EZ-USB

Development Kit Manual

Getting Started

Rev 1.0

• Cypress Semiconductor • Personal Communications Division •
• 15050 Avenue of Science • Suite 200 • San Diego, CA 92128 •

Cypress Disclaimer Agreement

The information in this document is subject to change without notice and should not be construed as a commitment by Cypress Semiconductor Corporation Incorporated. While reasonable precautions have been taken, Cypress Semiconductor Corporation assumes no responsibility for any errors that may appear in this document.

Cypress Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Cypress Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Cypress Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Cypress Semiconductor and its officers, employees, subsidiaries, affiliates and distributors harmless against all claims, costs, damages, expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Cypress Semiconductor was negligent regarding the design or manufacture of the part.

The acceptance of this document will be construed as an acceptance of the foregoing conditions.

EZ-USB Development Kit Manual Getting Started, Version 1.0.

Table of Contents

Getting Started

1.0 EZ-USB Development Kit Overview

1.1 Introduction

The EZ-USB Development Kit (DVK) is the best starting point for developing an EZ-USB based product. The DVK includes everything you will need to get started: A development board, example firmware, a generic device driver, documentation, and assorted tools.

This manual provides a general overview and installation guide for the DVK. The software installation of the kit includes additional help files and documentation more specific to the various components in the kit.

The DVK is designed to work with the EZ-USB FX2LP and FX1 chips. FX1 is a full-speed only version of FX2LP. Other than the absence of a high-speed transceiver, FX1 is identical to FX2LP. Except where distinction is required, both chips will be generically referred to as EZ-USB throughout this manual.

1.2 EZ-USB Development Kit Contents

The following list shows the components supplied in the EZ-USB EZ-USB Development Kit. They represent most of the development tools required to build a USB system.

Hardware

- EZ-USB Advanced Development Board
- EZ-USB Proto-typing Board (“Breadboard”)
- USB Cable
- RS-232 Cable
- Software Installation CD-ROM

Software on CD-ROM

- EZ-USB Firmware Library and Firmware Frameworks
- Firmware Sample Code
- Cypress Generic USB Driver
- Cypress USB Class Library (CyApi)

- Cypress USB Console
- Cypress GPIF Designer
- Cypress Firmware Download Driver sample
- EZ-USB Documentation and Help Files
- Reference Schematics
- Limited Evaluation Version of the Keil 8051 Development Tools (Compiler, Assembler, IDE, Debugger)

1.2.1 Required Tools Not Included

- Full retail Keil Development System (Keil uVision2)
- Microsoft Visual C++ (all PC sample code is developed under this platform)
- USB capable PC running Windows XP, 2000, or ME

1.2.2 Other Suggested Tools

- CATC USB Protocol Analyzer

2.0 EZ-USB Development Kit Software

2.1 System Requirements

Minimum System Requirements

- Microsoft Windows XP, Windows 2000, or Windows ME
- Administrative privileges on the system
- 64 MBytes RAM (256 MBytes Recommended)
- Available Disk Space
  - 50 MBytes - full installation of DVK software
  - 50 MBytes - Keil Development Tools
- Pentium III-class PC (300 MHz or higher recommended)
- Super VGA display (resolution 800 x 600 or higher)
- USB Host Controller (full-speed or high-speed)
- Microsoft Developers Studio Version 6.0 or later (in order to compile Windows software examples)
- Keil uVision Version 2.38 or later 8051 Tools (in order to compile firmware examples)

2.2 Compatibility with Earlier EZ-USB Development Kits

This EZ-USB Advanced DVK has the ability to co-exist with older EZ-USB Development Kits. This DVK uses the same basic directory structure as the older Development Kits.

The only conflict with older development kits is with driver binding. The older DVK used a different device driver - the EZ-USB General Purpose Driver. That device driver is bound to the VID/PID of the EZ-USB FX2 chip: VID=0x04B4 and PID=0x8613. The EZ-USB FX2LP shares the same PID. This conflict only applies when the FX2LP enumerates in "Default USB Device" mode (i.e. RENUM bit is not set and there is no VID/PID stored in EEPROM). You can bind FX2LP to the new Generic USB driver by right-clicking on it in device manager and choosing "update driver". This should present you with two driver choices. Choose the driver where the description contains "FX2LP".

2.3 Installation Instructions

Start by collecting the following materials:

- Installation CD
- The EZ-USB Advanced Development Board
- USB A-B Cable
- A Development Platform (PC) with USB support

***Apple Macintosh Note**: The EZ-USB Development kit is designed to work only on a Windows based PC. If you are developing on a Mac, an application note titled "EZ-USB Development for Mac OS" is available on the Cypress Website (www.cypress.com)*

2.4 Verifying that the host PC supports USB

Almost all PCs manufactured after 1997 include full-speed USB host controller hardware. As of mid-2003, most but not all PCs include a USB 2.0 high-speed host controller. Drivers for full-speed host controllers have been distributed with Windows 98 and later versions of Windows. High-speed drivers are a different matter. Support for high-speed USB did not ship in the initial releases of any of the DVK supported OSs. High-speed drivers for the supported OSs are available as follows:

- Windows 2000 - Install Service Pack 4 or later
- Windows XP - Install Service Pack 1 or later
- Windows ME - For PCI add-in adapters, drivers may be available from the adapter's manufacturer. There is no driver support available for the Intel integrated high-speed host controller (i.e. ICH4). Microsoft has not and will not release high-speed drivers for Windows ME. For this reason, use of Windows ME is discouraged for high-speed peripheral development.

- Windows 98 and 98SE – For PCI add-in adapters, drivers may be available from the adapter's manufacturer. However, these versions of Windows are not supported by the EZ-USB DVK.

You can verify that high-speed drivers are installed and correctly bound to your host controller by looking under Device Manager. To open the device manager, run devmgmt.msc which is located in the Windows System32 Directory. After starting Device Manager, scroll down and expand the “Universal Serial Bus Host Controllers” section. If you see a device entry containing the string “Enhanced Host Controller”, then your system is configured correctly. If you cannot locate this entry, then it is likely that the host controller is not correctly bound to the high-speed driver. Look in the “Other Devices” section of the Device Manager for a USB Device marked with a yellow question mark. Right click on this entry and choose “Update Driver”. Then choose the option that allows Windows to locate the driver for you. If you have installed the required Service Pack, Windows will bind the controller to the appropriate device driver. You should now see the “Enhanced Host Controller” entry under “Universal Serial Bus Host Controllers”. For further help, contact your PC manufacturer.

2.5 Installing the Development Kit Software

Insert the installation CD into your CD ROM drive and run the program "setup.exe" by going to the start menu and running Setup.exe on the CD root directory.

For first time users, Cypress recommends accepting the default option at each stage of the Setup. Even though the Setup will allow you to change the destination folder for the installation, it is strongly recommended that you accept the default installation directory (c:\cypress\usb) as firmware examples and applications may be path dependent. Also, if you have already installed the full retail version of the Keil 8051 Development Tools on your system you should choose the “Do not install” option for the Keil tools.

3.0 Installing the Hardware

The hardware installation procedure is simple and goes as follows:

- Connect the EZ-USB Development board to a USB port on your development PC using the USB cable.
- When the OS finds the new USB device, it will notify you that it is installing the driver. The driver, which was installed by the Setup program, will be automatically located and loaded. Note that the Cypress Generic USB Driver is not a signed driver. Because of this, Windows may present a warning dialog box. Choose “Continue” on the warning dialog. For more information on driver signing go to the Microsoft website and search for “WHQL” (Windows Hardware Quality Labs) or consult the Windows DDK documentation.
- The green light (BKPT/Monitor) on the Development Board should illuminate, indicating that the 8051 Keil monitor code is loaded and running.

The driver, cyusb.sys, was automatically installed into the Windows\System32\Drivers directory during installation of the DVK software. An .INF file was also created in the Windows\INF directory

3.1 Confirm Successful Installation using the Cypress USB Console

Run the Cypress USB Console application and perform a "Get Device Descriptor" operation. The Console is added to the Windows Start menu by the DVK Setup program under:

```
Start->Cypress->USB->CyConsole – EZ-USB mode
```

This link starts the USB Console in EZ-USB mode.

After starting the USB Console, click on the “GetDev” button to retrieve the Device Descriptor from your Development Board. The USB Console should display the response from the Development Board shown below. The “idVendor” (VID) value of 0x04B4 is the Cypress Semiconductor vendor ID, and the “idProduct” (PID) value of 0x0082 identifies EZ-USB the FX2LP Development Board. The PID will be different if using an FX1 Development Board. The important thing here is that the board responds to the request and that the VID is 0x04B4. You may wish first to clear the screen by clicking the “Clear” button. The “GetDev” button may be clicked anytime, as many times as you wish.

Opened Driver Successfully
Device Descriptor:
bLength: 18
bDescriptorType: 1
bcdUSB: 256
bDeviceClass: 0xff
bDeviceSubClass: 0xff
bDeviceProtocol: 0xff
bMaxPacketSize0: 0x40
idVendor: 0x4b4
idProduct: 0x0082
bcdDevice: 0x1
iManufacturer: 0x0
iProduct: 0x0
iSerialNumber: 0x0
bNumConfigurations: 0x1

More detailed instructions on using the Cypress USB Console are available under the Console’s HELP menu. There is also tutorial in the EZ-USB DVK Users Guide that demonstrates many of the Console’s features. A link to the Users Guide is added to the Windows Start menu by the DVK Setup.

4.0 EZ-USB Advanced Development Board

4.1 Introduction

The Cypress Semiconductor EZ-USB Advanced Development Board provides a compact evaluation and design vehicle for the EZ-USB family. The board provides expansion and interface signals on six 20-pin headers. A mating prototype board allows quick construction and testing of USB designs. All ICs on the board operate at 3.3 volts. The board may be powered from the USB connector or an external power supply.

The EZ-USB Advanced Development Board is supplied as part of the Cypress Semiconductor EZ-USB Development Kit, which includes an evaluation version of Cypress-customized software development tools from Keil Software Inc. The Keil 8051 assembler, C compiler and debugger work in concert with the development board to provide a complete code development environment. An 8051 debug monitor is typically loaded into development board expansion RAM to leave the internal RAM free for code development. The evaluation version of the Keil tools that ships with the DVK has several restrictions that make it inappropriate for real-world development. Most significantly, it limits the compiled object size to 4 Kbytes. The full retail version allows code of any size.

4.2 Schematic Summary

This description should be read while referring to the EZ-USB FX2LP Development Board Schematic and the FX2LP Development Board Assembly drawing. Both drawings are attached to the end of this document and are available in PDF format in the DVK hardware directory. With the exception of the EZ-USB chip, the development boards in the FX2LP and FX1 DVKs are identical and will be referred to generically as the EZ-USB Development Board in the following sections.

U1 is either EZ-USB FX2LP (CY7C68013A-128AC) or FX1 (CY7C64713-128AC). This is the full-function EZ-USB chip which brings out the 8051 address and data buses for external memory expansion. U2, a re-programmable GAL, provides RAM enable signals for four jumper-selectable memory maps. U3 is a 128 Kilobyte RAM, used for external 8051 memory expansion. Only 64K of this memory is addressed by the 8051; the A16 pin is connected to a pull-up resistor that may be attached to a GAL output to provide bank switching options.

U4 is a 3.3V, 500 milliamp voltage regulator. U5 and U6 are socketed EEPROMS, used for EZ-USB initialization and 8051 general purpose access. U7 converts the 3.3V 8051 serial port signals to bipolar RS-232 levels. U8 and U10 are Philips PCF8574 IO expanders, which attach to the EZ-USB I^2C bus and provide 8 general purpose input-output pins. U10 reads the four push-button switches S2-S5, and U11 drives the seven-segment readout U9.

Six 20-pin headers, P1-P6, provide interface signals to the plug-in proto-typing board supplied in this kit, as well as serving as connection points for HP(Agilent) logic analyzer pods. P8 contains a subset of signals from P1-P6 on a connector that is pinned out for connection to a ‘straight-through’ ATA cable.

Two slide switches, SW1 and SW2, control the connection and selection of the two socketed EEPROMS at U5 and U6.

4.3 Jumpers

Table 1. EZ-USB Development Board Jumpers

Jumper	Function	Default	Notes
JP1	Connects 3.3 volt power to the EZ-USB chip.	IN (1-2)	
JP2	Powers the on-board 3.3 volt regulator from USB Vbus pin	IN (1-2)	To operate the board in self-powered mode, remove JP2 and supply 4-5V to JP2-1, and GND to a ground pin (TP1 is a convenient GND point).
JP3	Connects four GAL pins to LEDS D2, D3, D4, D5	IN (1-2) (3-4) (5-6) (7-8)	U2, the on-board GAL, contains code to use the four LEDs as software settable indicators. If you wish to (a) use the GAL pins for something else, which requires re-programming the GAL or (b) wire the general purpose indicators D2-D5 to other parts of the board, first remove the appropriate shorting plug(s).
JP5	3.3 Volt Power	IN (1-2)	Supplies 3.3 volt power to the board. It may be removed and replaced with ammeter probes in series to measure board current.
JP6 JP7	Memory map selection	OUT (1-2)	These jumpers select one of four memory maps for U3, the external 128 Kilobyte RAM. See the Memory Map section for details.
JP8	Wakeup2 Pin	OUT (1-2)	Inserting a shorting plug into JP8 connects an on-board RC network (R42,C43) to the secondary remote wakeup pin WU2. This R-C network can be used to test periodic remote wakeup firmware when this dual-purpose pin (it defaults to PA3) is programmed as WU2.
JP9	I2C bus test points	n/a	The I2C bus SCL and SDA lines may be monitored or externally connected using JP9.

4.4 EEPROM Select & Enable Switches SW1 and SW2

SW1 selects between two socketed EEPROMS, one strapped to address 000 (U6), and the other strapped to address 001(U5).

SW2 enables or disables the EEPROM selected by SW1.

The EZ-USB chip has various start-up modes, which depend on the existence of an EEPROM connected to its SCL and SDA lines. Switches SW1 and SW2 allow the EEPROMS to be “disconnected” from FX2, or to be connected using one of two EEPROMS installed in sockets U5 and U6.

The EZ-USB chip contains two I^2C controllers, a “boot load” controller, and an 8051 controller. The boot load controller operates when EZ-USB comes out of reset, and the 8051 controller operates under firmware control once the 8051 is running, permitting the 8051 to access general- purpose I^2C devices connected to the SCL and SDA lines. The discussion below deals with the roles of SW1 and SW2 in accommodating the various boot load mechanisms.

The EZ-USB boot loader accommodates two EEPROM types, in “Small” and “Large” versions, as shown by Table 2.

Table 2. Typical EZ-USB external EEPROMS

EEPROM Type	Size	A2A1A0	Typical P/N
"Small"	16x8	000	24LC00
	128x8	000	24LC01
	256x8	000	24LC02
"Large"	8Kx8	001	24LC64/5

"Small" EEPROMS are typically used to supply custom VID and PID information, allowing the EZ-USB to enumerate with a driver associated with your EZ-USB design.

"Large" EEPROMS are typically used to boot-load code into internal EZ-USB RAM, and then start up the 8051 to execute this internal code which performs the enumeration.

The EZ-USB loader determines the EEPROM size by first initiating an I^2C transfer to address 1010000 (1010 is the EEPROM class address, and 000 is the sub-address). If the device supplies an I^2C acknowledge pulse, the EZ-USB loader writes a single EEPROM address byte to initialize the internal EEPROM address pointer to zero.

If the above transfer does not return an ACK pulse, the EZ-USB loader initiates a second I^2C transfer, this time to address 10100001 (1010=EEPROM, sub-address 001). If an ACK is returned by the I^2C-device, the EZ-USB loader writes two EEPROM address bytes to initialize the internal EEPROM address pointer to 0.

If neither transfer returns an ACK pulse, the EZ-USB loader boots in 'generic' mode (explained below).

Tree EZ-USB startup sequences, and the associated settings for SW1 and SW2, are shown below.

1. **Generic':** SW2=OFF, SW1=either position

   When no EEPROM is connected to SCL and SDA, the EZ-USB chip enumerates using its internal, "hard-wired" VID and PID values. This mode can be selected without requiring the removal of any socketed EEPROMS by switching SW2 to the 'off' (down) position. This electrically disconnects any EEPROMS that occupy the EEPROM sockets U5 and U6. The "OFF" mode is handy for starting up EZ-USB in a manner (using internal VID/PID) that binds the development system board to the Cypress debug tools such as the Control Panel and Keil debug tools. Once running, SW2 can be switched to the ON position to allow 8051 access, for example to re-program the EEPROM.

2. **'C0 Load':** SW2=ON, SW1=SMALL

   A "C0" load provides EZ-USB with external VID, PID and DID values, allowing it to enumerate with the EEPROM-supplied VID/PID/DID.

   At power-on, if the EZ-USB chip detects an EEPROM with the hex value 'C0' as its first byte, it continues to load seven additional EEPROM bytes, which correspond to the USB Vendor ID (VID), Product ID (PID), Device ID (DID), and an EZ-USB configuration byte. Then when EZ-USB enumerates, it uses these EEPROM values rather than the hard-wired internal values.

   Since only eight bytes of data are required, a 'small' EEPROM is generally used for this mode, for example the 16-byte 24LC00.

3. **'C2 Load':** SW2=ON, SW1=LARGE

   A "C2" load provides a method for loading the EZ-USB internal RAM with 8051 firmware before enumeration. This 'boot load' mechanism allows EZ-USB to enumerate as a fully custom device, since the 8051 code handles enumeration using VID/PID values embedded in the code.

   At power-on, if the EZ-USB chip detects an EEPROM with the hex value 'C2' as its first byte, it continues to load an EZ-USB configuration byte, followed by blocks of 8051 code. The last byte loaded takes the 8051 out of reset.

   This mode usually requires a large EEPROM, such as the 8 Kilobyte 24LC64.

NOTE: If an EEPROM is connected to the SCL and SDA lines, but does not contain 0xC0 or 0xC2 as its first byte, the loader reverts to case 1, 'generic'. In other words, the boot loader operates as though no EEPROM is connected. Once the 8051 is running, however, it has full access to any connected EEPROM, since the 8051 I^2C controller is completely independent of the boot load logic.

4.5 Interface Connectors

Table 3. Logic Analyzer Pinout

Agilent 01650-63203 Pod Pins			
CLK1	**3**	**4**	D15
D14	**5**	**6**	D13
D12	**7**	**8**	D11
D10	**9**	**10**	D9
D8	**11**	**12**	D7
D6	**13**	**14**	D5
D4	**15**	**16**	D3
D2	**17**	**18**	D1
D0	**19**	**20**	GND

Six 20-pin headers P1-P6 on the EZ-USB Development Board have pins assigned to be compatible with HP (Agilent) logic analyzers, as shown in Table 3. The slight bulge in the middle rows of the table (pins 9 and 11) indicates the connector key.

The six headers P1-P6 serve three purposes.

- They mate with the proto-typing board supplied in the EZ-USB Development Kit.
- They allow direct connection of HP (Agilent) Logic Analyzer pods (Agilent P/N 01650-63203).
- They allow general purpose probing by other logic analyzers or oscilloscopes.

Table 3 shows the logic analyzer pod pin designations. The EZ-USB signals on P1-P6 are arranged to fulfill the following requirements:

- High speed EZ-USB strobe signals (PSEN, WR#, CLKOUT, IFCLK, and RD#) are connected to pin 3 of each of the five connectors P1-P6, so that they may be used as the logic analyzer clock CLK1.
- CLK2 is not used. Instead, each connector brings 3.3V power from the EZ-USB Development Board up to the prototype board using pin 2.
- The signals are logically grouped. For example, the 8051 address bus is on P5, and the EZ-USB FIFO data (which shares PORTB and PORTD pins) is on P1.

Because the 20-pin headers on the proto-typing board are stackable, it is possible to build custom circuitry on the proto board, plug the board into the EZ-USB Development board, and still plug logic analyzer pods into the six connectors P1-P6.

Tables 4-9 show the EZ-USB pin designations for P1 through P6. For dual-mode pins, the power-on default signal names are shown in bold type, and the alternate pin names are shown in the outside columns.

Table 4.

Alternate	Default	P1		Default	Alternate
	N.C.	**1**	**2**	**3.3V**	
	PSEN#	**3**	**4**	**PD7**	FD[15]
FD[14]	**PD6**	**5**	**6**	**PD5**	FD[13]
FD[12]	**PD4**	**7**	**8**	**PD3**	FD[11]
FD[10]	**PD2**	**9**	**10**	**PD1**	FD[9]
FD[8]	**PD0**	**11**	**12**	**PB7**	FD[7]
FD[6]	**PB6**	**13**	**14**	**PB5**	FD[5]
FD[4]	**PB4**	**15**	**16**	**PB3**	FD[3]
FD[2]	**PB2**	**17**	**18**	**PB1**	FD[1]
FD[0]	**PB0**	**19**	**20**	**GND**	

Table 5.

Alternate	Default	P2		Default	Alternate
	N.C.	**1**	**2**	**3.3V**	
	N.C.	**3**	**4**	**RDY1**	SLWR
SLRD	**RDY0**	**5**	**6**	**CTL5**	
	CTL4	**7**	**8**	**CTL3**	
FLAGC	**CTL2**	**9**	**10**	**CTL1**	FLAGB
FLAGA	**CTL0**	**11**	**12**	**PA7**	FLAGD
PKTEND	**PA6**	**13**	**14**	**PA5**	FIFOADR1
FIFOADR0	**PA4**	**15**	**16**	**PA3**	WU2
SLOE	**PA2**	**17**	**18**	**PA1**	INT1#
INT0#	**PA0**	**19**	**20**	**GND**	

Table 6.

Alternate	Default	P3		Default	Alternate
	N.C.	1	2	3.3V	
	WR#	3	4	RDY5	
	RDY4	5	6	RDY3	
	RDY2	7	8	BKPT	
	RESET#	9	10	N.C.	
	N.C.	11	12	PC7	GPIFADR7
GPIFADR6	PC6	13	14	PC5	GPIFADR5
GPIFADR4	PC4	15	16	PC3	GPIFADR3
GPIFADR2	PC2	17	18	PC1	GPIFADR1
GPIFADR0	PC0	19	20	GND	

Table 7.

Alternate	Default	P4		Default	Alternate
	N.C.	1	2	3.3V	
	CLKOUT	3	4	GND	
	OE#	5	6	CS#	
	5V	7	8	5V	
	PLD2	9	10	PLD1	
	N.C.	11	12	D7	
	D6	13	14	D5	
	D4	15	16	D3	
	D2	17	18	D1	
	D0	19	20	GND	

Table 8.

Alternate	Default	P5		Default	Alternate
	N.C.	1	2	3.3V	
	IFCLK	3	4	A15	
	A14	5	6	A13	
	A12	7	8	A11	
	A10	9	10	A9	
	A8	11	12	A7	
	A6	13	14	A5	
	A4	15	16	A3	
	A2	17	18	A1	
	A0	19	20	GND	

Table 9.

Alternate	Default	P6		Default	Alternate
	N.C.	**1**	**2**	**3.3V**	
	RD#	**3**	**4**	**INT5#**	
	INT4	**5**	**6**	**T2**	
	T1	**7**	**8**	**T0**	
	WAKEUP#	**9**	**10**	**SDA**	
	SCL	**11**	**12**	**PE7**	GPIFADR8
T2EX	**PE6**	**13**	**14**	**PE5**	INT6
RxD1OUT	**PE4**	**15**	**16**	**PE3**	RxD0OUT
T2OUT	**PE2**	**17**	**18**	**PE1**	T1OUT
T0OUT	**PE0**	**19**	**20**	**GND**	

4.6 ATA Connector P8

Table 10 shows the pinout for P8, a 40-pin connector that interfaces with a standard ATA cable. Note: This is for ATA use only. SP1, 2, and 3 should be bridged with solder to connect the appropriate pull-up/down resistors required for ATA. An 80-pin cable is required for UDMA transfer modes and recommended for all transfer modes.

Table 10.

P8 (ATA)					
RESET#	**PA7**	**1**	**2**	**GND**	GND
DD7	**PB7**	**3**	**4**	**PD0**	DD8
DD6	**PB6**	**5**	**6**	**PD1**	DD9
DD5	**PB5**	**7**	**8**	**PD2**	DD10
DD4	**PB4**	**9**	**10**	**PD3**	DD11
DD3	**PB3**	**11**	**12**	**PD4**	DD12
DD2	**PB2**	**13**	**14**	**PD5**	DD13
DD1	**PB1**	**15**	**16**	**PD6**	DD14
DD0	**PB0**	**17**	**18**	**PD7**	DD15
GND	**GND**	**19**	**20**	**N.C.**	KEYPIN
DMARQ	**RDY1**	**21**	**22**	**GND**	GND
DIOW#	**CTL0**	**23**	**24**	**GND**	GND
DIOR#	**CTL1**	**25**	**26**	**GND**	GND
IORDY	**RDY0**	**27**	**28**	**GND**	CSEL
DMACK#	**CTL2**	**29**	**30**	**GND**	GND
INTRQ	**PA0**	**31**	**32**	**N.C.**	RESERVED
DA1	**PA2**	**33**	**34**	**N.C.**	PDIAG#
DA0	**PA1**	**35**	**36**	**PA3**	DA2
CS0#	**PA4**	**37**	**38**	**PA5**	CS1#
DASP#	**10K Pull-up**	**39**	**40**	**GND**	GND

4.7 U2 -- 22v10 GAL

A standard 22v10 GAL provides general purpose “glue logic” on the board. It provides the AND gate required to combine the PSEN and READ signals, adds memory map support, debug LEDs and provides three spare outputs for customer defined functions.

4.8 Memory Maps

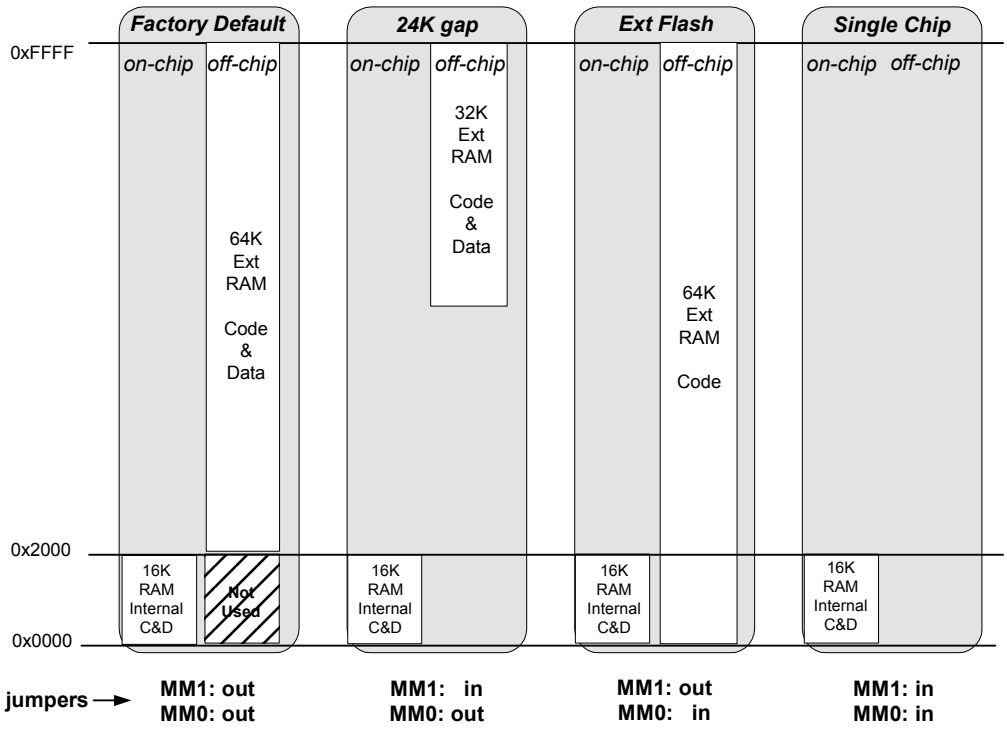


Figure 1-1. The four EZ-USB Dev Board memory maps. The GAL sets EA=1 for the Ext Flash configuration only, enabling external code memory.

The factory default is to have both MM1 and MM0 jumpers removed. This setting should be used for all development work using the Keil software tools.

– The default configuration provides 16 Kilobytes of on-chip code and data memory, plus 48 Kilobytes of external RAM. The 8051 begins execution from internal RAM (the GAL sets EA=0). Note that even though there is an apparent overlap between the internal 16 Kilobytes and bottom 16 Kilobytes of the external RAM, EZ-USB disables RAM strobes for the bottom 16 Kilobytes, so there is no conflict. This EZ-USB decoding allows a standard 64 Kilobyte RAM to be used without requiring external decoding to inhibit access to the bottom 16 Kilobytes.

– The second column, “24K gap”, enables the external RAM only for access to its upper 32 Kilobytes. This configuration is useful for systems that add external devices that require memory-mapped access. As with the default configuration, the 8051 begins execution from internal RAM (the GAL sets EA=0).

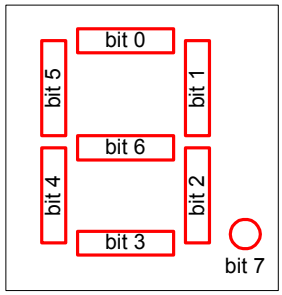
– The third column, “Ext Flash”, allows a flash memory (or other ROM) to be connected to the 8051 bus. This is the only configuration that starts 8051 execution from external memory (the GAL sets EA=1). Since external memory occupies the bottom 16K, the internal EZ-USB RAM is addressed only as data memory, instead of combined prog/data memory in the other three configurations.

– The fourth column, “Single Chip”, disables all external memory. This configuration is useful for testing final code to insure that it uses no external resources present in the development environment.

4.9 I^2C Expanders

U8 and U10 are Philips PCF8574 I/O expanders. They connect to the I^2C bus SCL and SDA pins, and provide 8 general-purpose input-output pins. U8 provides 8 output bits, connected to the seven-segment readout U9. U10 provides 8 input bits, four of which connect to push buttons S2-S5, and four of which are uncommitted.

U8 connects to the 7-segment readout (U9) using the following bit assignments:



U8 has the group address 0100, and is strapped to unit address 001. Therefore to write a value to the 7-segment readout, 8051 firmware sends a control byte of 01000010 (the LSB indicates a write operation), followed by the data byte.

U10 uses its I/O pins as inputs, connected to S2-S5 according to the following table:

Bit	Switch
0	S2
1	S3
2	S4
3	S5

U9 has the group address 0100, and is strapped to unit address 000. Therefore to read the switch values, 8051 firmware sends a control byte of 01000001 (the LSB indicates a read operation), and then reads the data byte.

4.10 Indicators—Power and Breakpoint

LED D1 is connected to the PCB 5 volt supply, which is normally supplied from the USB cable (VBUS pin). Alternatively, JP2 may be removed and external 5 volt power can be applied to JP2 pin 1. In either case, D1 indicates the presence of the 5 volt power.

LED D6 is connected to the 3.3 volt voltage regulator output.

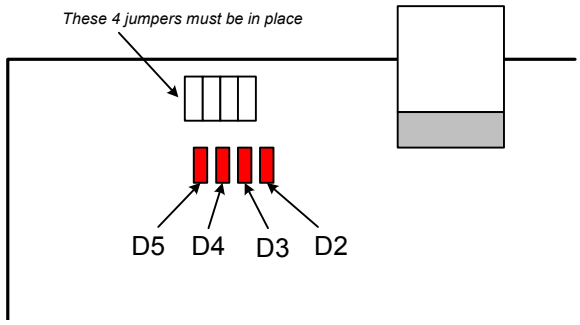
LED D7 is connected to the EZ-USB BKPT (Breakpoint) pin. When using the Keil software development tools, this green LED indicates that the EZ-USB Development Board has enumerated, and the Keil monitor has loaded and started running.

4.11 General Purpose Indicators

A portion of the GAL (U2) decodes 8051 reads to certain external memory addresses to turn the four general-purpose indicators D2-D5 on and off. The following figure shows the positions of the four indicator LEDS, and a table of the external 8051 addresses which are read to turn them on and off. Note that the four jumpers above the LEDS must be installed to use this feature. These jumpers connect the LEDS to four GAL outputs.

Note: The CLKOUT signal is used as a clock to latch the LED output signals from the GAL. If CLKOUT is disabled, the LEDs will not update.

NOTE: If you wish to use the LEDS for other purposes, for example to wire to other PC board signals for observation, first remove the shorting plug to disconnect the LED from the GAL. The LED terminal is the bottom pin of the connector, and the GAL I/O pin is the top pin.



Indicator	Turn ON by reading	Turn OFF by reading
D2	0x88--	0x80--
D3	0x98--	0x90--
D4	0xA8--	0xA0--
D5	0xB8--	0xB0--

The low address byte is “don’t care”. This means you can very efficiently add software test points using the following code:

```
D5ON:     mov MPAGE,#B8h            ; turn D5 on
          movx a,@r0                ; dummy read
;
D5OFF:    mov MPAGE,#B0h            ; turn D5 off
          movx a,@r0                ; dummy read
```

This code example uses the 8051 8-bit indirect addressing mode. The MPAGE register (SFR 0x92) supplies the high address byte, and r0 supplies the low address byte. Register r0 does not require initialization since the low address byte is “don’t care” for the LED decoding.

To turn the LEDS on and off using C code, declare the external memory locations, and then read their values into dummy variables:

```
xdata volatile unsigned char D5ON         _at_ 0xB800;
xdata volatile unsigned char D5OFF        _at_ 0xB000;

unsigned char dum;

dum = D5ON;       // turn D5 on
dum = D5OFF;      // turn D5 off
```

Note: Program execution at these addresses will NOT activate the LEDs.

5.0 Frequently Asked Questions

Q1: What should I do first (after viewing the printed material from the box)?

A1: Make sure the hardware works well enough to run the Tutorial. After software installation, plug in a Dev Board, and go through the DVK tutorial. The tutorial is located in the “EZ-USB DVK User’s Guide“ which can be found in the “Start“ menu under Cypress >usb >help. The tutorial is short and worthwhile.

Q2: What is the first example to try?

A2: While following the tutorial, you will read the Device ID from the Development Board, and then load the dev_io example. It turns on the LED so you know that firmware has been loaded and runs OK.

Q3: Where's a soft copy of the EZ-USB Getting Started?

A3: Look at "C:\CYPRESS\USB\Doc\General\EZ-USB Getting Started.PDF".

Q4: Where's a soft copy of the EZ-USB Technical Reference Manual?

A4: Look at "C:\CYPRESS\USB\Doc\FX2LP\EZ-USB_TRM.pdf". This is a very well written and readable manual, and is the key reference you will want to use. Bring up the soft copy so you can search for the material you are most interested in very quickly.

Q5: Where's the EZ-USB datasheet?

A5: See "C:\CYPRESS\USB\Doc\FX2LP\CY7C68013.pdf".

Q6: Where's the EZ-USB Development Board schematic (pdf and Orcad files)?

A6: See "C:\CYPRESS\USB\Hardware\FX2LP\DEVBD REVx\" where "x" is the latest Rev.

Q7: Where can I find the errata sheet(s)?

A7: Look at: "C:\CYPRESS\USB\Doc\FX2LP\EZ-USB beta release notes.pdf" and "C:\CYPRESS\USB\Doc\FX2LP\EZ-USB Chip Errata.pdf"

Q8: How do I to generate "myapp" from (frameworks)?

A8: Create a (frameworks based) project folder by just copying the "fw" example folder (C:\CYPRESS\USB\Target\Fw\LP) to a new location (i.e. under "Examples") and then rename the folder to the new project name. Remove the .hex file, and .Uv2 file. Rename periph.c to <NewPrj>.c, and then create a new uV2 project file. See: the “EZ-USB Firmware Frameworks” section of the “EZ-USB DVK User’s Guide” for more information.

Q9: How do I build an EEPROM image for burning my code?

A9: See the tutorial for information about generating EEPROMs.

Q10: Where can I get a summary of the registers?

A10: See the register summary in the TRM.

Q11: Are there any examples?

A11: Yes, see the examples and readme files in the Examples folder.

Q12: What's all this set environment stuff?

A12: If you install into the default directory, "C:\CYPRESS\USB" then you can build and debug examples with the Keil uV2 project files provided. Since the project files have hard-coded paths in them, installing to a different, non-default directory location will break these project files. Also, there are build.bat files for the projects in the Example folders. In order to run the build.bat files from the command line, you need to set some paths and environment variables, which can be done for you by running the batch file C:\CYPRESS\USB\Bin\setenv.bat prior to typing "build". Again, if the Dev Kit software or Keil tools are installed to a non-default location, you would need to modify the setenv.bat file. The setenv.bat also has directions on how to create a Start menu option to open an MS-DOS window with the correct environment set up.

Q13: Which DB-9 do I plug my mon-51 cable into?

A13: Use SIO-1 by default. There are other versions of the monitor in "C:\CYPRESS\USB\Target\Monitor". They can be loaded using the Control Panel. There are different versions which load internal or external, and use SIO-0 or SIO-1, as indicated by the name and the readme.txt file.

6.0 Appendix A: U2 (GAL) code (file is 'FX2LP.ABL')

```
MODULE fx2lp

" Swapped dipswitch settings 00 and 10 on 4-3-98 to allow the all-switch-
on default

   x,c,z  = .X.,.C.,.Z.;

"Inputs
   A12,A13,A14,A15                  pin 11,12,13,16;
   A11                              pin 4;
   nRD,nPSEN,CLKOUT                 pin 6,5,2;
   mm1,mm0                          pin 9,7;
"Outputs
   EA,nRAMOE,nRAMCE                 pin 21,25,27;
   PF0,PF1,PF2,PF3                  pin 17,18,19,20 istype 'reg_sr';

modesw = [mm1,mm0];                          " two dipswitches
addr = [A15,A14,A13,A12,A11,nRD];" high nibble of the address bus + RD

equations
" The 3681 board turns PF0 on at 0x80xx reads and off at 0x81xx reads.
" This board turns PF0 on at 0x8xxx reads and off at 0x88xx reads.
PF0.S = (addr == ^b100000);
PF0.R = (addr == ^b100010);
PF0.CLK = CLKOUT;

PF1.S = (addr == ^b100100);
PF1.R = (addr == ^b100110);
PF1.CLK = CLKOUT;

PF2.S = (addr == ^b101000);
PF2.R = (addr == ^b101010);
PF2.CLK = CLKOUT;

PF3.S = (addr == ^b101100);
PF3.R = (addr == ^b101110);
PF3.CLK = CLKOUT;

WHEN      (modesw == 00) THEN" No external memory
{
nRAMCE =  1;
nRAMOE=   1;
EA =      0;
}
ELSE WHEN(modesw == 01) THEN" Ext P&D mem at 8000 (can add mem to 0-8K)
{
!nRAMCE=  A15;
!nRAMOE=  !nRD # !nPSEN;" Combine program & data memory
EA =      0;
}
ELSE WHEN(modesw == 11) THEN" Ext P&D mem at 0000 and 8000
{
!nRAMCE = 1;
```

```
!nRAMOE=  !nRD # !nPSEN;
EA =      0;
}
ELSE WHEN(modesw == 10) THEN" All program mem external
{
!nRAMCE = 1;
!nRAMOE =!nRD # !nPSEN;
EA =      1;
}

test_vectors

([mm1,mm0,A15,nRD,nPSEN] -> [nRAMCE, nRAMOE, EA])
 [ 0 , 0 , x , x ,  x  ] -> [ 1  ,  1 ,  0];" 10: all mem selects and
strobes OFF

 [ 0 , 1 , 0 , 1 ,  1  ] -> [ 1  ,  1 ,  0];" 01: top of mem for rd or
psen
 [ 0 , 1 , 1 , 1 ,  0  ] -> [ 0  ,  0 ,  0];" PSEN only
 [ 0 , 1 , 1 , 0 ,  1  ] -> [ 0  ,  0 ,  0];" RD only
 [ 0 , 1 , 1 , 1 ,  1  ] -> [ 0  ,  1 ,  0];" Neither RD or PSEN

 [ 1 , 1 , 0 , 1 ,  0  ] -> [ 0  ,  0 ,  0];" 11: top and bot mem for rd
or psen
 [ 1 , 1 , 0 , 0 ,  1  ] -> [ 0  ,  0 ,  0];
 [ 1 , 1 , 0 , 1 ,  1  ] -> [ 0  ,  1 ,  0];
 [ 1 , 1 , 1 , 1 ,  0  ] -> [ 1  ,  0 ,  0];" PSEN
 [ 1 , 1 , 1 , 0 ,  1  ] -> [ 1  ,  0 ,  0];" RD
 [ 1 , 1 , 1 , 1 ,  1  ] -> [ 1  ,  1 ,  0];" neither

 [ 1 , 0 , 1 , 1 ,  0  ] -> [ 1  ,  0 ,  1];" PSEN
 [ 1 , 0 , 1 , 0 ,  1  ] -> [ 1  ,  0 ,  1];" RD
 [ 1 , 0 , 1 , 1 ,  1  ] -> [ 1  ,  1 ,  1];" neither

test_vectors

([nRD,nPSEN] -> [nRAMOE])
 [ 0 , 0   ] -> [ 0  ];
 [ 0 , 1   ] -> [ 0  ];
 [ 1 , 0   ] -> [ 0  ];
 [ 1 , 1   ] -> [ 1  ];

 test_vectors
 (addr -> [PF0, PF1, PF2, PF3])
 [1,0,0,0,0,0] -> [0, 0, 0, 0];
 [1,0,0,0,1,0] -> [1, 0, 0, 0];

END
```

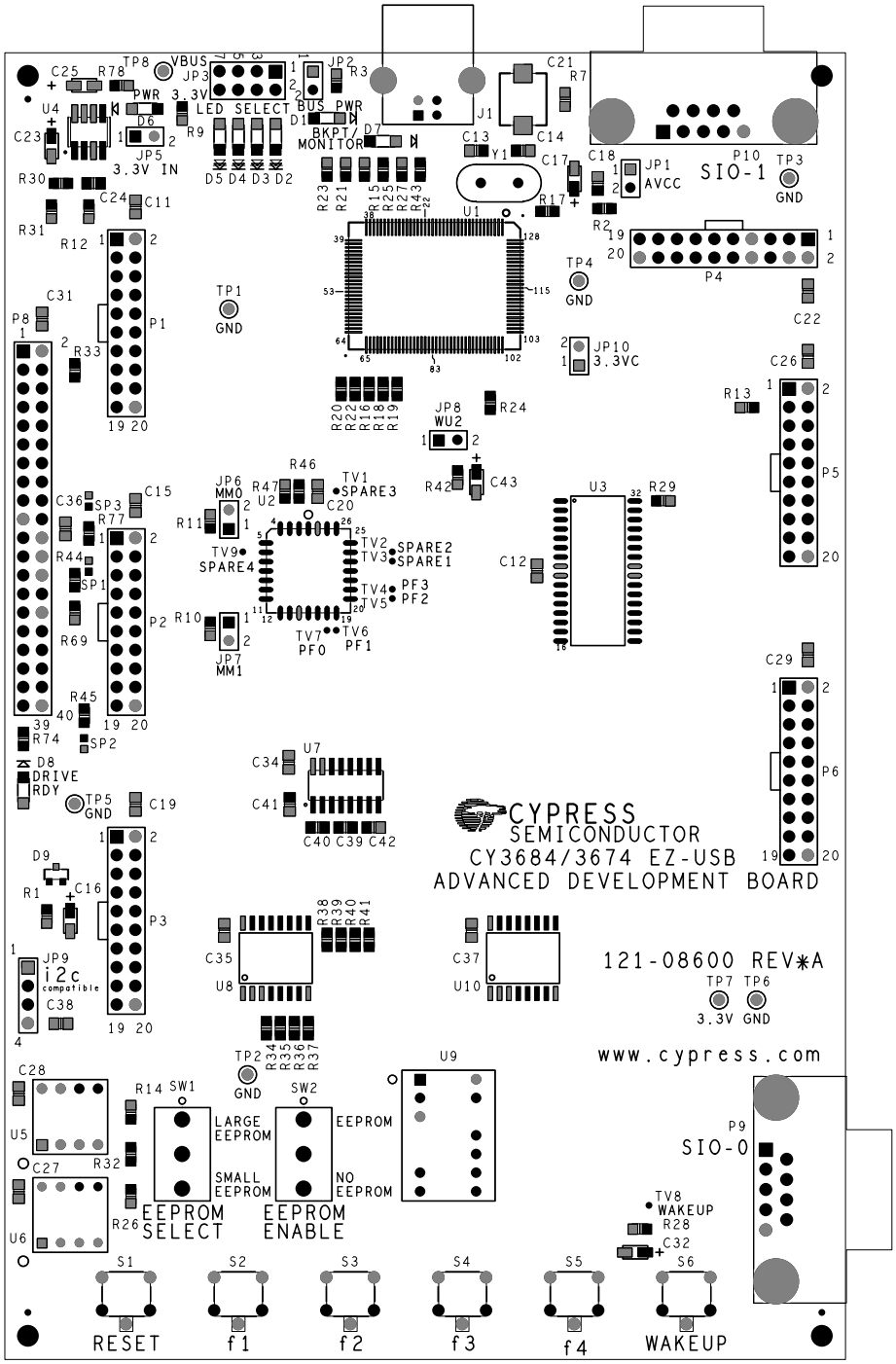
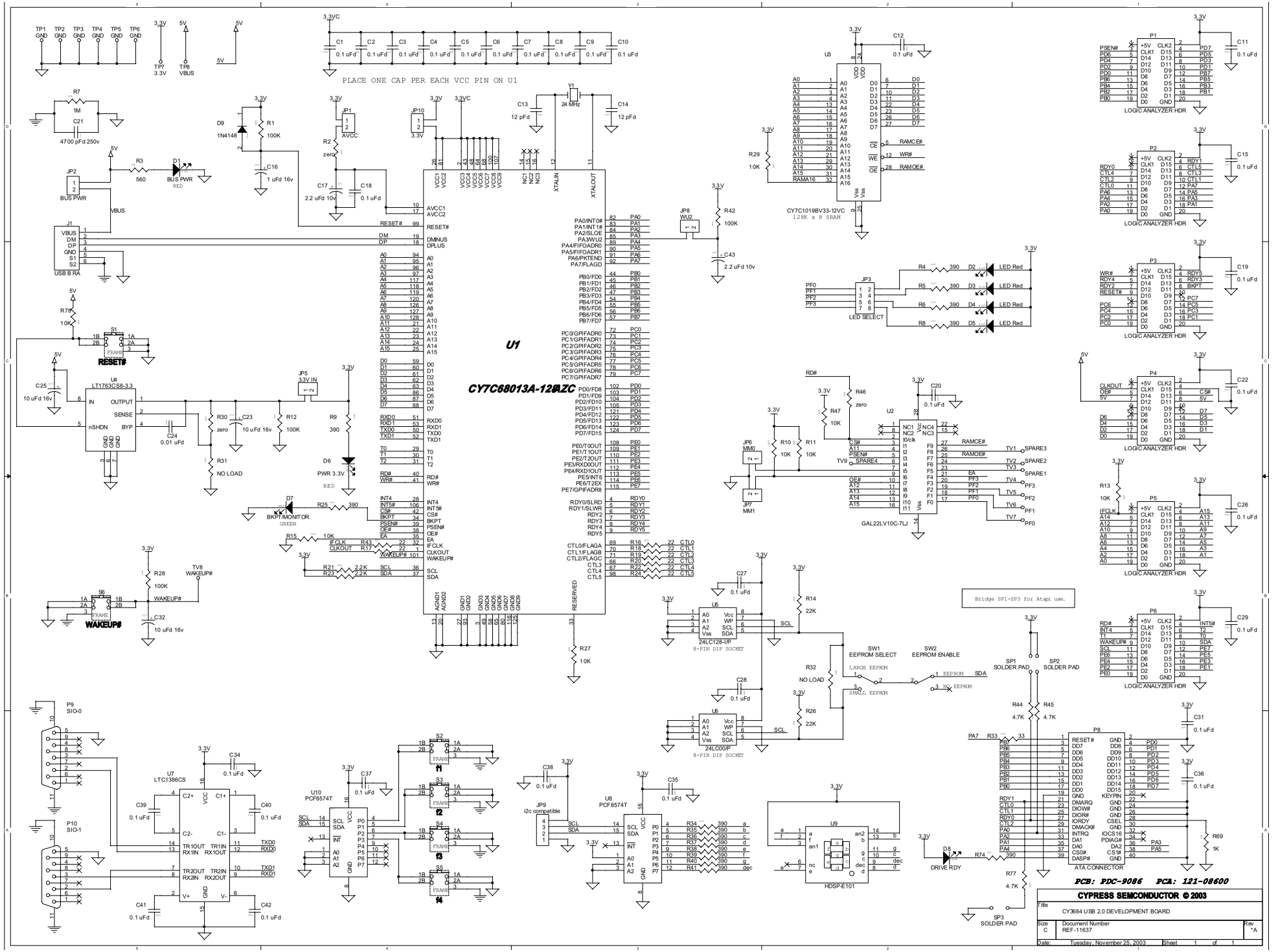


Figure 1-2. CY3684 Assembly Development Board Layout

The schematic for the CY3684 appears on the following page.


PLACE ONE CAP PER EACH VCC PIN ON U1
U1
CY7C68013A-128AXC
CY7C1019BV33-12VC
128K x 8 SRAM
LOGIC ANALYZER HDR
LED Red
LED SELECT
RESET#
WAKEUP#
BUS PWR
USB B RA
PWR 3.3V
BKPT/MONITOR
DRIVE RDY
LTC1763CS8-3.3
LTC1386CS
24LC128-I/P
24LC00/P
8-PIN DIP SOCKET
GAL22LV10C-7LJ
HDSP-E101
SW1
EEPROM SELECT
LARGE EEPROM
SMALL EEPROM
SW2
EEPROM ENABLE
SP1
SOLDER PAD
SP2
SOLDER PAD
SP3
SOLDER PAD
ATA CONNECTOR
Bridge SP1-SP3 for Atapi use.
PCB: PDC-9086 PCA: 121-08600
CYPRESS SEMICONDUCTOR © 2003
CY3684 USB 2.0 DEVELOPMENT BOARD
Document Number REF-11637
Rev *A
Tuesday, November 25, 2003
Sheet 1 of 1