

Kevin Wu
Embedded USB Final Project
Embedded USB, Summer 2008
2008-09-12

My final project is based on the SiLabs C8051F340 development kit, and uses the SiLabs USBXpress API. It performs A-to-D conversions on the built-in temperature sensor, or on an analog input pin. When a threshold temperature is reached, an on-board LED begins to flash, and stays lit when the temperature is in excess of the threshold. The LED symbolizes either a speaker or some other peripheral device that is activated when the threshold temperature is reached.

The USB interface is initialized with `USB_Clock_Start()` and `USB_Init()`, and USBXpress interrupts are enabled with `USB_Int_Enable()`. The driver for the device is the same driver that comes with USBXpress for `TestPanel.exe`. The device is bus-powered.

The ADC operates in single-ended mode (negative input is ground), with $V_{DD} = 3V$ as the reference voltage. It performs a conversion whenever timer2 overflows. timer2 is clocked by the system clock divided by 12, about 1MHz.

The associated host application allows the user to:

- switch between temperature sensor and input pin as the positive input to the ADC
- enable or disable timer
- display ADC results. Depending on which ADC input is selected, results are displayed in degrees Fahrenheit (if temperature sensor is selected) or as hex (if analog input pin is selected)
- check status of device (timer status, ADC status, ADC input)

Whenever the user selects an option on the host application, a packet is created, containing the selected option, and is sent to the device using `SI_Write()`. This triggers an interrupt on the device, and in servicing the interrupt the device reads the packet, using `Block_Read()`, and determines the purpose of the packet. Depending on the purpose, the device could do the following:

- set appropriate flags in the output buffer to indicate status of ADC, timer, and ADC input
- enable or disable timer2
- change `AMX0P` to select temperature sensor as ADC input
- change `AMX0P` to select pin P3.5 as ADC input

For all cases, the device will send a packet back to the host using `Block_Write()` after performing the selected operation. This is because the host application, after sending a packet to the device, waits for a return packet. To do this, it checks the receive queue, using `SI_CheckRXQueue()`, until the queue is ready. Once it is ready, it reads the queue, using `SI_Read()`. Depending on which option the user chose, this returned packet is either parsed for content display, or discarded.

In this sense, the operation of both the device and host application is passive. The device and host only send packets "upon request". On the device side, an interrupt is triggered when a packet is received, and a return packet is then sent back to the host.

On the host side, the receive queue is polled only after a packet has been sent to the device, thereby completing the communication circle. For example, if timer2 is enabled, the ADC is constantly performing conversions and setting values in a buffer, but it is only when the user requests the ADC result on the host that the buffer is sent from the device to the host.

See comments in device and host application source code for further detail.