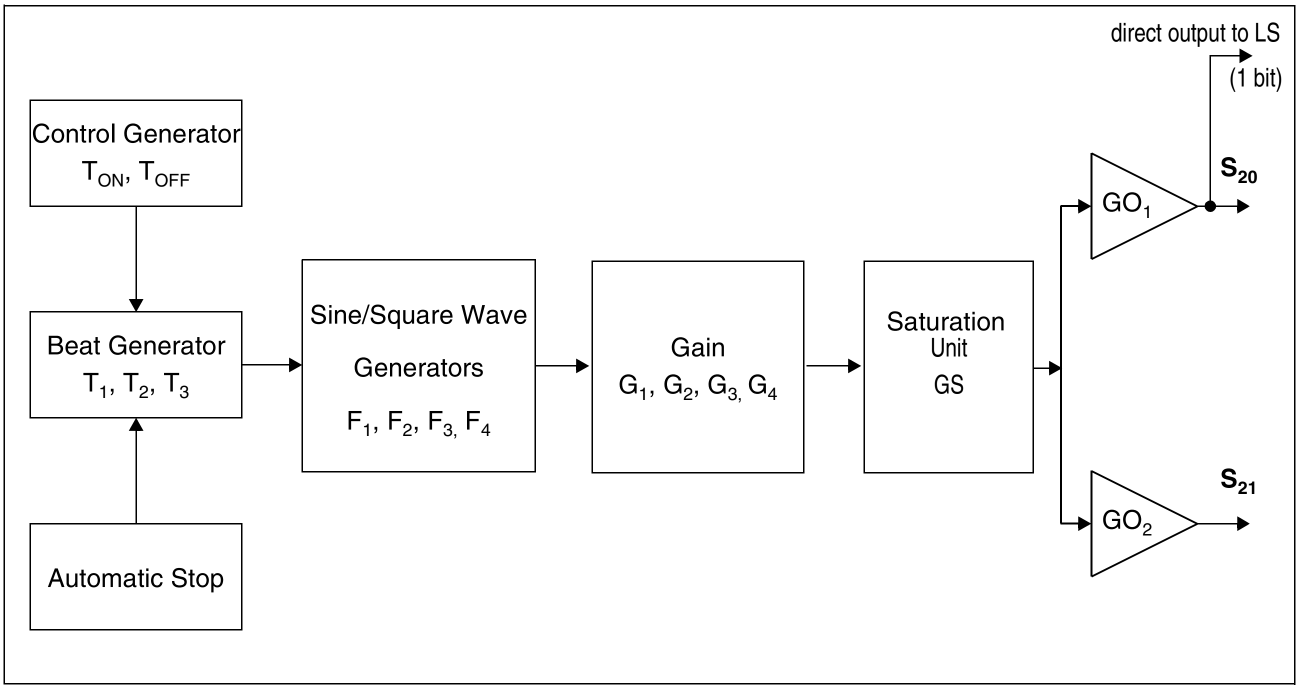


Figure 20-26 Tone Generator - Block Diagram

The heart of this unit are the four independent sine/square wave generators that can generate individually programmable frequencies (F_1, F_2, F_3, F_4). Each generator has an associated amplifier (G_1, G_2, G_3, G_4). The dynamic behavior of the tone generator is controlled by the beat generator.

If the beat generator is enabled, then the output is either a three tone cadence or a two tone cadence as shown in **Figure 20-27**.

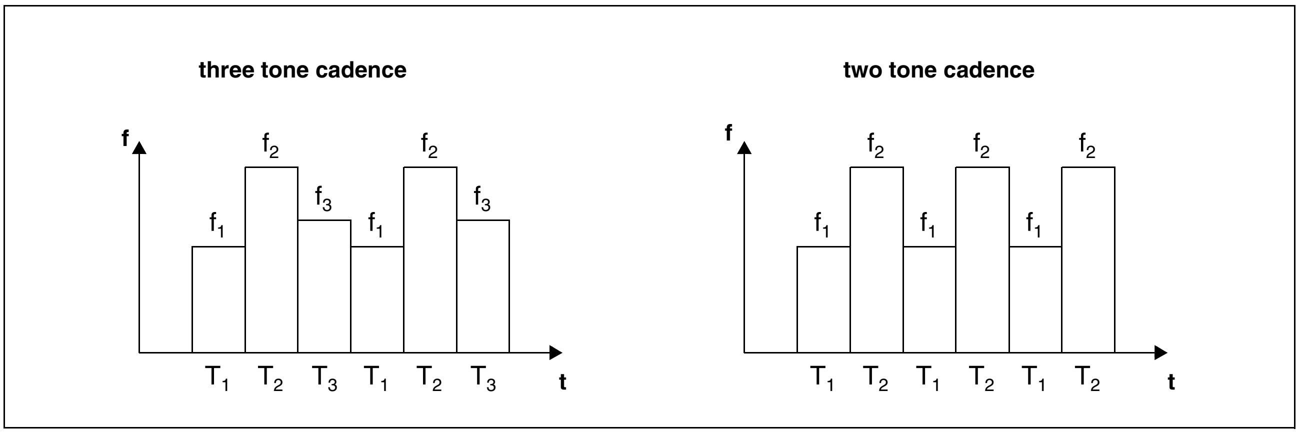


Figure 20-27 Tone Generator - Tone Sequences

The duration of each frequency is defined by T_1, T_2 and T_3. For each time slot either the associated frequency can be generated or a frequency pair .

Table 20-24 Tone Generator Modes

Timeslot	Option 1	Option 2
T_1	F_1	F_1+F_4
T_2	F_2	F_2+F_4
T_3	F_3	F_3+F_4

If the beat generator is disabled, then the output is a continuous signal of either F_1, F_2, F_1+F_4, F_2+F_4 or silence.

The control generator is used to enable the beat generator (during T_{ON}) and disable it during T_{OFF}. With the automatic stop feature, the cadence generation of the beat generator stops not immediately but after the end of a cadence (either T_2 or T_3). This avoids unpleasant sounds when stopping the tone generator unit.

Table 20-25 shows the registers associated with the tone and ringing generator.

Table 20-25 Tone Generator Registers

Register	# of Bits	Name	Comment
PIDDSTAT	1	TG	Status bit (Tone Generator on/off)
TGCTL	2	CGM	Control generator mode
TGCTL	1	DT	Dual tone enable (F4 on/off)
TGCTL	2	BGM	Beat generator mode (F1, F2, F1/F2 or F1/F2/F3)
TGCTL	1	SM	Stop mode (immediate or automatic)
TGCTL	1	WF	Waveform (sine or square)

Table 20-25 Tone Generator Registers

Register	# of Bits	Name	Comment
TGTON	13	TM	T_{ON}
TGTOFF	13	TM	T_{OFF}
TGTON	3	TE	T_{ON}
TGTOFF	3	TE	T_{OFF}
TGT1	16	TIME	T_1
TGT2	16	TIME	T_2
TGT3	16	TIME	T_3
TGF1	15	F	F_1
TGF2	15	F	F_2
TGF3	15	F	F_3
TGF4	15	F	F_4
TGG1	15	G	G_1
TGG2	15	G	G_2
TGG3	15	G	G_3
TGG4	15	G	G_4
TGGO1	15	G	GO_1
TGGO2	15	G	GO_2
TGGS	15	GS	GS, Saturation Gain

With the saturation unit additional gain GS can be added to the output signal. This will cause the signal to exceed the maximum value of the internal representation so that the peaks will be cut off. This induces more harmonics to make a tone sound louder.

The tone generator unit has two outputs (S_{20} and S_{21}). The signal level of these outputs can be programmed individually by the preceding gain stages (GO_1 and GO_2).

Additionally to S_{20} and S_{21} a direct output to the loudspeaker amplifier of the analog frontend exists.This direct output is the most significant bit of S_{20}. In order to use this path, the tone generator must be set to square wave generation (TGCTL.WF = 1).

If TGCTL.E16 = 0, the output of the tone generator is limited to 4 kHz maximum output frequency of the harmonics. This is normal telephone quality and corresponds to 8 kHz sample frequency. If direct output to the AFE loudspeaker amplifier is used, it may be desired to rise the frequency limit. With TGCTL.E16 = 1, the output of the tone generator is limited to 8kHz maximum output frequency of the harmonics, which corresponds to 16 kHz sample frequency. This allows to use more frequencies at the direct output but the normal outputs S_{20} and S_{21} can no more be used. Furthermore, this setting has the effect that the tone lengths are half the programmed time and the tone frequencies are double the programmed frequencies.

20.1.13 Peak Detector

The peak detector is usually not used in normal operation. It provides, however, an easy means to verify the minimum or maximum signal level of any signal S_i within the INCA-D. The peak detector stores either the maximum or the minimum signal value of the observed signal I_1 in the register PDDATA since the last read access to this register. Therefore it is not only possible to determine the absolute level of the signal but it can also be checked whether a DC offset is present. This can be done by first scanning for the maximum and then for the minimum value. If the minimum value is not (approximately) the negated positive value then a DC offset is present. The peak detector should be disabled if not needed.

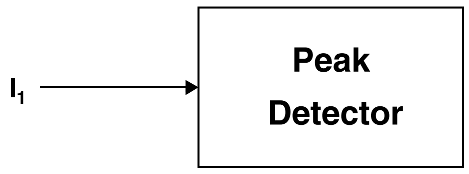


Figure 20-28 Peak Detector - Block Diagram

The register PDDATA gives the maximum or minimum integer depending on the mode selected by bit MM. As an example it may be assumed that the detection of the maximum is selected. Then with enabling the detector and with each read access to register PDDATA, PDDATA is set to the smallest possible value, which is the negative maximum integer. With each new maximum detected on signal I1, this maximum is provided by PDDATA.

Table 20-26 Peak Detector Registers

Register	# of Bits	Name	Comment
PDCTL	1	EN	Peak Detector Enable
PDCTL	1	MM	Minimum/Maximum selection
PDCTL	5	I1	Input signal selection
PDDATA	16		Min/Max signal value since last read access

20.1.14 Universal Summation Point

.

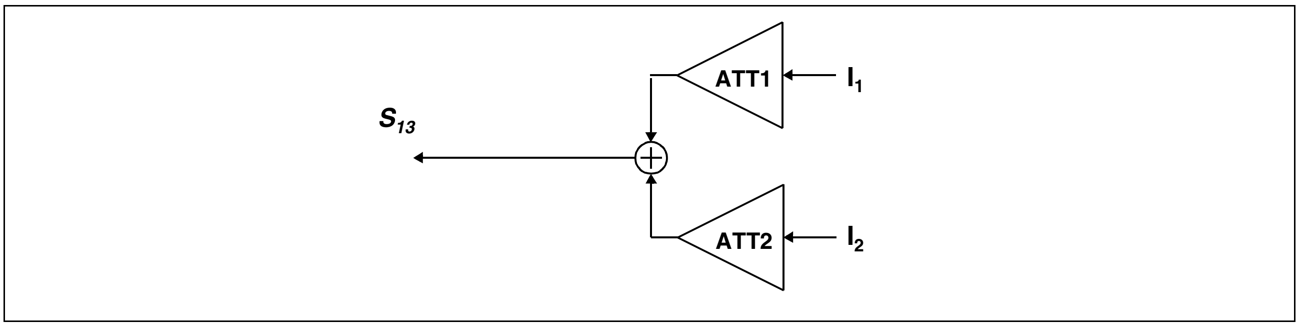


Figure 20-29 Universal Summation Point- Block Diagram

The parameters of the universal summation point are shown in **Table 20-27**.

Table 20-27 universal Summation Registers

Register	# of Bits	Name	Comment
SUMCTL	1	EN	Universal Summation Enable
SUMCTL	5	I1	Input 1 signal selection
SUMCTL	5	I2	Input 2 signal selection
SUMATT	8	ATT1	Attenuation of Input I1
SUMATT	8	ATT2	Attenuation of Input I2

20.2 Digital Interfaces

This section describes the interfaces of the DSP to the IOM-2 Bus (CDI) and the interface between CPU and DSP.

20.2.1 Interface to IOM handler (Codec Digital Interface - CDI)

Each channel has three configurable inputs and one output as shown in figure 20-30.

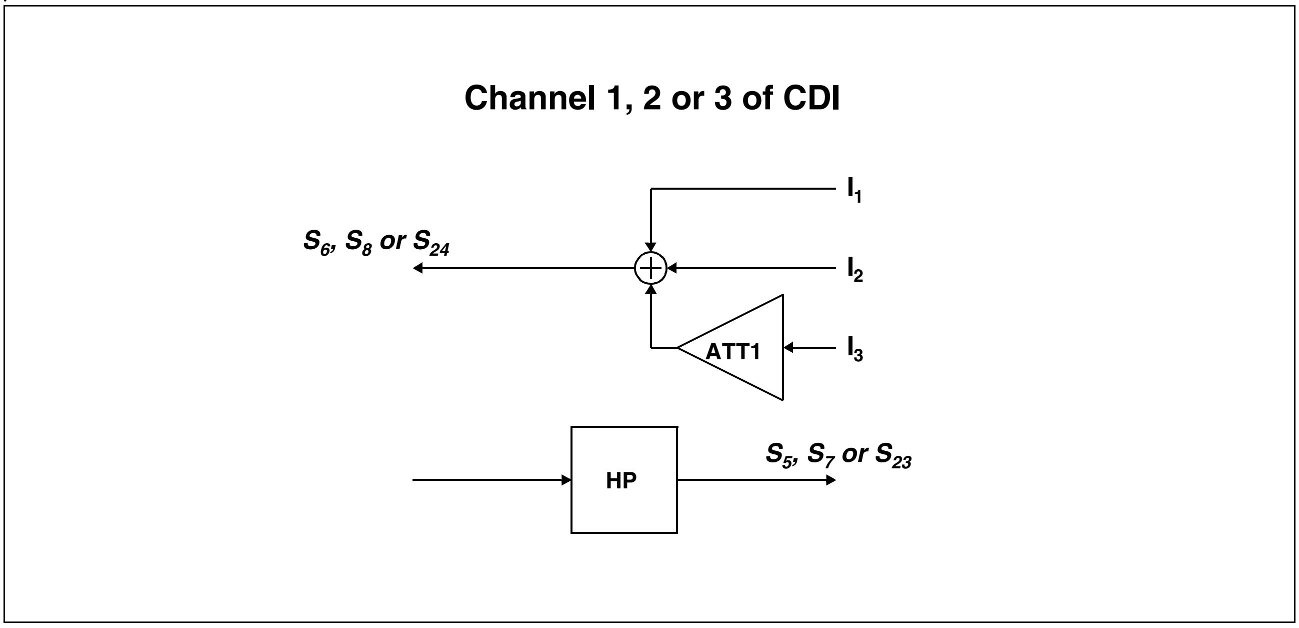


Figure 20-30 Digital Interface - Block Diagram

Each outgoing signal can be the sum of two signals with no attenuation and one signal with programmable attenuation (ATT). The attenuator can be used for artificial echo loss. Each input can be passed through an optional high-pass (HP). The associated registers are shown in table 20-28.

Table 20-28 Digital Interface Registers

Register	# of Bits	Name	Comment
IFS3	5	I1	Input signal 1 for S_6
IFS3	5	I2	Input signal 2 for S_6
IFS3	5	I3	Input signal 3 for S_6
IFS3	1	HP	High-pass for S_5
IFS4	5	I1	Input signal 1 for S_8
IFS4	5	I2	Input signal 2 for S_8
IFS4	5	I3	Input signal 3 for S_8
IFS4	1	HP	High-pass for S_7

Table 20-28 Digital Interface Registers

Register	# of Bits	Name	Comment
IFS5	5	I1	Input signal 1 for S_{24}
IFS5	5	I2	Input signal 2 for S_{24}
IFS5	5	I3	Input signal 3 for S_{24}
IFS5	1	HP	High-pass for S_{23}
IFG5	8	ATT1	Attenuation for input signal I3 (CDI Channel 1)
IFG5	8	ATT2	Attenuation for input signal I3 (CDI Channel 2)
IFG6	8	ATT3	Attenuation for input signal I3 (CDI Channel 3)

The INCA-D supports up to three channels simultaneously for data transfer (refer also to **Chapter 17.1.7**). If only two channels are used, then both the coding (PCM A-law, PCM µ-law or linear) and the data direction (DD/DU assignment for transmit/receive) can be programmed individually. The INCA-D supports a third channel by simply splitting the second 16 bit channel into two 8 bit channels.Therefore the following restriction occurs for channel 2 and 3 in this case:

- Channel two as well as three must use 8 bit coding (8 bit linear, A-law or µ-law)

To enable the channel splitting, bit CDICHN2:CS must be set and bit CDICHN2:PCM cleared. The selection of bit CDICHN2:PCD holds then for both channels.

Table 20-29 shows the registers used for configuration of the IOM®-2 interface.

Table 20-29 IOM®-2 Interface Registers

Register	# of Bits	Name	Comment
CDICHN1	1	EN	Channel 1 enable
CDICHN1	1	PCM	8 bit code or 16 bit linear PCM (channel 1)
CDICHN1	2	PCD	8 bit code (A-law or µ-law, channel 1)
CDICHN2	1	EN	Channel 2 enable
CDICHN2	1	CS	Channel 2 split (into two contiguous 8 bit channels)
CDICHN2	1	PCM	8 bit code or 16 bit linear PCM (channel 2)
CDICHN2	2	PCD	8 bit code (A-law or µ-law, channel 2)

20.2.2 Parallel Interface to the DSP Domain PIDD

The parallel interface from the XBUS to the DSP domain (PIDD) is shown in figure 20-31.

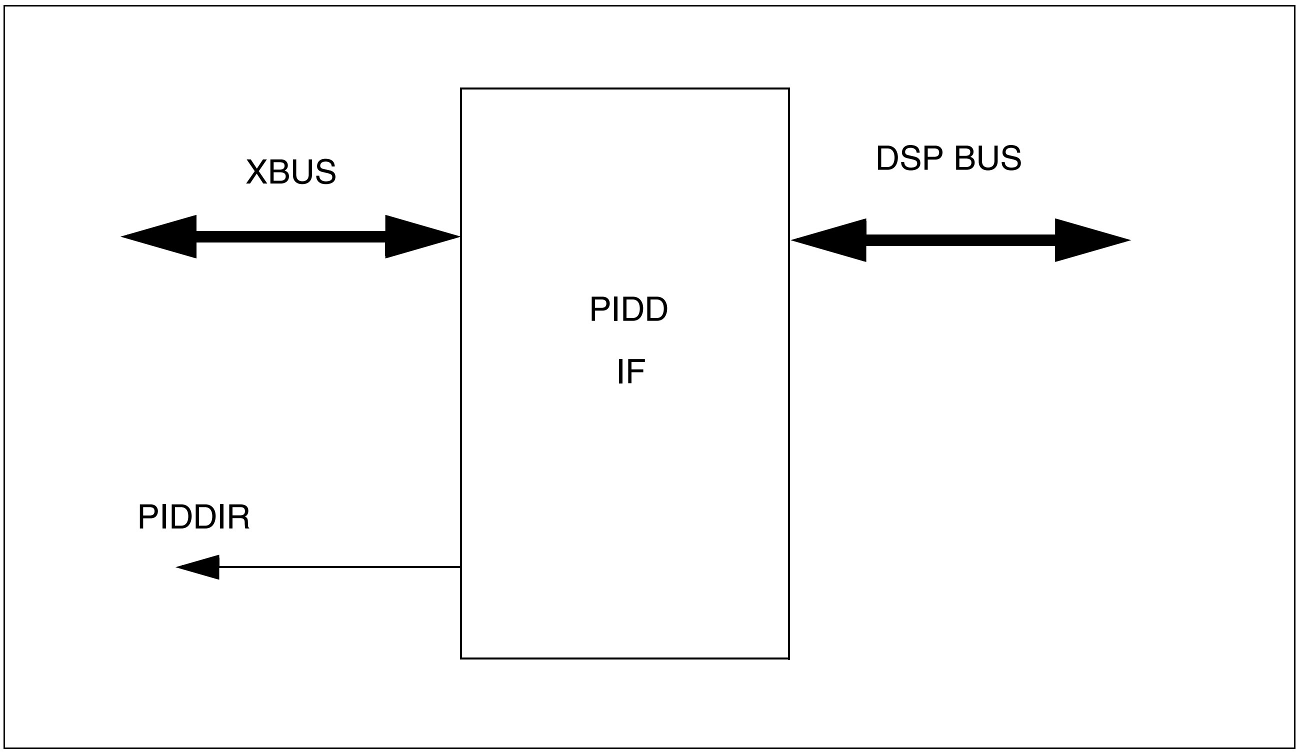


Figure 20-31 PIDD

The accesses to the DSP domain can be of the following types:

1. DSP Register Read,
2. DSP Register Write
3. Interrupt Acknowledge

The PIDD status register PIDDSTAT can be read at any time to determine, whether the DSP is able to handle the next access to its domain.

However, if the DSP is in power down mode (i.e. PIDDHWCFG.PD=1), a read access to the status register does not deliver valid data.

A “DSP Register Read” or “DSP Register Write” access can only be performed when the DSP is ready (i.e. PIDDSTAT.RDY=1). In that case the access itself is executed when the corresponding command is written into the command register PIDDCOM by the controller.

For a write access ("DSP Write Register”), the data must be written to the data register PIDDGPTD before the command register PIDDCOM is written. Upon this action, the PIDDSTAT.RDY bit is cleared by hardware and is set as soon as the DSP is ready again.

For a read access the command “DSP Register Read” has to be written into register PIDDCOM. Thus the bit RDY is cleared by hardware and is set as soon as valid data are available in register PIDDGPTC. As soon as the bit RDY is 1, the controller can read the valid data from register PIDDGPTC.

Any change of register PIDDSTAT causes an interrupt request PIDDIR, with one exception: This interrupt request is not generated, if the RDY bit changes, because the DSP has been woken up from power-down by a read or write access.

Note: Generally, the interrupt request PIDDIR can only be generated after reset or after an "Interrupt Acknowledge" command excuted by programming register PIDDCOM. This is to preserve overwriting register PIDDSTAT by a new interrupt, before the interrupt service routine of the first interrupt has read PIDDSTAT.

To clear the interrupt request PIDDIR, two actions are necessary. First, the command "*Interrupt Acknowledge*" must be written to the command register. Second, the status register must be read. The interrupt is then cleared with this second action. Both actions can be taken at any time regardless of the value of the RDY bit

The configuration register PIDDHWCFG can be read at any time. Please be aware that the configurations must be done before the modules that are affected by the configurations, are enabled. An easy way to ensure this is to write the configuration register when the DSP is in power down mode.

Table 20-30 Registers seen from the uC

Reg. name	Description	r/w	Address	Reset
PIDDCLC[1]	Clock control register of PIDD	rw	00’DFAC$_H$	0000$_H$
PIDDCOM[2]	Command register of PIDD	rw	00’DFAE$_H$	0000$_H$
PIDDSTAT[2]	Status register of PIDD	r	00’DFB0$_H$	0000$_H$
PIDDHWCFG[2]	Hardware configuration register	rw	00’DFB2$_H$	0000$_H$
PIDDGPTC	Data (**g**eneral **p**urpose) from DSP **t**o **c**ontroller via PIDD	r	00’DFB4$_H$	0000$_H$
PIDDGPTD	Data (**g**eneral **p**urpose) from controller **t**o **D**SP via PIDD	w	00’DFB6$_H$	0000$_H$

[1] see **Chapter 23.5**

[2] This register is described in detail in **Chapter 20.4.1**.

20.2.2.1 Interrupt

The DSP can generate an interrupt to inform the CPU of an update of the PIDDSTAT register according to table 20-31. An interrupt mask register (INTM) can be used to disable or enable the interrupting capability of each bit of the PIDDSTAT register individually.

Table 20-31 Interrupt Source Summary

STATUS (old)	STATUS (new)	Set by	Reset by
RDY=0	RDY=1	Command completed	Command issued
TG=1	TG=0	Tone generator active	Tone sequence finished or EN=0
DTV=0	DTV=1	DTMF tone detected	DTMF tone lost or EN=0
DTV=1	DTV=0	DTMF tone lost or EN=0	DTMF tone detected
ABT=0	ABT=1	Exception (non-maskable)	Write to REV Register

An interrupt is internally generated if any combination of these events occurs and the interrupt is not masked. The interrupts are issued immedeately after the status register update. The status register update for the event set bit DTV or ABT is performed immediately after the event occurs that causes the bit to be set. For the event clear bit DTV the status register update is performed with the next update of the RDY bit, i.e., up to 125 us after the event occurs that causes the bit to be set. For the event clear bit TG, the status register update is performed 125 us after the update of the RDY bit as the latest.

This internal interrupt is cleared only when the host executes the *Data Read Access with Interrupt Acknowledge* command. In this case, the internal interrupt is cleared when the first bit of the PIDDSTAT register is output. If a new event occurs while the host reads the status register, the status register is updated *after* the current access is terminated and a new interrupt is internally generated immediately after the access has ended.

20.2.2.2 Abort

If the DSP detects a corrupted configuration (e.g. due to a transient loss of power) it stops operation and initializes all read/write registers to their reset state. The DSP discards all commands with the exception of a write command to the revision register while ABT is set. Only after the write command to the revision register (with any value) the ABT bit is reset and a reinitialization can take place.

20.2.2.3 Revision Register

The INCA-D contains a revision register. This register is read only and does not influence operation in any way. A write to the revision register clears the ABT bit of the STATUS register but does not alter the content of the revision register.

20.2.3 Hardware Configuration

The INCA-D can be adapted to various external hardware configurations by a special register: PIDDHWCFG. This register is written once during initialization and must not be changed while the INCA-D is in active mode.

20.3 Restrictions and Mutual Dependencies of Modules

There are some restrictions concerning the modules that can be enabled at the same time (**Table 20-32**). A cell marked with 'X' indicates that the two modules (defined by the row and the column of the cell) must not be enabled at the same time.

The freely configurable noise reduction unit (Section 20.1.10) has some additional restrictions which are not stated in **Table 20-32**. Even if the noise reduction can be used, some limitations for the filter length of the echo cancellation unit exist. A detailed description of these restrictions is provided with the register description of register NRCTL.

Table 20-32 Dependencies of Modules

	Subband (normal)	Subband (ISDN)	Subband (enhanced)	Subband (reduced)	Fullband mode one	Fullband mode two	Noise Reduction	DTMF Detector	Line Echo Canceller	Equalizer, DTMF, Tone
Subband (normal)	-	X	X	X	X	X				
Subband (ISDN)	X	-	X	X	X	X	X	X	X	
Subband (enhanced)	X	X	-	X	X	X	X	X	X	X
Subband (reduced)	X	X	X	-	X	X				
Fullband mode one	X	X	X	X	-	X				
Fullband mode two	X	X	X	X	X	-		*	*	
Noise Reduction		X	X				-			
DTMF Detector		X	X			*		-	+	
Line Echo Canceller		X	X			*		+	-	
Equalizer 1/2, DTMF/Tone Generator			X							-

* number of filter taps restricted to 645

+ 'extended' mode of the line echo canceller not allowed (LECCTL.CM)

Some incompatible modules share the same internal memory space. If this is for example the case for the incompatible modules A and B, and both modules are going to be used at different times, then the following scenario appears: Module A has to be programmed prior to use. After a while, it is disabled, module B is programed and module B is enabled. At a later time, module B is disabled. Before module A is enabled, the parameters of module A must be reprogrammed. This reprograming is necessary for all combination listed in table 20-33.

Table 20-33 Reprogram parameters

After use of this	The parameter of these must be reprogrammed
AEC subband mode "Enhanced"	DTMF detector, line echo canceller, tone generator, DTMF generator
AEC subband mode "ISDN"	DTMF detector, line echo canceller
AEC fullband mode two with number of taps >645	DTMF detector, line echo canceller
noise redcuction module with mode MD=10	DTMF detector, line echo canceller

A further restriction may occur because of the resource costs (MIPS) of the simultaneously applied modules. Each module currently in use takes up some resources. The MIPS a module needs is listed in table 20-34. The sum of resources all applied modules must never exceed the maximum DSP clock frequency in MHz. Thus, it may be necessary to restrict the length of the FIR filter of the echo cancellation unit if several other units are operating at the same time. A special case is the slowest clock rate for the DSP (16 MHz). For this clock rate, the sum of all modules must not exceed 14 MIPS.

Table 20-34 Module Weights

Module	Weight	Comment
Equalizer	0.88	Per equalizer
DTMF Generator	0.71	
Tone Generator	2.33 / 4.16	8 kHz / 16 kHz rate
Peak Detector	0.19	
Speakerphone	3.97	
AGCX	0.70	
AGCR	0.73	

Table 20-34 Module Weights

Module	Weight	Comment
Echo Cancellation Fullband mode one	25.12 / 21, 78 / 18. 45 / 15. 11 / 11.78	Number of taps: 639 / 511 / 383 / 255 / 127
Echo Cancellation Fullband mode two	18.16 / 17.14 / 16.12 / 15.09 / 14,07 / 8.95	Number of taps: 768 / 640 / 512 / 384 / 256 / 128 Adaptation window: 256 / 256 / 256 / 256 / 256 / 128 taps
Echo Cancellation Subband mode	13.37 / 12.92 / 12.35 / 11.32	Mode: Enhanced, ISDN, Normal, Reduced
Noise Reduction of the Echo Cancellation	1.74 / 0	Fullband mode / subband mode
Noise Reduction Unit	2.62	
AGC	0.81	
Universal Attenuator	0.06	
Universal Summation	0.09	
PIDD Data Interface	0.10	
Digital Interface	0.54	Channel 1 of CDI
Digital Interface	0.54	Channel 2 of CDI
Digital Interface	1.13	Channel 2+3 CDI
Analog Frontend IF	5.79	
Clock Tracking	0.18	
Miscellaneous	3.59	Always active

Table 20-35 gives an example of a typical application for INCA-D: Handsfree operation is in full-duplex mode for an echo length of maximal 64 ms (ie. 512 taps), noise reduction and all digital interfaces are active.

Table 20-35 Example Mode

Module	Weight	Comment
Equalizer	1.76	Both equalizers active
DTMF Generator	0	Not used
Tone Generator	0	Not used
Peak Detector	0	Not used
Speakerphone	3.97	
AGCX	0.70	

Table 20-35 Example Mode (cont'd)

Module	**Weight**	**Comment**
AGCR	0	Not used, either AGCX or AGCR is sufficient.
Echo Cancellation Fullband mode one	0	Not used
Echo Cancellation Fullband mode two	16.12	Number of taps: 512 Adaptation window: 256
Echo Cancellation Subband mode	0	Not used
Noise Reduction of the Echo Cancellation	1.74	
Noise Reduction Unit	0	Not used
AGC	0.81	
Universal Attenuator	0.06	
Universal Summation	0.09	
PIDD Data Interface	0.10	
Digital Interface	0.54	Channel 1 of CDI
Digital Interface	1.13	Channel 2+3 of CDI
Analog Frontend IF	5.79	
Clock Tracking	0.18	
Miscellaneous	3.59	
SUM	36.58	

20.4 DSP Register Description

The INCA-D has a single status register (read only) and an array of data registers (read/write). The purpose of the status register is to inform the microcontroller of important status changes of the DSP and to provide a handshake mechanism for data register reading or writing. If the DSP generates an interrupt, the status register contains the reason of the interrupt.

20.4.1 PIDD Command Register

PIDDCOM (DFAE$_H$)

15					0
Command	0	0	0	0	DSP register address

Register Read

0	1	0	1	0	0	0	0	DSP register address

Register Write

0	1	0	0	0	0	0	0	DSP register address

Interrupt Acknowledge

0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

20.4.2 Status Register

PIDDSTAT (DFB0$_H$)

15															**0**
RDY	ABT	COV	0	0	0	0	0	0	0	0	DTV	0	TG	0	0

RDY	**Ready**
	0: The last command (if any) is still in progress.
	1: The last command has been executed.
ABT	**Abort**
	0: No exception during operation
	1: An exception caused the DSP to abort any operation currently in progress. The ABT bit is cleared by writing the revision register. No other command is accepted by the DSP while ABT is set.
COV	**Codec Overflow**
	0: No problems
	1: Computational overload on DSP caused malfunction of AFE
DTV	**DTMF Tone Valid**
	0: No new DTMF code available
	1: New DTMF code available in DDCTL
TG	**Tone Generator Status**
	0: Tone Generator not running
	1: Tone Generator running

20.4.3 Hardware Configuration Register

PIDDHWCFG - **Hardware Configuration Register** ($DFB2_H$)

15															**0**
IDLE	0	ACT	0	0	0	0	IDIS	PD	0	0	0	0	0	0	0

IDLE **IDLE Mode indication (read only)**

0: DSP is not in IDLE mode

1: DSP is in IDLE mode

ACT **AFE Clock Tracking**

0: AFECLK tracking disabled

1: AFECLK tracking enabled

IDIS **IDLE Mode Disable**

0: IDLE mode enabled (if DSP tasks finished)

1: IDLE mode disabled

PD **Power Down indication (read only)**

0: DSP is not in power down mode

1: DSP is in power down mode

Note: The configuration register PIDDHWCFG can be read at any time. Please be aware that the configurations must be done before the modules that are affected by the configurations, are enabled. An easy way to ensure this is to write the configuration register when the DSP is in power down mode.

20.4.4 DSP address domain

00_H REV Revision

15															0
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

The revision register can be read only.

Note: A write access to the revision register does not change its content. It does, however, clear the ABT bit of the PIDDSTAT register.

01_H CCTL Chip Control

15															0
0	0	0	0	0	0	0	PD	0	0	0	0	0	0	0	0
Reset Value															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PD Power Down

0: DSP is in active mode

1: enter power-down mode

02_H INTM Mask Register

15															0
RDY	1	COV	0	0	0	0	0	0	0	0	DTV	0	TG	0	0
Reset Value															
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

If a bit of this register is reset (set to 0), the corresponding bit of the status register does not generate an interrupt.

If a bit is set (set to 1), an external interrupt can be generated by the corresponding bit of the status register.

03_H AFECTL Analog Front End Interface Control

15															0
TGL	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EN

Reset Value

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TGL **Tone Generator Directly Linked to ALS Driver**

0: TG not directly linked to loudspeaker driver

1: TG directly linked to loudspeaker driver

EN **Interface Enable**

0: AFE interface disabled

1: AFE interface enabled

04_H IFS1 Interface Select 1

15			0
HP	I1	I2	I3

Reset Value

0	0	0	0

The signal selection fields I1, I2 and I3 of IFS1 determine the outgoing signal of the analog interface.

The HP bit enables a high-pass for the incoming signal of the analog interface.

HP **High-Pass for S_1**

0: D/C Highpass

1: Highpass with 3 dB/oct. limit at 180 Hz

I1 **Input signal 1 for IG2**

I2 **Input signal 2 for IG2**

I3 **Input signal 3 for IG2**

Note: Unused inputs must be set to silence (S_0).

05_h IFG1 Interface Gain 1

15	0
0	IG1
Reset Value	
0	8192 (0 dB)

IFG1 is associated with the incoming signal of the analog interface.

IG1

In order to obtain a gain *G* the parameter IG1 can be calculated by the following formula:

$$\mathrm{IG1} = 32768 \times 10^{(G - 12.04\ \mathrm{dB})/20\ \mathrm{dB}}$$

Value range: [0x0001..0x7FFF] = [-78.3dB..12.04dB]

06_h IFG2 Interface Gain 2

15	0
0	IG2
Reset Value	
0	8192 (0 dB)

IFG2 is associated with the outgoing signal of the analog interface.

IG2 Gain of Amplifier IG2

In order to obtain a gain *G* the parameter IG2 can be calculated by the following formula:

$$\mathrm{IG2} = 32768 \times 10^{(G - 12.04\ \mathrm{dB})/20\ \mathrm{dB}}$$

Value range: [0x0001..0x7FFF] = [-78.3dB..12.04dB]

$0B_H$ CDICHN1 CDI Channel 1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	PCD		EN	PCM	0	0	0	0	0	0	0

Reset Value

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PCD PCM Code

10	9	Description
0	0	A-law
0	1	µ-law
1	0	linear
1	1	reserved

EN Enable Interface

0: Interface is disabled

1: Interface is enabled

PCM PCM Mode

0: 16 bit Linear Coding (two timeslots); setting of PCD is don't care

1: 8 bit PCM Coding (one timeslot)

$0C_H$ IFS3 Interface Select 3

15			0
HP	I1	I2	I3

Reset Value

0	0	0	0

The signal selection fields I1, I2 and I3 of IFS3 determine the outgoing signal of channel 1 of the CDI.

The HP bit enables a high-pass for the incoming signal of channel 1 of the CDI.

HP	**High-Pass for S_5**
	0: Disabled
	1: Enabled
I1	**Input signal 1 for S_6**
I2	**Input signal 2 for S_6**
I3	**Input signal 3 for S_6**

Note: Unused inputs must be set to silence (S_0).

$0D_H$ CDICHN2 CDI Channel 2

15														0
CS	0	0	0	0	PCD	EN	PCM	0	0	0	0	0	0	0

Reset Value

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

CS **Channel Split**

0: Single 16 bit or single 8 bit channel, depending on CDICHN2.PCM

1: Two adjacent 8 bit channels, CDICHN2.PCM must be set to 0

PCD **PCM Code**

10	9	Description[1]
0	0	A-law
0	1	μ-law
1	0	linear
1	1	reserved

[1] in case of two adjacent 8bit channels, both channels use the same code

EN **Enable Interface**

0: Interface is disabled

1: Interface is enabled

PCM **PCM Mode**

0: 16 Bit Linear Coding (two timeslots)

1: 8 Bit PCM Coding (one timeslot)

$0E_H$ IFS4 Interface Select 4

15			0
HP	I1	I2	I3

Reset Value

0	0	0	0

The signal selection fields I1, I2 and I3 of IFS4 determine the outgoing signal of channel 2 of the CDI. The HP bit enables a high-pass for the incoming signal of channel 2 of the CDI.

HP **High-Pass for S_7**

0: Disabled

1: Enabled

I1 **Input signal 1 for S_8**

I2 **Input signal 2 for S_8**

I3 **Input signal 3 for S_8**

Note: Unused inputs must be set to silence (S_0).

$0F_H$ IFG5 Interface Gain 5

15	0
ATT1	ATT2
Reset Value	
255 (0 dB)	255 (0 dB)

ATT1 Attenuation for I3 (Channel 1)

In order to obtain an attenuation *A* at I3 of channel 1 of the CDI, the parameter ATT1 can be calculated by the following formula:

$$\text{ATT1} = 256 \times 10^{(-A)/20\ \text{dB}}$$

Value range: [0x01..0xFF] = [48.2dB..0.03dB]

ATT2 Attenuation for I3 (Channel 2)

In order to obtain an attenuation *A* at I3 of channel 2 of the CDI, the parameter ATT2 can be calculated by the following formula:

$$\text{ATT2} = 256 \times 10^{(-A)/20\ \text{dB}}$$

Value range: [0x01..0xFF] = [48.2dB..0.03dB]

10_H UA Universal Attenuator

15				0
ATT	0	0	0	I1
Reset Value				
0 (96 dB)	0	0	0	0

ATT Attenuation for UA

For a given attenuation *A* [dB] the parameter ATT can be calculated by the following formula:

$$\text{ATT} = 256 \times 10^{-A/20\ \text{dB}}$$

Value range: [0x01..0xFF] = [48.2dB..0.03dB]

I1 Input Selection for UA

11_H DGCTL DTMF Generator Control

15												0
EN	MD	0	0	0	0	0	0	0	0	0	0	DTC

Reset Value

0	0	0	0	0	0	0	0	0	0	0	0	0

EN **Generator Enable**

0: Disabled

1: Enabled

MD **Mode**

0: raw

1: cooked

DTC **Dial Tone Code (cooked mode)**

.

3	2	1	0	Digit	Frequency
0	0	0	0	1	697/1209
0	0	0	1	2	697/1336
0	0	1	0	3	697/1477
0	0	1	1	A	697/1633
0	1	0	0	4	770/1209
0	1	0	1	5	770/1336
0	1	1	0	6	770/1477
0	1	1	1	B	770/1633
1	0	0	0	7	852/1209
1	0	0	1	8	852/1336
1	0	1	0	9	852/1477
1	0	1	1	C	852/1633
1	1	0	0	*	941/1209
1	1	0	1	0	941/1336
1	1	1	0	#	941/1477
1	1	1	1	D	941/1633

12_H DGF1 DTMF Generator Frequency 1

15	0
0	FRQ

No reset value.

FRQ Frequency of Generator 1

The parameter FRQ for a given frequency *f* [Hz] can be calculated by the following formula:

$$FRQ = 32768 \times \frac{f}{4000Hz}$$

Value range: [0x0000..0x7FFF] = [0Hz..3999.9Hz]

13_H DGF2 DTMF Generator Frequency 2

15	0
0	FRQ

No reset value.

FRQ Frequency of Generator 2

he parameter FRQ for a given frequency *f* [Hz] can be calculated by the following formula:

$$FRQ = 32768 \times \frac{f}{4000Hz}$$

Value range: [0x0000..0x7FFF] = [0Hz..3999.9Hz]

14_H DGL DTMF Generator Level

15			0
0	LEV2	0	LEV1

No reset value.

LEV2 Signal Level of Generator 2

In order to obtain a signal level *L* (relative to the PCM maximum value) for generator 2 the value of LEV2 can be calculated according to the following formula:

$$\text{LEV2} = 128 \times 10^{L/20\ \text{dB}}$$

Value range: [0x01..0x7F] = [-42.2dB..-0.07dB]

LEV1 Signal Level of Generator 1

In order to obtain a signal level *L* (relative to the PCM maximum value) for generator 1 the value of LEV1 can be calculated according to the following formula:

$$\text{LEV1} = 128 \times 10^{L/20\ \text{dB}}$$

Value range: [0x01..0x7F] = [-42.2dB..-0.07dB]

15_H DGATT DTMF Generator Attenuation

15	0
ATT2	ATT1

No reset value.

ATT2 Attenuation of Signal S_{10}

In order to obtain attenuation *A* the parameter ATT2 can be calculated by the formula:

$$\text{ATT2} = \begin{cases} 128 + 1024 \times 10^{-A/20\ \text{dB}} & ;A > 18,1\ \text{dB} \\ 128 \times 10^{-A/20\ \text{dB}} & ;A < 18,1\ \text{dB} \end{cases}$$

Value range: [0x01..0x7F] = [42.2dB..0.07dB] and [0x81..0xFF] = [60.2dB..18.1dB]

ATT1 Attenuation of Signal S_9

In order to obtain attenuation *A* the parameter ATT1 can be calculated by the formula:

$$\text{ATT1} = \begin{cases} 128 + 1024 \times 10^{-A/20\ \text{dB}} & ;A > 18,1\ \text{dB} \\ 128 \times 10^{-A/20\ \text{dB}} & ;A < 18,1\ \text{dB} \end{cases}$$

Value range: [0x01..0x7F] = [42.2dB..0.07dB] and [0x81..0xFF] = [60.2dB..18.1dB]

$1E_H$ IFS5 Interface Select 5

15			0
HP	I1	I2	I3
Reset Value			
0	0	0	0

The signal selection fields I1, I2 and I3 of IFS5 determine the outgoing signal of channel 3 of the CDI. The HP bit enables a high-pass for the incoming signal of channel 3 of the CDI.

HP **High-Pass for S_{23}**

0: Disabled

1: Enabled

I1 **Input signal 1 for S_{24}**

I2 **Input signal 2 for S_{24}**

I3 **Input signal 3 for S_{24}**

Note: Unused inputs must be set to silence (S_0)..

$1F_H$ IFG6 Interface Gain 6

15								0
ATT3	0	0	0	0	0	0	0	0
Reset Value								
255 (0 dB)	00 (0 dB)							

ATT3 **Attenuation for I3 (Channel 3)**

In order to obtain an attenuation *A* at I3 of channel 3 of the CDI, the parameter ATT3 can be calculated by the following formula:

$$\mathrm{ATT3} = 256 \times 10^{(-A)/20\ \mathrm{dB}}$$

Value range: [0x01..0xFF] = [48.2dB..0.03dB]

20_H SUMCTL Universal Summation Point Control

15 … **0**

EN	MD	0	0	0	0	I1	I2

Reset Value

0	0	0	0	0	0	0	0

EN **Universal Summation Point Enable**

0: Disabled

1: Enabled

I2 **Input signal selection**

I1 **Input signal selection**

21_H SUMATT Universal Aummation Input Attenuation

15 … **0**

ATT1	ATT2

Reset Value

255 (0 dB)	255 (0 dB)

ATT1 **Attenuation for I3 (Channel 1)**

In order to obtain an attenuation *A* at I1 of the universal summation point, the parameter ATT1 can be calculated by the following formula:

$$\mathrm{ATT1} = 256 \times 10^{(-A)/20\ \mathrm{dB}}$$

Value range: [0x01..0xFF] = [48.2dB..0.03dB]

ATT2 **Attenuation for I3 (Channel 2)**

In order to obtain an attenuation *A* at I2 of the niversal summation point, the parameter ATT2 can be calculated by the following formula:

$$\mathrm{ATT2} = 256 \times 10^{(-A)/20\ \mathrm{dB}}$$

Value range: [0x01..0xFF] = [48.2dB..0.03dB]

25_H LECCTL Line Echo Cancellation Control

15						0
EN	CM	AS	SP	0	I1	I2
Reset Value						
0	0	0	0	0	0	0

EN **Enable**

0: Disabled

1: Enabled

CM **Line Echo Canceller Mode**

14	13	Line echo canceller mode
0	0	Normal mode (4ms)
0	1	Superior mode (4ms)
1	0	Extended mode (8ms)
1	1	Reserved

AS **Adaption Stop**

0: Adaption enabled

1: Adaption stopped

SP **Adaption Speed**

0: Fast adaptation (extended mode only), ie. only one half (upper or lower) of the coefficients is adapted alternately

1: Slow adaptation (extended mode only), ie. the complete cofficient set is adapted

I1 **Input signal selection for I_1**

I2 **Input signal selection for I_2**

26_H LECLEV Minimal Signal Level for Line Echo Cancellation

15	0
0	MIN

No reset value.

MIN

The parameter MIN for a minimal signal level *L* (dB) can be calculated by the following formula:

$$\text{MIN} = \frac{512 \times (96.3 + L)}{5 \times \log 2}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0.03dB]

27_H LECATT Externally Provided Attenuation

15	0
0	ATT

No reset value.

ATT

The parameter ATT for an externally provided attenuation *A* (dB) can be calculated by the following formula:

$$\text{ATT} = \frac{512 \times A}{5 \times \log 2}$$

Value range: [0x0000..0x7FFF] = [0dB..96.3dB]

ATT has a slightly different meaning in normal and in superior mode. In normal mode, it represents just the externally provided attenunation while in superior mode, it represents the externally provided attenuation minus a threshold. The formula above holds in both cases.

In superior mode, the parameter ATT is implemented in two's complement.

In normal mode, the parameter ATT is not allowed to be negative, i.e., the MSB (bit 15) must be 0.

28_H LECMGN Margin for Double Talk Detection

15	0
0	MGN

No reset value.

MGN

The parameter MGN for a margin of *L* (dB) can be calculated by the following formula:

$$MGN = \frac{512 \times L}{5 \times \log 2}$$

Value range: [0x0000..0x7FFF] = [0dB..96.3dB]

Note: MGM has a different meaning in normal and in superior mode. The formula above holds in any mode, though.

29_H DDCTL DTMF Detector Control

15							0
EN	0	0	I1	0	0	0	DTC[1]

Reset Value

0	0	0	0	0	0	0	-[2]

[1] The DTC code remains valid until a new DTMF tone has been detected.
[2] undefined

EN Enable DTMF tone detection

0: The DTMF detection is disabled

1: The DTMF detection is enabled

I1 Input signal selection

DTC DTMF Tone Code

4	3	2	1	0	Frequency	Digit
1	0	0	0	0	941 / 1633	D
1	0	0	0	1	697 / 1209	1
1	0	0	1	0	697 / 1336	2
1	0	0	1	1	697 / 1477	3
1	0	1	0	0	770 / 1209	4
1	0	1	0	1	770 / 1336	5
1	0	1	1	0	770 / 1477	6
1	0	1	1	1	852 / 1209	7
1	1	0	0	0	852 / 1336	8
1	1	0	0	1	852 / 1477	9
1	1	0	1	0	941 / 1336	0
1	1	0	1	1	941 / 1209	*
1	1	1	0	0	941 / 1477	#
1	1	1	0	1	697 / 1633	A
1	1	1	1	0	770 / 1633	B
1	1	1	1	1	852 / 1633	C

$2A_H$ DDTW DTMF Detector Signal Twist

15	0
0	TWIST

No reset value.

TWIST Signal twist for DTMF tone

In order to obtain a minimal signal twist *T* the parameter TWIST can be calculated by the following formula:

$$\text{TWIST} = 32768 \times 10^{(-0.5\text{ dB} - \text{T})/10\text{ dB}}$$

Value range: [0x1000..0x5000] = [8.53dB..1.54dB]

Note: TWIST must be in the range [4096,20480]

$2B_H$ DDLEV DTMF Detector Minimum Signal Level

15										0
1	1	1	1	1	1	1	1	1	1	MIN

No reset value.

MIN Minimum Signal Level

5	4	3	2	1	0	Description
0	0	1	1	1	0	-50 dB
0	0	1	1	1	1	-49 dB
...	...	...	...	...	...	...
1	0	0	0	0	1	-31 dB
1	0	0	0	1	0	-30 dB

Note: Values outside the given range are reserved and must not be used.

$2C_H$ FCFCTL1 Equalizer 1 Control

15 … 0

EN	0	ADR	0	0	0	I1

Reset Value

0	0	0	0	0	0	0

EN **Enable equalizer 1**

0: The equalizer is disabled

1: The equalizer is enabled

ADR **Coefficient address**

13	12	11	10	9	8	Coefficient
0	0	0	0	0	0	A1
0	0	0	0	0	1	A2
0	0	0	0	1	0	A3
0	0	0	0	1	1	A4
0	0	0	1	0	0	A5
0	0	0	1	0	1	A6
0	0	0	1	1	0	A7
0	0	0	1	1	1	A8
0	0	1	0	0	0	A9
0	0	1	0	0	1	B2
0	0	1	0	1	0	B3
0	0	1	0	1	1	B4
0	0	1	1	0	0	B5
0	0	1	1	0	1	B6
0	0	1	1	1	0	B7
0	0	1	1	1	1	B8
0	1	0	0	0	0	B9
0	1	0	0	0	1	C1
0	1	0	0	1	0	D1
0	1	0	0	1	1	D2
0	1	0	1	0	0	D3
0	1	0	1	0	1	D4
0	1	0	1	1	0	D5

13	12	11	10	9	8	Coefficient
0	1	0	1	1	1	D6
0	1	1	0	0	0	D7
0	1	1	0	0	1	D8
0	1	1	0	1	0	D9
0	1	1	0	1	1	D10
0	1	1	1	0	0	D11
0	1	1	1	0	1	D12
0	1	1	1	1	0	D13
0	1	1	1	1	1	D14
1	0	0	0	0	0	D15
1	0	0	0	0	1	D16
1	0	0	0	1	0	D17
1	0	0	0	1	1	C2

I1 Input signal selection

$2D_H$ FCFCOF1 Equalizer 1 Coefficient Data

15	0
V	

No reset value.

V Coefficient value

For the coefficients A_1-A_9, B_2-B_9 and D_1-D_{17}, the following formula can be used, where V denotes the coefficient value of the coefficient *c*:

$$V = 32768 \times c \qquad ; -1 \leq c < 1$$

For the coefficients C_1 and C_2, the following formula can be used, where V denotes the coefficient value of the coefficient *c*:

$$V = 128 \times c \qquad ; 1 \leq c < 256$$

$2E_H$ FCFCTL2 Equalizer 2 Control

15						0
EN	0	ADR	0	0	0	I1

Reset Value

0	0	0	0	0	0	0

EN **Enable equalizer 1**

0: The equalizer is disabled

1: The equalizer is enabled

ADR **Coefficient address**

13	12	11	10	9	8	Coefficient
0	0	0	0	0	0	A1
0	0	0	0	0	1	A2
0	0	0	0	1	0	A3
0	0	0	0	1	1	A4
0	0	0	1	0	0	A5
0	0	0	1	0	1	A6
0	0	0	1	1	0	A7
0	0	0	1	1	1	A8
0	0	1	0	0	0	A9
0	0	1	0	0	1	B2
0	0	1	0	1	0	B3
0	0	1	0	1	1	B4
0	0	1	1	0	0	B5
0	0	1	1	0	1	B6
0	0	1	1	1	0	B7
0	0	1	1	1	1	B8
0	1	0	0	0	0	B9
0	1	0	0	0	1	C1
0	1	0	0	1	0	D1
0	1	0	0	1	1	D2
0	1	0	1	0	0	D3
0	1	0	1	0	1	D4
0	1	0	1	1	0	D5

13	12	11	10	9	8	Coefficient
0	1	0	1	1	1	D6
0	1	1	0	0	0	D7
0	1	1	0	0	1	D8
0	1	1	0	1	0	D9
0	1	1	0	1	1	D10
0	1	1	1	0	0	D11
0	1	1	1	0	1	D12
0	1	1	1	1	0	D13
0	1	1	1	1	1	D14
1	0	0	0	0	0	D15
1	0	0	0	0	1	D16
1	0	0	0	1	0	D17
1	0	0	0	1	1	C2

I1 Input signal selection

$2F_H$ FCFCOF2 Equalizer 2 Coefficient Data

15	0
V	

No reset value.

V Coefficient value

For the coefficients A_1-A_9, B_2-B_9 and D_1-D_{17}, the following formula can be used, where V denotes the coefficient value of the coefficient *c*:

$$V = 32768 \times c \qquad ; -1 \le c < 1$$

For the coefficients C_1 and C_2, the following formula can be used, where V denotes the coefficient value of the coefficient *c*:

$$V = 128 \times c \qquad ; 1 \le c < 256$$

30_H TGCTL Tone Generator Control

15													0
E16	0	0	0	0	0	0	0	0	CGM	DT	BGM	SM	WF

Reset Value

0	0	0	0	0	0	0	0	0	0	0	0	0	0

E16 **Enable 16 kHz Sampling for Tone Generation**

0: Disabled

1: Enabled; generated frequency is twice as high as normal

CGM **Control Generator Mode**

6	5	Description
0	0	Tone Generator off
0	1	Tone Generator on
1	-	Tone Generator enabled/disabled by Control Generator

DT **Dual Tone**

0: F4 not added (option 1)

1: F4 added (option 2)

BGM **Beat Generator Mode**

3	2	Description
0	0	Continuous Tone F1
0	1	Continuous Tone F2
1	0	two tone cadence
1	1	three tone sequence

SM **Stop Mode**

0: Immediate

1: Controlled

WF **Waveform**

0: Sine Wave

1: Square Wave

31_H TGTON Tone Generator Time TON

15	0
TM	TE

No reset value.

TM Mantissa of TON

The mantissa TM for a time *t* ([ms]) can be calculated by the following formula:

$$TM = 8\frac{t}{2^{TE}}$$

Value range: [0x0001..0xFFFF] = [1ms..15min]

TE Exponent of TON

The exponent TE for a time *t* ([ms]) can be calculated by the following formula:

$$TE = \log_2 t$$

Note: TE > 0

32_H TGTOFF Tone Generator Time TOFF

15		0
TM		TE

No reset value.

TM Mantissa of TOFF

The mantissa TM for a time *t* ([ms]) can be calculated by the following formula:

$$TM = 8\frac{t}{2^{TE}}$$

Value range: [0x0001..0xFFFF] = [1ms..15min]

TE Exponent of TOFF

The exponent TE for a time *t* ([ms]) can be calculated by the following formula:

$$TE = \log_2 t$$

Note: TE > 0

33_H TGT1 Tone Generator Time T1

15	0
TIME	

No reset value.

TIME

The parameter TIME for a time *t* ([ms]) can be calculated by the following formula:
Value range: [0x0001..0xFFFF] = [0.125ms..8191.875ms]

$$TIME = 8t$$

Note: If the sampling frequency is set to 16 KHz (TGCL.E16 = 1), the value for the requested time (TIME) has to be doubled, i.e. in that case is TIME=16t.

34_H TGF1 Tone Generator Frequency F1

15	0
0	F

No reset value.

F Frequency

The parameter F for a frequency *f* ([Hz]) can be calculated by the following formula:

Value range: [0x0000..0x7FFF] = [0Hz..3999.9Hz]

$$F = 8.192 \times f$$

Note: If a sine waveform is selected, the frequency f is output. If a square waveform is selected the next lower frequency that is a harmonic frequency is output.

Note: If the sampling frequency is set to 16 KHz (TGCL.E16 = 1), the frequency is halfed

35_H TGG1 Tone Generator Gain G1

15	0
0	G

No reset value.

G Gain

The parameter G for a gain *g* ([dB]) can be calculated by the following formula:

$$F = 32768 \times 10^{g/20}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0dB]

36_H TGT2 Tone Generator Time T2

15 … 0
TIME

No reset value.

TIME

The parameter TIME for a time *t* ([ms]) can be calculated by the following formula:

Value range: [0x0001..0xFFFF] = [0.125ms..8191.875ms]

$$\mathrm{TIME} = 8t$$

Note: If the sampling frequency is set to 16 KHz (TGCL.E16 = 1), the value for the requested time (TIME) has to be doubled, i.e. in that case is TIME=16t.

37_h TGF2 Tone Generator Frequency F2

15	… 0
0	F

No reset value.

F Frequency

The parameter F for a frequency *f* ([Hz]) can be calculated by the following formula:

Value range: [0x0000..0x7FFF] = [0Hz..3999.9Hz]

$$F = 8.192 \times f$$

Note: If a sine waveform is selected, the frequency f is output. If a square waveform is selected the next lower frequency that is a harmonic frequency is output.

Note: If the sampling frequency is set to 16 KHz (TGCL.E16 = 1), the frequency is halfed

38_H TGG2 Tone Generator Gain G2

15	0
0	G

No reset value.

G **Gain**

The parameter G for a gain *g* ([dB]) can be calculated by the following formula:

$$F = 32768 \times 10^{g/20}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0dB]

39_H TGT3 Tone Generator Time T3

15	0
TIME	

TIME

The parameter TIME for a time *t* ([ms]) can be calculated by the following formula:

Value range: [0x0001..0xFFFF] = [0.125ms..8191.875ms]

$$TIME = 8t$$

Note: If the sampling frequency is set to 16 KHz (TGCL.E16 = 1), the value for the requested time (TIME) has to be doubled, i.e. in that case is TIME=16t.

$3A_h$ TGF3 Tone Generator Frequency F3

15	0
0	F

No reset value.

F **Frequency**

The parameter F for a frequency *f* ([Hz]) can be calculated by the following formula:

$$F = 8.192 \times f$$

Value range: [0x0000..0x7FFF] = [0Hz..3999.9Hz]

Note: If a sine waveform is selected, the frequency f is output. If a square waveform is selected the next lower frequency that is a harmonic frequency is output.

Note: If the sampling frequency is set to 16 KHz (TGCL.E16 = 1), the frequency is doubled.

$3B_H$ TGG3 Tone Generator Gain G3

15	0
0	G

No reset value.

G Gain

The parameter G for a gain *g* ([dB]) can be calculated by the following formula:

$$F = 32768 \times 10^{g/20}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0dB]

$3C_H$ TGF4 Tone Generator Frequency F4

15	0
0	F

No reset value.

F Frequency

The parameter F for a frequency *f* ([Hz]) can be calculated by the following formula:

$$F = 8.192 \times f$$

Value range: [0x0000..0x7FFF] = [0Hz..3999.9Hz]

Note: If a sine waveform is selected, the frequency f is output. If a square waveform is selected the next lower frequency that is a harmonic frequency is output.

$3D_H$ TGG4 Tone Generator Gain G4

15	0
0	G

No reset value.

G Gain

The parameter G for a gain *g* ([dB]) can be calculated by the following formula:

$$F = 32768 \times 10^{g/20}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0dB]

$3E_H$ TGGO1 Tone Generator Gain Output 1

15	0
0	G

No reset value.

G Gain

The parameter G for a gain *g* ([dB]) can be calculated by the following formula:

$$F = 32768 \times 10^{g/20}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0dB]

$3F_H$ TGGO2 Tone Generator Gain Output 2

15	0
0	G

No reset value.

G Gain

The parameter G for a gain *g* ([dB]) can be calculated by the following formula:

$$F = 32768 \times 10^{g/20}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0dB]

40_H TGGS Tone Generator Saturate

15	0
0	GS

No reset value.

GS Saturation Gain

The parameter G for a gain *g* ([dB]) can be calculated by the following formula:

$$F = 32768 \times 10^{\frac{g - 24dB}{20}}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..24dB]

41_H PIDCTL PIDD Control

15											0
EN	ENE	0	0	0	0	0	0	0	0	0	I1
Reset Value											
0	0	0	0	0	0	0	0	0	0	0	0

EN **PIDD Data Transfer Enable**

0: The data transfer is disabled

1: The data transfer is enabled

ENE **Transfer of Bytes only**

0: 16 bit linear are transferred to and from the DSP

1: 8 bit linear are transferred to and from the DSP

I1 **Input signal selection**

45_H PDCTL Peak Detector Control

15											0
EN	MM	0	0	0	0	0	0	0	0	0	I1
Reset Value											
0	0	0	0	0	0	0	0	0	0	0	0

EN **Peak Detector Enable**

0: Disabled

1: Enabled

MM **Min/Max**

0: Maximum

1: Minimum

I1 **Input signal selection**

46_H PDDATA Peak Detector Data

15	0
DATA	

No reset value.

DATA

Maximum or minimum value of signal since last read access.

Note: This register can only be read.

47H SPSCTL Speakerphone State Control

15															0
0	0	0	0	0	0	0	0	0	0	0	0	0	EN	SP1	SP0
Reset Value															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

EN **Enable SPS**

0: Disable

1: Enable

Table 20-36 SPS Encoding

SPS_0	SPS_1	Echo Suppression Unit State
0	0	no echo suppression operation
0	1	receive
1	0	transmit
1	1	idle

50_H ACR AFE Configuration Register

15															0
TRO	ALF2	ALFS	0	0	0	EADC	EDAC	EBIA	0	EHOP 1	EHON 1	EHOP 2	EHON 2	ELSP	ELSN

Reset Value

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TRO **Tone Ringing only**

0:Voice and ringing signals are added and given to the loudspeaker driver
1:The voice path is disconneced, ie. only the ringing signal is given to the loudspeaker

ALH2 **Analog Loop via AHO2**

0: Disabled
1: Enabled

ALFS **Analog Loop via AHO1 and ALS**

0: Disabled
1: Enabled

EADC **Enable A/D Control**

0: A/D is in power down mode
1: Enable A/D is active

EDAC **Enable D/A Control**

0: D/A and POFI are in power down mode
1: Enable D/A and POFI

EBIA **Enable BIAS (Reference Voltage Source)**

0: AFE (BIAS) is power down. All other parts of AFE must be powered down.
1: Enable bias. (Must be enabled before any other part of AFE is enabled.

EHOPn **Enable HOPn** Amplifier

0: Disable HOP amplifier (power down, output high impedance)

1: Enable HOP amplifier

EHONn	**Enable HONn** Amplifier

0: Disable HON amplifier (power down, output high impedance)
1: Enable HON amplifier

ELSP	**Enable LSP** Amplifier

0: Disable LSP amplifier
1: Enable LSP amplifier

ELSN	**Enable LSN** Amplifier

0: Disable LSN amplifier
1: Enable LSN amplifier

51_H ATCR AFE Transmit Configuration Register

15												0
0	0	0	0	0	0	0	0	MIC	0	OVRE	AIMX	
Reset Value												
0	0	0	0	0	0	0	0	0	0	0	0	0

MIC Microphone Amplifier (AMI) Control

Bit 7	6	5	4	Selected Mode
0	0	0	0	AMI and PREFI is in **power-down** mode
0	0	0	1	0 dB amplification
0	0	1	0	6 dB amplification
0	0	1	1	12 dB amplification
0	1	0	0	18 dB amplification
0	1	0	1	24 dB amplification
0	1	1	0	30 dB amplification
0	1	1	1	36 dB amplification
1	0	0	0	42 dB amplification
1	1	1	1	reserved

OVRE Overwrite Enable

0: Normal operation: LSC determines the amplification for the loudspeaker output
1: Loudhearing mode: LSC or OVLS determine the amplification for the loudspeaker output according to the speakerphone state

AIMX Analog Input Multiplexer

Bit 1	0	Selected Input
0	0	AMI is connected to the pins MIP1/MIN1 (differential input)
0	1	AMI is connected to the pins MIP2/MIN2 (differential input)
1	0	AMI is connected to the pins MIP3/MIN3 (differential input)
1	1	reserved

52_H ARCR AFE Receive Configuration Register

15			0
HOC2	LSC	HOC1	OVLS

Reset Value

0	0	0	0

HOCn **Handset Output Amplifier (AHO) Control**

Bit 3	2	1	0	Selected Mode
0	0	0	0	AHO is in **power-down** mode
0	0	0	1	2.5 dB amplification
0	0	1	0	– 0.5 dB amplification
0	0	1	1	– 3.5 dB amplification
0	1	0	0	– 6.5 dB amplification
0	1	0	1	– 9.5 dB amplification
0	1	1	0	– 12.5 dB amplification
0	1	1	1	– 15.5 dB amplification
1	0	0	0	– 18.5 dB amplification
1	0	0	1	– 21.5 dB amplification
1	1	1	1	reserved

LSC, OVLS **Loudspeaker Amplifier (ALS) Control**

Bit 3	2	1	0	Selected Mode
0	0	0	0	ALS is in **power-down** mode
0	0	0	1	11.5 dB amplification
0	0	1	0	8.5 dB amplification
0	0	1	1	5.5 dB amplification
0	1	0	0	2.5 dB amplification
0	1	0	1	– 0.5 dB amplification
0	1	1	0	– 3.5 dB amplification
0	1	1	1	– 6.5 dB amplification
1	0	0	0	– 9.5 dB amplification
1	0	0	1	– 12.5 dB amplification
1	0	1	0	– 15.5 dB amplification
1	0	1	1	– 18.5 dB amplification
1	1	0	0	– 21.5 dB amplification
1	1	0	1	– 24.5 dB amplification (only for TRL = ’1’)
1	1	1	0	– 27.5 dB amplification (only for TRL = ’1’)
1	1	1	1	reserved

58_H AGCCTL AGC Control

15							0
EN	0	0	0	0	0	I1	I2
Reset Value							
0	0	0	0	0	0	0	0

EN **Enable**

0: Disabled

1: Enabled

I1 **Input signal selection for I_1**

I2 **Input signal selection for I_2**

59_H AGCATT Automatic Gain Control Attenuation

15	0
0	ATT

No reset value.

ATT

The parameter ATT for an attenuation *A* ([dB]) can be calculated by the following formula:

$$\mathrm{ATT} = 32768 \times 10^{\frac{-A}{20}}$$

Value range: [0x0001..0x7FFF] = [96.3dB..0dB]

$5A_H$ AGC1 Automatic Gain Control 1

15	0
COM	AG_INIT

No reset value.

COM

The parameter COM for a signal level *L* ([dB]) can be calculated by the following formula:

$$\mathrm{COM} = \begin{cases} 128 + 10^{\frac{L + 66,22}{20}} & ;L < -42,14\ \mathrm{dB} \\ 10^{\frac{L + 42,14}{20}} & ;L > -42,14\ \mathrm{dB} \end{cases}$$

Value range: [0x01..0x7F] = [-42.14dB..-0.06dB] and [0x81..0xFF] = [-66.2dB..-24.1dB]

AG_INIT

In order to obtain an initial gain *G* ([db]) the parameter AG_INIT can be calculated by the following formula:

$$\mathrm{AG\_INIT} = \begin{cases} 128 + 10^{\frac{G + 18,06}{20}} & ;G < 6,02\ \mathrm{dB} \\ 10^{\frac{G - 6,02}{20}} & ;G > 6,02\ \mathrm{dB} \end{cases}$$

Value range: [0x01..0x7F] = [6.02dB..48.1dB] and [0x81..0xFF] = [-18.06dB..24.01dB]

$5B_H$ AGC2 Automatic Gain Control 2

15	0
SPEEDL	SPEEDH

No reset value.

SPEEDL

The parameter SPEEDL for a multiplication factor *M* is given by the following formula:

$$\text{SPEEDL} = M \times 8192$$

Value range: [0x00..0xFF]

SPEEDH

The parameter SPEEDH for a multiplication factor *M* is given by the following formula:

$$\text{SPEEDH} = M \times 256$$

Value range: [0x00..0xFF]

$5C_H$ AGC3 Automatic Gain Control 3

15	0
AG_GAIN	AG_ATT

No reset value.

AG_GAIN

The parameter AG_GAIN for a gain *G* ([dB]) can be calculated by the following formula:

$$\text{AG\_GAIN} = \begin{cases} 128 + 10^{\frac{G+18.06}{20}} & ;G < 24\text{ dB} \\ 10^{\frac{G-6.02}{20}} & ;G > 24\text{ dB} \end{cases}$$

Value range: [0x01..0x7F] = [6.02dB..48.1dB] and [0x81..0xFF] = [-18.06dB..24.01dB]

AG_ATT

The parameter AG_ATT for an attenuation *A* ([dB]) can be calculated by the following formula:

$$\mathrm{AG\_ATT} = 10^{\frac{A + 42.14}{20}}$$

Value range: [0x01..0x7F] = [-0.,06dB..-42.1dB]

$5D_H$ AGC4 Automatic Gain Control 4

15	0
DEC	LIM

No reset value.

DEC

The parameter DEC for a time constant *t* ([1/ms]) is given by the following formula:

$$DEC = \frac{256}{t}$$

Value range: [0x01..0x7F] = [256ms..4ms] and [0x00] = infinite

LIM

The parameter LIM for a signal level *L* ([dB]) can be calculated by the following formula:

$$LIM = \begin{cases} 128 + 10^{\frac{L + 90.3}{20}} & ;L < -50\text{ dB} \\ 10^{\frac{L + 66.22}{20}} & ;-25 > L > -50\text{ dB} \end{cases}$$

Value range: [0x01..0x7F] = [-66.2dB..-24.1dB] and [0x81..0xFF] = [-90.3dB..-48.2dB]

$5E_H$ AGC5 Automatic Gain Control 5

15								0
0	0	0	0	0	0	0	0	LP

No reset value.

LP

The parameter LP for a time constant *t* ([ms]) is given by the following formula:

$$LP = \frac{-16}{t}$$

Value range: [0x80..0xFC] = [0.125ms..4ms]

Note: The value for LP should be in the range of 80_h to FC_h (i.e. t<4ms) in order to avoid that the AGC becomes instable.

Note: The value for LP has be entered in 2-complement notation.

60_h SCTL Speakerphone Control

15													0
ENS	QU	EWF	0	NR	0	MD	SDR	SDX	ERD	0	AGR	AGX	0

Reset Value

0	0	0	0	0	0	0	0	0	0	0	0	0	0

ENS **Enable Echo Suppression**

0: The echo suppression unit is disabled

1: The echo suppression unit is enabled

QU **Echo Cancellation Quality Mode**

14	13	12	Description
0	0	0	Echo cancellation disabled (half duplex)
0	0	1	reserved
0	1	0	Fullband mode one (similar to PSB 2170 Version 1.1)
0	1	1	Fullband mode two
1	0	0	Subband, reduced filter length
1	0	1	Subband, analog line mode
1	1	0	Subband, ISDN mode
1	1	1	Subband, enhanced mode

EWF **Enable Wiener Filter**

0: The Wiener filter is disabled

1: The Wiener filter is enabled (in subband mode only)

NR **Noise Reduction Enable**

0: Noise reduction disabled

1: Noise reduction enabled

MD **Mode**

0: Speakerphone mode

1: Loudhearing mode

SDR	**Signal Source of SDR**
	0: after AGCR
	1: before AGCR
SDX	**Signal Source of SDX**
	0: after AGCX
	1: before AGCX
ERD	**Enable Reduced Delay**
	0: Normal operation in fullband mode two
	1: The delay of fullband mode two is reduced
AGR	**AGCR Enable**
	0: AGCR disabled
	1: AGCR enabled
AGX	**AGCX Enable**
	0: AGCX disabled
	1: AGCX enabled

62_h SSRC1 Speakerphone Source 1

15							0
0	0	0	0	0	0	I1	I2
Reset Value							
0	0	0	0	0	0	0	0

I1 **Input Signal Selection (Acoustic Source 1)**

I2 **Input Signal Selection (Acoustic Source 2)**

63_h SSRC2 Speakerphone Source 2

15							0
0	0	0	0	0	0	I3	I4
Reset Value							
0	0	0	0	0	0	0	0

I3 **Input Signal Selection (Line Source 1)**

I4 **Input Signal Selection (Line Source 2)**

64_h SSDX1 Speech Detector (Transmit) 1

15			0
0	LP2L	0	LIM

Reset Value

$1B3A_h$

LP2L

The parameter LP2L for a saturation level *L* ([dB]) can be calculated by the following formula:

$$LP2L = \frac{2 \times L}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..95.6dB]

Reset and default value for *L*: 20 dB

LIM

The parameter LIM for a minimum signal level *L* ([dB], relative to PCM max. value) can be calculated by the following formula:

$$LIM = \frac{2 \times (96.3 + L)}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [-96.3dB..-0.72dB]

Reset and default value for *L*: -52 dB

65_h SSDX2 Speech Detector (Transmit) 2

15		0
LP1	0	OFF

Reset Value

1006_h

LP1

The parameter LP1 for a time *t* ([ms]) can be calculated by the following formula:

$$LP1 = \begin{cases} 64/t & ;0.5 < t < 64 \\ 128 + 2048/t & ;16.2 < t < 2048 \end{cases}$$

Value range: [0x01..0x7F] = [64ms..0.5ms] and [0x81..0xFF] = [2048ms..16.13ms]

Reset and default value for *t*: 4 ms

OFF

The parameter OFF for a level offset of *O* ([dB]) can be calculated by the following formula:

$$OFF = \frac{2 \times O}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..95.6dB]

Reset and default value for *O*: 4.5 dB

66_h SSDX3 Speech Detector (Transmit) 3

15	0
PDN	LP2N

Reset Value

$C0C0_h$

PDN

The parameter PDN for a time *t* ([ms]) can be calculated by the following formula:

$$PDN = \begin{cases} 64/t & ;0.5 < t < 64 \\ 128 + 2048/t & ;16.2 < t < 2048 \end{cases}$$

Value range: [0x01..0x7F] = [64ms..0.5ms] and [0x81..0xFF] = [2048ms..16.13ms]

Reset and default value for *t*: 32 ms

LP2N

The parameter LP2N for a time *t* ([ms]) can be calculated by the following formula:

$$LP2N = \begin{cases} 64/t & ;0.5 < t < 64 \\ 128 + 2048/t & ;16.2 < t < 2048 \end{cases}$$

Value range: [0x01..0x7F] = [64ms..0.5ms] and [0x81..0xFF] = [2048ms..16.13ms]

Reset and default value for *t*: 32 ms

67_h SSDX4 Speech Detector (Transmit) 4

15		0
PDS	0	LP2S

Reset Value

9428_h

PDS

The parameter PDS for a time *t* ([ms]) can be calculated by the following formula:

$$PDS = \begin{cases} 64/t & ;0.5 < t < 64 \\ 128 + 2048/t & ;16.2 < t < 2048 \end{cases}$$

Value range: [0x01..0x7F] = [64ms..0.5ms] and [0x81..0xFF] = [2048ms..16.13ms]

Reset and default value for *t*: 102 ms

LP2S

The parameter LP2S for a time *t* ([ms]) can be calculated by the following formula:

$$LP2S = \frac{262144}{t}$$

Value range: [0x01..0x7F] = [262144ms..2064ms]

Reset and default value for *t*: 6.5 s

68_h SSDR1 Speech Detector (Receive) 1

15			0
0	LP2L	0	LIM

Reset Value
1437_h

LP2L

The parameter LP2L for a saturation level *L* ([dB]) can be calculated by the following formula:

$$LP2L = \frac{2 \times L}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..95.6dB]

Reset and default value for *L*: 15 dB

LIM

The parameter LIM for a minimum signal level *L* ([dB], relative to PCM max. value) can be calculated by the following formula:

$$LIM = \frac{2 \times (96.3 + L)}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [-96.3dB..-0.72dB]

Reset and default value for *L*: -55 dB

69_h SSDR2 Speech Detector (Receive) 2

15		0
LP1	0	OFF

Reset Value

1006_h

LP1

The parameter LP1 for a time *t* ([ms]) can be calculated by the following formula:

$$LP1 = \begin{cases} 64/t & ;0.5 < t < 64 \\ 128 + 2048/t & ;16.2 < t < 2048 \end{cases}$$

Value range: [0x01..0x7F] = [64ms..0.5ms] and [0x81..0xFF] = [2048ms..16.13ms]

Reset and default value for *t*: 4 ms

OFF

The parameter OFF for a level offset of *O* ([dB]) can be calculated by the following formula:

$$OFF = \frac{2 \times O}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..95.6dB]

Reset and default value for *O*: 4.5 dB

$6A_h$ SSDR3 Speech Detector (Receive) 3

15	0
PDN	LP2N

Reset Value

$C0C0_h$

PDN

The parameter PDN for a time *t* ([ms]) can be calculated by the following formula:

$$PDN = \begin{cases} 64/t & ;0.5 < t < 64 \\ 128 + 2048/t & ;16.2 < t < 2048 \end{cases}$$

Value range: [0x01..0x7F] = [64ms..0.5ms] and [0x81..0xFF] = [2048ms..16.13ms]

Reset and default value for *t*: 32 ms

LP2N

The parameter LP2N for a time *t* ([ms]) can be calculated by the following formula:

$$LP2N = \begin{cases} 64/t & ;0.5 < t < 64 \\ 128 + 2048/t & ;16.2 < t < 2048 \end{cases}$$

Value range: [0x01..0x7F] = [64ms..0.5ms] and [0x81..0xFF] = [2048ms..16.13ms]

Reset and default value for *t*: 32 ms

$6B_h$ SSDR4 Speech Detector (Receive) 4

15		0
PDS	0	LP2S

Reset Value

9428_h

PDS

The parameter PDS for a time *t* ([ms]) can be calculated by the following formula:

$$PDS = \begin{cases} 64/t & ;0.5 < t < 64 \\ 128 + 2048/t & ;16.2 < t < 2048 \end{cases}$$

Value range: [0x01..0x7F] = [64ms..0.5ms] and [0x81..0xFF] = [2048ms..16.13ms]

Reset and default value for *t*: 102 ms

LP2S

The parameter LP2S for a time *t* ([ms]) can be calculated by the following formula:

$$LP2S = \frac{262144}{t}$$

Value range: [0x01..0x7F] = [262144ms..2064ms]

Reset and default value for *t*: 6.5 s

$6C_h$ SSCAS1 Speech Comparator (Acoustic Side) 1

15	0
G	ET

Reset Value
0800_h

G

The parameter G for a gain *A* ([dB]) can be calculated by the following formula:

$$G = \frac{2 \times A}{5 \times \log 2}$$

Value range: [0x80..0x7F] = [-96.3dB..95.6dB]

Reset and default value for *A*: 6 dB

Note: The parameter G is interpreted in two's complement.

ET

The parameter ET for a time *t* ([ms]) can be calculated by the following formula:

$$ET = \frac{t}{4}$$

Value range: [0x00..0xFF] = [0ms..1020ms]

Reset and default value for *t*: 0 ms

$6D_h$ SSCAS2 Speech Comparator (Acoustic Side) 2

15		0
0	GDN	PDN

Reset Value

1006_h

GDN

The parameter GDN for a gain *G* ([dB]) can be calculated by the following formula:

$$GDN = \frac{4 \times G}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..47.8dB]

Reset and default value for *G*: 6 dB

PDN

The parameter PDN for a decay rate *R* ([ms/dB]) can be calculated by the following formula:

$$PDN = \frac{64}{5 \times \log 2 \times R}$$

Value range: [0x01..0xFF] = [47.8ms/dB..0.17ms/dB]

Reset and default value for *R*: 7 ms/dB

$6E_h$ SSCAS3 Speech Comparator (Acoustic Side) 3

15		0
0	GDS	PDS

Reset Value

1006_h

GDS

The parameter GDS for a gain *G* ([dB]) can be calculated by the following formula:

$$GDS = \frac{4 \times G}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..47.8dB]

Reset and default value for *G*: 6 dB

PDS

The parameter PDS for a decay rate *R* ([ms/dB]) can be calculated by the following formula:

$$PDS = \frac{64}{5 \times \log 2 \times R}$$

Value range: [0x01..0xFF] = [47.8ms/dB..0.17ms/dB]

Reset and default value for *R*: 7 ms/dB

$6F_h$ SSCLS1 Speech Comparator (Line Side) 1

15	0
G	ET

Reset Value
0000_h

G

The parameter G for a gain *A* ([dB]) can be calculated by the following formula:

$$G = \frac{2 \times A}{5 \times \log 2}$$

Value range: [0x80..0x7F] = [-96.3dB..95.6dB]

Reset and default value for *A*: 0 dB

Note: The parameter G is interpreted in two’s complement.

ET

The parameter ET for a time t ([ms]) can be calculated by the following formula:

$$ET = \frac{t}{4}$$

Value range: [0x01..0xFF] = [0ms..1020ms]

Reset and default value for *t*: 0ms

70_h SSCLS2 Speech Comparator (Line Side) 2

15		0
0	GDN	PDN

Reset Value
2002_h

GDN

The parameter GDN for a gain *G* ([dB]) can be calculated by the following formula:

$$\mathrm{GDN} = \frac{4 \times G}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..47.8dB]

Reset and default value for *G*: 12 dB

PDN

The parameter PDN for a decay rate *R* ([ms/dB]) can be calculated by the following formula:

$$\mathrm{PDN} = \frac{64}{5 \times \log 2 \times R}$$

Value range: [0x01..0xFF] = [47.8ms/dB..0.17ms/dB]

Reset and default value for *R*: 21.3 ms/dB

71_h SSCLS3 Speech Comparator (Line Side) 3

15		0
0	GDS	PDS

Reset Value
2002_h

GDS

The parameter GDS for a gain *G* ([dB]) can be calculated by the following formula:

$$\mathrm{GDS} = \frac{4 \times G}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..47.8dB]

Reset and default value for *G*: 12 dB

PDS

The parameter PDS for a decay rate *R* ([ms/dB]) can be calculated by the following formula:

$$\mathrm{PDS} = \frac{64}{5 \times \log 2 \times R}$$

Value range: [0x01..0xFF] = [47.8ms/dB..0.17ms/dB]

Reset and default value for *R*: 21.3 ms/dB

72_h SATT1 Attenuation Unit 1

15		0
0	ATT	SW

Reset Value

$2C6A_h$

ATT

The parameter ATT for an attenuation *A* ([dB]) can be calculated by the following formula:

$$\mathrm{ATT} = \frac{2 \times A}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..95.6dB]

Reset and default value for *A*: 36 dB

SW

The parameter SW for a switching rate *R* ([ms/dB]) can be calculated by the following formula:

$$\mathrm{SW} = \begin{cases} 128 + \dfrac{1}{5 \times \log 2 \times R} & ;0.0053 < R < 0.66 \\ \dfrac{16}{5 \times \log 2 \times R} & ;0.66 < R < 10.63 \end{cases}$$

Value range: [0x01..0x7F] = [0.84ms/dB..10.63ms/dB] and [0x81..0xFF] = [0.66ms/dB..0.005ms/dB]

Reset and default value for *R*: 0.1 ms/dB

73_h SATT2 Attenuation Unit 2

15	0
TW	DS

Reset Value

$0AFF_h$

TW

The parameter TW for a time *t* ([ms]) can be calculated by the following formula:

$$TW = \frac{t}{16}$$

Value range: [0x00..0xFF] = [0ms..4080ms]

DS

The parameter DS for a decay rate *R* ([ms/dB]) can be calculated by the following formula:

$$DS = \frac{5 \times \log 2 \times R - 1}{4}$$

Value range: [0x00..0xFF] = [0.66ms/dB..678.3ms/dB]

Note: The value 0xFF for the parameter DS specifies an infinite decay rate. Therefore the speakerphone will not return to the idle state in the absence of speech signals. It will remain in the current state until a speech signal is detected and a state change is necessary. This is also the reset and default value.

74_h SAGX1 Automatic Gain Control (Transmit) 1

15		0
AG_INIT	0	COM

Reset Value
$005F_h$

AG_INIT

The parameter AG_INIT for a gain *G* ([dB]) can be calculated by the following formula:

$$\text{AG\_INIT} = \frac{-2 \times G}{5 \times \log 2}$$

Value range: [0x80..0x7F] = [96.3dB..-95.6dB]

Reset and default value for *G*: 0 dB

Note: This parameter is interpreted in two's complement.

COM

The threshold COM for a level *L* ([dB]) can be calculated by the following formula:

$$\text{COM} = \frac{2 \times (96.3 + L)}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [-96.3dB..-0.72dB]

Reset and default value for *L*: -24 dB

75_h SAGX2 Automatic Gain Control (Transmit) 2

15		0
0	AG_ATT	SPEEDH

Reset Value

$7FFF_h$

AG_ATT

The parameter AG_ATT for a gain *G* ([dB]) can be calculated by the following formula:

$$AG\_ATT = \frac{-2 \times G}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..-95.6dB]

Reset and default value for *G*: 96 dB

SPEEDH

The parameter SPEEDH for the regulation speed *R* (ms/dB) can be calculated by the following formula:

$$SPEEDH = \frac{512}{D \times R}$$

Value range: [0x01..0xFF] = [512dB/ms x dB..16.1dB/ms x dB]

The variable D denotes the aberration ([dB]).

Reset and default value for *R*: 2 ms/dB

76_h SAGX3 Automatic Gain Control (Transmit) 3

15 AG_GAIN	SPEEDL 0

Reset Value

0014_h

AG_GAIN

The parameter AG_GAIN for a gain *G* ([dB]) can be calculated by the following formula:

$$\text{AG\_GAIN} = \frac{-2 \times G}{5 \times \log 2}$$

Value range: [0xC0..0x00] = [48.2dB..0dB]

Reset and default value for *G*: 0 dB

Note: The value has to be entered using 2-complement notation.

SPEEDL

The parameter SPEEDL for the regulation speed *R* ([ms/dB]) can be calculated by the following formula:

$$\text{SPEEDL} = \frac{4096}{D \times R}$$

Value range: [0x01..0xFF] = [4096dB/ms x dB..16.1dB/ms x dB]

The variable D denotes the aberration ([dB]).

Reset and default value for *R*: 160 ms/dB

77_h SAGX4 Automatic Gain Control (Transmit) 4

15			0
0	NOIS	0	LPA

Reset Value

4020_h

NOIS

The parameter NOIS for a threshold level *L* ([dB]) can be calculated by the following formula:

$$\mathrm{NOIS} = \frac{2 \times (96.3 + L)}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [-96.3dB..-0.72dB]

Reset and default value for *L*: -48 dB

LPA

The parameter LPA for a low pass time constant *T* ([ms]) can be calculated by the following formula:

$$\mathrm{LPA} = \frac{16}{T}$$

Value range: [0x01..0x7F] = [16ms..0.126ms]

Reset and default value for *T*: 2 ms

78_h SAGX5 Automatic Gain Control (Transmit) 5

15								0
AG_CUR	0	0	0	0	0	0	0	0

AG_CUR

The current gain *G* [dB] of the AGCX can be derived from the parameter Parameter AG_CUR by the following formula:

$$G = \frac{-5 \times \log 2 \times AG\_CUR}{2}$$

Value range: [0x80..0x7F] = [96.3dB..-95.6dB]

Note: AG_CUR is interpreted in two's complement.

79_h SAGR1 Automatic Gain Control (Receive) 1

15	0
AG_INIT	COM

Reset Value

$006F_h$

AG_INIT

The parameter AG_INIT for a gain *G* ([dB]) can be calculated by the following formula:

$$AG\_INIT = \frac{-2 \times G}{5 \times \log 2}$$

Value range: [0x80..0x7F] = [96.3dB..-95.6dB]

This parameter is interpreted in two's complement.

Reset and default value for *G*: 0 dB

COM

The parameter COM for a threshold *L* ([dB]) can be calculated by the following formula:

$$COM = \frac{2 \times (96.3 + L)}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [-96.3dB..-0.72dB]

This parameter is interpreted in two's complement.

Reset and default value for *L*: -12 dB

7A$_h$ SAGR2 Automatic Gain Control (Receive) 2

15		0
0	AG_ATT	SPEEDH

Reset Value

7FFF$_h$

AG_ATT

The parameter AG_ATT for a gain *G* ([dB]) can be calculated by the following formula:

$$AG\_ATT = \frac{-2 \times G}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [0dB..-95.6dB]

Reset and default value for *G*: 96 dB

SPEEDH

The parameter SPEEDH for the regulation speed *R* ([ms/dB]) can be calculated by the following formula:

$$SPEEDH = \frac{512}{D \times R}$$

Value range: [0x01..0xFF] = [512dB/ms x dB..16.1dB/ms x dB]

The variable D denotes the aberration ([dB]).

Reset and default value for *R*: 2 ms/dB

$7B_h$ SAGR3 Automatic Gain Control (Receive) 3

15	0
AG_GAIN	SPEEDL

Reset Value

$001A_h$

AG_GAIN

The parameter AG_GAIN for a gain *G* ([dB]) can be calculated by the following formula:

$$\mathrm{AG\_GAIN} = \frac{-2 \times G}{5 \times \log 2}$$

Value range: [0xC0..0x00] = [48.2dB..0dB]

Reset and default value for *G*: 0 dB

Note: The value has to be entered using 2-complement notation.

SPEEDL

The parameter SPEEDL for the regulation speed *R* ([ms/dB]) can be calculated by the following formula:

$$\mathrm{SPEEDL} = \frac{4096}{D \times R}$$

Value range: [0x01..0xFF] = [4096dB/ms x dB..16.1dB/ms x dB]

The variable D denotes the aberration ([dB]).

Reset and default value for *R*: 200 ms/dB

$7C_h$ SAGR4 Automatic Gain Control (Receive) 4

15			0
0	NOIS	0	LPA

Reset Value

4020_h

NOIS

The parameter NOIS for a threshold level *L* ([dB]) can be calculated by the following formula:

$$\mathrm{NOIS} = \frac{2 \times (96.3 + L)}{5 \times \log 2}$$

Value range: [0x00..0x7F] = [-96.3dB..-0.72dB]

Reset and default value for *L*: -48 dB

LPA

The parameter LPA for a low pass time constant *T* ([ms]) can be calculated by the following formula:

$$\mathrm{LPA} = \frac{16}{T}$$

Value range: [0x01..0x7F] = [16ms..0.126ms]

Reset and default value for *T*: 2 ms

$7D_h$ SAGR5 Automatic Gain Control (Receive) 5

15								0
AG_CUR	0	0	0	0	0	0	0	0

AG_CUR

The current gain *G* [dB] of the AGCR can be derived from the parameter Parameter AG_CUR by the following formula:

$$G = \frac{-5 \times \log 2 \times AG\_CUR}{2}$$

Value range: [0x80..0x7F] = [96.3dB..-95.6dB]

AG_CUR is interpreted in two’s complement.

$7E_h$ SLGA Line Gain

15		0
0	LGAR	LGAX

Reset Value

4040_h

LGAR

The parameter LGAR for a gain *G* ([dB]) is given by the following formula:

$$\text{LGAR} = 128 \times 10^{(G-12)/20}$$

Value range: [0x01..0x7F] = [-20.14dB..11.93dB]

Reset and default value for *G*: 6 dB

LGAX

The parameter LGAX for a gain *G* ([dB]) is given by the following formula:

$$\text{LGAX} = 256(128) \times 10^{(G-24)/20}$$

Value range: [0x01..0xFF] = [-20.16dB..23.97dB]

Reset and default value for *G*: 12 dB

$7F_h$ SAELEN Acoustic Echo Cancellation Length

15						0
0	0	0	0	0	0	FBLEN

Reset Value
0200_h

FBLEN

LEN denotes the number of FIR-taps used in fullband mode. FBLEN=127...MAX

MAX = 639 for fullband mode one
MAX = 768 for fullband mode two

80_h SAEAW Acoustic Echo Cancellation Adaption Window

15							0
0	0	0	0	0	0	0	FBADA

Reset Value
0100_h

FBADA

FBADA denotes the number of FIR-taps changed adaptively (in one step) in full band mode two

81_h SAEL Acoustic Echo Cancellation Reported Attenuation Limit

15	0
0	AECLIM

Reset Value
$7FFFF_h$

AECLIM

The maximum value for an attenuation *A* ([dB]) of the echo cancellation unit reported to the echo suppression unit given by the following formula:

$$\mathrm{AECLIM} = \frac{512 \times A}{5 \times \log 2}$$

Value range: [0x0800..0x7FFF] = [6.02dB..96.3dB]

Reset and default value is the maximum value for *A* (96 dB).

82_h SAEDTR Acoustic Echo Cancellation Double Talk Reduction

15	0
0	AECATT

Reset Value

1400_h

AECATT

The parameter AECATT for an attenuation reduction *A* ([dB]) during double talk is given by the following formula:

$$\text{AECATT} = \frac{512 \times A}{5 \times \log 2}$$

Value range: [0x0000..0x7FFF] = [0dB..96.3dB]

Reset and default value for *A*: 15 dB

83_h SAEDTL Acoustic Echo Cancellation Double Talk Limit

15	0
0	AECDTM

Reset Value

1800_h

AECDTM

The parameter AECDTM for a minimum energy *A* ([dB]) to detect double talk is given by the following formula:

$$\mathrm{AECDTM} = \frac{512 \times A}{5 \times \log 2}$$

Value range: [0x0000..0x7FFF] = [0dB..96.3dB]

Reset and default value for *A*: 16 dB

84_h SAEDTI Acoustic Echo Cancellation Double Talk Increment

15	0
0	AECDTI

Reset Value
0100_h

AECDTI

AECDTI determines the rate the attenuation reduction RED ([dB/s]) of the AEC is incremented with when double talk is detected:

$$AECDTI = 8192xRED$$

Value range: [0x0000..0x7FFF] = [0db/s..3999.9dB/s]

Reset and default value for for *RED*: 31.6 dB/s

85_h SAEDTD Acoustic Echo Cancellation Double Talk Decrement

15	0
0	AECDTD

Reset Value
0062_h

AECDTD

AECDTD determines the rate the attenuation reduction *RED* of the AEC is decremented with when double talk is detected:

$$AECDTD = 8192xRED$$

Value range: [0x0000..0x7FFF] = [0db/s..3999.9dB/s]

Reset and default value for *RED*: 12 dB/s

86_h SAEWFL Wiener Filter Limit Attenuation

15	0
0	WFATT

Reset Value
5000_h

AFATT

The parameter WFATT for a maximal attenuation *A* ([dB]) of the Wiener filter is given by the following formula:

$$\mathrm{LIMIT} = \frac{512 \times A}{5 \times \log 2}$$

Value range: [0x0000..0x7FFF] = [0dB..96.3 dB]

Reset and default value for *A*: 60 dB

87_h SNRATT Noise Reduction Attenuation

15	0
0	NRATT

Reset Value

2800_h

NRATT Noise Reduction Attenuation

The maximum attenuation *A* of frequencies with a high noise to signal ratio performed by the noise reduction unit can be calculated by the following formula:

$$A = 32768 \times 10^{-\mathrm{NRATT}\frac{\mathrm{dB}}{20}}$$

Value range: [0x0001..0x7FFF] = [90.3dB..0.0003dB]

Reset and default value for *A*: 15 dB

88_h SNRLNL Noise Reduction Lower Noise Limit

15	0
0	NRLOW

Reset Value

4000_h

NRLOW Noise Reduction Lower Limit

The level *L* ([dB]) of the noise to deactive the coupling between the Wiener filter and the noise reduction unit can be calculated by the following formula:

$$\mathrm{NRLOW} = \frac{512 \times (96.3 + L)}{5 \times \log 2}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0.03 dB]

Reset and default value for *L*: -48 dB

89_h SNRUNL Noise Reduction Upper Noise Limit

15	0
0	NRUP

Reset Value

5000_h

NRUP **Noise Reduction Upper Limit**

The level L ([dB]) of the noise to active the coupling between the Wiener filter and the noise reduction unit can be calculated by the following formula:

$$\mathrm{NRUP} = \frac{512 \times (96.3 + \mathrm{L})}{5 \times \log 2}$$

Value range: [0x0000..0x7FFF] = [-96.3dB..0.03 dB]

Reset and default value for *L*: -36 dB

$A6_h$ NRCTL Noise Reduction Control

15										0
EN	MD	0	0	0	0	0	0	0	0	I1

Reset Value

0	0	0	0	0	0	0	0	0	0	0

EN **Noise Reduction Enable**

0: Disabled

1: Enabled

MD **Speakerphone Mode Dependence**

14	13	Speakerphone Mode	Description
0	0	Fullband mode one	Number of taps must be restricted to 447
		Fullband mode two	Not allowed
		Subband mode	Not allowed
0	1	Fullband mode one	Not allowed
		Fullband mode two	Number of taps must be restricted to 414
		Subband mode	only subband mode ’Reduced’ is allowed
1	0	Fullband mode one	LEC or DTMF receiver must not be enabled
		Fullband mode two	LEC or DTMF receiver must not be enabled Number of taps must be restricted to 645
		Subband mode	LEC or DTMF receiver must not be enabled Subband modes "Normal" or "Reduced" are allowed
1	1		reserved

I1 **Input signal selection**

$A7_h$ NRATT Noise Reduction Attenuation

15	0
0	NRATT

NRATT Noise Reduction Attenuation

The maximum attenuation *A* ([dB]) of noisy frequencies performed by the noise reduction unit can be calculated by the following formula:

$$\mathrm{NRATT} = 32768 \times 10^{-\frac{A}{20}}$$

Value range: [0x0001..0x7FFF] = [90.3dB..0.0003 dB]

21 Terminal Specific Functions

The Terminal Specific Functions (TSF) comprises the keypad scanner and the LED multiplex unit.

There is no global clock control register, but the different modules can be en/-disabled individually by programming the only global register TSF_CONF.

21.1 Keypad Scanner

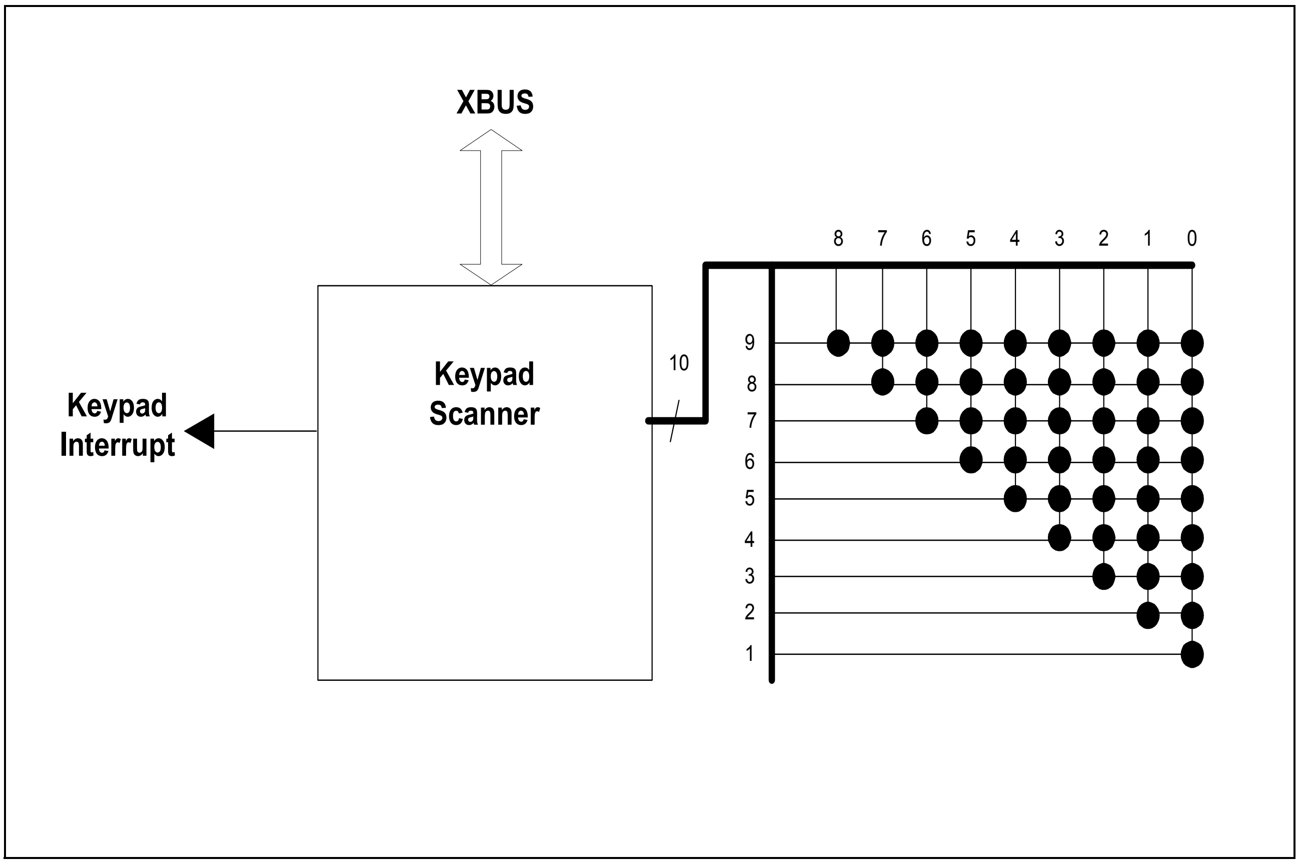


Figure 21-1 Key Scanner

The keypad scanner supports keypads with up to 45 keys organized in 9 rows and 9 columns in a 'half matrix' structure.

The matrix is shown in figure **Figure 21-1**. For the port pin of the keypad scanner the internal pull-up transistors can be enabled, so that no external resistors are needed.

A logical low value in **Figure 21-2** indicates a line which is driven to low by the INCA-D. An <XXX> indicates the period during which a line becomes an input and is evaluated by the keypad scanner. If the input level is low, the corresponding key is pressed.

Note: To recognize a certain number of concurrently pressed keys non-ambiguously, the selection of those keys has to be done in a certain manner. Because of the *implemented half-matrix structure, it could happen that while pressing some keys, the appropriate register contents does not reflect physically pressed keys only, but also some "virtual" pressed keys. If, for example, keys (0,1) and (1,2) are pressed concurrently, the key (0,2) will also be detected as "pressed", although it is not.*

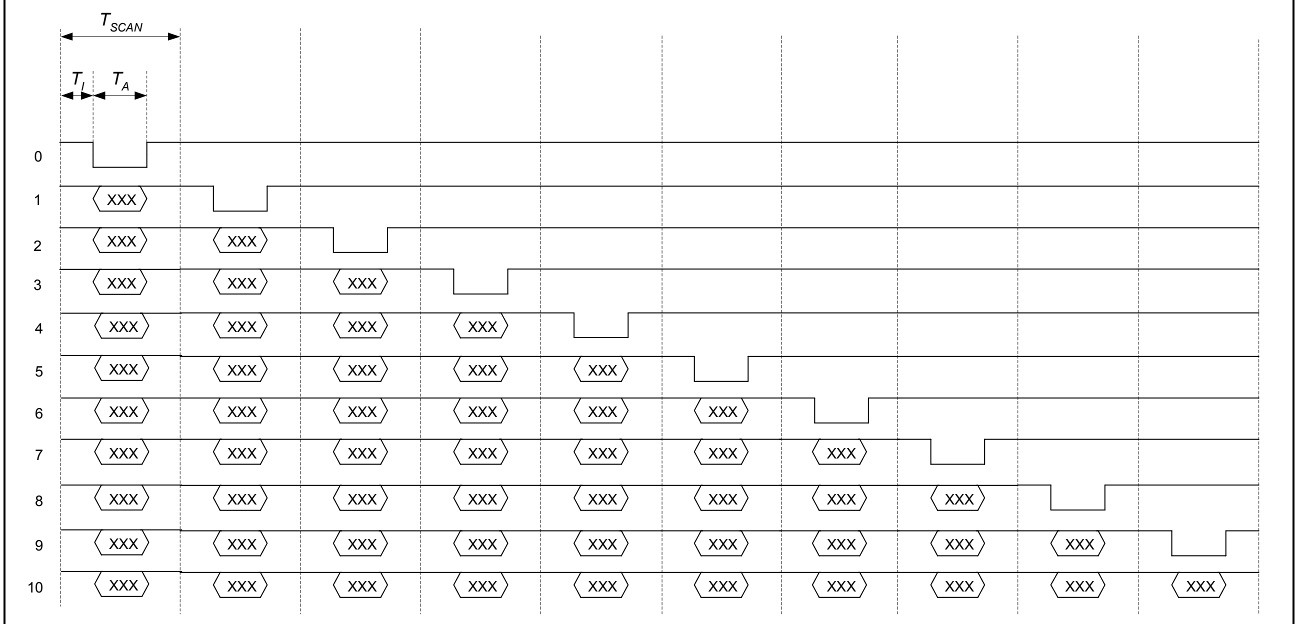


Figure 21-2 Timing Diagram for Keypad Scanner

The scan period for each line is approximately 1ms (T_{SCAN} in **Figure 21-2**), i.e. if all scan lines are enabled, a complete scan period over all lines lasts about 9ms. A scan period for a single line is divided into an idle phase T_I and an active phase T_A. This allows logic levels to settle before the next line is scanned. The idle phase is approximately as long as the active phase. Independent of the number of enabled scan lines, a new scan period always starts after 9 x T_{SCAN}. Thus, if a change occurs to the values, a delay of 9 ms is substituted before a new interrupt is generated.

The number of scan lines is programmable via the four bit value SL0-3 in the keysan control register KEY_CONF. This value defines the last scan line before the scanner returns to line 0. All unused scan lines are internally set to one so that the register always reports '1', regardless whether the pins are used as GPIO.

Interrupt Generation

If line '0' is driven 'LOW', the result of scanning lines 1-9 (if 9 lines enabled, otherwise less) is stored in register SREG0. When line '1' is driven 'LOW'', lines '2-9' are read. The corresponding results are stored in register SREG1 and so forth.

The behaviour of the keypadscanner is such that the SREGx registers contain the result of the current scan cycle and an interrupt is always generated whenever a change of the register value is detected within two successive scan cycles (to prevent bouncing).

If two pressed keys are not released in the same scan cycle (e.g. cycle #5 and #6 in the example below), an interrupt is also generated, i.e. releasing a key is also indicated by an interrupt (cycle #8).

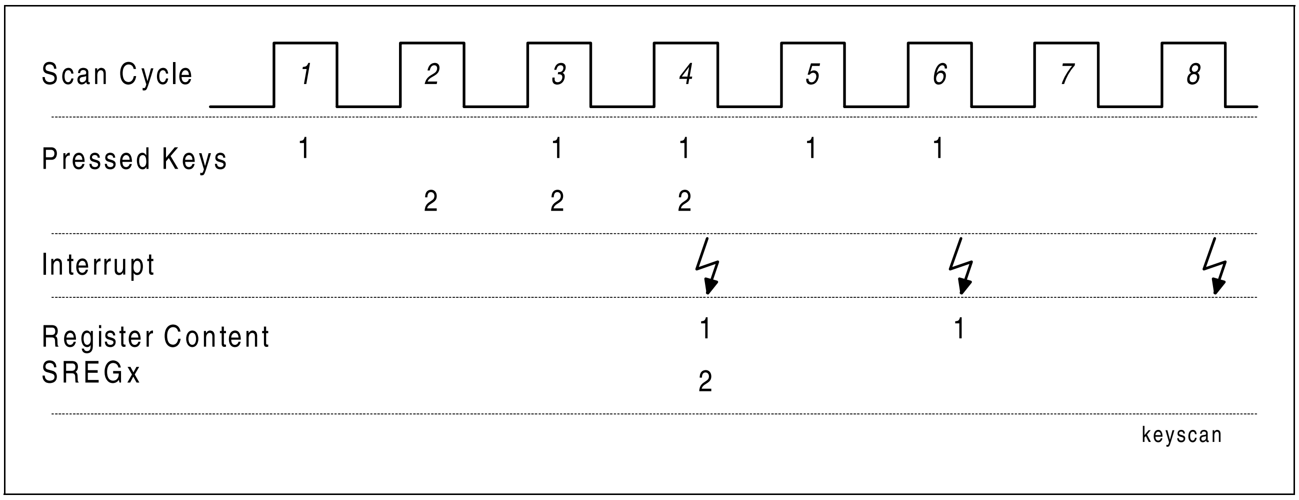


Figure 21-3 Behaviour of the Keypad Scanner

The scan results are stored in the SREGx registers with a mapping as shown in **Figure 21-4**.

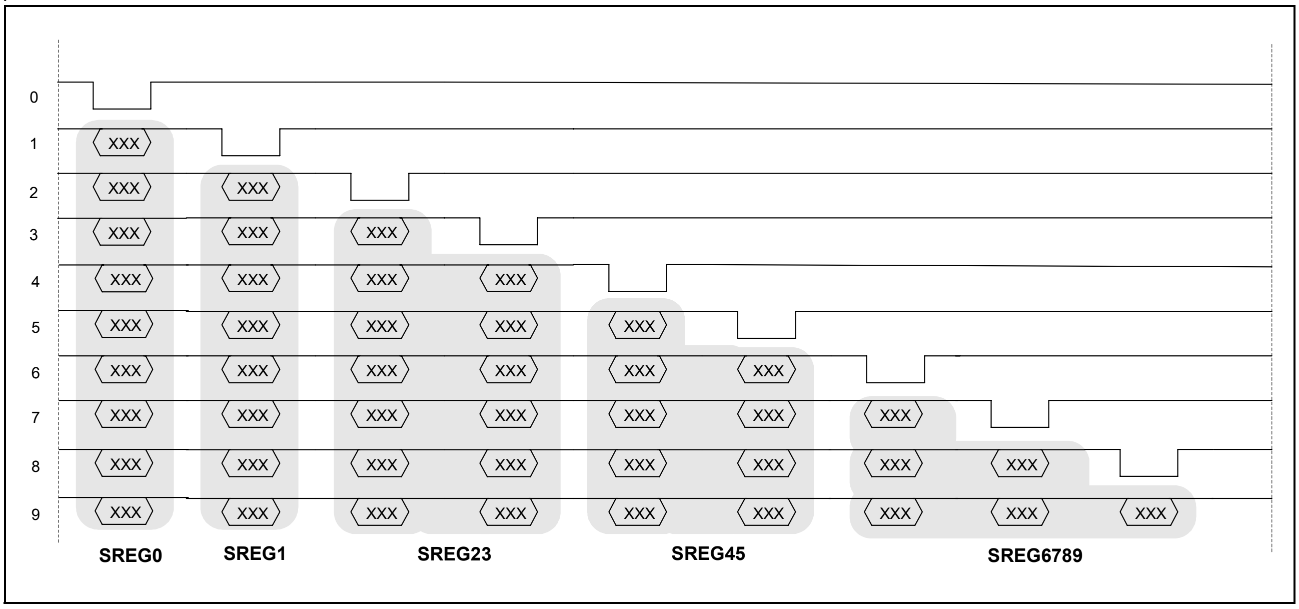


Figure 21-4 Buffer Structure

21.2 LED Matrix Control for up to 24 LED’s

A multiplex structure as described below has been implemented. It operates as follows:

Consider the case that only the first colum (thus actually no multiplexing) is triggered. The more LEDs are switched on, the bigger is the total current through all LEDs. When looking at the current through one LED when another one is switched on, there is a slight decrease in the current caused by the internal transistor resistor.

When using an appropriate transistor type, the current through each of the LEDs is approximately independent of the number of activated LEDs.

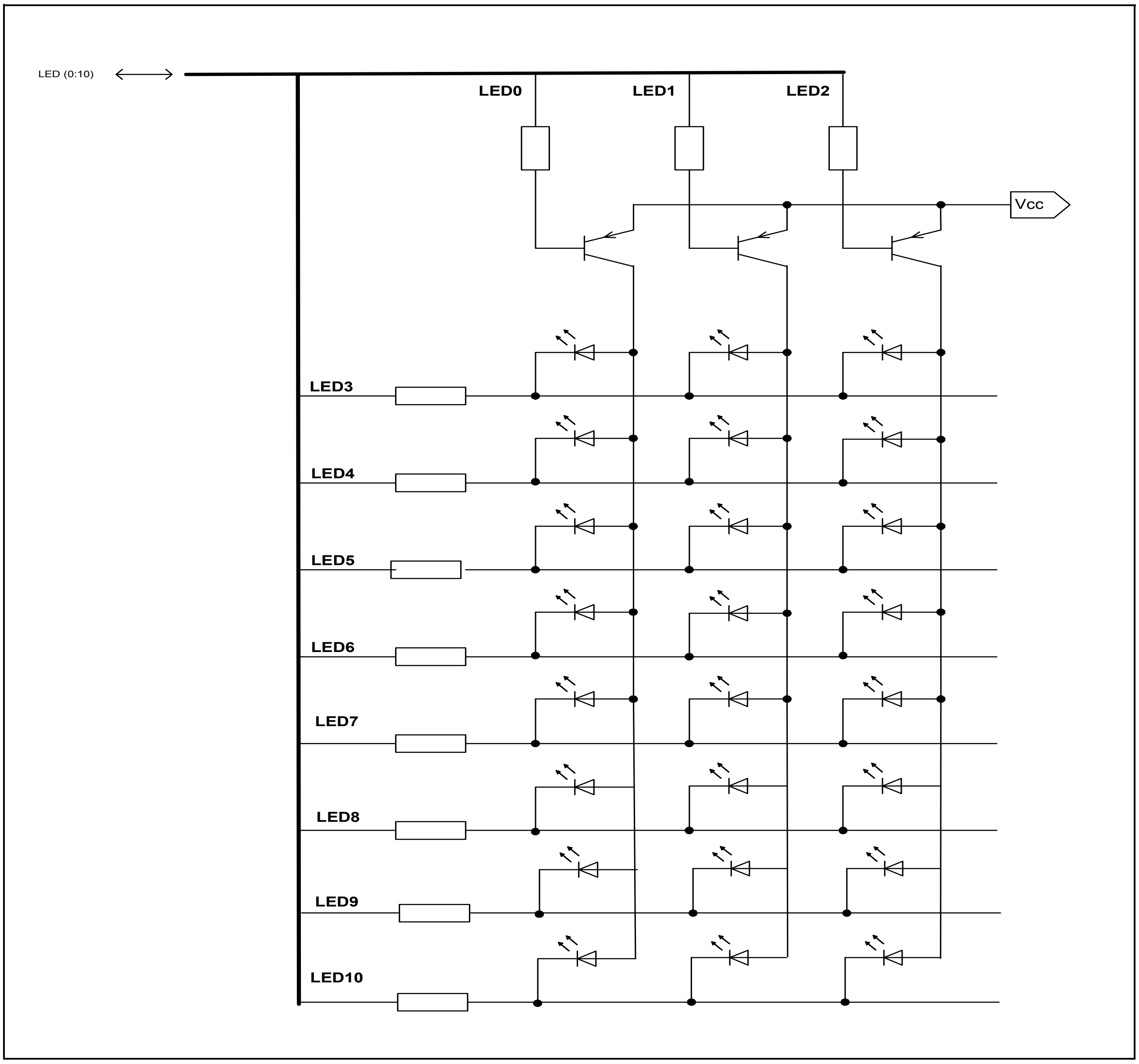


Figure 21-5 shows the basic timing of the LED matrix. The timing diagram assumes that all 8 LEDs in column 0 (line LED 0) are activated. The repetition rate is ca. 157 Hz. The line drivers are turned on and off with a delay to reduce the power consumption while switching.

Actually, the selection of the active LEDs has to be done via the LED registers LED_REG1 and LED_REG2. The timing for the subsequent two colums is equivalent to the timing of column 0.

The number of LED rows (LED3...10) depends on the setting of the alternate function register of Port2. Each line driver is able to sink 11 mA (sink current).

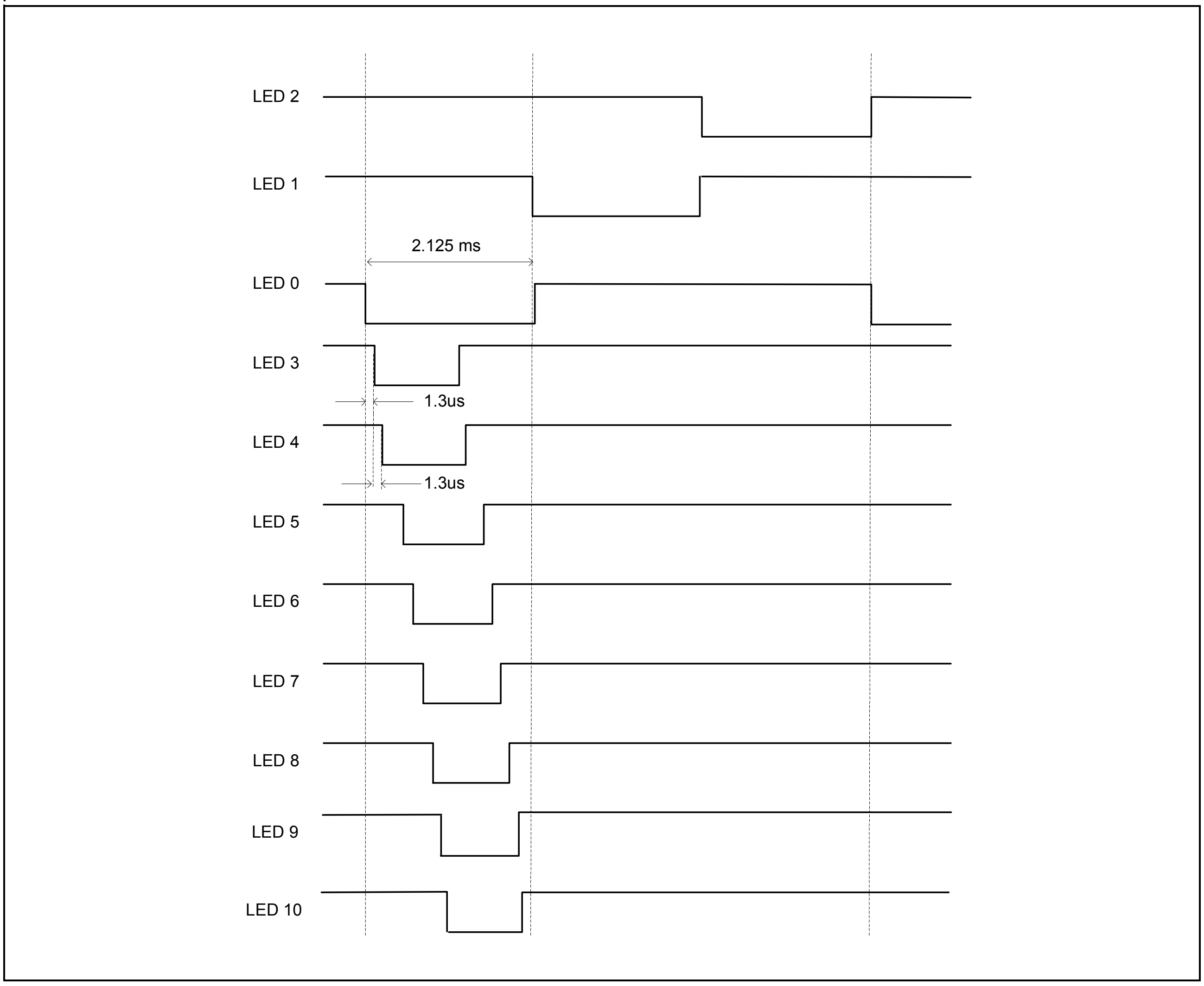


Figure 21-5 LED-Multiplex Timing

21.3 TSF Global Register Description

TSF_CONF (DF12$_H$) XBUS-SFR **Reset Value:00 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	LED EN	KEY EN	0	0	0	0	0	0	0	0	0	0	0	0
r	r	rw	rw	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Function
KEYEN	**Enable keypad scanner** **0:** Keypad scanner disabled **1:** Keypad scanner enabled
LEDEN	**Enable LED multiplex unit** **0:** LED multiplex unit disabled **1:** LED multiplex unit enabled

21.3.1 Keypad Scanner Register Description

SREG0 (DF00$_H$) XBUS-SFR **Reset Value: FCFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	RES9	RES8	RES7	RES6	RES5	RES4	RES3	RES2	RES1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Function
ResX	**Storage Buffer for results of scanning line 0** **X: Result of line X**

SREG1 (DF02$_H$) XBUS-SFR **Reset Value: FEFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	1	RES9	RES8	RES7	RES6	RES5	RES4	RES3	RES2
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Function
ResX	**Storage Buffer for results of scanning line 1** **X: Result of line X**

SREG23 (DF04$_H$) XBUS-SFR **Reset Value:FFFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	RES 29	RES 28	RES 27	RES 26	RES 25	RES 24	RES 23	1	1	RES 39	RES 38	RES 37	RES 36	RES 35	RES 34
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Function
Res2X	Storage Buffer for results of scanning line 2 X: Result of line X
Res3X	Storage Buffer for results of scanning line 3 X: Result of line X

SREG45(DF06$_H$) XBUS-SFR **Reset Value: FFFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	1	RES 49	RES 48	RES 47	RES 46	RES 45	1	1	1	1	RES 59	RES 58	RES 57	RES 56
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Function
Res4X	Storage Buffer for results of scanning line 4 X: Result of line X
Res5X	Storage Buffer for results of scanning line 5 X: Result of line X

SREG6789(DF08$_H$) XBUS-SFR **Reset Value: FFFF$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	RES 69	RES 68	RES 67	1	1	RES 79	RES 78	1	1	1	RES 89	1	1	1	1
r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r

Bit	Function
ResnX	Storage Buffer for results of scanning line n X: Result of line X

KEY_CONF (DF0A$_H$) XBUS-SFR **Reset Value:0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	SL.3	SL.2	SL.1	SL.0
r	r	r	r	r	r	r	r	r	r	r	r	rw	rw	rw	rw

Bit	Function
SL.0-3	Select *number* of total scan lines (not number of *last* line) from ’0000’ to ’1010’

21.3.2 LED Multiplex Unit register description

LED_REG1 (DF0E$_H$) XBUS-SFR **Reset Value:00 00$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
L08	L27	L17	L07	L26	L16	L06	L25	L15	L05	L24	L14	L04	L23	L13	L03
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

LED_REG2 (DF10$_H$) XBUS-SFR **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	L210	L110	L010	L29	L19	L09	L28	L18
rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
Lxy	Each bit represents the state of a LED, 0: LED is off 1: LED is on Select LED from column x and row y

22 USB Module

22.1 Introduction

The USB module handles all transactions between the universal serial bus USB and the internal (parallel) X-Bus of the CPU. The USB module includes several units which are required to support data handling with the USB bus:

- An onchip USB bus transceiver
- An USB memory with 2 pages of 256 bytes each
- The memory management unit (MMU) for USB module and µC memory access control
- The USB device core (UDC) for USB protocol handling
- A XBUS interface with the USB specific special function registers and an PD-Bus interface for interrupt control logic

Figure 22-1 shows the block diagram of the functional units of the USB module with their interfaces.

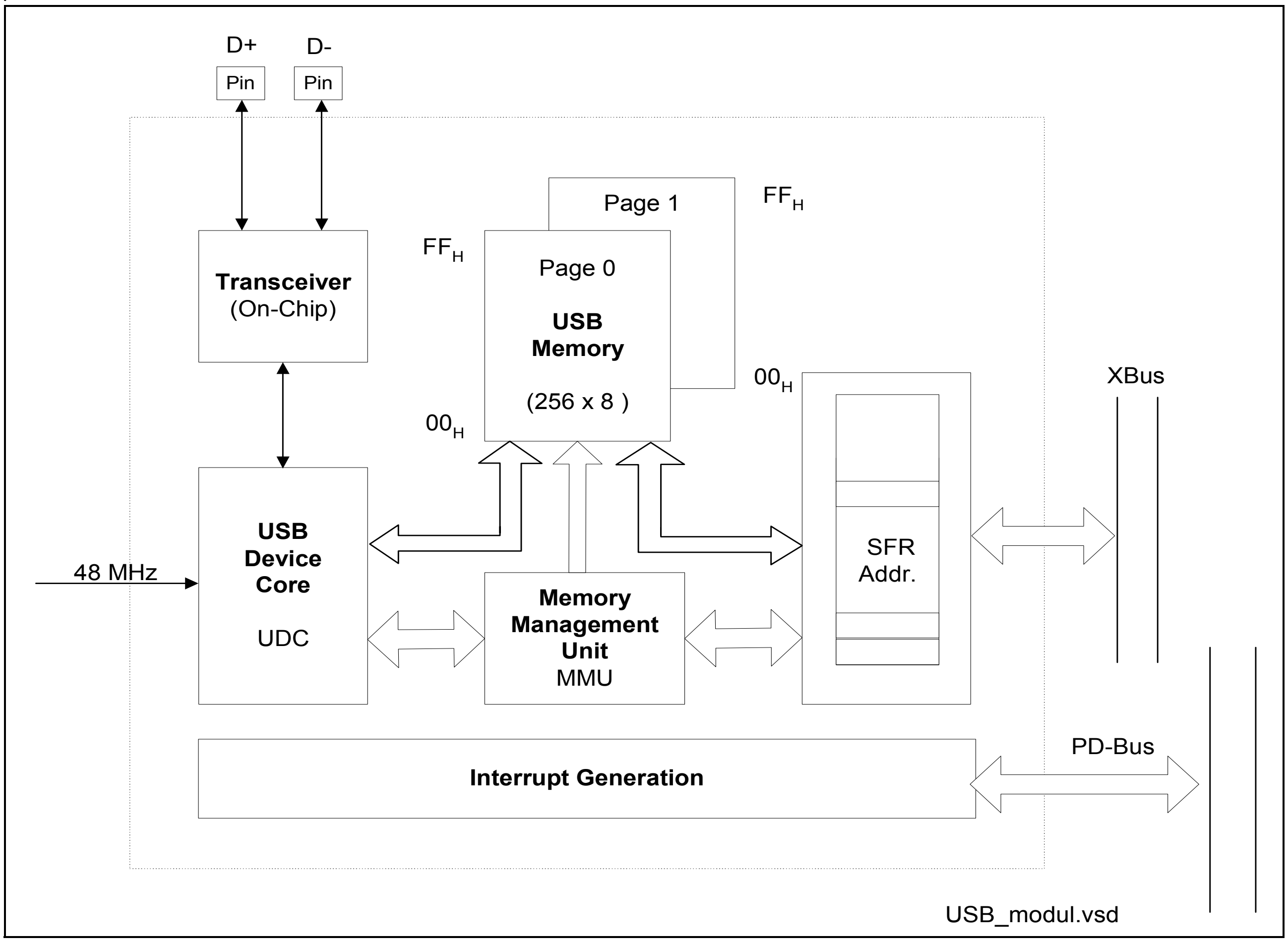


Figure 22-1 USB Module Block Diagram

The UDC will store all the information for each EndPoint. in a separate buffer (40 bit register called *EndPtBufn* internally). The number of these buffers depends on the number of ***Physical*** **Endpoints.**

Physical vs. Logical Endpoints

More than two ***Physical*** Endpoints can have the same ***Logical*** Endpoint number, which will be mapped to *Physical* Endpoints during initialization by setting of *Ep_Num*.

Each Physical Endpoint might there be associated with a particular Configuration or a Alternate Setting, but only to **one** direction (IN or OUT).

Generally, a Logical Endpoint can be bidirectional. However, because only the (logical) Endpoint number and **not** the direction is interpreted by the UDC, the endpoints have to be used for one direction only.

Note: An exeption is Endpoint 0, which is always bidirectional.

The USB commands issued by a host controller refer to Logical Endpoints.

22.2 USB Configurations

The implemented USB module supports communication pipes to 16 physical endpoints. The supported 15 interfaces can be used in the described four configurations(0-3).

Note: The USB specification v1.1 supports 1 configuration in addition to CONFIG0 only. (CONFIG0 and CONFIG1).

As the UDC interprets the Endpoint number only and not the direction of a specific Endpoint, the usage of the same logical Endpoint for IN and OUT transfers should be avoided.

The following abbreviations are used:

- IFC: Interface Configuration
- EP: Endpoint
- AS: Alternate Setting

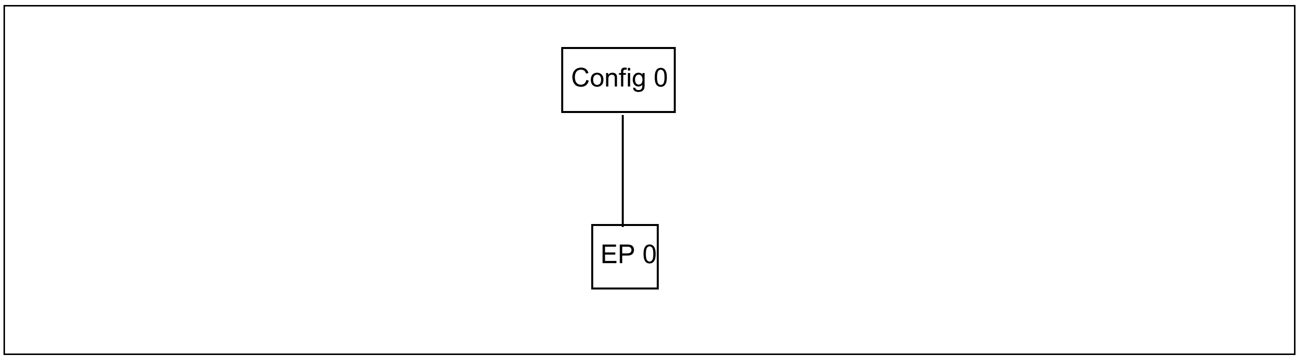


Figure 22-2 USB Configuration 0 with Control Endpoint 0

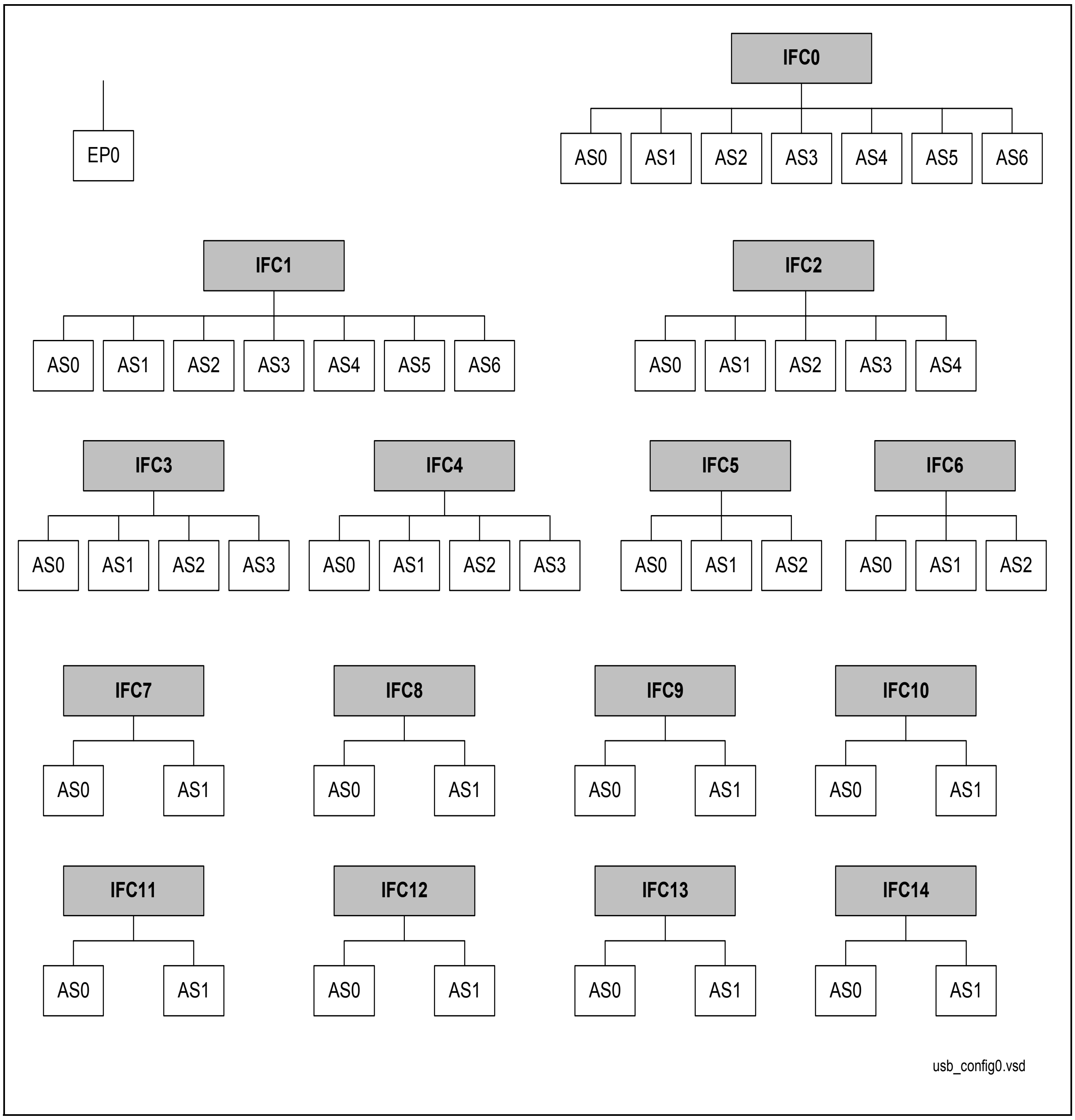


Figure 22-3 Configuration 1

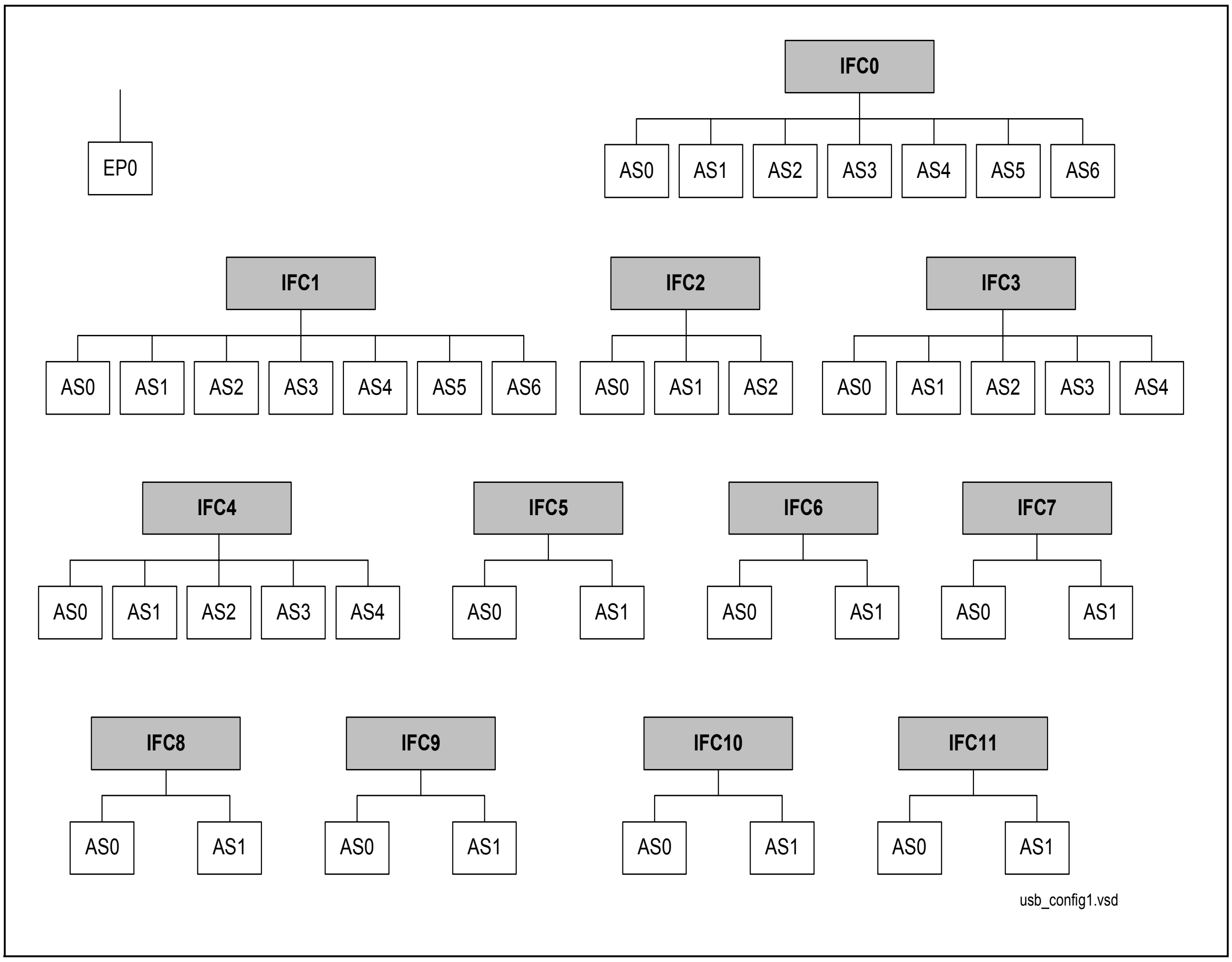


Figure 22-4 Configuration 2

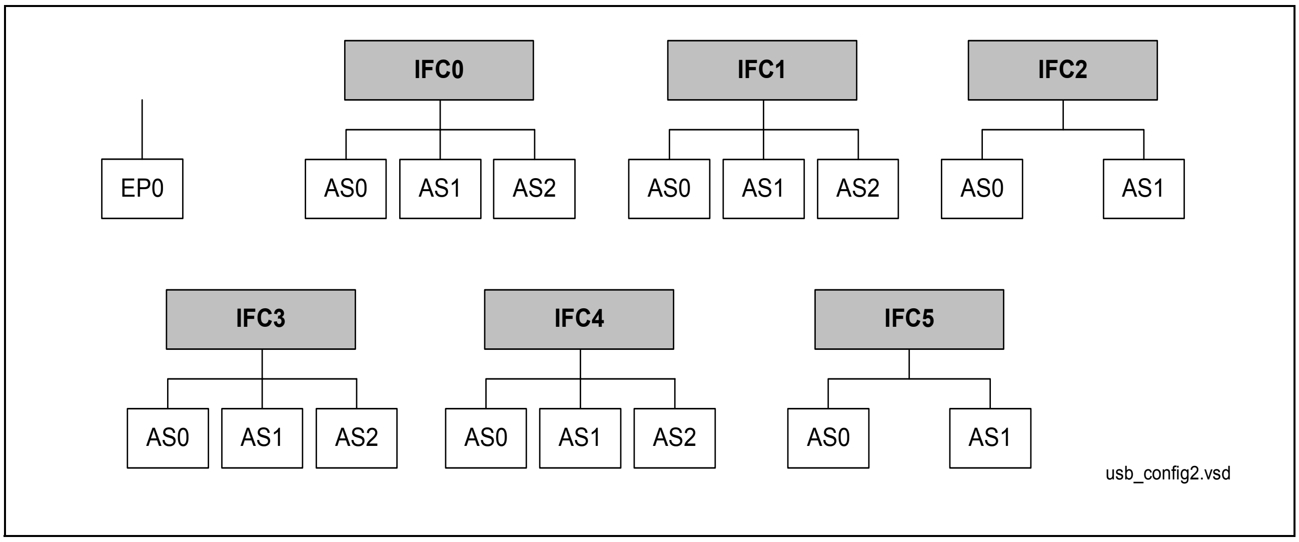


Figure 22-5 Configuration 3

22.3 Transfer Modes

USB data transfers take place between host and a particular endpoint on an USB device. The INCA-D supports multiple data transfer endpoints. The USB host treats communications with any endpoint of the INCA-D independently from any other endpoint. Such associations between the host software and a USB device endpoint are called pipes. As an example, the INCA-D could have an endpoint supporting a pipe for transporting B1-channel data from the host to the memory and another endpoint supporting a pipe for transporting B1-channel data from the INCA-D to the host. The USB architecture of the INCA-D comprehends all four basic types of data transfers, i.e. Control, Isochronous, Interrupt and Bulk.

Table 22-1 USB Transfer Modes

Mode	Function
Control	Control data are used to configure devices, data transmission is lossless. Control pipes are bidirectional, data transfer is possible in both directions via one pipe. Endpoint 0 is always configured as control endpoint with a maximum buffer length of 8 bytes. The control endpoint can be configured to handle data packets of 64 bytes maximum length.
Isochronous	Isochronous data are continuous and real-time in creation and consumption, such as voice data. In this case, real-time is defined from frame to frame. Isochronous data transfer has the highest priority, but is not always lossless. Isochronous pipes are always unidirectional, so one endpoint can be associated to an IN pipe or an OUT pipe. The INCA-D supports up to 64 bytes.
Interrupt	Interrupt data are a small amount of data, which are transferred every n frames, with n being programmable by the host. Data delivery is lossless. Interrupt pipes are always unidirectional pipes with selectable direction (IN or OUT pipe), the maximum data packet length is limited to 64 bytes.
Bulk	Bulk data can be a larger amount of data, which can be split by the host in several data packets within one frame. Data delivery is lossless. Bulk pipes are always unidirectional, so one endpoint can be associated to an IN pipe or an OUT pipe. The maximum data packet length is limited to 64 bytes.

22.4 Memory Buffer

22.4.1 Overview

Every endpoint of the USB module can operate in 2 modes, dual buffer mode and single buffer mode. Each mode provides random or sequential access to the USB memory. **Figure 22-6** shows the possible buffer modes.

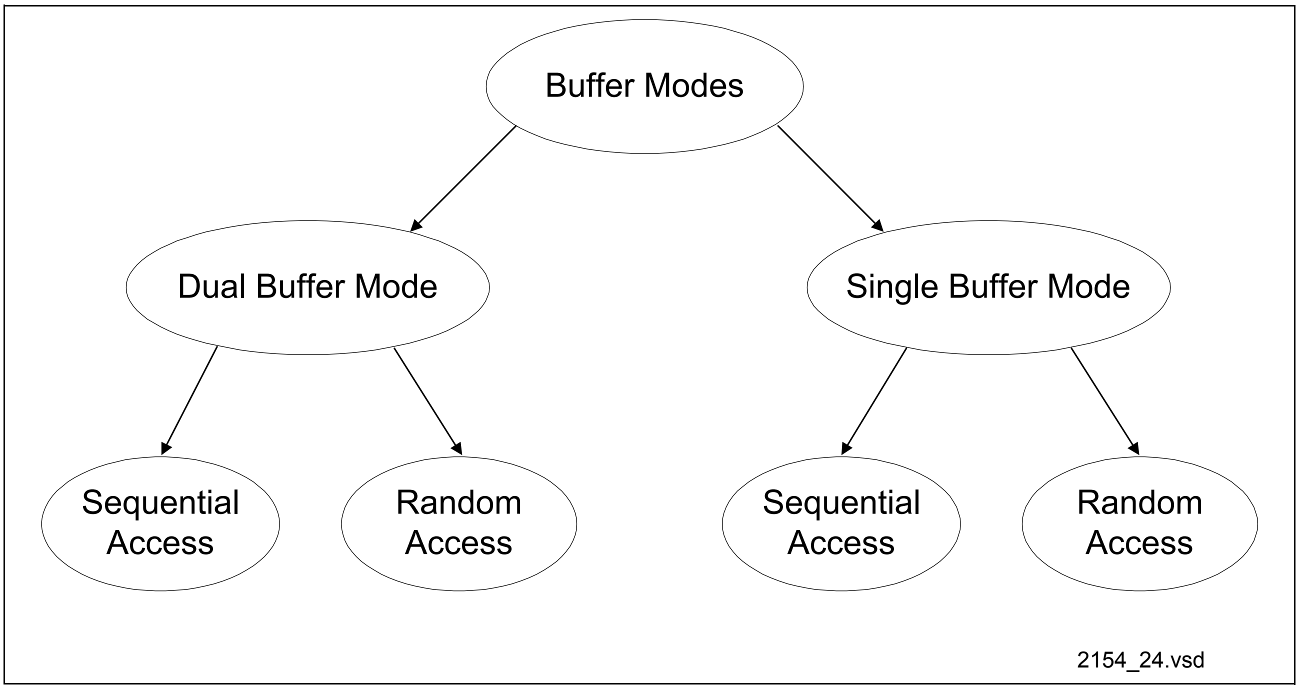


Figure 22-6 Memory Buffer Modes

Single Buffer Mode

In single buffer mode, the USB module and the CPU use one common memory page. The active buffer page is either page 0 or page 1.

Dual Buffer Mode

In dual buffer mode the USB module and the CPU write into different USB memory pages allowing back-to-back data transfers. Switching between the pages is done automatically, enabling a high data transfer rate between the CPU and the USB module.

Random Access

Random access is available in both modes. Random access allows to change only a few bytes in a data block of the USB memory buffer. When the CPU has modified the bytes in the data block, setting of bit DONE by software marks the buffer ready for transmission or reception of data over the USB pipe. For modification of a specific byte in the buffer, the CPU must write the address to SFR ADROFFn and read/write the data byte from/to register USBVALn.

Sequential Access

In sequential access mode the CPU accesses the data register USBVALn continuously without setting the address of the next USB memory buffer location. This is done automatically if bit INCE (increment enable) in the related SFR EPBCRn is set. After a specific number of CPU accesses (as done in SFR EPLENn), the buffer has been read/ written by the CPU and is empty/full. Setting of bit DONE in software, manually or automatically, marks the USB buffer ready.

22.4.2 Single Buffer Mode

In single buffer mode the USB and the CPU share one common USB memory page. The active buffer page can be either page 0 or page 1. Back-to-back transfers are not possible in this mode. Easy data storage and controlling can be achieved in this mode. E.g. a once created data set for an interrupt endpoint can be stored permanently in USB memory. As a result, an additional memory space for data storage is no longer needed.

22.4.2.1 USB Write Access

Figure 22-7 shows the basic flowchart of a USB write access to one USB memory buffer in single buffer mode.

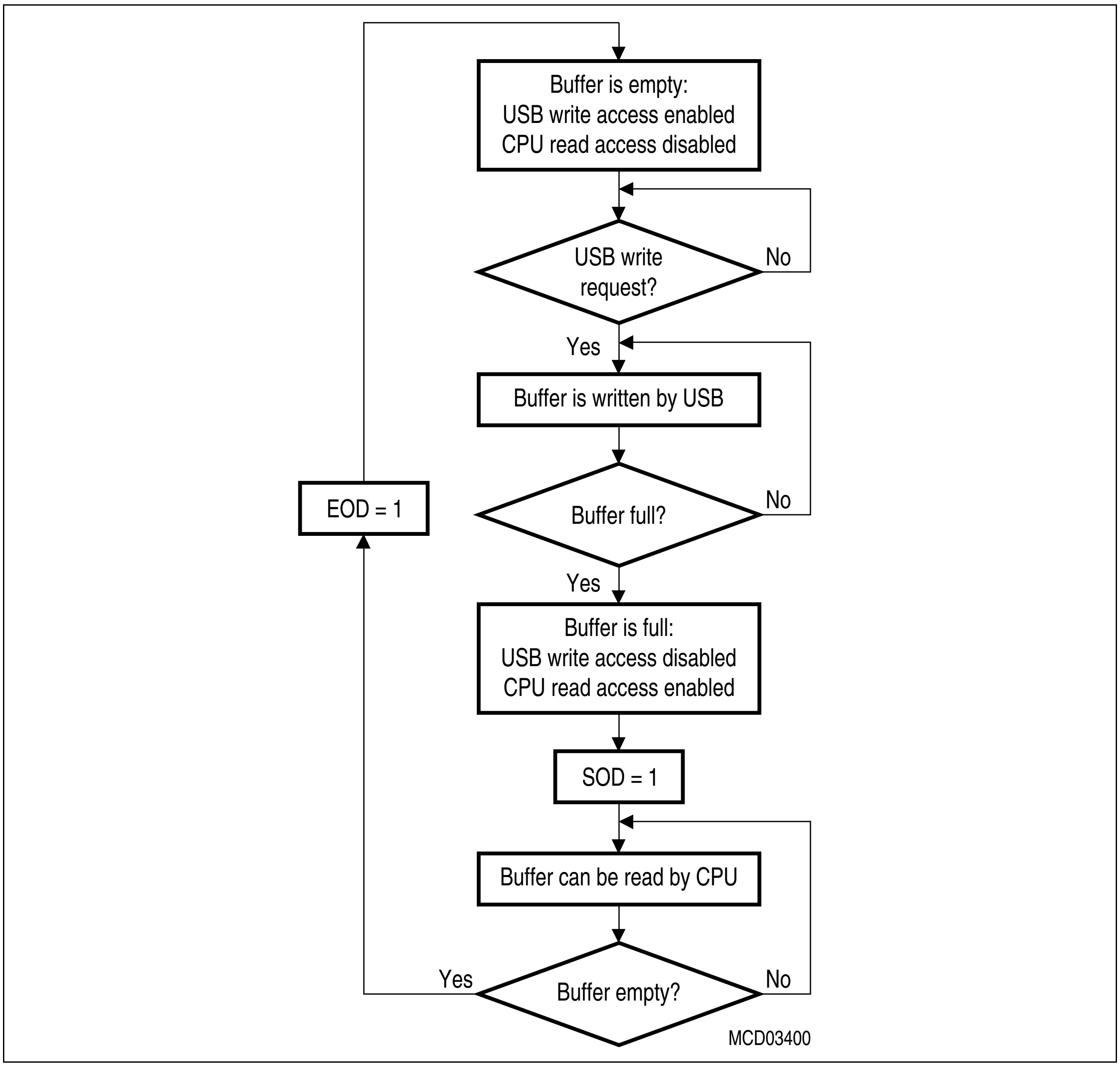


Figure 22-7 USB Write Access in Single Buffer Mode - Buffer Handling

Figure 22-8 shows more details of an USB write access to USB memory in single buffer mode. After SOF(n) (start of frame) occured at ①, the USB starts writing at ② a fixed number of bytes into the USB memory. A byte counter is incremented after every USB memory write operation. When the USB memory write operation (Len(n)) is finished correctly, bit SOD (start of data) is set at ③, indicating a full USB memory buffer. Furthermore, the byte counter value is stored in the corresponding length register, indicating the number of bytes which have been transferred and can be now read by the CPU. Subsequently, the CPU can read data bytes from USB memory, generating an EOD (end of data) at ④ after the last byte has been read. Bit EOD set indicates an empty USB buffer, which now can be written again by the USB.

Figure 22-8 also shows a second USB write access operation with a different number of bytes (Len(n+1)), where the CPU read operation from the USB memory is interrupted twice.

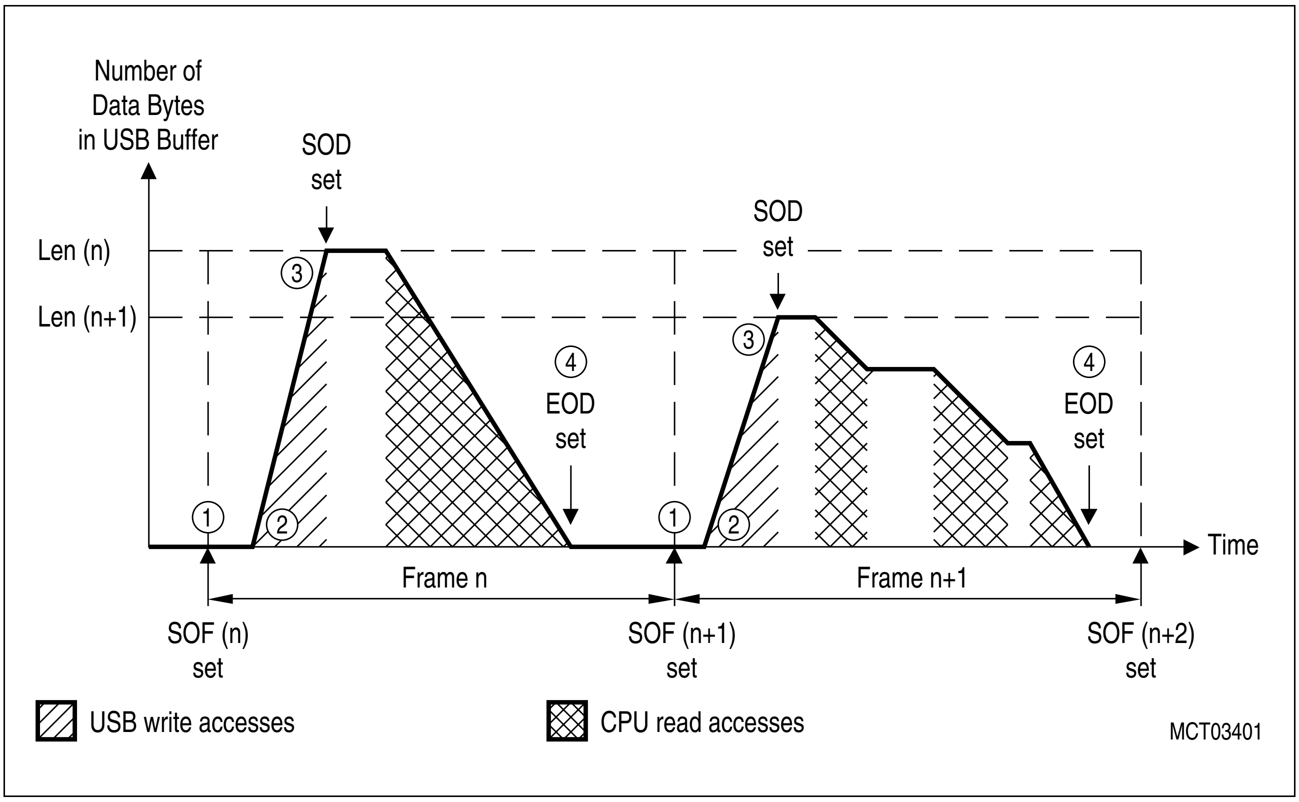


Figure 22-8 Single Buffer Mode : Standard USB Write Access

Note: The CPU accesses shown in the following diagrams assume that bit INCE in the corres-ponding endpoint control register is set.
A frame is the 1 ms time interval defined by the USB host.
Every frame begins with a SOF token (start-of-frame).

22.4.2.2 USB Read Access

Figure 22-9 shows the basic flowchart of a USB read access from one USB memory buffer in single buffer mode.

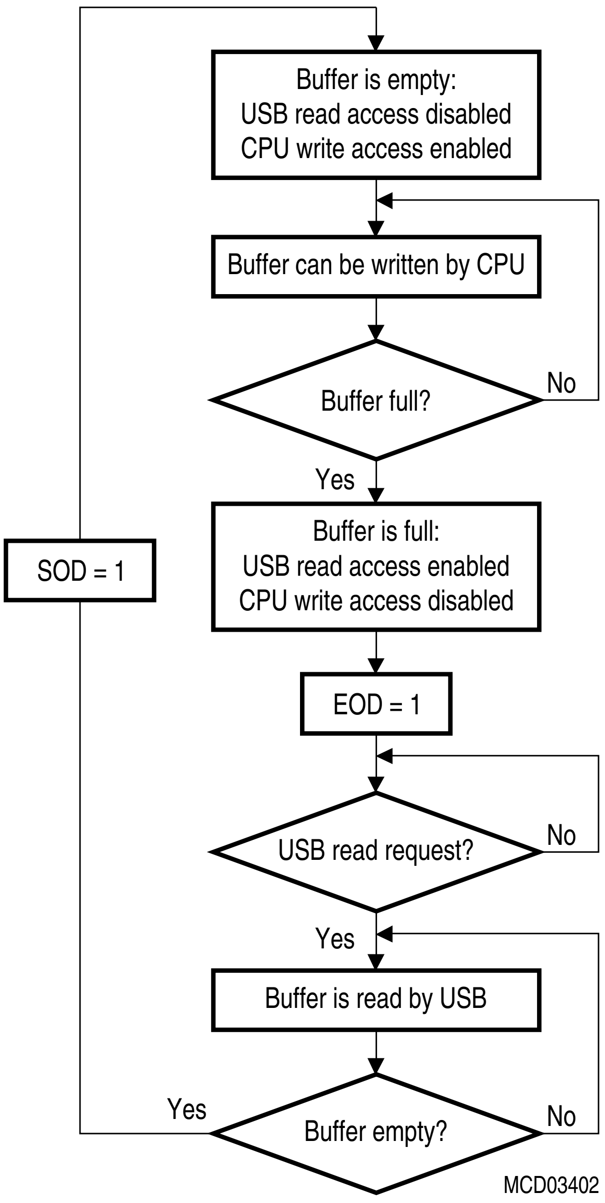


Figure 22-9 USB Read Access in Single Buffer Mode - Buffer Handling

The standard USB read access as shown in **figure 22-10** supports random and sequential CPU access mode of the USB memory. The memory buffer full condition is true when a predefined number of bytes (MaxLen) has been written by the CPU or when bit DONE has been set by software.

After SOF(n) occured at ① with a full USB memory buffer, the USB reads the buffer. Bit SOD is set at the end of the USB buffer read operation at ②, indicating an empty USB memory buffer. Now, the CPU can write again data into the USB memory buffer until a determined number (MaxLen) of bytes are transfered or until bit DONE has been set by software. The MaxLen value must be previously set by software. When the actual USB memory buffer address offset is equal to MaxLen, bit EOD is set at ③ to indicate a full buffer. The USB memory buffer address offset is automatically incremented with every CPU write access to USB memory buffer if bit INCE is set.

During the next frame (after SOF(n+1)) is set at ④) the USB memory buffer can be read by the USB. Bit SOD is set again when the USB memory buffer becomes empty again. If bit DONE is set by the CPU (at ⑤), the buffer is declared by the CPU to be full, even if the address offset does not reach the value of MaxLen.

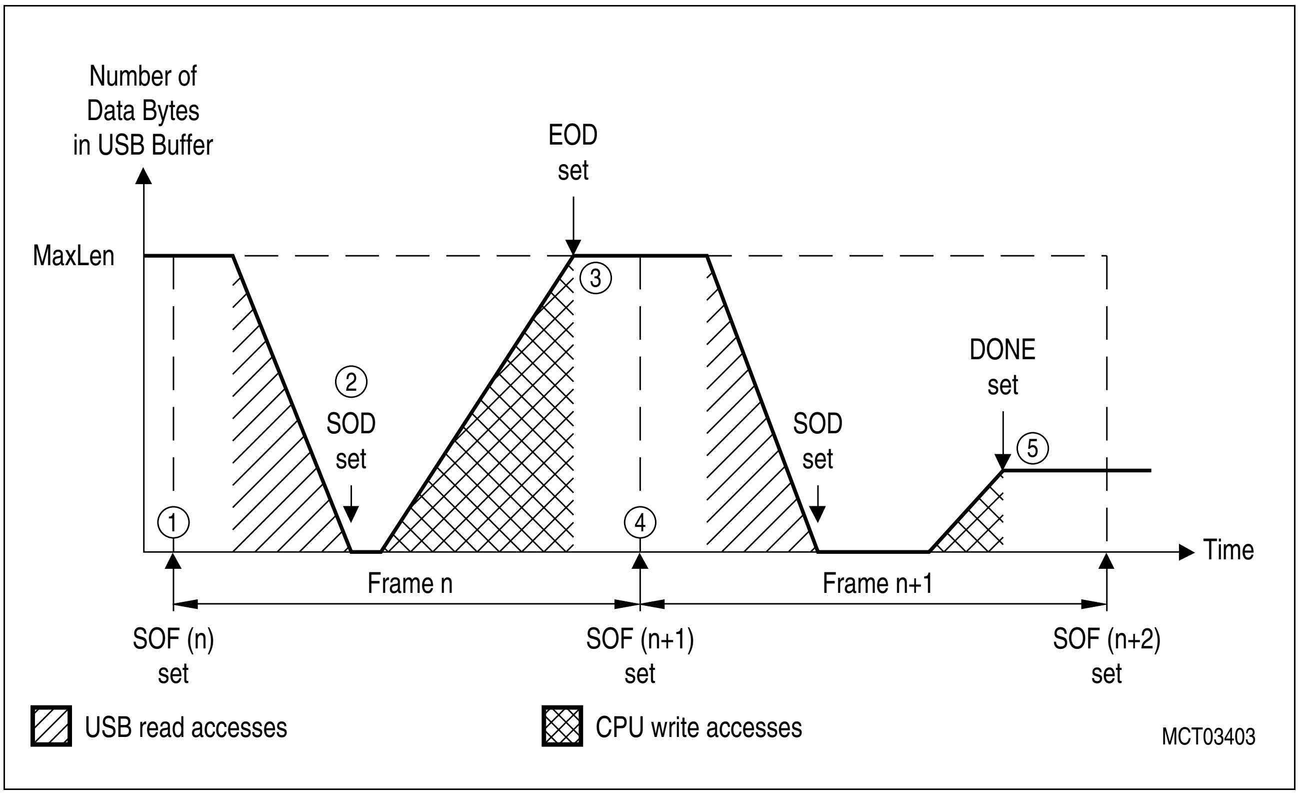


Figure 22-10 Single Buffer Mode : Standard USB Read Access

The start-of-frame-done enable feature (SOFDE=1) is useful for USB memory read accesses when the number of data bytes to be transferred from CPU to USB is not predictable (see **figure 22-11**). The CPU can write data as desired to USB memory until a SOF occures (every 1 ms). The automatic setting of bit SOF causes bit EOD to be set (at ①). This indicates the CPU that no CPU action on this buffer is required until a USB read operation has been finished (bit SOD set at ②). Setting of SOD indicates an empty USB memory to the CPU which can start again writing data into USB memory.

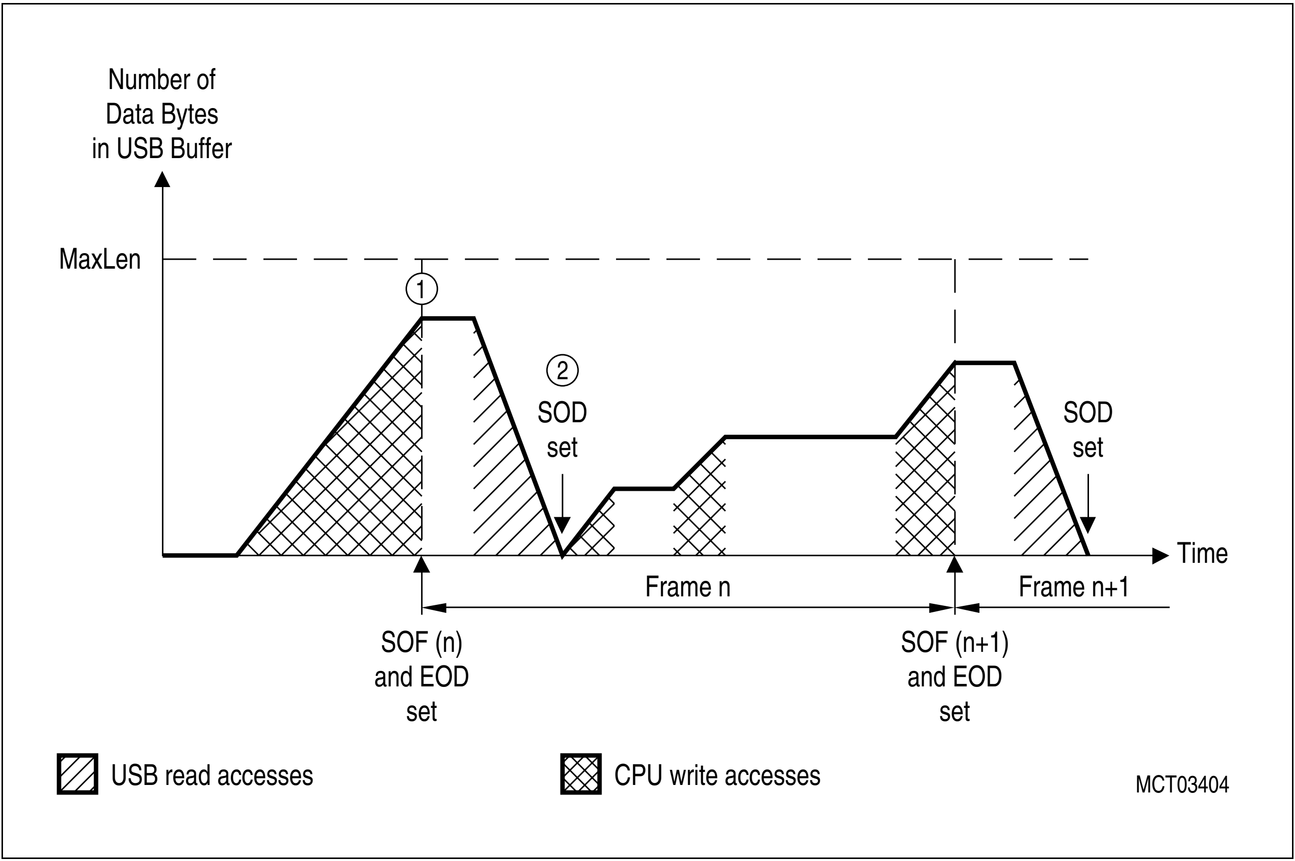


Figure 22-11 Single Buffer Mode : USB Read Access with Start-of-Frame-Done Enabled

22.4.3 Dual Buffer Mode

In dual buffer mode, both USB memory pages (page 0 and page 1) are used for data transfers. The logical assignment of the memory pages to CPU or USB module is automatically switched. The following two figures show the buffer handling concept in dual buffer mode for the USB module read access and USB write access.

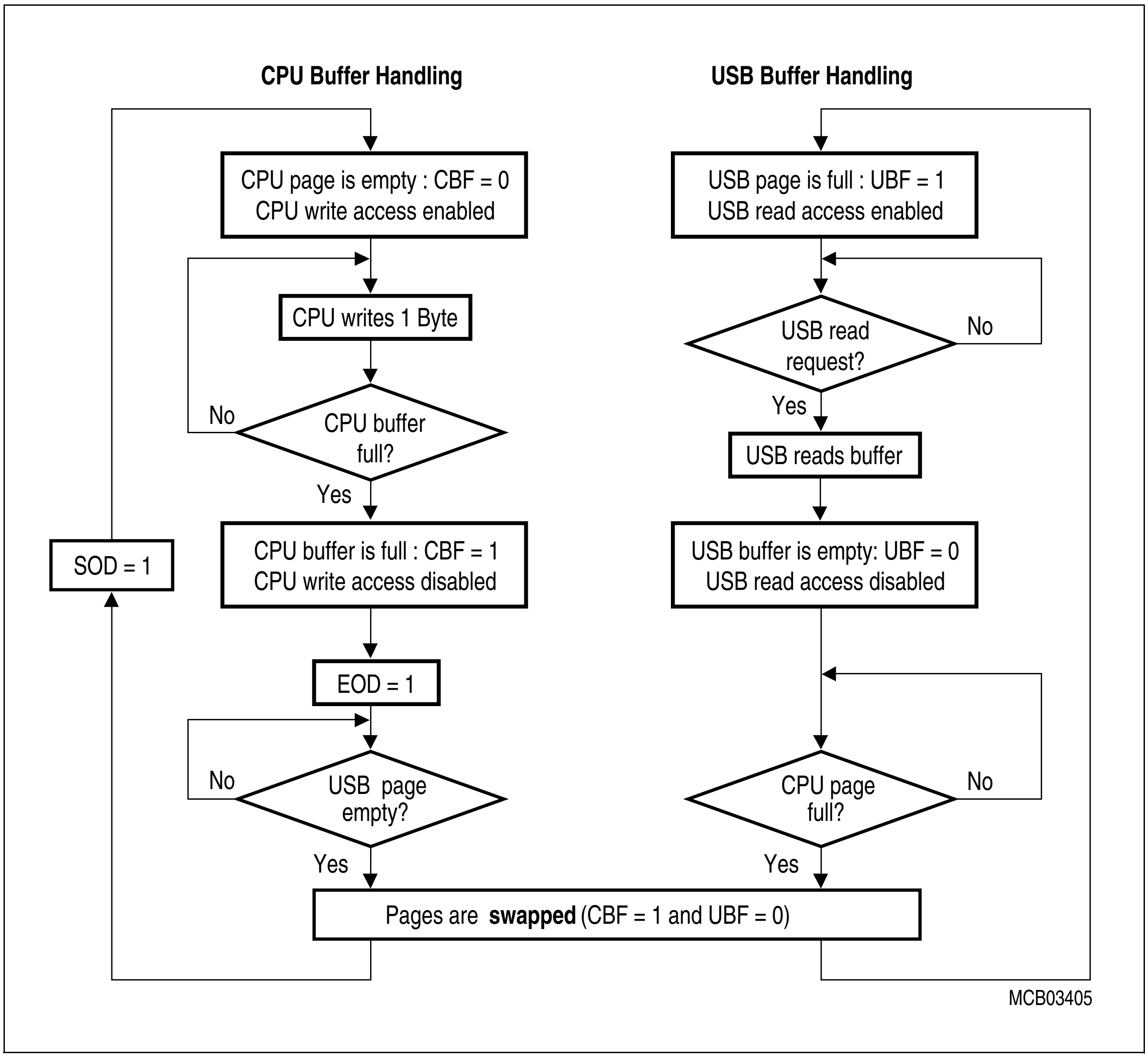


Figure 22-12 USB Read Access in Dual Buffer Mode - Buffer Handling

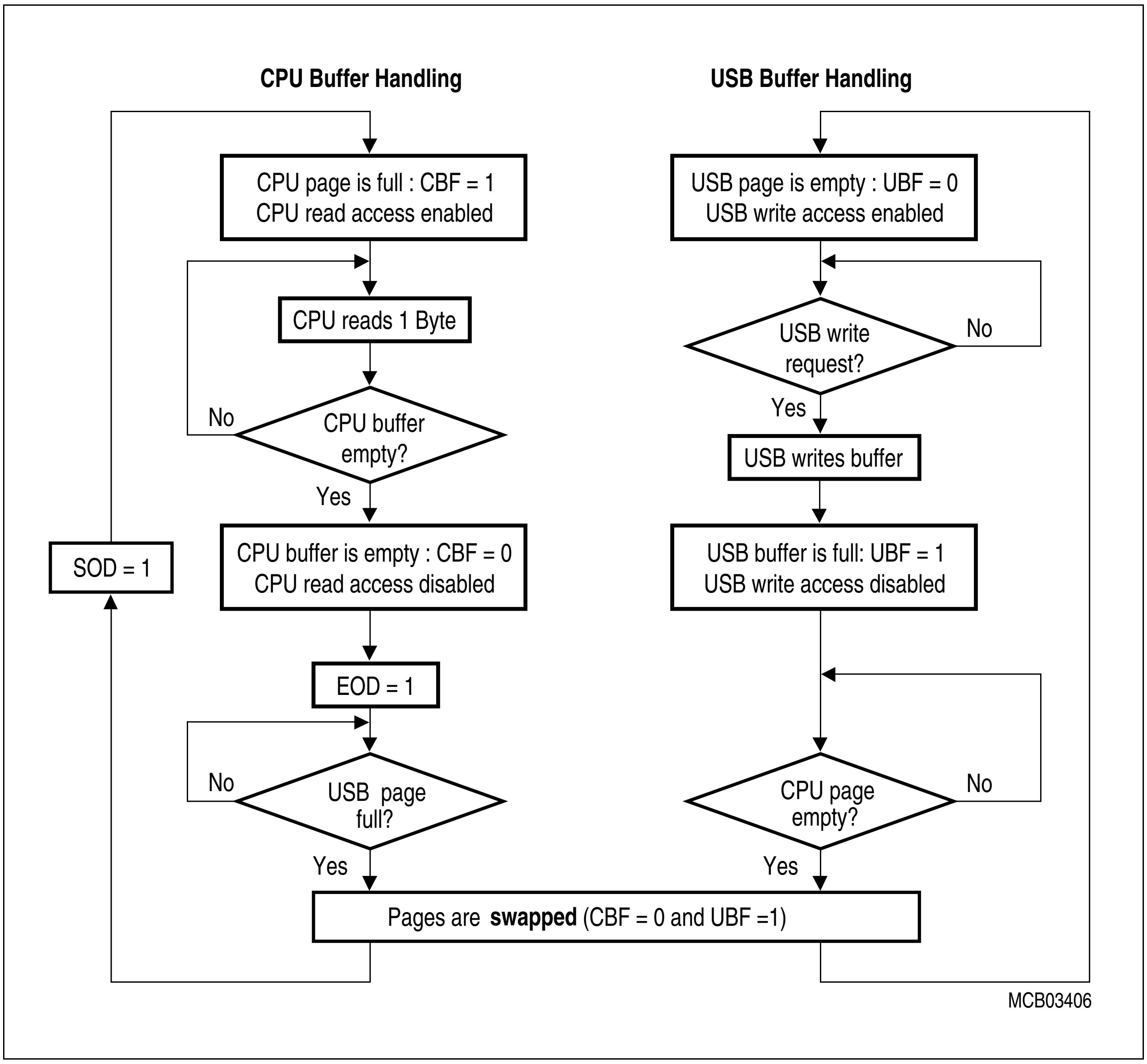


Figure 22-13 USB Write Access in Dual Buffer Mode - Buffer Handling

Figure 22-14 describes an example of a USB read operation in sequential mode with both buffers empty at the beginning of the USB read operation.

The CPU starts writing data with sequential access (INCE=1) to the buffer assigned to the CPU at ①. By definition, the buffer is full when MaxLen is reached at ②. The second buffer assigned to the USB is empty (UBF=0) and as a result both buffers are logically swapped. Now the buffer assigned to USB is full (UBF=1) and an USB read access can take place. After the USB read access, the buffer assigned to the USB is empty again with UBF=0. During the USB read access the CPU is still allowed to write into its assigned buffer. When reaching MaxLen at ③, the CPU buffer is full and both buffers are again logically swapped. The USB further execute its read access.

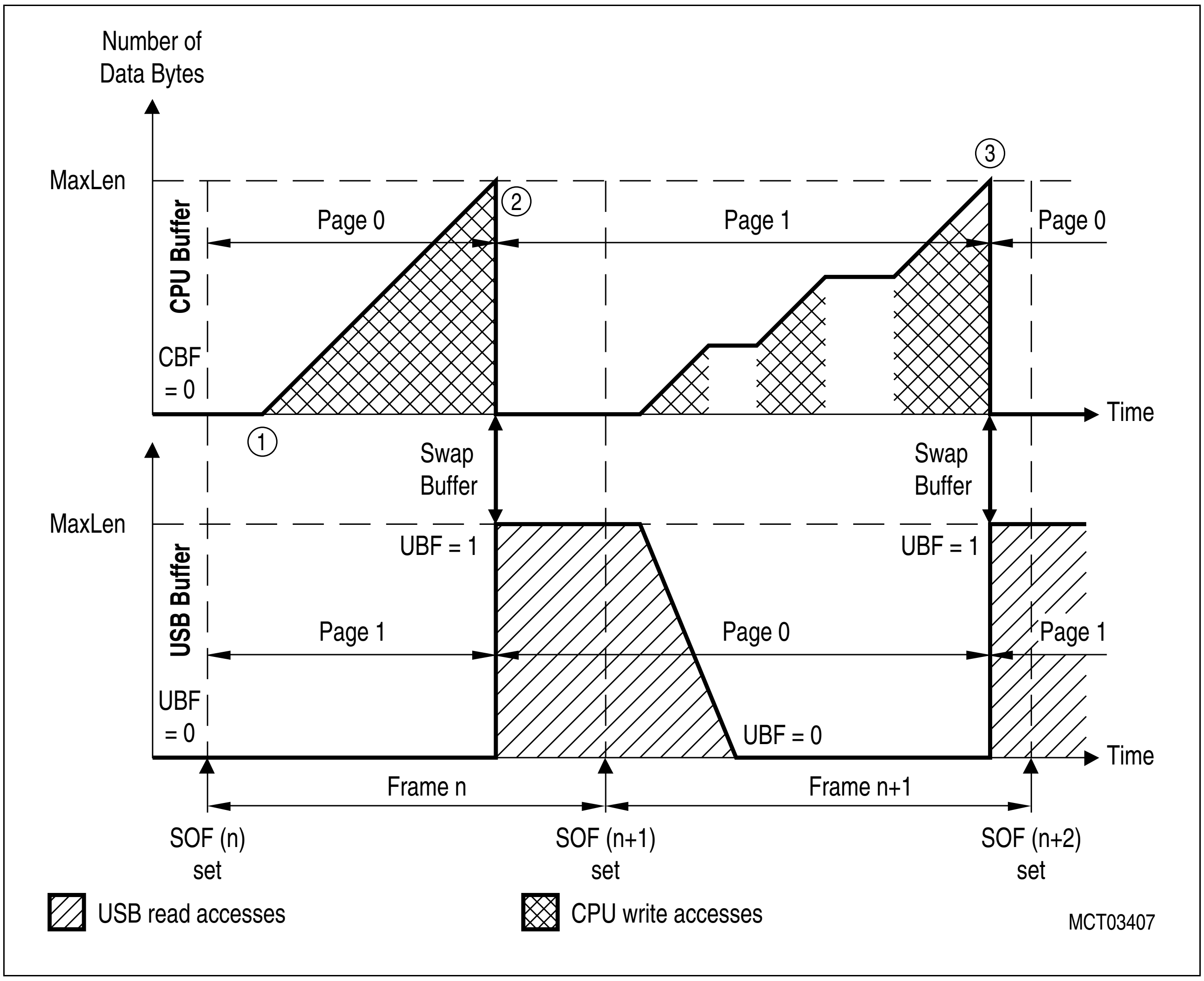


Figure 22-14 Dual Buffer Mode USB Read Access : Buffer Switching when MaxLen is reached

In dual buffer mode, the physical assignment of the USB memory pages (page 0 or page 1) to either CPU buffer or USB buffer is controlled automatically in the USB module and cannot be selected by software.

Another way to initiate buffer switching is setting bit DONE by software. This feature, which is shown in **figure 22-15** for USB read access, can be used to transfer a variable number of bytes. The maximum number of bytes to be transferred is still determined by MaxLen, which is not changed when bit DONE is set. The actual packet length (Len1 or Len2) is the number of bytes which have been written to the buffer before bit DONE is set.

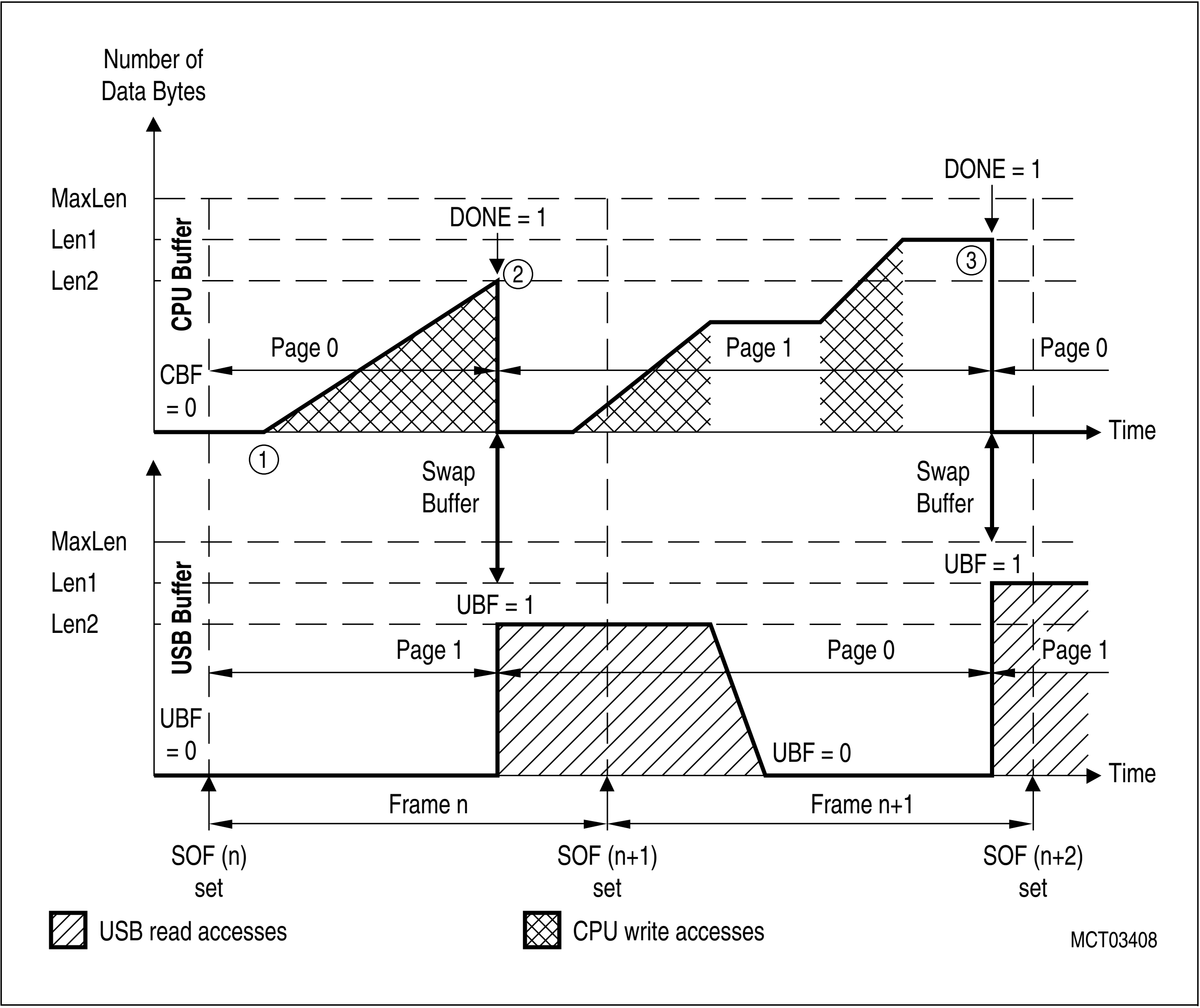


Figure 22-15 Dual Buffer Mode USB Read Access : Buffer Switching by Setting Bit DONE

If bit SOFDE is set, buffer switching is done automatically after SOF (start of frame) has been detected by the USB. **Figure 22-16** describes this functionality for USB read access for this case. The buffer which contains the latest data from the CPU is tagged valid for USB access (UBF=1) at ① and the buffers are swapped if the USB buffer is empty. After the USB read access has occured at ②, this buffer assigned to USB is empty again (UBF=0) and can be swapped again as soon as the CPU has filled its buffer (at ③). The number of bytes in the buffer is less or equal MaxLen. The MaxLen threshold is always active, but an occurrence of SOF (if SOFDE=1) or setting bit DONE by software are used to tag the CPU buffer full before reaching MaxLen.

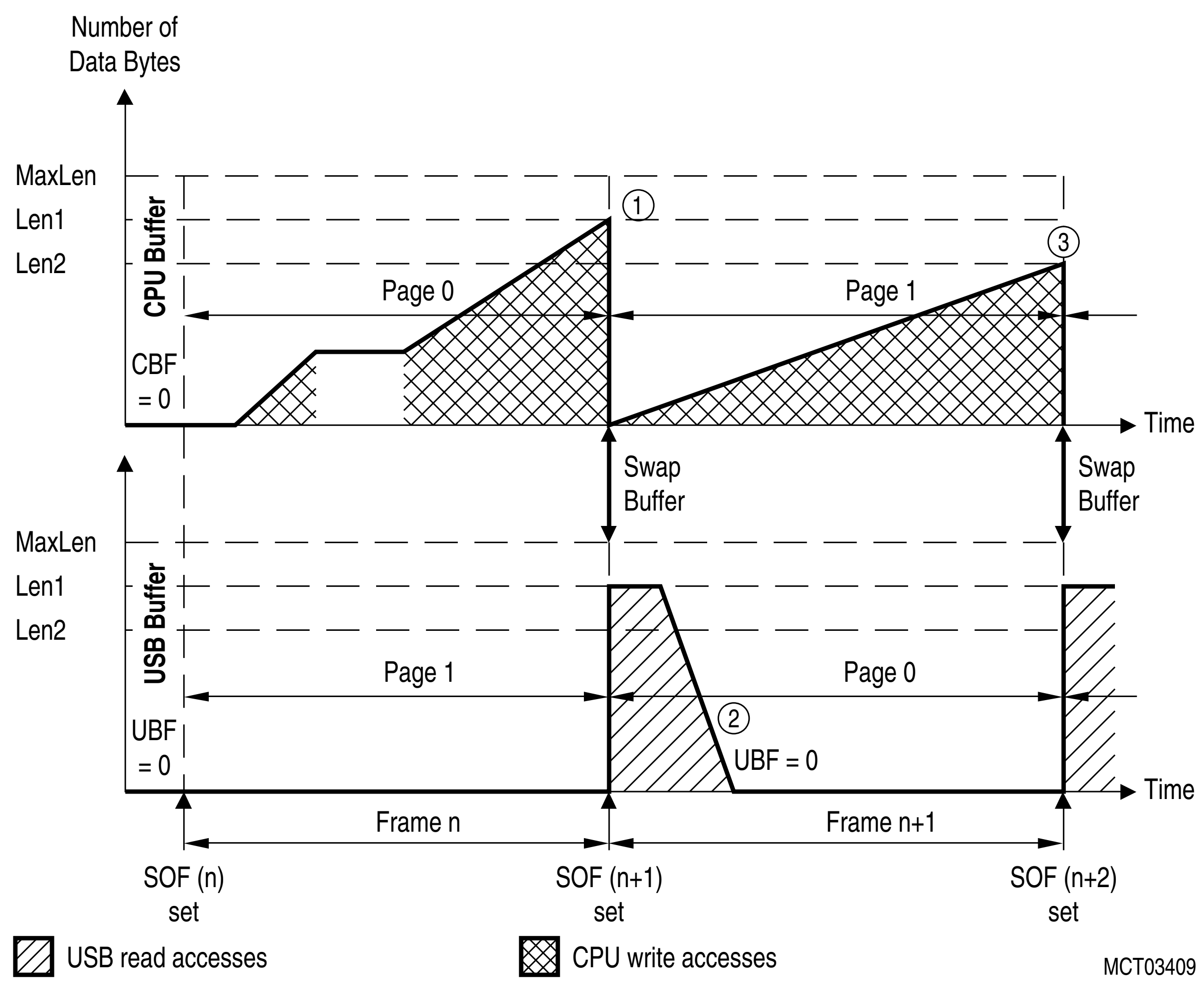


Figure 22-16 Dual Buffer Mode USB Read Access: Buffer Switching on SOF with SOFDE=1

If the number of data bytes to be transferred is greater than the maximum packet size (given by MaxLen), the data is split up automatically into packets, which are transferred one after the other. **Figure 22-17** gives an example of an USB read access, where data from the CPU is split up into two packets. When MaxLen is reached during the CPU write access, the currently active buffer is switched to USB side (UBF=1). The CPU continues writing data to the buffer. When the complete data packet has been written to the buffer by the CPU, bit DONE is set by software to indicate the end of the data packet (CBF=1). In the example, the USB buffer has not been read out. It is still full for the USB and can not be swapped (CBF=UBF=1). When the USB read access has occured (CBF=0), the buffers are automatically swapped and bit SOD is set.

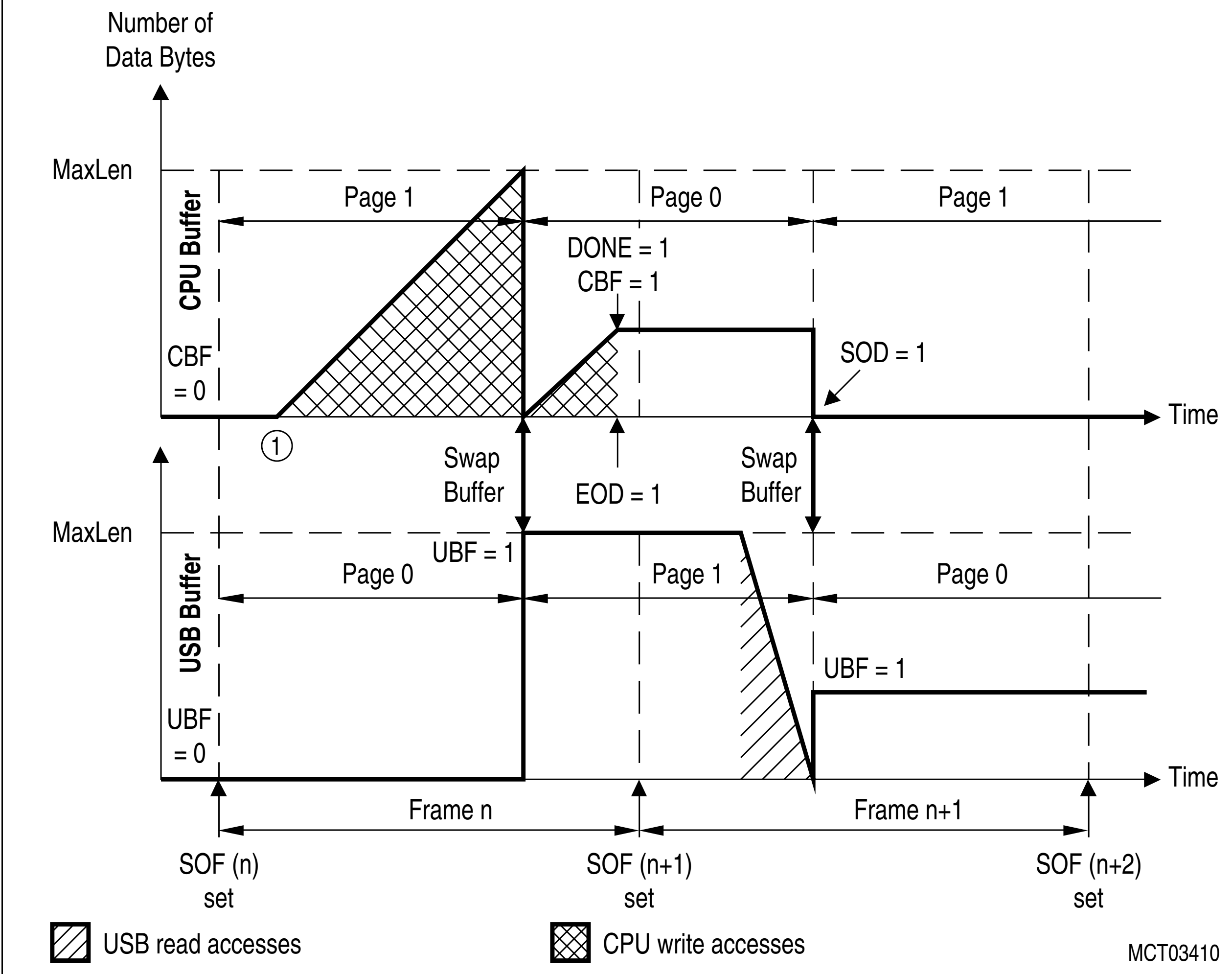


Figure 22-17 Double Buffer Mode USB Read Access: Data Length greater than Packet Length (MaxLen)

In general, three criteria for buffer switching are implemented in the USB module :

a) For sequential access, the address offset register ADROFFn is automatically incremented after each read or write action of the CPU. The address offset value (before incrementing) represents the number of bytes stored in USB memory for a specific endpoint. If the address offset value (after incrementing) reaches the value stored in endpoint length register EPLENn, the currently active buffer is tagged full (USB read access - all bytes have been written by CPU, CBF=1) or empty (USB write access - all bytes have been read by CPU, CBF=0).

b) When Bit DONE, which is located in the endpoint buffer status register EPBSn, is set, software buffer switching is initiated. This action is independent from the number of bytes which have been handled by the CPU (possible in sequential access mode (INCE=1) and random access mode (INCE=0)).
On CPU read accesses, the buffer is declared empty and bit CBF is cleared. If the buffer assigned to the USB is full (UBF=1), the buffers are immediately swapped. In this case, register EPLENn contains the number of received bytes.
On CPU write accesses, two different cases must be distinguished. For random accesses, the number of bytes of one packet is fixed by the value in register EPLENn and does not change. For sequential accesses, the number of written bytes represents the packet size. In this case, the actual value of register ADROFFn is transferred to register EPLENn when bit DONE is set.

c) The third criteria for buffer switching is the automatic buffer switching on detection of SOF (see **figure 22-17**). This feature can be individually enabled (SOFDE=1) or disabled (SOFDE=0) by software selectively for each endpoint.

22.4.4 Interruption of data transfers

A running data transfer to or from one of the FIFO buffers must not be interrupted to ensure a correct handling of the address pointer defined by ADROFFn.

Therefore all USB Endpoint related interrupts must be blocked while one of the FIFOs is currently accessed within an invoked InterruptServiceRoutine. For Endpoints 0 and 5-15 this is automatically realized, because they use the combined interrupt USBEPINT (see **Figure 8-2**).

22.5 Memory Buffer Organisation

The address generation of the USB memory buffer is based on the address offset and base address pointer. This scheme allows flexible and application specific buffer allocation and management. The length of an endpoint buffer can be up to 8, 16, 32 or 64 bytes. The start address of each endpoint buffer can be located to memory locations according to **table 22-2**.

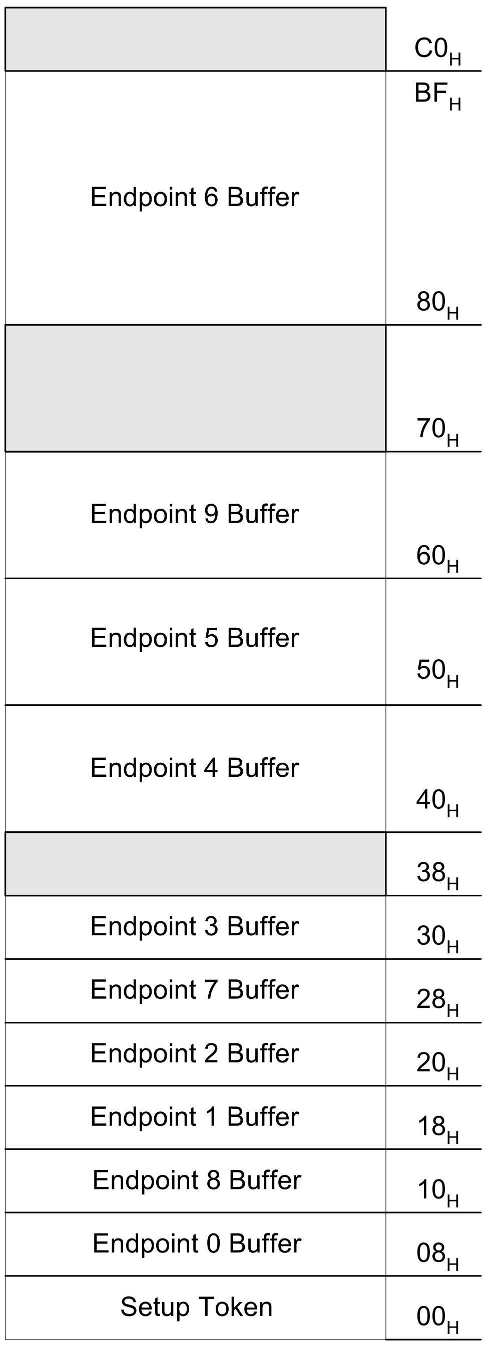
Table 22-2 Buffer Length and Base Address Values

Buffer Length	Valid Buffer Base Addresses
8 bytes	08_H, 10_H, 18_H, 20_H, 28_H, 30_H, 38_H, 40_H, 48_H, 50_H, 58_H, 60_H, 68_H, 70_H, 78_H, 80_H, 88_H, 90_H, 98_H, $A0_H$, $A8_H$, $B0_H$, $B8_H$, $C0_H$, $C8_H$, $D0_H$, $D8_H$, $E0_H$, $E8_H$, $F0_H$, $F8_H$
16 bytes	10_H, 20_H, 30_H, 40_H, 50_H, 60_H, 70_H, 80_H, 90_H, $A0_H$, $B0_H$, $C0_H$, $D0_H$, $E0_H$, $F0_H$
32 bytes	20_H, 40_H, 60_H, 80_H, $A0_H$, $C0_H$, $E0_H$
64 bytes	40_H, 80_H, $C0_H$

In order to avoid unused buffer space between two endpoint buffers, the largest buffer should be located at the highest address. This structure should be used to allocate USB memory for all endpoint buffers. The base address for the setup packet is always located at address 00_H. This leads to a typical USB buffer structure as shown in **figure 22-18**.

Endpoints which are not used in a configuration do not consume buffer space as the corresponding buffer length is 0.

The µC allocates the USB memory buffers for all endpoints by programming the endpoint base address (EPBAn) and the block size (EPLEn) within the total USB buffers.



Buffer Block	EPBAn	EPLENn
Endpoint 9	EPBA9=03_H	EPLEN9=10_H
Endpoint 8	EPBA8=08_H	EPLEN8=08_H
Endpoint 7	EPBA7=14_H	EPLEN7=08_H
Endpoint 6	EPBA6=40_H	EPLEN6=40_H
Endpoint 5	EPBA5=28_H	EPLEN5=10_H
Endpoint 4	EPBA4=20_H	EPLEN4=10_H
Endpoint 3	EPBA3=18_H	EPLEN3=08_H
Endpoint 2	EPBA2=10_H	EPLEN2=08_H
Endpoint 1	EPBA1=$0C_H$	EPLEN1=08_H
Endpoint 0	EPBA0=04_H	EPLEN0=08_H
Setup Token	Address 00_H on page 0	8 bytes

Note: ADDROFF = 00_H

EP-buf.vsd

Figure 22-18 Endpoint Buffer Allocation (Example: 9+1 Endpoints)

22.5.1 Memory Buffer Address Generation

The generation of a USB memory address for USB access (read or write) depends on the EPNum (endpoint number) information, which has been transmitted to the USB module during the software initialization procedure. The EPNum information is used for the selection of an endpoint specific base address register. As the maximum data packet length of each endpoint can individually be programmed, there are some fixed start addresses for the endpoints. The user program defines the base address for the first data byte of the corresponding endpoint by writing the endpoint specific base address register EPBAn. The length depends on the amount of data to be read or written. The user must take care to assign a buffer space at least as large as the maximum packet size of the endpoint.

The address of the currently accessed byte in the USB memory area of the selected endpoint is defined by an address offset which must be added to the endpoint base address in order to get the correct address for the USB memory buffer. The structure is shown in **figure 22-19**.

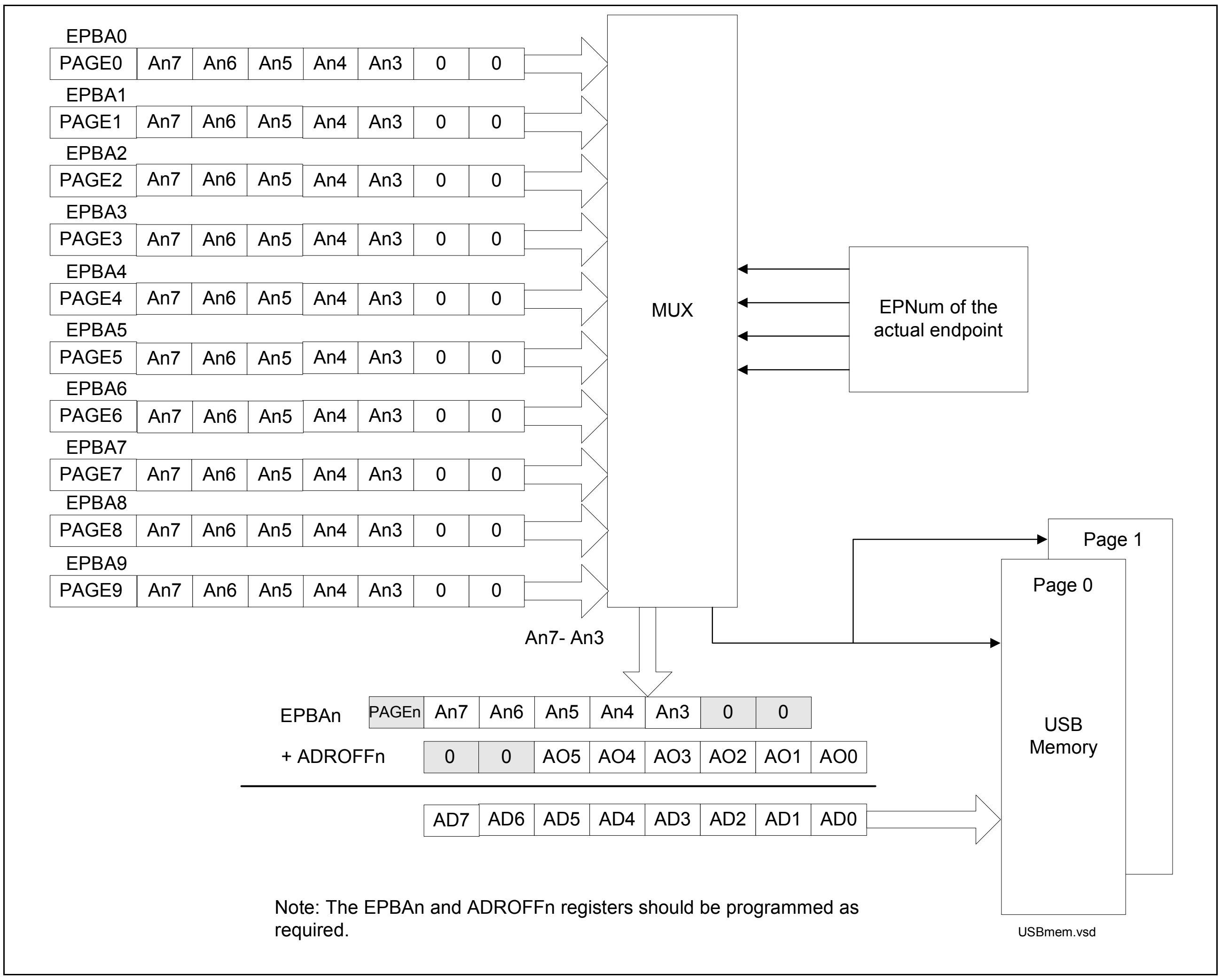

EPBA0
PAGE0 An7 An6 An5 An4 An3 0 0
EPBA1
PAGE1 An7 An6 An5 An4 An3 0 0
EPBA2
PAGE2 An7 An6 An5 An4 An3 0 0
EPBA3
PAGE3 An7 An6 An5 An4 An3 0 0
EPBA4
PAGE4 An7 An6 An5 An4 An3 0 0
EPBA5
PAGE5 An7 An6 An5 An4 An3 0 0
EPBA6
PAGE6 An7 An6 An5 An4 An3 0 0
EPBA7
PAGE7 An7 An6 An5 An4 An3 0 0
EPBA8
PAGE8 An7 An6 An5 An4 An3 0 0
EPBA9
PAGE9 An7 An6 An5 An4 An3 0 0
MUX
EPNum of the actual endpoint
Page 1
Page 0
USB Memory
An7- An3
EPBAn PAGEn An7 An6 An5 An4 An3 0 0
+ ADROFFn 0 0 AO5 AO4 AO3 AO2 AO1 AO0
AD7 AD6 AD5 AD4 AD3 AD2 AD1 AD0
Note: The EPBAn and ADROFFn registers should be programmed as required.
USBmem.vsd

Figure 22-19 USB Memory Address Generation (Example: 9+1 Endpoints)

22.6 USB block activation

If there is no traffic on the USB line, ie. a ’Start-of-Frame’ token has not been detected for more than 3 ms, the USB device interrupt DIRR.SBI (’suspend mode begins’) is set. The control bit DCR.SUSP is set to indicate the suspend mode. As soon as an USB specification compatible level has been detected, interrupt DIRR.SEI (’suspend mode ends’) is set.

Two clocks are connected to the USB block. First, the clock signal, which is connected to the memory management unit. This clock runs at the same frequency as the CPU.

For power saving purposes, this clock can be switched off by setting CLK_CONF.USB_DIS to ’1’.

Secondly, the 48 MHz clock signal, which clocks the USB device core itself. This clock can be disabled by setting bit UCLK in the DCR register. This clock can be switched off, if the USB module is in suspend mode. The DIRR.SEI (’suspend mode ends’) interrupt is detected even if the UCLK is still disabled.

22.7 Initialization of USB Module

The USB module must be functionally initialized by writing 6 configuration bytes which contain appropriate device information.

Note: The minimum CPU frequency for proper USB operation is 12 MHz.

A well defined procedure must be executed for the switching on the clock for the USB module:

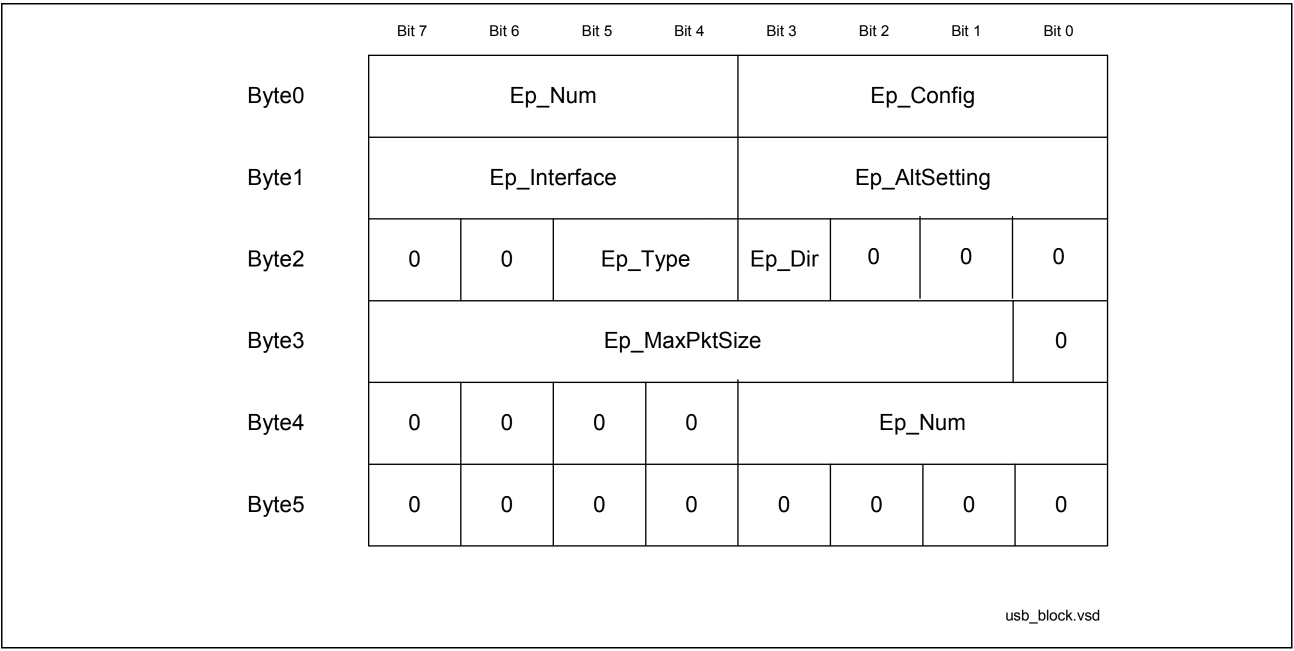
– PLL is switched on by setting bit PLLEN in register CLK_CONF
– waiting until PLL being locked (CLK_CONF.LOCK = 1)
– USB_DIS bit of the CLK_CONF register has to be cleared
– UCLK bit of the USB Device Control Register DCR has to be set to ’1’

The switch-on procedure after a hardware reset assures proper operation of the USB clock system. When the USB clock system is switched on, a software initialization procedure must follow.

This procedure must execute the following steps:

– Setting bit SWR in register DCR starts the software reset operation for the complete USB module.
– When the software reset is finished, bit SWR is cleared by hardware and bit DINIT is set to indicate the start of the initialization sequence.
– The USB module must be functionally initialized from the CPU by writing six configuration bytes for each endpoint to the USBVALn register. Thereafter, bit DONE0 in register EPBS0 must be set by software.

Figure 22-20 shows the 6-byte configuration block which must be transmitted by the CPU to the USB module via the USBVAL0 register for each endpoint.



	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte0	Ep_Num				Ep_Config			
Byte1	Ep_Interface				Ep_AltSetting			
Byte2	0	0	Ep_Type		Ep_Dir	0	0	0
Byte3	Ep_MaxPktSize							0
Byte4	0	0	0	0	Ep_Num			
Byte5	0	0	0	0	0	0	0	0

usb_block.vsd

Figure 22-20 USB Configuration Block

The six byte USB configuration block must be transfered sequentially (byte 0 to byte 5) from the CPU to the USB module for each endpoint (i.e. always for 16 endpoints) beginning with endpoint 0, followed by the USB configuration block for endpoint 1 and so on up to the USB configuration block for endpoint 15. EPNum is set for endpoints 0 up to15. After this action, bit DINIT is reset by hardware, and the software reset and initialization sequence are finished.

*Note: A software reset initiated by setting bit DCR.SWR to ’1’, which should be done at the earliest **4 clock cycles** after DCR.UCLK has been set to ’1’, resets also bit UCLK . This means that UCLK has to be set to ’1’ again. This reinforcement of DCR.UCLK has to be programmed earliest **20 clock cycles** after SWR has been set to ’1’.*

Table 22-3 Bitfield Definition of USB Configuration Block

EP_Num	This 4-bit field specifies the number of the endpoint (0-15) for which the actual configuration byte block is valid. This 4-bit field must be referenced in byte 0 and byte 4 of a configuration byte block.
EP_Config	This 4-bit field specifies the configuration number (0-3) to which this endpoint is configured to.
EP_Interface	This 4-bit field specifies the number of the interface (0-14) to which this endpoint is configured to.
EP_AltSetting	This 4-bit field selects the alternate setting of the correpsonding interface (EP_Interface) to which this endpoint is configured to.
EP_Type	This 2-bit field defines the type of the endpoint 00: Control endpoint 01: Isochronous endpoint 10: Bulk endpoint 11: Interrupt endpoint Endpoint 0 must be setup for control endpoint
EP_Dir	This bit defines the direction of the endpoint seen from the host 0: Out (packets to be transferred from Host to CPU) 1: In (packets to be transferred from CPU to Host)
EP_MaxPktSize	This 7-bit field defines the maximum packet size to be transferred to this endpoint within the range from 0 up to 64 bytes. The configuration of EPPackSize has to be in harmony with the USB specification.

22.8 USB Device Framework

22.8.1 Enumeration Process

The bus enumeration process consists of an interrogation sequence through which the USB host acquires information from the connected device, gives it an unique address, and assigns it a configuration value. In the simplest form, the process takes four steps:

Step 1: The host issues a *Get_Descriptor* command to the device through the default address using the control pipe. The device then provides information about itself, such as device class, vendor id, maximum packet size for endpoint 0, etc.

Step 2: The host sends an unique address in a data packet to the device using the *Set_Address* command. The device, under software control, gets the address through endpoint 0 and stores it.

Step 3: The host requests and reads the device configuration descriptor using the *Get_Configuration* command. The device responds with information about the number of interfaces and endpoints, endpoint transfer type, packet size and direction, maximum power requirements, power source, etc.

Step 4: The last step of the enumeration process is handled using the *Set_Configuration* command through which the host assigns a configuration value to the device.

After the enumeration process is complete, the device is configured and ready for USB data transmit and receive transactions.

22.8.2 Control Transfers

A control transfer consists of at least two and perhaps three stages. This chapter gives a short description of these stages of a control transfer and the associated control and status bits.

22.8.2.1 Setup Stage

A control transfer always begins with a setup stage that transfers information to a target device, defining the type of request being made to the USB device. The standard commands except the *Set_Descriptor*, *Get_Descriptor* and *Synch_Frame* commands are typically handled by the USB module automatically without CPU interaction (please refer also to page 562). If the command is not handled by the USB module automatically, a setup interrupt (SUI) indicates the end of a setup phase. Additionally, the status and control bits UBF (USB Buffer Full), CBF (CPU Buffer Full) and SOD (Start of Data) are reset.

22.8.2.2 Data Stage

This stage occurs only for requests that require data transfers. The direction of this data stage is always predicted to be from Host to Device (bit EPBSn.DIRn is automatically cleared after the setup stage occurred). The first data packet may immediately be sent from the Host to the control endpoint according to this configuration of bit EPBSn.DIRn, while NACK will automatically be returned from the Device to the Host in case of a USB read access. It also causes the direction bit to be changed (EPBSn.DIRn=1, USB read access).

The direction of the next transfer can also be predicted under software control (bit EPBSn.SETRDn) to be a USB read access (EPBSn.DIRn=1). This feature is used, if the direction of the data stage is known and the data packet to be transferred from the CPU to the Host is setup before the next USB access occurs.

Therefore, the direction bit must be changed under software control, to be able to transfer the data packet within the first USB read access. Status bit SOD is set under hardware control to indicate valid data to be read by the CPU in case of a USB write access, or data to be written by the CPU in case of a USB read access.

22.8.2.3 Status Stage

The status stage always occurs to report the result of the requested operation. A status stage initiated by the Host, but not terminated according to the setting EPBS0.ESP0=0, is indicated by a status interrupt (DIRR.STI). Bit EPBS0.ESP0 has to be set under software control to enable the acknowledge of the status stage.(Please refer also to page 562).

22.8.3 Standard Device Requests

Table 22-4 lists the standard requests possible. Only the Set_Descriptor, Get_Descriptor and Synch_Frame requests require intervention from the CPU (refer also to page 562). The Set_Configuration and Set_Interface standard device requests are handled by the UDC core and they are transparent to firmware. Since the device supports multiple device configurations, interfaces and alternate settings, a separate register CIAR informs the CPU which configuration, interface and alternate setting is active. For further information please refer to the USB Specification Version 1.1, Chapter 9.4 .

Table 22-4 Standard Device Requests

Request	Description
Get_Status	This request returns status for the specified recipient, which can be a device, interface or endpoint.
Clear_Feature	This request is used to clear or disable a specific feature. The UDC supports this command to clear the Endpoint_Stall feature for all supported logical endpoints and to clear the Device_Remote_ WakeUp feature.

Table 22-4 Standard Device Requests (cont'd)

Request	Description
Set_Feature	This request is used to set or enable a specific feature. The UDC supports the Set_Feature command to set the Endpoint_Stall feature for all supported logical endpoints and the Device_Remote _WakeUp feature.
Set_Address	This request sets the device address for all future device accesses. Stages after the initial Setup packet assume the same device address as the Setup packet. The USB device does not change its device address until after the Status stage of this request is completed successfully. This is the difference between this request and all other requests. For all other requests, the operation indicated must be completed before the Status stage.
Get_Descriptor	This request returns the specified descriptor if the descriptor exists. The standard request to a device supports three types of descriptors: DEVICE, CONFIGURATION & STRING. A request for a configuration descriptor returns the configuration descriptor, all interface descriptors, and endpoint descriptors for all the interfaces in a single request. Class-specific and/or vendor-specific descriptors follow the standard descriptors they extend or modify. All Get_Descriptor requests are passed to the µC by the UDC (refer also to page 562).
Set_Descriptor	This request may be used to update existing descriptors or add new descriptors.
Get_Configuration	This request returns the current device configuration value. If the returned value is zero, the device is not configured.
Set_Configuration	This request sets the device configuration.
Get_Interface	This request returns the selected alternate setting for the specified interface. This is a valid request only when the device is in the Configured state. Get_Interface requests are handled by the UDC automatically, the µC only gets a notification of it (refer also to page 562).

Table 22-4 Standard Device Requests (cont'd)

Request	Description
Set_Interface	This request allow the host to select an alternate setting for the specified interface.
Synch_Frame	This request is used to set and then report an endpoint's synchronization frame. When an endpoint supports isochronous transfers, the endpoint may also require per-frame transfers to vary in size according to a specific pattern. The host and the endpoint must agree on which frame the repeating pattern begins. The number of the frame in which the pattern began is returned to the host. This frame number is the one conveyed to the endpoint by the last SOF prior to the first frame of the pattern.

22.8.4 GET_DESCRIPTOR

After a GET_DESCRIPTOR request the device returns its attributes by sending a device DESCRIPTOR. Although certain Interfaces, Alternate Settings and Endpoints may not be listed in that DESCRIPTOR information, the UDC won't react to host requests to that objects by transmitting a STALL packet.

22.9 Onchip USB Transceiver

The INCA-D provides onchip receiver and transmitter circuitries which allows to connect the INCA-D directly to the USB bus. The USB driver circuitry is shown in **Figure 22-21**. The USB transceiver is capable of transmitting and receiving serial date at full speed (12 MBits/s) data rate. Transceiver and receiver can be separately disabled for power down mode operation. A single ended zero error condition (D+ and D- both at low level) can be detected.

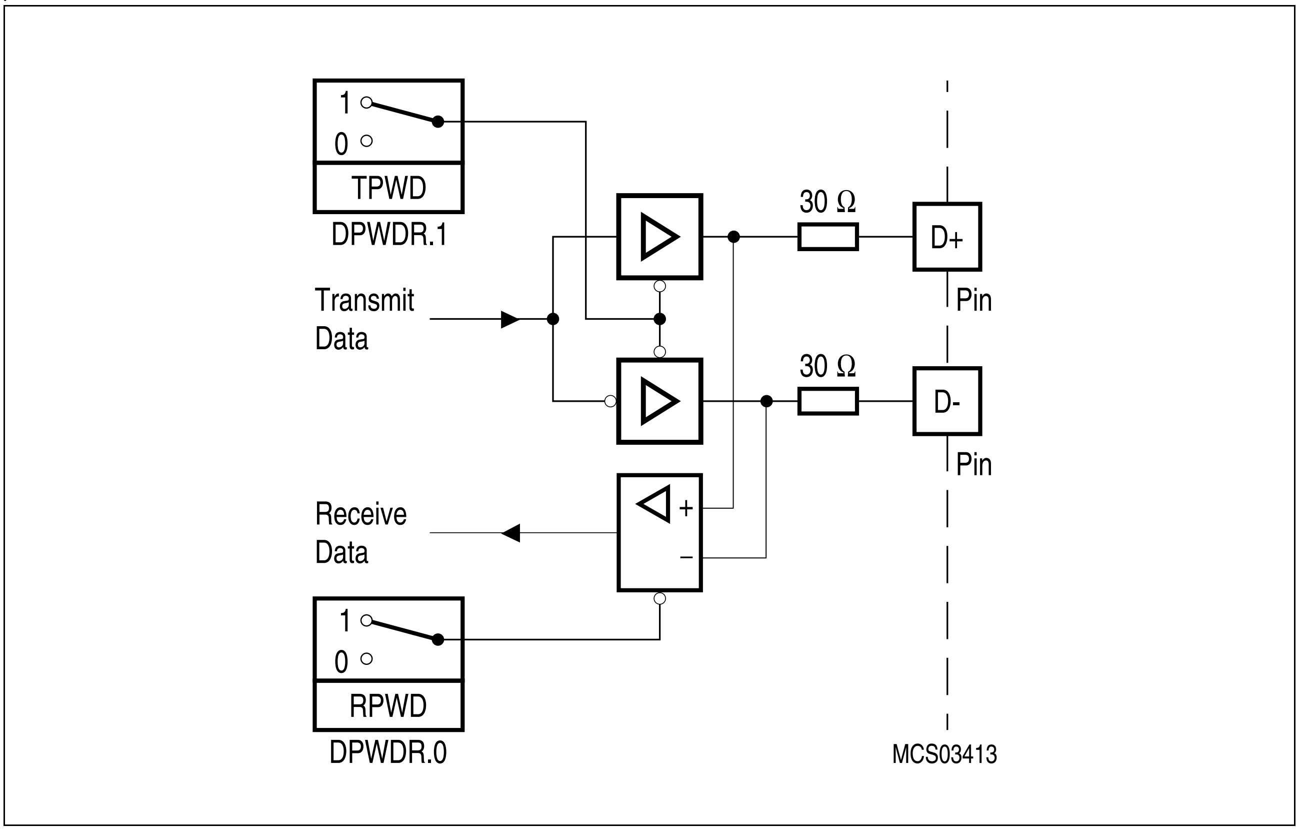


Figure 22-21 USB Onchip Driver Circuitry

The USB driver circuitry is a differential output driver which drives the USB data signal onto the cable of the USB bus. The static output swing of the transmitter is in low state below 0.3 V with a 1.5 kΩ load to 3.6 V and in high state above 2.8 V with a 15 kΩ load to V_{SS}. The driver outputs support tri-state operation to achieve bi-directional half duplex operation (control bits RPWD and TPWD). High impedance is also required to isolate the port from devices that are connected but powered down. The driver tolerates exposure to waveforms as specified in chapter 7.1 of the USB V1.1 Specification..

For a full speed USB connection the impedance of the USB driver must be between 29 Ω and 44 Ω. The data line rise and fall times are between 4 ns and 20 ns, smoothly rising or falling (monotonic), and are well matched to minimize RFI emissions and signal skew. **Figure 22-21** shows how the full speed driver is realized using two identical CMOS buffers. **Figure 22-22** shows the full speed driver signal waveforms.

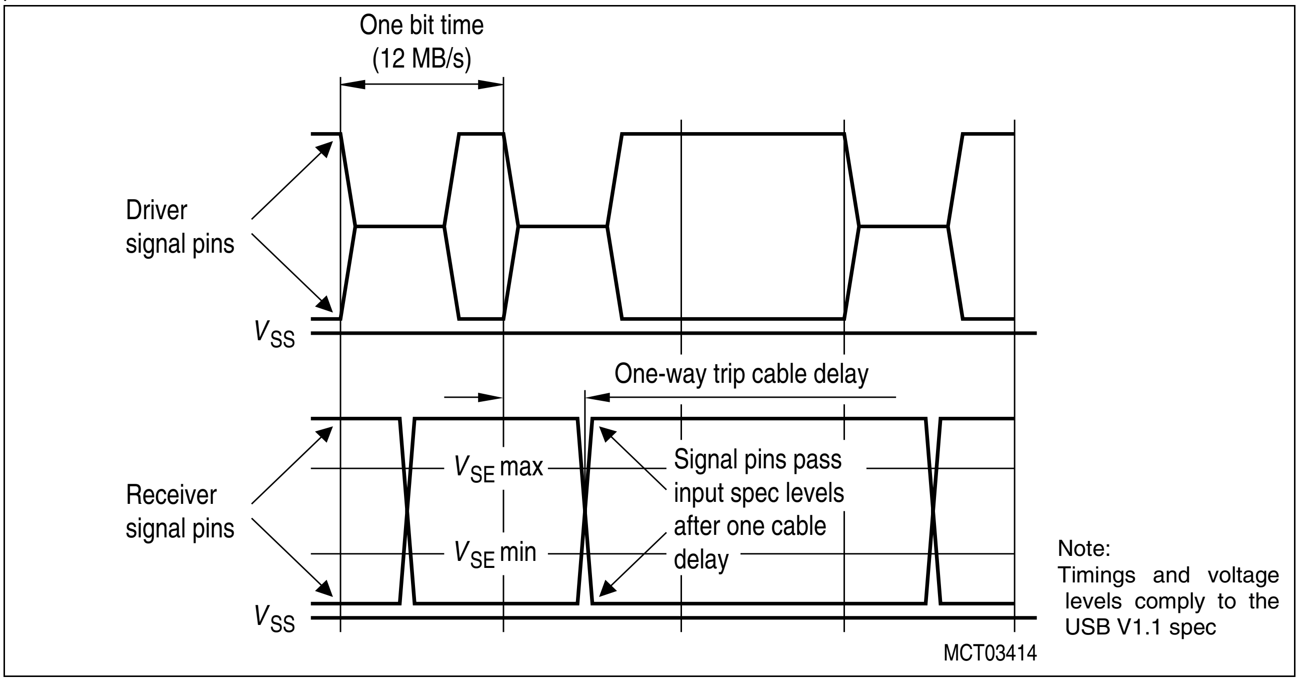


Figure 22-22 Full Speed USB Driver Signal Waveforms

Generally, full speed and low speed USB devices are differentiated by the position of the pullup resistor on the downstream end of the cable. Full speed devices are terminated with the pullup on the D+ line and low speed devices are terminated with the pullup in the D- line. As the INCA-D is a full speed device an external pull-up resistor (R_2) must be connected to the D+ line as shown in **figure 22-23**. The pullup terminator is a 1.5 kΩ resistor tied to a voltage source (VTERM) between 3.0 and 3.6 V referenced to the local ground. In some cases it might be necessary to hide the USB device from the host even when it is plugged in. To accomplish this, the pull-up resistor R_2 can be made switchable with a port a transistor. According to the USB specification, the voltage VTERM has be controlled by the voltage on the USB bus (VBUS=5V), ie. if the INCA-D is detached (VBUS=0), VTERM has to be 0V too.

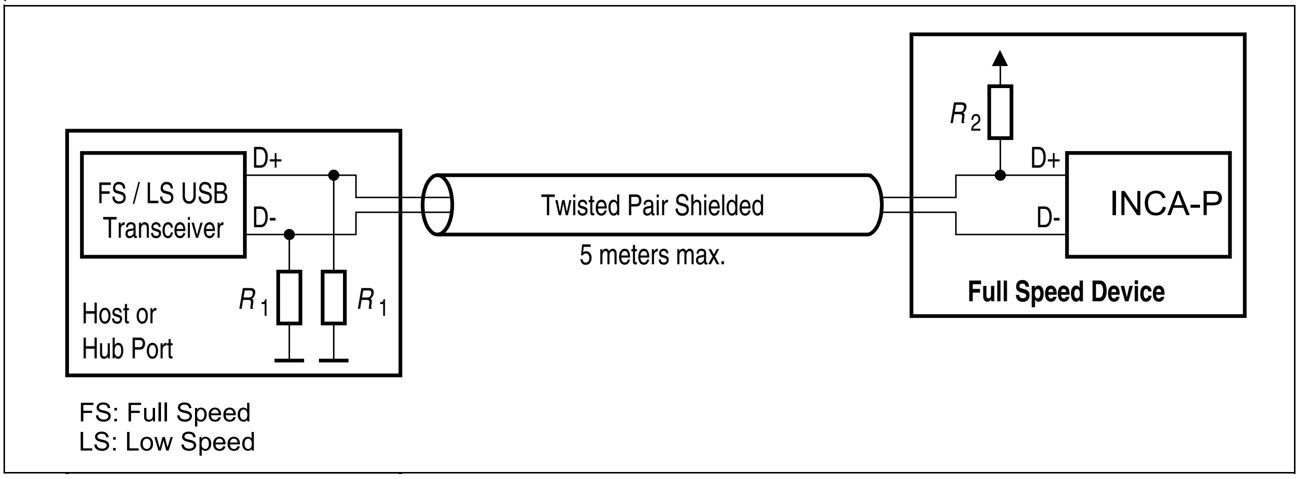


Figure 22-23 High Speed Device Cable and Resistor Connection

22.10 USB Registers Description.

Figure 22-24 explains the structure of the USB module registers.

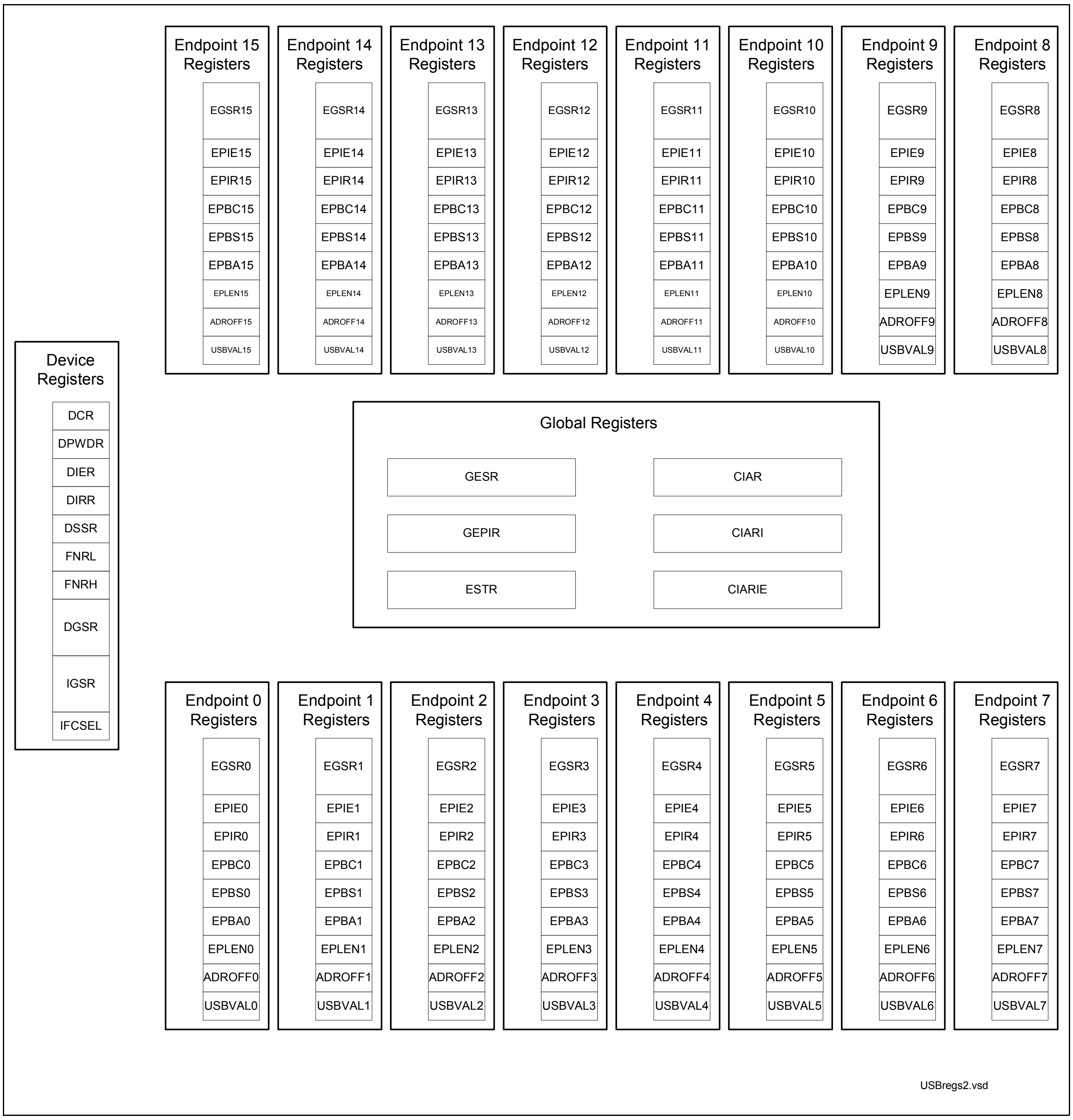


Figure 22-24 USB Register Set

For each of the 16 interfaces there is a 16-bit register that contains a status value which is transmitted to the USB host upon a Get_Status request (**figure 22-25**).

For setting the Interface Get Status Register (IGSR) for a specific interface, the µC first selects the interface number in the Interface Select Register (IFCSEL) and then writes the status value for the corresponding interface to IGSR. The status value itself (contained in IGSR) is fully specified by the USB spec. There is no function defined for that yet in USB V1.1 (as of 09.99), however the device fully supports the Get_Status request for potential changes in future.

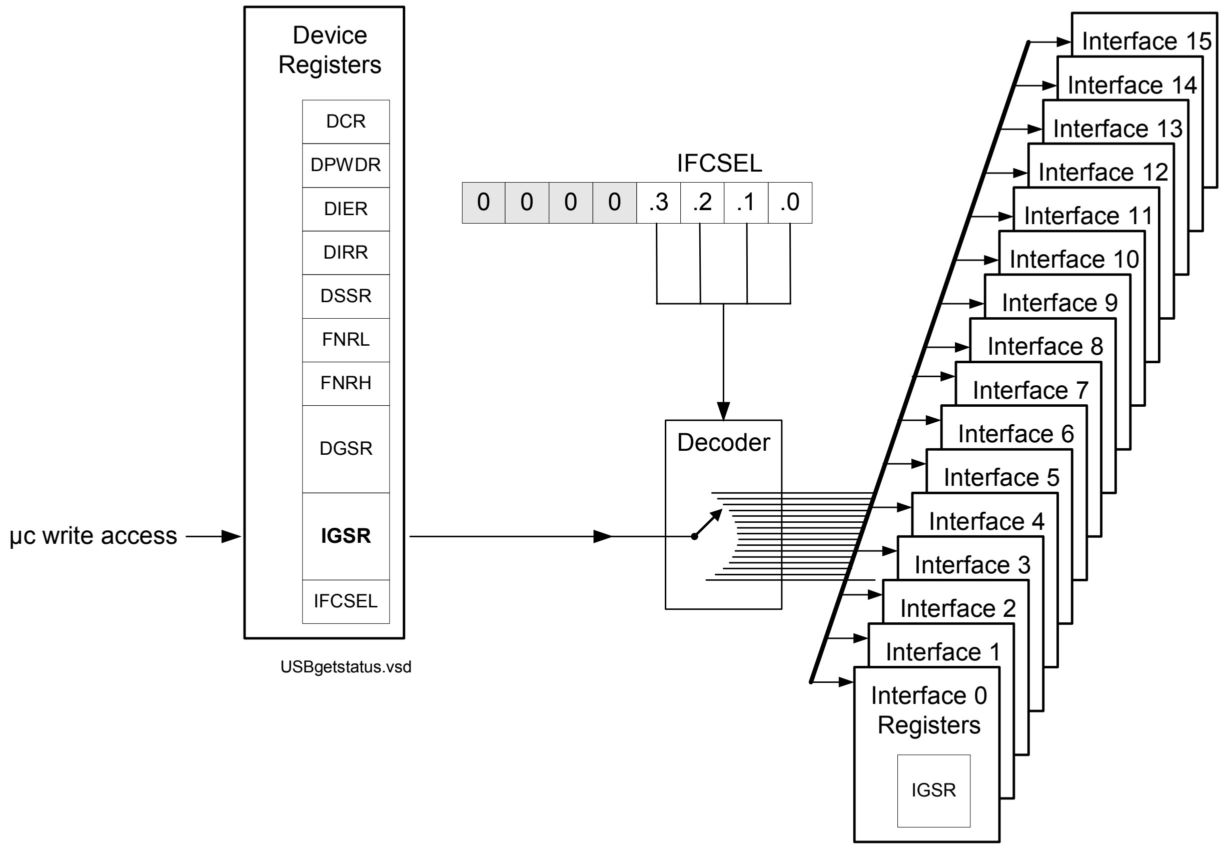


Figure 22-25 USB Interface Get_Status Registers

Two different kinds of registers are implemented in the USB module.

1.) The global registers and the device registers describe the basic functionality of the complete USB module and can be accessed via unique XBUS-SFR addresses. These are the

- GEPIR (Global Endpoint Interrupt Request Register)
- GESR (Global Endpoint Stall Register)
- CIAR (Configuration Request Register)
- CIARI (Configuration Request Interrupt Register)
- CIARIE (Configuration Request Interrupt Enable Register)
- DCR (Device Control Register)
- DPWDR (Device Power Down Register)
- DIER (Device Interrupt Enable Register)
- DIRR (Device Interrupt Request Register)
- DSSR (Device Setup and Status Register)
- FNRH, FNRL (Frame Number High/Low Registers)
- DGSR (Device Get Status Register)
- IGSR (Interface Get Status Register)
- IFCSEL (Interface Select Register)
- ESTR (Enable Setup Token)

2.) The endpoint specific functionality of the USB module is controlled via an individual set of registers for each of the 16 endpoints (n=0-15):

- EGSRn (Endpoint Get Status Register n)
- EPIEn (Endpoint Interrupt Enable Register n)
- EPIRn (Endpoint Interrupt Request Register n)
- EPBCn (Endpoint Buffer Control Register n)
- EPBSn (Endpoint Buffer Status Register n)
- EPBAn (Endpoint Base Address Register n)
- EPLENn (Endpoint Buffer Length Register n)
- ADROFFn (Address Offset Register n)
- USBVALn (USB Data Register n)

Standard Command Registers

The Set_Configuration and Set_Interface standard device commands are handled by the UDC and are transparent to the firmware. They appear at the end of the enumeration sequence, which can be detected by the first normal data transfer via USB.

22.10.1 USB Register handling interrupts

The registers listed in **table 22-5** and **table 22-6**, handle the interrupts which can be triggered by the USB module. For general interrupt handling refer also to **Chapter 8.2.1** on page 89.

The Base Address is 00’DD00$_{H,}$ i.e the listed addresses are address offsets.

Note: In the description of the USB module registers, bits are marked as rw, r or w. Bits marked as rw can be read and written. Bits marked as r can be read only. Writing any value to r bits has no effect. Bits marked as w are used to execute internal commands which are triggered by writing a 1. Writing a 0 to w bits has no effect. Reading w bits returns a 0.

Since the device supports multiple device configurations, interfaces and alternate settings, a separate register CIAR is needed to inform the device which configuration, interface and alternate setting is being selected. Any changes in the CIAR register through Set_Configuration or Set_Interface will generate an interrupt in the

Configuration Request Interrupt Register (CIARI). This interrupt can be enabled via a bit in the CIARIE register

Table 22-5 Device Specific USB Registers

Name	7	6	5	4	3	2	1	0	ADDR	R/W
GESRL	EPST7	EPST6	EPST5	EPST4	EPST3	EPST2	EPST1	EPST0	01_H	R/W
GESRH	EPST15	EPST14	EPST13	EPST12	EPST11	EPST10	EPST9	EPST8	02_H	R/W
CIARL	IF3	IF2	IF1	IF0	AS3	AS2	AS1	AS0	03_H	R
CIARH	0	0	0	0	CFG3	CFG2	CFG1	CFG0	04_H	R
GEPIRL	EPI7	EPI6	EPI5	0	0	0	0	EPI0	05_H	R
GEPIRH	EPI15	EPI14	EPI13	EPI12	EPI11	EPI10	EPI9	EPI8	06_H	R
CIARI	0	0	0	0	0	0	0	DRVI	07_H	R
CIARIE	IWIE	AIM	DSS	DRW	0	0	0	DRVIE	08_H	R/W
DCR	0	0	SWR	SUSP	DINIT	RSM	UCLK	0	09_H	R/W
DPWDR	0	0	0	0	0	0	TPWD	RPWD	$0A_H$	R/W
DIER	SE0IE	DAIE	0	SBIE	SEIE	STIE	SUIE	SOFIE	$0B_H$	R/W
DIRR	SE0I	DAI	0	SBI	SEI	STI	SUI	SOFI	$0C_H$	R
DSSR	0	0	0	0	0	0	IE	IR	$0D_H$	R
FNRL	FNR.7-0								$0E_H$	R
FNRH	0	0	0	0	0	FNR.10-8			$0F_H$	R
DGSRL	DST7	DST6	DST5	DST4	DST3	DST2	RWUP	PSTAT	10_H	R/W
DGSRH	DST15	DST14	DST13	DST12	DST11	DST10	DST9	DST8	11_H	R/W
IGSRL	IST7	IST6	IST5	IST4	IST3	IST2	IST1	IST0	12_H	R/W
IGSRH	IST15	IST14	IST13	IST12	IST11	IST10	IST9	IST8	13_H	R/W
IFCSEL	0	0	0	0	IF3	IF2	IF1	IF0	14_H	R/W
ESTR	STRE	0	0	0	0	0	0	0	15_H	R/W

ACTEP	0	0	0	0	EPN				16_H	R

Table 22-6 Endpoint Specific USB register

Name	7	6	5	4	3	2	1	0	ADDR	R/W
EGSR0L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	20_H	R/W
EGSR1L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	21_H	R/W
EGSR2L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	22_H	R/W
EGSR3L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	23_H	R/W
EGSR4L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	24_H	R/W
EGSR5L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	25_H	R/W
EGSR6L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	26_H	R/W
EGSR7L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	27_H	R/W
EGSR8L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	28_H	R/W
EGSR9L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	29_H	R/W
EGSR10L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	$2A_H$	R/W
EGSR11L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	$2B_H$	R/W
EGSR12L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	$2C_H$	R/W
EGSR13L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	$2D_H$	R/W
EGSR14L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	$2E_H$	R/W
EGSR15L	EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL	$2F_H$	R/W

EGSR0H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	30_H	R/W
EGSR1H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	31_H	R/W
EGSR2H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	32_H	R/W
EGSR3H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	33_H	R/W

EGSR4H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	34_H	R/W
EGSR5H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	35_H	R/W
EGSR6H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	36_H	R/W
EGSR7H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	37_H	R/W
EGSR8H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	38_H	R/W
EGSR9H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	39_H	R/W
EGSR10H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	$3A_H$	R/W
EGSR11H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	$3B_H$	R/W
EGSR12H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	$3C_H$	R/W
EGSR13H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	$3D_H$	R/W
EGSR14H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	$3E_H$	R/W
EGSR15H	EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8	$3F_H$	R/W

EPIE0	AIE0	NAIE0	RLEIE0	0	DNRIE0	NODIE0	EODIE0	SODIE0	40_H	R/W
EPIE1	AIE1	NAIE1	RLEIE1	0	DNRIE1	NODIE1	EODIE1	SODIE1	41_H	R/W
EPIE2	AIE2	NAIE2	RLEIE2	0	DNRIE2	NODIE2	EODIE2	SODIE2	42_H	R/W
EPIE3	AIE3	NAIE3	RLEIE3	0	DNRIE3	NODIE3	EODIE3	SODIE3	43_H	R/W
EPIE4	AIE4	NAIE4	RLEIE4	0	DNRIE4	NODIE4	EODIE4	SODIE4	44_H	R/W
EPIE5	AIE5	NAIE5	RLEIE5	0	DNRIE5	NODIE5	EODIE5	SODIE5	45_H	R/W
EPIE6	AIE6	NAIE6	RLEIE6	0	DNRIE6	NODIE6	EODIE6	SODIE6	46_H	R/W
EPIE7	AIE7	NAIE7	RLEIE7	0	DNRIE7	NODIE7	EODIE7	SODIE7	47_H	R/W
EPIE8	AIE8	NAIE8	RLEIE8	0	DNRIE8	NODIE8	EODIE8	SODIE8	48_H	R/W
EPIE9	AIE9	NAIE9	RLEIE9	0	DNRIE9	NODIE9	EODIE9	SODIE9	49_H	R/W

EPIE10	AIE10	NAIE10	RLEIE10	0	DNRIE10	NODIE10	EODIE10	SODIE10	$4A_H$	R/W
EPIE11	AIE11	NAIE11	RLEIE11	0	DNRIE11	NODIE11	EODIE11	SODIE11	$4B_H$	R/W
EPIE12	AIE12	NAIE12	RLEIE12	0	DNRIE12	NODIE12	EODIE12	SODIE12	$4C_H$	R/W
EPIE13	AIE13	NAIE13	RLEIE13	0	DNRIE13	NODIE13	EODIE13	SODIE13	$4D_H$	R/W
EPIE14	AIE14	NAIE14	RLEIE14	0	DNRIE14	NODIE14	EODIE14	SODIE14	$4E_H$	R/W
EPIE15	AIE15	NAIE15	RLEIE15	0	DNRIE15	NODIE15	EODIE15	SODIE15	$4F_H$	R/W

EPIR0	ACK0	NACK0	RLE0	0	DNR0	NOD0	EOD0	SOD0	50_H	R
EPIR1	ACK1	NACK1	RLE1	0	DNR1	NOD1	EOD1	SOD1	51_H	R
EPIR2	ACK2	NACK2	RLE2	0	DNR2	NOD2	EOD2	SOD2	52_H	R
EPIR3	ACK3	NACK3	RLE3	0	DNR3	NOD3	EOD3	SOD3	53_H	R
EPIR4	ACK4	NACK4	RLE4	0	DNR4	NOD4	EOD4	SOD4	54_H	R
EPIR5	ACK5	NACK5	RLE5	0	DNR5	NOD5	EOD5	SOD5	55_H	R
EPIR6	ACK6	NACK6	RLE6	0	DNR6	NOD6	EOD6	SOD6	56_H	R
EPIR7	ACK7	NACK7	RLE7	0	DNR7	NOD7	EOD7	SOD7	57_H	R
EPIR8	ACK8	NACK8	RLE8	0	DNR8	NOD8	EOD8	SOD8	58_H	R
EPIR9	ACK9	NACK9	RLE9	0	DNR9	NOD9	EOD9	SOD9	59_H	R
EPIR10	ACK10	NACK10	RLE10	0	DNR10	NOD10	EOD10	SOD10	$5A_H$	R
EPIR11	ACK11	NACK11	RLE11	0	DNR11	NOD11	EOD11	SOD11	$5B_H$	R
EPIR12	ACK12	NACK12	RLE12	0	DNR12	NOD12	EOD12	SOD12	$5C_H$	R
EPIR13	ACK13	NACK13	RLE13	0	DNR13	NOD13	EOD13	SOD13	$5D_H$	R
EPIR14	ACK14	NACK14	RLE14	0	DNR14	NOD14	EOD14	SOD14	$5E_H$	R
EPIR15	ACK15	NACK15	RLE15	0	DNR15	NOD15	EOD15	SOD15	$5F_H$	R

EPBC0	STALL0	0	0	GEPIE0	SOFDE0	INCE0	0	DBM0	60_H	R/W
EPBC1	STALL1	0	0	GEPIE1	SOFDE1	INCE1	0	DBM1	61_H	R/W
EPBC2	STALL2	0	0	GEPIE2	SOFDE2	INCE2	0	DBM2	62_H	R/W
EPBC3	STALL3	0	0	GEPIE3	SOFDE3	INCE3	0	DBM3	63_H	R/W
EPBC4	STALL4	0	0	GEPIE4	SOFDE4	INCE4	0	DBM4	64_H	R/W
EPBC5	STALL5	0	0	GEPIE5	SOFDE5	INCE5	0	DBM5	65_H	R/W
EPBC6	STALL6	0	0	GEPIE6	SOFDE6	INCE6	0	DBM6	66_H	R/W
EPBC7	STALL7	0	0	GEPIE7	SOFDE7	INCE7	0	DBM7	67_H	R/W
EPBC8	STALL8	0	0	GEPIE8	SOFDE8	INCE8	0	DBM8	68_H	R/W
EPBC9	STALL9	0	0	GEPIE9	SOFDE9	INCE9	0	DBM9	69_H	R/W
EPBC10	STALL10	0	0	GEPIE10	SOFDE10	INCE10	0	DBM10	$6A_H$	R/W
EPBC11	STALL11	0	0	GEPIE11	SOFDE11	INCE11	0	DBM11	$6B_H$	R/W
EPBC12	STALL12	0	0	GEPIE12	SOFDE12	INCE12	0	DBM12	$6C_H$	R/W
EPBC13	STALL13	0	0	GEPIE13	SOFDE13	INCE13	0	DBM13	$6D_H$	R/W
EPBC14	STALL14	0	0	GEPIE14	SOFDE14	INCE14	0	DBM14	$6E_H$	R/W
EPBC15	STALL15	0	0	GEPIE15	SOFDE15	INCE15	0	DBM15	$6F_H$	R/W

EPBS0	UBF0	CBF0	DIR0	ESP0	SETRD0	SETWR0	CLREP0	DONE0	70_H	R/W
EPBS1	UBF1	CBF1	DIR1	0	SETRD1	SETWR1	CLREP1	DONE1	71_H	R/W
EPBS2	UBF2	CBF2	DIR2	0	SETRD2	SETWR2	CLREP2	DONE2	72_H	R/W
EPBS3	UBF3	CBF3	DIR3	0	SETRD3	SETWR3	CLREP3	DONE3	73_H	R/W

EPBS4	UBF4	CBF4	DIR4	0	SETRD4	SETWR4	CLREP4	DONE4	74_H	R/W
EPBS5	UBF5	CBF5	DIR5	0	SETRD5	SETWR5	CLREP5	DONE5	75_H	R/W
EPBS6	UBF6	CBF6	DIR6	0	SETRD6	SETWR6	CLREP6	DONE6	76_H	R/W
EPBS7	UBF7	CBF7	DIR7	0	SETRD7	SETWR7	CLREP7	DONE7	77_H	R/W
EPBS8	UBF8	CBF8	DIR8	0	SETRD8	SETWR8	CLREP8	DONE8	78_H	R/W
EPBS9	UBF9	CBF9	DIR9	0	SETRD9	SETWR9	CLREP9	DONE9	79_H	R/W
EPBS10	UBF10	CBF10	DIR10	0	SETRD10	SETWR1 0	CLREP10	DONE10	$7A_H$	R/W
EPBS11	UBF11	CBF11	DIR11	0	SETRD11	SETWR1 1	CLREP11	DONE11	$7B_H$	R/W
EPBS12	UBF12	CBF12	DIR12	0	SETRD12	SETWR1 2	CLREP12	DONE12	$7C_H$	R/W
EPBS13	UBF13	CBF13	DIR13	0	SETRD13	SETWR1 3	CLREP13	DONE13	$7D_H$	R/W
EPBS14	UBF14	CBF14	DIR14	0	SETRD14	SETWR1 4	CLREP14	DONE14	$7E_H$	R/W
EPBS15	UBF15	CBF15	DIR15	0	SETRD15	SETWR1 5	CLREP15	DONE15	$7F_H$	R/W

EPBA0	PAGEn	An7	An6	An5	An4	An3	0	0	80_H	R/W
EPBA1	PAGEn	An7	An6	An5	An4	An3	0	0	81_H	R/W
EPBA2	PAGEn	An7	An6	An5	An4	An3	0	0	82_H	R/W
EPBA3	PAGEn	An7	An6	An5	An4	An3	0	0	83_H	R/W
EPBA4	PAGEn	An7	An6	An5	An4	An3	0	0	84_H	R/W
EPBA5	PAGEn	An7	An6	An5	An4	An3	0	0	85_H	R/W
EPBA6	PAGEn	An7	An6	An5	An4	An3	0	0	86_H	R/W
EPBA7	PAGEn	An7	An6	An5	An4	An3	0	0	87_H	R/W
EPBA8	PAGEn	An7	An6	An5	An4	An3	0	0	88_H	R/W

EPBA9	PAGEn	An7	An6	An5	An4	An3	0	0	89_H	R/W
EPBA10	PAGEn	An7	An6	An5	An4	An3	0	0	$8A_H$	R/W
EPBA11	PAGEn	An7	An6	An5	An4	An3	0	0	$8B_H$	R/W
EPBA12	PAGEn	An7	An6	An5	An4	An3	0	0	$8C_H$	R/W
EPBA13	PAGEn	An7	An6	An5	An4	An3	0	0	$8D_H$	R/W
EPBA14	PAGEn	An7	An6	An5	An4	An3	0	0	$8E_H$	R/W
EPBA15	PAGEn	An7	An6	An5	An4	An3	0	0	$8F_H$	R/W

EPLEN0	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	90_H	R/W
EPLEN1	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	91_H	R/W
EPLEN2	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	92_H	R/W
EPLEN3	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	93_H	R/W
EPLEN4	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	94_H	R/W
EPLEN5	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	95_H	R/W
EPLEN6	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	96_H	R/W
EPLEN7	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	97_H	R/W
EPLEN8	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	98_H	R/W
EPLEN9	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	99_H	R/W
EPLEN10	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	$9A_H$	R/W
EPLEN11	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	$9B_H$	R/W
EPLEN12	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	$9C_H$	R/W
EPLEN13	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	$9D_H$	R/W
EPLEN14	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	$9E_H$	R/W
EPLEN15	0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0	$9F_H$	R/W

ADROFF0	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A0_H$	R/W
ADROFF1	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A1_H$	R/W
ADROFF2	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A2_H$	R/W
ADROFF3	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A3_H$	R/W
ADROFF4	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A4_H$	R/W
ADROFF5	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A5_H$	R/W
ADROFF6	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A6_H$	R/W
ADROFF7	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A7_H$	R/W
ADROFF8	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A8_H$	R/W
ADROFF9	0	0	AO5	AO4	AO3	AO2	AO1	AO0	$A9_H$	R/W
ADROFF10	0	0	AO5	AO4	AO3	AO2	AO1	AO0	AA_H	R/W
ADROFF11	0	0	AO5	AO4	AO3	AO2	AO1	AO0	AB_H	R/W
ADROFF12	0	0	AO5	AO4	AO3	AO2	AO1	AO0	AC_H	R/W
ADROFF13	0	0	AO5	AO4	AO3	AO2	AO1	AO0	AD_H	R/W
ADROFF14	0	0	AO5	AO4	AO3	AO2	AO1	AO0	AE_H	R/W
ADROFF15	0	0	AO5	AO4	AO3	AO2	AO1	AO0	AF_H	R/W

USBVAL0	USBVAL0.7-0	$B0_H$	R/W
USBVAL1	USBVAL1.7-0	$B1_H$	R/W
USBVAL2	USBVAL2.7-0	$B2_H$	R/W
USBVAL3	USBVAL3.7-0	$B3_H$	R/W
USBVAL4	USBVAL4.7-0	$B4_H$	R/W
USBVAL5	USBVAL5.7-0	$B5_H$	R/W

USBVAL6	USBVAL6.7-0	$B6_H$	R/W
USBVAL7	USBVAL7.7-0	$B7_H$	R/W
USBVAL8	USBVAL8.7-0	$B8_H$	R/W
USBVAL9	USBVAL9.7-0	$B9_H$	R/W
USBVAL10	USBVAL10.7-0	BA_H	R/W
USBVAL11	USBVAL11.7-0	BB_H	R/W
USBVAL12	USBVAL12.7-0	BC_H	R/W
USBVAL13	USBVAL13.7-0	BD_H	R/W
USBVAL14	USBVAL14.7-0	BE_H	R/W
USBVAL15	USBVAL15.7-0	BF_H	R/W

22.11 Detailed Register Description - USB Device Registers

22.11.1 Global Endpoint Stall Register (GESR)

Reset value: 0000_H Address: 01_H

	7	6	5	4	3	2	1	0
01_H	EPST7	EPST6	EPST5	EPST4	EPST3	EPST2	EPST1	EPST0
	rw	rw	rw	rw	rw	rw	rw	rw
	15	14	13	12	11	10	9	8
02_H	EPST15	EPST14	EPST13	EPST12	EPST11	EPST10	EPST9	EPST8
	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
EPST15-0	**Endpoint x Stalled** EPSTx indicates if the corresponding endpoint is stalled ('1') or not ('0').

22.11.2 Configuration Request Register (CIAR)

Reset value: 0000_H Address: 03_H

	7	6	5	4	3	2	1	0
03_H	IF3	IF2	IF1	IF0	AS3	AS2	AS1	AS0
	r	r	r	r	r	r	r	r
	15	14	13	12	11	10	9	8
04_H	0	0	0	0	CFG3	CFG2	CFG1	CFG0
	r	r	r	r	r	r	r	r

Bit	Function
CFGn	**Configuration Value** Selects one of 4 Configurations (Config 0 - 3) *Note: The USB specification v1.1 supports 1 configuration in addition to CONFIG0 only. (CONFIG0 and CONFIG1).*
IFn	**Interface Number** Selects one of 15 Interfaces (Ifc 0 - 14)
ASn	**Alternate Setting** Selects one of 7 Alternate Settings (AS 0 - 6)

22.11.3 Global Endpoint Interrupt Request Register (GEPIR)

The global endpoint interrupt request register contains one flag for each endpoint which indicates whether one or more endpoint specific interrupt requests has become active. Because EP1-EP4 have dedicated interrupt requests (see **Table 8-1**), only bits EPI0 and EPI5-EPI15 have to be set accordingly.

If a request flag in GEPIR is set, it is automatically cleared after a read operation of the corresponding interrupt registers EPIRn.

Reset value: 0000_H Address: 05_H

	7	6	5	4	3	2	1	0
05_H	EPI7	EPI6	EPI5	0	0	0	0	EPI0
	r	r	r	r	r	r	r	r
	15	14	13	12	11	10	9	8
06_H	EPI15	EPI14	EPI13	EPI12	EPI13	EPI10	EPI9	EPI8
	r	r	r	r	r	r	r	r

Bit	Function
EPIn	**Endpoint Interrupt n request flag** If EPIn is set, an endpoint interrupt n request is pending.

22.11.4 Configuration Request Interrupt Register (CIARI)

The Configuration, Interface & Alternate Setting Interrupt Register (CIARI) sends an interrupt to the µC whenever the host programs multiple device configurations or interfaces.

Reset value: 00_H Address: 07_H

	7	6	5	4	3	2	1	0
07_H	0	0	0	0	0	0	0	DRVI
	r	r	r	r	r	r	r	r

The interrupt contained in the CIARI register can be masked in the CIARIE register.

Bit	Function
DRVI	**Device Request Value Interrupt** Bit DRVI is set each time the host sends a device request that contains one or more of the following: • Configuration Value • Interface • Alternate Setting • The flag is cleared by reading the CIARI register. The interrupt has to be enabled by bit DRVIE before being used.

22.11.5 Configuration Request Interrupt Enable (CIARIE)

Reset value: 00_H Address: 08_H

	7	6	5	4	3	2	1	0
08_H	IWIE	AIM	DSS	DRW	0	0	0	DRVIE
	rw	rw	r	r	r	r	r	rw

Bit	Function
AIM	**All Interrupts Masked** When 1, all USB endpoint related interrupts are masked
IWIE	Interrupt within interrupt enable When 1, the address offset of the current endpoint is not cleared before read.
DSS	Device Status Select When 1, the Status Phases for the standard commands {Set/Clear_Feature, Set_Configuration, Set_Interface} have also to be enabled by setting bit EPBS0.EPS0. **(see Note below)** By default this is only activated for {Get_Descriptor, Set_Descriptor, Synch_Frame} If the EPS0 bit is not set, a status interrupt (STI) will be issued;
DRW	Device Remote Wakeup If DRW=1 and DSS=1, the "remote wakeup feature" is disabled and a Set/Clear_Feature command will cause a STALL response. Otherwise "remote wakeup" is enabled and a Set/Clear_Feature command will cause sending of an ACK response.
DRVIE	**Device Request Value Interrupt Enable** When 1, this bit enables the device request value interrupt.

*Note: If the host initiates a standard request (Get_Descriptor, Set_Descriptor, Synch_Frame, Set/Clear_Feature, Set_Configuration, Set_Interface}, a setup interrupt (SUI) occurs. With this interrupt the corresponding setup information is read and the specific standard request can be determined. The four requests {Set/Clear_Feature, Set_Configuration, Set_Interface} will be handled automatically by the USB module. However, for the requests {Get_Descriptor, Set_Descriptor, Synch_Frame} the setup interrupt SUI has to be handled by the CPU, i.e. it is **not** handled automatically by the core.*

22.11.6 Device Control Register (DCR)

The device control register includes control and status bits which indicate the current status of the USB module and the status of the USB bus.

Reset value: 00000000_B Address: 09_H

	7	6	5	4	3	2	1	0
09_H	0	0	SWR	SUSP	DINIT	RSM	UCLK	0
	r	r	rw	r	r	rw	rw	r

Bit	Function
RSM	**Resume Bus Activity** When the USB device is in suspend mode, setting bit RSM resumes bus activity. In response to this action, the USB will deassert the suspend bit and perform the remote wake-up operation. Writing 0 to RSM has no effect, the bit is reset if bit SUSP is 0.
DINIT	**Device Initialization in Progress** At the end of a software reset, bit DINIT is set by hardware. After software reset of the USB module, it must be initialized by the CPU. When DINIT is set after a software reset, 6 bytes for each endpoint must be written to SFR USBVAL0. After the 96th byte, bit DONE0 has to be set by software. Bit DINIT is reset by software after a successful initialization sequence.
SUSP	**Suspend Mode** This bit is set when the USB is idle for more than 3 ms. It will remain set until there is a non idle state on the USB cable or when bit RSM is set. In addition to SUSP an interrupt can be generated when the suspend mode begins (DIRR.SBI) and when it ends (DIRR.SEI)
SWR	**Software Reset** Setting bit SWR initiates a software reset operation of the USB device. This bit is cleared by hardware after a successful reset operation. SWR can not be reset by software.
UCLK	**USB Clock Enable** Setting bit UCLK to "0" disables the 48 MHz USB clock signal (see **Figure 4-1**)

*Note: A software reset initiated by setting bit SWR to ’1’, which should be done **4 clock cycles** after UCLK has been set to ’1’, clears bit UCLK. This means UCLK has to be set to ’1’ again. Note that this reinforcement of DCR.UCLK has to be programmed **20 clock cycles** after SWR has been set.*

22.11.7 Device Power Down Register (DPWDR)

The device power down register (DPWDR) includes 2 bits which allow to switch off the USB transmitter and receiver circuitry selectively for power down mode operation.

Reset value: 00_H Address: $0A_H$

	7	6	5	4	3	2	1	0
$0A_H$	0	0	0	0	0	0	TPWD	RPWD
	r	r	r	r	r	r	rw	rw

Bit	Function
TPWD	**USB Transmitter Power Down** Setting bit TPWD to 1 puts the USB transmitter into power down mode. After a wake-up from software power down mode, bit TPWD must be cleared by software to enable data transmission again.
RPWD	**USB Receiver Power Down** *Note: Setting bit RPWD puts the USB receiver into power down mode. After a wake-up from software power down mode, bit RPWD must be cleared by software to enable data reception again. Note: If RPWD is set, the USB bus can not wake-up the device from power down mode.*

22.11.8 USB Device Interrupt Enable Register (DIER)

The device interrupt enable register DIER contains the enable bits for the different USB device interrupts. A device interrupt can only be generated if EUDI and EAL are set, too.

Reset value: 00_H Address: $0B_H$

	7	6	5	4	3	2	1	0
$0B_H$	SE0IE	DAIE	0	SBIE	SEIE	STIE	SUIE	SOFIE
	rw	rw	r	rw	rw	rw	rw	rw

The interrupts contained in the DIRR register can individually be masked in the DIER register.

Bit	Function
SE0IE	**Single Ended Zero Interrupt Enable**
DAIE	**Device Attached Interrupt Enable**
SBIE	**Suspend Begin Interrupt Enable**
SEIE	**Suspend End Interrupt Enable**
STIE	**Status Interrupt Enable**
SUIE	**Setup Interrupt Enable**
SOFIE	**Start of Frame Interrupt Enable**

22.11.9 USB Device Interrupt Request Register (DIRR)

Reset value: 00_H Address: $0C_H$

	7	6	5	4	3	2	1	0
$0C_H$	SE0I	DAI	0	SBI	SEI	STI	SUI	SOFI
	r	r	r	r	r	r	r	r

The USB device interrupt request register contains the device specific interrupt flags of the USB module. These flags are set after the occurrence of special events. If a request flag is set, it is automatically cleared after a read operation of the DIRR register.

The interrupts contained in the DIRR register can individually be masked in the DIER register.

Bit	Function
SE0I	**Single Ended Zero Interrupt** SE0I is set each time a single ended zero is detected for equal or greater than 2.5 µs. EOP (2 bit times) is not detected.
DAI	**Device Attached Interrupt** Bit DAI is automatically set after detection of the USB device being attached to the USB bus
SBI	**Suspend Begin Interrupt** SBI is automatically set when the suspend mode is entered.
SEI	**Suspend End Interrupt** SEI is automatically set when the suspend mode is left.
STI	**Status Interrupt** STI is set if the host requires a status transfer and the device answers with NACK (if bit ESP is set, the device answers with ACK and then STI is not set).

SUI	**Setup Interrupt** SUI is automatically set after a successful reception of a setup packet which is not handled by the USB module and must be forwarded to the CPU. The setup packet itself is limited to 8 bytes and stored at USB memory addresses 00H to 07H. If a setup interrupt occurs, the control and status bits UBF0, CBF0, SOD0 and DIR0 in the endpoint registers EPBS0 and EPIR0 are cleared. DIR0=0 predicts the direction of the next USB access (data phase) to be from host to CPU. If a read access is indicated in the SETUP packet, the CPU has to set bit SETRDn (set read), before data can be written to the corresponding buffer.
SOFI	**Start of Frame Interrupt** SOF is automatically set after detection of a start of frame packet on the USB.

22.11.10 Device Setup and Status Register (DSSR)

Reset value: 00_H Address: $0D_H$

	7	6	5	4	3	2	1	0
$0D_H$	0	0	0	0	0	0	IE	IR
	r	r	r	r	r	r	rw	r

.

Bit	Function
IR	**Interrupt Request** Interrupt request (IR=1) for all Standard Commands and "No error" indication
IE	**Interrupt Enable** Interrupt Enable(IE=1) for all Standard Commands and "No error" indication

22.11.11 Frame Number Register (FNR)

The frame number registers store an 11-bit value which defines the number of a USB frame. The frame number rolls over upon reaching its maximum value of 7FFH. The FNRH/FNRL registers are read only registers which are reset to 00H by a hardware reset.

Reset value: 0000 0XXX XXXX XXXX$_B$ Address: 0E

	7	6	5	4	3	2	1	0
0E$_H$	FNR7	FNR6	FNR5	FNR4	FNR3	FNR2	FNR1	FNR0
	r	r	r	r	r	r	r	r

	15	14	13	12	11	10	9	8
0F$_H$	0	0	0	0	0	FNR10	FNR9	FNR8
	r	r	r	r	r	r	r	r

Bit	Function
FNR10 - FNR0	**Frame Number Value** FNRH.2 - FNRH.0 and FNRL.7 - FNRL.0 hold the current 11-bit frame number of the latest SOF token.

22.11.12 Device Get_Status Register (DGSR)

These two registers hold a 16-bit value that is sent to the USB host upon a "Device Get Status" command.

Reset value: 0000_H Address: 10_H

	7	6	5	4	3	2	1	0
10_H	DST7	DST6	DST5	DST4	DST3	DST2	RWUP	PSTAT
	rw	rw	rw	rw	rw	rw	r	rw
	15	14	13	12	11	10	9	8
11_H	DST15	DST14	DST13	DST12	DST11	DST10	DST9	DST8
	rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
RWUP	Remote Wakeup (see also CIARIE.DRW on page 562) This bit indicates the status of the remote wakeup The remote wakeup status is configured by the host using the Set_Feature_Remote_Wakeup and the status is returned to the host with every Get_Status request. 0: remote wakeup disabled 1: remote wakeup enabled
PSTAT	USB Power Status This bit indicates to the host whether the USB device is operating in bus-powered mode ("0") or in self-powered mode ("1"). This bit has no effect on the functions of the device.

The function of this register is completely determined by the USB specification. .

22.11.13 Interface Get_Status Register (IGSR)

These two registers hold a 16-bit value that is sent to the USB host upon an "Interface Get Status" command.

Reset value: 0000_B Address: 12_H

	7	6	5	4	3	2	1	0
12_H	IST7	IST6	IST5	IST4	IST3	IST2	IST1	IST0
	rw	rw	rw	rw	rw	rw	rw	rw
	15	14	13	12	11	10	9	8
13_H	IST15	IST14	IST13	IST12	IST11	IST10	IST9	IST8
	rw	rw	rw	rw	rw	rw	rw	rw

The function of this register is completely determined by the USB specification. For further functional additions the bits of this register (IST15-0) can be programmed to "1" to fulfill future requirements.

22.11.14 Interface Select Registers (IFCSEL)

Reset value: 00_H Address: 14_H

	7	6	5	4	3	2	1	0
14_H	0	0	0	0	IF3	IF2	IF1	IF0
	r	r	r	r	r	r	r	r

Bit	Function
IF3-0	**Inteface Select Bits** These four bits are used to select the interface number (interface 0 - 14) for setting the corresponding 16-bit value in IGSR.

22.11.15 Enable Setup Token(ESTR)

Reset value: 00_H Address: 15_H

	7	6	5	4	3	2	1	0
15_H	STRE	0	0	0	0	0	0	0
	rw	r	r	r	r	r	r	r

Bit	Function
STRE	**Enable Setup Token** If set, setup token can be read by firmware and the current endpoint is set to EP0. Setting or clearing bit STRE resets the ADROFF0 register.

22.11.16 Active Endpoint Number (ACTEP)

Reset value: 00_H Address: 16_H

	7	6	5	4	3	2	1	0
16_H	0	0	0	0	EPN			
	r	r	r	r	r			

Bit	Function
EPN	**Endpoint number** Current CPU endpoint number

22.12 Detailed Register Description - USB Endpoint Registers

All registers described in this chapter are available 16-times, i.e. one register set per endpoint (n = 0 - 15). Therefore the register addresses depend on the specific endpoint which can be seen in **chapter 22.10.1**.

22.12.1 Endpoint Get_Status Register (EGSRn)

These two registers hold a 16-bit value that is sent to the USB host upon an "Endpoint Get Status" command.

An EGSR register is available for each of the 16 endpoints (n=0-15).

Reset value: 0000_H

7	6	5	4	3	2	1	0
EST7	EST6	EST5	EST4	EST3	EST2	EST1	STALL
rw	rw	rw	rw	rw	rw	rw	r

15	14	13	12	11	10	9	8
EST15	EST14	EST13	EST12	EST11	EST10	EST9	EST8
rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
STALL	**Endpoint Stalled (Set/Cleared by the USB host)** The endpoint stall status can be read by the µC in the GESR register and the status is returned with every Get_Status request for Endpoint. 0: endpoint is not stalled 1: endpoint is stalled This bit cannot be written by the µC.

The function of this register is completely determined by the USB specification. For further functional additions the higher bits of this register (EST15-1) can be programmed to "1" to fulfill future requirements.

22.12.2 Endpoint Interrupt Enable Register (EPIEn)

Reset value: 00_H

7	6	5	4	3	2	1	0
AIEn	NAIEn	RLEIEn	0	DNRIEn	NODIEn	EODIEn	SODIEn
rw	rw	rw	r	rw	rw	rw	rw

The endpoint interrupt enable registers contain the endpoint specific interrupt enable bits. With these bits, the endpoint specific interrupts can be individually enabled or disabled. In addition to a bit in an EPIEn register the following bits have to be set to get an interrupt active:

In case of endpoint 0 or endpoints 5-15, the global interrupt bit EPIn in GEPIR for and the interrupt enable flag of the corresponding interrupt node USBEPIE have to be set.

Because endpoints 1-4 are handled by separate interrupt nodes, the corresponding enable flags have to be set. (see **Table 8-1**)

An EPIE register is available for each of the 16 endpoints (n=0-15).

Bit	Function
AIEn	**Acknowledge Interrupt Enable** Bit AIEn enables the generation of an endpoint specific acknowledge interrupt when bit ACKn in register EPIRn is set.
NAIEn	**Not Acknowledged Interrupt Enable** Bit NAIEn enables the generation of an endpoint specific not acknowledged interrupt when bit NACKn in register EPIRn is set.
RLEIEn	**Read Length Error Interrupt Enable** Bit RLEIEn enables the generation of an endpoint specific read length error interrupt when bit RLEn in register EPIRn is set.
EPIEn.4	Not implemented. Reserved for future use.
DNRIEn	**Data Not Ready Interrupt Enable** Bit DNRIEn enables the generation of an endpoint specific data not ready interrupt when bit DNRn in register EPIRn is set.
NODIEn	**No Data Interrupt Enable** Bit NODIEn enables the generation of an endpoint specific no data interrupt when bit NODn in register EPIRn is set.
EODIEn	**End of Data Interrupt Enable** Bit EODIEn enables the generation of an endpoint specific end of data interrupt when bit EODn in register EPIRn is set.

SODIEn	**Start of Data Interrupt Enable** Bit SODIEn enables the generation of an endpoint specific start of data interrupt when bit SODn in register EPIRn is set.

22.12.3 Endpoint Interrupt Request Register (EPIRn)

Reset value EPIR0: 11_H, Reset value EPIR1-EPIR15: 10_H

7	6	5	4	3	2	1	0
ACKn	NACKn	RLEn	0	DNRn	NODn	EODn	SODn
r	r	r	r	r	r	r	r

The interrupts contained in the EPIRn register can individually be masked in the EPIEn register.

An EPIR register is available for each of the 16 endpoints (n=0-15).

Bit	Function
ACKn	**USB Acknowledge** Bit ACKn=1 indicates a successful action on the USB.
NACKn	**USB Not Acknowledge** Bit NACK is set for all unsuccessful actions on the USB.
RLEn	**Read Length Error** Bit RLEn is automatically set if the number of bytes read by the USB does not correspond to the packet length programmed by the CPU. A mismatch in packet size will not cause a data packet to be discarded. The RLEn bit gets automatically cleared if there is a USB reset (RESUBF = 1) or if Interrupt flags are cleared upon read of EPIRn.
EPIRn.4	Not implemented. Reserved for future use.
DNRn	**Data Not Ready** This bit is set by hardware if the USB module requires an access to USB memory, but no buffer is available. USB Read Action: DNRn is set if UBF is not set. USB Write Action: DNRn is set if UBF is set.
NODn	**No Data** This bit indicates an incorrect CPU read or write access to USB memory. It is set if the CPU processes a read access to an empty USB buffer or a write access to a full buffer. NODn is also set if the direction is write (DIRn=0 for USB write access) and the CPU tries to write to the USB memory buffer.

EODn	**End of Data** During a **USB read** access EODn is set if the CPU has written a programmable number (MaxLen) of bytes in the transmit buffer. As a result, the buffer is full and no more write actions from the CPU are allowed. During a **USB write** access EODn is set if the CPU has read a programmable number (USBLen) of bytes out of the receive buffer. As a result, the buffer is empty now and no more read actions from the CPU are allowed.
SODn	**Start of Data** During a **USB read** access SODn is set if the USB has read a fixed number (USBLen) of bytes from the transmit buffer. As a result, the buffer is now empty and the CPU can process write actions again. During a **USB write** access SODn is set if the USB has written a fixed number (USBLen) of bytes to the receive buffer. As a result, the buffer is full and the CPU can start read actions. The **SOD0** bit is set at the end of a status-in transfer and cleared with the next SETUP token. *Note: The SODn interrupt is not set during a status phase.*

In dual buffer mode, bits SODn and EODn can be set simultaneously if the corresponding buffer page is swapped.

If a request flag in EPIRn is set, it is automatically cleared after a read operation of the EPIRn register.

22.12.4 Endpoint Buffer Control Register (EPBCn)

The endpoint buffer control register controls the endpoint specific operations.

An EPBC register is available for each of the 16 endpoints (n=0-15).

Reset value: 00_H

7	6	5	4	3	2	1	0
STALLn	0	0	GEPIEn	SOFDEn	INCEn	0	DBMn
rw	r	r	rw	rw	rw	r	rw

Bit	Function
STALLn	**Endpoint Stall** Bit STALL can be set to indicate that the endpoint is stalled. If the stall bit for endpoint 0 (STALL0) is set, the next incoming setup token will automatically clear it. If STALL=0, the endpoint n is active If STALL=1, the endpoint n is stalled
GEPIEn	**Global Endpoint Interrupt Enable** Bit GEPIEn enables or disables the generation of the global endpoint interrupt indicated in register GEPIR for endpoint 0 and endpoints 5-15 GEPIE=0, the USB endpoint n interrupt is disabled. If GEPIE=1, the USB endpoint n interrupt is enabled
SOFDEn	**Start of Frame Done Enable** If bit SOFDE is set, the current CPU buffer[1)] in USB memory is automatically tagged full (data flow from the CPU to USB) or empty (data flow from USB to the CPU) on each detection of a start of frame on the USB (auto-done). If SOFDE=0, no action takes place on SOF If SOFDE=1, automatic generation of DONE on SOF is enabled
INCEn	**Auto Increment Enable** If bit INCE is set, the address offset register ADROFFn for CPU access to USB memory is automatically incremented after each data write or data read action of the USBVALn register. This allows the user to handle the USB memory like a FIFO without modification of the address of the desired memory location by software.

DBMn	**Dual Buffer Mode** Bit DBM allows the selection between single buffer mode and dual buffer mode. If DBM=0, single buffer mode is selected If DBM=1, dual buffer mode is selected

1) This means that the auto-swap function is limited to the last accessed endpoint buffer

22.12.5 Endpoint Buffer Status Register (EPBSn)

The bits of the endpoint buffer status registers indicate the status of the endpoint specific USB memory buffers and allows setting of certain USB memory buffer conditions.

An EPBS register is available for each of the 16 endpoints (n=0-15).

Reset value: 20_H

7	6	5	4	3	2	1	0
UBFn	CBFn	DIRn	ESP0[1)]	SETRDn	SETWRn	CLREPn	DONEn
r	r	r	w	w	w	w	w

1) This bit is only defined for endpoint 0, ie. for register EPBS0

Bit	Function
UBFn	**USB Buffer Full** Bit UBFn indicates the status of the USB memory buffer for endpoint n. USB read access: If UBFn=0, the USB buffer for endpoint n is empty. If UBFn=1, the USB buffer for endpoint n is not empty. USB write access: If UBFn=0, the USB buffer for endpoint n is not full. If UBFn=1, the USB buffer for endpoint n is full.
CBFn	**CPU Buffer Full** Bit CBFn indicates the status of the CPU memory buffer for endpoint n. CPU read access: If CBFn=0, the CPU buffer for endpoint n is empty. If CBFn=1, the CPU buffer for endpoint n is not empty. CPU write access: If CBFn=0, the CPU buffer for endpoint n is not full. If CBFn=1, the CPU buffer for endpoint n is full.
DIRn	**Direction of USB Memory Access** Bit DIRn indicates the direction of the last USB memory access for endpoint n. If DIRn=0, the last data flow for endpoint n was from host to CPU If DIRn=1, the last data flow for endpoint n was from CPU to host

ESP0	**Enable Status Phase (for Control Transfer)** If bit ESP0 is set, the next status phase of endpoint n will automatically be acknowledged (by an ACK or by a zero data packet, respectively) except if the endpoint n is stalled. If ESP0 is not set, a status interrupt request (DIRR.STI) is generated and the µC has to set ESP0 to acknowledge the status phase. • Data transfer Host => Device: After the host has transmitted the last valid data packet, the UDC will automatically transmit a zero data packet to the host if ESP0is set. • Data transfer Device => Host: After the UDC has transmitted the last valid data packet and the host has responded with a zero data packet, the UDC will automatically transmit an ACK to the host if ESP0 is set. If the status phase is successfully completed, bit ESP0 is automatically reset by the UDC. Additionally, the ESP0 bit is cleared with every setup frame. If the CPU detects a corrupted control transfer, bit STALL0 should be set by software instead of bit ESP0 in order to indicate an error condition from which the USB device can not recover by itself. The SOD0 bit is set at the end of a status-in transfer.
SETRDn	**Set Direction of USB Memory Buffer to Read** Bit SETRDn is used to predict the direction of the next USB access for endpoint n as a USB read access. A faulty prediction causes no errors since the USB module determines the real direction. A change in the data direction is only executed if both USB memory buffers are empty. SETRDn can not be set together with CLREPn because a change of bit DIRn during a transfer is not allowed. *Note: bits SETRDn and SETWRn must not be set at the same time.*
SETWRn	**Set Direction of USB Memory Buffer to Write** Bit SETWRn is used to predict the direction of the next USB access for endpoint n as a USB write access. A faulty prediction causes no errors since the USB module determines the real direction. A change in the data direction is only executed if both USB memory buffers are empty. SETWR can not be set together with CLREPn because a change of EPBSn.DIRn during a transfer is not allowed. *Note: bits SETWRn and SETRDn must not be set at the same time.*

CLREPn	**Clear Endpoint** Setting bit CLREPn will set the address offset register for a CPU access to USB memory to 0. The bits CBFn and UBFn will be reset when CLREPn is set. Bit CLREPn is reset by hardware. A read operation of this bit will always deliver 0. Setting of bit CLREPn does not change the direction of endpoint n. This means, bit DIRn is not changed. *Note: Note: When bits CLREP0 and ESP0 are set simultaneously with one instruction, bit ESP0 remains set and the next status phase is enabled. If only CLREP0 is set, bit ESP0 is reset and the status phase is disabled.* *Note: Setting bits CLREPn and SETRDn or SETWRn simultaneously with one instruction is not allowed. This means that the setting of SETRDn or SETWRn is ignored.*
DONEn	**Buffer Done by CPU** If bit DONE is set, the current USB memory buffer assigned to CPU is automatically tagged full (data flow from the CPU to USB) or empty (data flow from USB to the CPU). This bit is reset by hardware after it has been set. A read operation of this bit always delivers a 0. *Note: If the direction of the endpoint is read (USB read access) and auto-increment is enabled (INCEn=1) and DONEn is set, the content of register ADROFFn is copied automatically to register EPLENn of the actual endpoint. Register EPLENn is not changed if the auto-increment capability is disabled (INCEn=0).*

22.12.6 Endpoint Base Address Register (EPBAn)

The endpoint base address and length registers define the location and size (start address and length) of the endpoint specific buffers in the USB memory.

An EPBA register is available for each of the 16 endpoints (n=0-15).

Reset value: 00_H

7	6	5	4	3	2	1	0
PAGEn	An7	An6	An5	An4	An3	0	0
r	rw	rw	rw	rw	rw	r	r

Bit	Function
PAGEn	**Buffer Page for endpoint n (single buffer mode only)** In single buffer mode, the endpoint n can be either located on USB memory buffer page 0 (PAGEn=0) or on USB memory buffer page 1 (PAGEn=1) by clearing or setting this bit. In dual buffer mode this bit has no effect. *Note: The SETUP token is always stored on USB memory buffer page 0 at address 00H to 07H.*
An7-An3	**Endpoint n Buffer Start Address** The bits2-6 of EPBAn represent address bits 3-7 of the USB memory buffer start address for endpoint n. Bits 2-0 of the memory buffer start address are set to 0.

22.12.7 Endpoint Buffer Length Register (EPLENn)

The endpoint base address and length registers define the location and size (start address and length) of the endpoint specific buffers in the USB memory.

An EPLEN register is available for each of the 16 endpoints (n=0-15).

Reset value: 0XXXXXXX

7	6	5	4	3	2	1	0
0	Ln6	Ln5	Ln4	Ln3	Ln2	Ln1	Ln0
r	rw	rw	rw	rw	rw	rw	rw

Bit	Function
Ln6 - Ln0	**Endpoint n Buffer Length** The bits 0 to 6 of EPLENn define the length of the USB memory buffer for endpoint n and can not be written if DINIT=1.

22.12.8 Address Offset Register (ADROFFn)

An ADROFF register is available for each of the 16 endpoints (n=0-15).

Reset value: 00_H

7	6	5	4	3	2	1	0
0	0	AO5	AO4	AO3	AO2	AO1	AO0
r	r	rw	rw	rw	rw	rw	rw

Bit	Function
AO5 - AO0	**USB Address Offset** ADROFFn stores the 6-bit offset address for USB memory buffer addressing by the µC.

As the USB memory size is 256 bytes per page, the maximum packet length is limited to 64 bytes. Therefore, only the lowest 6 bits of ADROFFn (AO5...AO0) are required for offset definition. A write operation to ADROFFn is only successful if either EPBSn.DIRn=0 and CBF=1 (USB write operation) or EPBSn.DIRn=1 and CBF=0 (USB read operation).

22.12.9 USB Data Register (USBVALn)

An USBVAL register is available for each of the 16 endpoints (n=0-15).

Reset value: 00_H

7	6	5	4	3	2	1	0
USBVALn.7	USBVALn.6	USBVALn.5	USBVALn.4	USBVALn.3	USBVALn.2	USBVALn.1	USBVALn.0
rw	rw	rw	rw	rw	rw	rw	rw

Bit	Function
USBVALn.7 -USBVALn.0	**USB Data Value** USBVALn stores the 8-bit data byte during transfers from µC to USB memory and from USB memory to the µC. Bit NOD in the EPIRn register indicates when the µC processes a USBVALn read operation with an empty USB buffer or a USBVALn write operation to a full USB buffer.

The data transfers between USB memory and the µC are handled via the SFR USBVALn. With a µC write access to USBVALn, the value written into it is transferred to the USB memory location defined by the content of the endpoint specific base address register EPBAn and the address offset register ADROFFn. During USB memory read accesses by the µC, data is written in the reverse direction. Access to USBVALn is only successful if either EPBSn.DIRn=0 and CBF=1 (USB write operation) or EPBSn.DIRn=1 and CBF=0 (USB read operation).

23 Power Reduction Modes

Three different power reduction modes with different levels of power reduction have been implemented in the INCA-D.

In **Idle mode** the CPU is stopped, while the peripherals continue their operation, i.e. the clock of the XBUS is still running.

Idle mode can be terminated by any reset or interrupt request.

In **Power Down mode** both the CPU and the peripherals are stopped, because the XBUS clock gets killed.

Power Down mode can only be terminated by a hardware reset.

The **Sleep mode** is similar to Power Down mode, but the power consumption is further reduced, because the oscillator is switched off additionally.

An external interrupt at FEXT1 (P2.1) or an interrupt caused by the DASL line (Level Detect) wakes up the device from sleep mode.

Note: All external bus actions are completed before Idle, Sleep or Power Down mode of the CPU is entered.

23.1 Sleep Mode of the CPU

The Sleep mode is the most effective power reduction mode.

To enter sleep mode bit OSC_DWN has to be set to '1', while pin FEXT1 is configured as input and driven 'high'

After an interrupt at FEXT1 or LD (ie. some activity on the Line), the the osc wakes up again. To enable the wakeup from the line, bit EA_FIR has to be set to 1.

SLEEPCON (DF3A$_H$) **XBUS-SFR-b** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	-	-	-	-	EA-FIR	OSC_ DWN
-	-	-	-	-	-	-	-	-	-	-	-	-	-	rw	-

General description of SLEEPCON bits:

Bit	Function
EAFIR	**Enable Automatic Fast Interrupt** 0: Automativ wake-up due to LD disabled 1: Automativ wake-up due to LD enabled
OSC_DWN	**Oscillator Power Down** 0: Oscillator on 1: Oscillator off

For wake-up an internal logic uses the **rising edge** of interrupt node 13, which should represent **the Level Detect interrupt** of ithe transceiver.

This means for a proper wake-up from the line the interrupt status register IRQ13_STA carries the LD (Level Detect) interrupt of the line transceiver only, ie. all other interrupts have to be masked. (see **Figure 8-3**).

23.2 Idle Mode of the CPU

The power consumption of the INCA-D microcontroller can be decreased by entering Idle mode. In this mode all peripherals, **including** the watchdog timer, continue to operate normally, only the CPU operation is halted.

Idle mode is entered after the IDLE instruction has been executed and the instruction before the IDLE instruction has been completed. To prevent unintentional entry into Idle mode, the IDLE instruction has been implemented as a protected 32-bit instruction.

Idle mode is terminated by interrupt requests from any enabled interrupt source whose individual Interrupt Enable flag was set before the Idle mode was entered, regardless of bit IEN.

For a request selected for CPU interrupt service the associated interrupt service routine is entered if the priority level of the requesting source is higher than the current CPU priority and the interrupt system is globally enabled. After the RETI (Return from Interrupt) instruction of the interrupt service routine is executed the CPU continues executing the program with the instruction following the IDLE instruction. Otherwise, if the interrupt request cannot be serviced because of a too low priority or a globally disabled interrupt system the CPU immediately resumes normal program execution with the instruction following the IDLE instruction.

For a request which was programmed for PEC service a PEC data transfer is performed if the priority level of this request is higher than the current CPU priority and the interrupt system is globally enabled. After the PEC data transfer has been completed the CPU remains in Idle mode. Otherwise, if the PEC request cannot be serviced because of a too low priority or a globally disabled interrupt system the CPU does not remain in Idle mode but continues program execution with the instruction following the IDLE instruction.

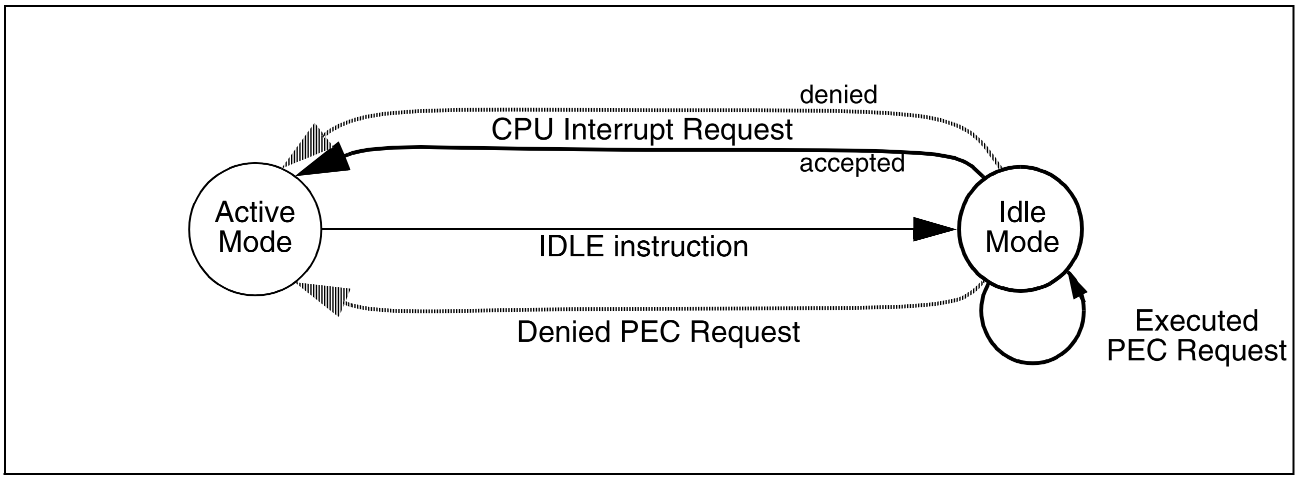


Figure 23-1 Transitions between Idle mode and active mode

Idle mode can also be terminated by a Non-Maskable Interrupt, ie. a high to low transition on the $\overline{\text{NMI}}$ pin. After Idle mode has been terminated by an interrupt or NMI request, the interrupt system performs a round of prioritization to determine the highest priority request. In the case of an NMI request, the NMI trap will always be entered.

Any interrupt request whose individual Interrupt Enable flag was set before Idle mode was entered will terminate Idle mode regardless of the current CPU priority. The CPU will **not** go back into Idle mode when a CPU interrupt request is detected, even when the interrupt was not serviced because of a higher CPU priority or a globally disabled interrupt system (IEN='0'). The CPU will **only** go back into Idle mode when the interrupt system is globally enabled (IEN='1') **and** a PEC service on a priority level higher than the current CPU level is requested and executed.

Note: An interrupt request which is individually enabled and assigned to priority level 0 will terminate Idle mode. The associated interrupt vector will not be accessed, however.

The watchdog timer may be used to monitor the Idle mode: an internal reset will be generated if no interrupt or NMI request occurs before the watchdog timer overflows. To prevent the watchdog timer from overflowing during Idle mode it must be programmed to a reasonable time interval before Idle mode is entered.

23.3 Power Down Mode of the CPU

To further reduce the power consumption the microcontroller can be switched to Power Down mode. Clocking of all internal blocks is stopped, the contents of the internal RAM, however, are preserved through the voltage supplied via the V_{DD} pins. The watchdog timer is stopped in Power Down mode. This mode can only be terminated by an external hardware reset, ie. by asserting a low level on the $\overline{\text{RSTIN}}$ pin. This reset will initialize all

SFRs and ports to their default state, but will not change the contents of the internal RAM.

There are two levels of protection against unintentionally entering Power Down mode. First, the PWRDN (Power Down) instruction which is used to enter this mode has been implemented as a protected 32-bit instruction. Second, this instruction is effective **only** if the $\overline{\text{NMI}}$ (Non Maskable Interrupt) pin is externally pulled low while the PWRDN instruction is executed. The microcontroller will enter Power Down mode after the PWRDN instruction has completed.

This feature can be used in conjunction with an external power failure signal which pulls the $\overline{\text{NMI}}$ pin low when a power failure is imminent. The microcontroller will enter the NMI trap routine which can save the internal state into RAM. After the internal state has been saved, the trap routine may set a flag or write a certain bit pattern into specific RAM locations, and then execute the PWRDN instruction. If the $\overline{\text{NMI}}$ pin is still low at this time, Power Down mode will be entered, otherwise program execution continues. During power down the voltage at the V_{DD} pins can be lowered to 2.5 V while the contents of the internal RAM will still be preserved.

The initialization routine (executed upon reset) can check the identification flag or bit pattern within RAM to determine whether the controller was initially switched on, or whether it was properly restarted from Power Down mode.

23.4 Status of Output Pins during Idle and Power Down Mode

During Idle mode the CPU clocks are turned off, while all peripherals continue their operation in the normal way. Therefore all ports pins, which are configured as general purpose output pins, output the last data value which was written to their port output latches. If the alternate output function of a port pin is used by a peripheral, the state of the pin is determined by the operation of the peripheral.

Port pins which are used for bus control functions go into that state which represents the inactive state of the respective function (eg. $\overline{\text{WR}}$), or to a defined state which is based on the last bus access (eg. $\overline{\text{BHE}}$). Port pins which are used as external address/data bus hold the address/data which was output during the last external memory access before entry into Idle mode under the following conditions:

P0H outputs the high byte of the last address if a multiplexed bus mode with 8-bit data bus is used, otherwise P0H is floating. P0L is always floating in Idle mode.

PORT1 outputs the lower 16 bits of the last address if a demultiplexed bus mode is used, otherwise the output pins of PORT1 represent the port latch data.

Port 4 outputs the segment address for the last access on those pins that were selected during reset, otherwise the output pins of Port 4 represent the port latch data.

During Power Down and Sleep Mode the oscillator and the clocks to the CPU and to the peripherals are turned off. Like in Idle mode, all port pins which are configured as general purpose output pins output the last data value which was written to their port output latches.

When the alternate output function of a port pin is used by a peripheral the state of this pin is determined by the last action of the peripheral before the clocks were switched off.

The table below summarizes the state of all INCA-D output pins during Idle and Power Down mode.

INCA-D Output Pin(s)	**Idle Mode**		**Power Down Mode**	
	No external bus	**External bus enabled**	**No external bus**	**External bus enabled**
ALE	Low	Low	Low	Low
$\overline{\text{RD}}$, $\overline{\text{WR}}$	High	High	High	High
CLKOUT	Active	Active	High	High
$\overline{\text{RSTOUT}}$	1)	1)	1)	1)
P0L	Port Latch Data	Floating	Port Latch Data	Floating
P0H	Port Latch Data	A15...A8 2) / Float	Port Latch Data	A15...A8 2) / Float
PORT1	Port Latch Data	Last Address 3) / Port Latch Data	Port Latch Data	Last Address 3) / Port Latch Data
Port 4	Port Latch Data	Port Latch Data/ Last segment	Port Latch Data	Port Latch Data/ Last segment
$\overline{\text{BHE}}$	Port Latch Data	Last value	Port Latch Data	Last value
$\overline{\text{CSx}}$	Port Latch Data	Last value 4)	Port Latch Data	Last value 4)
Other Port Output Pins	Port Latch Data / Alternate Function	Port Latch Data / Alternate Function	Port Latch Data / Alternate Function	Port Latch Data / Alternate Function

1): High if EINIT was executed before entering Idle or Power Down mode, Low otherwise.

2): For multiplexed buses with 8-bit data bus.

3): For demultiplexed buses.

4): The $\overline{\text{CS}}$ signal that corresponds to the last address remains active (low), all other enabled $\overline{\text{CS}}$ signals remain inactive (high). By accessing an on-chip X-Periperal prior to entering a power save mode all external $\overline{\text{CS}}$ signals can be deactivated.

23.5 Power Down Mode of Peripherals

Additionally to idle mode, sleep mode and power down mode of the CPU, each peripheral can get its clock disabled. This has two advantages. First this feature can be used in order to save power. Second, in case of a break from the on-chip debug support, it can be decided whether a peripheral stops or keeps on running.

For this purpose, the following peripherals which are connected to the PD-bus or to the XBUS (refer to **Figure 3-1**), ie. SSC0, SSC1, ASC , GPT12 and PIDD have a clock control register.

Note: The two clocks of the USB module can be disabled by setting CLK_CONF.USB_DIS to '1' and DCR_UCLK to '0'. For further details refer to Chapter 22.6

The clock control registers are called *mod*CLC where *mod* is substituted by the corresponding peripheral name, e.g. SSC, ASC or GPT12. With this registers, two modes can be selected for each peripheral:

Run mode

The peripheral works as expected.

Clock Disable Mode / OCDS Suspend Mode

In case the clock of the peripheral shall be disabled, any pending read and write access is finished before the actually peripherla gets disabled. then, no write access is possible as long as clock is disabled. A read to any register returns the register value. However, each read during disabled clock is non-destructive, i.e., no action is issued upon a read.

The peripheral SSC may serve as an example for this. After transmission, the receive buffer must be read to avoid getting a receive error with the next transmission. If however the SSC clock is disabled after a transmission, and the receive buffer is read after having disabled the ssc clock, then the peripheral does not notice this read and issues an receive error with the next transmission.

The addresses of the *mod*CLC registers are listed in the dedicated peripheral chapter.

Each clock control register consists of three bits. Two of these bits determine if the clock is disabled. The other bit indicates whether the peripheral clock is disabled or not. According to the peripherla, the base address refers to the PD-bus address space or the XBUS address space.

***mod* CLC (MODBASE+00$_H$)** **Reset Value: 0000$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
reserved													***mod* SPEN**	***mod* DISS**	***mod* DISR**
													rw	r	rw

Bit	Function
mod DISR	**Module Disable Request Bit** *mod* DISR = ‘0’: Module disable not requested *mod* DISR = ‘1’: Module disable requested
mod DISS	**Module Disable Status Bit** *mod* DISS = ‘0’: Module enabled *mod* DISS = ‘1’: Module disabled
mod SPEN	**Module Suspend Enable Bit for OCDS** *mod* SPEN = ‘0’: Module suspend disabled *mod* SPEN = ‘1’: Module suspend enabled

The functions of the bits of the clock control register are illustrated in figure 23-2. Note that the input line called ocds_p_suspend comes from the OCDS and is 1 if a break is active. Thus, with bit *mod* SPEN, it can be selected whether the peripheral *mod* gets stopped on a break or not.

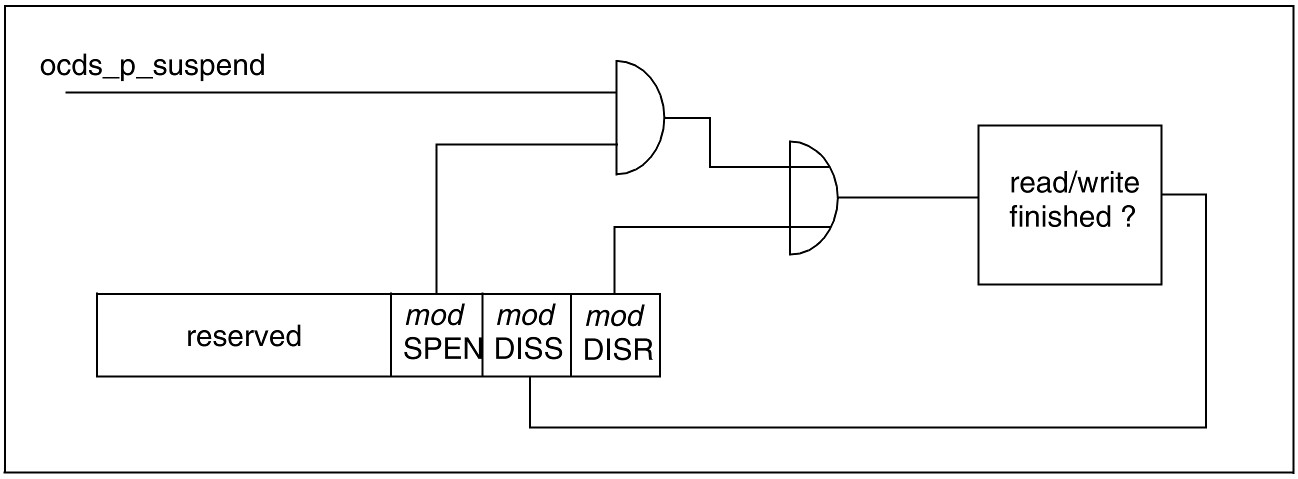


Figure 23-2 Peripherals´ Clock Control Block

23.6 Power Down Mode of the DSP

The DSP can be brought to power down mode by setting the power down bit (PD).

Table 23-1 Power Down Bit

Register	# of Bits	Name	Comment
CCTL	1	PD	power down mode

In power down mode the DSP and its peripherals are stopped. The DSP enters active mode again upon an access to a read/write register.

24 System Reset

The internal system reset function provides initialization of the INCA-D into a defined default state and is invoked either by asserting a hardware reset signal on pin $\overline{\text{RSTIN}}$ (Hardware Reset Input), upon the execution of the SRST instruction (Software Reset) or by an overflow of the watchdog timer.
The reset will deactivate the DASL-transceiver and it will abort any TIC-bus access currently in progress. The TIC-bus returns to idle.

If enabled an watchdog timeout can generate a reset on pin $\overline{\text{RSTOUT}}$. A hardware reset always generates a reset on pin $\overline{\text{RSTOUT}}$.

Whenever one of the listed conditions occurs, the INCA-D is reset into its predefined default state through an internal reset procedure. When a reset is initiated, pending internal hold states are cancelled and the current internal access cycle (if any) is completed. An external bus cycle is aborted, except for a watchdog reset (see description). After that the bus pin drivers and the I/O pin drivers are switched off (tristate).

The internal reset procedure requires 1024 CPU clock cycles in order to perform a complete reset sequence. This reset sequence is started upon a watchdog timer overflow, a SRST instruction or when the reset input signal $\overline{\text{RSTIN}}$ is latched low (hardware reset). The internal reset condition is active at least for the duration of the reset sequence and then until the $\overline{\text{RSTIN}}$ input is inactive. When this internal reset condition is removed (reset sequence complete and $\overline{\text{RSTIN}}$ inactive), the reset configuration is latched from PORT0, and pins ALE, $\overline{\text{RD}}$ and $\overline{\text{WR}}$ are driven to their inactive levels.
After the internal reset condition is removed, the CPU will start program execution from memory location $00'0000_H$ in code segment zero. This start location will typically hold a branch instruction to the start of a software initialization routine for the application specific configuration of peripherals and CPU Special Function Registers.

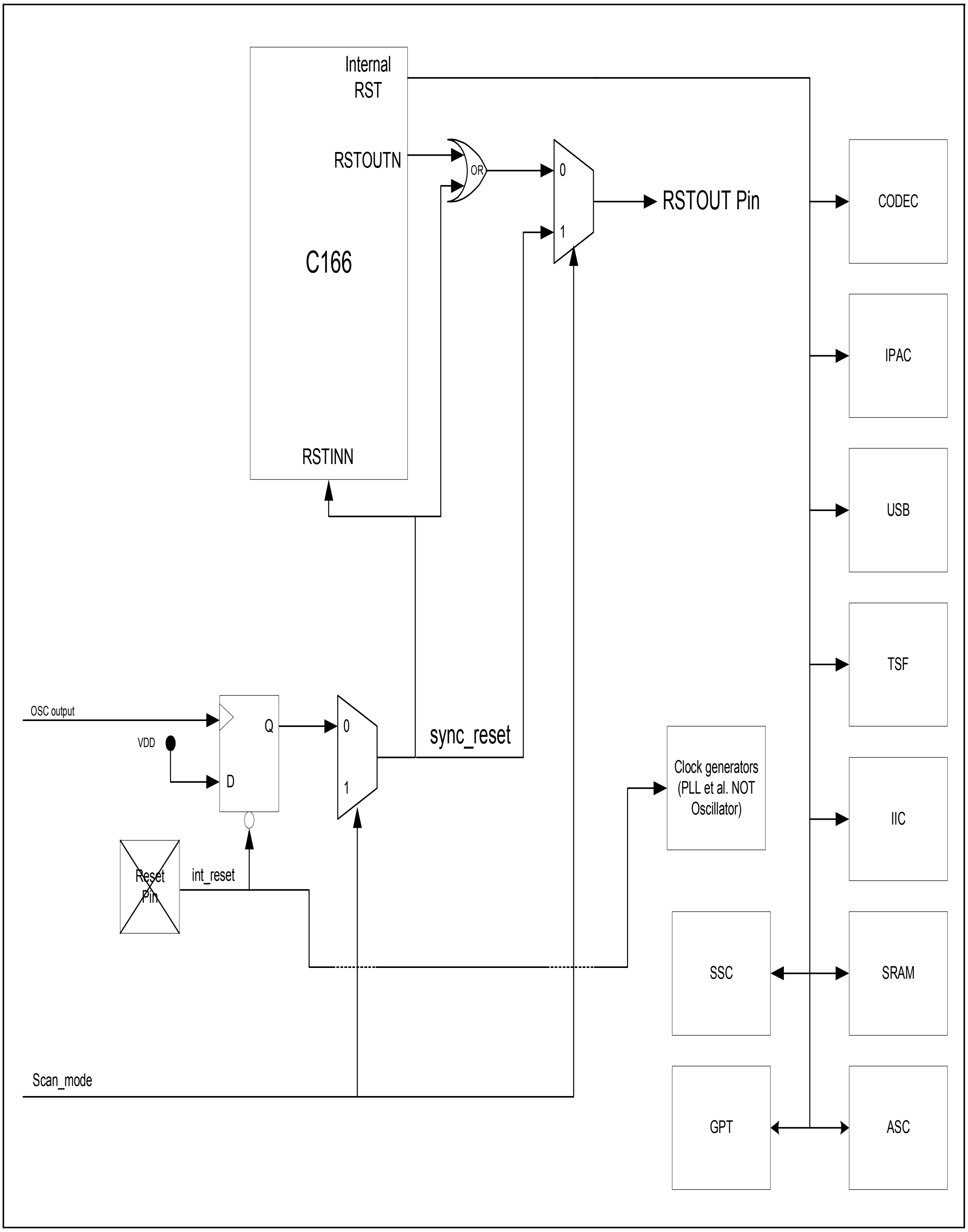


Figure 24-1 Reset Concept

24.1 Reset Types

The INCA-D differs and indicates the following reset types:

1) Minimum time of a short HW reset SHWR for safe latching

Table 24-1 Reset Types and Reset Conditions

Reset Type	Short-cut	Condition
Power-on Reset	PONR	Power-on
Short Hardware Reset	SHWR	16 TCL[1] < t_{RSTIN} ≤2048 TCL
Long Hardware Reset	LHWR	t_{RSTIN} > 2048 TCL
Watchdog Timer Reset	WDTR	WDT overflow
Software Reset	SWR	SRST command

The duration of 2048 TCL is determined by the internal reset sequence (see below) and by the necessary sampling time for the startup configuration on port P0.

External Hardware Reset

The INCA-D provides spike suppression for $\overline{\text{RSTIN}}$ pin with an input filter; input signals shorter than 10ns are suppressed, detection is guaranteed for minimum 150ns reset signals. The filtered and stable HW reset signal is buffered and distributed to the core and to the other modules including the pads (signal $\overline{\text{RST}}$). This internal system reset signal uses directly the filtered, but asynchronous $\overline{\text{RSTIN}}$ signal for its activation. Its trailing edge (deactivation) is determined either

- by the synchronized trailing edge of the $\overline{\text{RSTIN}}$ input (hardware reset), or
- by termination of the reset sequence (see below), if still running when $\overline{\text{RSTIN}}$ is deactivated, or
- by termination of the internal lengthening condition, e.g. osc/PLL stabilisation time (see below).

With deactivation of the internal RST signal, program execution is started in the core.

Three different kinds of external hardware resets are considered:

- Power-on Reset
  A complete power-on reset requires an active $\overline{\text{RSTIN}}$ time until a stable clock signal is available. Depending on the oscillation frequency the on-chip oscillator needs about 2...50 ms to stabilize.
- Long Hardware Reset
  A long hardware reset requires an active $\overline{\text{RSTIN}}$ time longer than the duration of the internal reset sequence. The duration of the internal reset sequence is 2048 TCL. After the internal reset sequence has been completed, the $\overline{\text{RSTIN}}$ input is sampled. As long as the reset input is still active the internal reset condition is prolonged.

Note: The hardware reset is also used for wakeup from power down state; in this case the internal system reset will be lengthened (execution of 1. instruction delayed) until the oscillator and PLL have been stabilized.

– Short Hardware Reset
  The active $\overline{\text{RSTIN}}$ time of a short hardware reset is between 16 TCL and 2048 TCL. If the $\overline{\text{RSTIN}}$ signal is active for at least 16 TCL clock cycles, the internal reset sequence is started (see below). If the $\overline{\text{RSTIN}}$ signal is active for more then 2048 TCL, its a long hardware reset.
  The short hardware reset type is also indicated to the WDT block.

24.1.1 Software Reset

This reset type is generated and controlled in the core. The core provides a control signal (8TCL) to start the reset sequence and to indicate this reset type in the WDTCON register. As long as the reset sequence is running the internal system reset signal RST is generated. The trailing edge of RST starts program execution.

After SW reset the startup configuration is latched in again.

24.1.2 WDT Reset

When the watchdog timer is not disabled during the initialization or serviced regularly during program execution it will overflow and trigger the watchdog timer reset (signal to the core). Other than a hardware reset the watchdog timer reset completes a running external bus cycle. Then the core provides a control signal (WDT Reset Active, 8TCL) to the SCU to start the reset sequence. As long as the reset sequence is running the internal system reset signal RST is generated. The trailing edge of RST starts program execution.

Note: The watchdog timer reset cannot occur while the device is in bootstrap loader mode!

24.2 Reset Sequence Control

A minimum reset duration of 2048 TCL (1024 CPU clock cycles) is necessary for latching of the startup/reset configuration from P0. This minimum reset time is controlled with execution of the so called reset sequence.

The reset sequence is started with every reset type. As long as the reset sequence is running, short reset types (SHWR, WDTR and SWR) are lengthened to the minimum reset duration

24.3 Reset Lengthening Control (Start Delay)

In case of a hardware reset the internal system reset time as controlled with the external reset input $\overline{\text{RSTIN}}$ (power-on or long HW-reset) or with the reset sequence time (short

HW-reset) will be lengthened if the internal clocking system (oscillator and PLL) is still not stabilized (see above). In these cases the internal system reset signal RST is lengthened and the execution of first instruction after reset is delayed until the PLL is stabilized.

Note: The start delay because of Flash voltage and/or clock system rampup and stabilisation time also has to be considered in case of wakeup from Idle mode. In these cases the start is not delayed by reset lengthening; but a setup-active signal is provided to the core to delay the execution of first instruction (system hold is lengthened by start delay of clock distribution).

The reset(setup) lengthening conditions are indicated to the SCU by dedicated lengthening or busy signals. The total time of the conditions delaying the execution of first instruction after reset or wakeup is called the setup time.

Note: The above described lengthening conditions for setup control do not lengthen the sampling time of startup configuration and the bidirectional reset on $\overline{\text{RSTIN}}$ pin .

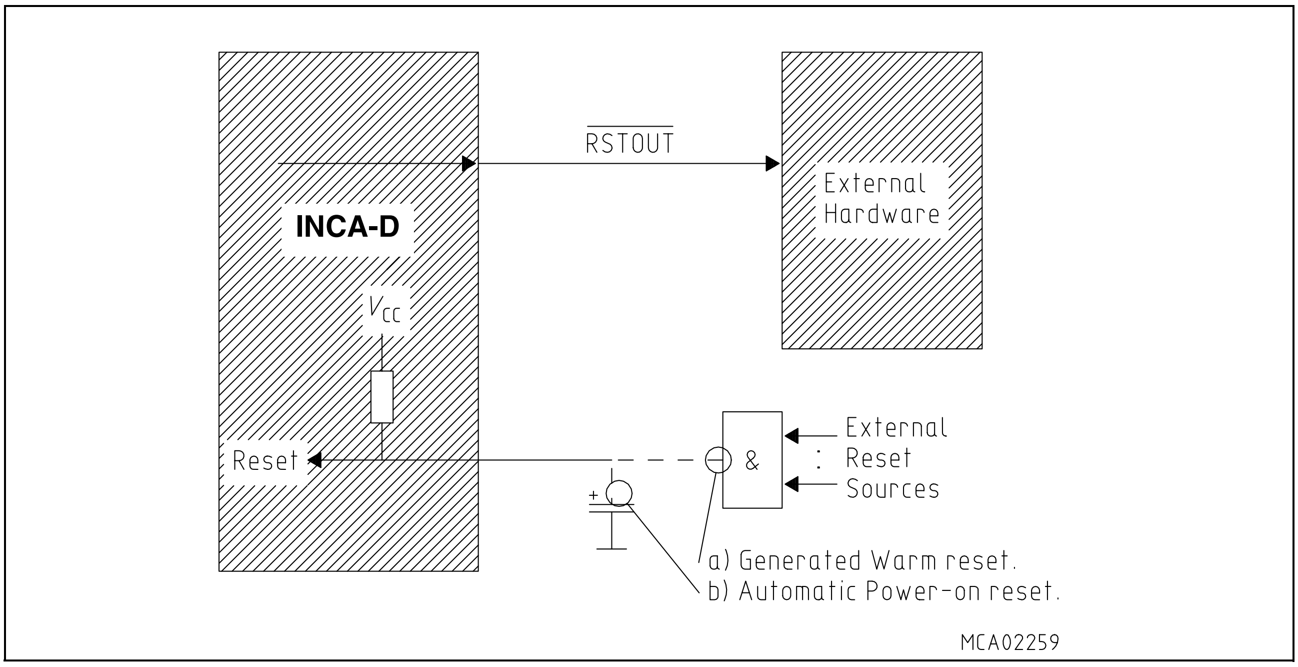


Figure 24-2 External Reset Circuitry

24.4 Power on reset

Because the input $\overline{\text{RSTIN}}$ provides an internal pullup device, a simple connection of an external capacitor is sufficient for an automatic power-on reset (see b) in figure above). $\overline{\text{RSTIN}}$ may also be connected to the output of other logic gates (see a) in figure above).

Note: A power-on reset requires an active time of two reset sequences (1036 CPU clock cycles) after a stable clock signal is available (about 10...50 ms to allow the on-chip oscillator to stabilize).

24.5 Software Reset for IOM-Handler and Transceiver

A dedicated register allows to issue a software reset for the HDLC Controller, the IOM-Handler and the Transceiver. It is not necessary to apply a SW reset after a hardware reset.

SRES - Software Reset Register (E63F$_H$) **XBUS-SFR**

Value after reset: 00$_H$

	7							0
SRES	0	0	0	0	RES_ HDLC	RES_ IOM	RES_ TR	0

RES_xx **... Reset_xx**

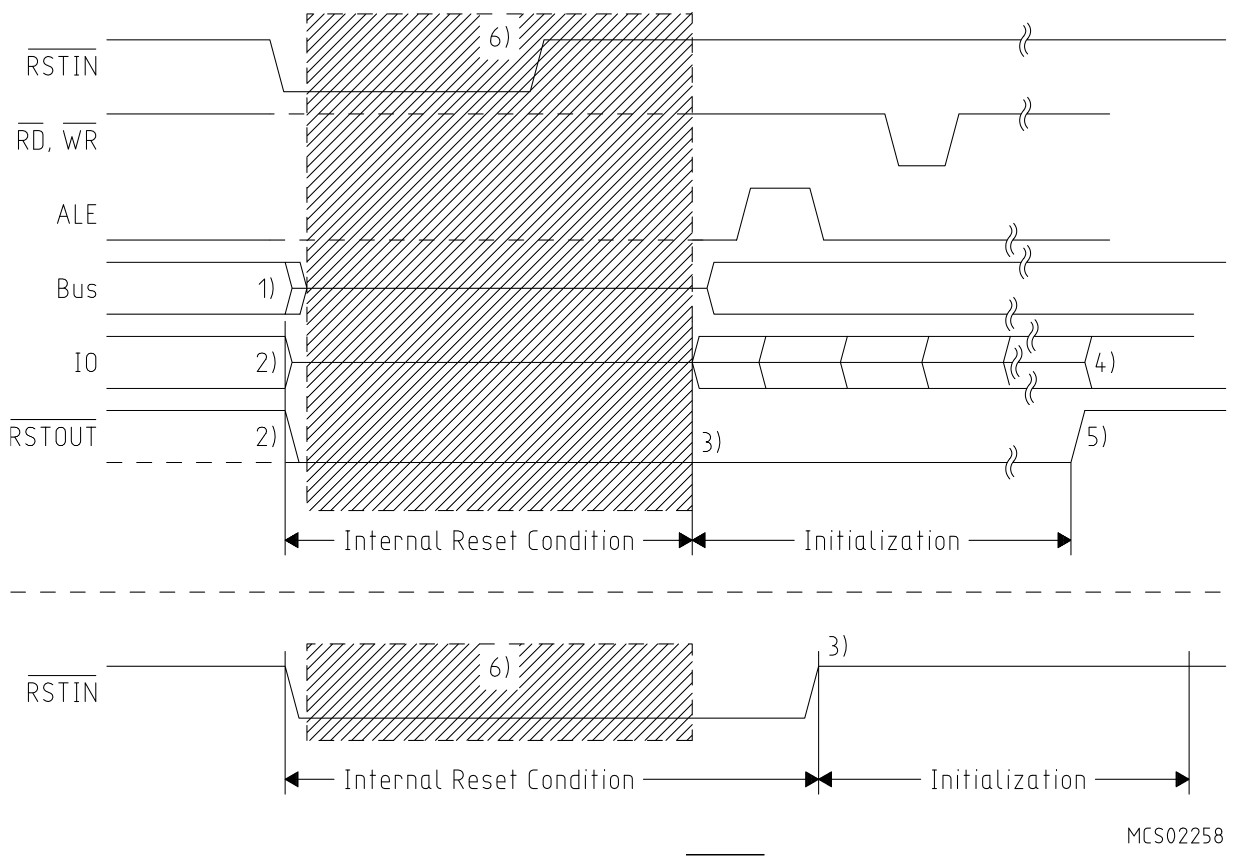
0: Deactivates the reset of the functional block xx
1: Activates the reset of the functional block xx
The reset state is activated as long as the bit is set to ’1’

Meaning of xx:

HDLC:	HDLC controller,
IOM:	IOM Handler,
TR:	Transceiver,

24.6 The INCA-D’s Pins after Reset

After the reset sequence the different groups of pins of the INCA-D are activated in different ways depending on their function. Bus and control signals are activated immediately after the reset sequence according to the configuration latched from PORT0, so either external accesses can take place or the external control signals are inactive. The general purpose I/O pins remain in input mode until reprogrammed via software (see figure below). The $\overline{\text{RSTOUT}}$ pin remains active (low) until the end of the initialization routine (see description).



When the internal reset condition is prolongued by $\overline{\text{RSTIN}}$, the activation of the output signals is delayed until the end of the internal reset condition.

1) Current bus cycle is completed or aborted.
2) Switches asynchronously with $\overline{\text{RSTIN}}$, synchronously upon software or watchdog reset.
3) The reset condition ends here. The INCA-D starts program execution.
4) Activation of the I/O pins is controlled by software.
5) Execution of the EINIT instruction.
6) The shaded area designates the internal reset sequence, which starts after synchronization of $\overline{\text{RSTIN}}$.

Figure 24-3 Reset Input and Output Signals

24.6.1 Reset Output Pin

The $\overline{\text{RSTOUT}}$ pin is dedicated to generate a reset signal for the system components besides the controller itself. $\overline{\text{RSTOUT}}$ will be driven active (low) at the begin of any reset sequence (triggered by hardware, the SRST instruction or a watchdog timer overflow). $\overline{\text{RSTOUT}}$ stays active (low) beyond the end of the internal reset sequence until the protected EINIT (End of Initialization) instruction is executed (see figure above). This allows the complete configuration of the controller including its on-chip peripheral units before releasing the reset signal for the external peripherals of the system.

24.6.2 Watchdog Timer Operation after Reset

The watchdog timer starts running after the internal reset has completed. It will be clocked with the internal system clock divided by 2 (19.2 MHz @ f_{CPU}=38.4 MHz), and its default reload value is 00_H, so a watchdog timer overflow will occur 131072 CPU clock cycles (3.41 ms @ f_{CPU}=38.4 MHz) after completion of the internal reset, unless it is disabled, serviced or reprogrammed meanwhile. When the system reset was caused by a watchdog timer overflow, the WDTR (Watchdog Timer Reset Indication) flag in register WDTCON will be set to '1'. This indicates the cause of the internal reset to the software initialization routine. WDTR is reset to '0' by an external hardware reset or by servicing the watchdog timer. After the internal reset has completed, the operation of the watchdog timer can be disabled by the DISWDT (Disable Watchdog Timer) instruction. This instruction has been implemented as a protected instruction. For further security, its execution is only enabled in the time period after a reset until either the SRVWDT (Service Watchdog Timer) or the EINIT instruction has been executed. Thereafter the DISWDT instruction will have no effect.

24.6.3 Reset and Power Down Mode of the DSP

The DSP with its peripherals can be in either reset mode, power down mode or active mode. During reset the DSP clears the hardware configuration registers and stops both internal and external activity. With the first access to a read/write register the DSP enters active mode. In this mode, normal operation takes place. The DSP with its peripherals is reset when the $\overline{\text{RSTOUT}}$ is active. Figure 24-4 shows a state chart of the modes of the DSP.

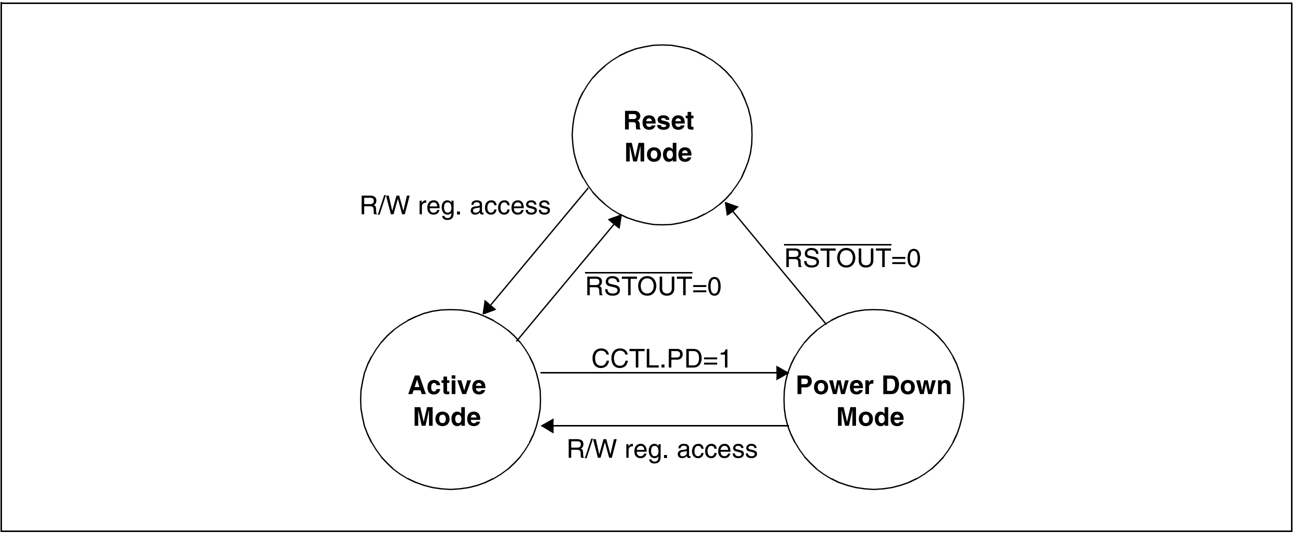


Figure 24-4 Operation Modes of DSP - State Chart

24.7 Ports and External Bus Configuration during Reset

During the internal reset sequence all of the INCA-D's port pins are configured as inputs by clearing the associated direction registers.

Most of the parallel port lines, i.e including the memory interface, are internally pulled up to HIGH per default.(For details refer to Chapter 9)

This ensures that the INCA-D and external devices will not try to drive the same pin to different levels, pin ALE is held low through an internal pulldown, and pins $\overline{\text{RD}}$ and $\overline{\text{WR}}$ are held high through internal pullups. Also the pins selected for $\overline{\text{CS}}$ output will be pulled high.

The registers SYSCON and BUSCON0 are initialized according to the configuration selected via PORT0 as well as the start-up configuration pins at P0L are initializes according to their reset values.

For an external start:

- the Bus Type field (BTYP) in register BUSCON0 is initialized according to P0L.7 and P0L.6
- bit BUSACT0 in register BUSCON0 is set to ‘1’
- bit ALECTL0 in register BUSCON0 is set to ‘1’
- bit BYTDIS in register SYSCON is set according to the data bus width

The other bits of register BUSCON0, and the other BUSCON registers are cleared. This default initialization selects the slowest possible external accesses using the configured bus type.

When the internal reset has completed, the configuration of PORT0, PORT1, Port 4, Port 6 and of the $\overline{\text{BHE}}$ signal (High Byte Enable, alternate function of P3.12) depends on the bus type which was selected during reset, i.e. PORT0 (and PORT1) will operate in the selected bus mode. Port 4 will output the selected number of segment address lines (all zero after reset) and Port 6 will drive the selected number of $\overline{\text{CS}}$ lines ($\overline{\text{CS0}}$ will be ‘0’, while the other active $\overline{\text{CS}}$ line will be ‘1’). When no memory accesses above 64 K are to be performed, segmentation may be disabled.

When the on-chip bootstrap loader was activated during reset, pin TxD0 (alternate function of P3.10) will be switched to output mode after the reception of the zero byte.

Application-Specific Initialization Routine

After the internal reset condition is removed the INCA-D fetches the first instruction from location 00’0000$_H$, which is the first vector in the trap/interrupt vector table, the reset vector. 4 words (locations 00’0000$_H$ through 00’0007$_H$) are provided in this table to start the initialization after reset. As a rule, this location holds a branch instruction to the actual initialization routine that may be located anywhere in the address space.

Note: When the Bootstrap Loader Mode was activated during a hardware reset the INCA-D does not fetch instructions from location 00'0000$_H$ but rather expects data via serial interface ASC.

The first instruction is fetched from external memory. To decrease the number of instructions required to initialize the INCA-D, each peripheral is programmed to a default configuration upon reset, but is disabled from operation. These default configurations can be found in the descriptions of the individual peripherals.

During the software design phase, portions of the internal memory space must be assigned to register banks and system stack. When initializating the stack pointer (SP) and the context pointer (CP), it must be ensured that these registers are initialized before any GPR or stack operation is performed. This includes interrupt processing, which is disabled upon completion of the internal reset, and should remain disabled until the SP is initialized.

Note: Traps may occur, even though the interrupt system is still disabled.

In addition, the stack overflow (STKOV) and the stack underflow (STKUN) registers should be initialized. After reset, the CP, SP, and STKUN registers all contain the same reset value 00'FC00$_H$, while the STKOV register contains 00'FA00$_H$. With the default reset initialization, 256 words of system stack are available, where the system stack selected by the SP grows downwards from 00'FBFE$_H$, while the register bank selected by the CP grows upwards from 00'FC00$_H$.

Based on the application, the user may wish to initialize portions of the internal memory before normal program operation. Once the register bank has been selected by programming the CP register, the desired portions of the internal memory can easily be initialized via indirect addressing.

At the end of the initialization, the interrupt system may be globally enabled by setting bit IEN in register PSW. Care must be taken not to enable the interrupt system before the initialization is complete.

The software initialization routine should be terminated with the EINIT instruction. This instruction has been implemented as a protected instruction. Execution of the EINIT instruction disables the action of the DISWDT instruction, disables write accesses to register SYSCON (see note) and causes the $\overline{\text{RSTOUT}}$ pin to go high. This signal can be used to indicate the end of the initialization routine and the proper operation of the microcontroller to external hardware.

Note: All configurations regarding register SYSCON (enable CLKOUT, stacksize, etc.) must be selected before the execution of EINIT.

24.8 System Startup Configuration

Although most of the programmable features of the INCA-D are either selected during the initialization phase or repeatedly during program execution, there are some features that must be selected earlier, because they are used for the first access of the program execution.

These selections are made during reset via the pins of PORT0, which are read at the end of the internal reset sequence. During reset internal pullup or pulldown devices are active on the PORT0 lines, so their input level is high or low, respectively. With the coding of the selections, as shown below, in many cases the default option, ie. high level, can be used.

The system startup configuration is sampled and latched with a specific reset signal provided by the Reset Control Block of the INCA-D. For latching of the high byte of port P0 configuration the register RP0H is used.

The **RP0H** register is defined as follows:

RP0H (F108$_H$ / 84$_H$)ESFR-bReset Value: --XX$_H$!Latching of Startup Configuration

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	-	-	-	-	-	-	-	-	-	R	SALSEL		CSSEL		WRCFC
-	-	-	-	-	-	-	-	-	-	r	r	r	r	r	r

Bit	Function
R	reserved
WRCFC	**Write Configuration Control** (also indicated in SYSCON register) ‘0’: Standard function of pins $\overline{WR}$ and $\overline{BHE}$ ‘1’: Pin $\overline{WR}$ acts as $\overline{WRL}$, pin $\overline{BHE}$ acts as $\overline{WRH}$
CSSEL	**Chip Select Lines Selection** Number of active $\overline{CS}$ outputs. Description of possible selections: see table below (Startup Configuration)
SALSEL	**Segment Address Lines Selection** Number of active segment address outputs. Description of possible selections: see table below (Startup Configuration)

For latching of P0 startup configuration (from XBUS) the following reset types - see description of Reset Control Block - are differentiated:

- Power-on reset (PONR) or long (warm) HW reset (LHWR)
- Short (warm) HW reset (SHWR)
- SW reset (SWR) or watchdog timer reset (WDTR)

The following table shows latching of system startup configuration in dependence of the reset type and of the bidirectional reset function.

X : Pin is sampled **-** : Pin is not transparent and not sampled	**PORT0**												
	Segm.	Addr. Lines	Chip Selects		WR Config.	Bus Type		System Modes				Adapt Mode	Emu Mode
Sample event	P0H.4	P0H.3	P0H.2	P0H.1	P0H.0	P0L.7	P0L.6	P0L.5	P0L.4	P0L.3	P0L.2	P0L.1	P0L.0
PONR	X	X	X	X	X	X	X	X	X	X	X	X	X
LHWR	X	X	X	X	X	X	X	X	X	X	X	X	X
SHWR	X	X	X	X	X	X	X	X	X	X	X	X	X
WDTR/SWR	X	X	X	X	X	X	X	-	-	-	-	-	-
	Latched into RP0H					BUSCON0		FFs					

Table 24-2 Latching of System Startup Configuration from PORT0

Code Definitions of Startup Configuration

The low order 6 bits of startup configuration define the system mode of operation. These bits are only **decoded in the core** for selection and control of the core related modes of operation, including the normal start after reset with instruction fetch from address 00'0000. Also the configuration of bus type is sampled and latched only in the core.

The following table showes in detail all supported selections for port P0 startup configuration. In this table, the note 1 is assigned to core related system modes of operation, which are controlled in the CB-Core.

H.7	H.6	H.5	H.4	H.3	H.2	H.1	H.0	L.7	L.6	L.5	L.4	L.3	L.2	L.1	L.0
R	**R**	**R**	**SALSEL**		**CSSEL**		**WRC**	**BUSTYP**		**SMOD**				**ADP**	**EMU**

Pin	Mode	Comment
EMU	Emulation Mode	internal test mode
ADP	Adapt Mode	internal test mode
SMOD (P0L.5:2)	**Special Modes System Operation Modes**	
1 0 1 1	Bootstrap Loader	Start from internal boot memory

1 1 1 1	Normal Start	Normal start
all other	reserved	
BUSTYP (P0L.7:6)	**External Data Bus Width**	External Address Bus Mode
0 0	8-bit Data	Demultiplexed Addresses
0 1	8-bit Data	Multiplexed Addresses
1 0	16-bit Data	Demultiplexed Addresses
1 1	16-bit Data	Multiplexed Addresses
WRC	**Write Configuration**	
CSSEL (P0H.2:1)	**Chip Select Lines**	
1 1	reserved	
1 0	None	P6.0, P6.1 and P6.2 usable as GPIOs
0 1	$\overline{CS1}$ and $\overline{CS0}$ availabe	two CS lines at pins P6.0 and P6.1
0 0	$\overline{CS2}$, $\overline{CS1}$ and $\overline{CS0}$ availabe	three CS lines at pins P6.0, P6.1 and P6.2
SALSEL (P0H.4:3)	**Segment Address Lines**	**Directly accessible Address Space**
1 1	Two: A17...A16	256 KByte (Default, without pull-downs)
1 0	A21...A16	4MByte (Maximum)
0 1	None	64 KByte (Minimum)
0 0	Four: A19...A16	1 MByte

Table 24-3 Code definitions of startup configurations

25 The Register Set

This section summarizes all registers, which are implemented in the INCA-D and explains the description format which is used in the chapters describing the function and layout of the (E)SFRs ((Extended) Special Function Register).

If not explicitely mentioned, SFR's refer to peripherals which are connected to the PD-bus (see **Figure 3-1**). Thus the corresponding memory area is limited to 00'FED0$_H$ - 00'FFFF$_H$ for the SFR's and 00'F000$_H$ -00'F200$_H$ for the ESFR's.

The peripherals which are connected to the XBUS refer to an address range given by 00'DD00$_H$ -00'DFFF$_H$.**Figure 25-1** gives an overview about the used memory locations for XBUS peripherals.

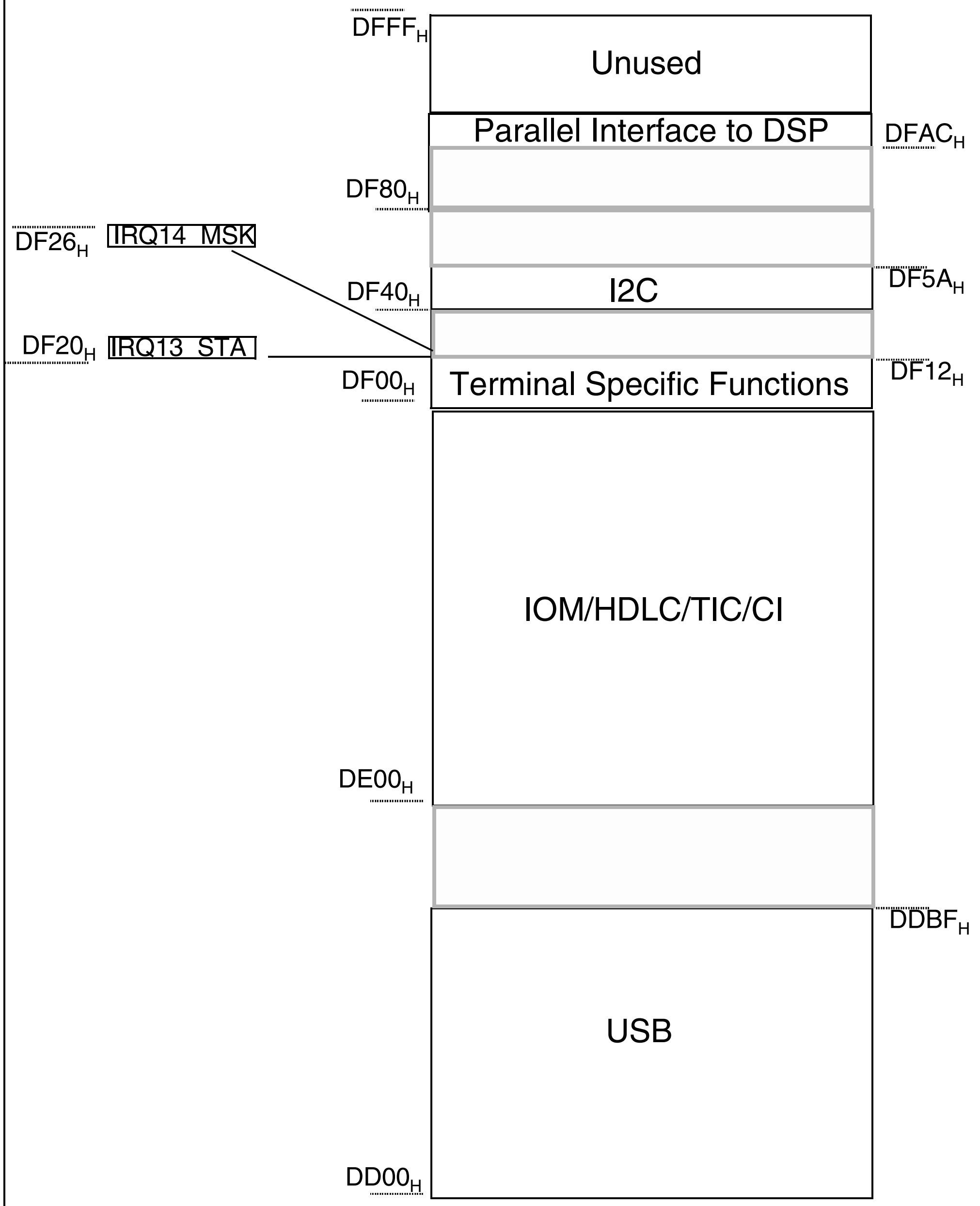


Figure 25-1 Memory area for XBUS peripherals

For easy reference the registers are ordered according to two different keys (except for GPRs):

- Ordered by address, to check which register a given address references,

25.1 Register Description Format

In the respective chapters the function and the layout of the SFRs is described in a specific format which provides a number of details about the described special function register. The example below shows how to interpret these details.

A word register looks like this:

REG_NAME **(*$A16_H$ / $A8_H$*)** **E**/SFR **Reset Value: * * * *$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
res.	**res.**	**res.**	**res.**	**res.**	**write only**	**hw bit**	**read only**	**std bit**	**hw bit**		**bitfield**			**bitfield**	
-	-	-	-	-	w	rw	r	rw	rw		rw			rw	

Bit	Function
bit(field)name	***Explanation of bit(field)name*** *Description of the functions controlled by this bit(field).*

A byte register looks like this:

REG_NAME **(*$A16_H$ / $A8_H$*)** **E**/SFR **Reset Value: - - * *$_H$**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
								std bit	**hw bit**		**bitfield**			**bitfield**	
-	-	-	-	-	-	-	-	rw	rw		rw			rw	

Elements:

REG_NAME	Name of this register
A16 / A8	Long 16-bit address / Short 8-bit address
SFR/**ESFR**/**XReg**	Register space (SFR, ESFR or External/XBUS Register)
(* *) * *	Register contents after reset **0**/**1**: defined value, '**X**': undefined, '**U**': unchanged (undefined ('**X**') after power up)
hwbit	Bits that are set/cleared by hardware are marked with a shaded access box

25.2 CPU General Purpose Registers (GPRs)

The GPRs form the register bank that the CPU works with. This register bank may be located anywhere within the internal RAM via the Context Pointer (CP). Due to the addressing mechanism, GPR banks can only reside within the internal RAM. All GPRs are bit-addressable.

Name	Physical Address	8-Bit Address	Description	Reset Value
R0	(CP) + 0	$F0_H$	CPU General Purpose (Word) Register R0	$UUUU_H$
R1	(CP) + 2	$F1_H$	CPU General Purpose (Word) Register R1	$UUUU_H$
R2	(CP) + 4	$F2_H$	CPU General Purpose (Word) Register R2	$UUUU_H$
R3	(CP) + 6	$F3_H$	CPU General Purpose (Word) Register R3	$UUUU_H$
R4	(CP) + 8	$F4_H$	CPU General Purpose (Word) Register R4	$UUUU_H$
R5	(CP) + 10	$F5_H$	CPU General Purpose (Word) Register R5	$UUUU_H$
R6	(CP) + 12	$F6_H$	CPU General Purpose (Word) Register R6	$UUUU_H$
R7	(CP) + 14	$F7_H$	CPU General Purpose (Word) Register R7	$UUUU_H$
R8	(CP) + 16	$F8_H$	CPU General Purpose (Word) Register R8	$UUUU_H$
R9	(CP) + 18	$F9_H$	CPU General Purpose (Word) Register R9	$UUUU_H$
R10	(CP) + 20	FA_H	CPU General Purpose (Word) Register R10	$UUUU_H$
R11	(CP) + 22	FB_H	CPU General Purpose (Word) Register R11	$UUUU_H$
R12	(CP) + 24	FC_H	CPU General Purpose (Word) Register R12	$UUUU_H$
R13	(CP) + 26	FD_H	CPU General Purpose (Word) Register R13	$UUUU_H$
R14	(CP) + 28	FE_H	CPU General Purpose (Word) Register R14	$UUUU_H$
R15	(CP) + 30	FF_H	CPU General Purpose (Word) Register R15	$UUUU_H$

The first 8 GPRs (R7...R0) may also be accessed bytewise. Other than with SFRs, writing to a GPR byte does not affect the other byte of the respective GPR.
The respective halves of the byte-accessible registers receive special names:

Name	Physical Address	8-Bit Address	Description	Reset Value
RL0	(CP) + 0	$F0_H$	CPU General Purpose (Byte) Register RL0	UU_H
RH0	(CP) + 1	$F1_H$	CPU General Purpose (Byte) Register RH0	UU_H
RL1	(CP) + 2	$F2_H$	CPU General Purpose (Byte) Register RL1	UU_H
RH1	(CP) + 3	$F3_H$	CPU General Purpose (Byte) Register RH1	UU_H
RL2	(CP) + 4	$F4_H$	CPU General Purpose (Byte) Register RL2	UU_H
RH2	(CP) + 5	$F5_H$	CPU General Purpose (Byte) Register RH2	UU_H
RL3	(CP) + 6	$F6_H$	CPU General Purpose (Byte) Register RL3	UU_H
RH3	(CP) + 7	$F7_H$	CPU General Purpose (Byte) Register RH3	UU_H
RL4	(CP) + 8	$F8_H$	CPU General Purpose (Byte) Register RL4	UU_H
RH4	(CP) + 9	$F9_H$	CPU General Purpose (Byte) Register RH4	UU_H

Name	Physical Address	8-Bit Address	Description	Reset Value
RL5	(CP) + 10	FA_H	CPU General Purpose (Byte) Register RL5	UU_H
RH5	(CP) + 11	FB_H	CPU General Purpose (Byte) Register RH5	UU_H
RL6	(CP) + 12	FC_H	CPU General Purpose (Byte) Register RL6	UU_H
RH6	(CP) + 13	FD_H	CPU General Purpose (Byte) Register RH6	UU_H
RL7	(CP) + 14	FE_H	CPU General Purpose (Byte) Register RL7	UU_H
RH7	(CP) + 14	FF_H	CPU General Purpose (Byte) Register RH7	UU_H

25.3 XBUS Special Function Registers ordered by Address

The following table lists all SFRs which are implemented in the INCA-D ordered by physical address. **Bit-addressable** SFRs are marked with the letter “**b**” in column “Type”.
SFRs within the **Extended SFR-Space** (ESFRs) are marked with the letter “**E**” in column “Type”.

Table 25-1 Registers ordered by Address

Physical Addr	Register Name	Description	Reset Value
$DD01_H$	GESRL	Global Endpoint Stall Register L	00_H
$DD02_H$	GESRH	Global Endpoint Stall Register H	00_H
$DD03_H$	CIARL	Configuration Request Register L	00_H
$DD04_H$	CIARH	Configuration Request Register H	00_H
$DD05_H$	GEPIRL	Global Endpoint Interrupt Register L	00_H
$DD06_H$	GEPIRH	Global Endpoint Interrupt Register H	00_H
$DD07_H$	CIARI	Configuration Request Interrupt Register	00_H
$DD08_H$	CIARIE	Configuration Request Interrupt Enable Register	00_H
$DD09_H$	DCR	Device Control Register	00_H
$DD0A_H$	DPWDR	Device Power Down Register	00_H
$DD0B_H$	DIER	Device Interrupt Enable Register	00_H
$DD0C_H$	DIRR	Device Interrupt Request Register	00_H

Physical Addr	Register Name	Description	Reset Value
$DD0D_H$	DSSR	Device Setup and Status Register	00_H
$DD0E_H$	FNRL	Frame Number Low Registers	XXXX $XXXX_B$
$DD0F_H$	FNRH	Frame Number High Registers	0000 $0XXX_B$
$DD10_H$	DGSRL	Device Get Status Register L	00_H
$DD11_H$	DGSRH	Device Get Status Register H	00_H
$DD12_H$	IGSRL	Interface Get Status Register L	00_B
$DD13_H$	IGSRLH	Interface Get Status Register H	00_H
$DD14_H$	IFCSEL	Interface Select Register	00_H
$DD15_H$	ESTR	Enable Setup Token	00_H
$DD16_H$	ACTEP	Active Endpoint	00_H
$DD20_H$	EGSR0L	Endpoint Get Status Register 0 L	00_H
$DD21_H$	EGSR1L	Endpoint Get Status Register 1 L	00_H
$DD22_H$	EGSR2L	Endpoint Get Status Register 2 L	00_H
$DD23_H$	EGSR3L	Endpoint Get Status Register 3 L	00_H
$DD24_H$	EGSR4L	Endpoint Get Status Register 4 L	00_H
$DD25_H$	EGSR5L	Endpoint Get Status Register 5 L	00_H
$DD26_H$	EGSR6L	Endpoint Get Status Register 6 L	00_H
$DD27_H$	EGSR7L	Endpoint Get Status Register 7 L	00_H
$DD28_H$	EGSR8L	Endpoint Get Status Register 8 L	00_H
$DD29_H$	EGSR9L	Endpoint Get Status Register 9 L	00_H
$DD2A_H$	EGSR10L	Endpoint Get Status Register 10 L	00_H

Physical Addr	Register Name	Description	Reset Value
DD2B$_H$	EGSR11L	Endpoint Get Status Register11 L	00$_H$
DD2C$_H$	EGSR12L	Endpoint Get Status Register 12 L	00$_H$
DD2D$_H$	EGSR13L	Endpoint Get Status Register 13 L	00$_H$
DD2E$_H$	EGSR14L	Endpoint Get Status Register 14 L	00$_H$
DD2F$_H$	EGSR15L	Endpoint Get Status Register 15 L	00$_H$
DD30$_H$	EGSR0H	Endpoint Get Status Register 0 H	00$_H$
DD31$_H$	EGSR1H	Endpoint Get Status Register 1 H	00$_H$
DD32$_H$	EGSR2H	Endpoint Get Status Register 2 H	00$_H$
DD33$_H$	EGSR3H	Endpoint Get Status Register 3 H	00$_H$
DD34$_H$	EGSR4H	Endpoint Get Status Register 4 H	00$_H$
DD35$_H$	EGSR5H	Endpoint Get Status Register 5 H	00$_H$
DD36$_H$	EGSR6H	Endpoint Get Status Register 6 H	00$_H$
DD37$_H$	EGSR7H	Endpoint Get Status Register 7 H	00$_H$
DD38$_H$	EGSR8H	Endpoint Get Status Register 8 H	00$_H$
DD39$_H$	EGSR9H	Endpoint Get Status Register 9 H	00$_H$
DD3A$_H$	EGSR10H	Endpoint Get Status Register 10 H	00$_H$
DD3B$_H$	EGSR11H	Endpoint Get Status Register11 H	00$_H$
DD3C$_H$	EGSR12H	Endpoint Get Status Register 12 H	00$_H$
DD3D$_H$	EGSR13H	Endpoint Get Status Register 13 H	00$_H$
DD3E$_H$	EGSR14H	Endpoint Get Status Register 14 H	00$_H$
DD3F$_H$	EGSR15H	Endpoint Get Status Register 15 H	00$_H$

Physical Addr	Register Name	Description	Reset Value
DD40$_H$	EPIE0	Endpoint Interrupt Enable Register 0	00$_H$
DD41$_H$	EPIE1	Endpoint Interrupt Enable Register 1	00$_H$
DD42$_H$	EPIE2	Endpoint Interrupt Enable Register 2	00$_H$
DD43$_H$	EPIE3	Endpoint Interrupt Enable Register 3	00$_H$
DD44$_H$	EPIE4	Endpoint Interrupt Enable Register 4	00$_H$
DD45$_H$	EPIE5	Endpoint Interrupt Enable Register 5	00$_H$
DD46$_H$	EPIE6	Endpoint Interrupt Enable Register 6	00$_H$
DD47$_H$	EPIE7	Endpoint Interrupt Enable Register 7	00$_H$
DD48$_H$	EPIE8	Endpoint Interrupt Enable Register 8	00$_H$
DD49$_H$	EPIE9	Endpoint Interrupt Enable Register 9	00$_H$
DD4A$_H$	EPIE10	Endpoint Interrupt Enable Register 10	00$_H$
DD4B$_H$	EPIE11	Endpoint Interrupt Enable Register 11	00$_H$
DD4C$_H$	EPIE12	Endpoint Interrupt Enable Register 12	00$_H$
DD4D$_H$	EPIE13	Endpoint Interrupt Enable Register 13	00$_H$
DD4E$_H$	EPIE14	Endpoint Interrupt Enable Register 14	00$_H$
DD4F$_H$	EPIE15	Endpoint Interrupt Enable Register 15	00$_H$
DD50$_H$	EPIR0	Endpoint Interrupt Request Register 0	11$_H$
DD51$_H$	EPIR1	Endpoint Interrupt Request Register 1	10$_H$
DD52$_H$	EPIR2	Endpoint Interrupt Request Register 2	10$_H$
DD53$_H$	EPIR3	Endpoint Interrupt Request Register 3	10$_H$
DD54$_H$	EPIR4	Endpoint Interrupt Request Register 4	10$_H$

Physical Addr	Register Name	Description	Reset Value
DD55$_H$	EPIR5	Endpoint Interrupt Request Register 5	10$_H$
DD56$_H$	EPIR6	Endpoint Interrupt Request Register 6	10$_H$
DD57$_H$	EPIR7	Endpoint Interrupt Request Register 7	10$_H$
DD58$_H$	EPIR8	Endpoint Interrupt Request Register 8	10$_H$
DD59$_H$	EPIR9	Endpoint Interrupt Request Register 9	10$_H$
DD5A$_H$	EPIR10	Endpoint Interrupt Request Register 10	10$_H$
DD5B$_H$	EPIR11	Endpoint Interrupt Request Register 11	10$_H$
DD5C$_H$	EPIR12	Endpoint Interrupt Request Register 12	10$_H$
DD5D$_H$	EPIR13	Endpoint Interrupt Request Register 13	10$_H$
DD5E$_H$	EPIR14	Endpoint Interrupt Request Register 14	10$_H$
DD5F$_H$	EPIR15	Endpoint Interrupt Request Register 15	10$_H$
DD60$_H$	EPBC0	Endpoint Buffer Control Register 0	00$_H$
DD61$_H$	EPBC1	Endpoint Buffer Control Register 1	00$_H$
DD62$_H$	EPBC2	Endpoint Buffer Control Register 2	00$_H$
DD63$_H$	EPBC3	Endpoint Buffer Control Register 3	00$_H$
DD64$_H$	EPBC4	Endpoint Buffer Control Register 4	00$_H$
DD65$_H$	EPBC5	Endpoint Buffer Control Register 5	00$_H$
DD66$_H$	EPBC6	Endpoint Buffer Control Register 6	00$_H$
DD67$_H$	EPBC7	Endpoint Buffer Control Register 7	00$_H$
DD68$_H$	EPBC8	Endpoint Buffer Control Register 8	00$_H$
DD69$_H$	EPBC9	Endpoint Buffer Control Register 9	00$_H$

Physical Addr	Register Name	Description	Reset Value
DD6A$_H$	EPBC10	Endpoint Buffer Control Register 10	00$_H$
DD6B$_H$	EPBC11	Endpoint Buffer Control Register 11	00$_H$
DD6C$_H$	EPBC12	Endpoint Buffer Control Register 12	00$_H$
DD6D$_H$	EPBC13	Endpoint Buffer Control Register 13	00$_H$
DD6E$_H$	EPBC14	Endpoint Buffer Control Register 14	00$_H$
DD6F$_H$	EPBC15	Endpoint Buffer Control Register 15	00$_H$
DD70$_H$	EPBS0	Endpoint Buffer Status Register 0	20$_H$
DD71$_H$	EPBS1	Endpoint Buffer Status Register 1	20$_H$
DD72$_H$	EPBS2	Endpoint Buffer Status Register 2	20$_H$
DD73$_H$	EPBS3	Endpoint Buffer Status Register 3	20$_H$
DD74$_H$	EPBS4	Endpoint Buffer Status Register 4	20$_H$
DD75$_H$	EPBS5	Endpoint Buffer Status Register 5	20$_H$
DD76$_H$	EPBS6	Endpoint Buffer Status Register 6	20$_H$
DD77$_H$	EPBS7	Endpoint Buffer Status Register 7	20$_H$
DD78$_H$	EPBS8	Endpoint Buffer Status Register 8	20$_H$
DD79$_H$	EPBS9	Endpoint Buffer Status Register 9	20$_H$
DD7A$_H$	EPBS10	Endpoint Buffer Status Register 10	20$_H$
DD7B$_H$	EPBS11	Endpoint Buffer Status Register 11	20$_H$
DD7C$_H$	EPBS12	Endpoint Buffer Status Register 12	20$_H$
DD7D$_H$	EPBS13	Endpoint Buffer Status Register 13	20$_H$
DD7E$_H$	EPBS14	Endpoint Buffer Status Register 14	20$_H$

Physical Addr	Register Name	Description	Reset Value
$DD7F_H$	EPBS15	Endpoint Buffer Status Register 15	20_H
$DD80_H$	EPBA0	Endpoint Base Address Register 0	00_H
$DD81_H$	EPBA1	Endpoint Base Address Register 1	00_H
$DD82_H$	EPBA2	Endpoint Base Address Register 2	00_H
$DD83_H$	EPBA3	Endpoint Base Address Register 3	00_H
$DD84_H$	EPBA4	Endpoint Base Address Register 4	00_H
$DD85_H$	EPBA5	Endpoint Base Address Register 5	00_H
$DD86_H$	EPBA6	Endpoint Base Address Register 6	00_H
$DD87_H$	EPBA7	Endpoint Base Address Register 7	00_H
$DD88_H$	EPBA8	Endpoint Base Address Register 8	00_H
$DD89_H$	EPBA9	Endpoint Base Address Register 9	00_H
$DD8A_H$	EPBA10	Endpoint Base Address Register 10	00_H
$DD8B_H$	EPBA11	Endpoint Base Address Register 11	00_H
$DD8C_H$	EPBA12	Endpoint Base Address Register 12	00_H
$DD8D_H$	EPBA13	Endpoint Base Address Register 13	00_H
$DD8E_H$	EPBA14	Endpoint Base Address Register 14	00_H
$DD8F_H$	EPBA15	Endpoint Base Address Register 15	00_H
$DD90_H$	EPLEN0	Endpoint Buffer Length Register 0	$0XXXXXX_B$
$DD91_H$	EPLEN1	Endpoint Buffer Length Register 1	$0XXXXXX_B$
$DD92_H$	EPLEN2	Endpoint Buffer Length Register 2	$0XXXXXX_B$
$DD93_H$	EPLEN3	Endpoint Buffer Length Register 3	$0XXXXXX_B$

Physical Addr	Register Name	Description	Reset Value
$DD94_H$	EPLEN4	Endpoint Buffer Length Register 4	$0XXXXXX_B$
$DD95_H$	EPLEN5	Endpoint Buffer Length Register 5	$0XXXXXX_B$
$DD96_H$	EPLEN6	Endpoint Buffer Length Register 6	$0XXXXXX_B$
$DD97_H$	EPLEN7	Endpoint Buffer Length Register 7	$0XXXXXX_B$
$DD98_H$	EPLEN8	Endpoint Buffer Length Register 8	$0XXXXXX_B$
$DD99_H$	EPLEN9	Endpoint Buffer Length Register 9	$0XXXXXX_B$
$DD9A_H$	EPLEN10	Endpoint Buffer Length Register 10	$0XXXXXX_B$
$DD9B_H$	EPLEN11	Endpoint Buffer Length Register 11	$0XXXXXX_B$
$DD9C_H$	EPLEN12	Endpoint Buffer Length Register 12	$0XXXXXX_B$
$DD9D_H$	EPLEN13	Endpoint Buffer Length Register 13	$0XXXXXX_B$
$DD9E_H$	EPLEN14	Endpoint Buffer Length Register 14	$0XXXXXX_B$
$DD9F_H$	EPLEN15	Endpoint Buffer Length Register 15	$0XXXXXX_B$
$DDA0_H$	ADROFF0	Address Offset Register 0	00_H
$DDA1_H$	ADROFF1	Address Offset Register 1	00_H
$DDA2_H$	ADROFF2	Address Offset Register 2	00_H
$DDA3_H$	ADROFF3	Address Offset Register 3	00_H
$DDA4_H$	ADROFF4	Address Offset Register 4	00_H
$DDA5_H$	ADROFF5	Address Offset Register 5	00_H
$DDA6_H$	ADROFF6	Address Offset Register 6	00_H
$DDA7_H$	ADROFF7	Address Offset Register 7	00_H

Physical Addr	Register Name	Description	Reset Value
$DDA8_H$	ADROFF8	Address Offset Register 8	00_H
$DDA9_H$	ADROFF9	Address Offset Register 9	00_H
$DDAA_H$	ADROFF10	Address Offset Register 10	00_H
$DDAB_H$	ADROFF11	Address Offset Register 11	00_H
$DDAC_H$	ADROFF12	Address Offset Register 12	00_H
$DDAD_H$	ADROFF13	Address Offset Register 13	00_H
$DDAE_H$	ADROFF14	Address Offset Register 14	00_H
$DDAF_H$	ADROFF15	Address Offset Register 15	00_H
$DDB0_H$	UBSVAL0	USB Data Register 0	00_H
$DDB1_H$	UBSVAL1	USB Data Register 1	00_H
$DDB2_H$	UBSVAL2	USB Data Register 2	00_H
$DDB3_H$	UBSVAL3	USB Data Register 3	00_H
$DDB4_H$	UBSVAL4	USB Data Register 4	00_H
$DDB5_H$	UBSVAL5	USB Data Register 5	00_H
$DDB6_H$	UBSVAL6	USB Data Register 6	00_H
$DDB7_H$	UBSVAL7	USB Data Register 7	00_H
$DDB8_H$	UBSVAL8	USB Data Register 8	00_H
$DDB9_H$	UBSVAL9	USB Data Register 9	00_H
$DDBA_H$	UBSVAL10	USB Data Register 10	00_H

Physical Addr	Register Name	Description	Reset Value
DDBB$_H$	UBSVAL11	USB Data Register 11	00$_H$
DDBC$_H$	UBSVAL12	USB Data Register 12	00$_H$
DDBD$_H$	UBSVAL13	USB Data Register 13	00$_H$
DDBE$_H$	UBSVAL14	USB Data Register 14	00$_H$
DDBF$_H$	UBSVAL15	USB Data Register 15	00$_H$
DE00$_H$ -DE1F$_H$	RFIFO	Receive FIFO	
DDD0$_H$ -DE1F$_H$	XFIFO	Transmit FIFO	
DE20$_H$	ISTAH	Interrupt Status Register HDLC	10$_H$
DE20$_H$	MASKH	Mask Register HDLC	FC$_H$
DE21$_H$	STAR	Status Register	40$_H$
DE21$_H$	CMDR	Command Register	00$_H$
DE22$_H$	MODEH	Mode Register	C0$_H$
DE23$_H$	EXMR	Extended Mode Register	00$_H$
DE25$_H$	SAP1	SAP1 Register	FC$_H$
DE26$_H$	SAP2	SAPI2 Register	FC$_H$
DE26$_H$	RBCL	Receive Frame Byte Count Low	00$_H$

Physical Addr	Register Name	Description	Reset Value
DE27$_H$	RBCH	Receive Frame Byte High	00$_H$
DE27$_H$	TEI1	REI1 Register	FF$_H$
DE28$_H$	TEI2	REI2 Register	FF$_H$
DE28$_H$	RSTA	Receive Status Register	0F$_H$
DE29$_H$	TMH	Test Mode Register HDLC	00$_H$
DE2A$_H$ -DE2D$_H$	reserved		
DE2E$_H$	CIR0	Command/Indication Receive 0	F3$_H$
DE2E$_H$	CIX0	Command/Indication Transmit 0	FE$_H$
DE2F$_H$	CIR1	Command/Indication Receive 1	FC$_H$
DE2F$_H$	CIX1	Command/Indication Transmit 1	FE$_H$
DE30$_H$	TR_ CONF0	Transceiver Configuration Register 0	00$_H$
DE31$_H$	TR_ CONF1	Transceiver Configuration Register 1	64$_H$
DE32$_H$	TR_ CONF2	Transceiver Configuration Register 2	00$_H$
DE33$_H$	TR_STA	Transceiver Status Register	00$_H$
DE34$_H$	TR_CMD	Transceiver Command Register	08$_H$
DE35$_H$		*reserved*	
DE36$_H$		*reserved*	
DE37$_H$		*reserved*	

Physical Addr	Register Name	Description	Reset Value
$DE38_H$	ISTATR	Interrupt Status Register Transceiver	00_H
$DE39_H$	MASKTR	Mask Register Transceiver	$7F_H$
$DE3A_H$	TR_MODE	Transceiver Mode Register	10_H
$DE40_H$	CDA10	Controller Data Access Register 10	FF_H
$DE41_H$	CDA11	Controller Data Access Register 11	FF_H
$DE42_H$	CDA20	Controller Data Access Register 20	FF_H
$DE43_H$	CDA21	Controller Data Access Register 21	FF_H
$DE44_H$	CDA_TSDP10	Time Slot and Data Selection for CH 10	00_H
$DE45_H$	CDA_TSDP11	Time Slot and Data Selection for CH 11	01_H
$DE46_H$	CDA_TSDP20	Time Slot and Data Selection for CH 20	80_H
$DE47_H$	CDA_TSDP21	Time Slot and Data Selection for CH 21	81_H
$DE48_H$	CO_TSDP10	Time Slot and Data Selection for CH 10	80_H
$DE49_H$	CO_TSDP11	Time Slot and Data Selection for CH 11	81_H
$DE4A_H$	CO_TSDP20	Time Slot and Data Selection for CH 20	81_H
$DE4B_H$	CO_TSDP21	Time Slot and Data Selection for CH 21	85_H
$DE4C_H$	TR_TSDP_B1	Time Slot and Data Selection for CH B1	01_H
$DE4D_H$	TR_TSDP_B2	Time Slot and Data Selection for CH B2	01_H
$DE4E_H$	CDA1_CR	Control Register Controller Data Access CH11	00_H
$DE4F_H$	CDA2_CR	Control Register Controller Data Access CH12	00_H

Physical Addr	Register Name	Description	Reset Value
DE50$_H$	TR_CR (CI_CS=0)	Control Register Transceiver Data	F8$_H$
DE51$_H$	BCHA_CR	Control Register B-Channel Data	80$_H$
DE52$_H$	BCHB_CR	Control Register B-Channel Data	81$_H$
DE53$_H$	DCI_CR (CI_CS=0)	Control Register for HDLC and CI1 Data	A0$_H$
DE53$_H$	DCI_CR (CI_CS=1)	Control Register for HDLC and CI0 Data	E0$_H$
DE55$_H$	SDS1_CR	Control Register Serial DataStrobe 1	00$_H$
DE56$_H$	SDS2_CR	Control Register Serial DataStrobe 2	00$_H$
DE57$_H$	IOM_CR	Control Register IOM Data	01$_H$
DE58$_H$	STI	Synchronous Transfer Interrupt	00$_H$
DE58$_H$	ASTI	Acknowledge Synchronous Transfer Interrupt	00$_H$
DE59$_H$	MSTI	Mask Synchronous Transfer Interrupt	FF$_H$
DE5A$_H$	SDS_CONF	Configuration Register for Serial Data Strobes	00$_H$
DE5B$_H$	MCDA	Monitoring CDA Bits	FF$_H$
DE60$_H$	ISTA	Interrupt Status Register	00$_H$
DE60$_H$	MASK	Mask Register	FF$_H$
DE62$_H$	MODDE	Mode1 Register	00$_H$
DE64$_H$	ID	Identification Register	01$_H$
DE64$_H$	SRES	Software Reset Register	00$_H$

Physical Addr	Register Name	Description	Reset Value
DE65$_H$ - DE6F$_H$			
DE90$_H$	ITRDU	IOM Transfer Unit, Read/Write Register DU	00$_H$
DE91$_H$	ITRDD	IOM Transfer Unit, Read/Write Register DD	00$_H$
DE92$_H$	ITRICV	IOM Transfer Unit, Idle CodeValue for Transfer mode 0	FF$_H$
DE94$_H$	ITR_CR	IOM Transfer Control Register Register	00 00$_H$
DE96$_H$	ITR_MSK0	IOM Transfer Time Slot and Data Port Selection Register 0	00 00$_H$
DE98$_H$	ITR_MSK1	IOM Transfer Time Slot and Data Port Selection Register 1	00 00$_H$
DE9A$_H$	ITR_MSK2	IOM Transfer Time Slot and Data Port Selection Register 2	00 00$_H$
DE9C$_H$	ITR_MSK3	IOM Transfer Time Slot and Data Port Selection Register 3	00 00$_H$
DE9E$_H$	ITR_MSK4	IOM Transfer Time Slot and Data Port Selection Register 4	00 00$_H$
DEA0$_H$	ITR_MSK5	IOM Transfer Time Slot and Data Port Selection Register 5	00 00$_H$
DEA2$_H$	ITR_MSK6	IOM Transfer Time Slot and Data Port Selection Register 6	00 00$_H$
DEA4$_H$	ITR_MSK7	IOM Transfer Time Slot and Data Port Selection Register 7	00 00$_H$
DEB0$_H$	IWSR	IOM Waitstate register	FFFF$_H$
DF00$_H$	SREG0	Keypad Scanner Register Description	FFFF$_H$
DF02$_H$	SREG1	Keypad Scanner Register Description	FFFF$_H$
DF04$_H$	SREG23	Keypad Scanner Register Description	FFFF$_H$

Physical Addr	Register Name	Description	Reset Value
DF06$_H$	SREG45	Keypad Scanner Register Description	FFFF$_H$
DF08$_H$	SREG6789	Keypad Scanner Register Description	FFFF$_H$
DF0A$_H$	Key_CONF	Keypad Scanner Register Description	0000$_H$
DF0C$_H$	LEDMUX_CONF	LED Multiplex Unit Register Description	0000$_H$
DF0E$_H$	LED_REG1	LED Multiplex Unit Register Description	00 00$_H$
DF10$_H$	LED_REG2	LED Multiplex Unit Register Description	0000$_H$
DF12$_H$	TSF_CONF	TSF Global Register Description	00 00$_H$
DF14$_H$	PWM1_WIDTH	Pulse Width Units Register Description	00 00$_H$
DF16$_H$	PWM2_WIDTH	Pulse Width Units Register Description	00 00$_H$
DF20$_H$	IRQ13_STA	Status register of interrupt node 13	
DF22$_H$	IRQ13_MSK	Mask register for IRQ13_STA	
DF24$_H$	IRQ14_STA	Status register of interrupt node 14	
DF26$_H$	IRQ14_MSK	Mask register for IRQ14_STA	
DF38$_H$	REXICON	Regular external interrupts control register	00 00$_H$
DF40$_H$	I2CCLC	Clock Control Register	0002$_H$
DF50$_H$	ICCFG	IIC Bus Control Register	00 00$_H$
DF52$_H$	ICCON	IIC System Control Register	00 00$_H$
DF54$_H$	ICST	IIC System Control Register	00 00$_H$
DF56$_H$	ICADR	IIC Bus Control Register	00 00$_H$
DF58$_H$	RTBL	IIC Receive Transmit Buffer	00 00$_H$
DF5A$_H$	RTBH	IIC Receive Transmit Buffer	00 00$_H$

Physical Addr	Register Name	Description	Reset Value
DFAA$_H$	CLK_CONF	Clock configuration register	0000$_H$
DFAC$_H$	PIDDCLC	Clock control register of PIDD	0000$_H$
DFAE$_H$	PIDDCOM	Command register of PIDD	0000$_H$
DFB0$_H$	PIDDSTAT	Status register of PIDD	0000$_H$
DFB2$_H$	PIDDHWCFG	Hardware configuration register	0000$_H$
DFB4$_H$	PIDDGPTC	Data (general prupose) from DSP to control via PIDD	0000$_H$
DFB6$_H$	PIDDGPTD	Data (general purpose) from controller to DSP via PIDD	0000$_H$

25.4 PD-Bus Special Function Registers ordered by Address

The following table lists all SFRs which are implemented in the INCA-D ordered by physical address. **Bit-addressable** SFRs are marked with the letter “**b**” in column “Type”.
SFRs within the **Extended SFR-Space** (ESFRs) are marked with the letter “**E**” in column “Type”.

Table 25-2 Registers ordered by Address

Physical Addr	Register Name	Type	8-bit Addr	Description	Reset Value
F000$_H$	reserved				
F002$_H$	reserved				
F004$_H$	reserved				
F006$_H$	reserved				
F008$_H$	reserved				
F00A$_H$	reserved				
F00C$_H$	reserved				
F010$_H$	reserved				
F012$_H$	reserved				
F014$_H$	XADRS1	ESFR	0A$_H$	XBUS Address Select Register 1	0000$_H$
F016$_H$	XADRS2	ESFR	0B$_H$	XBUS Address Select Register 2	0000$_H$
F018$_H$	XADRS3	ESFR	0C$_H$	XBUS Address Select Register 3	0000$_H$
F01A$_H$	XADRS4	ESFR	0D$_H$	XBUS Address Select Register 4	0000$_H$
F01C$_H$	*reserved*	ESFR	0E$_H$		
F01E$_H$	*reserved*	ESFR	0F$_H$		
F024$_H$	XPERCON	ESFR	12$_H$	XBUS Peripheral Control Register	0000$_H$
F068$_H$	*reserved*	ESFR	34$_H$	*reserved*	xxxx$_H$
F06A$_H$	*reserved*	ESFR	35$_H$	*reserved*	xxxx$_H$
F06C$_H$	*reserved*	ESFR	36$_H$	*reserved*	
F070$_H$	IDRT	ESFR	38$_H$	Identifier	0000$_H$

Physical Addr	Register Name	Type	8-bit Addr	Description	Reset Value
F076$_H$	IDMEM2	ESFR	3B$_H$	Identifier	0000$_H$
F078$_H$	IDPROG	ESFR	3C$_H$	Identifier	0000$_H$
F07A$_H$	IDMEM	ESFR	3D$_H$	Identifier	0000$_H$
F07C$_H$	IDCHIP	ESFR	3E$_H$	Identifier	0801$_H$
F07E$_H$	IDMANUF	ESFR	3F$_H$	Identifier	1824$_H$
F0B0$_H$	SSCTB	ESFR	58$_H$	SSC Transmit Buffer (WO)	0000$_H$
F0B2$_H$	SSCRB	ESFR	59$_H$	SSC Receive Buffer (RO)	0000$_H$
F0B4$_H$	SSCBR	ESFR	5A$_H$	SSC Baudrate Register	0000$_H$
F0B6$_H$	SSCCLC	ESFR	5B$_H$	SSC Clock Control Register	0000$_H$
F0D8$_H$	DTIDR	ESFR	6C$_H$	Task ID register)	0000[1]$_H$
F0DC$_H$	DCMPL	ESFR	6E$_H$	Hardware event range comparison register (less)	0000[1]$_H$
F0DE$_H$	DCMPLX	ESFR	6F$_H$	8 bit extension for DCMPL	0000[1]$_H$
F0E0$_H$	DCMPG	ESFR	70$_H$	Hardware event range comparison register (greater)	0000[1]$_H$
F0E2$_H$	DCMPGX	ESFR	71$_H$	8 bit extension for DCMPG	0000[1]$_H$
F0E4$_H$	DCMP0	ESFR	72$_H$	Hardware event equal comparison register 0	0000[1]$_H$
F0E6$_H$	DCMP0X	ESFR	73$_H$	8 bit extension for DCMP0	0000[1]$_H$
F0E8$_H$	DCMP1	ESFR	74$_H$	Hardware event equal comparison register 1	0000[1]$_H$
F0EA$_H$	DCMP1X	ESFR	75$_H$	8 bit extension for DCMP1	0000[1]$_H$
F0EC$_H$	DCMP2	ESFR	76$_H$	Hardware event equal comparison register 2	0000[1]$_H$
F0EE$_H$	DCMP2X	ESFR	77$_H$	8 bit extension for DCMP2	0000[1]$_H$
F0F0$_H$	DTREVT	ESFR	78$_H$	Specifies hardware triggers and action	0000[1]$_H$
F0F4$_H$	DSWEVT	ESFR	7A$_H$	Specifies action if DEBUG instruction is executed	0000[1]$_H$
F0F8$_H$	DEXEVT	ESFR	7C$_H$	Specifies action if external break pin is asserted	0000[1]$_H$
F0FA$_H$	DIPX	ESFR	7D$_H$	Instruction pointer register extension for OCDS	0000[1]$_H$
F0FC$_H$	DBGSR	ESFR	7E$_H$	Debug status register	0000[1]$_H$
F0FE$_H$	DIP	ESFR	7F$_H$	Instruction pointer register for OCDS	0000[1]$_H$
F100$_H$	DP0L	ESFR-b	80$_H$	P0L Direction Control Register	00$_H$
F102$_H$	DP0H	ESFR-b	81$_H$	P0H Direction Control Register	00$_H$
F104$_H$	DP1L	ESFR-b	82$_H$	P1L Direction Control Register	00$_H$
F106$_H$	DP1H	ESFR-b	83$_H$	P1H Direction Control Register	00$_H$
F108$_H$	RP0H	ESFR-b	84$_H$	System Startup Configuration Register (RO)	xx$_H$
F114$_H$	XBCON 1	ESFR-b	8A$_H$	XBUS Control register 1 (PIDD, TSF, I2C)	0000$_H$
F116$_H$	XBCON 2	ESFR-b	8B$_H$	XBUS Control register 2 (IOM, HDLC)	0000$_H$
F118$_H$	XBCON 3	ESFR-b	8C$_H$	XBUS Control register 3 (USB)	0000$_H$
F11A$_H$	XBCON 4	ESFR-b	8D$_H$	XBUS Control register 4 (XRAM)	0000$_H$

Physical Addr	Register Name	Type	8-bit Addr	Description	Reset Value
F11C$_H$	*reserved*	ESFR-b	8E$_H$		
F11E$_H$	*reserved*	ESFR-b	8F$_H$		
F122$_H$	P2ALTSEL0	ESFR	91$_H$	Alternate Function Enable Port 2	0000$_H$
F124$_H$	P2HALTSEL1	ESFR	92$_H$	Second Alternate Function Enable Port 2	0000$_H$
F126$_H$	P3ALTSEL0	ESFR	93$_H$	Alternate Function Enable Port 3	0000$_H$
F128$_H$	P3ALTSEL1	ESFR	94$_H$	Second Alternate Function Enable Port 3	0000$_H$
F12C$_H$	P6ALTSEL0	ESFR	96$_H$	Alternate Function Enable Port 6	0000$_H$
F12E$_H$	P7ALTSEL0	ESFR	97$_H$	Alternate Function Enable Port 7	0000$_H$
F180$_H$	PECCLIC	ESFR-b	C0$_H$	PEC Channel Link Interrupt Control Register	
F194$_H$	ICINTEIC	SFR-b	CA$_H$	I2C Data Transmision Interrupt Control Register	0000$_H$
F19C$_H$	S0TBIC	ESFR-b	CE$_H$	ASC Transmit Buffer Interrupt Control Register	0000$_H$
F1C0$_H$	EXICON	ESFR-b	E0$_H$	External Interrupt Control Register	0000$_H$
F1C2$_H$	ODP2	ESFR-b	E1$_H$	Port 2 Open Drain Control Register	0000$_H$
F1C6$_H$	ODP3	ESFR-b	E3$_H$	Port 3 Open Drain Control Register	0000$_H$
F1CA$_H$	ODP4	ESFR-b	E5$_H$	Port 4 Open Drain Control Register	00$_H$
F1CE$_H$	ODP6	ESFR-b	E7$_H$	Port 6 Open Drain Control Register	00$_H$
F1D0$_H$	*reserved*	ESFR-b	E8$_H$		0000$_H$
F1D2$_H$	ODP7	ESFR-b	E9$_H$	Port 7 Open Drain Control Register	00$_H$
F1D4$_H$	*reserved*	ESFR-b	EA$_H$	reserved - do not use	0000$_H$
F1D6$_H$	*reserved*	ESFR-b	EB$_H$	reserved - do not use	0000$_H$
F1D8$_H$	*reserved*	ESFR-b	EC$_H$	reserved - do not use	0000$_H$
F1DA$_H$	*reserved*	ESFR-b	ED$_H$	reserved - do not use	0000$_H$
F1DC$_H$	*reserved*	ESFR-b	EE$_H$	reserved - do not use	0000$_H$
F1DE$_H$	ISNC	ESFR-b	EF$_H$	Interrupt Sub Node Control Register	0000$_H$
FE00$_H$	DPP0	SFR	00$_H$	CPU Data Page Pointer 0 Register (10 bits)	0000$_H$
FE02$_H$	DPP1	SFR	01$_H$	CPU Data Page Pointer 1 Register (10 bits)	0001$_H$
FE04$_H$	DPP2	SFR	02$_H$	CPU Data Page Pointer 2 Register (10 bits)	0002$_H$
FE06$_H$	DPP3	SFR	03$_H$	CPU Data Page Pointer 3 Register (10 bits)	0003$_H$
FE08$_H$	CSP	SFR	04$_H$	CPU Code Segment Pointer Register (8 bits)	0000$_H$
FE0A$_H$	*reserved*	SFR	05$_H$	*reserved*	xxxx$_H$
FE0C$_H$	MDH	SFR	06$_H$	CPU Multiply Divide Register - High Word	0000$_H$
FE0E$_H$	MDL	SFR	07$_H$	CPU Multiply Divide Register - Low Word	0000$_H$
FE10$_H$	CP	SFR	08$_H$	CPU Context Pointer Register	FC00$_H$
FE12$_H$	SP	SFR	09$_H$	CPU System Stack Pointer Register	FC00$_H$
FE14$_H$	STKOV	SFR	0A$_H$	CPU Stack Overflow Pointer Register	FA00$_H$

Physical Addr	Register Name	Type	8-bit Addr	Description	Reset Value
FE16$_H$	STKUN	SFR	0B$_H$	CPU Stack Underflow Pointer Register	FC00$_H$
FE18$_H$	ADDRSEL1	SFR	0C$_H$	Address Select Register 1	0000$_H$
FE1A$_H$	ADDRSEL2	SFR	0D$_H$	Address Select Register 2	0000$_H$
FE1C$_H$	ADDRSEL3	SFR	0E$_H$	Address Select Register 3	0000$_H$
FE1E$_H$	ADDRSEL4	SFR	0F$_H$	Address Select Register 4	0000$_H$
FE20$_H$	ODP0L	SFR	10$_H$	Port 0 Open Drain Control Register Low	0000$_H$
FE22$_H$	ODP0H	SFR	11$_H$	Port 0 Open Drain Control Register High	0000$_H$
FE24$_H$	ODP1L	SFR	12$_H$	Port 1 Open Drain Control Register Low	0000$_H$
FE26$_H$	ODP1H	SFR	13$_H$	Port 1 Open Drain Control Register High	0000$_H$
FE40$_H$	T2	SFR	20$_H$	GPT1 Timer 2 Register	0000$_H$
FE42$_H$	T3	SFR	21$_H$	GPT1 Timer 3 Register	0000$_H$
FE44$_H$	T4	SFR	22$_H$	GPT1 Timer 4 Register	0000$_H$
FE46$_H$	T5	SFR	23$_H$	GPT2 Timer 5 Register	0000$_H$
FE48$_H$	T6	SFR	24$_H$	GPT2 Timer 6 Register	0000$_H$
FE4A$_H$	CAPREL	SFR	25$_H$	GPT1/2 Capture / Reload Register	0000$_H$
FE4C$_H$	GPTCLC	SFR	26$_H$	GPT1/2 Clock Control Register	0000$_H$
FE60$_H$	P0LPUDSEL	SFR	30$_H$	Port 0 Low Pull-Up/Down Select Register	00BF$_H$
FE62$_H$	P0HPUDSEL	SFR	31$_H$	Port 0 High Pull-Up/Down Select Register	00FB$_H$
FE64$_H$	P0LPUDEN	SFR	32$_H$	Port 0 Low Pull Switch On/Off Register	00FF
FE66$_H$	P0HPUDEN	SFR	33$_H$	Port 0 High Pull Switch On/Off Register	00FF$_H$
FE68$_H$	P0LPHEN	SFR	34$_H$	Port 0 Low Pin Hold Enable Register	0000$_H$
FE6A$_H$	P0HPHEN	SFR	35$_H$	Port 0 High Pin Hold Enable Register	0000$_H$
FE6C$_H$	P1LPUDSEL	SFR	36$_H$	Port 1 Low Pull-Up/Down Select Register	00FF$_H$
FE6E$_H$	P1HPUDSEL	SFR	37$_H$	Port 1 High Pull-Up/Down Select Register	00FF$_H$
FE70$_H$	P1LPUDEN	SFR	38$_H$	Port 1 Low Pull Switch On/Off Register	00FF$_H$
FE72$_H$	P1HPUDEN	SFR	39$_H$	Port 1 High Pull Switch On/Off Register	00FF$_H$
FE74$_H$	P1LPHEN	SFR	3A$_H$	Port 1 Low Pin Hold Enable Register	0000$_H$
FE76$_H$	P1HPHEN	SFR	3B$_H$	Port 1 High Pin Hold Enable Register	0000$_H$
FE78$_H$	P2PUDSEL	SFR	3C$_H$	Port 2 Pull-Up/Down Select Register	00FF$_H$
FE7A$_H$	P2PUDEN	SFR	3D$_H$	Port 2 Pull Switch On/Off Register	00FF$_H$
FE7C$_H$	P2PHEN	SFR	3E$_H$	Port 2 Pin Hold Enable Register	0000$_H$
FE7E$_H$	P3PUDSEL	SFR	3F$_H$	Port 3 Pull-Up/Down Select Register	00FF$_H$
FE80$_H$	P3PUDEN	SFR	40$_H$	Port 3 Pull Switch On/Off Register	00FF$_H$
FE82$_H$	P3PHEN	SFR	41$_H$	Port 3 Pin Hold Enable Register	0000$_H$
FE84$_H$	P4PUDSEL	SFR	42$_H$	Port 4 Pull-Up/Down Select Register	00FF$_H$

Physical Addr	Register Name	Type	8-bit Addr	Description	Reset Value
FE86$_H$	P4PUDEN	SFR	43$_H$	Port 4 Pull Switch On/Off Register	00FF$_H$
FE88$_H$	P4PHEN	SFR	44$_H$	Port 4 Pin Hold Enable Register	0000$_H$
FE90$_H$	P6PUDSEL	SFR	48$_H$	Port 6 Pull-Up/Down Select Register	00FF$_H$
FE92$_H$	P6PUDEN	SFR	49$_H$	Port 6 Pull Switch On/Off Register	00FF$_H$
FE94$_H$	P6PHEN	SFR	4A$_H$	Port 6 Pin Hold Enable Register	0000$_H$
FE96$_H$	P7PUDSEL	SFR	4B$_H$	Port 7 Pull-Up/Down Select Register	00FF$_H$
FE98$_H$	P7PUDEN	SFR	4C$_H$	Port 7 Pull Switch On/Off Register	00FF$_H$
FE9A$_H$	P7PHEN	SFR	4D$_H$	Port 7 Pin Hold Enable Register	0000$_H$
FEAA$_H$	S0PMW	SFR	55$_H$	ASC IrDA PMW Control Register	0000$_H$
FEAE$_H$	WDT	SFR	57$_H$	Watchdog Timer Register (RO)	0000$_H$
FEB0$_H$	S0TBUF	SFR	58$_H$	Serial Channel 0 Transmit Buffer Register (WO)	0000$_H$
FEB2$_H$	S0RBUF	SFR	59$_H$	Serial Channel 0 Receive Buffer Register (RO)	0000$_H$
FEB4$_H$	S0BG	SFR	5A$_H$	Serial Channel 0 Baud Rate Generator Reload Register	0000$_H$
FEB6$_H$	S0FDV	SFR	5B$_H$	ASC Fractional Divide Register	0000$_H$
FEC0$_H$	PECC0	SFR	60$_H$	PEC Channel 0 Control Register	0000$_H$
FEC2$_H$	PECC1	SFR	61$_H$	PEC Channel 1 Control Register	0000$_H$
FEC4$_H$	PECC2	SFR	62$_H$	PEC Channel 2 Control Register	0000$_H$
FEC6$_H$	PECC3	SFR	63$_H$	PEC Channel 3 Control Register	0000$_H$
FEC8$_H$	PECC4	SFR	64$_H$	PEC Channel 4 Control Register	0000$_H$
FECA$_H$	PECC5	SFR	65$_H$	PEC Channel 5 Control Register	0000$_H$
FECC$_H$	PECC6	SFR	66$_H$	PEC Channel 6 Control Register	0000$_H$
FECE$_H$	PECC7	SFR	67$_H$	PEC Channel 7 Control Register	0000$_H$
FED0$_H$	PECSN0	SFR	68$_H$	PEC Segment No Register	0000$_H$
FED2$_H$	PECSN1	SFR	69$_H$	PEC Segment No Register	0000$_H$
FED4$_H$	PECSN2	SFR	6A$_H$	PEC Segment No Register	0000$_H$
FED6$_H$	PECSN3	SFR	6B$_H$	PEC Segment No Register	0000$_H$
FED8$_H$	PECSN4	SFR	6C$_H$	PEC Segment No Register	0000$_H$
FEDA$_H$	PECSN5	SFR	6D$_H$	PEC Segment No Register	0000$_H$
FEDC$_H$	PECSN6	SFR	6E$_H$	PEC Segment No Register	0000$_H$
FEDE$_H$	PECSN7	SFR	6F$_H$	PEC Segment No Register	0000$_H$
FEF0$_H$	PECXC0	SFR	78$_H$	PEC Channel 0 Extended Control Register	0000$_H$
FEF2$_H$	PECXC1	SFR	79$_H$	PEC Channel 1 Extended Control Register	0000$_H$
FEF8$_H$	ABS0CON	SFR	7C$_H$		0000$_H$
FEFE$_H$	ABSTAT	SFR	7F$_H$		0000$_H$
FF00$_H$	P0L	SFR-b	80$_H$	Port 0 Low Register (Lower half)	00$_H$

Physical Addr	Register Name	Type	8-bit Addr	Description	Reset Value
FF02$_H$	P0H	SFR-b	81$_H$	Port 0 High Register (Upper half)	00$_H$
FF04$_H$	P1L	SFR-b	82$_H$	Port 1 Low Register (Lower half)	00$_H$
FF06$_H$	P1H	SFR-b	83$_H$	Port 1 High Register (Upper half)	00$_H$
FF0C$_H$	BUSCON0	SFR-b	86$_H$	Bus Configuration Register 0	0680$_H$
FF0E$_H$	MDC	SFR-b	87$_H$	CPU Multiply Divide Control Register	0000$_H$
FF10$_H$	PSW	SFR-b	88$_H$	CPU Program Status Word	0000$_H$
FF12$_H$	SYSCON	SFR-b	89$_H$	CPU System Configuration Register	0xx0$_H$
FF14$_H$	BUSCON1	SFR-b	8A$_H$	Bus Configuration Register 1	0000$_H$
FF16$_H$	BUSCON2	SFR-b	8B$_H$	Bus Configuration Register 2	0000$_H$
FF18$_H$	BUSCON3	SFR-b	8C$_H$	Bus Configuration Register 3	0000$_H$
FF1A$_H$	BUSCON4	SFR-b	8D$_H$	Bus Configuration Register 4	0000$_H$
FF1C$_H$	ZEROS	SFR-b	8E$_H$	Constant Value 0sRegister'	0000$_H$
FF1E$_H$	ONES	SFR-b	8F$_H$	Constant Value 1sRegister'	FFFF$_H$
FF40$_H$	T2CON	SFR-b	A0$_H$	GPT1 Timer 2 Control Register	0000$_H$
FF42$_H$	T3CON	SFR-b	A1$_H$	GPT1 Timer 3 Control Register	0000$_H$
FF44$_H$	T4CON	SFR-b	A2$_H$	GPT1 Timer 4 Control Register	0000$_H$
FF46$_H$	T5CON	SFR-b	A3$_H$	GPT2 Timer 5 Control Register	0000$_H$
FF48$_H$	T6CON	SFR-b	A4$_H$	GPT2 Timer 6 Control Register	0000$_H$
FF60$_H$	IOM5IC	ESFR-b	B0$_H$	IOM Transfer Unit 5 Interrupt Control Registe	0000$_H$
FF62$_H$	IOM6IC	SFR-b	B1$_H$	IOM Transfer Unit 6Interrupt Control Registe	0000$_H$
FF64$_H$	IOM7IC	SFR-b	B2$_H$	IOM Transfer Unit 7 Interrupt Control Registe	0000$_H$
FF66$_H$	IOM8IC	SFR-b	B3$_H$	IOM Transfer Unit 8 Interrupt Control Registe	0000$_H$
FF68$_H$	COMB1IC	SFR-b	B4$_H$	Combined Interrupt Control Register 1	0000$_H$
FF6A$_H$	IICINTDIC	SFR-b	B5$_H$	I2C Data Transfer Interrupt Control Register	0000$_H$
FF6C$_H$	COMB2IC	SFR-b	B6$_H$	Combined Interrupt Control Register 2	0000$_H$
FF6E$_H$	S0RIC	SFR-b	B7$_H$	ASC Receive Interrupt Control Register	0000$_H$
FF70$_H$	USBEP3IC	SFR-b	B8$_H$	USB Endpoint 3 Interrupt Control Registe	0000$_H$
FF72$_H$	SSCTIC	SFR-b	B9$_H$	SSC Transmit Interrupt Control Register	0000$_H$
FF74$_H$	SSCRIC	SFR-b	BA$_H$	SSC Receive Interrupt Control Register	0000$_H$
FF76$_H$	USBEP2IC	ESFR-b	BB$_H$	USB Endpoint 2 Interrupt Control Register	0000$_H$
FF78$_H$	USBEP4IC	SFR-b	BC$_H$	USB Endpoint 4 Interrupt Control Register	0000$_H$

Physical Addr	Register Name	Type	8-bit Addr	Description	Reset Value
$FF7E_H$	IOM4IC	SFR-b	BF_H	IOM Transfer Unit 4 Interrupt Control Registe	0000_H
$FF80_H$	IOM3IC	SFR-b	$C0_H$	IOM Transfer Unit 3 Interrupt Control Registe	0000_H
$FF82_H$	IOM2IC	SFR-b	$C1_H$	IOM Transfer Unit 2 Interrupt Control Registe	0000_H
$FF84_H$	IOM1IC	SFR-b	$C2_H$	IOM Transfer Unit 1 Interrupt Control Register	0000_H
$FF86_H$	T2IC	SFR-b	$B0_H$	GPT1 Timer 2 Interrupt Control Register	0000_H
$FF88_H$	FEX0IC	SFR-b	$C4_H$	Fast External Interrupt 0 Control Register	0000_H
$FF8A_H$	FEX1IC	SFR-b	$C5_H$	Fast External Interrupt 1 Control Register	0000_H
$FF8C_H$	FEX2IC	SFR-b	$C6_H$	Fast External Interrupt 2 Control Register	0000_H
$FF98_H$	USBEPIC	SFR-b	CB_H	USB Endpoint 0 and 5-15 Interrupt Control Register	0000_H
$FF9A_H$	USBIC	SFR-b	CD_H	USB Interrupt Control Register	0000_H
$FF9C_H$	USBEP1IC	SFR-b	CE_H	USB Endpoint 1 Interrupt Control Register	0000_H
$FF9E_H$	T3IC	SFR-b	CF_H	GPT1 Timer 3 Interrupt Control Register	0000_H
$FFA6_H$					
$FFA8_H$	CLISNC	SFR-b	$D4_H$	The channel link interrupt subnode register	0000_H
$FFAA_H$	FOCON	SFR-b	$D5_H$	Frequency Output Control Register	0000_H
$FFAC_H$	TFR	SFR-b	$D6_H$	Trap Flag Register	0000_H
$FFAE_H$	WDTCON	SFR-b	$D7_H$	Watchdog Timer Control Register	$003C_H$
$FFB0_H$	S0CON	SFR-b	$D8_H$	Serial Channel 0 Control Register	0000_H
$FFB2_H$	SSCCON	SFR-b	$D9_H$	SSC Control Register	0000_H
$FFBA_H$	S0CLC	SFR-b	DD_H	ASC Clock Control Register	0000_H
$FFC0_H$	P2	SFR-b	$E0_H$	Port 2 Register	0000_H
$FFC2_H$	DP2	SFR-b	$E1_H$	Port 2 Direction Control Register	0000_H
$FFC4_H$	P3	SFR-b	$E2_H$	Port 3 Register	0000_H
$FFC6_H$	DP3	SFR-b	$E3_H$	Port 3 Direction Control Register	0000_H
$FFC8_H$	P4	SFR-b	$E4_H$	Port 4 Register (8 bits)	00_H
$FFCA_H$	DP4	SFR-b	$E5_H$	Port 4 Direction Control Register	00_H
$FFCC_H$	P6	SFR-b	$E6_H$	Port 6 Register (8 bits)	00_H
$FFCE_H$	DP6	SFR-b	$E7_H$	Port 6 Direction Control Register	00_H
$FFD0_H$	P7	SFR-b	$E8_H$	Port 7 Register (8 bits)	00_H
$FFD2_H$	DP7	SFR-b	$E9_H$	Port 7 Direction Control Register	00_H

[1] When the OCDS module is active, this register is not reset by an INCA-D reset

25.5 DSP Register Address Summary

00h	REV	Revision	422
01h	CCTL	Chip Control	422
02h	INTM	Mask Register	422
03h	AFECTL	Analog Front End Interface Control	423
04h	IFS1	Interface Select 1	423
05h	IFG1	Interface Gain 1	424
06h	IFG2	Interface Gain 2	424
0Bh	CDICHN1	CDI Channel 1	425
0Ch	IFS3	Interface Select 3	426
0Dh	CDICHN2	CDI Channel 2	427
0Eh	IFS4	Interface Select 4	428
0Fh	IFG5	Interface Gain 5	429
10h	UA	Universal Attenuator	429
11h	DGCTL	DTMF Generator Control	430
12h	DGF1	DTMF Generator Frequency 1	431
13h	DGF2	DTMF Generator Frequency 2	431
14h	DGL	DTMF Generator Level	432
15h	DGATT	DTMF Generator Attenuation	432
1Eh	IFS5	Interface Select 5	433
1Fh	IFG6	Interface Gain 6	433
20h	SUMCTL	Universal Summation Point Control	434
21h	SUMATT	Universal Aummation Input Attenuation	434
25h	LECCTL	Line Echo Cancellation Control	435
26h	LECLEV	Minimal Signal Level for Line Echo Cancellation	436
27h	LECATT	Externally Provided Attenuation	436
28h	LECMGN	Margin for Double Talk Detection	437
29h	DDCTL	DTMF Detector Control	438
2Ah	DDTW	DTMF Detector Signal Twist	439
2Bh	DDLEV	DTMF Detector Minimum Signal Level	439
2Ch	FCFCTL1	Equalizer 1 Control	440
2Dh	FCFCOF1	Equalizer 1 Coefficient Data	441
2Eh	FCFCTL2	Equalizer 2 Control	442
2Fh	FCFCOF2	Equalizer 2 Coefficient Data	443
30h	TGCTL	Tone Generator Control	444
31h	TGTON	Tone Generator Time TON	445
32h	TGTOFF	Tone Generator Time TOFF	446
33h	TGT1	Tone Generator Time T1	446
34h	TGF1	Tone Generator Frequency F1	447
35h	TGG1	Tone Generator Gain G1	447
36h	TGT2	Tone Generator Time T2	448
37h	TGF2	Tone Generator Frequency F2	448
38h	TGG2	Tone Generator Gain G2	449
39h	TGT3	Tone Generator Time T3	449
3Ah	TGF3	Tone Generator Frequency F3	449
3Bh	TGG3	Tone Generator Gain G3	450
3Ch	TGF4	Tone Generator Frequency F4	450
3Dh	TGG4	Tone Generator Gain G4	451
3Eh	TGGO1	Tone Generator Gain Output 1	451
3Fh	TGGO2	Tone Generator Gain Output 2	452
40h	TGGS	Tone Generator Saturate	452

41h . . . . PIDCTL . . . . PIDD Control 453
45h . . . . PDCTL . . . . Peak Detector Control 453
46h . . . . PDDATA . . . . . . Peak Detector Data 454
47h . . . . SPSCTL . . . . Speakerphone State Control 454
50h . . . . ACR . . . . AFE Configuration Register .455
51h . . . . ATCR . . . . AFE Transmit Configuration Register 457
52h . . . . ARCR . . . . AFE Receive Configuration Register 458
58h . . . . AGCCTL . . . . . . AGC Control. 459
59h . . . . AGCATT . . . . . . Automatic Gain Control Attenuation 459
5Ah . . . . AGC1 . . . . Automatic Gain Control 1. 460
5Bh . . . . AGC2 . . . . Automatic Gain Control 2. 461
5Ch . . . . AGC3 . . . . Automatic Gain Control 3. 461
5Dh . . . . AGC4 . . . . Automatic Gain Control 4. 463
5Eh . . . . AGC5 . . . . Automatic Gain Control 5. 463
60h . . . . SCTL . . . . Speakerphone Control. 465
62h . . . . SSC1 . . . . Speakerphone Source 1 467
63h . . . . SSRC2 . . . . Speakerphone Source 2 467
64h . . . . SSDX1 . . . . Speech Detector (Transmit) 1 468
65h . . . . SSDX2 . . . . Speech Detector (Transmit) 2 469
66h . . . . SSDX3 . . . . Speech Detector (Transmit) 3 470
67h . . . . SSDX4 . . . . Speech Detector (Transmit) 4 471
68h . . . . SSDR1 . . . . Speech Detector (Receive) 1. 472
69h . . . . SSDR2 . . . . Speech Detector (Receive) 2. 473
6Ah . . . . SSDR3 . . . . Speech Detector (Receive) 3. 474
6Bh . . . . SSDR4 . . . . Speech Detector (Receive) 4. 475
6Ch . . . . SSCAS1 . . . . . . Speech Comparator (Acoustic Side) 1 476
6Dh . . . . SSCAS2 . . . . . . Speech Comparator (Acoustic Side) 2 477
6Eh . . . . SSCAS3 . . . . . . Speech Comparator (Acoustic Side) 3 478
6Fh . . . . SSCLS1 . . . . . . Speech Comparator (Line Side) 1 479
70h . . . . SSCLS2 . . . . . . Speech Comparator (Line Side) 2 480
71h . . . . SSCLS3 . . . . . . Speech Comparator (Line Side) 3 481
72h . . . . SATT1 . . . . Attenuation Unit 1 482
73h . . . . SATT2 . . . . Attenuation Unit 2 483
74h . . . . SAGX1 . . . . Automatic Gain Control (Transmit) 1 484
75h . . . . SAGX2 . . . . Automatic Gain Control (Transmit) 2 485
76h . . . . SAGX3 . . . . Automatic Gain Control (Transmit) 3 486
77h . . . . SAGX4 . . . . Automatic Gain Control (Transmit) 4 487
78h . . . . SAGX5 . . . . Automatic Gain Control (Transmit) 5 488
79h . . . . SAGR1 . . . . Automatic Gain Control (Receive) 1 489
7Ah . . . . SAGR2 . . . . Automatic Gain Control (Receive) 2 490
7Bh . . . . SAGR3 . . . . Automatic Gain Control (Receive) 3 491
7Ch . . . . SAGR4 . . . . Automatic Gain Control (Receive) 4 492
7Dh . . . . SAGR5 . . . . Automatic Gain Control (Receive) 5 493
7Eh . . . . SLGA . . . . Line Gain 494
7Fh . . . . SAELEN . . . . . . Acoustic Echo Cancellation Length 495
80h . . . . SAEAW . . . . Acoustic Echo Cancellation Adaption Window 496
81h . . . . SAEL . . . . Acoustic Echo Cancellation Reported Att. Limit . . .497
82h . . . . SAEDTR . . . . . . Acoustic Echo Cancellation Double Talk Red. 498
83h . . . . SAEDTL . . . . . . Acoustic Echo Cancellation Double Talk Limit 499
84h . . . . SAEDTI . . . . Acoustic Echo Cancellation Double Talk Incr. 500
85h . . . . SAEDTD . . . . . . Acoustic Echo Cancellation Double Talk Decr. 501
86h . . . . SAEWFL . . . . . . Wiener Filter Limit Attenuation. 502

87h . . . . SNRATT . . . . . . Noise Reduction Attenuation. 503
88h . . . . SNRLNL . . . . . . Noise Reduction Lower Noise Limit. 504
89h . . . . SNRUNL . . . . . . Noise Reduction Upper Noise Limit. 505
A6h . . . . NRCTL. . . . . Noise Reduction Control 506
A7h . . . . NRATT . . . . Noise Reduction Attenuation. 507

26 Electrical Characteristics

26.1 Electrical Characteristics (general)

Table 26-1 Absolute Maximum Ratings

Parameter	Symbol	Limit Values		Unit
		min.	max.	
Storage temperature	T_{STG}	– 55	150	°C
Voltage on VDD pins	V_{DD}	– 0.3	4.6	V
Voltage range for input pins with exception of BGREF, MINn and MIPn	V_{in}	(-0.3 ...Vdd) + 3.3 (max. < 5.5)		V
Voltage range for ouput pins with exception of LSP, LSN, HOPn and HONn	V_{out}	(-0.3 ...Vdd)+ 3.3 (max. < 3.9)		V
Voltage with respect to ground on pins BGREF, MINn, MIPn, LSP, LSN, HOPn and HONn	V_S	0	3.6	V

Note: Maximum ratings are stress ratings only, and functional operation and reliability under conditions beyond those defined in the "recommended operating conditions" is not guaranteed. Stresses above the maximum ratings are likely to cause permant damage.

26.2 Recommended Operating Conditions

Parameter	Symbol	Limit Values		Unit	Test Condition
		min.	max.		
Ambient temperature	T_A	0	70	°C	
Supply voltage	V_{DD}	3.135	3.6	V	
Ground	V_{SS}	0	0	V	

Note: In the operating range, the functions given in the circuit description are fulfilled.

26.3 DC-Characteristics

Explanation of parameters

V_{IH} and V_{IL} are input voltages that are required for a low-to-high or a high-to-low transition of an input. These thresholds depend on supply voltage and temperature variations, therefore a range is given. Additionally the absolute maximum ratings have to be considered for the input voltage. All inputs have Schmitt-Trigger properties.

V_{OL} is the saturation voltage of the open-drain transistor in the on-state. V_{OL} depends on the current drawn from the output.

I_{PU} and I_{PU} are of interest if the internal pull-up/down transistors are enabled. The current drawn by the pull transistors for a certain voltage V at the port pin is stated. This is typically translated into an equivalent resistance. For Port 6 and 7 the pull-up/down resitance is approx. 33kΩ.

Table 26-2 General DC-Characteristics

V_{DD} **= 3.135V-3.6V**, VSS = 0 V; T_A = 0 to 70 °C

Parameter	Symbol	Limit Values			Unit	Test Condition
		min.	typ.	max.		
Input low voltage for digital pins[1)]	V_{IL}	0		0.9	V	
Input high voltage for digital pins[1)]	V_{IH}	2.0		5.0	V	
Output high voltage for P0, P1, P2.0-P2.2, P3, P4, P6, P7, DD, DU, FSC, DCL,TDO, $\overline{\text{BRKOUT}}$ ALE, $\overline{\text{RD}}$ and $\overline{\text{WR}}$	V_{OH}	V_{DD}-0.6			V	I_{OL} = -3.5 mA
Output high voltage for P0, P1, P2.0-P2.2, P3, P4, P6, P7, DD, DU, FSC, DCL,TDO, $\overline{\text{BRKOUT}}$ ALE, $\overline{\text{RD}}$ and $\overline{\text{WR}}$	V_{OH}	V_{DD}-0.9			V	I_{OL} = -5.5 mA
Output high voltage for P2.3-P2.13	V_{OH}	V_{DD}-0.6			V	I_{OL} = -10.2 mA

Table 26-2 General DC-Characteristics

V_{DD} **= 3.135V-3.6V**, VSS = 0 V; T_A = 0 to 70 °C

Parameter	Symbol	Limit Values			Unit	Test Condition
		min.	typ.	max.		
Output low voltage for P0, P1, P2.0-P2.2, P3, P4, DD, DU, FSC, DCL, TDO, $\overline{\text{BRKOUT}}$, ALE, $\overline{\text{RD}}$ and $\overline{\text{WR}}$	V_{OL}			0.45	V	I_{OL} = 3 mA
Output low voltage for P0, P1, P2.0-P2.2, P3, P4, DD, DU, FSC, DCL, TDO, $\overline{\text{BRKOUT}}$, ALE, $\overline{\text{RD}}$ and $\overline{\text{WR}}$	V_{OL}			0.6	V	I_{OL} = 6 mA
Output low voltage for P2.3-P2.13	V_{OL}			0.45	V	I_{OL} = 11.2 mA
Output low voltage for P6 and P7 (keyscanner)	V_{OL}	0.005	0.007	0.01	V	160 µA max. Pull-up current
Internal pull-up current for P0, P1 and P4	I_{PU}	12	20	31	µA	
Internal pull-down current for P0, P1and P4	I_{PD}	-10	-21	-40	µA	
Internal pull-up current for P2, P3, P6, and P7	I_{PU}	62	103	157	µA	
Internal pull-down current for RSTIN, P2, P3, P6 and P7	I_{PD}	-57	-110	-189	µA	
Avg. power supply	I_{CC} (AV)		110		mA	V_{DD} = 3.3 V, T_A = 25 °C: Simple phone connection, Handset mode

Table 26-2 General DC-Characteristics

V_{DD} **= 3.135V-3.6V**, VSS = 0 V; T_A = 0 to 70 °C

Parameter	**Symbol**	**Limit Values**			**Unit**	**Test Condition**
		min.	**typ.**	**max.**		
Avg. power supply current during idle mode	I_{ID} (AV)		8		mA	V_{DD} = 3.3 V, T_A = 25 °C: All peripherals stopped
Avg. power supply current during power down mode	I_{PD} (AV)		7		mA	V_{DD} = 3.3 V T_A = 25 °C
Input leakage current Output leakage current (all pins except LIa, LIb,XTAL1,2 BGREF, VREF)	I_{LI} I_{LO}	-1 -1		1 1	µA µA	0V< V_{IN}<V_{DD} 0V< V_{OUT}<V_{DD}

[1] P0, P1, P2, P3, P4, P6, P7, TDI, TMS, $\overline{\text{RSTIN}}$, TCK, $\overline{\text{TRSTN}}$, $\overline{\text{BRKIN}}$

26.4 Capacities

V_{DD} **= 3.135V - 3.6V** , VSS = 0 V; T_A = 0 to 70 °C; f_C = 1 MHz; unmeasured pins grounded.

Table 26-3 Capacitances

Parameter	Symbol	Limit Values		Unit	Remarks
		min.	max.		
Input Capacitance I/O Capacitance	C_{IN} $C_{I/O}$		7 7	pF pF	All pins except LIa and LIb
Output Capacitance against V_{SS}	C_{OUT}		25	pF	pins LIa, LIb
Load Capacitance	C_L		60	pF	pins XTAL1,2

26.5 Oscillator Specification

Recommended Oscillator Circuit

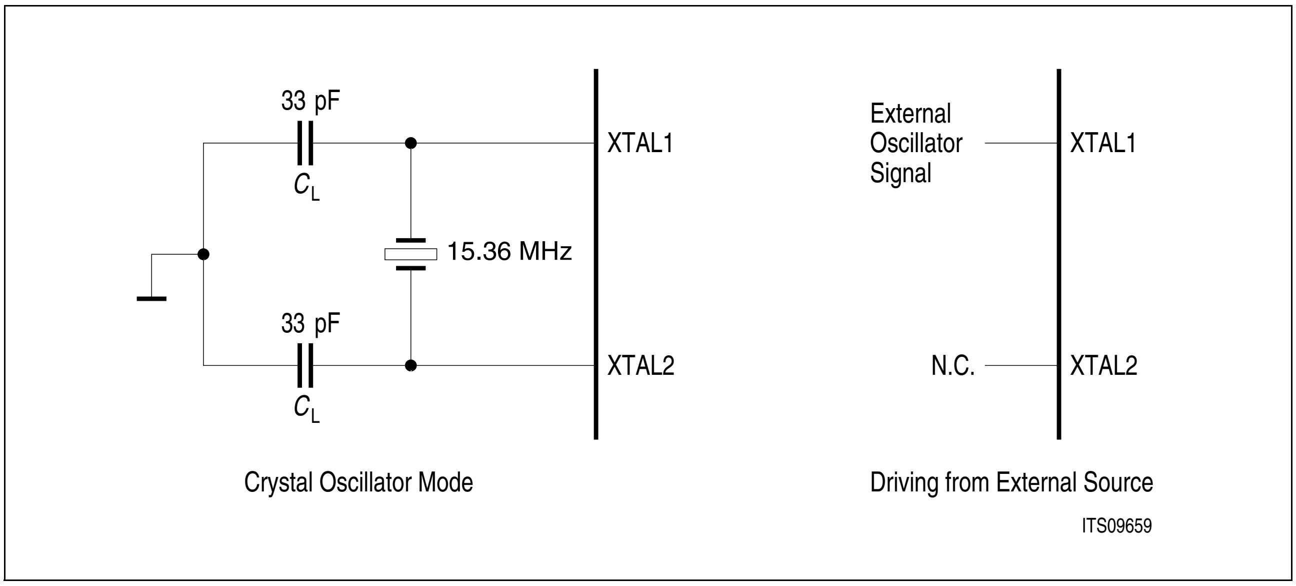


Figure 26-1 Oscillator Circuit

Crystal Specification

Parameter	Symbol	Limit Values	Unit
Frequency	f	15.36	MHz
Frequency calibration tolerance		max. 100	ppm

Parameter	Symbol	Limit Values	Unit
Load capacitance	C_L	max. 50	pF
Oscillator mode		fundamental	

Note: The load capacitance C_L depends on the recommendation of the crystal specification. Typical values for C_L are 22...33 pF.

External Clock Drive XTAL1

V_{DD} = 3.3 V (-5% / +10%); V_{SS} = 0 V
T_A = -40 to +85 °C

Table 26-4 External Clock Drive XTAL 1

Parameter	Symbol	PLL 15.36-MHz input			Unit
		min	typical	max	
Oscillator period	t_{OSCSR}	[1]	65.10	–	ns
High time	t_{1SR}	21.7		43.4	ns
Low time	t_{2SR}	21.7		43.4	ns
Rise time	t_{3SR}	–	<3.5		ns
Fall time	t_{4SR}	–	<3.5		ns

[1] The minimum and maximum oscillator periods for PLL operation depend on the selected CPU clock generation mode.

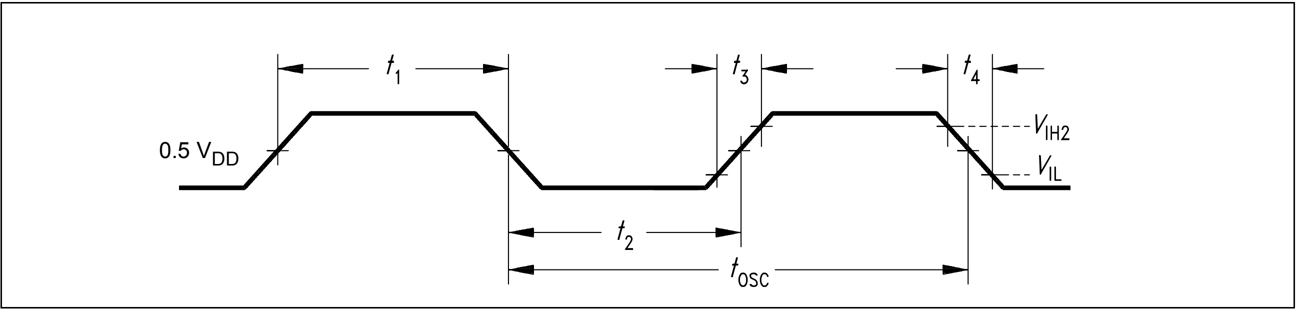


Figure 26-2 External Clock Drive XTAL1

26.6 AC Characteristics

V_{DD} = **3.135V-3.6V**), VSS = 0 V; T_A = 0 to 70 °C

Inputs are driven to 2.4 V for a logical "1" and to 0.45 V for a logical "0". Timing measurements are made at 2.0 V for a logical "1" and 0.8 V for a logical "0". The AC testing input/output waveforms are shown in **figure 26-3.**

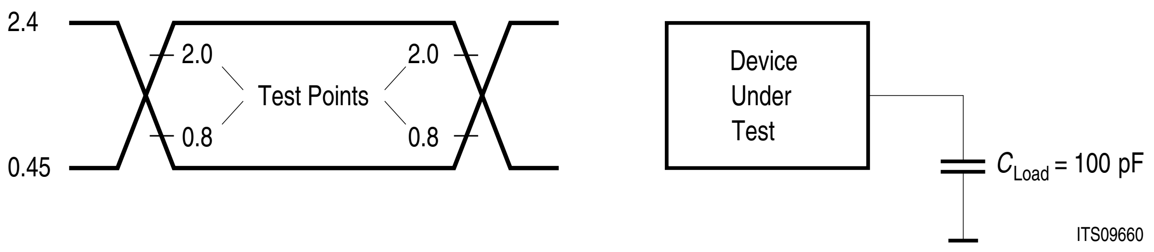


Figure 26-3 Input/Output Waveform for AC Tests

Memory Cycle Variables

The timing tables below use three variables which are derived from the BUSCONx registers and represent the special characteristics of the programmed memory cycle. The following table describes, how these variables are to be computed.

Table 26-5 Memory Cycle Variables

Description	Symbol	Values
ALE Extension	t_A	TCL * <ALECTL>
Memory Cycle Time Waitstates	t_C	2TCL * (15 - <MCTC>)
Memory Tristate Time	t_F	2TCL * (1 - <MTTC>)

26.6.1 AC Characteristics of the External Bus Interface

Multiplexed Bus

ALE cycle time = 6 TCL + 2t_A + t_C + t_F (93.75 ns at 32 MHz CPU clock without waitstates)

The following timings are valid for a CPU frequency of 32 MHz.

Parameter	Symbol	Max. CPU Clock 32 MHz		Unit
		min.	max.	
ALE high time	t_5	16 + t_A		ns
Address setup to ALE	t_6	10 + t_A		ns
Address hold after ALE	t_7	36 + t_A		ns
ALE falling edge to $\overline{RD}$, $\overline{WR}$ (with RW-delay)	t_8	17 + t_A		ns
ALE falling edge to $\overline{RD}$, $\overline{WR}$ (no RW-delay)	t_9	0 + t_A		ns
Address float after $\overline{RD}$, $\overline{WR}$ (with RW-delay)	t_{10}		12	ns
Address float after $\overline{RD}$, $\overline{WR}$ (no RW-delay)	t_{11}		27	ns
$\overline{RD}$, $\overline{WR}$ low time (with RW-delay)	t_{12}	62 + t_C		ns
$\overline{RD}$, $\overline{WR}$ low time (no RW-delay)	t_{13}	77 + t_C		ns
$\overline{RD}$ to valid data in (no RW-delay)	t_{14}		33 + t_C	ns
$\overline{RD}$ to valid data in (with RW-delay)	t_{15}		47+ t_C	ns
ALE low to valid data in	t_{16}		49+ t_C	ns
Address to valid data in	t_{17}		15 + 2t_A + t_C	ns
Data hold after $\overline{RD}$ rising edge	t_{18}	29 + t_F		ns
Data float after $\overline{RD}$	t_{19}		62 + t_F	ns
Data valid to $\overline{WR}$	t_{22}	54.63 + t_C	56.56 + t_C	ns
Data hold after $\overline{WR}$	t_{23}	66+ t_F		ns
ALE rising edge after $\overline{RD}$, $\overline{WR}$	t_{25}	70 + t_F		ns

Multiplexed Bus (cont'd)
ALE cycle time = 6 TCL + $2t_A + t_C + t_F$ (93.75 ns at 32 MHz CPU clock without waitstates)

The following timings are valid for a CPU frequency of 32 MHz.

Parameter	Symbol	Max. CPU Clock 32 MHz		Unit
		min.	**max.**	
Address hold after $\overline{\text{RD}}$, $\overline{\text{WR}}$	t_{27}	$63 + t_F$		ns
ALE falling edge to $\overline{\text{CS}}$ [1]	t_{38}	$-0.3 - t_A$	$1.20 - t_A$	ns
$\overline{\text{CS}}$ low to Valid Data In [1]	t_{39}		$85+ t_C + 2t_A$	ns
$\overline{\text{CS}}$ hold after $\overline{\text{RD}}$, $\overline{\text{WR}}$ [1]	t_{40}	$65 + t_F$		ns

[1] These parameters refer to the latched chip select signals ($\overline{\text{CSxL}}$). The early chip select signals ($\overline{\text{CSxE}}$) are specified together with the address and signal $\overline{\text{BHE}}$ (see figures below).

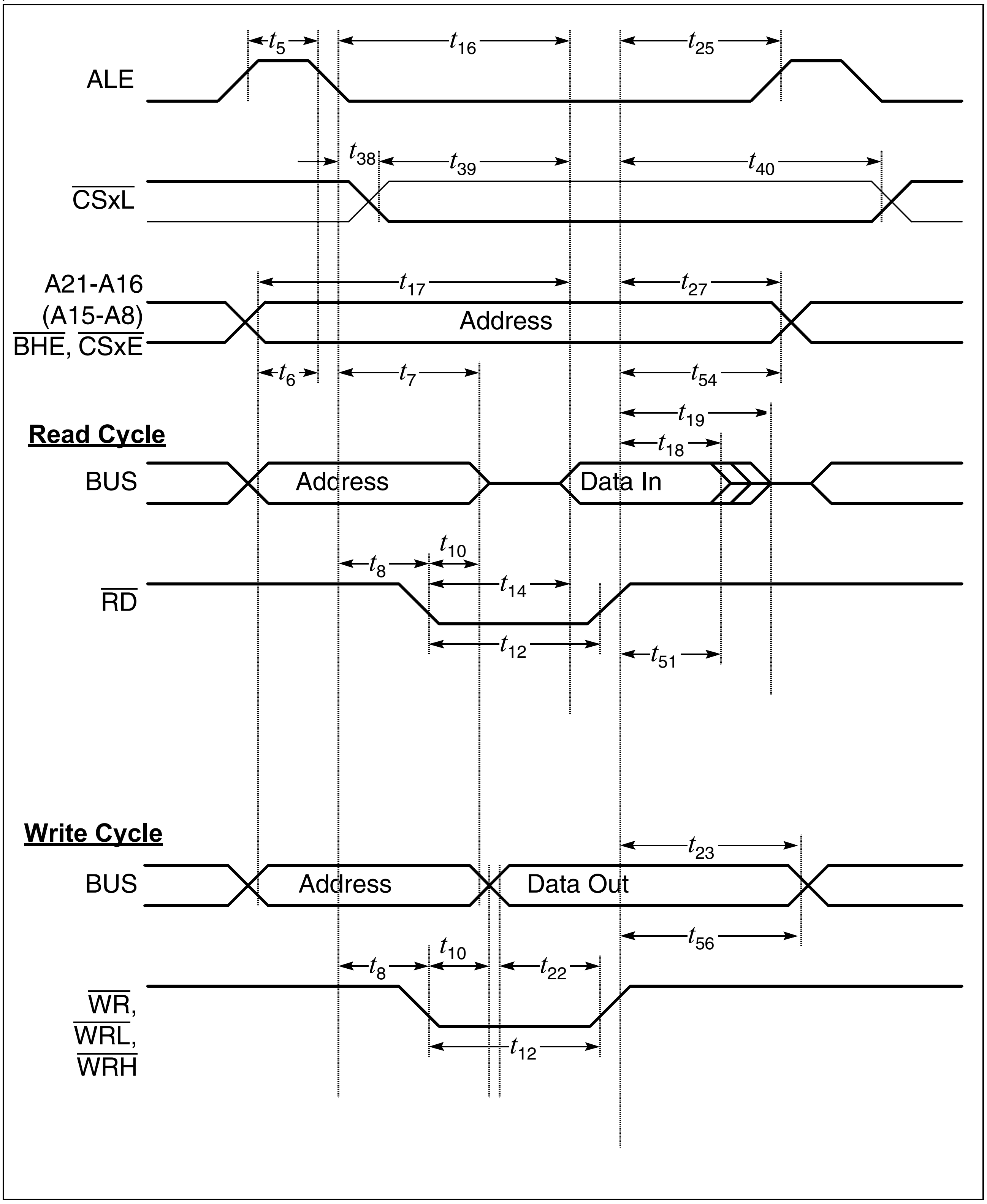


Figure 26-4 External Memory Cycle: Multiplexed Bus, With Read/Write Delay, Normal ALE

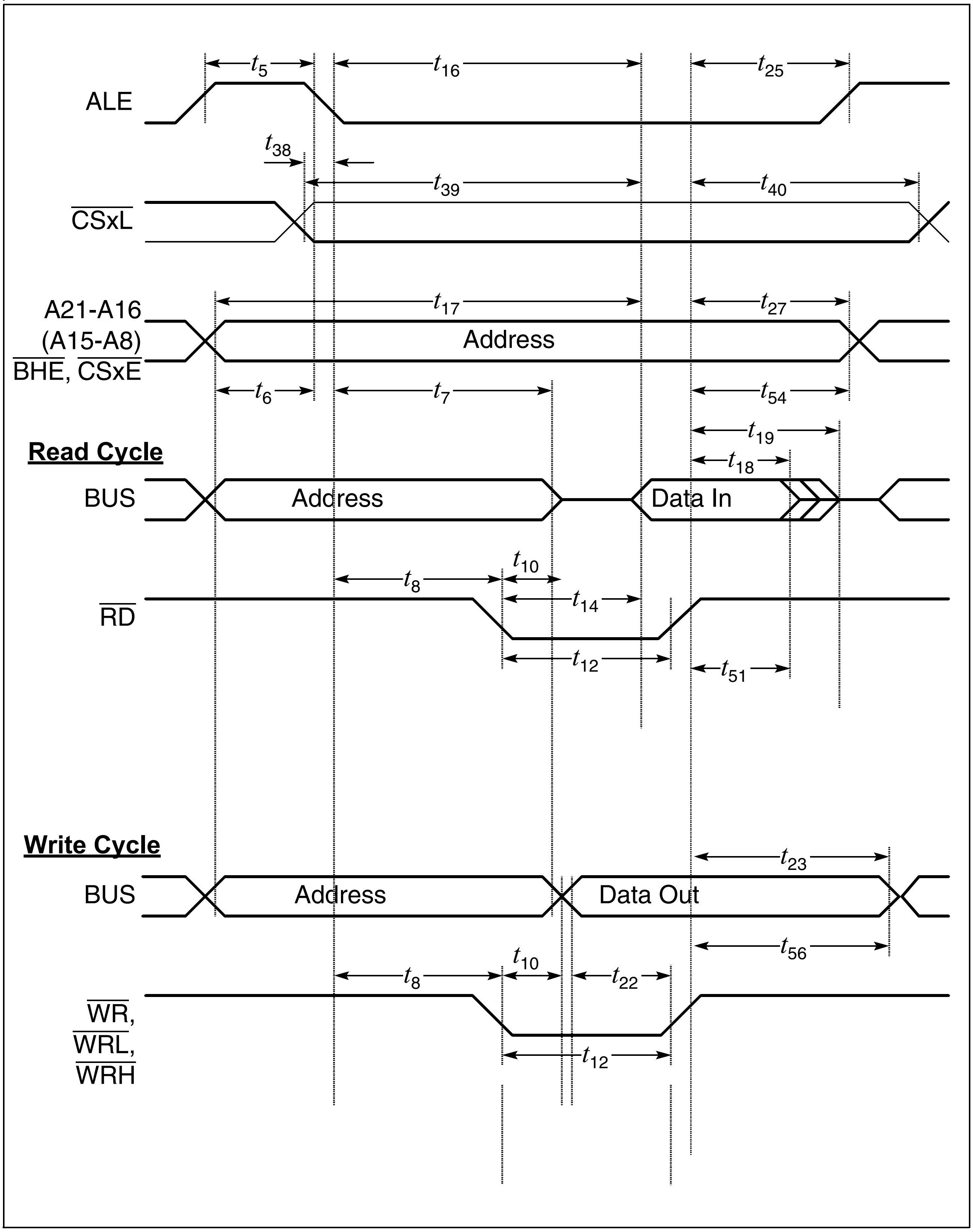


**Figure 26-5 External Memory Cycle:
Multiplexed Bus, With Read/Write Delay, Extended ALE**

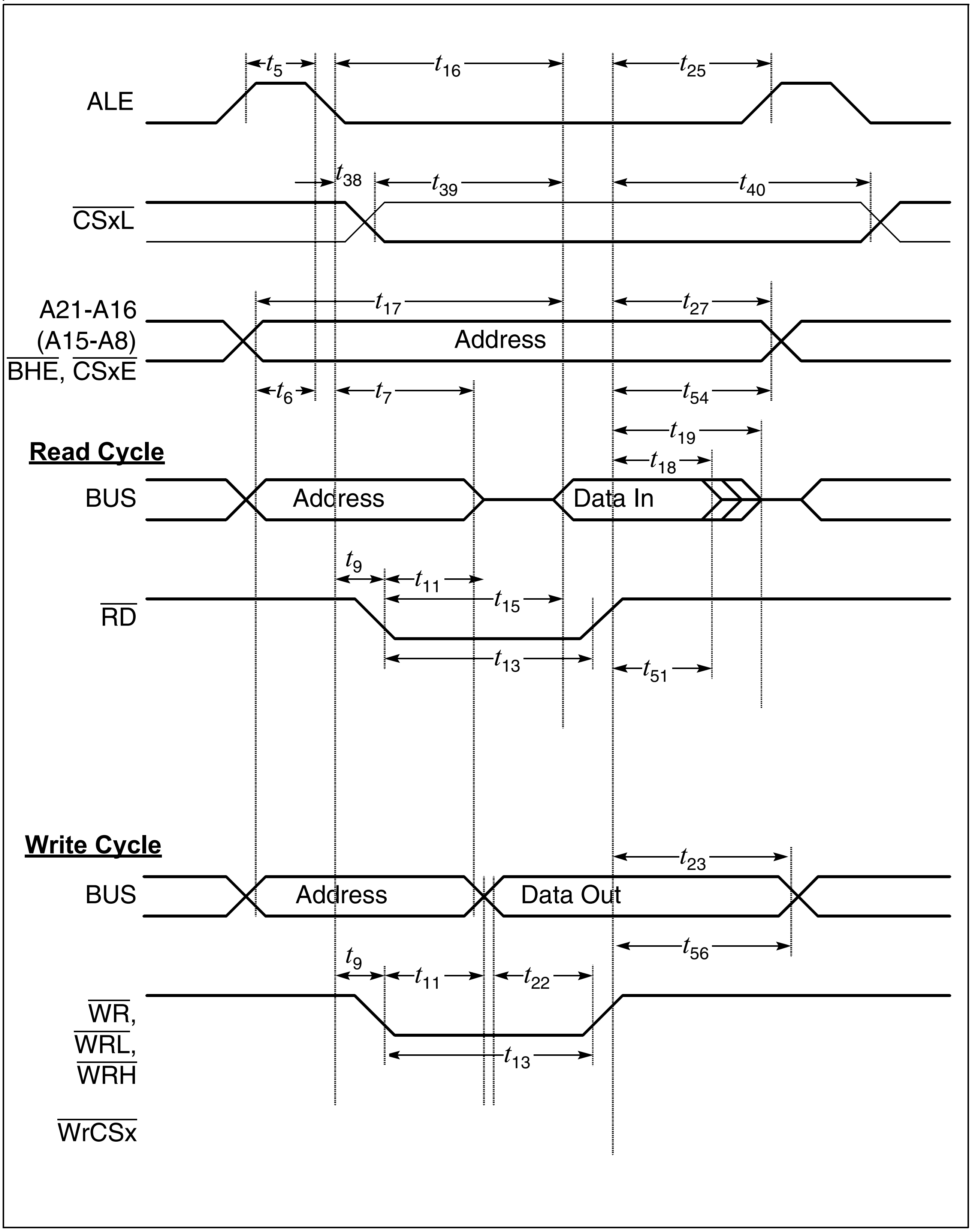


**Figure 26-6 External Memory Cycle:
Multiplexed Bus, No Read/Write Delay, Normal ALE**

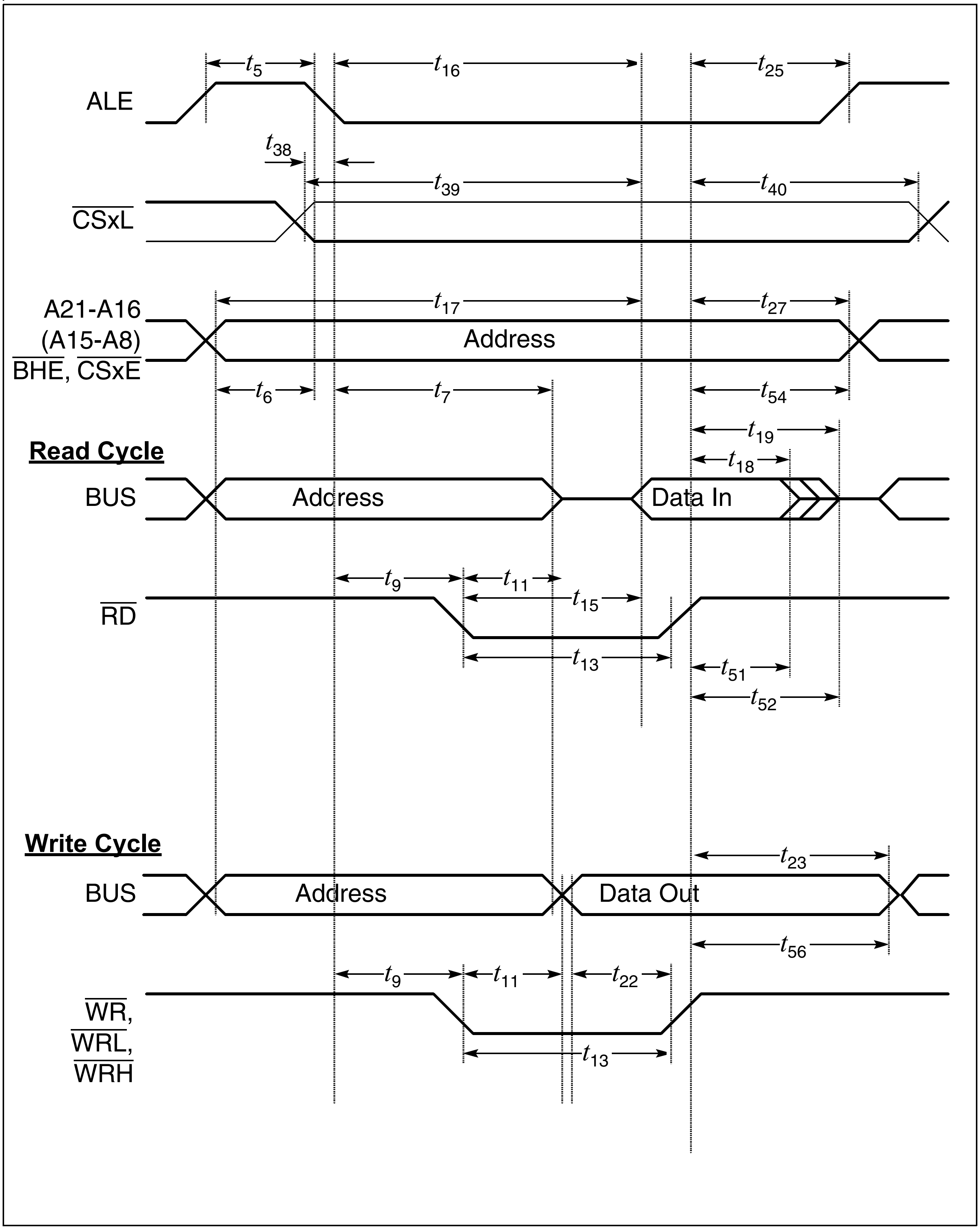


Figure 26-7 External Memory Cycle: Multiplexed Bus, No Read/Write Delay, Extended ALE

Demultiplexed Bus

ALE cycle time = 4 TCL + $2t_A + t_C + t_F$ (62.5 ns at 32 MHz CPU clock without waitstates)

The following timings are valid for a CPU frequency of 32 MHz.

Parameter	Symbol	Max. CPU Clock 32 MHz		Unit
		min.	max.	
ALE high time	t_5	$29 + t_A$		ns
Address setup to ALE	t_6	$16 + t_A$		ns
ALE falling edge to $\overline{\text{RD}}$, $\overline{\text{WR}}$ (with RW-delay)	t_8	$17 + t_A$		ns
ALE falling edge to $\overline{\text{RD}}$, $\overline{\text{WR}}$ (no RW-delay)	t_9	$2 + t_A$		ns
$\overline{\text{RD}}$, $\overline{\text{WR}}$ low time (with RW-delay)	t_{12}	$62 + t_C$		ns
$\overline{\text{RD}}$, $\overline{\text{WR}}$ low time (no RW-delay)	t_{13}	$79 + t_C$		ns
$\overline{\text{RD}}$ to valid data in (with RW-delay)	t_{14}		$39 + t_C$	ns
$\overline{\text{RD}}$ to valid data in (no RW-delay)	t_{15}		$53 + t_C$	ns
ALE low to valid data in	t_{16}		$54 + t_A + t_C$	ns
Address to valid data in	t_{17}		$67 + 2t_A + t_C$	ns
Data hold after $\overline{\text{RD}}$ rising edge	t_{18}	$112 + t_F$		ns
Data valid to $\overline{\text{WR}}$	t_{22}	$63 + t_C$		ns
Data hold after $\overline{\text{WR}}$	t_{24}	$51 + t_F$		ns
ALE rising edge after $\overline{\text{RD}}$, $\overline{\text{WR}}$	t_{26}	$32 + t_F$		ns
Address hold after $\overline{\text{WR}}$ [1)]	t_{28}	$38 + t_F$		ns
ALE falling edge to $\overline{\text{CS}}$ [2)]	t_{38}	$0.5 - t_A$	$1.0 - t_A$	ns
$\overline{\text{CS}}$ low to Valid Data In [2)]	t_{39}		$46 + t_C + 2t_A$	ns
$\overline{\text{CS}}$ hold after $\overline{\text{RD}}$, $\overline{\text{WR}}$ [2)]	t_{41}	$50 + t_F$		ns

1) Read data are latched with the same clock edge that triggers the address change and the rising $\overline{\text{RD}}$ edge. Therefore address changes before the end of $\overline{\text{RD}}$ have no impact on read cycles.

2) These parameters refer to the latched chip select signals ($\overline{\text{CSxL}}$). The early chip select signals ($\overline{\text{CSxE}}$) are specified together with the address and signal $\overline{\text{BHE}}$ (see figures below).

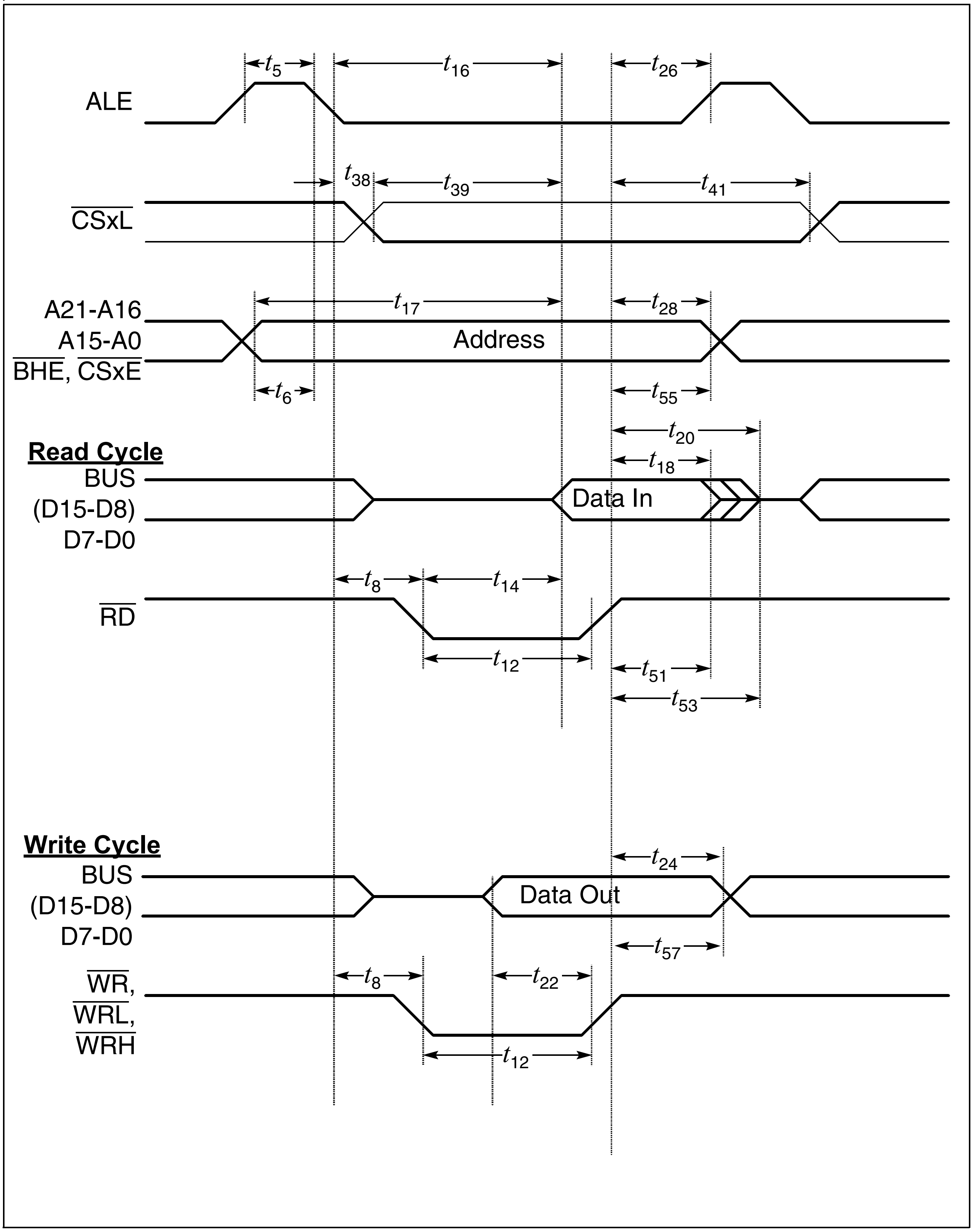


Figure 26-8 External Memory Cycle: Demultiplexed Bus, With Read/Write Delay, Normal ALE

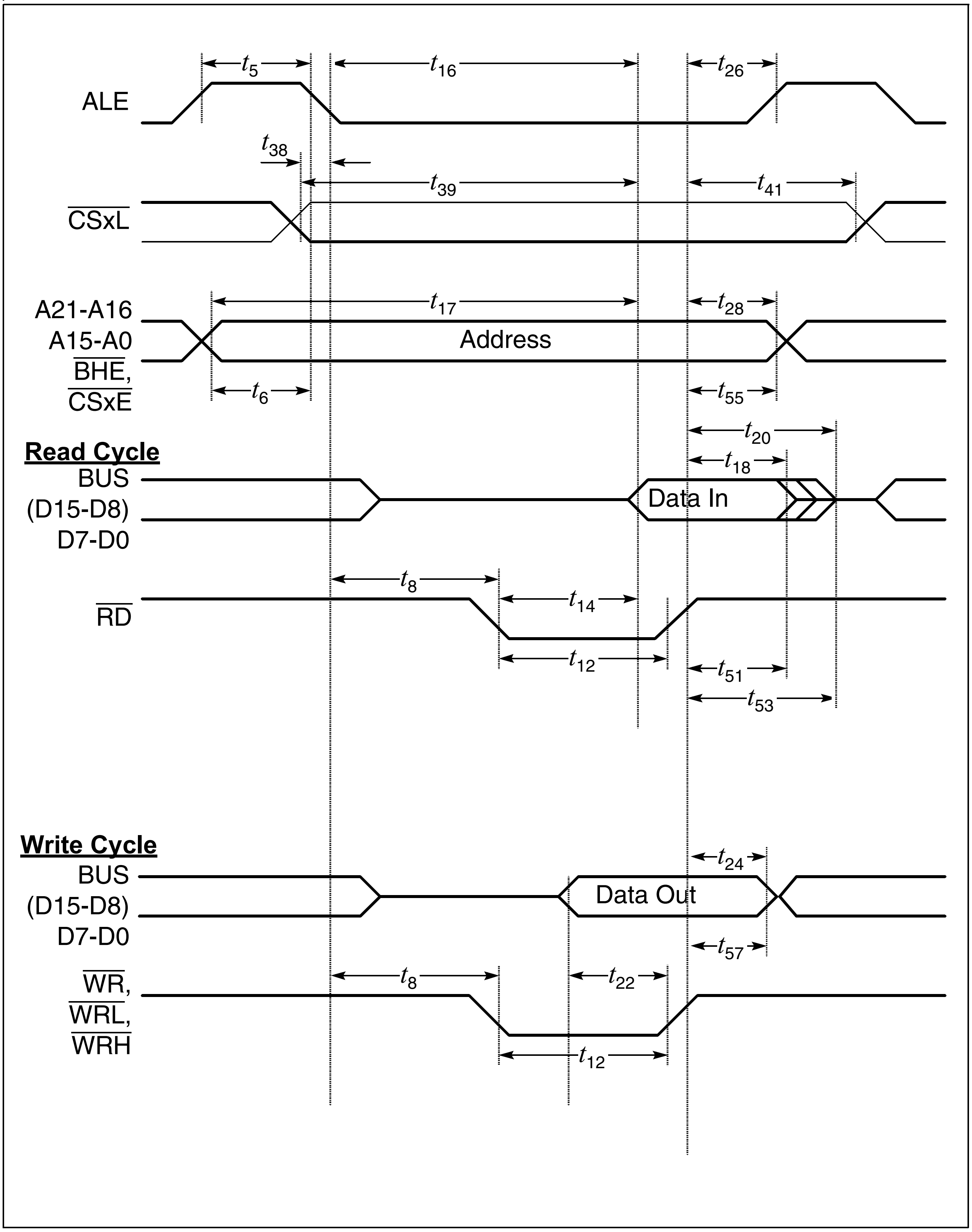


Figure 26-9 External Memory Cycle: Demultiplexed Bus, With Read/Write Delay, Extended ALE

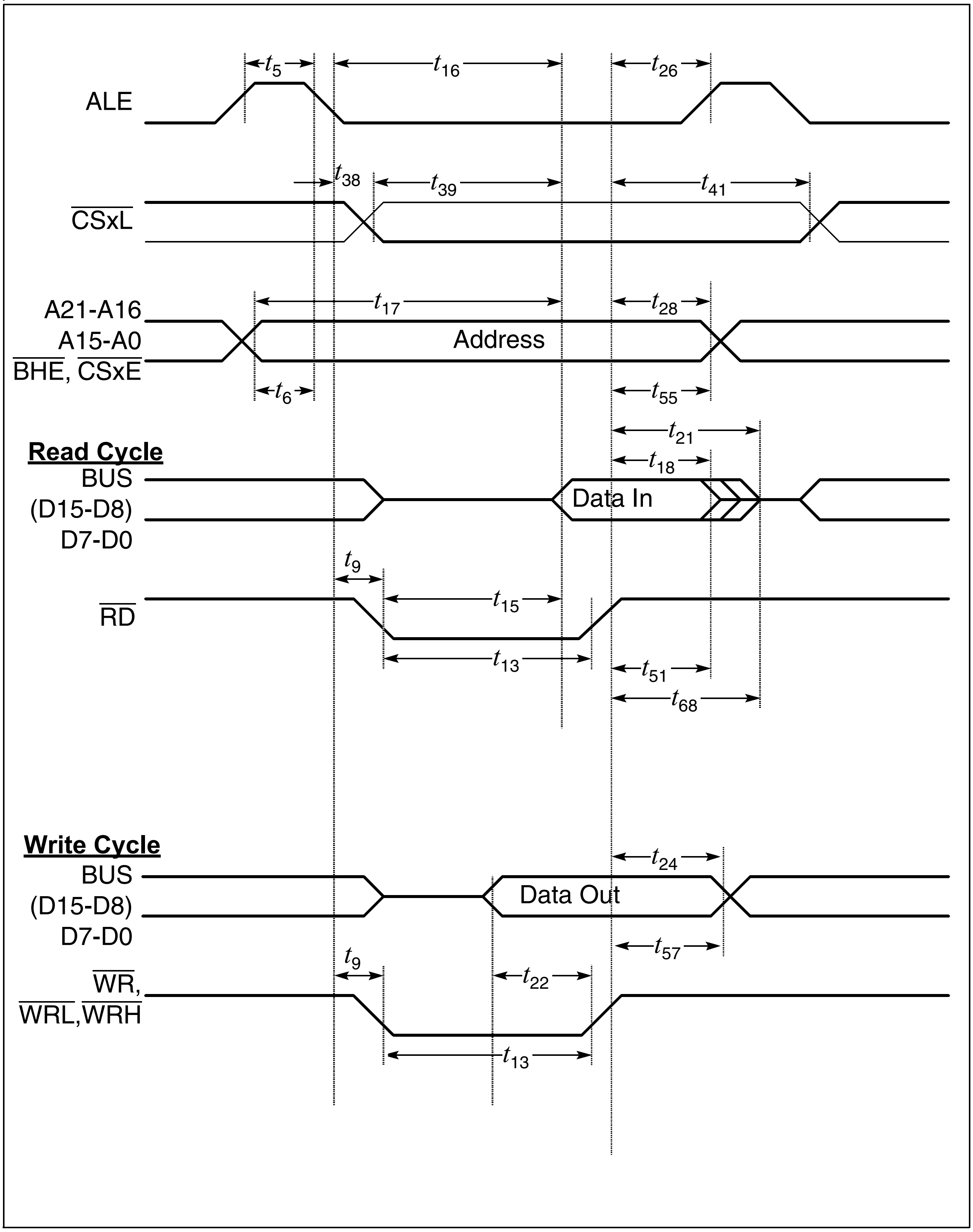


Figure 26-10 External Memory Cycle: Demultiplexed Bus, No Read/Write Delay, Normal ALE

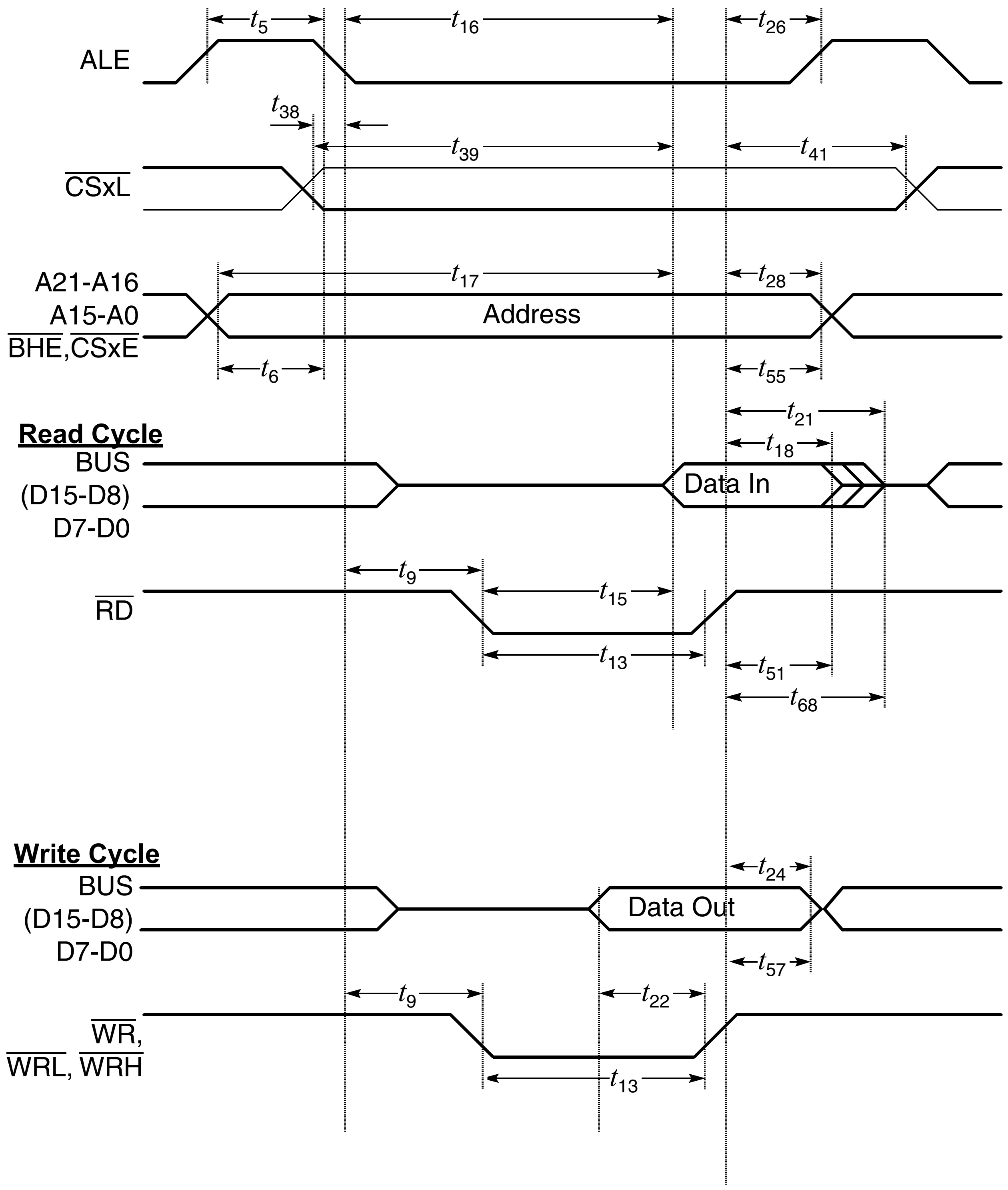


Figure 26-11 External Memory Cycle: Demultiplexed Bus, No Read/Write Delay, Extended ALE

26.6.2 SSC0 and SSC1 Timing

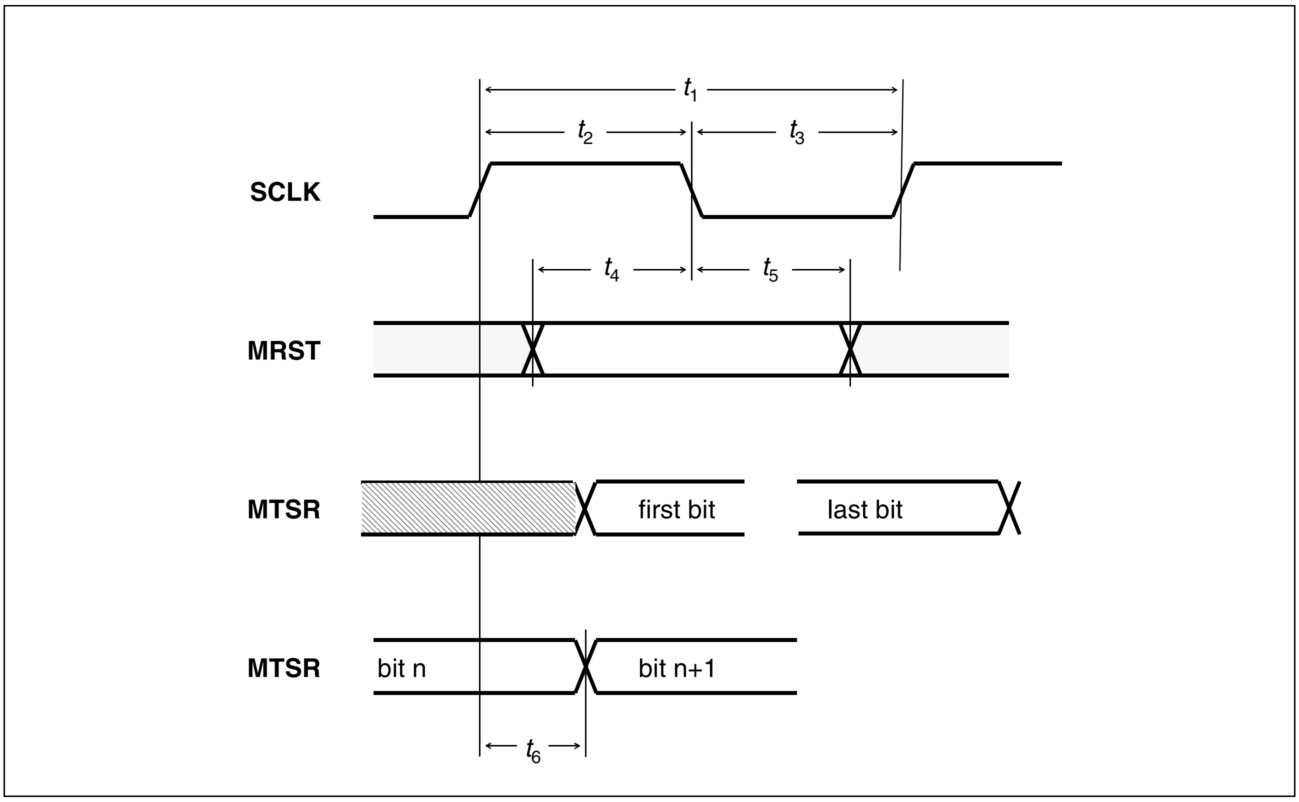


Figure 26-12 SSC0/1 Interface

The timings in the table below have been measured for a CPU frequency of 32 MHz , SSCCON (master) = CB57$_H$ and SSCBR= 0000$_H$ (16 MBaud)

Parameter Serial interface - SSC	Symbol	Min	Typ	Max	Unit
SCLK period	t_1		62.50		ns
SCLK high	t_2		31.25		ns
SCLK low	t_3		31.25		ns
Input data setup	t_4	39			ns
Input data hold	t_5	0			ns
Output data delay from clock	t_6		0		ns

Table 26-6 SSS Timings for Master Mode

The timings in the table below have been measured for a CPU frequency of 32 MHz , SSCCON (slave) = 8F47$_H$ and SSCBR= 0001 $_H$(8 MBaud)

Parameter **Serial interface - SSC**	**Symbol**	**Min**	**Typ**	**Max**	**Unit**
SCLK period	t_1		125,00		ns
SCLK high	t_2		62.50		ns
SCLK low	t_3		62.50		ns
Input data setup	t_4	39			ns
Input data hold	t_5	27			ns
Output data delay from clock	t_6			14	ns

Table 26-7 SSS Timings for Master Mode

26.6.3 ASC Timing Synchronous mode

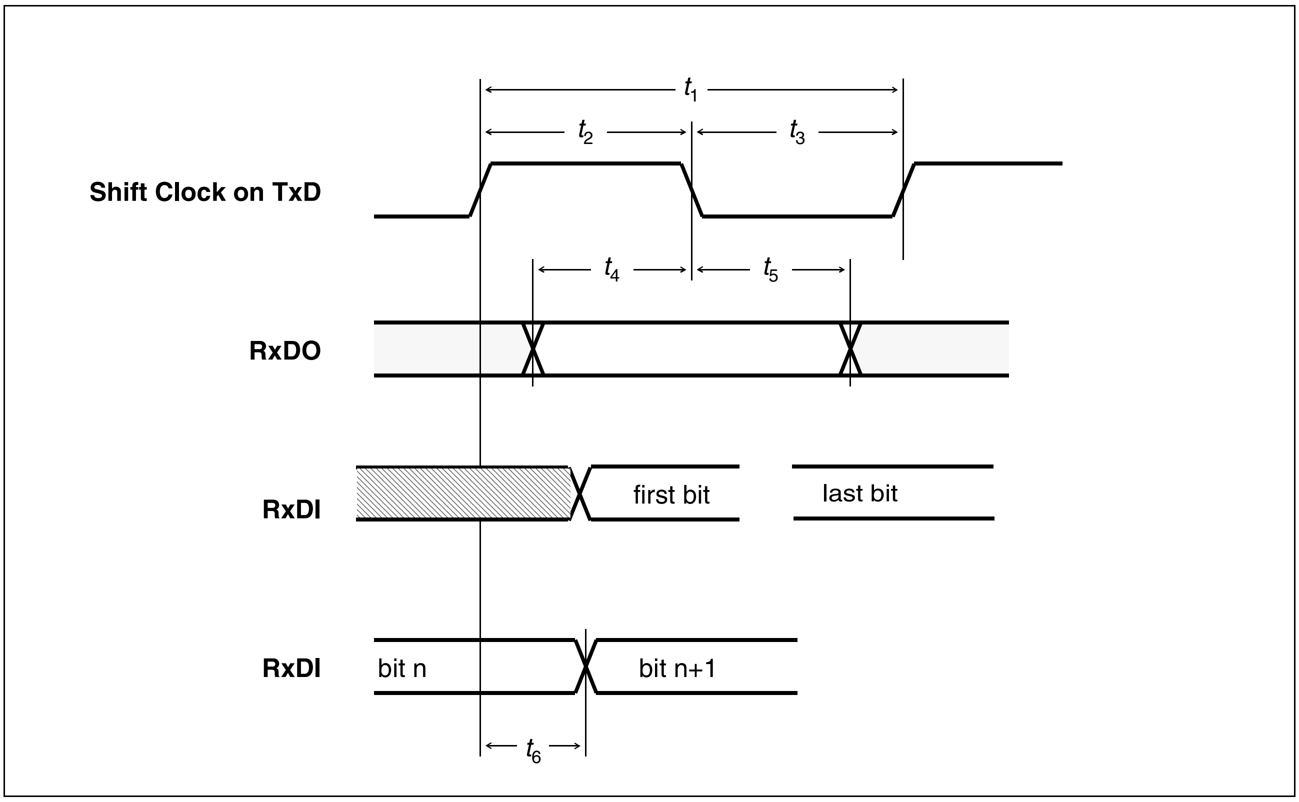


Figure 26-13 ASC Interface

The timings in the table below have been measured for a CPU frequency of 24 MHz, S0CON.BRS = 0, S0BG = 0000_H (3 MBaud), S0CON = 8000_H for receive disabled and S0CON = 8010_H for receive enabled.

Parameter **Serial interface - ASC**	**Symbol**		**Typical values**		**Unit**
Shift clock period	t_1		333.31		ns
Shift Clock high	t_2		166.66		ns
Shift Clock low	t_3		166.66		ns
Input data setup	t_4	87			ns
Input data hold	t_5		0		ns
Output data delay from clock	t_6		0		ns

26.7 IOM-2 Interface Timing

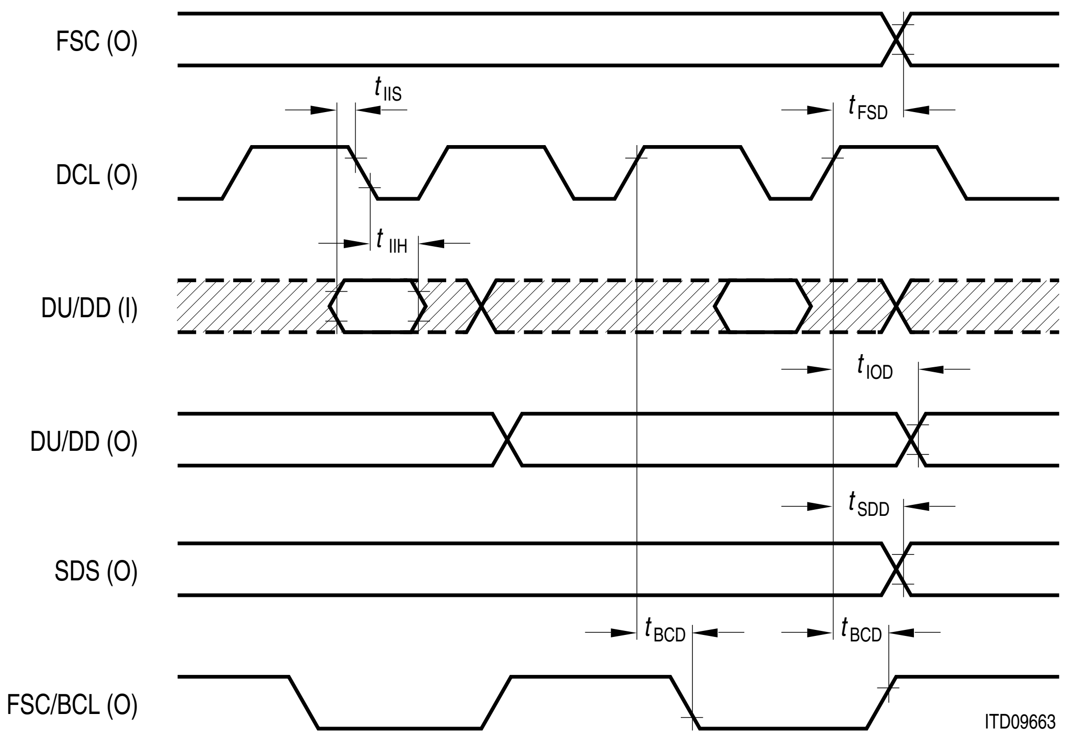


Figure 26-14 IOM-2 Timing

Parameter	Symbol	Limit Values		Unit
		min.	max.	
IOM output data delay	t_{IOD}		100	ns
IOM input data setup	t_{IIS}	20		ns
IOM input data hold	t_{IIH}	20		ns
FSC strobe delay	t_{FSD}	-130	t.b.d.	ns
Strobe signal delay	t_{SDD}		120	ns
BCL / FSC delay	t_{BCD}		100	ns
Frame sync setup	t_{FSS}	50		ns
Frame sync hold	t_{FSH}	30		ns
Frame sync width	t_{FSW}	40		ns

DCL Clock Output Characteristics

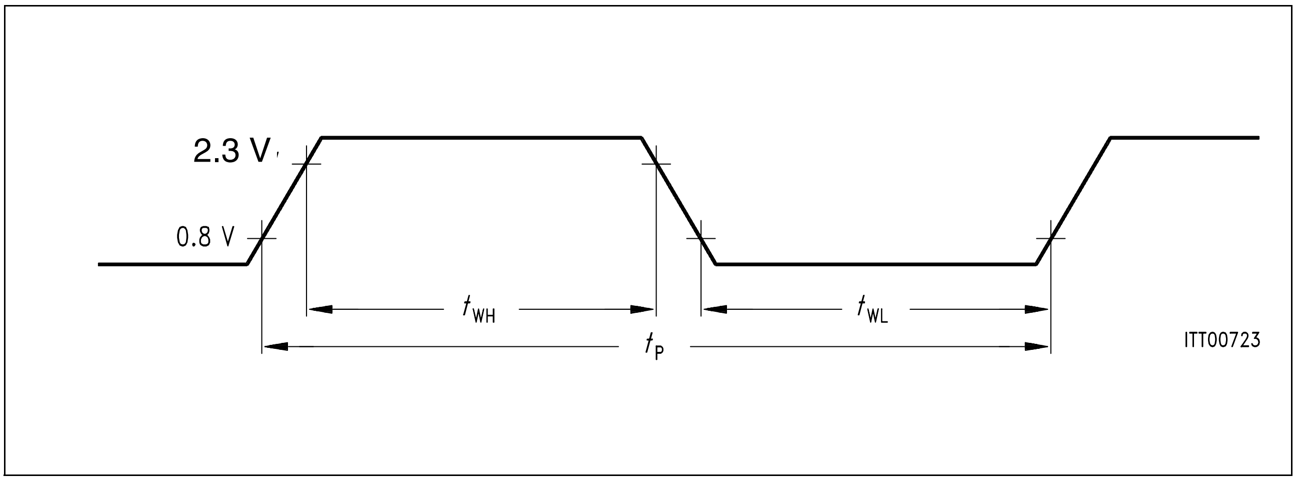


Figure 26-15 Definition of Clock Period and Width

Symbol	Limit Values			Unit	Test Condition
	min.	typ.	max.		
t_P	585	651	717	ns	osc ± 100 ppm
t_{WH}	260	325	391	ns	osc ± 100 ppm
t_{WL}	260	325	391	ns	osc ± 100 ppm

26.8 Electrical Characteristics Transceiver

AC Characteristics
V_{DD} **= 3.135V-3.6V**, VSS = 0 V; T_A = 0 to 70 °C

Parameter	Symbol	Limit Values		Unit	Test Condition	Remarks
		min	max			
Transmitter output impedance	Z_X	10	30	Ω	I_{OUT} = 40mA	LIa, LIb
Receiver input impedance	Z_R	20		kΩ	Transmitter inactive	LIa, LIb single ended
Bit Error Rate	BER		10^{-7}		cable attenuation: 18 dB @ 200 kHz 17.2 dB @ 192 kHz max. noise 10 $\mu V/\sqrt{Hz}$ pulse 1.8 Vpk@100Ω	TE-Mode[1]
loop length for TR mode			100	m		TR[2]-Mode

[1] including No-L1-TE (c)

[2] including No-L1-TE (a) and (b)

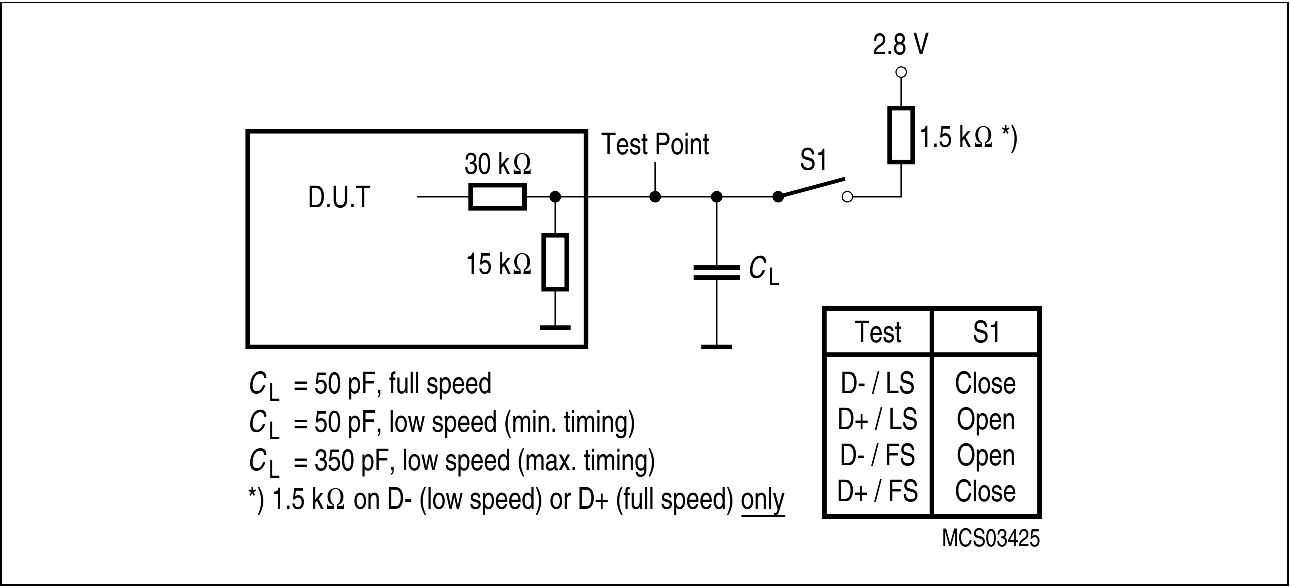
26.9 USB Transceiver Characteristics

The electrical characteristics of the USB device core in INCA-D are compliant to the USB V1.1 specification.

V_{DD} **= 3.135V-3.6V**, VSS = 0 V, T_A = 0 to 70 °C

Parameter	Symbol	Limit Values		Unit	Test Condition
		min.	max.		
Output impedance (high state)	R_{DH}	28	43	Ω	
Output impedance (low state)	R_{DL}	28	51	Ω	
Input leakage current	I_I		5	µA	$V_{IN}=V_{SS}$ or V_{CC}
Tristate output off-state current	I_{OZ}		10	µA	$V_{OUT}=V_{SS}$ or V_{CC} 1)
Crossover point	V_{CR}	1.3	2.0	V	50 pF capacitance

Parameter	Symbol	Limit Values		Unit
		min.	max.	
Rise Time	t_{FR}	4	20	ns
Fall Time	t_{FF}	4	20	ns



C_L = 50 pF, full speed
C_L = 50 pF, low speed (min. timing)
C_L = 350 pF, low speed (max. timing)
*) 1.5 kΩ on D- (low speed) or D+ (full speed) only

Test	S1
D- / LS	Close
D+ / LS	Open
D- / FS	Open
D+ / FS	Close

Figure 26-16 Load for D+/D-

Receiver Sensitivity

The input sensitivity is at least 200 mV when both differential data inputs are in the differential common mode range of 0.8V to 2.5V.

.

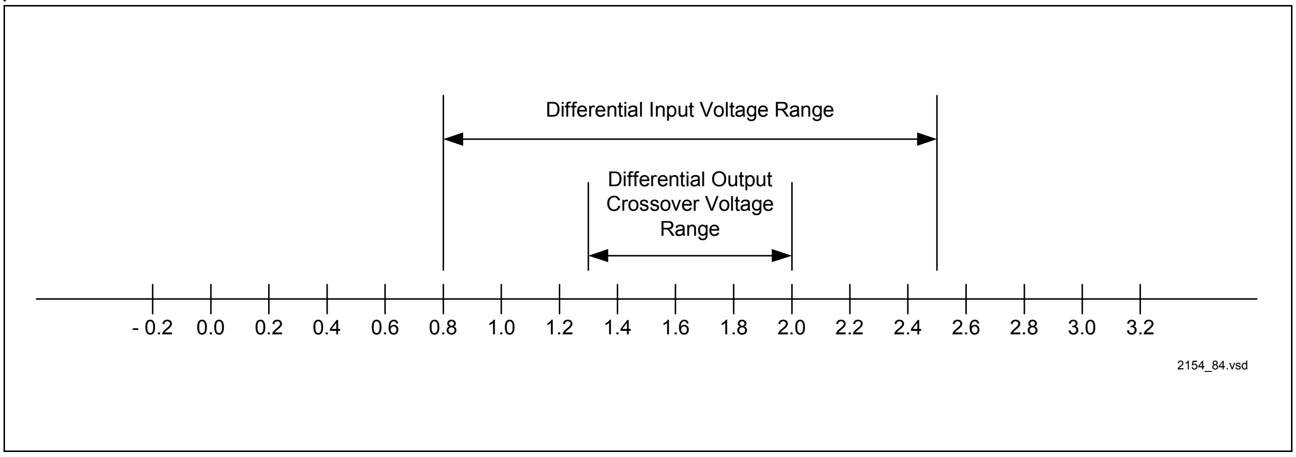

Differential Input Voltage Range
Differential Output Crossover Voltage Range
- 0.2
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
1.8
2.0
2.2
2.4
2.6
2.8
3.0
3.2
2154_84.vsd

Figure 26-17 Differential Input Sensitivity Range

26.10 Electrical Characteristics Analog Front End

26.10.1 DC Characteristics

V_{DD} = **3.15 V – 3.6 V**, VSS = 0 V; T_A = 0 to 70 °C

Parameter	Symbol	Limit Values			Unit	Test Condition
		min.	typ.	max.		
Handset Mode (AFE)	I_{HS}		5.7		mA	
Speakerphone Mode (AFE)	I_{SP}		5.4		mA	
Loudhearing Mode (AFE)	I_{LH}		9.0		mA	

*Note: Operating power dissipation is measured with all analog outputs open.
All analog inputs are set to VREF*

26.10.2 Transmission Characteristics

V_{DD} = **3.135V-3.6V**, VSS = 0 V; T_A = 0 to 70 °C

Parameter	Limit Values		Unit	Test Condition
	min.	max.		
Digital/ Analog Conversion	Transmit : 0 dBm0 corresponds to 661 mV (-1.37 dBm) Receive : 0 dBm0 corresponds to 554 mV (-2.91 dBm)			
Overall programming range (With specified transmission characteristics)	 – 21.5 – 21.5 0	 11.5 2.5 36	 dB dB dB	Receive: loudspeaker earpiece Transmit: differential inputs
Programmable AFE gain	– 0.5 – 1.0	0.5 1.0	dB dB	step accuracy overall accuracy
Attenuation Distortion @ 0 dBm0	0 – 0.25 – 0.25 – 0.25 – 0.25 0	 0.25 0.45 0.9 	dB dB dB dB dB dB	< 200 Hz 200 – 300 Hz 300 – 2400 Hz 2400 – 3000 Hz 3000 – 3400 Hz > 3400 Hz

V_{DD} **= 3.135V-3.6V**, VSS = 0 V; T_A = 0 to 70 °C

Parameter	**Limit Values**		**Unit**	**Test Condition**
	min.	**max.**		
Out-of-band signals		 – 35 – 45 – 35 – 40	 dB dB dB dB	receive: 4.6 kHz 8.0 kHz transmit: 4.6 kHz 8.0 kHz
Group delay distortion @ 0 dBm0		750 380 130 750	µs µs µs µs	500 – 600 Hz 600 – 1000 Hz 1000 – 2600 Hz 2600 – 2800 Hz
Signal-to-total distortion (method 2, sinewave 1kHz)	35 29 24		dB dB dB	0 to – 30 dBm0 – 40 dBm0 – 45 dBm0
Gain tracking (method 2) @ – 10 dBm0	– 0.3 – 0.6 – 1.6	0.3 0.6 1.6	dB dB dB	3 to – 40 dBm0 – 40 to – 50 dBm0 – 50 to – 55 dBm0
Idle-channel noise		– 75 – 66	dBm0 dBm0	receive (A-Law; Psoph.) transmit (A-Law; Psoph.)
Cross-talk		– 66	dB	Reference: 0 dBm0

26.10.3 Analog Front End Input Characteristics

V_{DD} **= 3.135V-3.6V**, VSS = 0 V; T_A = 0 to 70 °C

Parameter	**Symbol**	**Limit Values**			**Unit**	**Test Condition**
		min.	**typ**	**max.**		
AMI-input impedance	Z_{AMI}	12.5	15		kΩ	300 – 3400 Hz
AMI-input voltage swing with specified transmission characteristics	V_{AMI}			19	mVp	36 dB; V_{DD}=3.3V
	V_{AMI_dif}			1.2	Vp	differential; 0 dB

26.10.4 Analog Front End Output Characteristics

V_{DD} **=3.135V-3.6V** VSS = 0 V; T_A = 0 to 70 °C						
HOP/HON-output impedance	Z_{AHO}			2	Ω	300 – 3400 Hz
LSP/LSN-output impedance	Z_{ALS}			2	Ω	300 – 3400 Hz
V_{REF} output impedance	Z_{VREF}		7	10	Ω	Load measured from V_{REF} to V_{SSA}
V_{REF} output voltage	V_{VREF}	1.4	1.5	1.6	V	I_{VREF} = – 1 mA
BGREF output voltage			1.2		V	
BGREF output impedance	Z_{BGREF}	200	300	400	kΩ	
HOP/HON1-output voltage swing	V_{AHO}			1.6	Vpk	Load (200Ω) measured from HOP1 to HON1
HOP/HON2-output voltage swing	V_{AHO}			1.6	Vpk	Load (200Ω) measured from HOP2 to HON2
LSP/LSN-output voltage swing	V_{ALS}			2.0	Vpk	Load 25Ω or 20Ω measured from LSP to LSN

The maximum output voltage swing corresponds to the maximum incoming PCM-code (± 127)

26.11 Electrical Characteristics DSP

DTMF Detector

Parameter	Symbol	Limit Values			Unit	Test Condition
		min.	typ.	max.		
Frequency deviation accept		-1.5		1.5	%	
Frequency deviation reject		3.5		-3.5	%	
Acceptance level		-45		0	dB	rel. to max. PCM
Rejection level				-50	dB	rel. to max. PCM
Twist deviation accept		+/-2		+/-8	dB	programmable
Noise Tolerance				12	dB	
Signal duration accept		40			ms	
Signal duration reject				19	ms	
Gap duration accept				23	ms	

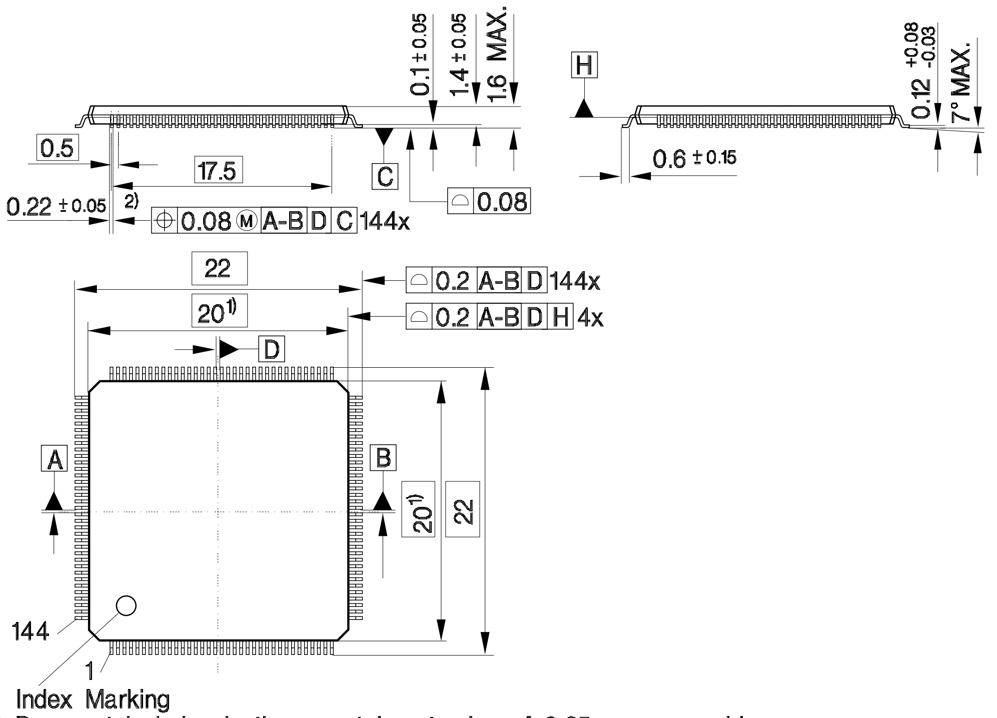
.

Echo Cancellation Unit (subband mode

subband (Hz)		filter length (ms)			
lower limit	upper limit	reduced	analog	ISDN	enhanced
0	250	45.0	97.5	106.5	114.0
250	750	49.5	111.0	135.0	141.0
750	1250	49.5	87.0	103.5	114.0
1250	1750	45.0	60.0	72.0	82.5
1750	2250	40.0	60.0	72.0	82.5
2250	2750	36.0	57.0	63.0	70.5
2750	3250	34.5	30.0	39.0	52.5
3250	3750	33.0	30.0	39.0	52.5

27 Package Outlines

Plastic Package, P-TQFP-144 (SMD)
(Plastic Thin Quad Flat Package)



1) Does not include plastic or metal protrusion of 0.25 max. per side
2) Does not include dambar protrusion of 0.08 max. per side

GPM05247

Sorts of Packing
Package outlines for tubes, trays etc. are contained in our Data Book "Package Information".

SMD = Surface Mounted Device

Dimensions in mm

Infineon goes for Business Excellence

“Business excellence means intelligent approaches and clearly defined processes, which are both constantly under review and ultimately lead to good operating results.
Better operating results and business excellence mean less idleness and wastefulness for all of us, more professional success, more accurate information, a better overview and, thereby, less frustration and more satisfaction.”

Dr. Ulrich Schumacher

http://www.infineon.com

Published by Infineon Technologies AG