Basic Embedded Host Using the SL811HS

AN1215

Introduction

The SL811HS is a full-featured USB embedded host controller. It utilizes a standard address/data bus typical of most 16- and 32-bit embedded processors as well as some 8-bit microcontrollers. This application note addresses the usage of the SL811HS in an embedded USB host application.

System Interface

The SL811HS incorporates an industry-standard address/data bus. The requirements of the embedded processor signals are laid out in the following list.

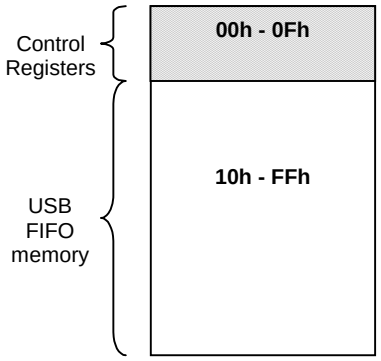
- Active LOW CHIP SELECT signal
- Active LOW READ signal
- Active LOW WRITE signal
- Active HIGH INTERRUPT signal
- Address bus or GPIO
- Data bus, at least 8 bits wide
- GPIO to drive RESET and USB bus power enable

See the application note “Interfacing an External Processor to the SL811HS/S” for more details on host circuitry configuration. This application note provides complete details and examples of the signaling interface to the SL811HS. Most microcontrollers will be able to interface to the SL811HS with little or no glue logic.

Programming Interface

The SL811HS uses a memory mapped interface with an 8-bit address range. The SL811HS supports both a host and peripheral interface, however only the host registers are described in this document. The first 16 addresses (00h-0Fh) are filled with 20 registers used to control the USB host SIE. The addresses ranging from 10h-FFh are used as user assignable USB FIFO buffers. Figure 1 shows the SL811HS memory map.

Figure 1. SL811HS Memory Map



The 20 host control registers are used to enable transactions, interrupts, and report status. Table 1 is a summary of the SL811HS register set. The table is followed by a brief description of each register

Table 1. SL811HS Register Set Summary

Addr.	Write Function	Read Function
0x00	USB-A Control	USB-A Control
0x01	USB-A Address	USB-A Address
0x02	USB-A Length	USB-A Length
0x03	USB-A PID/EP	USB-A Status
0x04	USB-A Address	USB-A Count
0x05	Ctrl1	Ctrl1
0x06	Int. Enable	Int. Enable
0x08	USB-B Control	USB-B Control
0x09	USB-B Address	USB-B Address
0x0A	USB-B Length	USB-B Length
0x0B	USB-B PID/EP	USB-B Status
0x0C	USB-B Address	USB-B Count
0x0D	Int. Status	Int. Status
0x0E	SOF Low	HW Revision
0x0F	SOF High/Ctrl2	SOF High/Ctrl2

USB-A/B Host Control (0x00, 0x08, R/W) – This register is used to provide control over basic host transactions. For example, the register enables USB transactions, sets the transaction direction, and controls the data toggle.

USB-A/B Base Address (0x01, 0x09, R/W) – This register acts as a memory pointer in the range of 10h-FFh. Data that is sent to a USB peripheral is gathered from this internal memory location and sent over the USB. Data reported from a USB peripheral is put at the memory location pointed to by this register.

USB-A/B Base Length (0x02, 0x0A, R/W) – The base length is used to determine the maximum length of a transaction. When the SL811HS sends data to a USB peripheral this register determines the length of the data in the packet. When the USB peripheral reports data back to the host this register determines the maximum data length that will be accepted.

USB-A/B PID/Endpoint (0x03, 0x0B, W) – This register contains the host PID (i.e., SETUP, IN, OUT) and the target endpoint number.

USB-A/B Status (0x03, 0x0B, R) – This register contains the status of the last performed USB transaction. The status includes the received PID (ACK, NAK, and STALL), data toggle, and any error condition.

USB-A/B Address (0x04, 0x0C, W) – This register contains the USB peripheral device address

USB-A/B Transfer Count (0x04, 0x0C, R) – This register contains the residual transfer count after a USB transaction has taken place. In either transfer direction this register value represents the difference between the value written to the Base Length register and the actual number of bytes written from/read into the SL811HS internal memory. If the peripheral tries to send too large of a packet for the SL811HS to handle, the error will be noted in the Status register.

Control 1 (0x05, R/W) – This register enables SOF generation, resets the SIE, allows software to set the USB data line states, sets the USB bus speed, and suspends the SL811HS. The ability to set the USB data lines states is particularly useful for signaling a USB bus reset.

Interrupt Enable (0x06, R/W) – This register allows software to enable an interrupt signal (INTRQ HIGH) on certain events. These events include transaction completion, SOF, device insertion/removal, and resume signaling detection.

Interrupt Status (0x0D, R/W) – This register is read by the external processor upon an interrupt event to find which event caused the interrupt. The events reported in this register correspond to the events enabled in the Interrupt Enable register. The interrupt is deasserted by writing a “1” to any asserted interrupt bit.

SOF Counter Low (0x0E, W) – Sets the low byte of the timer that tracks SOF timing. This register should be written with E0h after reset.

Hardware Revision (0x0E, R) – This register allows device firmware to read the current silicon revision. See the SL81HS data sheet for the most current valid values.

SOF Counter High/Control 2 (0x0F, R/W) – Sets the high byte of the timer that tracks SOF timing, allows software to swap D±, and selects host or peripheral modes. This register should be written with the value AEh after reset to enable host mode and proper SOF timing. The SOF Counter High/Low registers must be initialized before enabling SOF generation.

Two transaction engines, USB-A and USB-B, are provided so that one transaction can be set up while the other is taking place. The transaction engines are symmetric, so it does not matter which one is used and device software is not required to interact with both engines. In fact some simpler applications, such as using a mouse and keyboard on a set top box, would typically only use one engine because throughput requirements for these devices are minuscule and software complexity can be reduced by dealing with one engine only. Both transaction engines are more commonly used in high-throughput applications such as video or mass storage.

USB Host Operation

There are essentially eight parts to interacting with the SL811HS during USB host operation. These parts include:

- Enabling interrupts
- Vbus-on and device attachment
- USB bus reset
- SOF/EOP generation
- USB transactions
- Errors
- USB bus suspend/resume
- Remote wake-up

Each of these items will be described in detail in the following section.

1. **Enabling Interrupts –** Initially the “insert/remove” interrupt should be enabled. This interrupt can be enabled by writing to the Interrupt Enable register previously described.
2. **Vbus-on and device attachment –** Power should be applied to the bus after the interrupts are enabled so that the SL811HS will properly report the attachment of any peripheral devices. Device attachment is reported by the assertion of INTRQ pin. The interrupt source can be read via the Interrupt Status register. If the “insert/remove” interrupt is asserted, the “device detect” bit should be polled. If “device detect” is asserted (asserted = 0b), then a peripheral is attached and the interrupt was more than likely not caused by a peripheral power-on glitch. However, a safer method to absolutely determine device attachment is to wait 5–10 ms before proceeding with device attachment processing. Since the device attachment/detachment

hardware in the SL811HS is not de-bounced, multiple interrupts may occur after the device tries to initially attach to the USB. Once the device firmware has determined that a device is definitely attached, the “D+” bit in the Interrupt Status register is used to determine the speed of the attached peripheral. If “D+” is asserted then the attached peripheral is a full-speed device. If “D+” is not asserted then the device is a low-speed peripheral. USB operational speed is set with the “USB speed” bit in the Control 1 register. If a device attach/detach interrupt occurs and the “device detect” bit is deasserted, software should immediately consider the device as detached.

3. **USB bus reset –** After a device is attached the host is required to generate a USB bus reset. USB reset through a hub is not discussed in this document, however more information can be found in the USB specification. USB bus reset is generated when the SL811HS drives both D+ and D– LOW for 50 ms or more. Bits 3 and 4 of the Control 1 register allow software to directly control the states of the SL811HS D± pins and set both pins LOW. After 50 ms, control of the D± pins should be returned to the SL811HS SIE via the same control bits.

4. **SOF/EOP generation –** The SL811HS must wait at least 2.5 µs after USB bus reset deassertion before beginning any transaction including “Start of Frame” (SOF) transactions. During this “reset relaxation” time software should set up the SOF counter high/low registers as described in the previous section. After the reset relaxation period, SOF generation can be started by writing to the “SOF enable/disable” bit in the Control 1 register. The “ARM” bit in the USB-A Host Control register must also be set to enable SOF generation.

5. **USB transactions –** Transactions should not be started any sooner than 100 ms after the deassertion of USB bus reset. Before performing any USB transactions, the “USB-A” and “USB-B” interrupts (if B engine is used) should be enabled in the Interrupt Enable register. Each transaction will require software to set up to five register values. These include the internal buffer memory address (USB-A/B Host Base Address), the length of the transaction (USB-A/B Host Base Length), the peripheral device address (USB-A/B Host Device Address), the USB PID and peripheral endpoint (USB-A Host PID, Device Endpoint), and the transaction enable (USB-A/B Control). The value of these registers should not be modified after the transaction is enabled or before the transaction is complete. Upon transaction completion, an interrupt will be signaled to the controlling processor. At this time the USB-A/B Status register should be read to determine any errors. The USB-A/B Host Transfer Counter register should be read as well to determine if the entire data packet was properly sent.

6. **Errors –** Errors may occur during USB transactions; for instance if a misbehaving device is attached, or a device is detached in the middle of a transaction. Any error must be dealt with in software on the controlling processor. How a USB host stack should deal with specific errors is beyond the scope of this document, however the USB specification covers most error cases in detail.

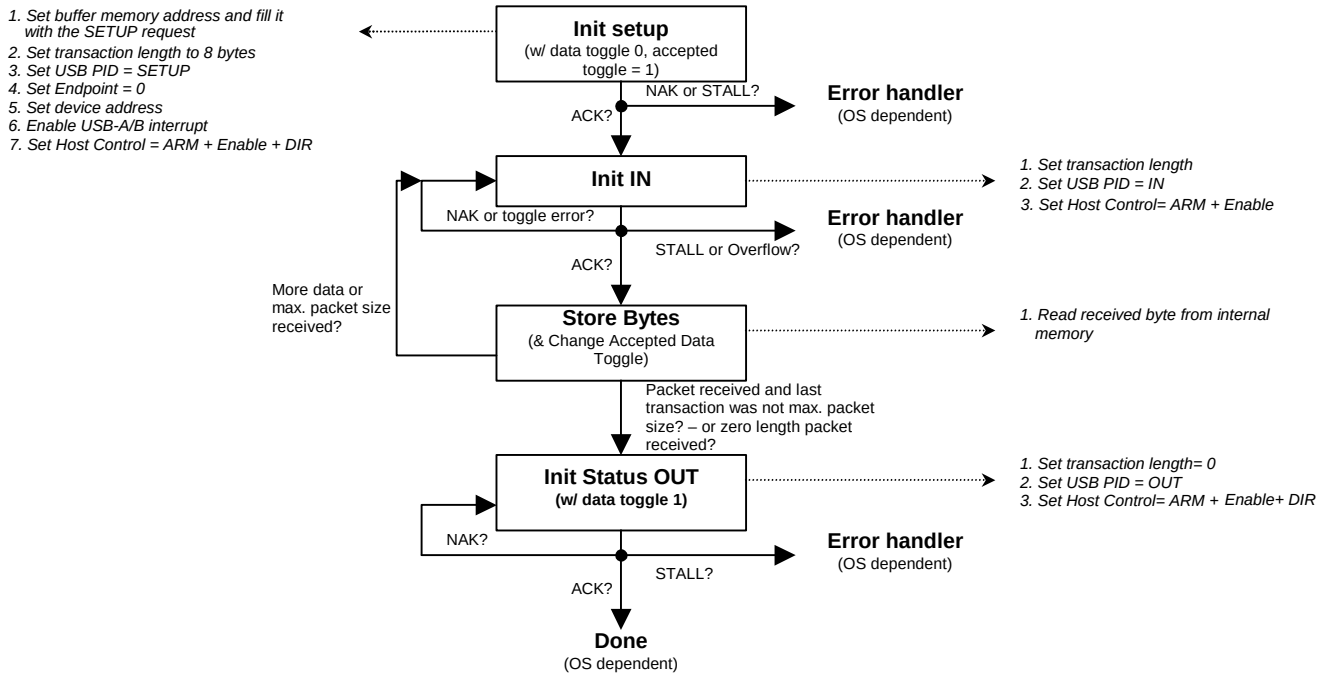
7. **USB bus suspend/resume –** USB bus suspend essentially entails shutting off SOF generation and not performing any more transactions until the USB is resumed. SOF generation can be disabled in the Control 1 register using the “SOF enable/disable” bit. A USB resume simply requires that SOF generation be turned back on.

8. **Remote Wake-up –** Remote wake-up signaling can be automatically detected and can cause an interrupt. Resume detection should be enabled before the USB is put into suspend. First, the “device detect/resume” bit in the interrupt enable register is set in order to detect the resume event. This interrupt enable bit should only be used to detect resume signaling. This interrupt should not be set to detect device insertion, as it will continuously interrupt during USB traffic. After the USB resumes, this bit should be de-asserted. The final step is to set the “suspend” bit in the Control 1 register to enable resume signaling detection.

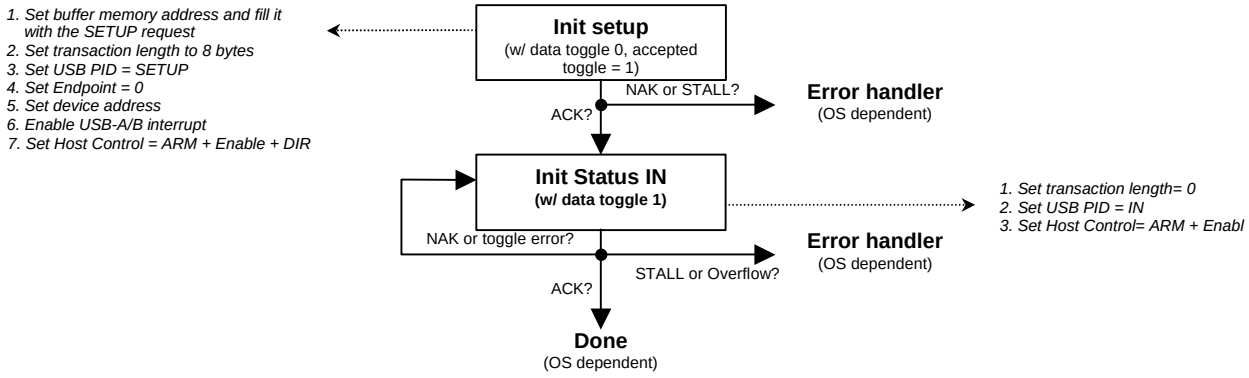
Endpoint 0 Control Transactions

This section provides state diagrams of USB host controller transactions. A USB host may need to support up to three types of transactions on endpoint 0. These transactions include Control-Read, No-Data Control, and Control-Write transactions. Operations that require one or more register writes are shown in bold. State change decisions are shown with a question mark and can be determined by reading the USB Status register and the USB Host Transfer Count register. Upon any disconnect event during a transaction the flow should automatically transition to the error handler. Non-control endpoint transactions, such as bulk, isochronous, or interrupt are essentially simpler subsets of control transactions, so their details are not covered in this document.

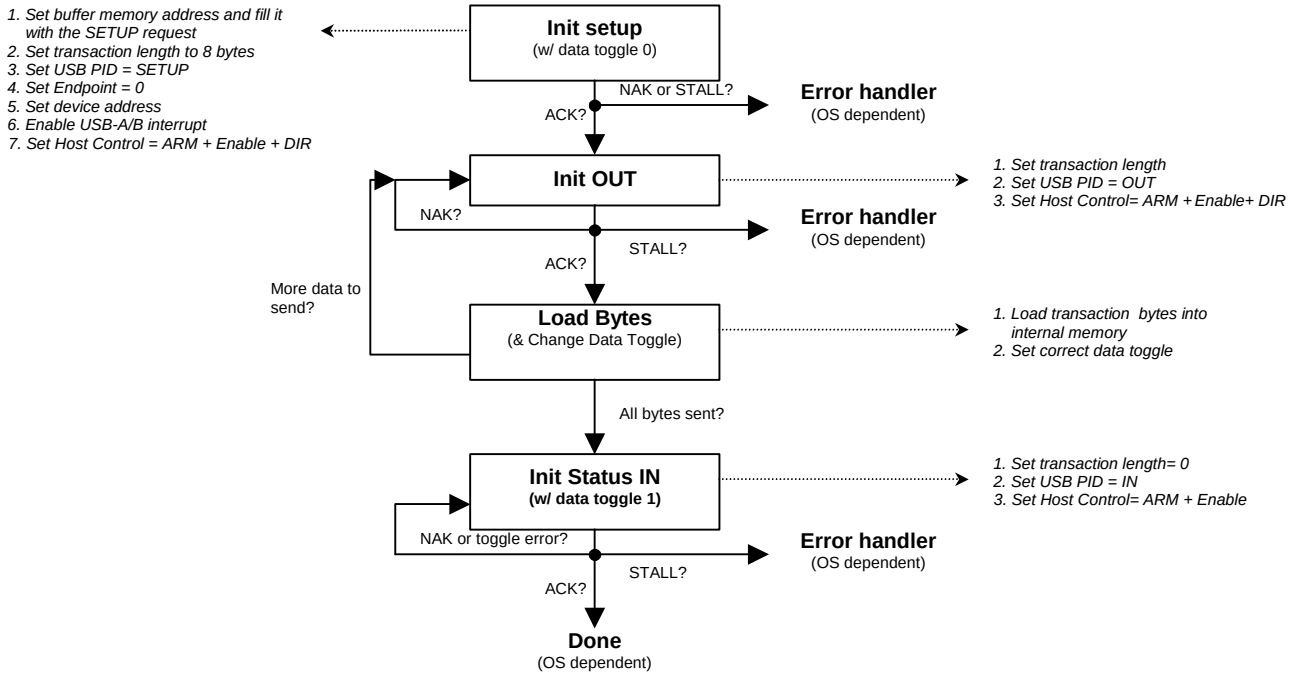
Control-Read



No-Data Control



Control Write



USB Host Stack Support

Cypress supports a number of host stack implementations for the SL811HS by providing a host controller driver. Supported operating system stacks include VxWorks, WinCE, and Linux. Cypress also offers a SL811HS development kit with host firmware examples. Some implementations may be available on the Cypress web site. Others not listed on the web site are available from Cypress USB applications support upon request.

Summary

The SL811HS can be used as a versatile and full-featured embedded USB host controller. The combination of a standard signaling interface and simple control registers allows the SL811HS to be integrated with a small or large scale embedded USB host stack. For further questions and assistance please contact Cypress USB applications support.

In March of 2007, Cypress recataloged all of its Application Notes using a new documentation number and revision code. This new documentation number and revision code (001-xxxxx, beginning with rev. **), located in the footer of the document, will be used in all subsequent revisions.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709
Phone: 408-943-2600
Fax: 408-943-4730
http://www.cypress.com