Development of a Novel Android Controlled USB File Transfer Hub

Aaron Don M. Africa, Anna Patricia Nicole B. Mesina, Jullian Lance C. Izon, Bryan Carlo N. Quitevis
Department of Electronics and Communications Engineering,
De La Salle University, 2401 Taft Ave Manila 1004, Philippines
aaron.africa@dlsu.edu.ph

***Abstract*— The Universal Serial Bus (USB) is a mainstay in computer electronics and has been one of the most prominent peripherals in the world because of its portability and accessibility. On the other hand, smartphones have taken the world by storm because of the features they offer and convenience they seek to provide. This study focuses on implementing a USB hub controlled by software installed on an Android phone. Using Bluetooth technology, the phone will communicate to the hub wirelessly to perform file transfer operations. The condition which shows optimum system performance was determined through varying USB flash drives and different file sizes during testing. The tests verified different transfer rates per file for the copy & paste function. The system was also able to overwrite, rename, delete and move files. Based on the test results, the system was able to achieve at least 75% of the speed shown by the two test PCs on certain file size ranges considering the average, best and worst case transfer conditions.**

***Index Terms*— Electronic Design; Electronic Devices; USB Embedded Host; Embedded Systems; USB Communications.**

I. Introduction

Universal Serial Bus (USB) flash drives remain as one of the most popular and widely-used file storage device ever since its invention [1]. Majority of computer users are inclined to use USB flash drives for their file storage needs because of its portability, efficiency, and speed [2]. PCs are typically used to access and perform operations, such as move and delete, on the files in the flash drive [3]. However, a PC or laptop is needed to access the files and perform file transfer between multiple flash drives [4]. Despite the portability of the flash drive, users will have no means to access the files in the flash drive without a computer readily available [5]. Users are also unable to share and transfer files between multiple flash drives [6]. Without a computer to act as a host and facilitate the communication between the USB devices, file transfer cannot occur [7].

Android devices have been an important innovation in modern phone technology not only because of the touch screen feature but also because these devices share some features similar to computers making phones much more powerful than before [8]. One of these features is using Serial Port Profile (SPP) of Bluetooth to send and receive serial data to other Bluetooth enabled devices [9]. This means that the Bluetooth features of phones are not limited to just transferring files like before but it can now also give commands and control other Bluetooth enabled devices [10]. Another powerful feature of android devices is its app development where anyone can be a developer and test their app on android devices [11]. The people can also publish it so that other users can try the application made by the developer on their Android devices [12].

Combining the concepts of Android, Bluetooth, and USB technology, this research focuses on the creation of a USB hub that will enable file transfer and other operations between flash drives and an interface which is an Android application that will control the said USB hub through Bluetooth. The creation of this device will advance USB flash drive movability further by making not only the peripheral devices portable but also the host device. By making a portable host device and a mobile phone application for control, communication between USB flash drives can be moveable as well, making it more convenient.

II. Design

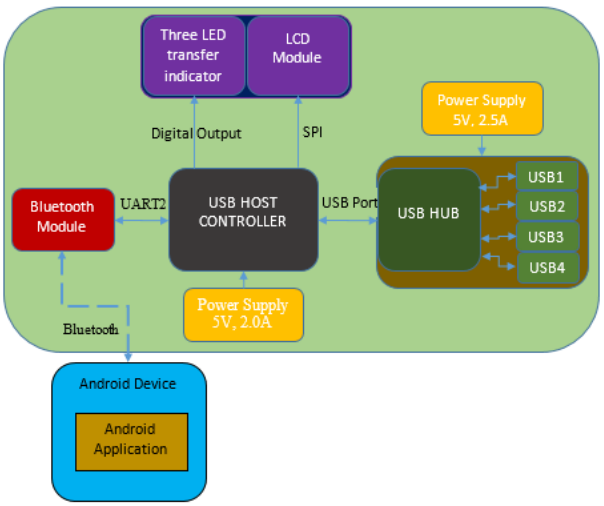

Figure 1: System block diagram

The PIC32 USB Starter Kit II is the USB Host Controller used for the system as shown in figure 1. It integrates all of the modules in the system through its peripheral pins and the breakout board. The USB Hub module can accommodate up to four flash drives. It operates in self-powered mode and is connected to the USB Host Controller module through the USB Type-A connector. Also connected to the USB Host Controller module is the Bluetooth module which receives the commands from the Android device. The LCD module communicates with the Host Controller Module through the PIC16F877A microcontroller connected to the SPI pins in the breakout board of the USB Host Controller Module. The Android Application serves as the Graphical User Interface for the system and relays user input to the USB Host Controller Module.

A. USB Host Controller Module

Since the provided USB host code by Microchip supports only a single device, the group modified the USB Embedded Host code for multiple device support. This layer of code handles enumeration and configuration of a USB Flash Drive upon attaches, therefore it is crucial so the system can handle up to four devices. It identifies the device through its descriptors and decides which client driver it will be initialized with by accessing the Target Peripheral List (TPL). For the system operation, the Target Peripheral List includes the Hub device class and the Mass Storage Device Class.

To support all of these classes, the group modified to code to include a structure array, portdeviceinfo, with the size of 5 to handle a maximum of 5 devices attached to the USB host, namely the USB Hub and 4 Mass Storage Devices. This structure contains information about the device currently being serviced or enumerated by the host so it can switch between devices during system operation. Since some of the functions performed during operation in the host driver require the deviceAddress parameter, the function USB FindDeviceAddress was also created to return the deviceAddress given the DeviceNumber. This allowed the host to set the parameters of a performed function to the portdeviceInfo structure array entry corresponding to the current device being serviced. Furthermore, the group also replaced the variables in the host driver to accommodate the created portdeviceInfo structure to allow enumeration and servicing of multiple devices.

Another modification done was the addition of a USB Host Device Detach function called upon the generation of a Detach Interrupt. This is to inform the application layer that the device has been detached so the application can perform the necessary functions. This function also overrides the state of the port where the detach was detected to a detached state to enable other USB peripherals to connect to the said port.

Microchip's Embedded Host Stack did not include a hub client driver since it was not intended for multiple device support. Along with the modification of the host driver, the group also included a USB Hub Client driver to enable multiple device support for the system. The hub client driver performs the functions to manage the devices attached to the hub. This includes device connect or disconnects detection, power management, bus error detection and recovery, and full-speed or low-speed device support.

Another layer of code that the group had to modify in the Embedded Host Stack is the Mass Storage Device SCSI driver code. The variables in this code were altered to arrays to support 4 mass storage devices. Aside from multiple device support, the group has also created the functions USB Host MSDSCSI MultiSectorRead and USBHostMSDSCSIMultiSectorWrite to enable multiple sectors read/write to enable the host to transfer as many bytes as it can in a single read or write command. The code was modified such that the number of sectors to be transferred in a single read or write command is determined dynamically.

The research group employed the FATFS File System Library by ELM Chan to handle file manipulations such as view directory, copy, paste, and delete. The FATFS module was ported by the group to the USB Embedded Host Stack through its device control interface or diskio code. The diskio code was modified by the group to call the SCSI Layer functions.

The application code implements the desired operation of the designed system. The group created the code to serve as the main application of the program. The first part initializes the different configuration bits to optimize the performance of the PIC32MX MCU. The value of the clock frequency, 80 MHz, was chosen since it is the maximum operating frequency of the PIC32 microcontroller. The group created the event handler, GetBTCommand, to handle the commands received from the Bluetooth module and translate it to functions which have to be performed by the USB Host. The commands that the Android application can perform are called and executed in this function.

B. USB Hub Module

The research group utilized the CY4607 4-Port USB Hub by Cypress to accommodate up to four flash drives. The Hub was chosen since it can support USB 2.0 devices despite being connected to a full-speed host with its backward compatibility. The Hub operates in a self-powered mode using its external regulator to provide ample current for the system. The USB Hub is connected to the USB Host Controller Module through the Type B USB connector in the upstream port to the Type A USB socket in the USB host controller, allowing it to make use of the USB Host Controller's Embedded Host Module.

C. Bluetooth Module

The research group used the EGBT-045MS Shielded Bluetooth module by E-gizmo. A shielded Bluetooth module is used for an easier access to the module's pins and for the built in 5.0 to 3.3V voltage regulator. EGBT-045MS is a class II Bluetooth device which ensures a stable connection ten meters away from the Android device. The Bluetooth module uses Serial Port Profile (SPP) for sending and receiving wireless serial data. Bluetooth V2.1 is implemented in the module where sniff sub-rating is applied to increase battery life and Simple Secure Pairing (SSP) technology is used for the security of serial data transferred and simplicity in pairing to other Bluetooth devices.

D. LCD Module

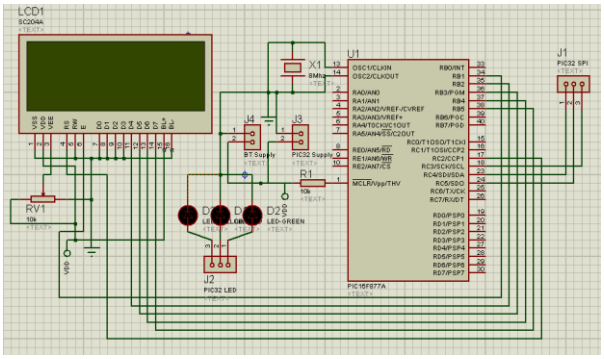

Figure 2: LCD Module schematic diagram

The research group utilized an LCD module which used 4-bit mode to receive data from a PIC16F877A as shown in figure 2. The 4-bit mode let the PIC16 to transmit data twice where the first batch of data contains the four higher bits and the second batch contains the four lower bits. This mode is slower than its counterpart 8-bit mode but the group used this mode to save more pins of PIC16. The variable resistor connected to the VEE pin of LCD adjust the contrast of the characters displayed.

E. Android Application

The Android application is made by using Android Studio. The application serves as the GUI of the system. It will send and receive commands to and from the Bluetooth module via Serial Bluetooth communication. Each command transmits single characters to minimize the transfer time and increase the reliability of the wireless serial connection. Furthermore, the rename command can only accommodate a maximum of eight characters to be sent to the Bluetooth module. The application will automatically connect to the Bluetooth module of the prototype at the start of the program given that the Bluetooth module is already paired to the Android device.

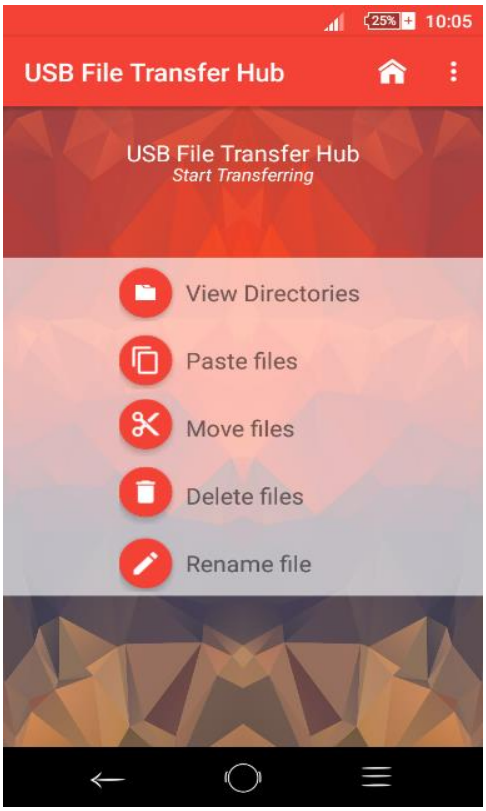

Figure 3: Android application home screen

The group used fragments to separate the command screen from the USB selection and File Selection screen to have a more user-friendly interface as shown in Figure 3.

III. DATA AND RESULTS

To assess how the prototype performs in comparison with a PC and laptop, transfer speeds under the average, best and worst conditions are compared with the average data rate observed in the PC when performing the same transfer under the same parameters. Two PCs were used to compare the speed obtained in the prototype. These test PCs are the Dell Latitude 410 with a 1.73 GHz Pentium M processor and the Acer Aspire V5 with an Intel i5 2.5 GHz processor.

Figure 4 shows that the prototype was able to attain an average transfer speed of 75% of the Pentium M transfer rate for Single-to-Single file transfer with a file size of up to 120KB. The prototype was also able to attain 75% average transfer speed of the PC with i5 processor for Single-to-Single file transfer with file size up to 50KB. However, the prototype speed falls-off when compared to PCs when larger file sizes are transferred.

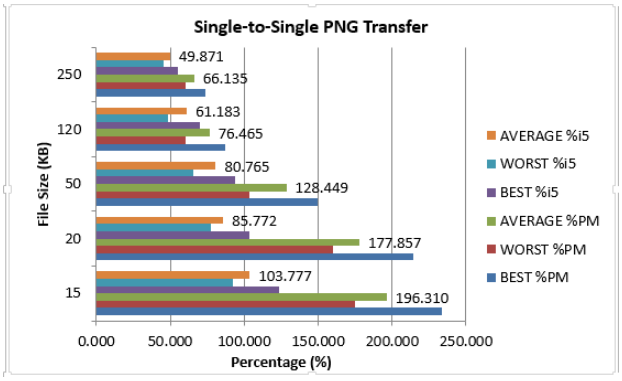

Figure 4: Single-to-Single PNG Transfer Prototype vs. Pentium M and i5 PC Comparison

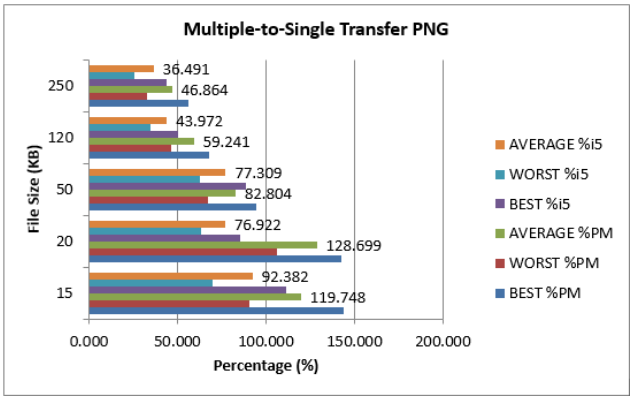

Figure 5: Multiple-to-Single PNG Transfer Prototype vs. Pentium M and i5 PC Comparison

Figure 5 displays that the prototype was able to attain an average transfer speed of 75% of the Pentium M and i5 transfer rate for Multiple-to-Single file transfer with file sizes within 50KB.

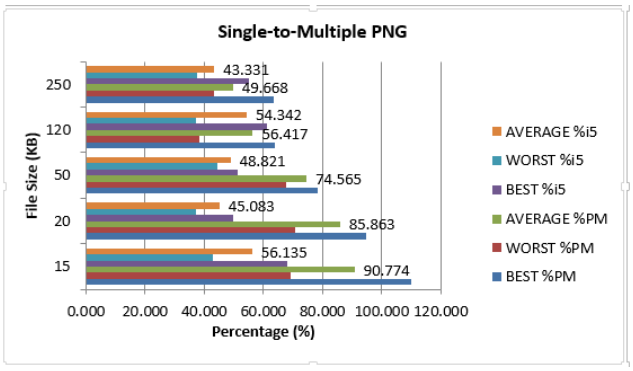

Figure 6: Single-to-Multiple PNG Transfer Prototype vs. Pentium M and i5 PC Comparison

For single-to-multiple point transfers, the efficiency of the prototype can be observed up to the 50 KB size range as shown in figure 6. The best transfer conditions exhibited transfer speed greater than 75% of the speed in the Pentium M PC. Compared to the i5 PC, however, the prototype only reached the target percentage in transferring the 20KB PDF file. Nevertheless, the ideal performance of the system in relation to the performance of the Pentium M PC was observed when transferring files with sizes less than 20 KB.

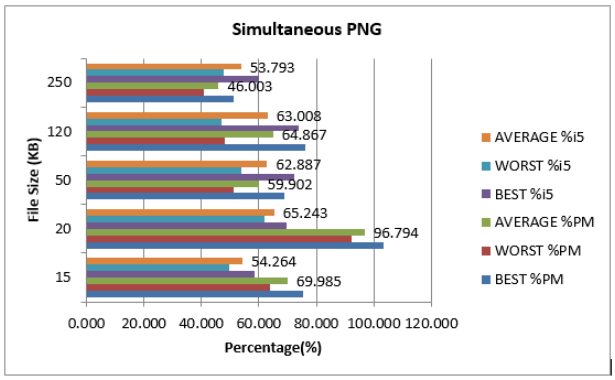

Figure 7: Simultaneous Point-to-Point PNG Transfer Prototype vs. Pentium M and i5 PC Comparison

For the simultaneous transfers, a similar pattern can be observed in the comparison of the prototype speed with both PCs for all the file types as shown in figure 7. There is an observed decrease in the percentage of similarity as the file size increases. The efficiency of the system compared to the Pentium M PC can be observed in the small file size ranges, particularly in the 15 KB and 20 KB ranges.

Table 1
Duration of a Rename Operation

SIZE (KB)	SOURCE Sandisk Time (s)	Transcend Time (s)	Kingston Time (s)	Kingmax Time (s)
15.542	0.300	0.287	0.294	0.309
20.390	0.303	0.288	0.291	0.313
50.916	0.305	0.285	0.290	0.312
122.469	0.306	0.288	0.292	0.309
256.13	0.302	0.284	0.293	0.310

The prototype is also able to do a file rename which lets the user choose the file they wish to rename and enter the desired new name. Table 1 shows the duration of each rename operation done in the prototype for the PNG files. It is seen that the time it takes to perform a rename is the same for all the file sizes.

The prototype also allows the user to delete the selected files in a mass storage device. Table 2 shows the time it takes to execute one delete operation in the prototype for a PNG file. It is seen that the delete time for a single file is the same throughout, while the time for the batch delete is longer since it has to do the delete function repeatedly for all the files in the batch.

Table 2
Duration of a Delete Operation

SIZE (KB)	SOURCE Sandisk Time (s)	Transcend Time (s)	Kingston Time (s)	Kingmax Time (s)
15.542	0.093	0.095	0.095	0.114
20.390	0.095	0.094	0.094	0.114
50.916	0.094	0.095	0.094	0.114
122.469	0.093	0.093	0.093	0.116
256.13	0.094	0.094	0.095	0.116
245.86	0.182	0.183	0.179	0.221

The prototype also allows the user to perform a file move between flash drives. Table 3 shows the transfer speed when a move operation is performed with a PNG file from one source drive to three destination drives or a single-to-multiple point transfer. It is observed that a move operation takes more time to complete than a paste operation since the prototype has to delete the file in the source drive after pasting it in the destination drive.

Table 3
Duration of a Move Operation for a Single-to-Multiple Point Transfer

SIZE (KB)	SOURCE Sandisk Time (s)	Transcend Time (s)	Kingston Time (s)	Kingmax Time (s)
15.542	0.340	1.161	0.327	0.317
20.390	0.376	0.453	0.423	0.327
50.916	0.705	0.664	1.466	0.609
122.469	2.431	1.673	2.327	1.340
256.13	2.982	3.737	2.989	2.591

IV. Conclusion and Recommendations

The research group was able to construct the working prototype and enabled it to function similar to a traditional PC interfaced USB Hub. The research group has demonstrated full functionality of the hub alongside the Android phone coupled with the Bluetooth interface. The group was able to demonstrate file transfer operations from the system namely: copy & paste, rename, move, and delete, and develop an Android application that served as the user interface. File transfers could also be performed from a single source to multiple destinations, multiple sources to a single destination, single source to multiple destinations, and simultaneous transfers. The transfer rates attained by the prototype did not fluctuate and remained consistent throughout the trials and tests performed. The type of file being transferred has minimal effect on the transfer rate since the same pattern was observed for different file types of the same size. The group was able to observe which transfer conditions would bring the system to its optimum performance.

The PIC32MX795F512L used by the group is only limited to USB1.1 compliant speed. The group suggests utilizing a superior hardware which is commercially available. The research group recommends for future research to use USB2.0 and USB3.0 compliant microcontrollers particularly from manufacturers with devices dedicated to USB projects such as Cypress and Microchip. In addition, using these later specifications will lead to further development to the usage of Human Interface Devices such as keyboards, printers and webcams.

References

[1] D. Pham, A. Syed, and M. Halgamuge, "Universal serial bus based software attacks and protection solutions," Digital Investigation, vol. 7, no. 3-4, pp. 172–184, 2011.
[2] J. Clark, S. Leblanc, and S. Knight, "Compromise through USBbased Hardware Trojan Horse device," Future Generation Computer Systems, vol. 27, no. 5, pp. 555–563, 2011.
[3] J. Raj, S. Rahman, and S. Anand, "Microcontroller USB interfacing with MATLAB GUI for low cost medical ultrasound scanners," Engineering Science and Technology, an International Journal, vol. 19, no. 2, pp. 964–969, 2016.
[4] J. Grabara, "The Individuals' Level of Computer and Internet Use in Polish Information Society in Comparison to European Average," Procedia Economics and Finance, vol. 27, no. 1, pp. 718–725, 2015.
[5] C. Sethna, C. Breen, M. Pradhan, C. Green, B. Kaplan,and K. Meyers, "USB Drives for Communication of Medical Information in a Pediatric Dialysis Unit," The Journal of Pediatrics, vol. 155, no. 3, pp. 444–445, 2009.
[6] Chakravorty, and R. Suryawanshi,"Data Transfer between Two USB Flash SCSI Disks using a Touch Screen" International Journal of Engineering Trends and Technology, vol. 13, no. 2, pp. 71-75, 2014.
[7] Android File Transfer. https://www.android.com/filetransfer/. 2016
[8] K. Ono, and H. Ogawai, "Personal Robot Using Android Smartphone" Procedia Technology, vol. 18, no. 1, pp.37-41, 2014.

[9] N. Sriskanthan, F. Tan, and A. Karande, "Bluetooth based home automation system," Microprocessors and Microsystems, vol. 26, no. 6, pp. 281-289, 2002.

[10] J. Sun, D. Zhao, X. Yao and Y. Zhang, "MCU-Controlling Based Bluetooth Data Transferring System," Procedia Engineering, vol. 29, no. 1, pp. 2109–2115, 2012.

[11] Dong, and X. Liu,"Development of Android Application for Language Studies" IERI Procedia, vol. 4, no. 1, pp.8-16, 2013.

[12] A. Tobias, and E. Tobias, "Developing educational iPhone, Android and Windows smartphone cross-platform apps to facilitate understanding of clinical genomics terminology," Applied & Translational Genomics, vol. 6, no. 1, pp.15-17, 2015.