# Simple Method Opens Path for USB in Embedded Markets

*Storage devices are increasingly important to add value to a system. Now there is a method to add Universal Serial Bus (USB) storage to non-computer terminals.*

Universal Serial Bus (USB) interfaces are everywhere today. Low-cost flash memory drives and all kinds of USB peripherals are readily available, but they focus heavily on the personal computer market. When designers attempt to use these peripherals in the 8- and 16-bit embedded market, they quickly learn that implementation, cost and power consumption are prime issues. One reason for the challenge lies in the embedded controllers that embedded systems often use. Devices like the PIC family of controllers from Microchip serve widely with a broad range of memory densities and peripherals, but they lack the interfaces, resources and performance to incorporate a USB host controller.

This article looks at the detailed implementation and programming of a design to link a low cost PIC microcontroller to a flash drive through a USB2.0 full-speed interface. Topics include the hardware design of an embedded interface based on a PIC microcontroller and a Vinculum USB interface chip by Future Technology Devices International. (FTDI). There also is discussion of the typical programming of both items, to enable ubiquitous USB flash drives to work as removable storage in a wide array of embedded applications.

In this example, a VNC1L Vinculum controller IC provides the interface between the PIC as the system controller and a USB2.0 full-speed port. This allows the creation of a USB flash memory drive connection, for example, while requiring a minimum of implementation time and overhead

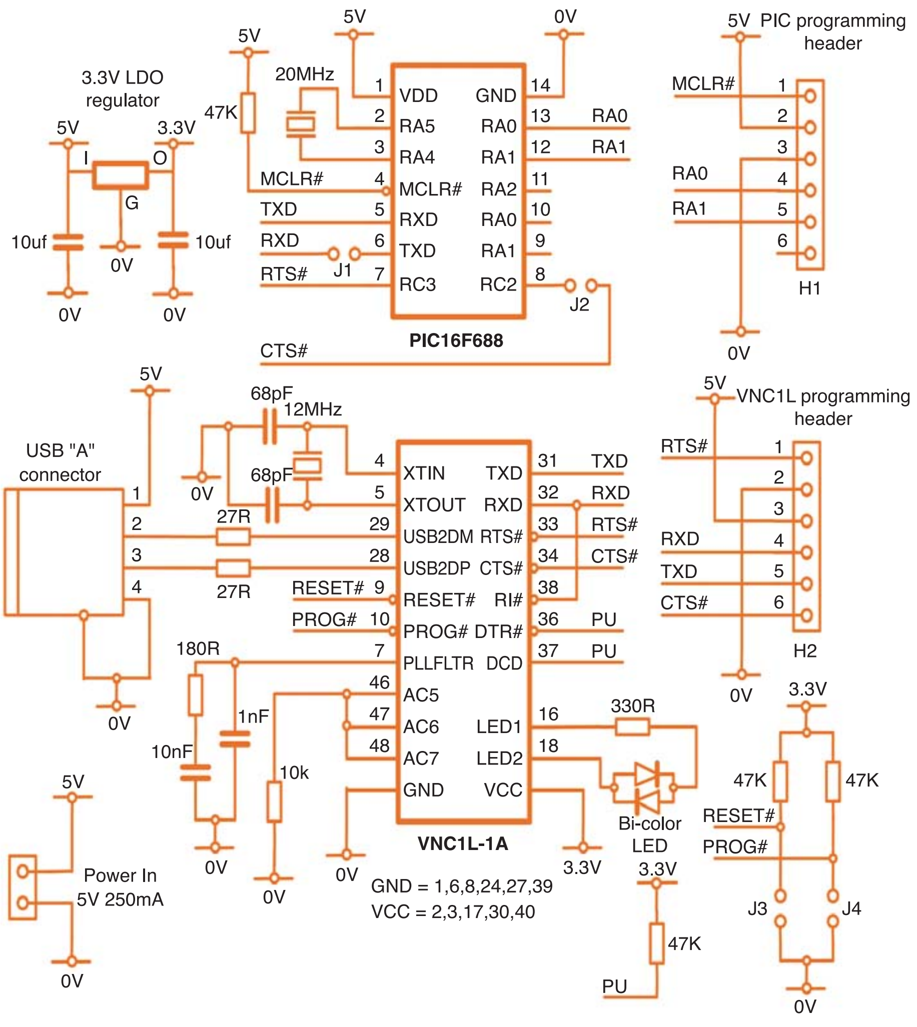

Fig. 1: Interfacing a USB flash drive to a PIC microcontroller

## A Look at the Controller

The controller is based on a custom processor core with twin direct memory access (DMA) engines to accelerate data transfers, and a 32-bit numeric co-processor to optimize the calculations for the file system. All these devices occupy a single chip that offers 64KB of embedded flash program memory as well as 4KB of internal data SRAM.

Vinculum specifically targets the embedded USB controller market and requires a minimum of external support components. One key feature of the Vinculum core is that its code length is significantly shorter than that of common microcontroller unit (MCU) cores. Reducing the code overhead of the core makes it possible to squeeze much more functionality than before into the on-chip e-flash memory. These features are complementary to a PIC-based embedded system. In the system (Fig. 1), Vinculum links a small PIC MCU to a USB "A" connector and hence to a USB flash drive.

Copyright ©2007 Dempa Publications, Inc.

## System Operation

The PIC is the system controller, taking data from sensors or other sources through its general purpose input-output (I-O) pins (RC0, RC1, RA2 on pins 9,10,11). It then converts the data format and writes that data in a stream to a file on the flash drive. Commands and data travel through TXD (pin 6) to the VNC1L RXD (pin 32). VNC1L handles the creation of the file allocation tables (FAT-12, -16 or -32) and data storage on the USB flash drive.

Communication with the drive takes place through USB2DM and USB2DP on pins 28 and 29. The same pins handle reading of data from the flash drive. The system then sends the data from the VNC1L TXD (pin 31) to the RXD (pin 5) of the PIC for use by the system firmware.

Firmware on the PIC controls the system. Additionally, the PIC issues instructions that control the transfers. Standard firmware on the Vinculum interprets these instructions. This is merely a simple description of the system. Other items and steps are necessary to complete the design. The devices need power, they need crystals to control their clocks, and they need programming.

Using a 20MHz crystal on pins 2 and 3 of the PIC allows baud rates of up to 115,200 bits per second in the Universal Asynchronous Receiver Transmitter (UART) interface. For comparison, the maximum baud rate with the internal 8MHz oscillator is 9,600 bits per second. Therefore, the 20MHz crystal greatly improves the performance of the system. The PIC firmware uses I-O pins RC2 and RC3 to simulate the Request to Send (RTS) and Clear to Send (CTS) handshake signals with the VNC1Ls UART interface.

The system requires a regulated, 5V power supply unit at 250mA, providing up to 200mA at the USB “A” connector, as well as 25mA to power the VNC1L and 25mA to power the PIC16F688. VNC1L requires a 3.3V supply, which comes from a 3.3V low dropout regulator, and has 5V-tolerant I-O pins, enabling it to connect to the PIC without using level shifters.

For low-power applications, the VNC1L offers a 2mA sleep mode. To wake the device, the system can strobe the ring indicator (RI) pin (pin 38) of the UART interface. If this is connected to the RXD line, it can be triggered by an incoming dummy command to wake up the device.

This design also incorporates a bi-color light-emitting diode (LED) that serves as a status indicator. Power for this item comes from pins 16 and 18. The status light indicates successful enumeration of the USB flash drive, as well as access to the file system.

## Firmware Programming

To program the ViNC1L, there is standard firmware, called VDAP (Vinculum Disk and Peripheral) that interprets the commands coming from the PIC. These VDAP commands are DOS-like instructions like DIR, RD and WR. The command set also supports single-byte hex commands, which are suitable for control by a microprocessor.

VDAP commands are part of the PIC firmware, helping to control access to the USB flash drive. A typical sequence would be to create a file, read or write data to the file and then close the file.

## PIC Programming

This design contains two programming headers, one for each device, on the assumption that users will want to operate in development environment. For a production design, users could pre-program both devices before insertion on a printed circuit board, thereby eliminating the headers and jumpers.

During normal operation, J1 and J2 should be populated and the other jumpers left open. To program the VNC1L, designers would remove J1 and J2 jumpers to isolate the VNC1L UART inputs from the PIC outputs. They can disconnect the 5V power supply unit, and then connect a TTL-232R-3V3 cable to H2. The USB side of this cable connects to a personal computer with the VPROG programming utility installed. Designers can populate J4 to pull the PROG# pin of the VNCL1A low. They then can temporarily short J3 to reset the device and put it into programming mode. After programming, it is important to restore the jumper settings to the operational positions.

The programming header for the PIC connects to pins RA0 and RA1 and MCLR# of the device. The header supplies the 5V programming voltage and supply. Engineers should disconnect the 5V power supply unit before programming the PIC microcontroller. If the header connects to a standard PIC development environment like PICKit2, it becomes possible to use Microchip’s debug and downloading tools.

FTDI has prepared a sample of the source code in C programming language, and has made it available at http://www.vinculum.com/projects/SampleCsource.zip. In this sample, the PIC waits for the system to detect a flash disk, then opens a file called hello.txt. The system next writes the entire “Hello World” text, including carriage return and line feed characters to the file. It then closes the file and waits for the user to remove the disk.

## Conclusion

FTDI’s Vinculum VNC1L provides an easy-to-use, easy-to-program interface between a low-cost microcontroller and a USB 2.0 low- to full-speed peripheral. The DOS-like command set makes it easy to write and debug a data transfer routine within the microcontroller environment. The simple layout provides a low-cost USB host implementation for embedded systems. This allows the use of low-cost, ubiquitous USB flash drives as the data storage media for the system. It also supports software upgrades in the field. The VNC1L device also can help to connect many other USB peripherals besides mass storage devices.

Additionally, the Vinculum IC accounts for less than 10 percent of the power budget of the USB interface and represents even less of the system’s power budget of the system. With this device, engineers can add USB2.0 host controller ports to portable devices in an easy and straightforward manner. The VDAP firmware and a document describing the complete command set are available from FTDI’s Vinculum Web site at www.vinculum.com.

## About This Article

*The author, Fred Dart, is Managing Director of Future Technology Devices International, (www.ftdi*chip.com), a *fabless semiconductor* company *in Glasgow, Scotland. Its speciality is the design and supply of silicon and software solutions for USB. FTDI has offices in Oregon, the United States and in Taipei, Taiwan.*

Copyright ©2007 Dempa Publications, Inc.