BIT-BANG USB--PERHAPS THE EASIEST USB INTERFACE YET!

By Don L. Powrie

Introduction

Considering the complexity of the USB interface, using a USB port to toggle an LED is a little like using a sledgehammer to drive a small nail. But that is exactly what this article is going to show you how to do.

Just imagine being able to control a bank of eight relays, read eight switches, or a mixture of both--all via USB using a single chip with no microcontroller or firmware required! In the arena of "easily-implemented USB", the folks at FTDI (**www.ftdichip.com**) have done it again by releasing their new FT245BM USB-FIFO and FT232BM USB-UART devices. This article will also show a rather simple design for a low-frequency arbitrary waveform generator; again, with no microcontroller required.

By combining one of these new devices with FTDI's royalty-free drivers (compatible with Windows 98/ME/2000/XP), users can interface an electronic device to a host PC via USB in a matter of minutes or hours rather than weeks or months.

FT245BM Overview

For those of you already familiar with FTDI's previous device, the FT8U245AM, the list of new features includes a new "Bit-Bang" mode (which we will cover in this article), reduced external component count, features that improve throughput performance, and support for isochronous transfers. (Consult the FT245BM datasheet for all of the details associated with these new features.)

For those of you who are new to USB or FTDI's USB-FIFO devices, please read on for a quick overview. The USB ports on the back of most of today's PC's offer a high-speed digital interface between the host PC and an electronic peripheral. Electronic devices designed around the FT245BM device can achieve host data rates approaching 8 megabits per second. They also offer "hot swapping", are self-powered up to 500 milliamps (no more wall warts!), generate no interrupt conflicts to contend with, and provide the convenience of knowing that most PC's on the planet have the correct interface for your new project built right in.

The FT245BM is FTDI’s second-generation USB-to-FIFO chip that makes easy work of connecting your electronic device to a host PC via USB. The electrical interface between your microcontroller and the FT245BM is comprised of eight data lines and four handshaking lines. (If you will be using the Bit-Bang mode, all that is needed are the eight data lines.) Device drivers are provided royalty free that make the FT245BM look like an RS-232 device. Once a serial port has been opened by your application program, bytes sent to the port are rerouted to the Windows scheduler and USB port via FTDI's drivers.

A second version of the device drivers based on a DLL (Dynamically Linked Library) is also available as a free download. The DLL version of the drivers offers a higher data rate but requires that your program load the DLL at runtime. Example source code for Visual C++, Visual Basic, and other programming languages illustrating how to load the DLL is available for download from both the **ftdichip.com** and **dlpdesign.com** websites.

The FT245BM is only available in surface mount form from FTDI. You can either design a PCB yourself and use a fine-tipped soldering iron to solder the chip in place or use the DLP-USB245M module as shown later in this article. The DLP-USB245M takes advantage of the new features of the FT245BM and allows for easy integration into your hardware design via its standard 0.6-inch DIP interface. Features of this module include an EEPROM for storing description strings, up to an 8-megabit-per-second data rate, and full compatibility with FTDI’s royalty-free drivers. A datasheet for this module (including a schematic) can be downloaded from **dlpdesign.com**.

Bit-Bang Mode

Both the FT232BM and FT245BM devices support the new Bit-Bang mode. Utilizing the Bit-Bang mode requires that the application program running on the host uses the DLL version of FTDI's USB device drivers.

Keep in mind that the Bit-Bang mode is a side feature of sorts in that the primary intention of the FT245BM is to interface a microcontroller/DSP/FPGA/etc. to a host PC. The Bit-Bang mode is simply one of the latest features of this extremely useful little chip. For a more detailed description of what the FT245BM can do for your next project, refer to my earlier articles that discuss the FT8U245AM in the May 2001 and November 2001 editions of *Nuts & Volts*. (Both of these articles are available for review on-line through the **dlpdesign.com** site)

Four commands are used to access the Bit-Bang mode. The first, FT_SetDivisor(), controls the rate at which data is latched to the output data lines on the chip. For example:

```
FT_STATUS status;
USHORT dta;
Dta = 0x400;
status = SetDivisor(dta);
if(status != FT_OK)
{
        CString str;
        str.Format("Data entered (%d) is not a valid divisor.", dta);
        AfxMessageBox(str);
}
```

This allows data to be clocked out on the eight data lines at a predetermined rate. Note that in Bit-Bang mode there is no output line provided for “latching” data into an external device—the data simply appears on the eight data lines. One of the eight data lines could, however, be used as a latch requiring that the host software keep track of the state of both the seven-bit data and the latch line separately. The second command, FT_Write(), actually sends the data to the FIFO memory in the FT245BM where it waits to written to the output data lines. The third command, FT_SetBitMode(), allows you to select which bits are inputs and which are outputs. This command is also used to enable and disable the Bit-Bang mode. For example:

```
//enable bit bang mode
status = SetBitMode(0x0f, 0xff);
if(status == FT_OK)
{
        //bit bang mode active
}
```

This code will activate Bit-Bang mode (2nd parameter of SetBitMode) with D7 through D4 set to be inputs and D3 through D0 set to be outputs (1st parameter of SetBitMode).

The forth command, FT_GetBitMode(), is used to read the current state (high/low) of the eight data lines.

Data that is transferred from the FIFO to the output data lines is latched on the lines until a different byte is sent. No external buffer or latch is required to maintain the status of the data lines while in Bit-Bang mode. (When not in Bit-Bang mode, data is only held on the data lines as long as RD# is held low.) This technique is demonstrated in the next section in which I will use the sledgehammer to drive a small nail…

Switch/LED Example Circuit

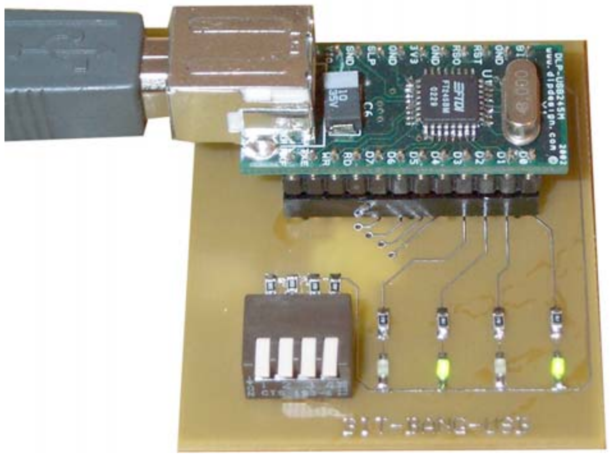

Figure 1 – Switch and LED circuit using the DLP-USB245M ($25 from **www.dlpdesign.com**)

Figure 1 shows a DLP Design DLP-USB245M module configured into what is probably the simplest implementation of a USB device ever seen. The circuit (**Figure 2**) will allow the host program to read the state of the four switches and turn each of the four LED's on or off individually with no microcontroller required. In fact, if Version 2.0 of the DLP Design Test Application (more on this later) is used to control this circuit, then <u>no</u> software development of any kind is required to toggle the data lines.

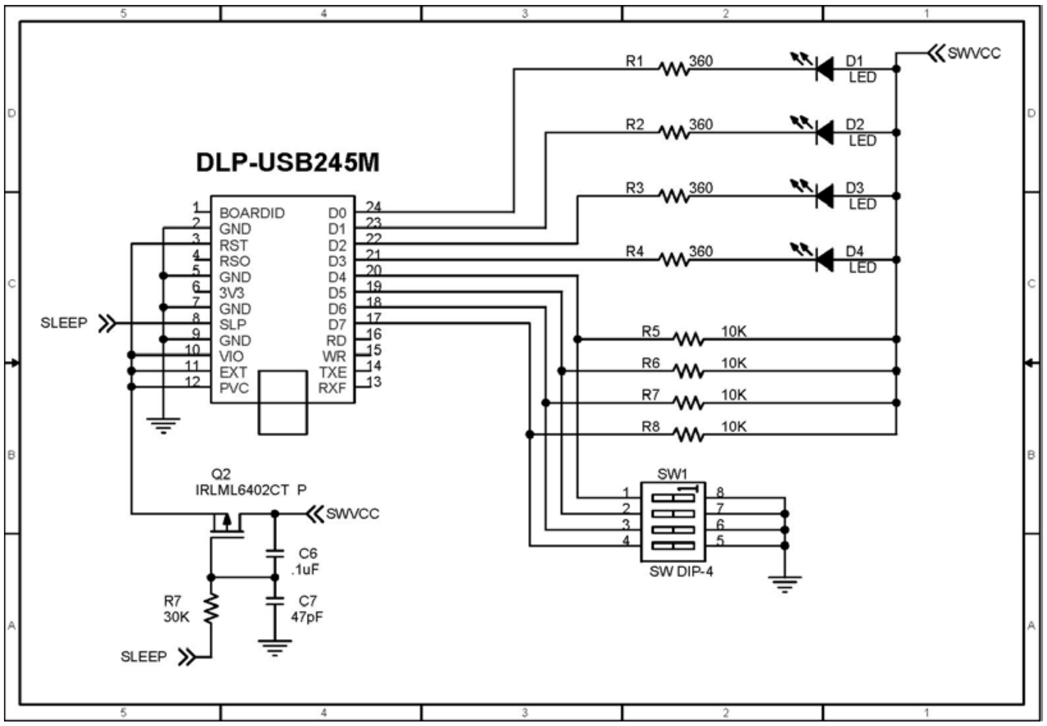

Figure 2 – Switch and LED circuit

Driver transistors and relays for USB control of up to eight relays could replace the switches and LED's. With some additional external circuitry, devices like event timers, device programmers, etc. could also be developed that require no firmware.

Both this circuit and the next employ a P-channel MOSFET to control power going to the target electronics. Once Windows has enumerated the DLP-USB245M module, the FT245BM takes its SLEEP# line low. This circuit could also be used to control power to higher-current applications since the RC network connected to the gate of the MOSFET sets the rise-time (limiting the inrush current) for the power going to the target electronics.

Note: Care must be taken to ensure that all target circuitry does not exceed the maximum 500mA (assuming direct connection to a host computer or self-powered hub) available from the host USB port. If your peripheral is connected to a bus-powered hub, then it must draw no more than 100mA.

Arbitrary Waveform Generator

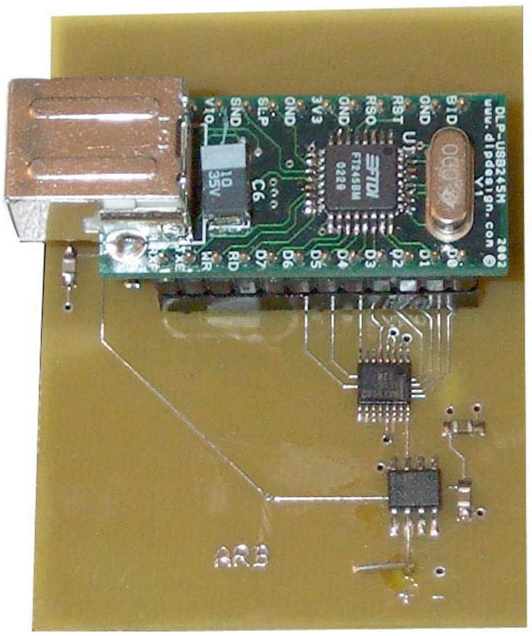

Figure 3 – Arbitrary Waveform Generator

Figure 3 illustrates another design that takes advantage of the new Bit-Bang mode. The DLP Design DLP-USB245M is used once again to prevent having to work with the surface-mount FT245BM device.

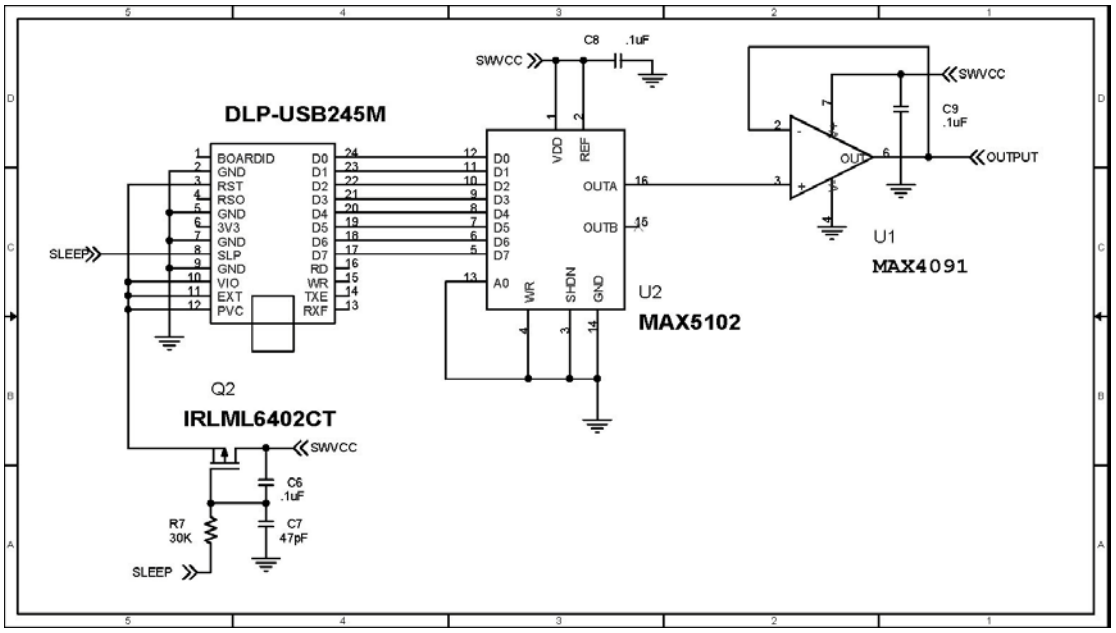

Figure 4 – Schematic for the Arbitrary Waveform Generator

In this example circuit (**Figure 4**), a simple eight-bit DAC is used to create a rather basic low-frequency arbitrary waveform generator. The data latch in the DAC is placed in transparent mode such that the data that shows up at its parallel input is passed directly through to the device's output without the need to be latched by an external device. By setting the update rate using the FT_SetDivisor() command, the update rate into the DAC can be controlled, and relatively precise waveforms can be generated. The desired waveform can be created on the host PC and then written to the FT245BM via USB for recreation by the DAC. (Additional circuitry could also be added to expand the output voltage beyond the 0-5 volts provided by the USB port.)

Test Application

So you say you don't want to have to write an application program for the host? Well, you may not need to depending upon the complexity of your application. If all you want to do is send a few bytes to the USB device and observe the response coming back, then the DLP Design Test Application (available as a free download from **dlpdesign.com**) may be all that you need. This application is compatible with both versions of FTDI's device drivers, and it is the perfect companion for helping debug new designs. A second version of the Test Application (Version 2.0) that supports all of the latest features in FTDI’s USB DLL drivers (including Bit-Bang mode) is also available for a shareware-level fee of $20 ($13 with the purchase of any DLP Design product).

Conclusion

As mentioned earlier, the Bit-Bang mode only utilizes one small feature of the FT245BM device…but just imagine the possibilities! An electronic device that fits in the palm of your hand, requires no other power source, works with just about any PC, and requires no microcontroller or in-depth knowledge of USB. The enhanced functionality of the new FT245BM device provides developers with just about any level of experience the ability to easily connect to the USB interface.