International Refereed Journal of Engineering and Science (IRJES)
ISSN (Online) 2319-183X, (Print) 2319-1821
Volume 2, Issue 3(March 2013), PP.17-23
www.irjes.com

Flash Drive To Flash Drive Data Transfer

Mukesh Tiwari, Siddhartha Motghare, Jimit Gada, Unmesh Barhate
Electrical Department, Veermata Jijabai Technological Institute, India

ABSTRACT : *This paper explains the methodology and mechanism for the transfer of data from one flash drive to another. The mechanism includes the use of VDIP2 module with VNC1L chip incorporated on it for the purpose of data transfer. VNC1L chip runs on different application oriented firmwares, out of which VDFC firmware is used to connect two BOMS devices to VDIP2 module, thus it acts as a USB host controller. This module is controlled using microcontroller over serial UART communication. Thus microcontroller acts as a monitor which is used to command this module in order to perform desired task. In this way, it is possible eliminate the use of bulky desktop for the purpose for data transfer between two pen drives using microcontroller.*
Keywords – *Flash drive, Pen drive copier, USB data transfer, VDIP2 module, VNC1L chip, FTDI.*

I. INTRODUCTION

The popularity of Universal Serial Bus (USB) storage devices is an indication of the computer user's need for a fast, large capacity and easily accessible system for data storage and transfer. As the development of USB enabled peripherals increases, the USB has rapidly become a de facto standard in communication with the Personal Computer and has led to new technologies for interfacing memory devices. These memory/storage devices connect to the USB ports and appear as removable storage device in personal computers, the most popular of which is the USB Flash Drive. The disadvantage of USB storage devices is that being a peripheral device, it needs a host, usually a Computer to initiate and mediate communications between two USB storage devices or other peripheral devices.

As a solution to the USB Flash Drive disadvantage, the project aims to develop a device that allows file transfers between two USB memory devices without the need for a personal Computer. This report discusses the current implementation of VDIP2 module having VNC1L chip incorporated on it. Module is connectable because of its compatibility with standard 40pin DIP socket.

VNC1L contains host controller, which means, that it can manage all USB communication without any personal computer. To control the function of VNC1L are used commands, which are sent via UART, SPI or FIFO interface from the microcontroller. These commands can be sent like ASCII characters (IPA mode) or like hex code values (IPH mode). There can be chosen Shortened Command Set (SCS) or for more functionality Extended Command Set (ECS).

VNC1L is very powerful chip and it can be used in many different ways. This work shows one small part of the possible usage.

II. HARDWARE DESCRIPTION

The VNC1L chip on VDIP2 module is used as a USB host controller. The detailed description about USB hardware can be found in [1] [2] [3]. It has two USB ports, one of which acts as source and the other acts as a target which can be configured during programming. The USB flash drives are connected to these ports. A monitor or ATMEGA16 Microcontroller can be used to command VDIP2 module. Here, monitor acts as a master and VNC1L as a slave. The commands are sent to the module by the master via communication channel. The mode of communication can be chosen from UART, SPI or FIFO. UART is used here as a mode of communication between monitor and the module. Fig. I shows the block diagram.

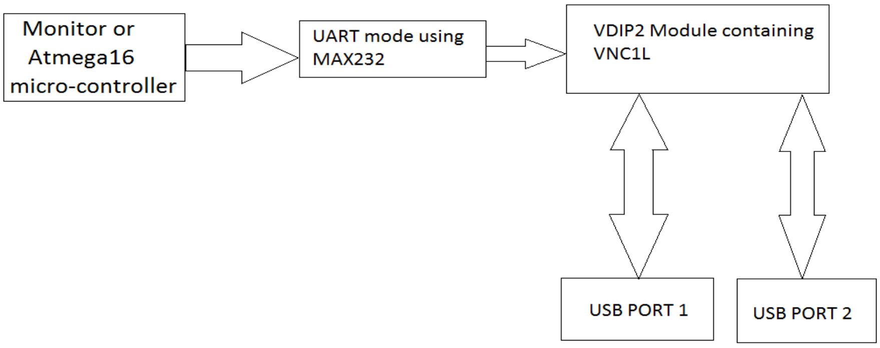

Figure I: Block Diagram

2.1 VDIP2 Module

The VDIP2 module is an MCU to embedded USB host controller development module for the VNC1L IC device. The VDIP2 is supplied on a PCB designed to fit into a 40 pin DIP socket, and provides access to the UART, parallel FIFO, and SPI interface pins on the VNC1L device, via its AD and AC bus pins. All other Vinculum I/O pins are also accessible. Not only it is ideal for developing and rapid prototyping of VNC1L designs, but also an attractive quantity discount structure makes this module suitable for incorporation into low and medium volume finished product designs [4].

Figure II: VDIP2 Module

2.2 VNC1L Chip

The VNC1L is the first of FTDI□s Vinculum family of Embedded USB host controller integrated circuit devices. Vinculum can also encapsulate certain USB device classes handling the USB Host Interface and data transfer functions using the in-built MCU and embedded Flash memory. When interfacing to mass storage devices, such as USB Flash drives, Vinculum transparently handles the FAT File Structure using a simple to implement command set. Vinculum provides a cost effective solution for introducing USB host capability into products that previously did not have the hardware resources to do so. The VNC1L has a Combined Interface which interfaces a controlling application with the Command Monitor. The combined interfaces are UART, Parallel FIFO and SPI. The VNC1L chip features an integrated 8/32-bit MCU and 64k embedded Flash memory. Not only does the chip handle data transfer functions on two USB Host/Client interfaces; it encapsulates several USB device classes as well. One doesn't have to worry about writing firmware to implement those functions. When interfacing to mass storage devices such as USB Flash drives, the VNC1L transparently handles the FAT file structure communicating via UART, SPI or parallel FIFO interfaces via a simple-to-implement command set. The VNC1L device features two USB ports which can be individually configured by firmware as Host or Slave (client) ports. VNC1L brings cost-effective USB Host capability to products that previously did not have the hardware resources available [5].

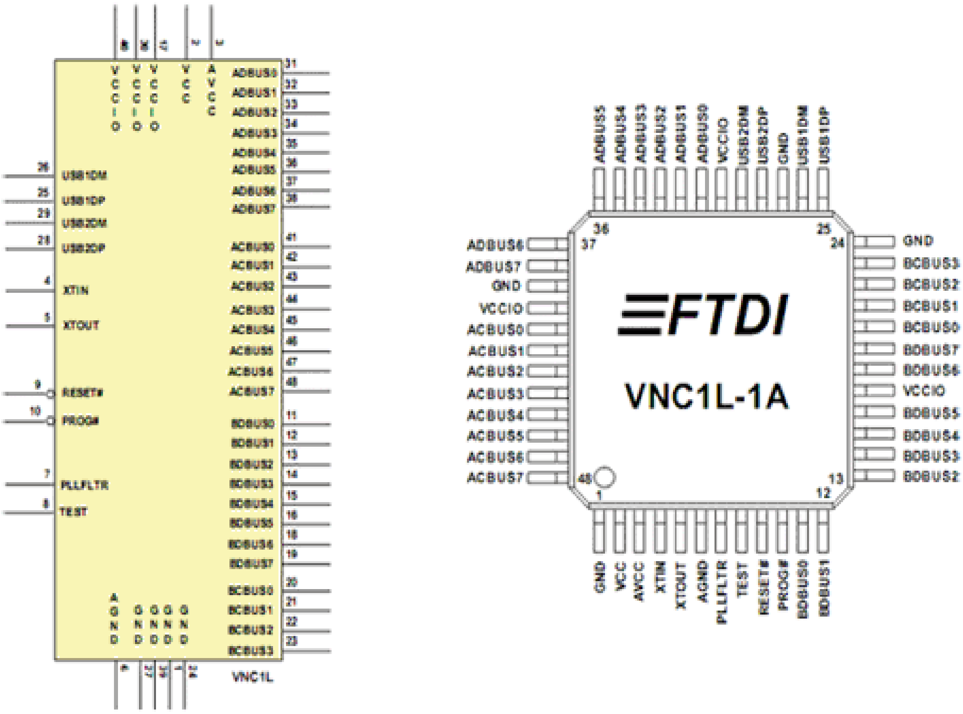

Figure III: VNC1L Chip

2.3 ATMEGA 16

ATmega16 is an 8-bit high performance microcontroller of Atmel's Mega AVR family with low power consumption. Atmega16 is based on enhanced RISC (Reduced Instruction Set Computing) architecture with 131 powerful instructions. Most of the instructions execute in one machine cycle. Atmega16 can work on a maximum frequency of 16MHz. ATmega16 has 16 KB programmable flash memory, static RAM of 1 KB and EEPROM of 512 Bytes. The endurance cycle of flash memory and EEPROM is 10,000 and 100,000, respectively. ATmega16 is a 40 pin microcontroller. There are 32 I/O (input/output) lines which are divided into four 8-bit ports designated as PORTA, PORTB, PORTC and PORTD. It also has various in-built peripherals like USART, ADC, Analog Comparator, SPI, JTAG etc. Each I/O pin has an alternative task related to in-built peripherals. The following table shows the pin description of ATmega16 [6].

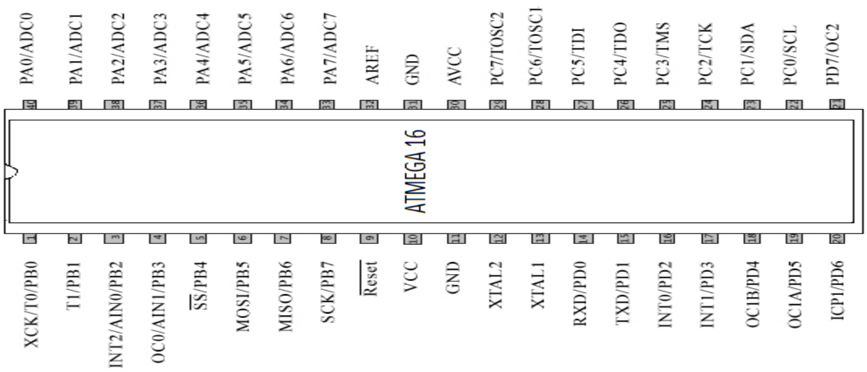

Figure IV: ATMEGA 16 Pin out

III. SCHEMATIC

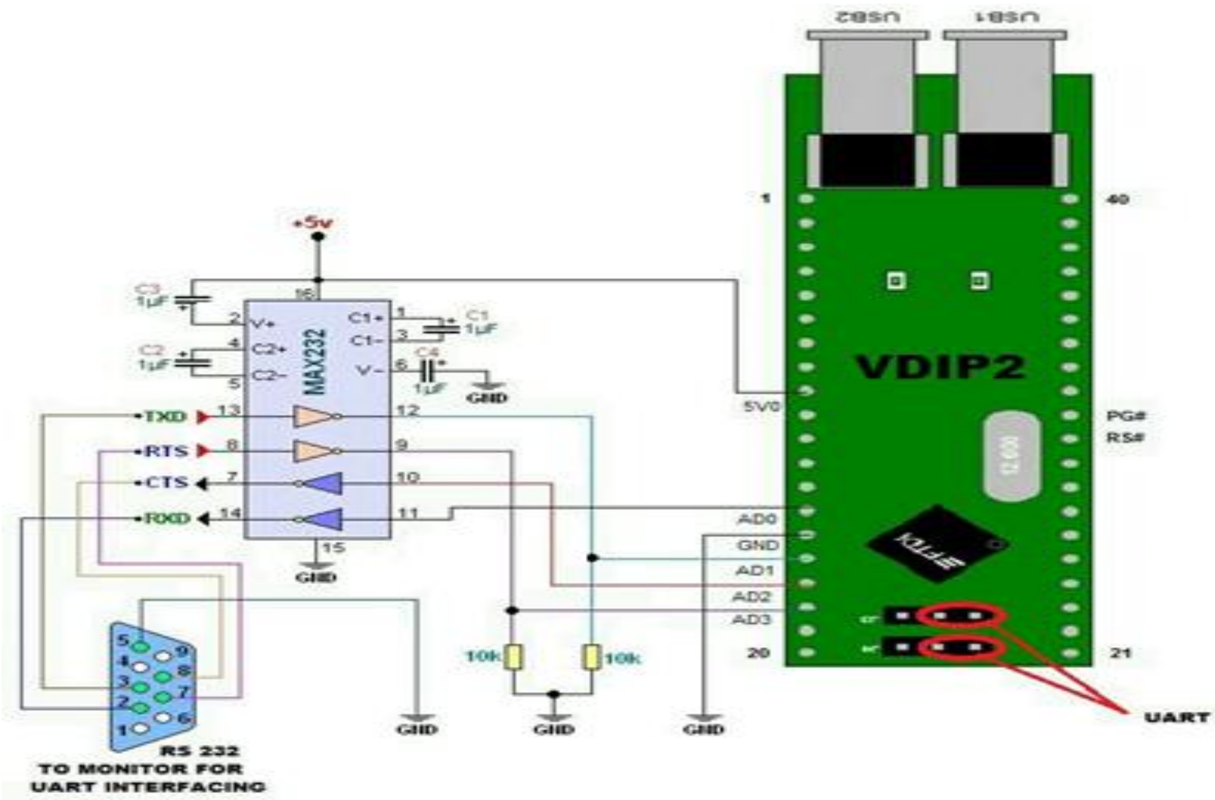

Figure V: Schematic Diagram

Fig. V shows schematic of the project [7] [8]. It is basically the UART connection form the monitor to the host USB controller module VDIP2. The pins which serve as a UART on the VDIP2 are AD3, AD2, AD1 and AD0. Table I shows the pins and their respective functions.

No.	PIN	Function
14	AD0	TXD
16	AD1	RXD
17	AD2	RTS
18	AD3	CTS

Table I: VDIP2 Module Pin Functions

These pins are connected to the RS232 adapter via MAX232 IC [9]. The MAX232 is an IC that converts signals from an RS232 serial port to signals suitable for use in TTL compatible digital logic circuits. The MAX232 is a dual driver/receiver and typically converts the RX, TX, CTS and RTS signals. The VDIP2 module also has two jumpers J3 and J4 which needs to be configured for the use of the 4 signals as a UART. The middle pins of both J3 and J4 are connected to AC5 and AC6 pins of VDIP2 respectively. If we want to use the pins as UART then the jumpers need to be connected to either VCC (5V) or GND. The jumpers are connected to VCC as shown in the above circuit diagram. The 5V supply is provided to the MAX232 IC and VDIP2 module. Ground of all the three VDIP2, MAX232 and RS232 are shorted. The 5V supply for the VDIP2 is connected to pin no. 9 and Ground is connected at pin no. 15. The capacitors C1, C2, C3 and C4 are used with MAX 232 to set the direction of the internal buffers. All the capacitors are of 0.1 uF. The capacitors are connected as shown in table II (a).

Capacitor	Between the pins of MAX232 (+ and -)
C1	1 and 3
C2	4 and 5
C3	2 and 16
C4	15 and 6

Table II(a): Pin connections

Connections with RS232 adapter with MAX232 IC is shown in the table II (b).

MAX232	RS232
7(CTS)	8
8(RTS)	7
13(TXD)	3
14(RXD)	2
15(GND)	5

Table III(b): Pin connections

IV. SOFTWARE DESCRIPTION

4.1 HyperTerminal

HyperTerminal (also known as HyperTerm) is a communications and terminal emulation program HyperTerminal can be used to set up a dial-up connection to another computer through the internal modem using Telnet or to access a bulletin board service (BBS) in another computer. It can also be used to set up a connection for data transfer between two computers (such as your desktop computer and a portable computer) using the serial ports and for serial-port control of external devices or systems such as scientific instruments, robots, or radio communications stations. HyperTerminal can also be used as a troubleshooting tool when setting up and using a modem. You can send commands through HyperTerminal to make sure that your modem is connected properly. We can use HyperTerminal mainly for debugging purpose as it displays all the process messages and error messages on screen [10].

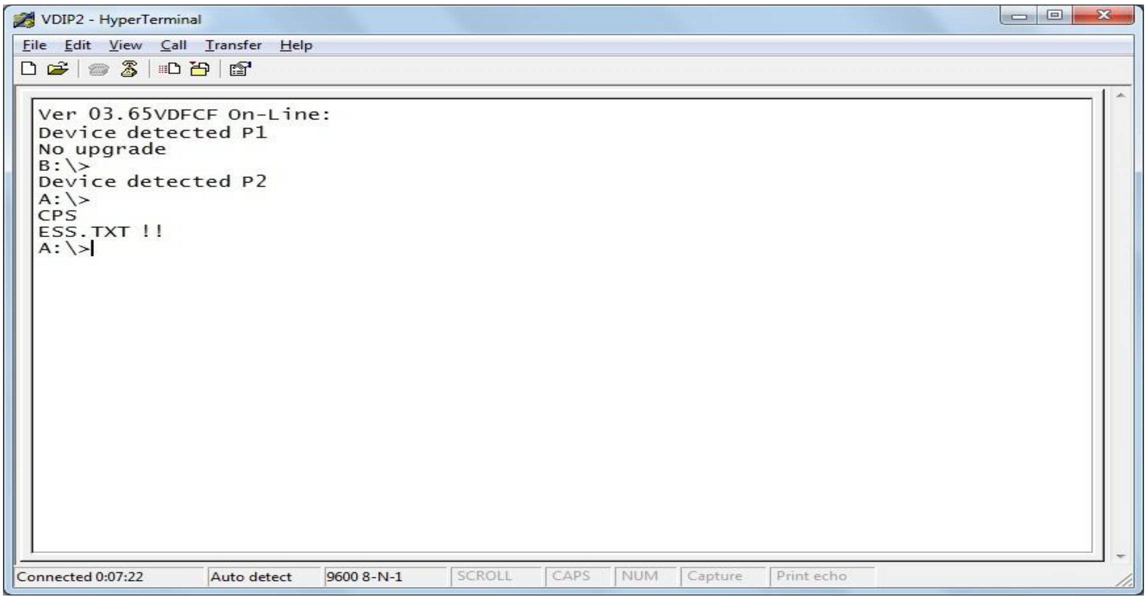

Figure VI: HyperTerminal screen snapshot

4.2 WINAVR AVRDUDE

AVR Downloader Uploader is a program for downloading and uploading the on-chip memories of Atmel’s AVR microcontrollers. It can program the Flash and EEPROM, and where supported by the serial programming protocol, it can program fuse and lock bits. AVRDUDE also supplies a direct instruction mode allowing one to issue any programming instruction to the AVR chip regardless of whether AVRDUDE implements that specific feature of a particular chip. AVRDUDE can be used effectively via the command line to read or write all chip memory types (EEPROM, flash, fuse bits and lock bits, signature bytes) or via an interactive (terminal) mode. Using AVRDUDE from the command line works well for programming the entire memory of the chip from the contents of a file, while interactive mode is useful for exploring memory contents, modifying individual bytes of EEPROM, programming fuse/lock bits, etc. Programmers Notepad is an Open Source editor with some IDE features. It contains the Open Source Scintilla editor component as the basis for its editor. It can be used to write/compile code and generate/burn HEX file. MFile is an automatic makefile generator for AVR GCC written in Tcl/Tk and can run on various platforms including Windows, FreeBSD, Linux, etc. This utility quickly generate a makefile on some simple menu input. MFile for the Windows platform uses the WinAVR Makefile Template for its template.

4.3 VNC1L FIRMWARE

The VNC1L is shipped as a blank device. Initial in-circuit programming (using the downloaded .rom firmware file) can only be done via the UART interface. When upgrading VNC1L in-situ, then the device can be programmed via the UART interface (.rom file). Alternatively, it can be upgraded via a USB Flash disk using a file called “ftrfb.ftd”. Both file types can be downloaded from the FTDI website. Any firmware downloaded from the FTDI website should be changed to match this filename. VNC1L devices can also be programmed before being assembled in a system using the VPROG1 VNC1L standalone programmer.

V. APPLICATIONS

VDAP firmware is the most general purpose and supports most devices on Port 1 and Port 2. The VMSC and VCDC builds are based on the VDAP code and so provide similar functionality. VDPS firmware is required if the VNC1L is to be used as both a host and a slave device. VDIF should only be used in exceptional applications where the monitor must be accessible on USB Port 1. VNC1L can be used to provide USB Host functionality to a microcontroller, which is a typical application for VDAP firmware. This will have the monitor port on the combined interface allowing BOMS devices to be connected to Port 2 and USB Slave Peripherals to Port 1. The VNC1L is used to connect two BOMS devices to the VDIP2 module using the VDFC Firmware. The monitor port allows access to either disk and the facility to copy between the disks. Selected communication class devices can be connected to the Vinculum VNC1L to provide modem-like functionality to microcontrollers. The CDC device must support the sub-class of Abstract Control Model and have compatible bulk endpoints. The CDC firmware requires that call management information can be sent or received over the Data Class interface. When a CDC device is connected to either USB port on the Vinculum VNC1L it is automatically connected in Data Mode to the Monitor interface. There are separate output pins for modem control (RI#, DCD#) used by the VCDC firmware while the connection is active. This device can be used for fast USB to USB file transfer in offices or other places without the help of mediators such as Personal Computer to handle such transfers. Also this device can be made capable to transfer files based on the user requirements.

VI. CONCLUSION

This paper discusses the hardware and software description along with the schematic. The paper puts light on such a device which has great application since it eliminates the use of bulky computer for the purpose of transferring data. The goal of transferring data from one flash drive to another can be successfully achieved by devising a device which works on the principle described in this paper. With the given constraints and assumptions, UART can be selected as a mode of communication between the ATMEGA 16 (monitor) and VDIP2 module (VNC1L). The problems faced now will leave future work on the project with great possibilities. Further research on this paper can lead to the solution for the problems faced.

VII. FUTURE PLANS

VMSC firmware is similar to VDAP firmware but introduces new commands to allow playback through an MP3 decoder chip. Data is sent to a VLSI VS1003 MP3 decoder over an SPI bus and control of playback is performed by a microcontroller. Current track filename is displayed when playback of a track is started. The elapsed time of the current track is displayed every second. VMSC firmware relies on hardware connections to devices found on a VMUSIC1 or VMUSIC2 module and will therefore not function with other modules. The VMUSIC1 and VMUSIC2 modules do not include USB Port 1. This device can hence be used to play songs stored in your Flash drive or iPhone on speakers. Also since it’s a low power device it can work on batteries. Firmware named VDPS can be used to interface the flash drive to various slow devices like printer, scanner or fax machine. It can be of immense use for shops which are specifically opened for getting printouts or for scanning images or for faxing documents.

VIII. ACKNOWLEDGEMENTS

We wish to acknowledge Dr. R. D. Daruwala for his throughout support and guidance in every step from conceptualization to implementation aiding in successful completion of this paper.

REFERENCES

[1] JAN AXELSON, *USB Complete* Penram Publications, 2nd edition, pp. 5-7. Nov 1999.
[2] JAN AXELSON, *USB mass storage*, Lakeview Publications, @2nd edition, chap. 4, pp 57-69. 2000.
[3] USB Implementers Forum Inc., www.usb.org/ interoductionusb-2.0papers
[4] FTDI Inc. http://www.ftdichip.com/Products/Modules/DevelopmentModules.htm
[5] VNC1L Datasheet http://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_VNC1L.pdf
[6] ATMEGA 16 http://www.engineersgarage.com/electronic-components/atmega16-microcontroller
[7] Bhushan Kamble, Amey Priolkar, Ashitosh Kamble and S O Rajankar. Article: *USB to USB Bridge for Computer Independent Data Transfer*. IJCA Proceedings on International Conference on Intuitive Systems and Solutions 2012 ICISS (1):38-42, August 2012. Published by Foundation of Computer Science, New York, USA
[8] Schematics http://www.ftdichip.com/Support/Documents/DataSheets/Modules/DS_VDIP2.pdf
[9] MAX232 http://en.wikipedia.org/wiki/MAX232
[10] HyperTerminal http://www.avitresearch.co.uk/USBTTL/hyperterminal.htm