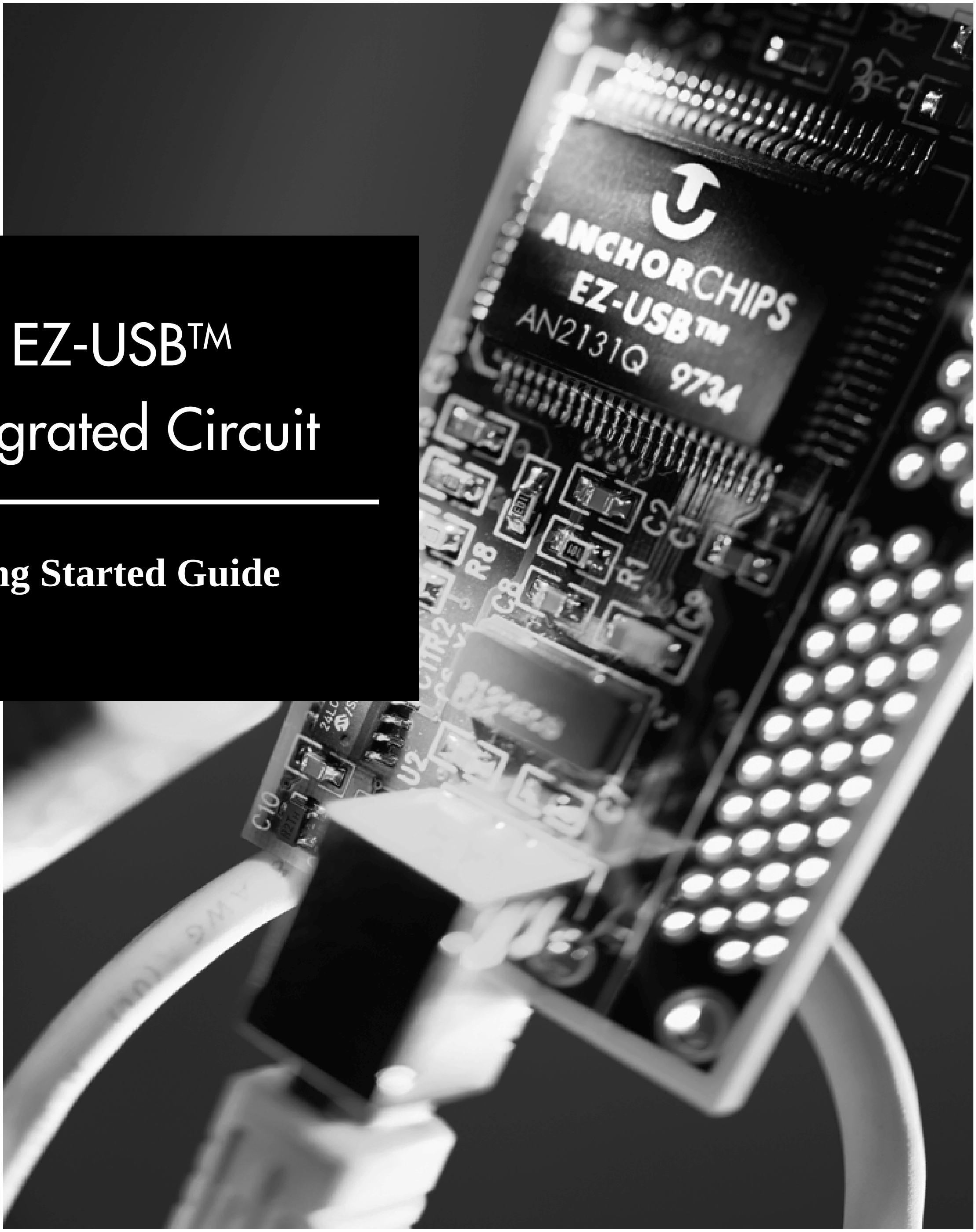


The EZ-USB™ Integrated Circuit

Getting Started Guide

ANCHORCHIPS

Version 1.5

Anchor Chips 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 Anchor Chips product could create a situation where personal injury or death may occur. Should Buyer purchase or use Anchor Chips products for any such unintended or unauthorized application, Buyer shall indemnify and hold Anchor Chips 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 Anchor Chips 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.

The EZ-USB™ Integrated Circuit Getting Started Guide

Document: AN2131-GS

Printed in USA

Getting Started Guide

Contents

EZ-USB Development Kit Overview 1-1

EZ-USB Development Kit Contents 2-1

EZ-USB Development Kit Software 3-1

Installing the Hardware 4-1

EZ-USB Development Board 5-1

CHAPTER

1

EZ-USB Development Kit Overview

Introduction

The Anchor Chips EZ-USB is a compact integrated circuit that provides a highly integrated solution for implementing a USB peripheral device. Three key EZ-USB features are:

- The EZ-USB family provides a “soft” (RAM-based) solution that allows unlimited configuration and upgrades.
- The EZ-USB family delivers full USB throughput. Designs that use EZ-USB are not limited by number of endpoints, buffer sizes or transfer speeds.
- The EZ-USB family does much of the USB housekeeping in the EZ-USB core, simplifying code and accelerating the USB learning curve.

To help reduce the development time and learning curve for making a USB peripheral using the Anchor Chips EZ-USB device, Anchor Chips provides the EZ-USB Developer’s Kit which this manual describes.

The Anchor Chips EZ-USB™ family (AN21XX/AN23XX) provides significant improvements over other USB architectures by including an enhanced 8051 core, 4, 8 or 16 Kbytes of RAM, and an intelligent USB core. The family includes 16 different products to accommodate the needs of different systems. The enhanced 8051 core provides five times the performance of the standard 8051, while maintaining 8051 software compatibility. With on-chip RAM, firmware code can be downloaded from the host PC. This allows the peripheral manufacturer to easily modify and transfer new code to current and new users. This on-chip memory eliminates the need for external memory in many systems.

The EZ-USB family supports high-bandwidth transfers by providing an efficient mechanism to move data between external memory and the USB FIFOs. Using this “turbo” mode, the 8051 core can transfer 1024 bytes of data in or out of an isochronous FIFO in 338 microseconds. This leaves a high percentage of the bandwidth for the processor to service the application. The EZ-USB family also supports an equivalent data transfer rate for bulk packets of over 2 Mbytes per second, which exceeds the USB bandwidth. The EZ-USB family conforms to the high-speed (12 Mbps) requirements of USB Specification version 1.0, including support for remote wake-up. The internal SRAM replaces Flash memory, EEPROM, EPROM, or masked ROM that is conventionally used in other USB solutions.

The EZ-USB family offers two packages, a 44 PQFP and an 80 PQFP. All EZ-USB devices are pin- and software-compatible. And, all RAM versions have ROM equivalents to allow easy migration for high-volume applications.

Tools Needed

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

- EZ-USB Development Board
- EZ-USB Peripheral Board
- EZ-USB Firmware Library and Firmware Frameworks
- EZ-USB General-Purpose Device Driver
- EZ-USB Driver and Firmware Sample Code
- Anchor Loader Driver sample code and driver
- EZ-USB Control Panel
- EZ-USB Documentation
- Reference Schematics
- 8051 Limited C Compiler from Keil Software
- 8051 Assembler from Keil Software
- 8051 Custom Debugger from Keil Software

Required Tools Not Included

- Full retail Keil Development System (if your application requires it)
- Microsoft Visual C++ (all PC sample code is developed under this platform)
- Microsoft WDM DDK (available through Microsoft Developer’s Network MSDN)

- Windows 98 PC

Other Suggested Tools

- Numega Softice for PC software debugging
- CATC USB Protocol Analyzer

EZ-USB Block Diagram

The Anchor Chips EZ-USB chip packs the intelligence required by a USB peripheral interface into a compact integrated circuit. As Figure 1-1 illustrates, an integrated USB transceiver connects to the USB bus pins D+ and D-. A Serial Interface Engine (SIE) decodes and encodes the serial data and performs error correction, bit stuffing, and other signaling-level details required by USB, and ultimately transfers data bytes to and from the USB interface.

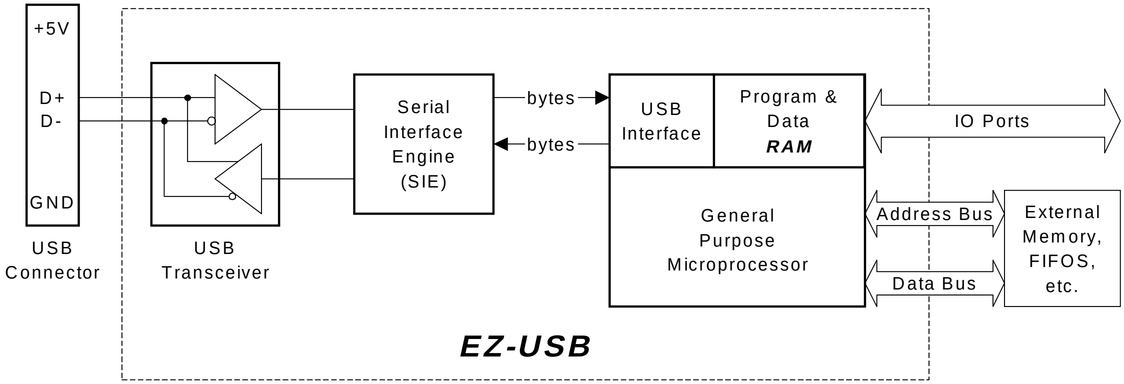


Figure 1-1: AN2313S (80 pin) Simplified Block Diagram

The internal microprocessor is an enhanced 8051 with fast execution time and enhanced features. It uses internal RAM for program and data storage, making the EZ-USB family a “soft” solution. The USB host downloads 8051 program code and device personality into RAM over the USB bus, and then the EZ-USB chip re-connects as the custom device defined by the loaded code.

The EZ-USB family uses an enhanced SIE/USB interface (called the Anchor “core”) which has the intelligence to function as a full

USB device even before the 8051 is running. This allows the USB host to download code into EZ-USB RAM, and then start the 8051. The enhanced core simplifies 8051 code by implementing much of the USB protocol.

EZ-USB chips operate at 3.3 volts. This simplifies the design of bus-powered USB devices, since the 5V power available in the USB connector (which the USB specification allows to be as low as 4.4V) can drive a 3.3 volt regulator to deliver clean isolated power to the EZ-USB chip.

The AN2131Q, an 80-pin version of the EZ-USB family has in addition to the 24 IO pins, a 16-bit address bus and 8-bit data bus for external memory expansion.

A special “fast transfer” mode moves data directly between external logic and internal USB FIFOs. The fast transfer mode, along with abundant endpoint resources, allows the EZ-USB family to support transfer bandwidths beyond the maximum required by the USB specification.

Final EZ-USB System Block Diagram

Your final USB application most likely will consist of a few major pieces: a custom Windows USB device driver, the standard Windows USB drivers, EZ-USB Application firmware, and an optional custom Windows USB Application program.

Figure 1-2 shows a typical EZ-USB Peripheral application block diagram.

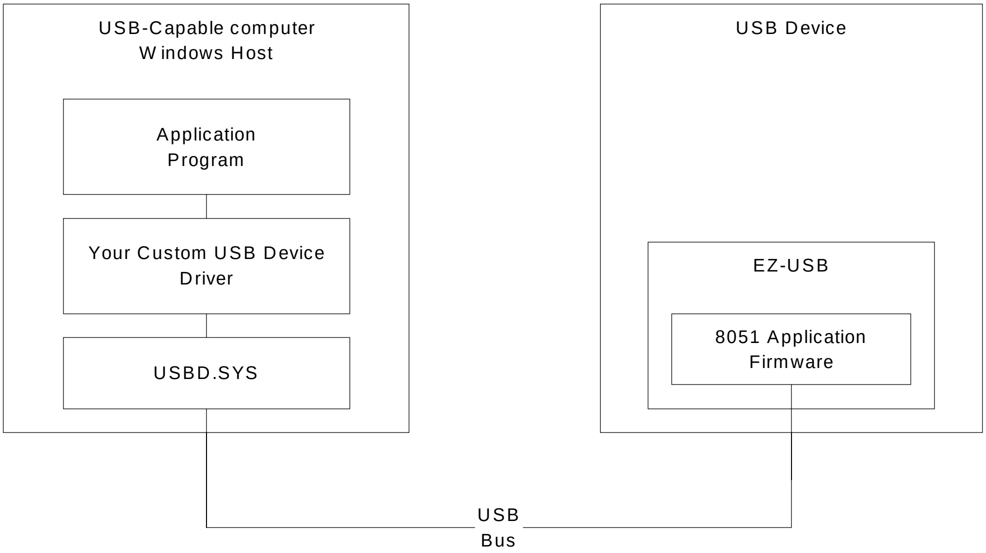


Figure 1-2: Final EZ-USB System Block Diagram

Initial Download Process

The EZ-USB family provides a “soft” (RAM-based) solution that allows unlimited configuration and upgrades. In order to accomplish this, your system must go through an initial download process before your final application is running.

The following list details the state of your system from the moment the EZ-USB based device is plugged into the USB bus.

- The Host detects that your device is plugged in and asks the device for it’s ID’s
- The device responds with ID’s that represent your non-downloaded device: (Vendor=XYZ Corp, Product=GIZMOLOADER).
- The Host PC sees these ID’s and knows to load your Loader Driver (based on the EZLOADER example in the Driver directory on the CD).

- Your Loader Driver performs an Anchor download of the embedded 8051 firmware into the EZ-USB chip.
- The Loader Driver then releases the Reset of the EZ-USB 8051 processor and your firmware application code starts running.
- Your 8051 firmware electrically disconnects the EZ-USB from the USB bus.
- The host, seeing the original device disappear, discards the Loader Driver from host memory.
- Your 8051 firmware Reconnects the EZ-USB to the USB bus.
- The Host detects that your device is plugged in and asks the device for it's ID's.
- The 8051 firmware responds with ID's that represent a your fully downloaded and configured device: (Vendor=XYZ Corp, Product=GIZMO).
- The Host PC sees these ID's and knows to load your Custom Device Driver (based on the Generic Driver example).
- The Host Driver and EZ-USB firmware are now loaded and running and perform the data transfers and control functions necessary for your application. You may also have a Windows application that calls your device driver that sends and receives data.

EZ-USB Development System Block Diagram

During system development, the EZ-USB system looks slightly different than your final system. The development environment provides debugging tools for the system under development. In order to accomplish this, a monitor program is first downloaded over a serial port and run in the EZ-USB 8051 processor. Then, using the Keil Debugger, you can load programs and debug your firmware application by single stepping and setting breakpoints. Once your firmware is loaded, the Anchor Control panel (which uses the Windows USB driver) allows you to generate USB traffic to test your USB control firmware.

Figure 1-3 shows the EZ-USB Developer's Kit System block diagram.

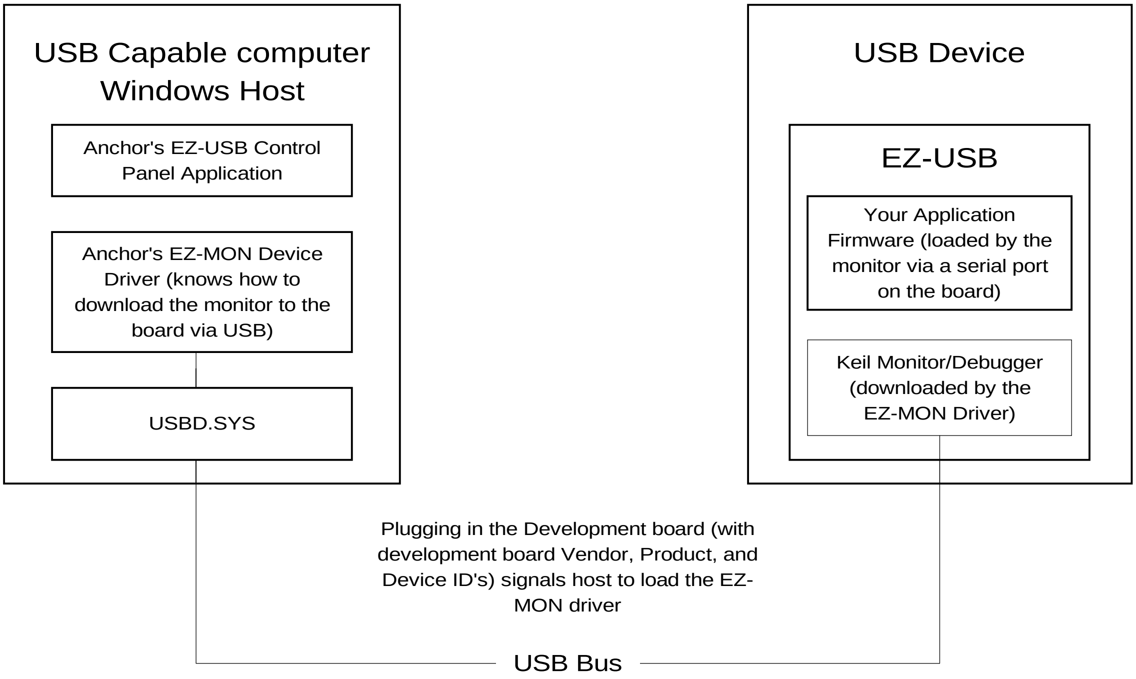

USB Capable computer
Windows Host
Anchor's EZ-USB Control Panel Application
Anchor's EZ-MON Device Driver (knows how to download the monitor to the board via USB)
USBD.SYS
USB Device
EZ-USB
Your Application Firmware (loaded by the monitor via a serial port on the board)
Keil Monitor/Debugger (downloaded by the EZ-MON Driver)
Plugging in the Development board (with development board Vendor, Product, and Device ID's) signals host to load the EZ-MON driver
USB Bus

Figure 1-3: EZ-USB Developer's Kit System Block Diagram

Setting up the Development Environment

The following list outlines the steps needed for setting up and using the development environment.

- The Host detects that a USB device has been plugged in and asks the device for it’s ID’s
- The device responds with ID’s that indicate an Anchor non-downloaded Developer’s Kit device: (Vendor=Anchor Chips, Product=EZ-USB Developer’s Kit).
- The Host PC sees these ID’s and knows to load the EZMON Driver (this driver knows how to download the Keil Monitor program to the EZ-USB Developer’s Board).
- The EZMON Driver performs an Anchor download of the Keil Monitor firmware to the embedded 8051 RAM outside of the EZ-USB chip.
- The EZMON Driver then releases the Reset of the EZ-USB 8051 processor and starts the Keil Monitor code running. The green light on the development board turns on when the Keil monitor firmware is running.
- Once the Keil Monitor is running, you run the Keil Debugger on the Host computer and download your application code using the serial port on the Developer’s Kit board to the 8051 RAM, either internal or external (but not on top of the Monitor).
- The Keil Debugger then starts running your Application firmware, allowing you to set breakpoints and single step.
- From here, your 8051 Application firmware is running.
- Your 8051 firmware Disconnects the EZ-USB from the USB bus.
- The host, seeing the EZ-USB Developer’s Kit device disappear, discards the EZMON Driver from host memory.
- Your 8051 firmware Reconnects the EZ-USB to the USB bus.
- The Host detects that your device is plugged in and asks the device for it’s ID’s.

- The 8051 firmware responds with ID’s that represent a your fully downloaded and configured device: (Vendor=XYZ Corp, Product=GIZMO).
- The Host PC sees these ID’s and knows to load your Custom Device Driver (based on the Generic Driver example).
- The Host Driver and EZ-USB firmware are now loaded and running and perform the data transfers and control functions necessary for your application. You may also have a Windows application that calls your device driver that sends and receives data.

CHAPTER

2

EZ-USB Development Kit Contents

The EZ-USB Xcelerator™ development kit provides a complete hardware and software solution for accelerating the firmware and device driver development for all the members of the EZ-USB family. Other USB development kits use emulation of the eventual USB device. The EZ-USB Xcelerator developer kit uses the actual device, the AN2131QC, during the entire development.

EZ-USB Xcelerator Development Kit Contents

Hardware

- EZ-USB Developer Board (AN2131DK)
- Shielded Series A-B USB Cable
- Breadboard
- RS-232 Cable

Software on CD-ROM

Custom Software Utilities & Documentation

- EZ-USB Control Panel Application Software
- USB Generic Driver
- Source file for USB Generic Driver
- EZ-USB Loader Driver
- Example Source file for EZ-USB Loader Driver
- EZ-USB Firmware Library
- EZ-USB Applications Framework
- Source file for Firmware Frameworks and Firmware Library
- 8051 Example Code
- EZ-USB Control Panel User’s Guide
- USB Generic Driver Documentation
- EZ-USB Loader Driver Documentation
- 8051 Firmware Library Documentation
- EZ-USB Technical Reference Manuals

Keil Development Tools (2K Limited Edition) (8051 Build Environment)

- Compiler
- Assembler
- Linker
- Debugger
- Monitor
- Documentation and User Guide

Keil Development Tools (4K Limited Edition) (8051 Build Environment)

- Compiler
- Assembler
- Linker
- Debugger
- Monitor
- Documentation and User Guide

EZ-USB Tutorials

Printed Documentation

- Getting Started Manual
- EZ-USB Technical Reference Manual

CHAPTER

3

EZ-USB Development Kit Software

Beginning the Install Procedure

Start by collecting the following materials:

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

Verify that USB support is present on the development PC before proceeding. There should be at least one USB connector socket available on the PC chassis (a flat connector socket). The motherboard BIOS must also contain USB support.

Generally speaking, you must have one of the following operating systems which provide USB support.

- Win 95 OSR2.1 (version 4.03.1214 or later)
- Windows 98 (highly recommended)

The OS version can be found in the registry at:

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows \CurrentVersion -> VersionNumber

If you are using Win95 and are not up to version 1214, then you need to get the most current version of usbsupp.exe and install it. usbsupp.exe is available through Microsoft Developer’s Network (MSDN).

To quickly determine that USB support is present on your PC, do the following:

- Select “**Start\Settings\Control Panel**”.
- Double-click on the “**System**” icon.
- Select the “**Device Manager**” tab.
- You should see a “**Universal Serial Bus Controller**” icon.

This icon will probably have a Host Controller and a USB Root Hub listed beneath it.

Installing the Control Panel, Drivers and Documentation

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.

You will see a menu of possible install options:

- Adobe Acrobat Reader 3.0
- Anchor Chips EZ-USB Control Panel
- Keil 8051 2K Development Tools
- Keil 8051 4K Development Tools

IMPORTANT NOTE:

You must run the Setup.exe program once for every item you would like to install. The typical installation includes the Anchor Chips EZ-USB Control Panel and the Keil 8051 2K Development Tools.

Adobe Acrobat Reader allows you to view the documentation which is in PDF file format.

Select the appropriate menu item and follow the install directions on the screen.

The Anchor Chips EZ-USB Control Panel is a Windows program that allows sending and receiving of data over the USB to an Anchor Chips EZ-USB IC. This selection also installs all of the sample code, examples, drivers and documentation.

After you have clicked “Finish” to complete the Anchor EZ-USB Control Panel installation, you may run the application by selecting: “**Start\Programs\Anchor\EzMr.**”

The Control Panel application may be uninstalled completely by using the “**Add/Remove Programs**” component available in the Windows Control Panel.

The Keil Development Environment allows you to compile C and assembly code for the 8051, link the object files and debug your program using the Monitor program included with the kit. The Keil package includes the uVision C Source code development environment, and the dScope debugger. The Keil C compiler and linker are limited versions intended for evaluation and beginning development. Because of this, two versions of the Keil system are included on the CD. Both versions operate in the same way, and differ only by code size and code location limitations.

You may need to restart your PC after installing the Keil software.

For information on how to operate the Keil development environment, refer to the help files in the Keil “BIN” directory. Double-click on “contents.hlp” for a beginning page of topic links.

“**Keil 8051 2K Development tools**” loads the Keil firmware development system onto your PC. This version can compile and locate up to 2 kilobytes of code, which can be located anywhere in the 8051 64K memory. This means that 2 kilobytes of code can be located either in the internal 8 kilobyte EZ-USB RAM, or in the external 64 kilobyte RAM on the development board.

“**Keil 8051 4K Development tools**” loads another version of the Keil firmware development system onto your PC. This version can compile and locate up to 4K of code, but the code must be located above 0x4000 in the program memory space of the 8051, which means it must reside external to the EZ-USB on-chip RAM.

CHAPTER

4

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 in the steps before, will be automatically located and loaded. The green light (BKPT/Monitor) should illuminate, indicating that the 8051 Keil monitor code is loaded and running.

(The driver, ezusb.sys, was automatically installed into the Windows\System directory during setup of the Control Panel. An .inf file was also created in the Windows\INF directory.)

Confirming Installation Using the Control Panel

Run the Control Panel application and perform a “Get Device Descriptor” operation by clicking the “GetDev” button.

The control panel should display the response from the Development board shown below. The “idVendor” value of 0x547 is the Anchor Chips vendor ID, and the “idProduct” value of 0x80 identifies the EZ-USB Development Board. 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:  0x547
idProduct:  0x80
bcdDevice:  0x1
iManufacturer:  0x0
iProduct:  0x0
iSerialNumber:  0x0
bNumConfigurations:  0x1
```

More detailed instructions on this procedure are available in a tutorial which can be viewed in help files. Start the EZ-USB Control Panel and then select”**Help\Help Topics**” and choose the topic, “**EZ-USB Developer’s Kit Manual**”, sub-topic “**EZ-USB Developer’s Kit Tutorials**.”

CHAPTER

5

EZ-USB Development Board

The Anchor Chips EZ-USB Development Board provides a compact evaluation and design vehicle for the EZ-USB Integrated Circuit family. The board uses the 80-pin AN2131Q chip, and provides expansion and interface signals on five 20-pin headers. A mating prototype board allows quick construction and testing of USB designs.   All IC's 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 Development Board is supplied as part of the Anchor Chips Xcelerator™ Development Kit, which includes an evaluation version of Anchor-customized software development tools from Keil Software Inc. The Keil 8051 assembler, C 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 EZ-USB RAM free for code development. Even though you can use only 2K of internal 8K RAM with the evaluation version of the Keil tools, the full retail version allows code of any size to be located anywhere.

Board Overview

The EZ-USB Development Board contains the following components:

1. AN2313Q EZ-USB Integrated Circuit, which includes a high performance 8051 core.

2. 64 Kilobytes of 8051 Expansion RAM (two 32 Kilobyte RAM chips).

3. RS-232 level converters for the two 8051 UARTS. UART signals are brought out on two DB-9 female connectors.

4. DIP Switches to select RAM memory maps and EEPROM I^{2}C address.

5. Two I^{2}C bus 8-bit IO expander chips. One drives a seven-segment LED, the other reads eight switches.

6. A 3.3 volt regulator that accepts 5 volt power from the USB or an external supply.

7. Indicator lights for power and break/monitor operation.

8. Expansion connectors that mate to a supplied prototype board.

8051Resources Used

If you plan to expand the EZ-USB Development Board by using the prototype board, it is important to know which 8051 signals are used by the main board to avoid signal conflicts. These signals are RD#, WR#, RXD1 and TXD1. The RD# and WR# signals are hard-wired to the memory decode PAL. For proper operation of the 8051 monitor, the RD# and WR# signals must be active. Never change the PORTCCFG bits 7-6 or 1-0 from “1” to insure proper operation of the debug monitor.

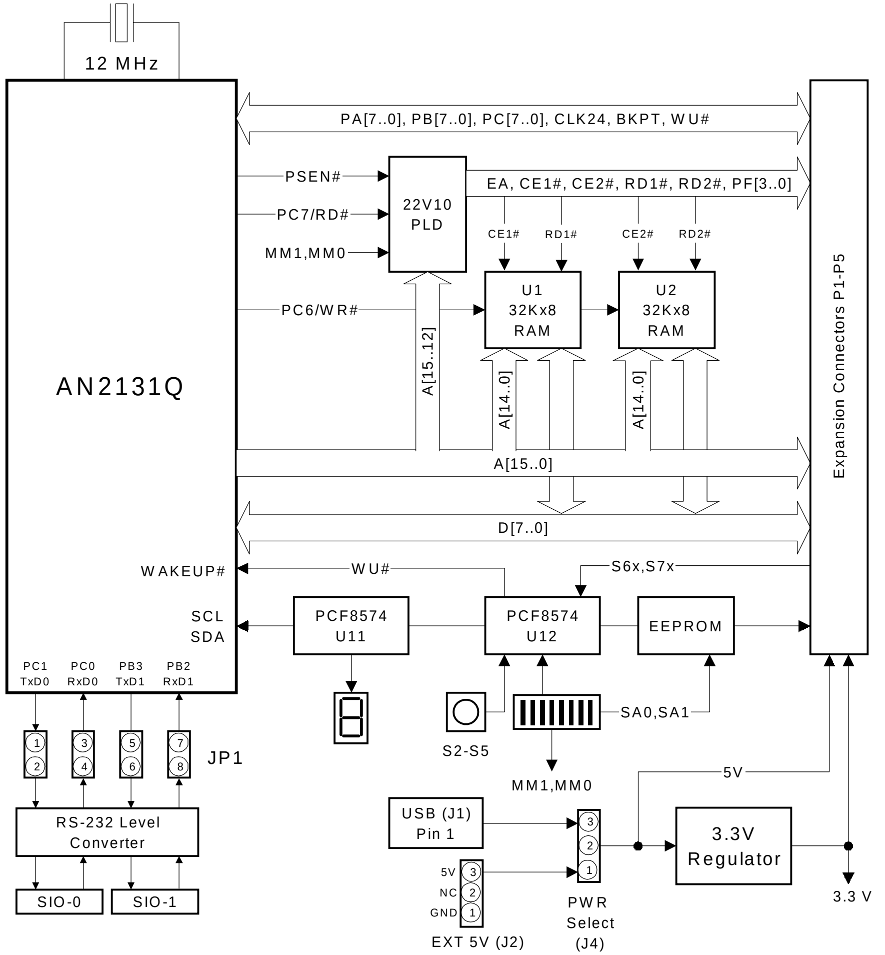


Figure 5-1: EZ-USB Development Board Components

Circuit Details

The EZ-USB I^{2}C bus is connected to two 8-bit IO expander chips (Philips PCF8574) and an EEPROM used for USB device identification, EZ-USB boot loads and 8051 access. The EEPROM is socketed (8-pin DIP) to allow the use of various EEPROM densities.

Push-buttons, selector switches, and a seven-segment numeric readout are connected to the AN2131Q I^{2}C bus via the PCF8574 chips to give the firmware developer debug support while leaving the EZ-USB IO Ports A, B and C available for the application under development. U11 provides eight outputs to the seven-segment numeric display. U12 provides eight inputs from four push-buttons (S2-S5), two rocker switches, and two general purpose signals (S6X, S7X) from the plug-in prototype board. A state change on any U12 input triggers the AN2131Q WAKEUP# pin. Thus any of the push buttons can be used to initiate a remote USB wakeup.

Two 32 Kilobyte RAMs supplement the 8 Kilobytes of program and data RAM in the AN2131Q. Debug monitor code may be loaded into this external RAM to keep the AN2131Q internal RAM free for user code. A 22V10 PAL provides the external RAM enable signals. Two configuration switches allow four different memory maps for the external RAM.

The EZ-USB 8051 contains two UARTS. Both are brought out to jumper block JP1, which connects the UART signals to an RS-232 level shifter and two DB-9 female connectors. The shorting plugs on JP1 may be removed to disconnect the RS-232 level shifters if the IO pin functions are required by the system.

An eight position DIP Switch selects various board options. The signals MM1 and MM0 connect to PLD U5, a 22V10, to select one of four memory maps. (Refer to “Memory Maps” on page 5-11.) The signals SA1 and SA0 connect to EEPROM address lines to select various I^{2}C bus secondary addresses. Two switch signals connect to an 8574 (U12) for general purpose use. The 8051 in the AN2131Q can read the state of these switches using the I^{2}C bus.

Jumpers

Table 5-1: UART Buffer Connections

JP1 Pins	Function	Default
1-2	TXD0	Installed
3-4	RXD0	Installed
5-6	TXD1	Installed
7-8	RXD1	Installed

Table 5-2: Board Power Source

J4 Pins	Function	Default
1-2	External power (supplied on J2)	
2-3	USB Connector	Installed

Table 5-3: USB Disconnect on Push-button RESET

J5 Pins	Function	Default
1-2	No	Installed
2-3	Yes (1)	Installed

(1). If a shorting plug is installed on J5 pins 2-3, pressing the EZ-USB Development Board RESET push-button resets the AN2131Q **and** simulates a USB bus disconnect. Releasing the RESET button re-connects the AN2131Q to the USB. If the disconnect feature is not required place the shorting plug on J5 pins 1-2.

The advantage of keeping J5 jumpered to pins 2-3 is that if your firmware crashes, you can press the RESET button to re-connect as a “clean” USB device and automatically re-load the Keil debug monitor (turning on the green light).

DIP Switch Settings

Table 5-4: 8-position DIP switch

S6 Position	Function	Description	Default
1	PCF8574 bit 4	8051 reads switch state over I²C bus	OFF(H)
2	PCF8574 bit 5	8051 reads switch state over I²C bus	OFF(H)
3	MM0	Memory Mode Select (Note 1)	OFF(H)
4	MM1	Memory Mode Select (Note 1)	OFF(H)
5	NC		OFF(H)
6	NC		OFF(H)
7	SA0	EEPROM Slave Address A0 (Note 2)	OFF(H)
8	SA1	EEPROM Slave Address A1 (Note 2)	OFF(H)

General Note: One side of each S6 switch is connected to ground, and the other side is connected to 3.3V through a pullup resistor. Therefore an “ON” switch produces 0V or logic LO, and an “OFF” switch produces 3.3V or logic HI.

Note 1: Refer to “Memory Maps” on page 5-11for the four memory map choices selected by these switches.

Note 2: To use the U9 EEPROM as an AN2131 boot loader, set switches 7 and 8 as shown in Table 5-5. The 24LC00 supplied on the board has no slave address pins, so the switch settings are “don’t care”. The boot load mechanism is described in the EZ-USB Technical Reference Manual.

Table 5-5: SW7, SW8 Settings for Various EEPROM Types

EEPROM	SW7	SW8
24LC00	Either	Either
24LC01	ON	ON
24LC02	ON	ON
24LC32	OFF	ON
24LC64	OFF	ON

Connectors

Table 5-6: USB Connector

J1 Pin	Signal
1	Vcc
2	D-
3	D+
4	GND

Table 5-7: External 5V Power Connector

J2 Pin	Signal
1	GND
2	NC
3	5V

Table 5-8: I^2C Bus Connector

J3 Pin	Signal
1	GND
2	SDA (Note 1)
3	SCL (Note 1)
4	3.3V

Note 1: The I^2C bus pins SCL and SDA are pulled up to 3.3 volts on the development board using 2.2K resistors. External I^2C devices connected to the bus using J3 do not require pullup resistors. The on-board 3.3 volt regulator can supply up to 50 ma to external devices.

Table 5-9: SIO-0 Connector

P6 Pin	Signal
1	NC
2	TXD0 (RS-232)
3	RXD0 (RS-232)
4	NC
5	GND
6	NC
7	NC
8	NC
9	NC

Table 5-10: SIO-1 Connector

P7 Pin	Signal
1	NC
2	TXD1 (RS-232)
3	RXD1 (RS-232)
4	NC
5	GND
6	NC
7	NC
8	NC
9	NC

NOTE:

The UART transmit and receive signals are connected to connectors P6 and P7 only if the JP1 jumpers are installed.

Memory Maps

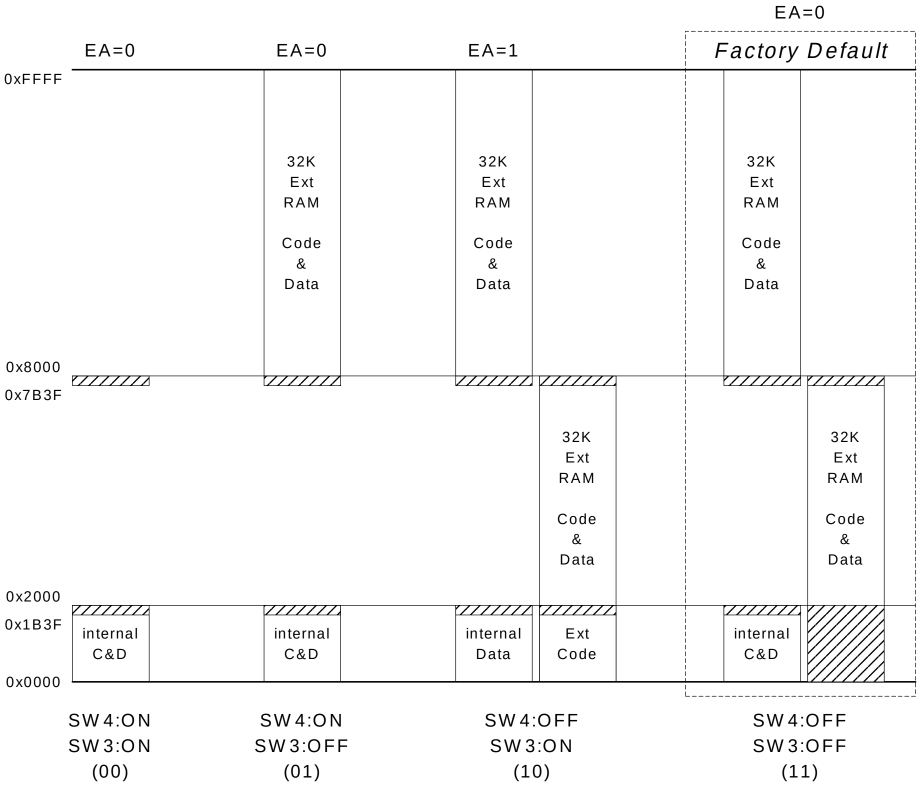


Figure 5-2: Memory Maps Based on Switches SW4-SW3

Configuration 00

None of the development board external RAM is used in configuration 00. Only the internal 8 Kilobyte EZ-USB RAM is available for 8051 code and data (C&D). This mode is useful for checking out final code, since the EZ-USB chip operates stand-alone.

Configuration 01

This configuration adds 32 Kilobytes of external RAM at 8000-FFFF. This mode is useful for developing systems that require external memory or memory-mapped peripherals. The debug monitor is loaded into the top 32 Kilobytes of RAM, the user code is loaded into the EZ-USB internal RAM, and the bottom 32 Kilobytes (minus the bottom 8K) is free for memory expansion (See Note 1).

Configuration 11

The default memory configuration populates the full 8051 external memory space with 64 Kilobytes of RAM. The 22V10 sets EA=0 so that internal EZ-USB 8K RAM occupies the address space 0000-1FFF (See Note 1).

Configuration 10

This configuration is almost identical to configuration 11. The only difference is that the 22V10 sets EA=1, forcing code fetches from external RAM rather than from internal EZ-USB RAM. This simulates ROM versions of the EZ-USB family, where the internal 8K RAM is data-only.

Note 1. The EZ-USB RD# and WR# signals are inactive when the EZ-USB 8051 accesses internal RAM. Therefore a 32 Kilobyte memory can be used at 0000-7FFF without requiring external address decoding to de-select accesses from 0000-1FFF.

Note 2. EZ-USB bulk endpoint buffers reside at 7B40 and are aliased at 1B40 to allow unused buffers to add to available RAM. For details see the EZ-USB Technical Reference Manual Chapter 3, “EZ-USB Memory”.

PLD Signals

U5 is a 3.3 volt 22V10 PLD that develops the RAM chip enable and read signals. It also has spare inputs and outputs, as shown in the following table. Signals shown as “Not used” can be used for any purpose by re-programming the 22V10 based on the ABEL file shown in “PLD Source File” on page 5-15.

Table 5-11: 22V10 Signals

U5 Pin	Signal	Function
1	NC	---
2	PA2/OE#	Not used
3	PA3/CS#	Not used
4	BKPT	Not used
5	PSEN	AN2131 PSEN# pin
6	PC7/RD#	AN2131 RD# pin
7	MM0	Memory Mode Switch Select 0
8	NC	---
9	MM1	Memory Mode Switch Select 1
10	SPARE	Not Used
11	A12	AN2131 A12
12	A13	AN2131 A13
13	A14	AN2131 A14
14	GND	---
15	NC	---
16	A15	AN2131 A15
17	PF0	Not Used (expansion P2-11)
18	PF1	Not used (expansion P2-9)
19	PF2	Not used (expansion P2-7)
20	PF3	Not used (expansion P2-5)
21	EA	AN2131Q EA pin

Table 5-11: 22V10 Signals

U5 Pin	Signal	Function
22	NC	---
23	PRD#	OR’d PSEN or RD (connected to P2-13)
24	RD2#	U2 RAM Read Strobe
25	RD1#	U1 RAM Read Strobe
26	CE2#	U2 RAM Chip Enable
27	CE1#	U1 RAM Chip Enable
28	Vcc=3.3V	---

PLD Source File

The following ABEL code was used to generate the programming file for U5.

```
MODULE EZmap10

“ 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;
      nRD,nPSEN,BKPT,PA3,PA2         pin 6,5,4,3,2;
      mm1,mm0                        pin 9,7;
“Outputs
      EA,nPRD,nRD2,nRD1,nCE2,nCE1pin 21,23,24,25,26,27;

modesw = [mm1,mm0];                  “ two dipswitches

equations

!nPRD =    !nRD # !nPSEN;

WHEN       (modesw == 00) THEN       “ No external memory

{
nCE1  =    1;
nCE2  =    1;
nRD1  =    1;
nRD2  =    1;
EA    =    0;
}
ELSE WHEN(modesw == 01) THEN“ Ext P&D mem at 8000 (can add mem
to 0-8K)
{
```

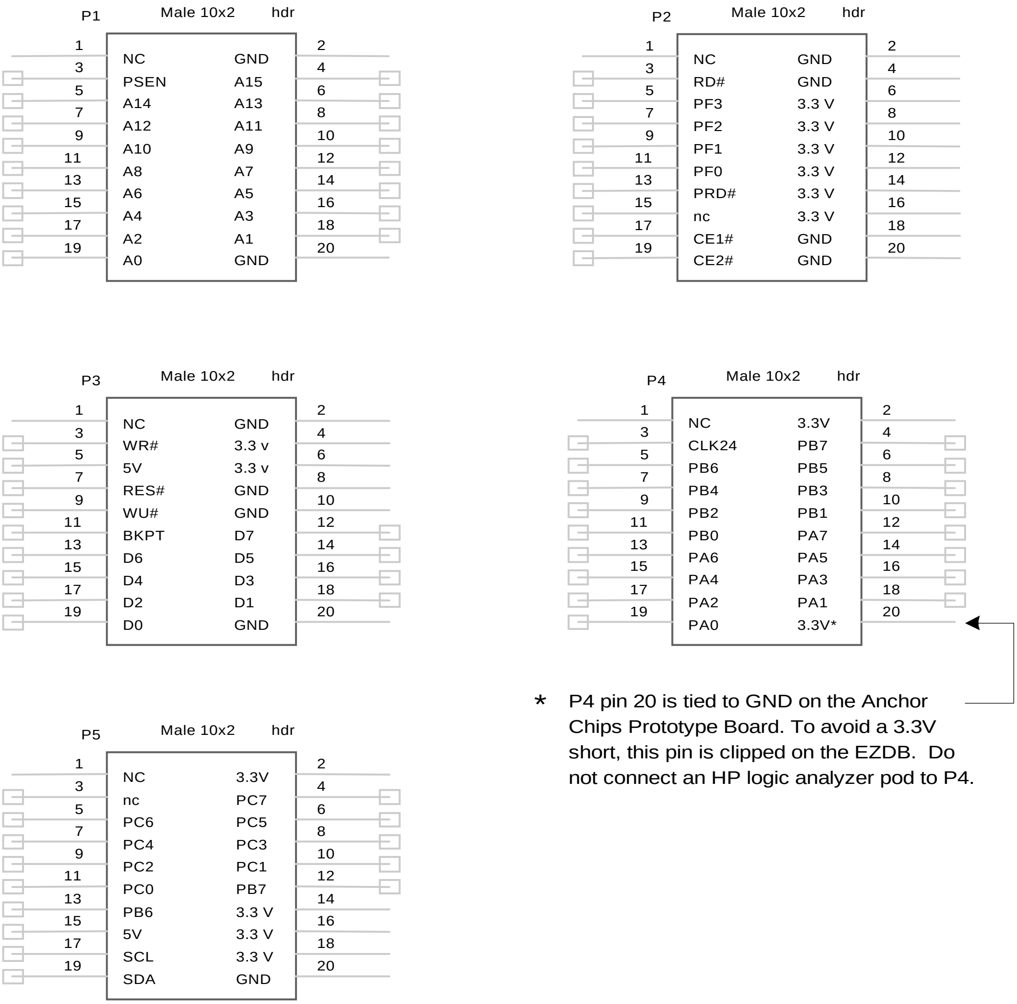
```
!nCE1 =    A15;
nCE2  =    1;
!nRD1 =    !nRD # !nPSEN;    “ Combine program & data memory
nRD2  =    1;     “ inactive RD doesn’t strobe for lowest power
EA    =    0;
}

ELSE WHEN(modesw == 11) THEN“ Ext P&D mem at 0000 and 8000
{
!nCE1 =    !A15;
!nCE2 =    A15;
!nRD1 =    !nRD # !nPSEN;
!nRD2 =    !nRD # !nPSEN;
EA    =    0;
}
ELSE WHEN(modesw == 10) THEN“ All program mem external
{
!nCE1 =    !A15;
!nCE2 =    A15;
!nRD1 =    !nRD # !nPSEN;
!nRD2 =    !nRD # !nPSEN;
EA    =    1;
}

test_vectors

([mm1,mm0,A15,nRD,nPSEN] -> [nCE1,nCE2,nRD1,nRD2,EA])
 [ 0 , 0 , x , x ,  x  ] -> [ 1  ,  1 ,  1 ,  1 , 0];“ 10: all
mem selects and strobes OFF

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

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

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

test_vectors

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

END
```

Prototype Board Expansion Connectors P1-P5

Five 20-pin headers connect interface signals from the EZ-USB Development board to the prototype board. Headers P1-P3 and P5 are pin compatible with HP model 1660 Logic Analyzers to simplify pod connection. P4 has 3.3V on pin 20, and is not pod compatible.

The following table describes these signals. In the table, the EZ-USB Development Board is abbreviated as “EZDB”. Active-low signals end with “#”. Direction is relative to the EZDB.



* P4 pin 20 is tied to GND on the Anchor Chips Prototype Board. To avoid a 3.3V short, this pin is clipped on the EZDB. Do not connect an HP logic analyzer pod to P4.

Figure 5-3: Five 20-Pin Headers

Table 5-12: Expansion Connectors Signal Descriptions

Signal	Type	Description
Power Signals		
GND	PWR	Signal Ground. Connected to proto board ground plane.
3.3V	PWR	3.3 volts from the EZDB Regulator U6
5V	PWR	5 volts supplied from either of two sources, depending on the position of the shorting plug on EZDB J4 (Power Select). If pins 2-3 are jumpered (“BUS”), the power is supplied from pin 1 of the USB connector. If pins 1-2 are jumpered (“EXT”) the power is supplied externally from J2 (“EXT 5V). The red “PWR” LED is connected to this signal. NOTE: USB supplies 100 ma max (hub) or 500 ma max (direct), and can be as low as 4.4V due to cable and connector drops.
NC		Unconnected on the EZDB.
8051 Interface Signals		
A[15..0]	OUT	8051 Address Bus. Always drives.
D[7..0]	IO	8051 Bi-directional Data Bus. Also carries the data for fast transfers (using FRD#/FWR strobe signals). Drives low in suspend.
PSEN#	OUT	8051 Program Store Enable strobe
RD#	OUT	8051 external memory Read strobe
WR#	OUT	8051 external memory Write strobe
Programmable Control Signals (from EZDB U5, a 22V10) See page 5-15 for equations.		
CE1#	OUT	Chip enable signal for EZDB U1, a 32Kbyte RAM
CE2#	OUT	Chip enable signal for EZDB U2, a 32 Kbyte RAM
RD1#	OUT	Read Strobe for EZDB U1
RD2#	OUT	Read Strobe for EZDB U2
PRD#	OUT	OR’d PSEN and RD
PF[3..0]	IO	General purpose 22V10 outputs (or inputs)
Other EZ-USB Signals		
BKPT	OUT	EZ-USB Breakpoint output signal. Also buffered and connected to the green “BKPT/Monitor” LED
WU#	OUT	EZ-USB Wakeup signal (from push-buttons)

Table 5-12: Expansion Connectors Signal Descriptions

Signal	Type	Description
RES#	OUT	Buffered output from RESET button
PA[7..0]	IO	EZ-USB IO port A
PB[7..0]	IO	EZ-USB IO port B
PC[7..0]	IO	EZ-USB IO port C
CLK24	OUT	24 MHz, 50% duty cycle clock (8051 clock)
SCL	OD	Open-drain I^{2}C clock, pulled up to 3.3V with 2.2K resistor on EZDB
SDA	OD	Open-drain I2C data, pulled up to 3.3V with 2.2K resistor on EZDB

Philips PCF8574 IO Expander

The EZ-USB Development Board uses two Philips PCF8574 IO Expander chips (U11 and U12) to provide debugging IO without interfering with the EZ-USB IO ports A, B and C. The PCF8574 is described in the Philips I^{2}C Peripherals Handbook (Philips Publication “IC12”). This section summarizes the basic device characteristics for programmers who wish to write 8051 code that utilizes the onboard devices connected to U11 and U12.

The PCF8574 is a slave-only device.

- The Slave Address is 0100.
- The eight IO pins are “quasi-bidirectional”. Setting a pin HI makes it an input with a weak pull-up that can be driven low by an external signal. Setting a pin LO drives LOW and thus makes it an output. IO pins are HI (inputs) at power-on.
- The Development Board switches and push-buttons connect to U2, which is strapped to address 0010000d (“d” is the I^{2}C direction bit). To read the eight bits, send an I^{2}C command byte of 01000001 to address the 8574 and set the ‘read’ direction, then read the I^{2}C data byte.
- The 7-segment display is at address 01000000 (the last bit is 0 since the display pins are outputs). To set the segments, send an I^{2}C command byte of 01000000 to address the 8574 and set the ‘write’ direction, then write the I^{2}C data byte. A “0” turns on a segment. The segments are mapped as follows:

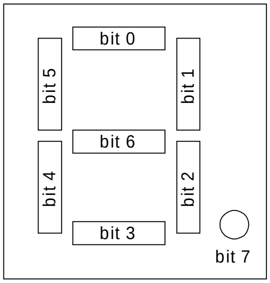


Figure 5-4: Seven-Segment Bit Assignments

- 8574 pin 13, /INT, is an open-drain output that goes low whenever any of the inputs changes state. U12, the 8574 used for input, has pin 13 connected to the EZ-USB WAKEUP# input. This allows any of the push-buttons to initiate a USB remote wakeup. The /INT pin returns to the HI state when data on the port returns to its original setting or when data is read from the 8574.

Connector, Switch Jumper and Indicator Locations

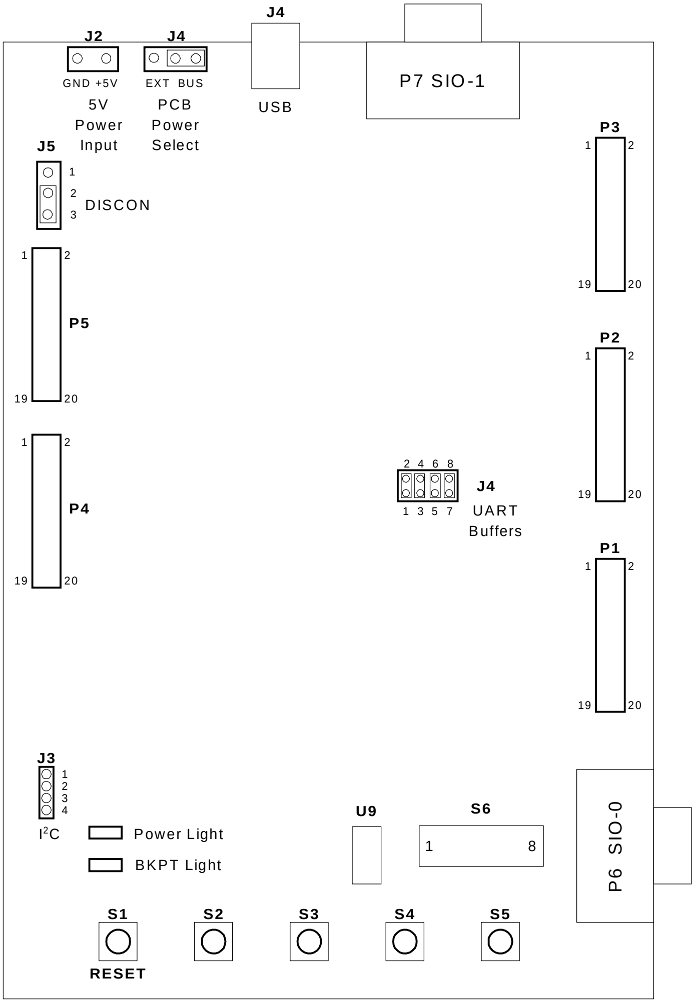


Figure 5-5: Locations of Switches and Jumpers

The development board has two indicators.

The red "Power" LED is connected to the input of the 3.3 volt regulator. It is ON whenever USB power or external power is applied to the board.

The green LED is connected through a buffer to the AN2131Q BKPT (breakpoint) pin. This signal goes high whenever the 8051 address matches the value in a pair of EZ-USB registers. It is used by the Anchor development tools to indicate that the debug monitor is running.