SECTION 17
IEEE 1149.1 TEST ACCESS PORT (JTAG)

The MCF5307 includes dedicated user-accessible test logic that is fully compliant with the IEEE standard 1149.1 *Standard Test Access Port and Boundary Scan Architecture*. Use the following description in conjunction with the supporting IEEE document listed above. This section includes the description of those chip-specific items that the IEEE standard requires as well as those items specific to the MCF5307 implementation.

The MCF5307 JTAG test architecture implementation currently supports circuit board test strategies that are based on the IEEE standard. This architecture provides access to all of the data and chip control pins from the board edge connector through the standard four-pin test access port (TAP) and the active-low JTAG reset pin, $\overline{\text{TRST}}$. The test logic itself uses a static design and is wholly independent of the system logic, except where the JTAG is subordinate to other complimentary test modes (see the **Debug Support** section for more information). When in subordinate mode, the JTAG test logic is placed in reset and the TAP pins can be used for other purposes in accordance with the rules and restrictions set forth using a JTAG compliance-enable pin.

The MCF5307 JTAG implementation can:

- Perform boundary-scan operations to test circuit board electrical continuity
- Bypass the MCF5307 device by reducing the shift register path to a single cell
- Sample the MCF5307 system pins during operation and transparently shift out the result
- Set the MCF5307 output drive pins to fixed logic values while reducing the shift register path to a single cell
- Protect the MCF5307 system output and input pins from backdriving and random toggling (such as during in-circuit testing) by placing all system signal pins to high-impedance state

NOTE

The IEEE Standard 1149.1 test logic cannot be considered completely benign to those planning not to use JTAG capability. You must observe certain precautions to ensure that this logic does not interfere with system or debug operation. Refer to Section **17.6 Disabling the IEEE 1149.1 Standard Operation**.

17.1 OVERVIEW

Figure 17-1 is a block diagram of the MCF5307 implementation of the 1149.1 IEEE Standard. The test logic includes several test data registers, an instruction register, instruction register control decode, and a 16-state dedicated TAP controller.

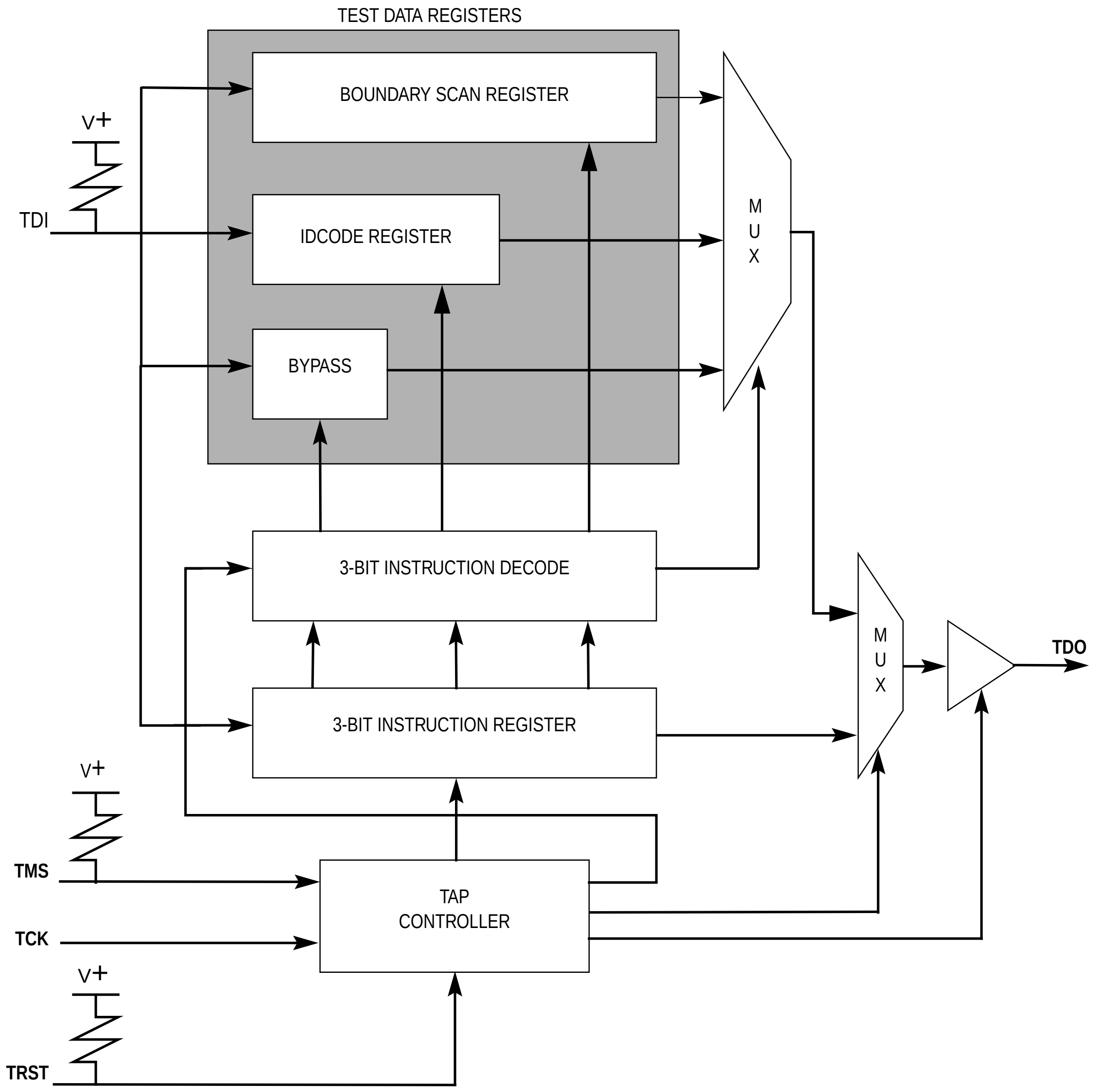


Figure 17-1. JTAG Test Logic Block Diagram

17.2 JTAG SIGNAL DESCRIPTIONS

The JTAG operation on the MCF5307 is enabled when MTMOD[3:0]= 0001. The external pin descriptions in Table 17-1 will now apply.Otherwise, the JTAG Test Access Port signals (TCK/TMS/TDI/TDO/TRST) are interpreted as the Debug port pins.

When the compliance-enable state is set for JTAG mode, apply.

Table 17-1. JTAG Pin Descriptions

PIN	DESCRIPTION
TCK	A test clock input that synchronizes test logic operations
TMS	A test mode select input with a default internal pullup resistor that is sampled on the rising edge of TCK to sequence the TAP controller
TDI	A serial test data input with a default internal pullup resistor that is sampled on the rising edge of TCK
TDO	A three-state test data output that is actively driven only in the Shift-IR and Shift-DR controller states and only updates on the falling edge of TCK
$\overline{\text{TRST}}$	An active-low asynchronous reset with a default internal pullup resistor that forces the TAP controller into the test-logic-reset state.

17.2.1 Test Clock - (TCK)

TCK is the dedicated JTAG test logic clock that is independent of the MCF5307 processor clock. Various JTAG operations occur on the rising or falling edge of TCK. The internal JTAG controller logic is designed such that holding TCK high or low for an indefinite period of time will not cause the JTAG test logic to lose state information. If TCK will not be used, it should be tied to ground.

17.2.2 Test Reset/Development Serial Clock - ($\overline{\text{TRST}}$/DSCLK)

The MTMOD[3:0] signals determine the function of this dual-purpose pin.If MTMOD[3:0]=0000, the DSCLK function is selected. If MTMOD[3:0]= 0001, the TRST function is selected. MTMOD[3:0] should not be changed while $\overline{\text{RSTI}}$ = 1. When used as $\overline{\text{TRST}}$, this pin will asynchronously reset the internal JTAG controller to the test logic reset state, causing the JTAG instruction register to choose the “idcode” command. When this occurs, all the JTAG logic is benign and will not interfere with the normal functionality of the MCF5307 processor. Although this signal is asynchronous, Motorola recommends that $\overline{\text{TRST}}$ make only a 0 to 1 (asserted to negated) transition while TMS is held at a logic 1 value. $\overline{\text{TRST}}$ has an internal pullup so that if it is not driven low its value will default to a logic level of 1. However, if $\overline{\text{TRST}}$ will not be used, it can either be tied to ground or, if TCK is clocked, it can be tied to VDD. The former connection will place the JTAG controller in the test logic reset state immediately, while the later connection will cause the JTAG controller (if TMS is a logic 1) to eventually end up in the test logic reset state after 5 clocks of TCK.

This pin is also used as the development serial clock (DSCLK) for the serial interface to the Debug Module.The maximum frequency for the DSCLK signal is 1/2 the BCLKO frequency. See the **Debug Support** section for additional information on this signal.

17.2.3 Test Mode Select/ Breakpoint (TMS/$\overline{\text{BKPT}}$)

The MTMOD[3:0] signals determine this pin’s dual function. If MTMOD[3:0] =0000, the $\overline{\text{BKPT}}$ function is selected. If MTMOD[3:0] = 0001, then the TMS function is selected. MTMOD[3:0] should not change while $\overline{\text{RSTI}}$ = 1. When used as TMS, this input signal provides the JTAG controller with information to determine which test operation mode should be performed. The value of TMS and current state of the internal 16-state JTAG controller state machine at the rising edge of TCK determine whether the JTAG controller holds its current state or advances to the next state. This directly controls whether JTAG data or instruction operations occur. TMS has an internal pullup so that if it is not driven low, its value will default to a logic level of 1. However, if TMS will not be used, it should be tied to VDD. This pin also signals a hardware breakpoint to the processor when in the debug mode. See the **Debug Support** section for additional information on this signal.

17.2.4 Test Data Input/Development Serial Input - (TDI/DSI)

This is a dual-function pin. If MTMOD[3:0] = 0000, then DSI is selected. If MTMOD[3:0] = 0001, then TDI is selected. When used as TDI, this input signal provides the serial data port for loading the various JTAG shift registers composed of the boundary scan register, the bypass register, and the instruction register. Shifting in of data depends on the state of the JTAG controller state machine and the instruction currently in the instruction register. This data shift occurs on the rising edge of TCK. TDI also has an internal pullup so that if it is not driven low its value will default to a logic level of 1. However, if TDI will not be used, it should be tied to VDD.

This pin also provides the single-bit communication for the debug module commands. See the **Debug Support** section for additional information on this signal.

17.2.5 Test Data Output/Development Serial Output - (TDO/DSO)

This is a dual-function pin. When MTMOD[3:0] = 0000, then DSO is selected. When MTMOD[3:0] = 0001, TDO is selected. When used as TDO, this output signal provides the serial data port for outputting data from the JTAG logic. Shifting out of data depends on the state of the JTAG controller state machine and the instruction currently in the instruction register. This data shift occurs on the falling edge of TCK. When TDO is not outputting test data, it is three-stated. TDO can also be placed in three-state mode to allow bussed or parallel connections to other devices having JTAG. This signal also provides single-bit communication for the debug module responses. See the **Debug Support** section for additional information on this signal.

17.3 JTAG REGISTER DESCRIPTIONS

17.3.1 JTAG Instruction Shift Register

The MCF5307 IEEE 1149.1 Standard implementation uses a 3-bit instruction-shift register without parity. This register transfers its value to a parallel hold register and applies one of six possible instructions on the falling edge of TCK when the TAP state machine is in the update-IR state. To load the instructions into the shift portion of the register, place the serial data on the TDI pin prior to each rising edge of TCK. The MSB of the instruction shift register is the bit closest to the TDI pin and the LSB is the bit closest to the TDO pin.

Table 17-2 lists the public customer-usable instructions that are supported along with their encoding.

Table 17-2. JTAG Instructions

INSTRUCTION	ABBR	CLASS	IR[2:0]	INSTRUCTION SUMMARY
EXTEST	EXT	Required	000	Select BS register while applying fixed values to output pins and asserting functional reset
IDCODE	IDC	Optional	001	Selects IDCODE register for shift
SAMPLE/ PRELOAD	SMP	Required	100	Selects BS register for shift, sample, and preload without disturbing functional operation
HIGHZ	HIZ	Optional	101	Selects the bypass register while three-stating all output pins and asserting functional reset
CLAMP	CMP	Optional	110	Selects bypass while applying fixed values to output pins and asserting functional reset
BYPASS	BYP	Required	111	Selects the bypass register for data operations

The IEEE 1149.1 Standard requires the EXTEST, SAMPLE/PRELOAD, and BYPASS instructions. IDCODE, CLAMP and HIGHZ are optional standard instructions that the MCF5307 implementation supports and are described in the IEEE Standard 1149.1.

17.3.1.1 EXTEST INSTRUCTION. The external test instruction (EXTEST) selects the boundary-scan register. The EXTEST instruction forces all output pins and bidirectional pins configured as outputs to the preloaded fixed values (with the SAMPLE/PRELOAD instruction) and held in the boundary-scan update registers. The EXTEST instruction can also configure the direction of bidirectional pins and establish high-impedance states on some pins. The EXTEST instruction becomes active on the falling edge of TCK in the update-IR state when the data held in the instruction-shift register is equivalent to octal 0.

17.3.1.2 IDCODE. The IDCODE instruction selects the 32-bit IDcode register for connection as a shift path between the TDI pin and the TDO pin. This instruction lets you interrogate the MCF5307 to determine its version number and other part identification data. The IDcode register has been implemented in accordance with IEEE 1149.1 so that the least significant bit of the shift register stage is set to logic 1 on the rising edge of TCK following entry into the capture-DR state. Therefore, the first bit to be shifted out after selecting the IDcode register is always a logic 1. The remaining 31-bits are also set to fixed values (see **17.3.2 IDcode Register**) on the rising edge of TCK following entry into the capture-DR state.

The IDCODE instruction is the default value placed in the instruction register when a JTAG reset is accomplished by either asserting $\overline{\text{TRST}}$ or holding TMS high while clocking TCK through at least five rising edges and the falling edge after the fifth rising edge. A JTAG reset will cause the TAP state machine to enter the test-logic-reset state (normal operation of the TAP state machine into the test-logic-reset state will also result in placing the default value of octal 1 into the instruction register). The shift register portion of the instruction register is loaded with the default value of octal 1 when in the Capture-IR state and a rising edge of TCK occurs.

17.3.1.3 SAMPLE/PRELOAD INSTRUCTION. The SAMPLE/PRELOAD instruction provides two separate functions. First, it obtains a sample of the system data and control signals present at the MCF5307 input pins and just prior to the boundary scan cell at the output pins. This sampling occurs on the rising edge of TCK in the capture-DR state when an instruction encoding of octal 4 is resident in the instruction register. You can observe this sampled data by shifting it through the boundary-scan register to the output TDO by using the shift-DR state. Both the data capture and the shift operation are transparent to system operation. You are responsible for providing some form of external synchronization to achieve meaningful results because there is no internal synchronization between TCK and the system clock, CLK.

The second function of the SAMPLE/PRELOAD instruction is to initialize the boundary scan register update cells before selecting EXTEST or CLAMP. This is achieved by ignoring the data being shifted out of the TDO pin while shifting in initialization data. The update-DR state in conjunction with the falling edge of TCK can then transfer this data to the update cells. This data will be applied to the external output pins when one of the instructions listed above is applied.

17.3.1.4 HIGHZ INSTRUCTION. The HIGHZ instruction anticipates the need to backdrive the output pins and protect the input pins from random toggling during circuit board testing. The HIGHZ instruction selects the bypass register, forcing all output and bidirectional pins to the high-impedance state.

The HIGHZ instruction goes active on the falling edge of TCK in the update-IR state when the data held in the instruction shift register is equivalent to octal 5.

17.3.1.5 CLAMP INSTRUCTION. The CLAMP instruction selects the bypass register and asserts functional reset while simultaneously forcing all output pins and bidirectional pins configured as outputs to the fixed values that are preloaded and held in the boundary-scan update registers. This instruction enhances test efficiency by reducing the overall shift path to a single bit (the bypass register) while conducting an EXTEST type of instruction through the boundary-scan register. The CLAMP instruction becomes active on the falling edge of TCK in the update-IR state when the data held in the instruction-shift register is equivalent to octal 6.

17.3.1.6 BYPASS INSTRUCTION. The BYPASS instruction selects the single-bit bypass register, creating a single-bit shift register path from the TDI pin to the bypass register to the TDO pin. This instruction enhances test efficiency by reducing the overall shift path when a device other than the MCF5307 processor becomes the device under test on a board design with multiple chips on the overall 1149.1 defined boundary-scan chain. The bypass register has been implemented in accordance with 1149.1 so that the shift register stage is set to logic zero on the rising edge of TCK following entry into the capture-DR state. Therefore, the first bit to be shifted out after selecting the bypass register is always a logic zero (to differentiate a part that supports an IDCODE register from a part that supports only the bypass register). The BYPASS instruction goes active on the falling edge of TCK in the update-IR state when the data held in the instruction shift register is equivalent to octal 7.

17.3.2 IDcode Register

An IEEE 1149.1 compliant JTAG identification register has been included on the MCF5307. The MCF5307 JTAG instruction encoded as octal 1 provides for reading the JTAG IDcode register.

ID code Register

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VERSION NO				0	1	0	0	1	1	0	0	0	0	0	0

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

Bits 31-28 Version Number

Indicates the revision number of the MCF5307.

Bits 27-22 Design Center

Indicates the ColdFire design center.

Bits 21-12 Device Number

Indicates an MCF5307.

Bits 11-1 JEDEC ID

Indicates the reduced JEDEC ID for Motorola (JEDEC refers to the Joint Electron Device Engineering Council. Refer to JEDEC publication 106-A and chapter 11 of the IEEE 1149.1 Standard for further information on this field).

Bit 0

Differentiates this register as the JTAG IDcode register (as opposed to the bypass register) according to the IEEE 1149.1 Standard.

17.3.3 JTAG BOUNDARY-SCAN REGISTER

The MCF5307 model includes an IEEE 1149.1-compliant boundary-scan register. The boundary-scan register is connected between TDI and TDO when the EXTEST or SAMPLE/PRELOAD instructions are selected. This register captures signal pin data on the input pins, forces fixed values on the output signal pins, and selects the direction and drive characteristics (a logic value or high impedance) of the bidirectional and three-state signal pins.

Table 17-3. Boundary-Scan Bit Definitions

BIT	CELL TYPE	PINCELL NAME	PIN TYPE
0	O.Ctl	PP(0) enable	-
1	O.Pin	PP(0)	I/O
2	I.Pin	PP(0)	I/O
3	IO.Ctl	PP(1) enable	-
4	O.Pin	PP(1)	I/O
5	I.Pin	PP(1)	I/O
6	IO.Ctl	PP(2) enable	-
7	O.Pin	PP(2)	I/O
8	I.Pin	PP(2)	I/O
9	IO.Ctl	PP(3) enable	-
10	O.Pin	PP(3)	I/O
11	I.Pin	PP(3)	I/O
12	IO.Ctl	PP(4) enable	-
13	O.Pin	PP(4)	I/O
14	I.Pin	PP(4)	I/O
15	IO.Ctl	PP(5) enable	-
16	O.Pin	PP(5)	I/O
17	I.Pin	PP(5)	I/O
18	IO.Ctl	PP(6) enable	-
19	O.Pin	PP(6)	I/O
20	I.Pin	PP(6)	I/O
21	IO.Ctl	PP(7) enable	-
22	O.Pin	PP(7)	I/O
23	I.Pin	PP(7)	I/O
24	O.Pin	PST(3)	O
25	O.Pin	PST(2)	O
26	O.Pin	PST(1)	O
27	O.Pin	PST(0)	O
28	O.Pin	DDATA(3)	O
29	O.Pin	DDATA(2)	O
30	O.Pin	DDATA(1)	O
31	O.Pin	DDATA(0)	O
32	O.Pin	PSTCLK	O
33	I.Pin	CLKIN	I
34	IO.Ctl	XRSTO enable	-
35	O.Pin	XRSTO	I/O
36	I.Pin	XRSTO	I/O
37	O.Pin	BCLKO	O
38	I.Pin	EDGESEL	I
39	O.Pin	TXD1	O
40	I.Pin	RXD1	I
41	O.Pin	RTS1	O
42	I.Pin	CTS1	I
43	O.Pin	TXD2	O

Table 17-3. Boundary-Scan Bit Definitions (Continued)

BIT	CELL TYPE	PINCELL NAME	PIN TYPE
44	I.Pin	RXD2	I
45	O.Pin	RTS2	O
46	I.Pin	CTS2	I
47	I.Pin	XHIZ	I
48	IO.Ctl	DATA enable	-
49	O.Pin	DATA(0)	I/O
50	I.Pin	DATA(0)	I/O
51	O.Pin	DATA(1)	I/O
52	I.Pin	DATA(1)	I/O
53	O.Pin	DATA(2)	I/O
54	I.Pin	DATA(2)	I/O
55	O.Pin	DATA(3)	I/O
56	I.Pin	DATA(3)	I/O
57	O.Pin	DATA(4)	I/O
58	I.Pin	DATA(4)	I/O
59	O.Pin	DATA(5)	I/O
60	I.Pin	DATA(5)	I/O
61	O.Pin	DATA(6)	I/O
62	I.Pin	DATA(6)	I/O
63	O.Pin	DATA(7)	I/O
64	I.Pin	DATA(7)	I/O
65	O.Pin	DATA(8)	I/O
66	I.Pin	DATA(8)	I/O
67	O.Pin	DATA(9)	I/O
68	I.Pin	DATA(9)	I/O
69	O.Pin	DATA(10)	I/O
70	I.Pin	DATA(10)	I/O
71	O.Pin	DATA(11)	I/O
72	I.Pin	DATA(11)	I/O
73	O.Pin	DATA(12)	I/O
74	I.Pin	DATA(12)	I/O
75	O.Pin	DATA(13)	I/O
76	I.Pin	DATA(13)	I/O
77	O.Pin	DATA(14)	I/O
78	I.Pin	DATA(14)	I/O
79	O.Pin	DATA(15)	I/O
80	I.Pin	DATA(15)	I/O
81	O.Pin	DATA(16)	I/O
82	I.Pin	DATA(16)	I/O
83	O.Pin	DATA(17)	I/O
84	I.Pin	DATA(17)	I/O
85	O.Pin	DATA(18)	I/O
86	I.Pin	DATA(18)	I/O
87	O.Pin	DATA(19)	I/O
88	I.Pin	DATA(19)	I/O
89	O.Pin	DATA(20)	I/O
90	I.Pin	DATA(20)	I/O

Table 17-3. Boundary-Scan Bit Definitions (Continued)

BIT	CELL TYPE	PINCELL NAME	PIN TYPE
91	O.Pin	DATA(21)	I/O
92	I.Pin	DATA(21)	I/O
93	O.Pin	DATA(22)	I/O
94	I.Pin	DATA(22)	I/O
95	O.Pin	DATA(23)	I/O
96	I.Pin	DATA(23)	I/O
97	O.Pin	DATA(24)	I/O
98	I.Pin	DATA(24)	I/O
99	O.Pin	DATA(25)	I/O
100	I.Pin	DATA(25)	I/O
101	O.Pin	DATA(26)	I/O
102	I.Pin	DATA(26)	I/O
103	O.Pin	DATA(27)	I/O
104	I.Pin	DATA(27)	I/O
105	O.Pin	DATA(28)	I/O
106	I.Pin	DATA(28)	I/O
107	O.Pin	DATA(29)	I/O
108	I.Pin	DATA(29)	I/O
109	O.Pin	DATA(30)	I/O
110	I.Pin	DATA(30)	I/O
111	O.Pin	DATA(31)	I/O
112	I.Pin	DATA(31)	I/O
113	O.Pin	SDA	OD
114	I.Pin	SDA	I
115	O.Pin	SCL	OD
116	I.Pin	SCL	I
117	O.Pin	XBE(3)	O
118	O.Pin	XBE(2)	O
119	O.Pin	XBE(1)	O
120	O.Pin	XBE(0)	O
121	O.Pin	SCKE	O
122	O.Pin	SCAS	O
123	O.Pin	SRAS	O
124	O.Pin	XDRAMW	O
125	O.Pin	XCAS(3)	O
126	O.Pin	XCAS(2)	O
127	O.Pin	XCAS(1)	O
128	O.Pin	XCAS(0)	O
129	O.Pin	XRAS(1)	O
130	O.Pin	XRAS(0)	O
131	I.Pin	TIN1	I
132	I.Pin	TIN0	I
133	O.Pin	TOUT0	O
134	O.Pin	TOUT1	O
135	I.Pin	XBG	I
136	O.Pin	XBD	O
137	O.Pin	XBR	O

Table 17-3. Boundary-Scan Bit Definitions (Continued)

BIT	CELL TYPE	PINCELL NAME	PIN TYPE
138	I.Pin	XIRQ1	I
139	I.Pin	XIRQ3	I
140	I.Pin	XIRQ5	I
141	I.Pin	XIRQ7	I
142	I.Pin	XRSTI	I
143	O.Pin	XTS	I/O
144	I.Pin	XTS	I/O
145	IO.Ctl	XTA enable	-
146	O.Pin	XTA	I/O
147	I.Pin	XTA	I/O
148	O.Pin	RW	I/O
149	I.Pin	RW	I/O
150	O.Pin	XAS	I/O
151	I.Pin	XAS	I/O
152	O.Pin	XCS(7)	O
153	O.Pin	XCS(6)	O
154	O.Pin	XCS(5)	O
155	O.Pin	XCS(4)	O
156	O.Pin	XCS(3)	O
157	O.Pin	XCS(2)	O
158	O.Pin	XCS(1)	O
159	O.Pin	XCS(0)	O
160	O.Pin	XOE	O
161	O.Pin	SIZ(1)	I/O
162	I.Pin	SIZ(1)	I/O
163	O.Pin	SIZ(0)	I/O
164	I.Pin	SIZ(0)	I/O
165	IO.Ctl	PP(15) enable	-
166	I.Pin	PP(15)	I/O
167	O.Pin	PP(15)	I/O
168	IO.Ctl	PP(14) enable	-
169	I.Pin	PP(14)	I/O
170	O.Pin	PP(14)	I/O
171	IO.Ctl	PP(13) enable	-
172	I.Pin	PP(13)	I/O
173	O.Pin	PP(13)	I/O
174	IO.Ctl	PP(12) enable	-
175	I.Pin	PP(12)	I/O
176	O.Pin	PP(12)	I/O
177	IO.Ctl	PP(11) enable	-
178	I.Pin	PP(11)	I/O
179	O.Pin	PP(11)	I/O
180	IO.Ctl	PP(10) enable	-
181	I.Pin	PP(10)	I/O
182	O.Pin	PP(10)	I/O
183	IO.Ctl	PP(9) enable	-
184	I.Pin	PP(9)	I/O

Table 17-3. Boundary-Scan Bit Definitions (Continued)

BIT	CELL TYPE	PINCELL NAME	PIN TYPE
185	O.Pin	PP(9)	I/O
186	IO.Ctl	PP(8) enable	-
187	I.Pin	PP(8)	I/O
188	O.Pin	PP(8)	I/O
189	IO.Ctl	XTS / RW / SIZ enable	-
190	IO.Ctl	ADDR enable	-
191	O.Pin	ADDR(23)	I/O
192	I.Pin	ADDR(23)	I/O
193	O.Pin	ADDR(22)	I/O
194	I.Pin	ADDR(22)	I/O
195	O.Pin	ADDR(21)	I/O
196	I.Pin	ADDR(21)	I/O
197	O.Pin	ADDR(20)	I/O
198	I.Pin	ADDR(20)	I/O
199	O.Pin	ADDR(19)	I/O
200	I.Pin	ADDR(19)	I/O
201	O.Pin	ADDR(18)	I/O
202	I.Pin	ADDR(18)	I/O
203	O.Pin	ADDR(17)	I/O
204	I.Pin	ADDR(17)	I/O
205	O.Pin	ADDR(16)	I/O
206	I.Pin	ADDR(16)	I/O
207	O.Pin	ADDR(15)	I/O
208	I.Pin	ADDR(15)	I/O
209	O.Pin	ADDR(14)	I/O
210	I.Pin	ADDR(14)	I/O
211	O.Pin	ADDR(13)	I/O
212	I.Pin	ADDR(13)	I/O
213	O.Pin	ADDR(12)	I/O
214	I.Pin	ADDR(12)	I/O
215	O.Pin	ADDR(11)	I/O
216	I.Pin	ADDR(11)	I/O
217	O.Pin	ADDR(10)	I/O
218	I.Pin	ADDR(10)	I/O
219	O.Pin	ADDR(9)	I/O
220	I.Pin	ADDR(9)	I/O
221	O.Pin	ADDR(8)	I/O
222	I.Pin	ADDR(8)	I/O
223	O.Pin	ADDR(7)	I/O
224	I.Pin	ADDR(7)	I/O
225	O.Pin	ADDR(6)	I/O
226	I.Pin	ADDR(6)	I/O
227	O.Pin	ADDR(5)	I/O
228	I.Pin	ADDR(5)	I/O
229	O.Pin	ADDR(4)	I/O
230	I.Pin	ADDR(4)	I/O
231	O.Pin	ADDR(3)	I/O

Table 17-3. Boundary-Scan Bit Definitions (Continued)

BIT	CELL TYPE	PINCELL NAME	PIN TYPE
232	I.Pin	ADDR(3)	I/O
233	O.Pin	ADDR(2)	I/O
234	I.Pin	ADDR(2)	I/O
235	O.Pin	ADDR(1)	I/O
236	I.Pin	ADDR(1)	I/O
237	O.Pin	ADDR(0)	I/O
238	I.Pin	ADDR(0)	I/O

17.3.4 JTAG BYPASS REGISTER

The MCF5307 includes an IEEE 1149.1-compliant bypass register, which creates a single bit shift register path from TDI to the bypass register to TDO when the BYPASS instruction is selected.

17.4 TAP CONTROLLER

The value of TMS at the rising edge of TCK determines the current state of the TAP controller. There are basically two paths that the TAP controller can follow: The first, for executing JTAG instructions; the second, for manipulating JTAG data based on the JTAG instructions. The various states of the TAP controller are shown in Figure 17-2. For more detail on each state, refer to the IEEE 1149.1 Standard JTAG document. Do note, though, that from any state the TAP controller is in, Test-Logic-Reset can be entered if TMS is held high for at least five rising edges of TCK.

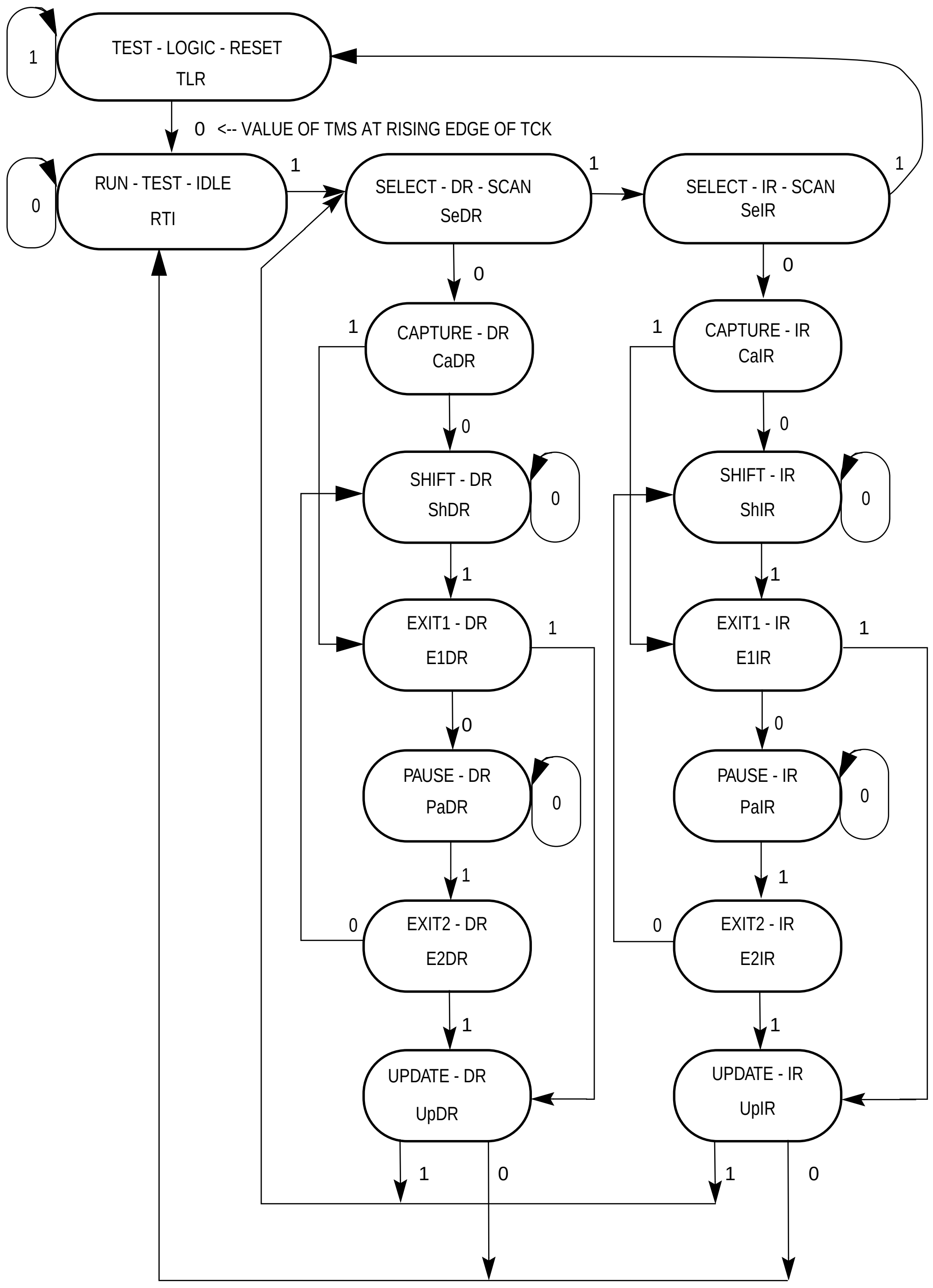


Figure 17-2. JTAG TAP Controller State Machine

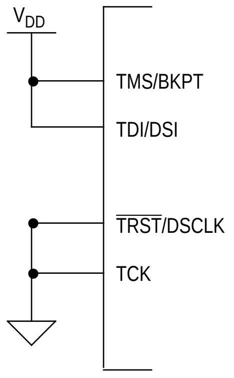
17.5 RESTRICTIONS

The test logic is implemented using static logic design, and TCK can be stopped in either a high or low state without loss of data. The system logic, however, operates on a different system clock which is not synchronized to TCK internally. Any mixed operation requiring the use of 1149.1 test logic in conjunction with system functional logic that uses both clocks, must have coordination and synchronization of these clocks done externally to the MCF5307.

17.6 DISABLING THE IEEE 1149.1 STANDARD OPERATION

There are two methods by which the MCF5307 can be used without the IEEE 1149.1 test logic being active: 1) Nonuse of the JTAG test logic by either nontermination (disconnection) or intentional fixing of TAP logic values, and 2) Intentional disabling of the JTAG test logic by **NOT setting MTMOD[3:0]= 0001** (entering Debug mode).

There are several considerations that must be addressed if the IEEE 1149.1 logic is not going to be used once the MCF5307 is assembled onto a board. The prime consideration is to ensure that the IEEE 1149.1 test logic remains transparent and benign to the system logic during functional operation. This requires the minimum of either connecting the $\overline{\text{TRST}}$ pin to logic 0, or connecting the TCK clock pin to a clock source that will supply five rising edges and the falling edge after the fifth rising edge, to ensure that the part enters the test-logic-reset state. The recommended solution is to connect $\overline{\text{TRST}}$ to logic 0. Another consideration is that the TCK pin does not have an internal pullup as is required on the TMS, TDI, and $\overline{\text{TRST}}$ pins; therefore, it should not be left unterminated to preclude mid-level input values. Figure 17-3 shows pin values recommended for disabling JTAG with the MCF5307 in JTAG mode.

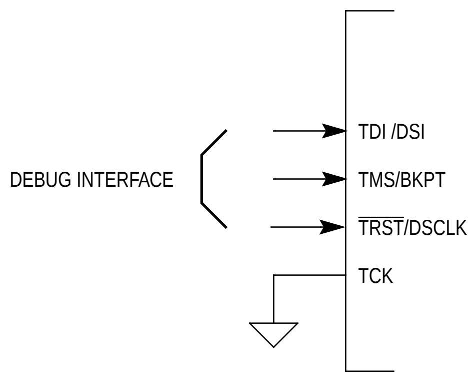


NOTE: MTMOD[3:0] SET TO ‘ 0001’ ALLOWS JTAG MODE.

Figure 17-3. Disabling JTAG in JTAG Mode

A second method of using the MCF5307 without the IEEE 1149.1 logic being active is to select Debug mode by **setting MTMOD[3:0]= 0000**. The IEEE 1149.1 test controller is now placed in the test-logic-reset state by the internal assertion of the $\overline{\text{TRST}}$ signal to the controller and the TAP pins function as Debug mode pins. While in JTAG mode, input pins

TDI/DSI, TMS/$\overline{\text{BKPT}}$, and $\overline{\text{TRST}}$/DSCLK have internal pullups enabled. Figure 17-4 shows pin values recommended for disabling JTAG with the MCF5307 in Debug mode.



NOTE: MTMOD[3:0] NOT SET TO' 0001' PROHIBITS JTAG.

Figure 17-4. Disabling JTAG in Debug Mode

17.7 MCF5307 BSDL FILE

```
-- M O T O R O L A   S S D T   J T A G   S O F T W A R E
-- BSDL File Generated: Wed Jul  2 18:00:53 1997
--
-- Revision History:
--

entity MCF5307 is
generic (PHYSICAL_PIN_MAP : string := "TQFP_208");

port ( BKPT:inbit;
        DSI:inbit;
        DSO:outbit;
      DSCLK:inbit;
        TCK:inbit;
       ADDR:inoutbit_vector(0 to 23);
        SIZ:inoutbit_vector(0 to 1);
        XOE:bufferbit;
        XCS:bufferbit_vector(0 to 7);
        XAS:inoutbit;
         RW:inoutbit;
        XTA:inoutbit;
        XTS:inoutbit;
      XRSTI:inbit;
        XBR:bufferbit;
        XBD:bufferbit;
        XBG:inbit;
      XIRQ7:inbit;
      XIRQ5:inbit;
      XIRQ3:inbit;
      XIRQ1:inbit;
      TOUT1:bufferbit;
      TOUT0:bufferbit;
       TIN0:inbit;
       TIN1:inbit;
       XRAS:bufferbit_vector(0 to 1);
       XCAS:bufferbit_vector(0 to 3);
     XDRAMW:bufferbit;
       SRAS:bufferbit;
       SCAS:bufferbit;
       SCKE:bufferbit;
        XBE:bufferbit_vector(0 to 3);
        SCL:inoutbit;
        SDA:inoutbit;
       DATA:inoutbit_vector(0 to 31);
       XHIZ:inbit;
      MTMOD:linkagebit_vector(0 to 3);
       CTS2:inbit;
       RTS2:bufferbit;
       RXD2:inbit;
       TXD2:bufferbit;
       CTS1:inbit;
       RTS1:bufferbit;
       RXD1:inbit;
       TXD1:bufferbit;
    EDGESEL:inbit;
      BCLKO:bufferbit;
      XRSTO:inoutbit;
      CLKIN:inbit;
     PSTCLK:bufferbit;
      DDATA:bufferbit_vector(0 to 3);
        PST:bufferbit_vector(0 to 3);
         PP:inoutbit_vector(0 to 15);
        VDD:linkagebit_vector(0 to 27);
        GND:linkagebit_vector(0 to 28);
       PVDD:linkagebit_vector(0 to 1);
       PGND:linkagebit_vector(0 to 1);
     PLLTPA:linkagebit);

use STD_1149_1_1994.all;

attribute COMPONENT_CONFORMANCE of MCF5307 : entity is "STD_1149_1_1993";

attribute PIN_MAP of MCF5307 : entity is PHYSICAL_PIN_MAP;

constant TQFP_208 : PIN_MAP_STRING :=
        "VDD:      (1, 7, 13, 21, 29, 37, 45, 52, 57, 65, 73, 81, 89, 97, 105,
113, 121, 129, 137,  " &
"145, 152, 157, 165, 173, 177, 191, 197, 205), " &
        "ADDR:     (2, 3, 5, 6, 8, 9, 11, 12, 14, 15, 16, 18, 19, 20, 22, 23,
24, 26, 27, 28, 30,  " &
"31, 32, 34), " &
         "GND:      (4, 10, 17, 25, 33, 41, 48, 53, 61, 69, 77, 85, 93, 101,
104, 109, 117, 125,  " &
"133, 141, 148, 156, 161, 169, 175, 188, 194, 201, 208), " &
```

```
        "PP:      (207, 206, 204, 203, 202, 200, 199, 198, 35, 36, 38, 39, 40,
42, 43, 44), " &
         "SIZ:      (46, 47), " &
         "XOE:       49, " &
         "XCS:      (50, 51, 54, 55, 56, 58, 59, 60), " &
         "XAS:       62, " &
         "RW:        63, " &
         "XTA:       64, " &
         "XTS:       66, " &
         "XRSTI:     67, " &
         "XIRQ7:     68, " &
         "XIRQ5:     70, " &
         "XIRQ3:     71, " &
         "XIRQ1:     72, " &
         "XBR:       74, " &
         "XBD:       75, " &
         "XBG:       76, " &
         "TOUT1:     78, " &
         "TOUT0:     79, " &
         "TIN1:      80, " &
         "TIN0:      82, " &
         "XRAS:     (83, 84), " &
         "XCAS:     (86, 87, 88, 90), " &
         "XDRAMW:    91, " &
         "SRAS:      92, " &
         "SCAS:      94, " &
         "SCKE:      95, " &
         "XBE:      (96, 98, 99, 100), " &
         "SCL:       102, " &
         "SDA:       103, " &
        "DATA:    (147, 146, 144, 143, 142, 140, 139, 138, 136, 135, 134, 132,
131, 130, 128, 127,  " &
"126, 124, 123, 122, 120, 119, 118, 116, 115, 114, 112, 111, 110, 108, 107, 106),
" &
         "DSCLK:     149, " &
         "TCK:       150, " &
         "DSO:       151, " &
         "DSI:       153, " &
         "BKPT:      154, " &
         "XHIZ:      155, " &
         "MTMOD:    (158, 159, 160, 162), " &
         "CTS2:      163, " &
         "RTS2:      164, " &
         "RXD2:      166, " &
         "TXD2:      167, " &
         "CTS1:      168, " &
         "RTS1:      170, " &
         "RXD1:      171, " &
         "TXD1:      172, " &
         "EDGESEL:   174, " &
         "BCLKO:     176, " &
         "XRSTO:     178, " &
         "PVDD:     (179, 185), " &
         "CLKIN:     180, " &
         "PGND:     (181, 183), " &
         "PLLTPA:    182, " &
         "PSTCLK:    184, " &
         "DDATA:    (186, 187, 189, 190), " &
         "PST:      (192, 193, 195, 196) ";

attribute TAP_SCAN_IN    of    DSI : signal is true;
attribute TAP_SCAN_OUT   of    DSO : signal is true;
attribute TAP_SCAN_MODE  of   BKPT : signal is true;
attribute TAP_SCAN_RESET of  DSCLK : signal is true;
attribute TAP_SCAN_CLOCK of    TCK : signal is (20.0e6, BOTH);

attribute INSTRUCTION_LENGTH of MCF5307 : entity is 3;

attribute INSTRUCTION_OPCODE of MCF5307 : entity is
   "EXTEST  (000)," &
   "SAMPLE  (100)," &
   "IDCODE  (001)," &
   "CLAMP   (110)," &
   "HIGHZ   (101)," &
   "BYPASS  (111)";

attribute INSTRUCTION_CAPTURE of MCF5307 : entity is "001";
attribute IDCODE_REGISTER   of MCF5307 : entity is
   "0000"          & -- version
   "010011"        & -- manufacturer's use
   "0000000011"    & -- sequence number
   "00000001110"   & -- manufacturer identity
   "1";              -- 1149.1 requirement

attribute BOUNDARY_LENGTH of MCF5307 : entity is 239;

attribute BOUNDARY_REGISTER of MCF5307 : entity is
-- num    cell   port   func           safe [ccell  dis  rslt]
   "0     (BC_2, *,        controlr,      1)," &
   "1     (BC_2, PP(0),    output3,       X,      0,   1,   Z)," &
   "2     (BC_4, PP(0),    input,         X)," &
   "3     (BC_2, *,        controlr,      1)," &
   "4     (BC_2, PP(1),    output3,       X,      3,   1,   Z)," &
   "5     (BC_4, PP(1),    input,         X)," &
   "6     (BC_2, *,        controlr,      1)," &
   "7     (BC_2, PP(2),    output3,       X,      6,   1,   Z)," &
   "8     (BC_4, PP(2),    input,         X)," &
   "9     (BC_2, *,        controlr,      1)," &
   "10    (BC_2, PP(3),    output3,       X,      9,   1,   Z)," &
   "11    (BC_4, PP(3),    input,         X)," &
   "12    (BC_2, *,        controlr,      1)," &
   "13    (BC_2, PP(4),    output3,       X,     12,   1,   Z)," &
   "14    (BC_4, PP(4),    input,         X)," &
   "15    (BC_2, *,        controlr,      1)," &
   "16    (BC_2, PP(5),    output3,       X,     15,   1,   Z)," &
   "17    (BC_4, PP(5),    input,         X)," &
   "18    (BC_2, *,        controlr,      1)," &
   "19    (BC_2, PP(6),    output3,       X,     18,   1,   Z)," &
```

```
-- num    cell   port   func           safe [ccell  dis  rslt]
   "20    (BC_4, PP(6),    input,         X)," &
   "21    (BC_2, *,        controlr,      1)," &
   "22    (BC_2, PP(7),    output3,       X,     21,   1,   Z)," &
   "23    (BC_4, PP(7),    input,         X)," &
   "24    (BC_2, PST(3),   output2,       X)," &
   "25    (BC_2, PST(2),   output2,       X)," &
   "26    (BC_2, PST(1),   output2,       X)," &
   "27    (BC_2, PST(0),   output2,       X)," &
   "28    (BC_2, DDATA(3), output2,       X)," &
   "29    (BC_2, DDATA(2), output2,       X)," &
   "30    (BC_2, DDATA(1), output2,       X)," &
   "31    (BC_2, DDATA(0), output2,       X)," &
   "32    (BC_2, PSTCLK,   output2,       X)," &
   "33    (BC_4, CLKIN,    input,         X)," &
   "34    (BC_2, *,        controlr,      1)," &
   "35    (BC_2, XRSTO,    output3,       X,     34,   1,   Z)," &
   "36    (BC_4, XRSTO,    input,         X)," &
   "37    (BC_2, BCLKO,    output2,       X)," &
   "38    (BC_4, EDGESEL,  input,         X)," &
   "39    (BC_2, TXD1,     output2,       X)," &
-- num    cell   port   func           safe [ccell  dis  rslt]
   "40    (BC_4, RXD1,     input,         X)," &
   "41    (BC_2, RTS1,     output2,       X)," &
   "42    (BC_4, CTS1,     input,         X)," &
   "43    (BC_2, TXD2,     output2,       X)," &
   "44    (BC_4, RXD2,     input,         X)," &
   "45    (BC_2, RTS2,     output2,       X)," &
   "46    (BC_4, CTS2,     input,         X)," &
   "47    (BC_4, XHIZ,     input,         X)," &
   "48    (BC_2, *,        controlr,      1)," &
   "49    (BC_2, DATA(0),  output3,       X,     48,   1,   Z)," &
   "50    (BC_4, DATA(0),  input,         X)," &
   "51    (BC_2, DATA(1),  output3,       X,     48,   1,   Z)," &
   "52    (BC_4, DATA(1),  input,         X)," &
   "53    (BC_2, DATA(2),  output3,       X,     48,   1,   Z)," &
   "54    (BC_4, DATA(2),  input,         X)," &
   "55    (BC_2, DATA(3),  output3,       X,     48,   1,   Z)," &
   "56    (BC_4, DATA(3),  input,         X)," &
   "57    (BC_2, DATA(4),  output3,       X,     48,   1,   Z)," &
   "58    (BC_4, DATA(4),  input,         X)," &
   "59    (BC_2, DATA(5),  output3,       X,     48,   1,   Z)," &
-- num    cell   port   func           safe [ccell  dis  rslt]
   "60    (BC_4, DATA(5),  input,         X)," &
   "61    (BC_2, DATA(6),  output3,       X,     48,   1,   Z)," &
   "62    (BC_4, DATA(6),  input,         X)," &
   "63    (BC_2, DATA(7),  output3,       X,     48,   1,   Z)," &
   "64    (BC_4, DATA(7),  input,         X)," &
   "65    (BC_2, DATA(8),  output3,       X,     48,   1,   Z)," &
   "66    (BC_4, DATA(8),  input,         X)," &
   "67    (BC_2, DATA(9),  output3,       X,     48,   1,   Z)," &
   "68    (BC_4, DATA(9),  input,         X)," &
   "69    (BC_2, DATA(10), output3,       X,     48,   1,   Z)," &
   "70    (BC_4, DATA(10), input,         X)," &
   "71    (BC_2, DATA(11), output3,       X,     48,   1,   Z)," &
   "72    (BC_4, DATA(11), input,         X)," &
   "73    (BC_2, DATA(12), output3,       X,     48,   1,   Z)," &
   "74    (BC_4, DATA(12), input,         X)," &
   "75    (BC_2, DATA(13), output3,       X,     48,   1,   Z)," &
   "76    (BC_4, DATA(13), input,         X)," &
   "77    (BC_2, DATA(14), output3,       X,     48,   1,   Z)," &
   "78    (BC_4, DATA(14), input,         X)," &
   "79    (BC_2, DATA(15), output3,       X,     48,   1,   Z)," &
-- num    cell   port   func          safe [ccell  dis  rslt]
   "80    (BC_4, DATA(15), input,         X)," &
   "81    (BC_2, DATA(16), output3,       X,     48,   1,   Z)," &
   "82    (BC_4, DATA(16), input,         X)," &
   "83    (BC_2, DATA(17), output3,       X,     48,   1,   Z)," &
   "84    (BC_4, DATA(17), input,         X)," &
   "85    (BC_2, DATA(18), output3,       X,     48,   1,   Z)," &
   "86    (BC_4, DATA(18), input,         X)," &
   "87    (BC_2, DATA(19), output3,       X,     48,   1,   Z)," &
   "88    (BC_4, DATA(19), input,         X)," &
   "89    (BC_2, DATA(20), output3,       X,     48,   1,   Z)," &
   "90    (BC_4, DATA(20), input,         X)," &
   "91    (BC_2, DATA(21), output3,       X,     48,   1,   Z)," &
   "92    (BC_4, DATA(21), input,         X)," &
   "93    (BC_2, DATA(22), output3,       X,     48,   1,   Z)," &
   "94    (BC_4, DATA(22), input,         X)," &
   "95    (BC_2, DATA(23), output3,       X,     48,   1,   Z)," &
   "96    (BC_4, DATA(23), input,         X)," &
   "97    (BC_2, DATA(24), output3,       X,     48,   1,   Z)," &
   "98    (BC_4, DATA(24), input,         X)," &
   "99    (BC_2, DATA(25), output3,       X,     48,   1,   Z)," &
-- num    cell   port   func          safe [ccell  dis  rslt]
   "100   (BC_4, DATA(25), input,         X)," &
   "101   (BC_2, DATA(26), output3,       X,     48,   1,   Z)," &
   "102   (BC_4, DATA(26), input,         X)," &
   "103   (BC_2, DATA(27), output3,       X,     48,   1,   Z)," &
   "104   (BC_4, DATA(27), input,         X)," &
   "105   (BC_2, DATA(28), output3,       X,     48,   1,   Z)," &
   "106   (BC_4, DATA(28), input,         X)," &
   "107   (BC_2, DATA(29), output3,       X,     48,   1,   Z)," &
   "108   (BC_4, DATA(29), input,         X)," &
   "109   (BC_2, DATA(30), output3,       X,     48,   1,   Z)," &
   "110   (BC_4, DATA(30), input,         X)," &
   "111   (BC_2, DATA(31), output3,       X,     48,   1,   Z)," &
   "112   (BC_4, DATA(31), input,         X)," &
   "113   (BC_2, SDA,      output2,       1,    113,   1,   Weak1)," &
   "114   (BC_4, SDA,      input,         X)," &
   "115   (BC_2, SCL,      output2,       1,    115,   1,   Weak1)," &
   "116   (BC_4, SCL,      input,         X)," &
   "117   (BC_2, XBE(3),   output2,       X)," &
   "118   (BC_2, XBE(2),   output2,       X)," &
   "119   (BC_2, XBE(1),   output2,       X)," &
-- num    cell   port   func          safe [ccell  dis  rslt]
   "120   (BC_2, XBE(0),   output2,       X)," &
   "121   (BC_2, SCKE,     output2,       X)," &
   "122   (BC_2, SCAS,     output2,       X)," &
   "123   (BC_2, SRAS,     output2,       X)," &
   "124   (BC_2, XDRAMW,   output2,       X)," &
   "125   (BC_2, XCAS(3),  output2,       X)," &
   "126   (BC_2, XCAS(2),  output2,       X)," &
   "127   (BC_2, XCAS(1),  output2,       X)," &
   "128   (BC_2, XCAS(0),  output2,       X)," &
   "129   (BC_2, XRAS(1),  output2,       X)," &
   "130   (BC_2, XRAS(0),  output2,       X)," &
   "131   (BC_4, TIN1,     input,         X)," &
   "132   (BC_4, TIN0,     input,         X)," &
   "133   (BC_2, TOUT0,    output2,       X)," &
   "134   (BC_2, TOUT1,    output2,       X)," &
   "135   (BC_4, XBG,      input,         X)," &
   "136   (BC_2, XBD,      output2,       X)," &
   "137   (BC_2, XBR,      output2,       X)," &
   "138   (BC_4, XIRQ1,    input,         X)," &
   "139   (BC_4, XIRQ3,    input,         X)," &
-- num    cell   port   func           safe [ccell  dis  rslt]
   "140   (BC_4, XIRQ5,    input,         X)," &
   "141   (BC_4, XIRQ7,    input,         X)," &
   "142   (BC_4, XRSTI,    input,         X)," &
   "143   (BC_2, XTS,      output3,       X,    189,   1,   Z)," &
   "144   (BC_4, XTS,      input,         X)," &
   "145   (BC_2, *,        controlr,      1)," &
   "146   (BC_2, XTA,      output3,       X,    145,   1,   Z)," &
   "147   (BC_4, XTA,      input,         X)," &
   "148   (BC_2, RW,       output3,       X,    189,   1,   Z)," &
   "149   (BC_4, RW,       input,         X)," &
   "150   (BC_2, XAS,      output3,       X,    189,   1,   Z)," &
   "151   (BC_4, XAS,      input,         X)," &
   "152   (BC_2, XCS(7),   output2,       X)," &
   "153   (BC_2, XCS(6),   output2,       X)," &
   "154   (BC_2, XCS(5),   output2,       X)," &
   "155   (BC_2, XCS(4),   output2,       X)," &
   "156   (BC_2, XCS(3),   output2,       X)," &
   "157   (BC_2, XCS(2),   output2,       X)," &
   "158   (BC_2, XCS(1),   output2,       X)," &
   "159   (BC_2, XCS(0),   output2,       X)," &
-- num    cell   port   func           safe [ccell  dis  rslt]
   "160   (BC_2, XOE,      output2,       X)," &
   "161   (BC_2, SIZ(1),   output3,       X,    189,   1,   Z)," &
   "162   (BC_4, SIZ(1),   input,         X)," &
   "163   (BC_2, SIZ(0),   output3,       X,    189,   1,   Z)," &
   "164   (BC_4, SIZ(0),   input,         X)," &
   "165   (BC_2, *,        controlr,      1)," &
   "166   (BC_4, PP(15),   input,         X)," &
   "167   (BC_2, PP(15),   output3,       X,    165,   1,   Z)," &
   "168   (BC_2, *,        controlr,      1)," &
   "169   (BC_4, PP(14),   input,         X)," &
   "170   (BC_2, PP(14),   output3,       X,    168,   1,   Z)," &
   "171   (BC_2, *,        controlr,      1)," &
   "172   (BC_4, PP(13),   input,         X)," &
   "173   (BC_2, PP(13),   output3,       X,    171,   1,   Z)," &
   "174   (BC_2, *,        controlr,      1)," &
   "175   (BC_4, PP(12),   input,         X)," &
   "176   (BC_2, PP(12),   output3,       X,    174,   1,   Z)," &
   "177   (BC_2, *,        controlr,      1)," &
   "178   (BC_4, PP(11),   input,         X)," &
   "179   (BC_2, PP(11),   output3,       X,    177,   1,   Z)," &
-- num    cell   port   func           safe [ccell  dis  rslt]
   "180   (BC_2, *,        controlr,      1)," &
   "181   (BC_4, PP(10),   input,         X)," &
   "182   (BC_2, PP(10),   output3,       X,    180,   1,   Z)," &
   "183   (BC_2, *,        controlr,      1)," &
   "184   (BC_4, PP(9),    input,         X)," &
   "185   (BC_2, PP(9),    output3,       X,    183,   1,   Z)," &
   "186   (BC_2, *,        controlr,      1)," &
   "187   (BC_4, PP(8),    input,         X)," &
   "188   (BC_2, PP(8),    output3,       X,    186,   1,   Z)," &
   "189   (BC_2, *,        controlr,      1)," &
   "190   (BC_2, *,        controlr,      1)," &
   "191   (BC_2, ADDR(23), output3,       X,    190,   1,   Z)," &
   "192   (BC_4, ADDR(23), input,         X)," &
   "193   (BC_2, ADDR(22), output3,       X,    190,   1,   Z)," &
   "194   (BC_4, ADDR(22), input,         X)," &
   "195   (BC_2, ADDR(21), output3,       X,    190,   1,   Z)," &
   "196   (BC_4, ADDR(21), input,         X)," &
   "197   (BC_2, ADDR(20), output3,       X,    190,   1,   Z)," &
   "198   (BC_4, ADDR(20), input,         X)," &
   "199   (BC_2, ADDR(19), output3,       X,    190,   1,   Z)," &
-- num    cell   port   func           safe [ccell  dis  rslt]
   "200   (BC_4, ADDR(19), input,         X)," &
   "201   (BC_2, ADDR(18), output3,       X,    190,   1,   Z)," &
   "202   (BC_4, ADDR(18), input,         X)," &
   "203   (BC_2, ADDR(17), output3,       X,    190,   1,   Z)," &
   "204   (BC_4, ADDR(17), input,         X)," &
   "205   (BC_2, ADDR(16), output3,       X,    190,   1,   Z)," &
   "206   (BC_4, ADDR(16), input,         X)," &
   "207   (BC_2, ADDR(15), output3,       X,    190,   1,   Z)," &
   "208   (BC_4, ADDR(15), input,         X)," &
   "209   (BC_2, ADDR(14), output3,       X,    190,   1,   Z)," &
   "210   (BC_4, ADDR(14), input,         X)," &
   "211   (BC_2, ADDR(13), output3,       X,    190,   1,   Z)," &
   "212   (BC_4, ADDR(13), input,         X)," &
   "213   (BC_2, ADDR(12), output3,       X,    190,   1,   Z)," &
   "214   (BC_4, ADDR(12), input,         X)," &
   "215   (BC_2, ADDR(11), output3,       X,    190,   1,   Z)," &
   "216   (BC_4, ADDR(11), input,         X)," &
   "217   (BC_2, ADDR(10), output3,       X,    190,   1,   Z)," &
   "218   (BC_4, ADDR(10), input,         X)," &
   "219   (BC_2, ADDR(9),  output3,       X,    190,   1,   Z)," &
-- num    cell   port   func           safe [ccell  dis  rslt]
   "220   (BC_4, ADDR(9),  input,         X)," &
   "221   (BC_2, ADDR(8),  output3,       X,    190,   1,   Z)," &
   "222   (BC_4, ADDR(8),  input,         X)," &
   "223   (BC_2, ADDR(7),  output3,       X,    190,   1,   Z)," &
   "224   (BC_4, ADDR(7),  input,         X)," &
   "225   (BC_2, ADDR(6),  output3,       X,    190,   1,   Z)," &
   "226   (BC_4, ADDR(6),  input,         X)," &
   "227   (BC_2, ADDR(5),  output3,       X,    190,   1,   Z)," &
   "228   (BC_4, ADDR(5),  input,         X)," &
   "229   (BC_2, ADDR(4),  output3,       X,    190,   1,   Z)," &
   "230   (BC_4, ADDR(4),  input,         X)," &
   "231   (BC_2, ADDR(3),  output3,       X,    190,   1,   Z)," &
   "232   (BC_4, ADDR(3),  input,         X)," &
   "233   (BC_2, ADDR(2),  output3,       X,    190,   1,   Z)," &
   "234   (BC_4, ADDR(2),  input,         X)," &
   "235   (BC_2, ADDR(1),  output3,       X,    190,   1,   Z)," &
   "236   (BC_4, ADDR(1),  input,         X)," &
   "237   (BC_2, ADDR(0),  output3,       X,    190,   1,   Z)," &
   "238   (BC_4, ADDR(0),  input,         X)";

end MCF5307;
```

17.8 OBTAINING THE IEEE 1149.1 STANDARD

The IEEE 1149 Standard JTAG specification is a copyrighted document and must be obtained directly from the IEEE:

IEEE Standards Department
445 Hoes Lane
P.O. Box 1331
Piscataway, NJ 08855-1331
USA

http://stdsbbs.ieee.org/

Fax: 908-981-9667
Information: 908-981-0060 or 1-800-678-4333