Buletin Teknik Elektro dan Informatika
(Bulletin of Electrical Engineering and Informatics)
Vol.1, No.4, December 2012, pp. 263~278
ISSN: 2089-3191

Design and Implementation of a Low Cost USB Duplicator

Muhammad Usman Rashid[1], Z. H. Khan[2]
[1]Dept. of Electrical Engineering Center for Emerging Sciences, Engineering & Technology Islamabad, Pakistan
[2]Dept. of Electrical Engineering Center for Emerging Sciences, Engineering & Technology Islamabad, Pakistan
e-mail: usmanrashid@ceset.pk, zkhan@ceset.pk

Abstract

USB devices and flash disks are the most common storage device being used but very little work has been done on it at university project level. Considering the increasing capacity and popularity of USB flash disks, it is believed that in the near future they will replace CDs/DVDs and will be used as a primary storage medium. For this purpose, a USB flash disk duplicator will also be required just as a CD/DVD duplicator is required today. The aim of the present work is to develop a portable and stand-alone device capable of copying the contents of one USB flash disk/mass storage device to another USB flash drive without the use of an intermediary computer.

Keywords: USB, duplicator, PuTTY, LCD

1. Introduction

The USB Duplicator is a device capable of copying media from one USB flash disk to another. USB is one of the most easy-to-use and widely used electronic devices, providing quick connectivity with a variety of equipment and computing devices. USB has different versions; the first was USB 1.0 with a lower bandwidth of 1.5 Mbits/sec, which eventually led to development of version 1.1 which has a full bandwidth of 12 Mbits/sec and finally the current version 2.0 with a high data rate of 480 Mbits/sec. The further enhancement of USB 2.0, known as USB OTG (On-The-Go) provides for the additional capability of inter-transmission of media between two USB OTG devices. It may also be noted that it is faster than USB 2.0. In many parts of the world, USB 1.1 devices are still in common use. If our storage device is not USB OTG, then to copy or duplicate one USB's contents into another USB, a computer is required because USB version 1.1 does not support inter-transmission of media between USBs. USB OTG devices are not very popular. There are many devices which do not have USB OTG function, so for those devices we need a PC to duplicate one USB's data into another. In this world of growing technology, and urge for better electronic devices, it is the need of the hour to duplicate the data of one storage device into the other directly.

The aim of this research is to design a prototype device that will be capable of copying media from one master USB Flash disk on to one slave USB Flash disk without using computer. The complexity of the project is the synchronization between the master USB and slave USB. The cost of the project is expected to be less than the devices which are currently available in the market.

The current research explains the main components of the USB duplicator design which includes the USB host controller IC and a microcontroller. All design steps are taken, which were necessary for the successful implementation of the USB duplicator. A hardware design of the system is systematically constructed after careful analysis of the previous work in this field. The components to be used in the hardware are also identified in the design step. The software is logically explained while programming the appropriate firmware on the host controller Integrated Circuit (IC) chip. Initial testing of the USB host controller IC was done using PuTTY. Finally, it is interfaced with the microcontroller. All the components and a prototype of the design were tested in lab.

Received July 15, 2012; Revised September 24, 2012; Accepted October 13, 2012

This paper starts off with the background information in relevance to our research, detailing what devices are available in the market and what research has been conducted in this field. Key facts about the USB will be explained in Section II, where technical aspects like power management, USB protocols, type of USB packets, device limitations etc., are also discussed. Section III describes the USB duplicator design and implementation. Section IV provides the architecture, explains the technical aspects of this project, where the hardware used is discussed and explained in great detail. In Section V, the experimental setup, where selection of the proper firmware and its programming using appropriate tools is explained. Finally, the testing of the hardware is discussed in the following section VI which also details the end results of the approach undertaken in this research. Future work is described in section VII.

2. Universal Serial Bus (USB)

An electronic hardware which is used to store information is known as a mass storage device [1]. It can store information which can be retrieved as well as edited when required. This information is stored in files. A file system is needed to organize these files in storage media.

There are many types of storage devices like hard drive, CD, DVD. CD's and DVD's are portable storage devices but if they are not stored in proper safety box and if they are not handled properly than the data on them can be lost. Data loss occur due to human mishandling which causes breakage of CD's and DVD's A lot of care should be taken in order to keep data safe. Hard drives can provide a huge storage for files but the only issue with them is that they are not portable [2]. In order to cover these issues Universal Serial Bus (USB) was developed by the union of companies in 1990s. It features a master-slave configuration which requires host (master) and client (slave) [3].

The main characteristics of this technology are the ease of connecting without even opening the chassis of computer. All peripheral can be connected through a direct USB port [4]. Wired USB technologies can be categorized into following main categories:

A. USB 1.0

USB 1.0 has a full speed of 12 Mbits/sec and a low speed of 1.5 Mbits/sec.

B. USB 2.0

USB 2.0 is almost 40 times faster than USB1.0. It can provide full speed of up to 480 Mbits/sec.

C. USB 3.0

It is targeted to be 10 times fast than USB 2.0. However it is under development process.

Due to its high bandwidth, bidirectional communication, Plug and Play (PnP) and low cost, current electronic devices, such as portable storage devices and digital video or audio devices, are commonly equipped with USB interfaces for connection with computers. Flash disks, digital cameras, Personal Digital Assistants, and MP3 players are one of its examples [5]. USB provides connection of a computer with up to 127 peripheral devices. It can be found in over six billion PCs [6].

From the past few decades' technology, whether it is a processor or storage device is continuously in the state of evolution because of the need for faster, portable and more efficient computing devices. Similarly our desire and need for new, fast and portable data transfer methods associated with USB's, plays a major role in the development and research towards USB data duplicator devices.

The work done by Chunping Wang and team in 2007 it is observed that a better method of Universal Serial Bus (USB) data transfer between the USB host and the USB device has been provided [7]. Previous version only has a bulk endpoint buffer which handles the data transfer for all types of data. The problem with the that invention was that the bulk end point buffer was only responsible for sending and receiving data, the buffer requires repeated clearance for storage of transmitted and received data, hence delaying the transmission and degrading the performance of entire system.
Unfortunately the data rate for the device has not been given in their research.

On the other hand, this USB device cannot support bulk-only transmission mode to transfer the data with a host. But it comprises a bulk endpoint buffer, an interrupt endpoint buffer

and a multiplexer [8]. The bulk endpoint buffer stores the data to be transferred through a bulk endpoint of the USB device. The interrupt endpoint buffer stores data transferred between the device and the host through an interrupt endpoint of the USB device. Multiplexer is responsible for switching the path of the data.

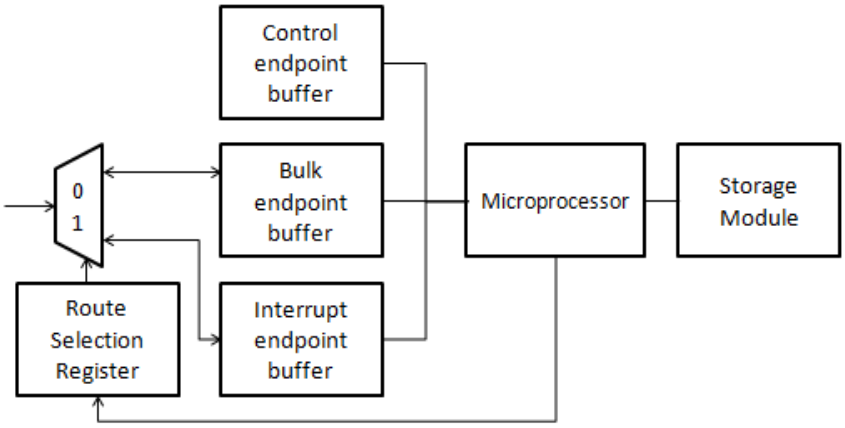

Figure 1. Architecture of USB data transfer method [9].

Data transferred between the host and the device is categorized into three categories including common data, command block wrappers (CBW) and command status Wrappers (CSW) [10]. Common data is transferred through bulk end point buffer and data belonging to CBS and CSW is transferred through interrupt point buffer. A microprocessor controls the operations storage module, the control endpoint buffer, the bulk endpoint buffer and the interrupt endpoint buffer [11].

This invention uses an interrupt endpoint buffer to manage CBW data and CSW data. Thus, the bulk endpoint buffer is not required to frequently clear and move the data stored therein and is dedicated to the transmission of common data. The data transmission bandwidth of the USB device is thus increased, and the performance of the USB device is improved. Though the performance is improved even then the buffers need repeated clearance, due to which optimized speed cannot be achieved. Increasing the buffer size can be one good idea to optimize the speed but this can reduce system performance [12].

The research illustrates the transferring of data from the USB to the processor and receiving data from processor to the USB. It is very useful in understanding the concept of data transfer between the USB host and the device. In case of USB data duplicator this technique can be useful to receive the data from the source USB in to the host controller where it can store it and transmit it to targeted USB. In 2008 M. R. Bohm and A. Ghosh proposed that a single USB device may be shared across multiple USB hosts without needing to be re-configured each and every time the upstream hosts try to access the USB device [13]. The multi-host capable device includes separate buffers for each host, and configures itself to respond to USB requests from more than one host. The device maintains a dedicated address, configuration and response information for each host. Each host can then establish a dedicated USB connection with the sharing device. In order to achieve this they divided the USB device into three segments. The first block comprises a USB interface and other circuitry necessary to send or receive data. The second block consists of an Endpoint Buffer Block, which is used by the first and third block to buffer data. The third block comprises the “Peripheral Function” itself, which has the necessary circuitry for the specific USB device function.

J. Wang with his team member P. Liang invented a system for batched USB data transfer in 2005. In order to achieve an improved system that reduces the interrupts to microprocessor and improves the data throughput on USB for a host system having an embedded USB architecture. The invention comprises a USB host controller system, batch of USB devices, microprocessor, memory associated with microprocessor and memory associated

with the host controller system. Host controller is interface between the Batch of USB devices and the microprocessor.

The host controller unit assembles first batch of USB transactions in a memory. It sequentially execute each of the USB transactions of the first batch in the memory and then send an interrupt signal to a host processor indicating that the first batch of transactions has been completed. The processor then executes a single interrupt and sends the required data to host controller. Host controller stores that data in memory and sends the data sequentially to the USB devices [14].

In another aspect USB host controller is adapted to receive and assemble information indicative of a plurality of USB transactions from a host processor but the USB host controller executes an assembled batch of a plurality of USB transactions upon meeting a certain condition and then the host processor is informed of the availability of the status of the plurality of transactions upon completion of the batch. In this way the host controller reduces the overall resource demand on the host processor during execution of the batch of USB transactions [15].

Though this method reduces the overall resource demand of the microprocessor but it does not help in copying the data from USB to the processor. The research illustrates the transferring of data from the processor to USB host controller system. Host controller then sequentially sends the data to the batch of USB attached. It is very useful in understanding the concept of data transfer between the USB host and the device.

In 2006, two researchers, F. Loo and C. Kuo provided a data duplication method and system used between USB devices so that two or more USB devices can perform inter-transmission of file data without any assistant from a computer [16].

A digital data duplication system is used to control data duplication between a source USB device and at least a target USB device. The duplication system comprises at least a Serial Interface Engine (SIE) circuit, a CPU, a LED or LCD, and a data buffer unit. The CPU is connected to the source USB device and the target USB devices via SIE circuits. The central processing unit controls the SIE circuits to transmit digital data from the source USB device to the target USB devices. The data buffer unit is connected to these SIE circuits and the CPU, and is used to provide memory buffer space required during the digital data duplication process. LED or LCD is connected to show the statues of the device [16].

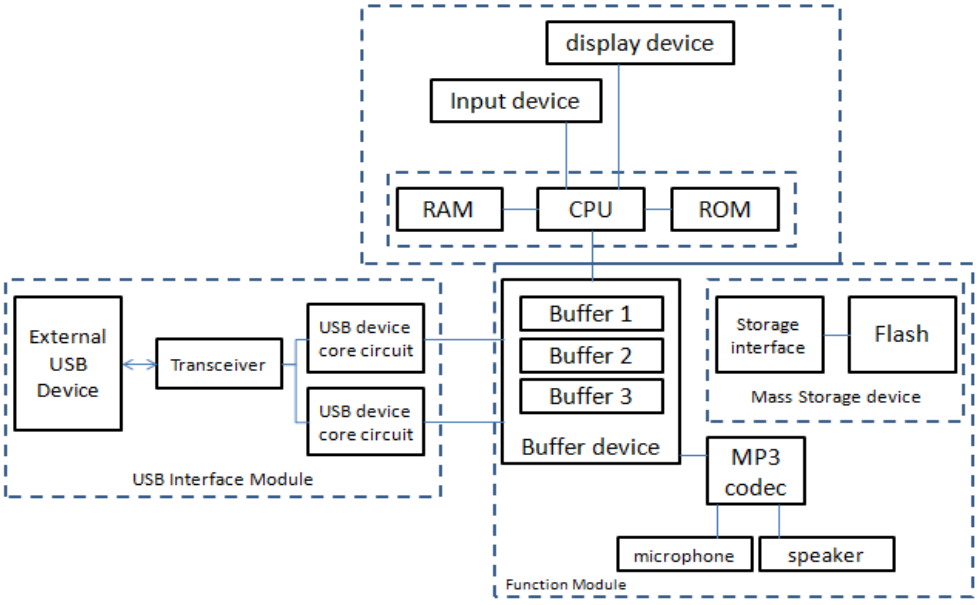

Figure 2. Architecture of Data exchangeable USB devices.

C. Hung with his team members, in 2006, presented an USB data storage device that can directly accesses another USB device to retrieve or store data without being through a host

computer. This invention was also equipped with a MP3 decoder/encoder (codec) to play MP3 music. It can also take an analog voice data through microphone connected to the USB port and store it in a digital voice data form [17]. Another aspect of this invention is that the USB interface module comprises an USB host core circuit and an USB device core circuit. The USB host core circuit is activated as an interface to an USB device if the USB host is at device mode. On the other hand, the USB device core circuit of the USB interface module is activated as an interface to an USB device if the USB device is at host mode [17]. The researchers give a detailed description of data transferring between two USB devices without even use of computer. Serial interface was used for duplicating data of one device to another. This method given in this research is quiet similar as USB duplicator however the hardware to perform these functions is different e.g. memory module for storing the MP3 music and A/D convertor for converting analogue voice data in digital data.

The above mentioned researches are very useful in order to start the USB data duplicator project. It will use the methods which are described above to transfer data between the source device and the target device. USB data duplicator is aimed to provide standalone hardware for the duplication of data between the master USB and the slave USB. It only has one path on which it has to transfer data. So it does not need any specific memory for storing the data path for the data. Furthermore speed and performance of the hardware can be enhanced by attaching a dedicated microcontroller to the USB host which will be responsible for data handling. It will be smaller in size as it will duplicate data between only two USBs.

3. Methodology

A USB duplicator can copy the contents of one USB to the other without the aid of computer. The basic architecture of the USB duplicator is simply based on a microcontroller and a USB controller IC. The microcontroller and USB controller are interfaced with each other. Type “A” USB sockets are attached to the host controller where the USB devices can be attached. So the USB devices will be connected to the USB controller with the help of Type “A” USB connectors. The microcontroller, USB host controller and the USB devices will require a constant voltage in order to be in working order. Microcontroller issues commands to the USB controller which handles the actual transaction between the two flash disks.

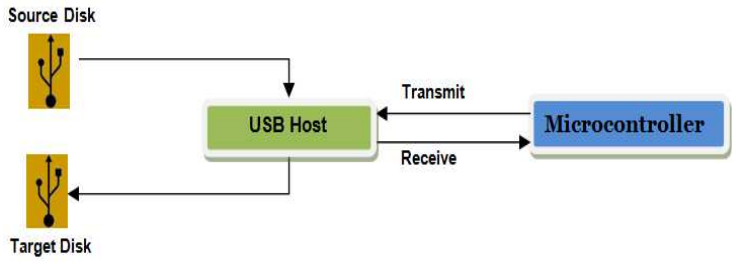

Figure 3. Basic Architecture of USB Duplicator

4. Architecture of USB Duplicator

1) Hardware identification

The basic architecture shows that a microcontroller will be interfaced to a USB host controller which will be further responsible for connecting flash disk with it. Many options were there for choosing the Host controller but Vinculum’s Vdip2 module was chosen as the USB Host Controller.

a) VNC1L I.C

VNC1L is a USB host controller; it supports two independent USB 2.0 host ports. It can be interfaced with other devices via UART, FIFO or SPI interfaces. Entire USB protocol is handled on this chip. There are separate DMA controllers for both USB interfaces [27]. It can not only handle entire USB protocols but the integrated firmware allows reading from and writing to any FAT format USB device.

VNC1L-A is only available in SMT packaging. The chip has extremely small dimensions. Due to these small dimensions it was necessary to use the chip with an SMT to DIP adapter. The PCB design for this adapter required the trace width to be no more than 9 mils, while the CNC machine available could only create traces of 20 mils or more. Therefore, in this project I used the VDIP 2 prototyping module of VNC1L-A [27].

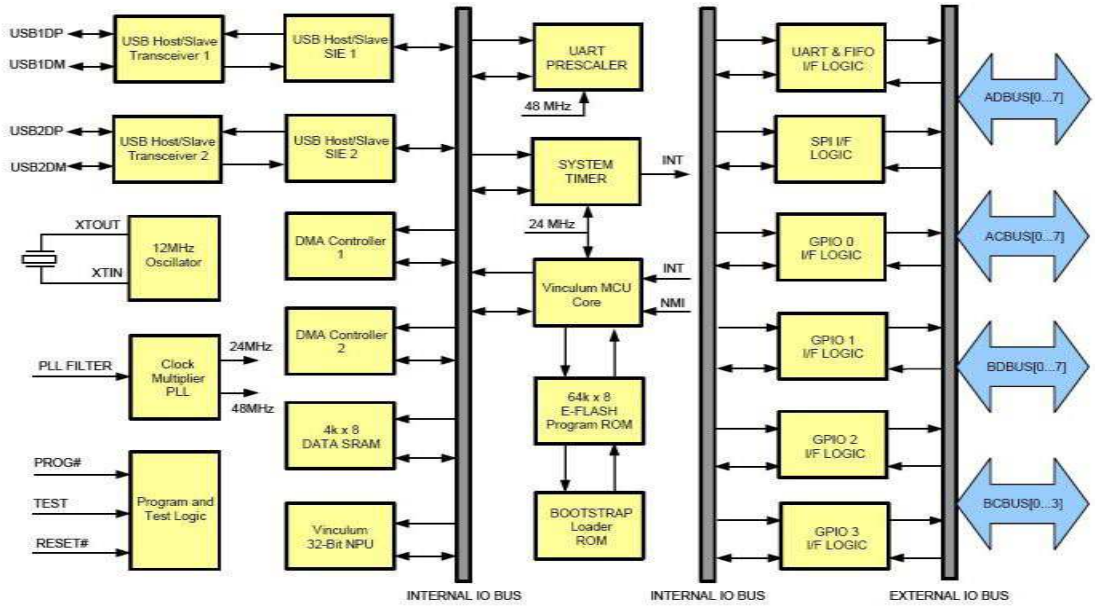

Figure 4. VNC1L Block Diagram.

b) VDIP 2 Module

VDIP 2 module is used to communicate between the microcontroller and the USB devices. It is an embedded USB host controller development module for the VNC1L I.C. device which is attached to two vertically mounted USB Type “A” sockets. USB flash drives will be attached to these sockets. It has 40 pins and is supplied on a PCB design. This module has been chosen for the project because it can provide access to the UART, parallel FIFO, and SPI interface pins on the VNC1L devices [28].

Figure 5. VDIP 2 Module

For UART interface the VNC1L works at 12Mb/sec [29]. The actual throughput does not totally depend upon the VNC1L but also on the device attached to the USB port. Factors such as disk size, format and how full a disk is all impact on the overall speed as the bulk of the processing time is related to handling the FAT table.

The peak and sustained throughputs of the FIFO interface at any given baud rate will be similar to the UART interface at the same baud rate due to the architecture of the VNC1L device [29]. Reading data is faster as the VNC1L can store the data on the chip before passing to the monitor port. The peak and sustained throughputs of the SPI interface at any given baud rate will be lower than the UART interface at the same baud rate due to the architecture of the VNC1L device [29].

c) TTL-232-3V3 Cable

A TTL-232-3V3 cable is needed to program VDFC firmware on the VNC1L I.C. It is incorporated with FTDI's FT232RQ USB to Serial UART interface IC device which handles all the USB signaling and protocols. It provides fast, simple way to connect devices with a logic level serial interface to USB [23].

d) PIC Microcontroller

Any microcontroller which supports UART, FIFO or SPI can be used for this project. The PICDEM development board includes PIC 16f877A with it, so it was used for project. PIC 16f877A can not only support UART but also have a feature of FIFO and SPI interface. It can be programmed to send the commands to the host controller by either using its USART, FIFO or SPI interface.

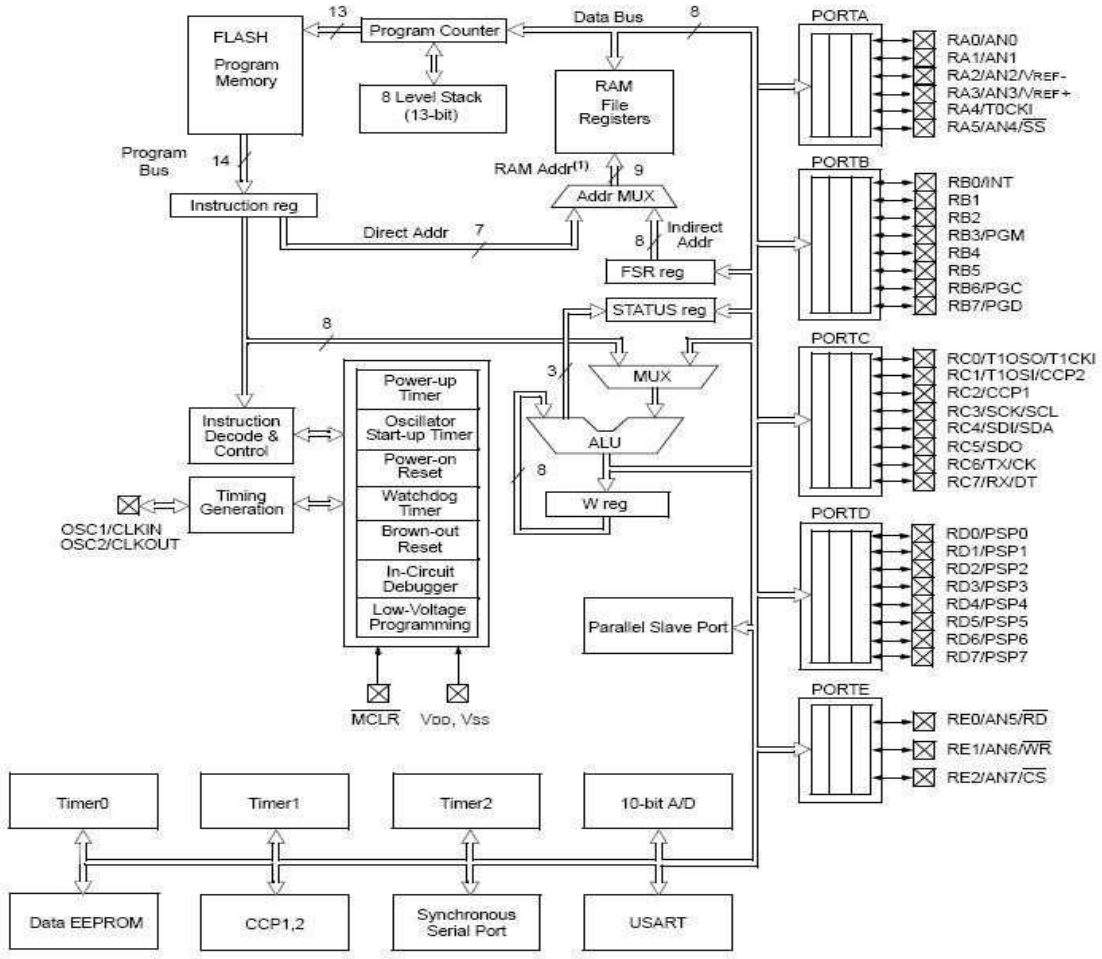

Figure 6. PIC 16f877A Block diagram.

e) PIC development board

There were some PIC development boards already available within the university but all of them had integrated microcontroller on the board. So the development board for this project will be PICDEM 2 plus demo board. It has onboard microcontroller which can be programmed and removed from the board for other purposes. It is required to program the microcontroller. It has an in circuit debugger (e.g. MPLAB) which will also be very useful for testing the microcontroller.

Figure 7. PICDEM 2 PLUS Board.

f) Software identification

Two major softwares were used in the compilation of project are PuTTY and MPLAB, as described below.

g) PuTTY

PuTTY is a free and open source terminal emulator application which can act as a client for the SSH, Telnet, rlogin, and raw TCP computing protocols and as a serial console client. It is used for serial communication of devices. It is a replacement for Hyper Terminal. It will be used for initializing the hardware. After initializing this software will be used in testing the working of the hardware produced.

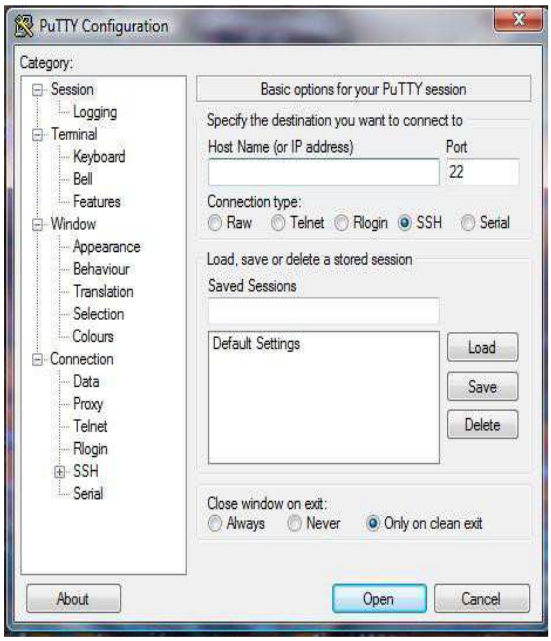

Figure 8. PuTTY Configuration GUI.

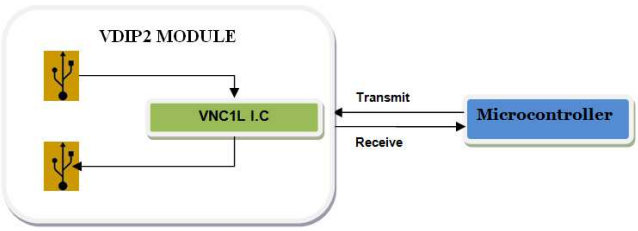

Figure 9. Block Diagram of USB Duplicator

h) MPLAB

It is tool for programming the microcontroller on the development board. It has built in ability to convert the C code to assembly code and generates HEX files for programming the microcontroller.

5. Experimental Setup

As mentioned above that VDIP 2 module has been used as the USB host controller. It has on VNC1L I.C on it; which handles all the USB protocols. The Vinculum VNC1L device has two USB host ports as well as a combined serial UART, SPI or FIFO interface. The combined interface is selectable using a pair of pins connected to either pull-ups or pull-downs.

5.1 VNC1L Firmware

The firmware for the Vinculum VNC1L allows a command Monitor port to be active on either the combined interface or one of the USB Ports. The main function of the Monitor is to allow an embedded device to communicate via the VNC1L's UART, parallel FIFO or SPI interface port with USB peripheral devices [30]. Several classes of device can be connected to this I.C. There is different firmware for each class. Each version of firmware code allows a different combination of devices and monitor ports. There are several pre-compiled firmware codes available online.

VDAP (disk and peripherals).
VDIF (disk and FTDI Interface).
VMSC (music player).
VDPS (disk, PC monitor and slave port).
VCDC (communication class device).
VDFC (disk and file copier).

Table 1. VDFC Command Set

Command	Function
A:	Select Source Disk
B:	Select Target Disk
IMS	Create Snapshot Image of source disk
IMS file	Create Snapshot image of a file of source disk
CPS	Copy source disk to target disk
CPS file	Copy file from source disk to target disk

The VDFC firmware was used in USB duplicator project because it not only allows device interface but also allows file copying. Command set of the firmware can be seen in Table 1.

a) Firmware Setup

The VNC1L boot loader uses the UART interface to load new firmware into the Vinculum Flash memory. This was done using TTL-232-3V3 cable. Pins of TTL cable and VDIP2 module that were used while programming the firmware are shown in table 2 and 3 respectively.

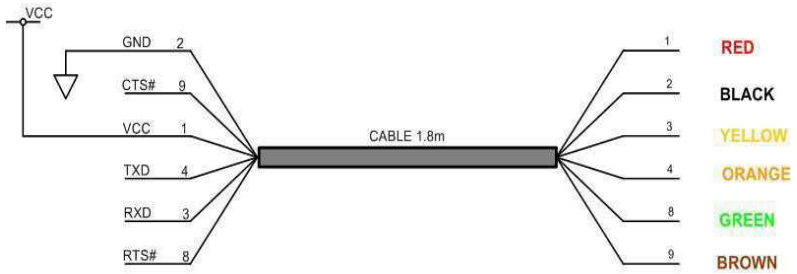

Figure 10. Generic Cable connections of TTl-232-3V3.

Table 2. Pin Signal Description of TTL-232-3V3 Cable.

Color	Name	Type	Description
Black	GND	GND	Device Ground Supply In
Brown	CTS#	Input	Clear to send control input/Handshake signal
Red	VCC	Input/ Output	Power supply output/Input
Orange	TXD	Output	Transmit Asynchronous Data Output
Yellow	RXD	Input	Receive Asynchronous Data Input
Green	RTS#	Output	Request to send control output/handshake signal

b) Firmware Programming

TTL cable was connected to VDIP2 according to the pin description given in table 2 and 3. To enable the boot loader, the PROG# pin of VDIP2must be driven low and VNC1L must then be reset by driving the RESET# pin low then high. Run mode can be enabled by driving the PROG# pin high and then resetting VNC1L by driving the RESET# pin low then high. This was done by toggling one of the output pin of the microcontroller using delay functions. VPROG application was used to program the VNC1L. The COM port is selectedon which the VDIP2 module is connected through TTL cable and the ROM file for the desired firmware is then selected and is programmed on to the VNC1L.

Table 3. Pin Signal Description of VDIP2.

PIN no	Name	Type	Description
12,13, 15,26	GND	GND	Device Ground Supply In
2,3,9	VCC	Input	Power supply output/Input
14	TXD	Output	Transmit Asynchronous Data output
16	RXD	Input	Receive Asynchronous Data Input
17	RTS#	Output	Request to send control output/ HS signal
18	CTS#	Input	Clear to send control input/HS signal
30	RS#	Input	Reset
31	PG#	Input	Used in combination with pin 30

Table 4. Jumper Configurations

Parameter	I/O Mode
Baud Rate	9600
Mode	Asynchronous
Parity	No Parity
Number of data bits	8

c) VDIP2 Setup

As mentioned in section 4.2.1.2 that VINC1L achieves it maximum data rate when UART interface is used because UART mode transfers data using 8 bit data packet where as in SPI mode data is transferred serially bit by bit. So using UART mode 8 bits can be send at a time. Therefore VDIP2 was used in serial UART mode. Its jumpers have to be adjusted in order to use its serial UART.

d) Microcontroller Setup

USART of the PIC16f877A microcontroller was used to communicate with the USB controller. The USART has to be initialized before any communication can take place.

Initialization includes setting of the Baud Rate, number of data bits, parity and enabling the transmitter or receiver depending upon usage.

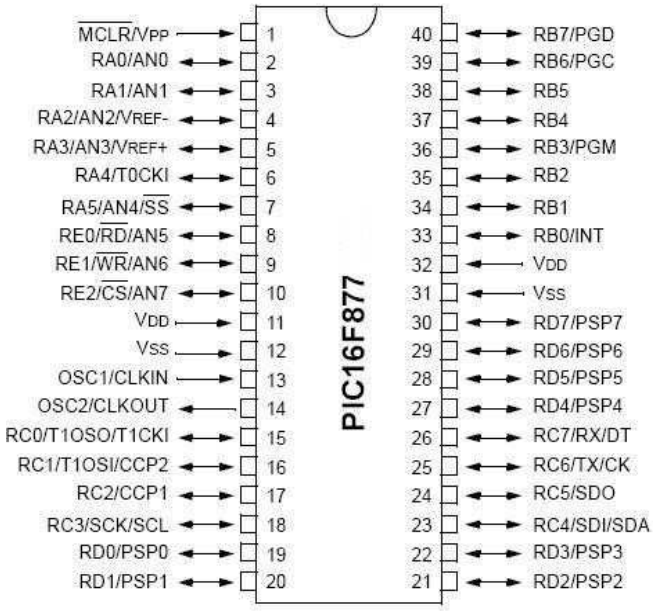

Figure 11. PIC16F877A pin configuration

Table 5. USART Setting of Microcontroller

ACBUS6 (VNC1L PIN 47)	ACBUSS (VNC1L PIN 46)	I/O Mode
Pull down	Pull down	Serial UART
Pull up	Pull down	SPI
Pull down	Pull up	Parallel FIFO
Pull up	Pull up	Serial UART

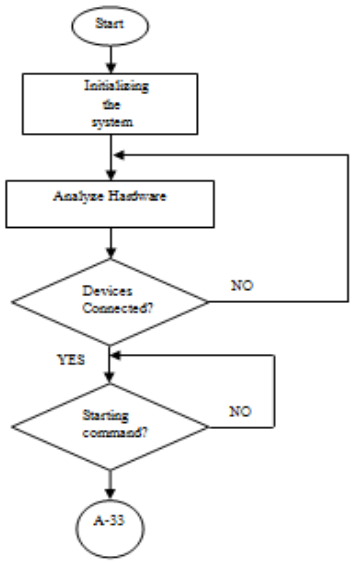

Figure 12. Code flowchart 1

Pin 25 (Port C pin 6) and pin 26 (Port C pin 7) are the USART transmit and receive respectively. The USART was used with the settings as shown in Table 5 below. Working of the code written is explained in the flowchart presented in Figure 12 and 13.

6. Testing

6.1 VDIP2 Test with PuTTY

USB host controller IC was tested separately first, by interfacing it with PC's PuTTY application using the TTL cable. After programming VDFC firmware the firmware was tested by issuing it different commands. The responses from these commands were required for writing the program for the microcontroller.

PuTTY was used with the settings as shown in Table 6 below.

Table 6. *PuTTY Settings*

Parameter	Value
Baud Rate	9600
Mode	Asynchronous
Parity	No Parity
Number of data bits	8
Flow Control	Hardware

Testing the VDIP2 with the hyper terminal has provided a great ease in programming the microcontroller as it gives the response we would receive serially from the VDIP2 whenever an event is going to occur. Figure 31shows the message which is seen on PuTTY whenever it detects a device on the VDIP2 Module.

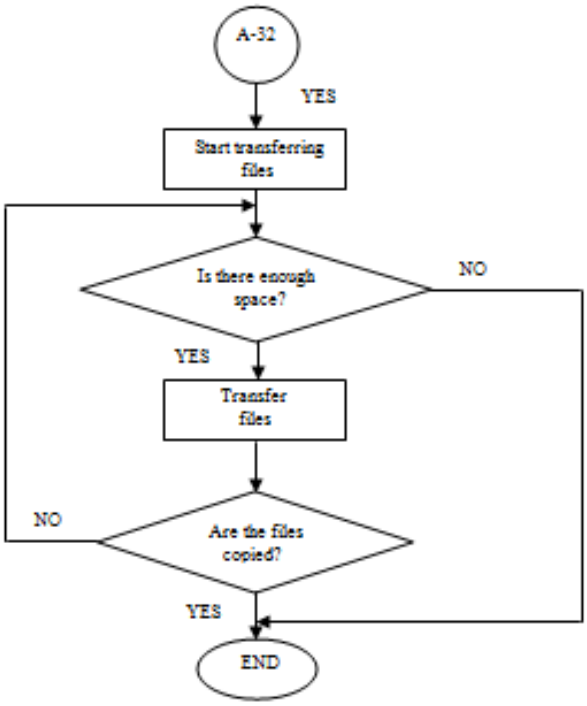

Figure 13. Code flowchart 2

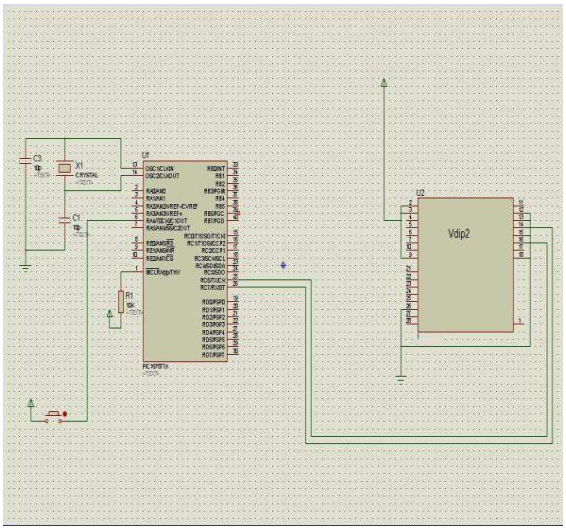

Figure 14. USB Duplicator Schematics

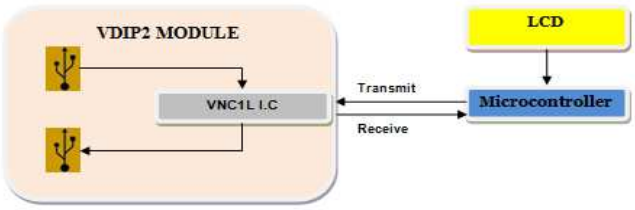

Figure 15. Block Diagram of USB Duplicator

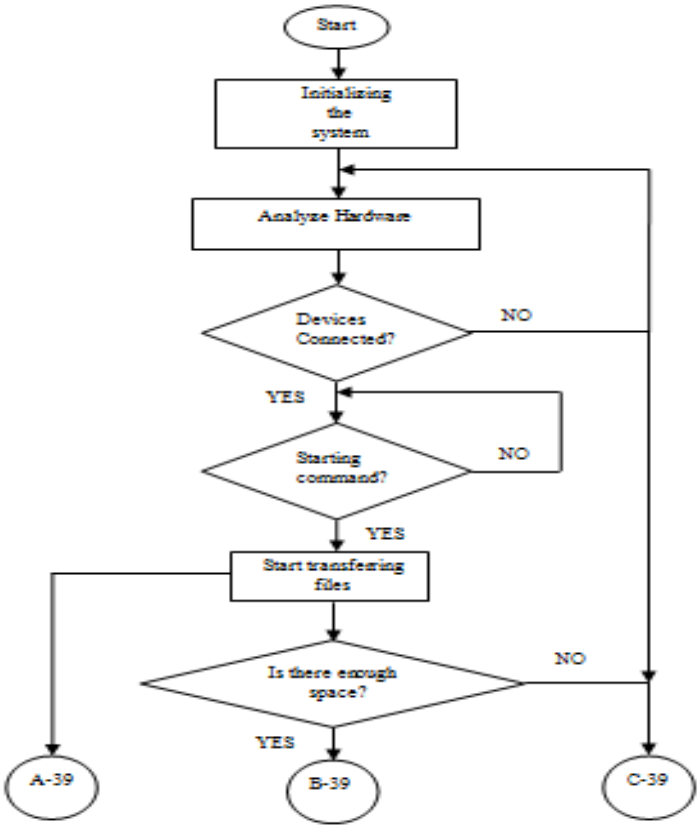

Figure 16. Code flowchart 1 with LCD interfacing

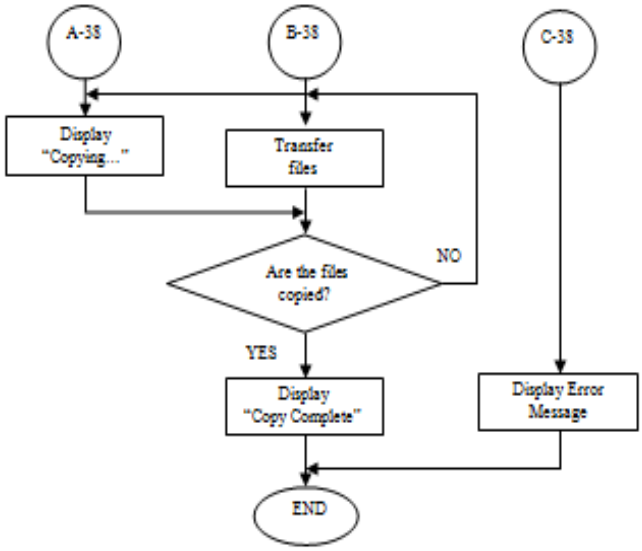

Figure 17. Code flowchart 2 with LCD interfacing

6.2 PIC16f877A Test with PuTTY

To test the serial UART of the PIC16f877A was connected with PuTTY using the TTL-232 USB cable used for serial communication. Sample data was sent to the PuTTY and received from it to make sure that the UART was configured properly. The same PuTTY settings as that for VDIP2 were used in this case.

6.3 Schematics

Finally the microcontroller was interfaced with VDIP2 which is shown in Figure 16.

7. Conclusion

The final hardware is capable of copying contents of one USB Flash drive to other USB flash drive. The project has helped in understanding the serial communication of devices with computer. It also helped in finding the ways to interface and communicate microcontroller with the USB host controller which in this case is VDIP2 Module. Complete understanding of using MPLAB environment was carried out for writing the serial code for microcontroller.

After the initial setup the microcontroller waits for the signal that both the devices has been connected to the VDIP2

Module and are ready for the data transfer. Once the data transfer has been started it cannot be stopped unless or until any error occurs. If the target disk has the same files then they will be overwritten. If the target disk ran out of space while data is being transferred then the data transfer is stopped. The files that have already been copied are not deleted. The data rate using UART mode is 12 Mbits/s. This was mentioned in the datasheet of the VDIP2 Module.

The effective cost for developing this project was around £20. An additional cost was the PICDEM 2 demo board that was used to program the PIC to perform project specific functions, which was £66. This is a very cost effective solution as compared to what is available in the market at the moment. The investment in the PICDEM 2 board is a one off cost, which is able to program many PICs so this greatly reduces the cost of the device itself.

8. Future Work

The research discussed so far is implemented on the bread board and it only checks whether the USBs have been attached to the Module before issuing the data transfer command. If sufficient time was available it would have been given a PCB design and a LCD would have also been attached to the microcontroller for displaying the current status. The block diagram after interfacing the LCD with microcontroller is shown in Figure 16 and 17 respectively.

Interfacing the LCD with the microcontroller can also be used for showing an error message if any error occurs. The Module has already been tested for the type of errors which can occur during data transfer. In case of error VDIP2 module writes the error message on its status register. Microcontroller can be programmed for checking the status register of the module using polling techniques i.e. continuously checking the status register of VDIP2 module. By reading the status register of the Module during file transfer microcontroller can display the current status on the LCD.

Acknowledgment

The authors are grateful to the Embedded Systems Lab, School of Engineering and Digital Arts, University of Kent, UK for providing adequate experimental setup and guidance needed to accomplish this research.

References

[1] J. Axelson, Mass Storage Basic. Lakeview Research LLC, 2006.
[2] J. Axelson. (October 12, 2005, USB Complete (second edition Ed.).
[3] J. Aakash, K. Phillip and L. Benjamin, "On IEEE 1394," vol. 1, pp. April 9, 2010, December 20, 2005, 2005.
[4] P. Polishuk, "1394 Newletter," vol. 2, 1998.
[5] D. Anderson, "USB system architecture (USB 2.0)," Mindshare Inc, Addison-Wesley, 2001.
[6] Intel, "Universal Serial Bus," pp. 1, 2010.
[7] Bi Bo, Sun Shuying and Wang Chunping, "Design of data acquisition equipment based on USB," in Electronic Measurement and Instruments, 2007. ICEMI '07. 8th International Conference on, 2007, pp. 1-866-1-869.
[8] USB Implementors Forum, "Universal Serial Bus Mass Storage Class Specification Overview," vol. 1.5, september 5, 2008, 2008.
[9] C. Tseng and Y. Hsu, "USB data transfer methods," 20070174533A1, July 26, 2007, 2007.
[10] USB Implementors Forum, "USB Serial Bus Mass Storage Device," vol. 1.1, september 31, 1999., 1999.
[11] K. Dan, "USB to multiple connect and support Bays for peripherial devices," 5841424, Mar 3, 1997., 1997.
[12] Y. L. Su-Lan, "Universal serial Bus Connector," 5725395, 10 mar, 1998, 1998.
[13] M. R. Bohm and A. Ghosh, "Multi host USB device," 2009/0106474, April 23, 2009, 2009.
[14] J. Wang and P. Liang, "System and methods for batched USB data transfer," 2005/0033896, February 10, 2005, 2005.
[15] J. Wang and P. Liang, "USB host controller and interface with batched data transfer," 2002/0116565, August 22, 2002, 2002.
[16] C. Kuo and F. Loo, "Data duplication methods and system used between USB devices," 2006/0206631, september 14, 2006, 2006.
[17] C. Hung, K. Lee and S. Juan, "Data exchangeable USB device and method therewith," 20060053238, March 9, 2006, 2006.
[18] Anonymous "Universal Serial Bus Overview," vol. 2010, pp. 5, 2009.
[19] Anonymous (2009, pp. 1. Available: http://www.intel.com/technology/usb/.
[20] M. Steve, "USB 2.0 Tranciever Macrocell Interface (UTMI) specification," vol. 1.05, pp. 1, March 29, 2001, 2001.
[21] Anonymous "Standards and specs," vol. 2010, pp. 10, 26/04/2005, 2005.
[22] A. Meyev, "Roundup: 2.5-inch Hard Disk Drives with 500 GB, 640 GB and 750 GB Storage Capacities," 06/16/2010, 2010.
[23] Cypress Semiconductor Ltd. (2002, October 18, 2002). EZ-USB technical reference manual document # 001-13670 rev. *A. System Soft Corporation; Intel Corporation. San Jose, USA. [Online]. Available: http://www.brian-beattie.com/Projects/fx2/ez_usb_rtechnical_reference_manual__trm14.pdf.
[24] C. Shellagh, L. Dave, H. John S. and M. Steve. (1999, October 27, 1999). USB feature specification Shared endpoints. System Soft Corporation; Intel Corporation. USA. [Online]. Available: http://cscott.net/usb_dev/data/devclass/ccsSharedEPFeature1_0.pdf.
[25] R. A. Quinell, "USB: a neat package with a few loose ends," vol. 2010, pp. 25, 1996.
[26] A. Winkle, "All about Universal Serial Bus," vol. 2010, pp. 15, 20/01/2009, 2009.

[27] Future Technology Devices International Ltd. (2009, May 11, 2009). Vinculum VNC1L embedded USB host controller IC datasheet version 2.01. FTDI Ltd. United Kingdom. [Online]. Available: http://www.vinculum.com/documents/datasheets/ DS_VNC1L.pdf.
[28] Future Technology Devices International Ltd. (2008, March 03, 2008). VDIP2 vinculum VNC1L module datasheet. FTDI Ltd. United Kingdom. http://www.vinculum.com/documents/ datasheets/ DS_VDIP2.pdf.
[29] Future Technology Devices International Ltd., "VNC1L data transfer speeds," FTD1, United Kingdom, Tech. Rep. FT_000099, Feb 16, 2009. 2009.
[30] Future Technology Devices International Ltd., "Vinculum firmware user manual version 2.05," FTDI, united Kingdom, Tech. Rep. FT_000006, Aug 04, 2008. 2008.
[31] Future Technology Devices International Ltd. (2010, April 02, 2010). TTL to USB serial converter generic cables datasheet version 1.1. FTDI Ltd. United Kingdom. [Online]. Available: http://www.ftdichip.com/Support/Documents/DataSheets/Cables /DS_TTL-232RG_CABLES.pdf.