EZ-811HS

Development Kit Manual

Getting Started

Rev 1.4

• Cypress Semiconductor • Interface Products Division •
• 15050 Avenue of Science • Suite 200 • San Diego, CA 92128 •

Table of Contents

Getting Started

1.0 EZ-811HS Development Kit

1.1 Introduction

SL811HS is an embedded USB Host/Slave Controller capable of communicating with either full-speed or low-speed peripherals. The SL811HS can interface to devices, such as microprocessors, microcontrollers, DSPs, etc. The EZ-USB development kit proves to be the ideal starting platform for SL811HS development, utilizing the internal 8051 core as the MCU interface to the SL811HS, and using the EZ-USB interface for debugging purpose. Figure 1 shows the basic interconnect block diagram of the entire development platform.

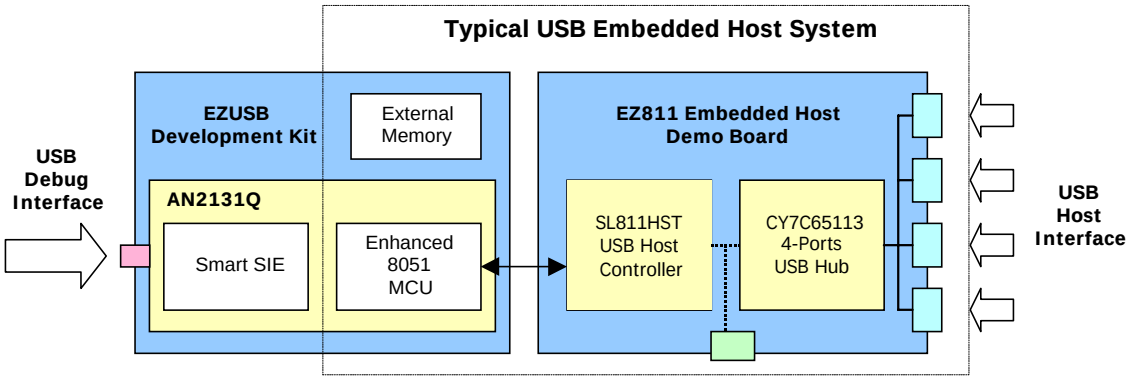

Figure 1. Interfacing SL811HST with EZ-USB Development Kit

2.0 Development Kit Content

The following is a list of the components supplied in the EZ-811HS Development Kit.

- SL811HS daughter card
- EZ-USB development board

- USB cable
- Serial cable
- CD

2.1 CD Contents

The following directories are installed for the EZ-811HS Development Kit:

```
C:\CYPRESS\USB\ez811\bin                - Binary files and firmware files
C:\CYPRESS\USB\ez811\doc                - EZ-811HS related documentation
                                          Getting Started and Data Sheets
C:\CYPRESS\USB\ez811\EZ811_DK           - EZ-811 PC Host application
C:\CYPRESS\USB\ez811\firmware\Emb_Host  - Embedded Host firmware
C:\CYPRESS\USB\ez811\firmware\slave     - Slave mode firmware
C:\CYPRESS\USB\ez811\firmware\USBHost   - Simple embedded USB Host application
                                          documentation and firmware
C:\CYPRESS\USB\ez811\HARDWARE           - Hardware related information
C:\CYPRESS\USB\ez811\Drivers\Linux      - Linux driver for 811HS, see included
                                          User Guide
```

2.2 About the Hardware

The development kit includes a hardware daughter card that houses the SL811HST and the CY7C65113 4-port hub. This daughter card was designed so that it can be carefully inserted directly to the header sockets available on the EZ-USB development kit, without any unnecessary wire connections. The 4-port USB hub extends the number of downstream ports available to a maximum of four. You will have a choice of either 1 USB port (directly from SL811HST) or 4 down-stream ports (through CY7C65113). Figure 2 is a photograph of the demo kit. There are several jumpers so that you can select different configuration modes.

JP1 – Master/Slave Mode Select Jumper

JP2 – Slave Speed Select Jumper

JP3 – Power Select Jumper

The Master/Slave mode select jumper allows you a choice of the following:

- choose between the SL811HST USB host only, with one downstream port (J1)
- choose to automatically allow the SL811HST to enumerate with the on-board hub to extend the downstream ports to four (H1~H4)
- choose slave operation when SL811HST is configured as a slave USB controller. This slave mode will be configured together with the Slave Speed select jumper.

The Power Select source (JP3 on the EZ-811) can also be selected from an external DC power adapter, EZ-USB development kit, or from USB bus during slave mode operation. Data and control signals are also available on the header to ease connection of the SL811HS to other types of CPU interfaces.

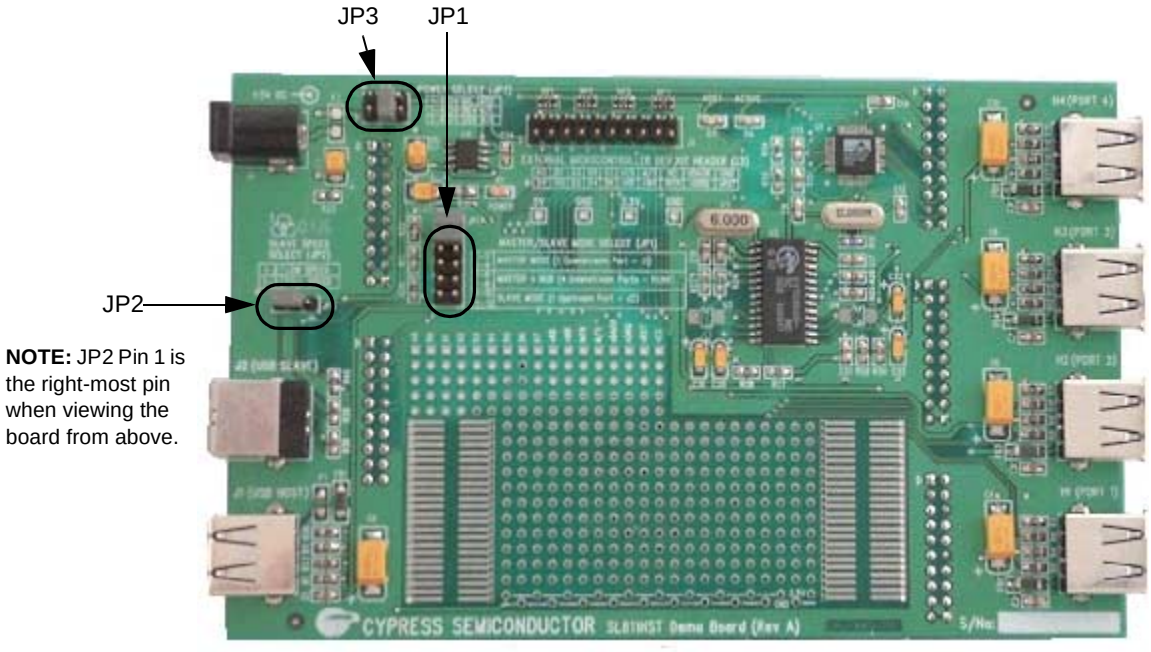

Figure 2. EZ-811HS Development Board

Configuring the EZ-USB development kit is easy, only one DIP switch setting needs to be made. DIP switch SW4 of S6 has to be in the ON position to disable the lower 32K external memory; this allows the SL811HST's address and data registers to be mapped to this area. The upper 32K of external memory had been allocated to firmware code and variables. Note that when the SL811HST demo board is inserted onto the headers (P1~P5) of the EZ-USB development kit, and the EZ-USB enumerates, it should comes out as "Cypress EZ-USB Development Board", indicating that the monitor code for debugging was loaded successfully (green LED (D5 on the Dev Board) should turn ON), and the Keil Debugger can be used to download and debug code. Code and variable spaces have to be carefully place to avoid disruption to the monitor code space.

Firmware code can be downloaded in two different ways, using USB or serial connection. The Control Panel Software for the EZ-USB development kit can be use to download code through the USB into the 8051 core, or Keil uVision 2 can be used to download code through the serial port. Whichever method is used for downloading, upon a successful download the HOST LED (D3 on the EZ-811 Board) should turn ON, indicating that SL811HST is in host mode, and the ACTIVE LED should be blinking indicating that it is waiting for any slave devices that are attached to the SL811HST to enumerate. It is recommended that LEDs also be connected to indicate successful enumeration of the hub's downstream devices. Port A bits 7-4 have been configured in firmware to indicate which downstream Port 1 to 4 (respectively) has successfully completed the enumeration process.

3.0 Host Mode Operation

3.1 About the Host Firmware

The Firmware includes detection of any USB slave devices, enumeration of standard devices, and support for HID and HUB devices. The firmware is limited to support a maximum of five simultaneously connected devices. This consists of a 4-port hub connected directly to the SL811HS followed by four non-hub devices downstream. The firmware does not support a multiple hub tier structure, multiple interface devices, or ISO-transfers. After successful enumeration of a hub device, any downstream devices will also be detected and will go through the process of enumeration. Dynamic USB address allocation of slave device is used, meaning that the next available USB address will be assigned to the current USB device plugged in. Any device plugged directly into the SL811HST will always start with USB address 1.

The firmware is based on the EZ-USB framework, allowing modular code development. Below is a list of files and routines indicating their functions:

- **fw.c** : EZ-USB framework, handles enumeration of the EZ-USB development kit, and provides the EZ-USB interface to the PCs or Laptop.
- **host2131.c** : Handles the interface to the EZ-USB development board. This includes sending bulk command data to the 8051 for the SL811HST and returning data captured from the slave device back to the PC. Basically, it handles the IN and OUT transfers to and from EZ-USB's host PC.
- **dscr.a51** : Defines the EZ-USB interface and endpoint pipes.
- **host_811.c** : This is the main file that implements all functions to communicate with the SL811HST. This is the main core of the SL811HST code. You can easily extract the code required to implement just the host controller portion to interface to others microprocessor or MCUs. Below is a list of main routines and their functions explained:
  - ***slave_detect()*** : This is the main loop that is called repeatedly by the TD_Poll() routine in host2131.c. If there is a slave device detect, it updates the data structures containing information about the host controller and downstream ports.
  - ***speed_detect()*** : This function performs three basic setups: detects new device-attachment, identifies the speed of the device attached, and sets up various control registers for proper operation at the correct speed.
  - ***EnumUsbDev()*** : This routine supports the entire enumeration process of a slave USB device. It provides the top-level routines to perform each individual USB request required to enumerate a device. Including USB reset, get descriptors (device, configuration, string and class-specific), set address and set configuration. Note that the current firmware only supports a single interface configuration. At the same time, to facilitate EZ-USB's host software device attachment update, necessary slave device

information is stored during the enumeration process. Examples are VIDs, PIDs, EP0 maximum packet size and each data endpoint's attributes like endpoint address, direction, payload size, etc. This routine only stores up to a maximum of 4 data endpoints of information, which can be increased as required.

– ***ep0Xfer()*** : This function handles control endpoint 0 transactions during enumeration, as well as any other USB host requests. It will called *usbXfer()* to setup and initiate any host requests as necessary by passing information like USB address, payload size, SETUP request types and buffer location of returned data (should there be any). Basically, a USB request will consists of three stages, namely the setup stage (SETUP token with request DATA0), data stage (IN/OUT token with DATAx) and status stage (IN/OUT token with null DATA0). Actual USB traffic is initiated by the routine *usbXfer()*.

– ***usbXfer()*** : This is the core of all USB data and control transactions process. It writes to appropriate registers of SL811HST to initiate a USB transaction as required, be it a write (SETUP/OUT + data) or a read (IN). It also handles low-speed transaction through a hub by appending a PREamble token for any request that goes down all the way to a low-speed device attached to a hub. After each host request is sent, it will wait for an acknowledgement from the slave device by means of an interrupt. It will then determine the type of response from the slave device and terminate as necessary. If there is a request for multiple data from device that is greater than the maximum endpoint zero payload size, it will need to re-arm the SL811HST to grab the next set of data from device until all have been received. For sending data to the device, simply store data into the buffer, give SL811HST the start address of this buffer, set the data length and arm the SL811HST to start USB's SETUP/OUT data transaction.

– ***DataRW()*** : This function is similar to *ep0Xfer()*, except that it is used for data transfer only. By specifying the USB device address, endpoint address, maximum payload size, data length and buffer address, we are able to initiate any data transaction, including IN to or OUT of, the slave device. Of course, you will need to write to the buffer if you are doing an OUT data transfer.

– ***HubPortEnum()*** : Upon successful enumeration of a hub device, endpoint 1 of the hub is used to determine any port change should there be a downstream device attachment or detachment. During a change this routine detects which port had changes, and whether it is a device connect or disconnect. During an attachment, it will also perform reset to the slave device, determine the speed of the device, and finally call *EnumUsbDev()* to enumerate the attached USB device. When a device is disconnected, it simply clears the connection state change-bit of the hub.

As mentioned before, care has to be taken in organizing the code and data memory of the firmware. Below is how the code and data spaces have been allocated.

- Internal 8K EZ-USB Code/Xdata Memory:
  Both USBJmpTb.a51 and Dscr.a51 code are located to EZ-USB internal 8K memory.

- Lower 32K External Memory:
  SL811H_ADDR address at 0x4000
  SL811H_DATA address at 0x4001

- Upper 32K External Memory:
  - Program's Code/Xdata Memory Allocation
    1. Xdata Space: 0x2000 to 0x23FF, using EZ-USB's ISO buffer.
       (size: 0x0400, 1 Kbytes)
    2. Code Space1: 0x8000 to 0x9EFF, for EZ-USB's Fw.c and host2131.c
       (size: 0x1F00, 7 Kbytes)
    3. Code Space2: 0xA000 to 0xDFFF, for host_811.c
       (size: 0x4000, 16 Kbytes)
- Monitor's Code/Xdata Memory Allocation (Should never be used)
    1. Monitor xdata Space: 0x9F00 ~ 0x9FFF
       (size: 0x0100, 256 bytes)
    2. Monitor Code Space: 0xE000 ~ 0xFFFF
       (size: 0x2000, 8 Kbytes)

3.2 About the PC Application Software

The PC Application is used to demonstrate and exercise the EZ-811HS kit in Host mode.

Since the SL811HST demo board works on the EZ-USB development kit platform, the EZ-USB's general purpose driver (Cypress EZ-USB Sample Device) is used. We use this driver to perform basic control of the SL811HST in the form of a Windows-based software application program as shown in Figure 3. With the general-purpose driver, we can communicate with the EZ-USB and request information from slave devices that are attached on the SL811HST.

See firmware section for current device support limitations.

The software can update the devices and display USB device information like USB address, speed type, class type, VID, PID, control and data endpoint's attributes. By changing the USB address field, you can communicate directly with that device, requesting its device descriptor, configuration descriptor, string descriptor or class descriptor. By selecting the appropriate endpoint number and data length, the software is able to initiate data transactions to that endpoint. The slave USB device will in turn respond with data to the host controller, the SL811HST, which will then be transferred back to the EZ-USB's host software. You can easily add functions in the software to help it ensure that the embedded host development is working correctly as expected.

To use this application, perform the following steps:

1. To use mode "Master+Hub" mode, set jumpers 5-6 and 7-8 on the SL811 Demo Board.
2. Connect the USB Cable from the PC to the EZ-USB Development Board.
3. Start the EZ-USB Control Panel by selecting:

   Start\Programs\Cypress\USB\EZ-USB Control Panel

   The EZ-USB Control Panel should open with a single device showing: "Ezusb-0".

4. Select the "Download" button and select the following file:

   C:\CYPRESS\USB\ez811\bin\ Emb_Host.hex

   On the SL811HST Demo Board, first the "HOST" LED will turn on, and then the "ACTIVE" LED will turn on.

5. Start the PC application software (EZ811_DK.exe) in C:\CYPRESS\USB\ez811\bin.
6. You may now plug in a peripheral device such as a mouse to an SL811HS Demo board port (Port1-Port4).
7. The "Downstream Poll" button is there to control the downstream polling function. It can be turned off to allow easier analysis of USB bus traces of downstream devices.

   It should be selected to allow the SL811HS to detect downstream device attach and detach. You may use the "Refresh" button to update the "Device Info" window.
8. You may then select an interrupt endpoint (as on a mouse or keyboard) by selecting the end-point directly in the "USB Device Info" window - the Address, EP, and type are filled in.
9. You can select "Repeat Transfer", and then move the mouse, or select a mouse button, and see the output from the device in the "USB Data Transfers" window.
10. Deselect “Repeat Transfer” to stop transfers.
11. Press “Quit” to exit the Host Application.

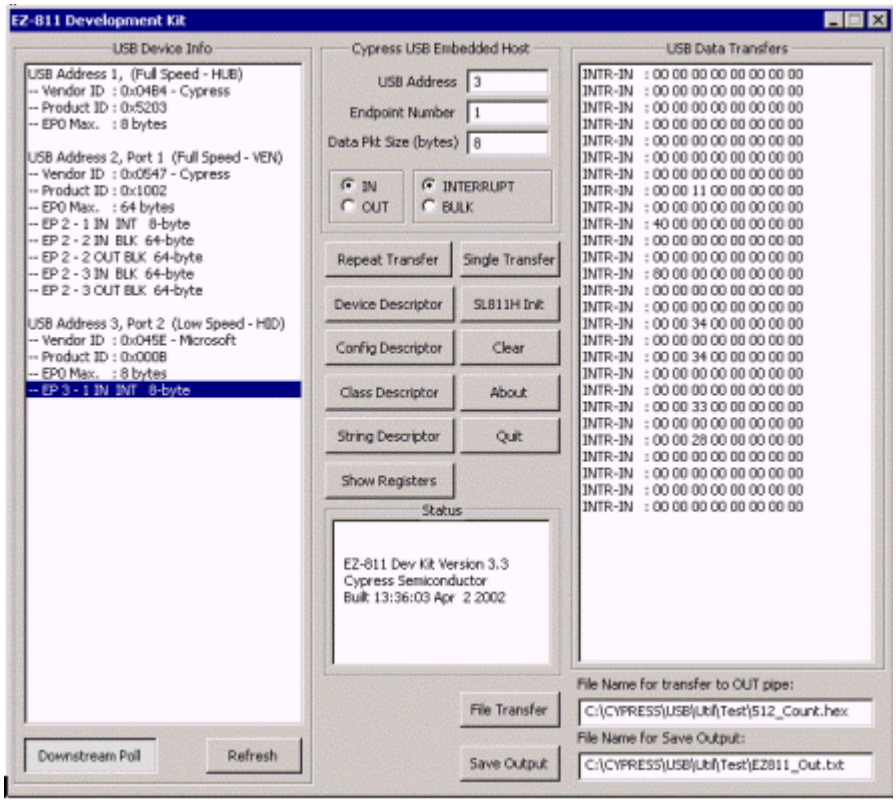

Figure 3. EZ-811HS Host Demo software

4.0 Slave Mode Operation

4.1 About the Slave Mode Firmware

The C:\CYPRESS\USB\ez811\firmware\slave directory contains demo code that will enumerate the SL811HS (in Slave Mode) as a simple HID Consumer Control device. You may download this firmware to the EZ-USB Dev Board (using the EZ-USB Control Panel, as described in Section 3.2), which will configure the SL811HST Demo Board appropriately.

Three consumer buttons are implemented: audio mute, volume up, and volume down buttons.

The demo code will perform the necessary USB and HID-Class requests for successful enumeration of the slave device.

To setup the SL811HST demo board for this example, perform the following steps:

1. Insert the SL811HST demo board onto the ezusb dev kit.
2. On the ezusb dev kit, turn ON bit 4 of S6.
3. On the SL811HST Demo Board, set JP1 to Slave Mode (set jumpers 9-10 and 11-12), JP2 to Full Speed (jumper 2-3), and JP3 to 5V Dev Kit.

JP2 = Full Speed means the left-most jumpers are connected, since pin 1 is on the right.

4. (Optionally) Add 3 tact switches on Port C - Bit 0 to Bit 2.

To connect switches: pull up P5-9 to 3.3v with a 100K resistor, and connect P5-9 with an SPST switch to ground.

Repeat this set-up using P5-10 and then again using P5-11.

You will need three 100K resistors and three SPST switches.

On the EZ-811, P5-9 is Bit 2, P5-10 is Bit 1, P5-11 is Bit 0.

Refer to the schematic for the EZ-USB Development Kit Board.

Mute, Vol Up, Vol Down respectively. All buttons are active low.

5. You may omit Step 4 if you are interested in viewing enumeration only.
6. Now, insert the USB cable to connect the EZUSB board.

   Use the EZ-USB Control Panel to download ez811\bin\HIDSlave.hex.

7. When that is done, you should see host LED off and Active LED off.
8. Using another USB cable and USB port on your PC, connect J2 (USB connector) to the PC (D4 on the EZ-811 Board will turn on).
   a. If necessary, select "`C:\windows\inf`" as the location containing the driver file.
9. If successful, in the "Device Manager" you should see two new entries:
   a. HID-Compliant consumer control device.
   b. USB Human Interface Device.
10. Try the 3 buttons to control the Mute, Volume Up/Down of your sound card.

5.0 Conclusion

With the EZ-811HS development kit, you will be able to setup a system to develop embedded host products using SL811HST. The hardware and firmware provide a good starting point in understanding the basic connection setup using 8051 as the controller to communicate with SL811HST. Code is provided that can be used to initialize the SL811HST as a host controller and perform various USB data and control transaction with external slave USB devices.

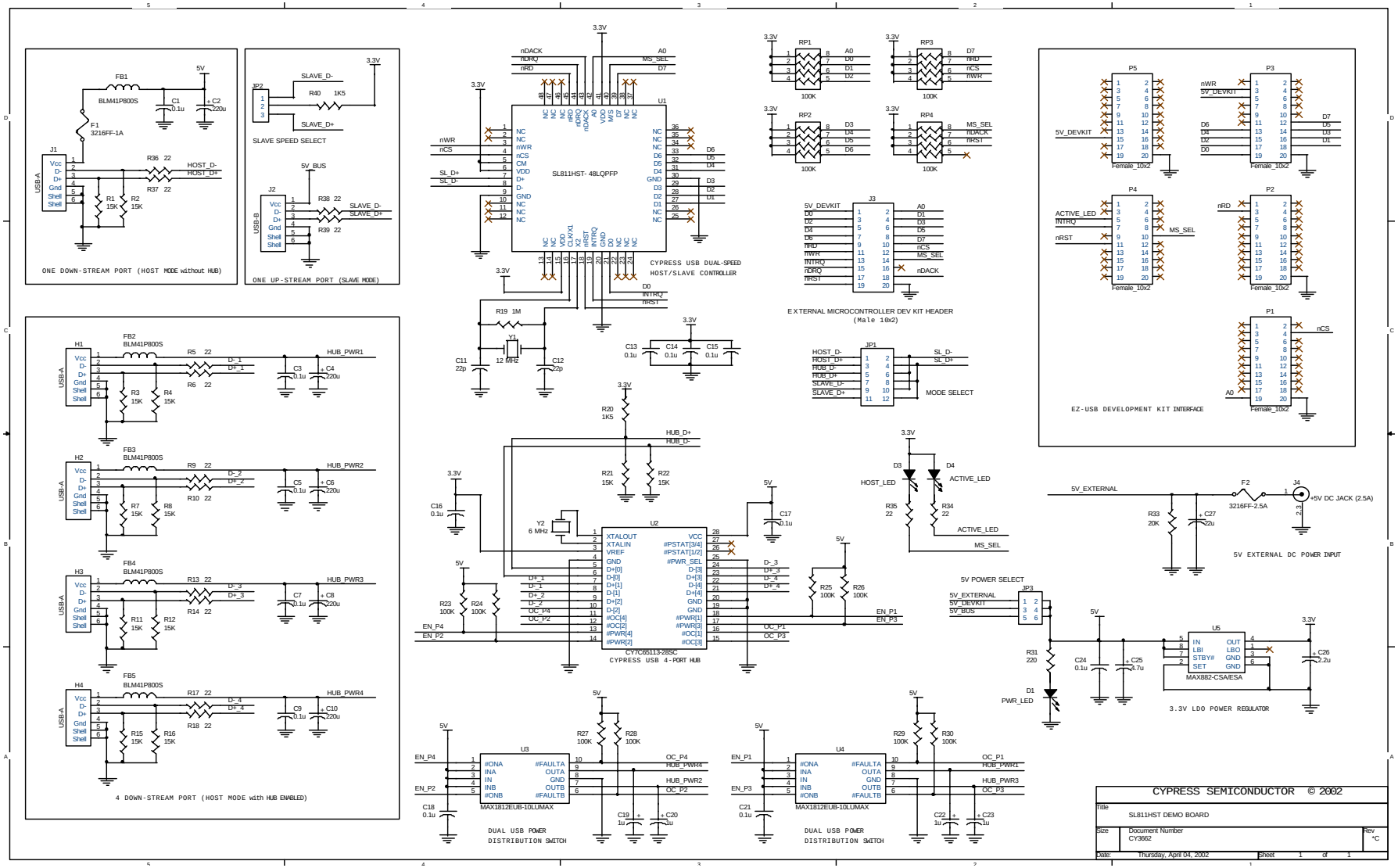

Figure 4. EZ-811HS Development Board - Schematic