Vinculum IC speeds USB flash drive connectivity to microcontrollers

By Fred Dart (MD, FTDI)

Although this media has been readily available for a number of years, use of USB flash drives has, to date, been restricted to platforms with adequate processing power such as PCs and 32-bit embedded systems. FTDI has now opened up the use of USB flash disks to microcontrollers with the introduction of their Vinculum series of intelligent USB host controllers.

The **Vinculum VNC1L** IC provides USB host interface and data transfer, and supports the most popular device classes; mass storage, printer and human interface device (HID). HID class devices typically include USB keyboards, joysticks and mice. When interfacing to flash drives, Vinculum manages the file allocation table (FAT) structure by using a straightforward command set. The device has an 8-bit core together with a 32-bit co-processor, dual DMA controllers, 64 k embedded flash and 4 k internal SRAM memory. Vinculum features two USB 2.0 low and full speed, host and slave ports, universal asynchronous receiver transmitter (UART), serial peripheral interface (SPI) and parallel first in first out (FIFO) interfaces. It also has two PS2 legacy ports for keyboard and mouse, and up to 28 general-purpose input output (GPIO) pins depending on configuration. The current Vinculum handles both low and full speed USB 2.0, which provides data link at up to 12Mbytes/s and will interface to all USB2.0 peripherals, as well as older USB1.1 devices. This is more than sufficient for USB flash drive applications and is deliberately targeted to keep the size, cost and power down to a level that is acceptable for embedded applications. Power consumption is 25 mA for the 3.3 V core and the 5-V-safe I/O interface.

Vinculum provides USB host capability to microcontroller-based products that previously did not have the hardware resources available. A wide range of consumer and industrial products, such as intelligent domestic appliances, meter readers and vending machines, can now incorporate USB peripheral connectivity. For product designers this is now greatly simplified by the availability of FTDI's new **VDRIVE2 module**. Packaged in a neat snap-in enclosure (**Figure 1**), VDRIVE2 consists of a Vinculum IC, USB "A" socket and a few support components. Only four signal lines plus a 5V supply and ground are required. By using the Vinculum Disk Interface Firmware Specification (DIFS) the I/O interface can be selected between the serial UART and SPI using the on-board jumper pins. A bi-colour led provides power and status indication.

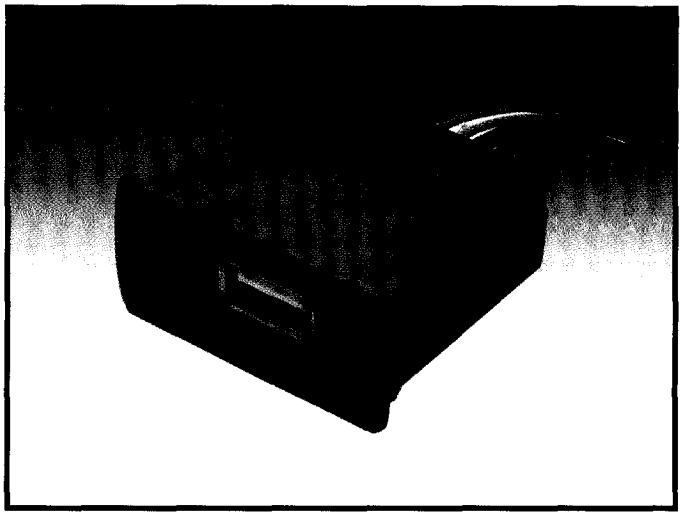

Figure 1. The VDRIVE2 snap-in module.

Adding a PIC microcontroller and a few other components, the VDRIVE2 module can be turned into a flash disk based data logger. **Figure 2** shows the schematic of a simple application. The AC signal input is connected to the 10-bit analogue to digital converter on board the Microchip PIC. The PIC code takes a predefined number of samples and then writes the corresponding ASCII values to a comma separated value (CSV) file on the USB flash disk attached to the VDRIVE2 module. Vinculum's DOS like ASCII commands simplify the task of file handling. An extended ASCII command set is designed for use with a terminal during test and development, whilst a shortened hexadecimal version is used with a microcontroller. Currently, Vinculum's command set has five categories: Directory, File, Power management, Debug and Miscellaneous. **Table 1** illustrates some example commands.

(070014-I)

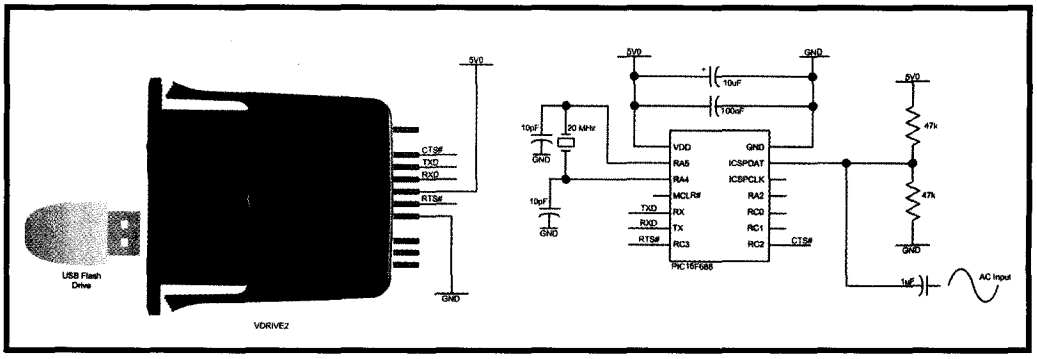

Figure 2. Connecting the VDRIVE2 to a PIC micro is as simple as this.

Table 1. Vinculum — command monitor system examples.

Extended ASCII command for terminal mode	Hexadecimal command for microprocessor mode	Command function
Directory examples		
DIR<cr>	$01,$0D	Lists the current directory
MKD<cr>	$07,$20,<name>,$0D	Make directory
CD<sp><name><cr>	$02,$20,<name>,$0D	Current directory is changed to the new directory <name>
File examples		
RDF<sp><size in hex (4 bytes)><cr>	$0B,$20,size in hex(4 bytes),$0D	Reads the data of<size in hex> from the current open file
OPW<sp><name><cr>	$09,$20,<name>,$0D	Opens a file for writing with the command WRF
Power management examples		
SUD<cr>	$15,$0D	Suspend the disk when not is use to conserve power
WKD<cr>	$16.$0D	Wake disk