djungo wireless

December 17, 2000
Dr. Andrew Rawicz
School of Engineering Science
Simon Fraser University
Burnaby, BC
V5A 1S6

Re: ENSC 340 Process Report

Dear Dr. Rawicz:

The following package contains the Process Report of our ENSC 340 project Syndeo Wireless Flash Writer. This document outlines what the group went through during the design and implementation of this project. The purpose of our project is to design and implement a wireless file transfer system through the use of the Bluetooth™ technology.

This document details the current state of the project, the deviations from our original plans, and our future plans for the Syndeo Wireless Flash Writer. In addition, we outline the budgetary and time constraints we encountered during the course of the project and explain the inter-personal and technical experience each member gained from working on the project.

Our team consists of four energetic and dedicated fourth-year engineering students – Andrew Morning-Smith, Ryan Lee, Savio Lau and Veljko Jovanovic. If you have any questions or concerns about our project or this document, feel free to contact us by phone or by email at ensc340-vars@sfu.ca

Sincerely,

Savio Lau

Enclosure: Process Report

Syndeo Module Process Report

Proprietary and Confidential

Version 1.0
12/18/00

Table of Contents

List of Tables

List of Figures

Glossary

Table 1: List of acronyms

HSD	Host Side Device (the PC-based Bluetooth device typically responsible for providing files)
iDAC	Internet Digital Audio Cassette (the MP3 player developed by the DAE group during ENSC 340 1999-3)
Kbps	Kilo Bits Per Second
MB	Mega Bytes (~10^6 bytes)
LCD	Liquid Crystal Display
MP3	A compressed music format
PDA	Personal Digital Assistant
PSD	Portable Side Device (the module that uses Bluetooth to support writing to the mobile device's flash)

1 Introduction

For the past thirteen weeks, this project has drawn together four individuals – Andrew Morning-Smith, Ryan Lee, Savio Lau, Veljko Jovanovic to work tirelessly towards the completion of the Syndeo Wireless Flash Writer. This report takes a close look at the thirteen weeks of the project, from the initial proposal of the project to the current state. In addition, this report also documents the personal experiences of each of the four members through the design and development of this challenging project.

2 Overview Of the Project

2.1 Current State of the Device

The Syndeo module enables the owner of a portable device with Flash Memory to do away with all the messy cables needed for synchronization with a desktop or for downloading/uploading files. The wireless file transfer is achieved through Bluetooth technology.

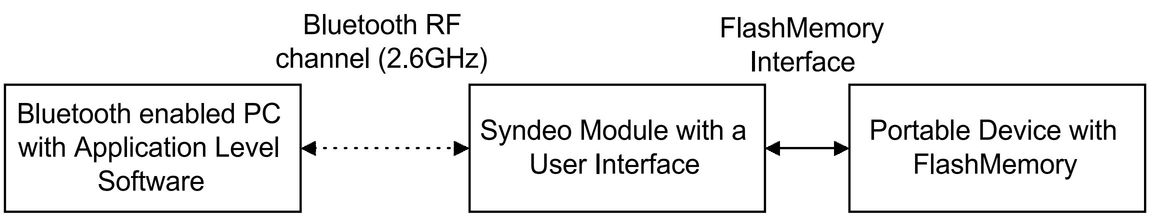

Figure 1: System Overview

2.2 Deviation from the Original Plans

The Syndeo module enables the user to browse files available for download/ upload remotely and it eliminates the need for line of sight, which was a limiting factor for infrared devices. Figure 2 shows our original design for the module.

We originally intended to use an Atmel 8051 series micro-controller to control the LCD, Bluetooth module, and take care of writing to the Flash Memory of the portable device. However, 256 bytes of RAM available on the Atmel were not sufficient to handle some of the Bluetooth packets. Consequently we had to change the micro-controller we are using. After a shorter review of the alternative micro-controllers we selected in our original search we decided to user the Phillips P89C51RB2 micro-controller with 512 bytes of RAM and 16KB of Flash.

With the new micro-controller we had to give up the idea of using our own controller to write to the Flash Memory of the portable device. This is when we realized that in actuality most devices with Flash Memory already have a controller of some sort that writes to the device's Flash. In the case of our proof of concept device with the iDAC mp3 player such a controller already exists and we interfaced with it instead of directly with the Flash Memory. Figure 3 shows the new structure of the Syndeo module. In future iterations of the module an additional interface will be created which will communicate directly with the Flash Memory of the portable device.

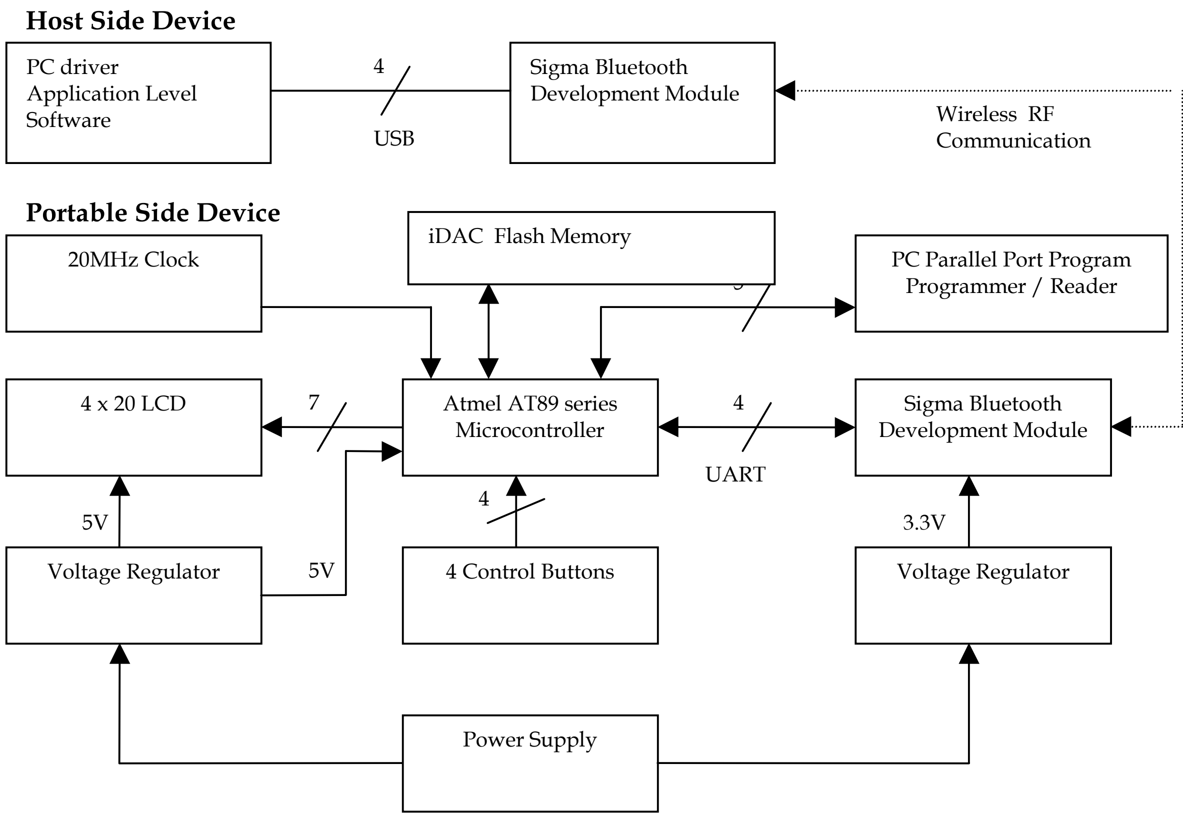

Figure 2: Intended System Design

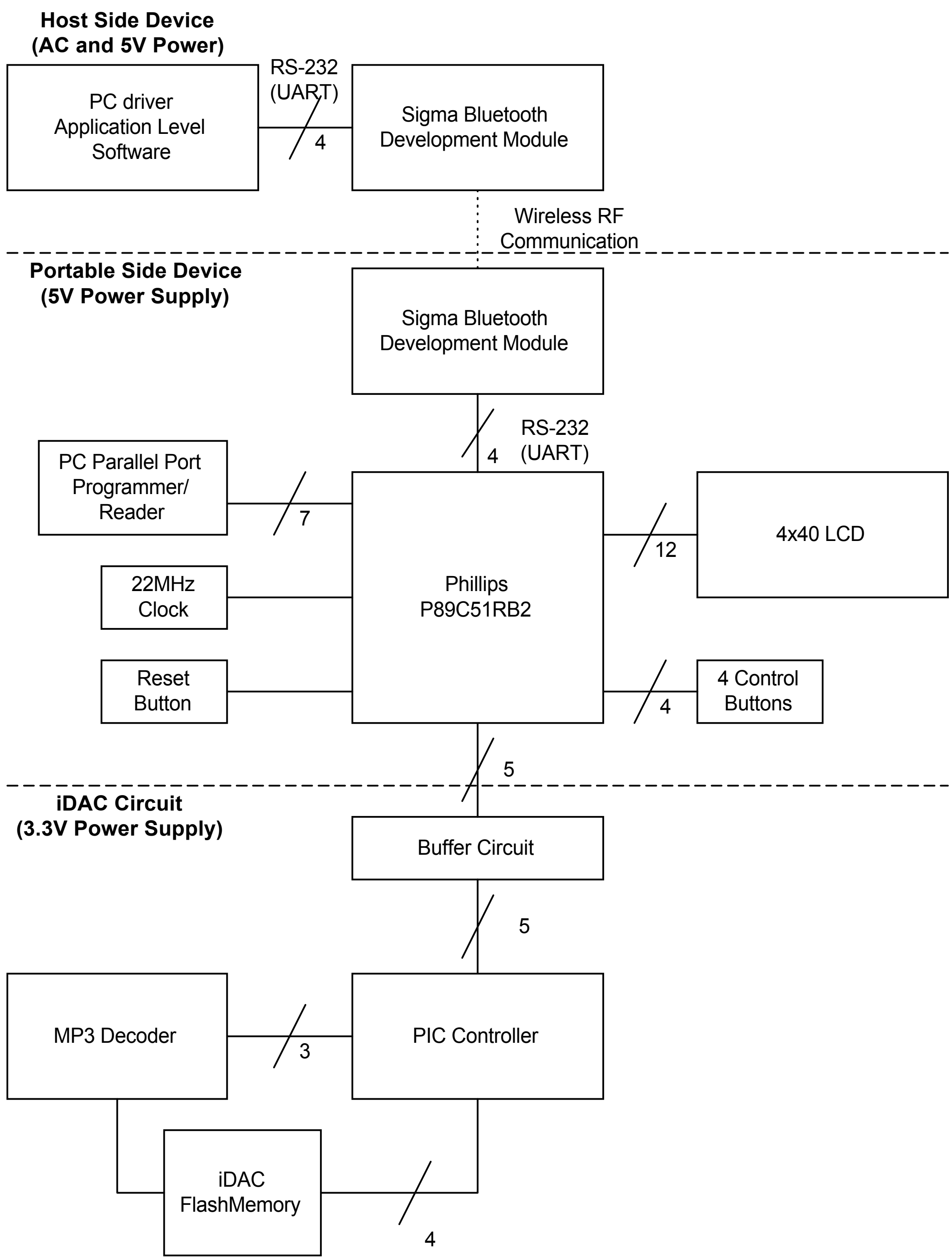

Figure 3: Actual System Implementation

3 Technical Implementation

3.1 Host Side Device

The host side device defines the interface the user will see when they define which songs will be available for download by the portable device. This program is known as the Adjungo Wireless BT File Browser. This program is a single document interface similar to a simple text editor. This program was created with Microsoft Visual C++ because of the maturity and support offered by system under Windows 98.

Displayed on the main screen is a log that keeps track of operations performed by the program. Additionally, the program interfaces to an MS Access database that keeps track of the file, which will be available for download.

The file viewer consists of a list view that shows the songs available for download. The user can add songs, which will increase the number of songs available for download. In addition the user can remove songs from the lists. The MP3 files ID3 tag determines identification of songs. This tag contains information such as artist, song name genre, etc. If a song does not have a valid ID3 tag, the user enters the song attributes. The user can modify the file's description at any time. The only information the portable device will see is the artist and song title.

Bluetooth functionality is taken care of by including a library that handles all interactions with the Bluetooth device. This code was written by Andrew and information pertaining to the implementation of Bluetooth stack is located in section 3.3.

A control program was written so that events generated by the portable device could be handled. The portable device can ask for more songs, select songs or download songs. The selection of songs is handled by the file browser due to memory constraints placed upon system memory on the portable device.

The program is similar to the one outlined in the project specification and has not deviated from expected implementation.

3.2 Portable Side Device

The following describes the implementation used with respect to the core items on the portable side device. In addition, deviations from the original design are also discussed.

3.2.1 Microcontroller

Under the design specification, we selected the Atmel 89 series microcontroller (Intel 8051 compatible) as the preferred choice. However, we deviated from that selection and we ended up using the Phillips 8051 compatible microcontroller instead as we find out that we require more RAM memory than we had originally envisioned (referred to Section 4 – Problems Encountered for more details). In addition, the Flash Writing capability has been transferred to the iDAC's Flash / MP3 decoder controller, the PIC microcontroller. The Phillips microcontroller now acts as an emulated parallel port defined in the iDAC's specification. This deviation was needed as neither the Atmel nor the Phillips have the performance required to handle this task of its own (refer to Section 4). This also acts to completely decouple to wireless transfer from the existing device, something that companies would be more likely to use, as no modifications at all need to be made to their products. Besides the change in microcontroller (which was code compatible) and the addition of a Flash Writing controller, no other deviations from the original design are required. In the final implementation, the microcontroller is responsible for the following things:

a. The LCD – The 8051 compatible microcontroller is still responsible for displaying and updating the user interface when the user or the Bluetooth module generate inputs. This is done by sending data and control information to the Hitachi HD44780 compatible LCD controller that is located at the LCD module at 4-bit mode. (there are 4 data lines + 3 control lines)

b. The Buttons – The microcontroller accepts user input from the 4 buttons and responds by updating the LCD. In addition, the microcontroller also sends messages to the Host PC through the Bluetooth interface, if necessary in situation such as download or requesting song titles.

c. Bluetooth Communication – The microcontroller accepts packets through the Sigma Bluetooth Module and interprets them according to one of eight types. Each of these types specify the type of information that is being sent to simplify the communication complexity between the HSD and the PSD.

The microcontroller code, as defined in the design specification, is written in C programming language to maximize portability and enhance debugging. The microcontroller code is compiled with the free SDCC (Small Device C programming language Compiler). SDCC generates the Intel 8051 compatible binary code required for the Philips microcontroller.

The microcontroller code is separated into the user interface, flash writing section, Bluetooth communication, interrupt handling and idle loop sections:

3.2.1.1 User Interface

The User Interface Code is called whenever there is a need to change the display due to user or Bluetooth Input (after interrupts are generated). So, this code is usually resident in memory but infrequently used. Due to the limited number of input buttons (4), the user interface is implemented with 3 lines to display song names / other information with the 4^{th} line acting as the menu for the push buttons.

3.2.1.2 Flash Writing

The Flash Writing Code is only invoked when the user choose to download files from the HSD. The Flash Writing Code is called by the main loop upon the return from the interrupt handler. It first requests the HSD to send data into one of the two buffers. When the data is available the buffer content is then outputted into Port 0 of the microcontroller to the PIC microcontroller for Flash Writing. Two buffers are used such that at any one time, one buffer is used for reading and the other is used to buffer the incoming data from the Bluetooth module. This code section repeats its operation until the Bluetooth module stop sending data.

3.2.1.3 Bluetooth Communication

This section is invoked when the response to certain user inputs requires communicating with the HSD. Some of such actions are requesting file / song titles, downloading file / songs and selecting songs. This communciation code is used exclusively by the user interface and Flash Writing code sections.

3.2.1.4 Interrupt Handler

The push button Interrupt Handler changes the status of a status indicator. Upon return to the main loop, different functions are called to facilitate the requests (mostly UI functions). For Serial Port /UART interrupts, the interrupt handler is responsible for handling the packet and stores the info into the buffer.

3.2.1.5 Idle Loop

The idle loop contains a loop where the controller will enter a “no operation” state. This state continues until interrupts (from user input or Bluetooth) are generated.

3.2.2 Character LCD

The initial character LCD was a 4x20 LCD loan to us from the TA Jason Rothe. He indicated that he had been unable to get it to work. For almost a week we had tried to get the LCD to run but only the operation with the backlight was successful. Later, through some help from Fred Heap, we were able to determine that the LCD was no longer functional. So, in other words, we had been working on a dead LCD for almost a week. Within a few days, we were able to borrow another LCD from a group that is not using theirs. Fortunately, the 4x40 LCD that we borrowed uses the same Hitachi HD44780 compatible controller as the 4x20 LCD so all that was required to make the new LCD work was slight modification to the code. If the original LCD had worked, we would not have deviated from our original design – as 4x20 characters are already more than adequate for our purposes of displaying song title and menu.

3.2.3 Voltage Regulators

From the documentations that we were given, the Bluetooth chip, being one of the "new" portable devices, was set to run at the power saving 3.3V. Later, we found out that although the Bluetooth does indeed run at 3.3V, the Sigma module, for USB compatibility reasons, runs at 5V with internal step-down regulators to supply the Bluetooth chip with 3.3Vs. Besides this, there was no deviation from our plans in terms of supplying power.

3.3 Bluetooth Stack

The Bluetooth Stack is used by both the HSD and the PSD, so it does not really belong to either. Initially, we had hoped that the Sigma Bluetooth Module would include code that we could reuse for the PC and the PSD. However, the stack code was precompiled to use the USB port instead of the UART that we wanted to use. We had no choice but to write our own, because of

1) We were using the UART, so the USB compiled code does not help us in this case.

2) The provided code could not be ported to the microcontroller.

3) Even if the code had been portable, it required an enormous 169kB of program space, which is more than 13x the amount of Flash memory we have in the microcontroller.

So with regards to the Bluetooth stack, we had to divert manpower and resources to writing it. This is a daunting task because Bluetooth, as a new standard, sometimes contain vague documentation. In addition, the total Bluetooth specification is over 1000 pages long, which made it very difficult for a small group of four to tackle.

A great deal of thought was put in to designing the Bluetooth stack so that it would be efficient to use it for both the PC and portable devices. The Bluetooth code was broken down into a low level UART layer that could be rearranged to suit either the host or portable's different hardware configurations. Middle layers were broken down into their respective functionalities, such as a class for handling events, a class for handling data transmission and a class for handling commands. On top of all this sits an easy to use interface for doing general purpose operations (initialization, finding other devices, connecting to other devices and sending various types of data). This layer made using the Bluetooth stack easier for the other programmers.

In terms of deviation, we put massive effort into this area. Even after writing three thousand lines of code, we were only able to get minimal functionality such that we get basic transmit, receive and handshaking functions.

4 Problems Encountered

Throughout the course of this project, we have encountered a number of issues. Most of these are very daunting and at times brought our project to its knees. In Retrospect, there are a number of things we can and may have changed but given that this was our first design project, there are many issues that are learned through trail and error. Our technical, financial, scheduling and group dynamics issues are outlined below:

4.1 Technical Issues

4.1.1 Lack of Ram Memory

When choosing our microcontroller we chose it primarily based on the amount of non-volatile flash memory that it had, knowing that the implementation of the Bluetooth stack would require a great deal of code space. However, the choice of microcontrollers that offered enough flash was limited and we ended up picking one that had 256 bytes of RAM, which we thought would be enough. We didn't realize at that time (not knowing much about Bluetooth) that some of the packets that we would have to send and receive would be quite large and would have to be stored in our RAM. We also didn't realize that nearly a quarter of the RAM that they quoted to us as having is allocated for special purposes (i.e. Control of timers, ports, etc.) and could not be used by our code. Our code was optimized to be as memory efficient as possible but in the end we needed to find a new microcontroller with 512 bytes of RAM.

4.1.2 Speed of Processor

The speed of the processor was also something that we overlooked. We chose the Atmel 8051 series processor that ran up to 24 MHz, which seemed sufficient in the beginning, however, we soon realized that each instruction took 12 clock cycles to execute which made the execution less than impressive. This caused a great deal of trouble, especially in dealing with interrupts that were occurring to frequently and stealing all the CPU's time. We were eventually able to optimize some of this code to get reasonable performance.

4.1.3 Malfunctioning LCD

We spent several days of research and coding trying to figure out how to write to an LCD and couldn't even get a character to display properly. In the end we discovered that the LCD was not operational (after asking people with more experience with LCDs, they determined that since none of the crystals went black after power-up there was something wrong. Fortunately, we were able to obtain another LCD from another group that was controller in a similar way so we could reuse our code.

4.1.4 Familiarity with Visual C++

Our team had limited experience working with Microsoft's Visual C++ so it took some time to figure out how to effectively prepare a User interface for the PC side of our apparatus. In addition we had to interface this with a Access database, something we had never done before. We also had to interface this user interface with our Bluetooth stack, which required investigation into how to use multiple threads (so that we could run this as a real-time application) which was totally new to us under Visual C++.

4.1.5 Portability of Purchased Bluetooth Code

We had originally hoped to write the Bluetooth code based on the stack that was provided to us with the Bluetooth modules that we purchased. When the product arrived, we discovered that the stack that was provided only contained pre-compiled code to be used with USB. Since we could not compile this code for a different processor (and had little control even if we compiled it for the PC) we decided to implement our own Bluetooth stack. This was a very large undertaking, considering the Bluetooth specification is over 1000 pages long and some companies are in business just selling Bluetooth stacks they've created. When implementing this stack we tried to keep it as minimal as possible to ensure that it would fit onto our microcontroller's flash while still keeping it versatile so that we could use it for the PC as well. That being said, it still turned into a very large code development effort.

4.1.6 Unnecessary Work

Our group started working on a 3.3V RS232 adapter (to adapt from the 5V the microcontroller runs at) but when we started looking into it we discovered that it was already part of the Bluetooth module we had purchased. This adapter module will, however, prove useful in future work on Bluetooth devices when we are working with the actual Bluetooth chips instead of these pre-fabricated modules with the adapter built in.

4.2 Financial Issues

4.2.1 Obtaining Bluetooth Modules

When we first investigated Bluetooth technology we read that it was supposed to be a very affordable technology that could replace cables for prices coming down to $5 per chip. Although we didn't expect to get chips for development this cheaply we did expect to get them for less than $500 US. We had also planned on buying one module to use with the PC and then buying an individual Bluetooth chip to use on our board. However, in order to purchase individual chips we found out that we would have a minimum 12-week delay – far too long for this project. In the end we purchased 2 Bluetooth modules, which allowed us to achieve what we wanted, at a student price of $500 US. This was more than we wanted to spend but was by far cheaper than any other development kit on the market.

4.2.2 Failures to Meet Proposal Deadline

Due to failures in all of our group members' memories, our group missed the application deadline for the ESSEF. We had originally hoped to obtain a few hundred dollars from this fund but due to the missed application we did not get any.

4.2.3 Failure to Attract Investment

Although we felt confident in our application to the Wighton Fund and meeting its requirements, we did not receive any notification of funding for our project.

4.3 Scheduling Issues

Our group encountered many scheduling issues. The main cause of scheduling conflict was commitment to other classes. Even though it is suggested by the school that our group's members should all take the same courses, some member deviated from the recommended course scheduling. This meant that certain members would have assignments to complete while others were available to work on the project. This was not a major issue until integration of all the individual parts was to take place. Members were unable to attend integration sessions which left other member to decode the logic of the absent members work. Additionally, certain group members had a large commitment to activities outside of the school. This meant that they were not present or unable to attend some important meetings. The largest contributor to poor scheduling was the overload of class work that reduced the available time to work on the project.

4.4 Group Dynamics Issues

In terms of group member selection, we are still happy with the group that we have got so far. However, in retrospect, we could have included more members as we underestimated the size of this project and the daunting task of implementing the bleeding edge Bluetooth technology.

Throughout this semester, the major group dynamics issue that we encountered was that we have no defined group leader. Thus, ideas were passed around with no one having the final say, making the final decisions or directing the group in terms of tasks to be completed and their respective deadlines. So, in some cases, confusion arises with some work being completed by two people and some other work being neglected.

5 Solutions / Alternatives to Our Original Plan

As a group, we knew that from the very beginning that the use of cutting edge technology such as Bluetooth carries a hefty price tag. However, looking at the state of Bluetooth, it may not have been the best idea to pursue a Bluetooth project at this point in time, as the technology is not yet mature enough to support small scale project such as ours. This is indicated by the fact that the earliest commercial Bluetooth products will not be readily available until Q1 2001.

With larger companies, the problem above was not so great, since they have the manpower and the monetary resources at their disposal to make the project work. However, with a project such as this, the limited timeframe and the lack of documentation support proved to be a bit more than we could handle. We ended up having to develop a Bluetooth stack which threw us a great deal off our intended schedule. Thus, if we were to pursue this project again, we probably would have picked another wireless technology that is more mature with better documentation and a smaller price tag.

We also found that we are a bit lacking in terms of previous design expertise. Throughout the course of the project, we found that we had made unnecessary mistakes (like building a voltage control board for the Sigma module when it was not needed and selecting a microcontroller that proved to have less memory and performance than we thought it had). To solve this problem, we could have included extra members who are more familiar with designing and selecting hardware or to ask for professional help from others such as Fred, Gary or Patrick.

In addition, we found that we were constantly lacking in time this semester. This led us to believe that in order for us to pursue the project as it was originally envisioned, we would have to reduce our course load to 12 or maybe even 9 credits considering the amount of time and work required for the other courses such as ENSC325, 327 or 383.

6 Interpersonal And Technical Experience

6.1 Andrew Morning-Smith

Through my work on this Engineering Science 340 project I have learned new things in a number of areas, primarily with respect to group dynamics and technical knowledge of Bluetooth technology.

In regards to group dynamics, I've discovered that no group works well without someone to lead them – a completely flat structure tends to leave no one responsible for anything in particular and once this happens tasks may get delayed while no one picks them up. I found that at least by assigning each person in the group with a defined task, written down, with deadlines included things seems to get done more efficiently.

As far as my improved technical knowledge goes, I have learned a great deal about the Bluetooth protocol and issues involved in implementing it. This knowledge also extends to beyond Bluetooth and into other wireless technologies as I did a good deal of research into the competing wireless technologies before choosing to go with Bluetooth.

On a more practical side, I also discovered that there are some things that can be done by companies that are not so feasible to be done by a group of students. After being on several coops I found that for a company to be successful, they would often go to great financial lengths to achieve their goals – if succeeding in something you were doing meant going and buying some equipment that cost thousands of dollars then you did it. By following this model to some degree with our project, we found ourselves buying a new and upcoming technology (that is still very immature and expensive) for more money than we should have likely spent. Instead, a better idea may have been to choose a different technology, despite the interest we had in working with technology on the cutting edge.

In summary, I think that I've learned a great deal about how to work on small projects more effectively and have gained some very essential technical knowledge along the way, both of which will help me a great deal in my future engineering career.

6.2 Ryan Lee

This project has taught me many things about working on Engineering Projects. Not only have I learned a great deal about the technical aspects of Bluetooth, I have learned a great deal about team dynamics.

During the course of this project I was responsible for coding the user interface for the host side device. I learned a lot about the writing of Windows based programming using MFC.

With regards to Bluetooth, I have learned that cutting edge technologies are a mixed bag. They may be the next great technology of the future but with those great promises come pitfalls. I have noticed that many new technologies have very limited documentation and are quite expensive.

Working with this team has taught me many things about team dynamics. Most importantly, I have learned that you have to be patient and understanding with other group members. If you do not like the ways things are being done you must remember that they are all working on the same goal.

In retrospect, I have learned many valuable lessons that I will apply to my future endeavors. I felt that this project allowed me to enhance my ability to work with others. If I were to do this project again, I would choose a cheaper and more mature wireless transfer technology.

6.3 Savio Lau

Through the work that I have done for this 340 project I have learned a lot of important lessons. On the technical side, I learned how to work with small devices and their interfaces. In addition, I learned a lesson on group dynamics and management. On a more personal note, I learned the need of balancing workload with respect to time for courses and projects.

In terms of small devices, I have great appreciation for the need of memory management and also code optimization, through the development of the User Interface for the PSD. In many cases, there are very stringent constraints imposed on the use of small devices in terms of performance, I/O ports and memory. I have learned to limit and optimize code such that these constraints are met. In addition, I learned about using some of the most common interfaces that are used in the industry such as UART, RS232 and SPI. Furthermore, I learned a great deal in terms of communication between devices.

In regards to group dynamics, I have discovered that flat structure does not work too well in a group such as this. With so many things being verbally communicated, we ended up having some items “fall through the cracks”. This led to untimely delays and also increased the time that we required to make decisions. By the end, I learned that it's probably better to have things written down with hard deadlines. If I have to do this project again, I would make sure that there is a definite leader in the group that would push and guide the group along the project.

Furthermore, I have gained new respect and appreciation for Firmware. After this project, I am considering pursuing a career in Firmware as I find that it's one of the things that I enjoy to do.

On a personal note, I have find that I have underestimated the work required for this engineering project and the various other courses for this semester. As a result, my academic and project performance suffered. In retrospect, I think I would have reduced my workload to 12 or less credits so that I can spend more time on each subject instead of spreading myself so thin on many tasks.

In summary, I think that I have learned a lot more about designing and implementing projects effectively. In addition, I have gained much technical knowledge along the way. All in all, it's been an enjoying yet stressful experience working with three other wonderful friends.

6.4 Veljko Jovanovic

The technical and interpersonal communication challenge of working on a project from its inception to full completion makes ENSC 340 one of the most valuable courses that I have taken so far. It is very unfortunate that the course is scheduled in a semester with four other upper division ENSC courses making time-management very difficult. Our group managed to handle most of our difficulties through our weekly meetings and closely working together on the various parts of the project, but the course load of the remaining courses made things extremely difficult nonetheless.

By being in charge of the hardware development for our project I have improved my knowledge and skills in part acquisition and circuit construction. Things were far from perfect when considering part acquisition. For example, in the case of the biggest item in our budget, the two Bluetooth modules, I had hoped that my contacts within Ericsson would make it easier and cheaper to acquire the modules. However, this was not so and we were forced to order the modules from a vendor of Ericsson parts at a student discount price (which was still quite high). Circuit design and construction were a challenge of their own, but I count that as one of the most productive and interesting experiences I have had during this project.

My work on the hardware and on the code for the micro-controller enabled me to enhance my knowledge about interfacing a micro-controller with LCD's, PC's (through UART), and PIC's in general. Analysis of the circuit used in the iDAC project also gave me a better understanding of interfacing a Flash Card with a PIC and an MP3 decoder.

By far the most valuable part of this project was experiencing the team dynamics during the various stages of the project. It has made me realize the importance of clarity when defining problems and devising solutions, and the importance of patience and an open mind when looking at team productivity. It was a real pleasure to work on this project with the members of our group, and I believe that through the triumphs and defeats we had we also became better friends.

7 Conclusion / Recommendations / Future Plans

This project was very ambitious and we paid the price for going with cutting edge technology. The immaturity of the Bluetooth protocol meant that documentation was very limited and minimal. In addition, there were almost no sources of external technical assistance. We believe that going with a different transfer mechanism may have reduced that amount of headache caused by Bluetooth. Some choices for another transfer protocol included (but are not limited to) Wireless Ethernet (802.11), Infra Red and Radio Frequency.

Some functionality, which may be included in future revisions of the project, include the ability to upload songs from the portable to the host side device. We may also include so advancements as streaming audio transfer. Upgrading both the portable and host side of the Bluetooth connection would also be advantageous in making the most of the transfer rate provided by the modules. This project could also be modified to interact with other Bluetooth enabled devices, such as phones, and personal digital assistants.

We are quite happy with the direction that this project has gone in and we think that portions of this project could be taken for work as a thesis project for one or several of our Engineering degrees. We also plan to make our web page more visible to companies looking for technologies such as this for their products and would be willing to enhance the robustness and functionality of our system for a fee. The web page could also serve as advertising for ourselves to companies looking to hire new employees with experience in the Bluetooth field.

8 Sources and References

1) Sigma Comtec, http://www.comtec.sigma.se/

2) Bluetooth Solutions from Ericsson, http://bluetooth.ericsson.se/

3) Ericsson, http://www.ericsson.se/

4) Engineering Science Communications Handbook: Module 3 7th Edition, S. Stevenson, S. Whitmore, 1999.

5) *Syndeo Module Project Proposal*, A. Morning-Smith, R. Lee, S. Lau, V. Jovanovic, September 25th, 2000.

6) Specification of the Bluetooth System, Volume 1 – Core, v1.0 B, December 1st 1999

7) Specification of the Bluetooth System, Volume 2 – Profile, v1.0 B, December 1st 1999

8) A commercial Bluetooth stack for microcontrollers, http://www.adamya.com/sysdiv/bt/bt.htm .

9) Flash Memory Specifications, http://www.sandisk.com

10) Bluetooth specification, www.bluetooth.com

11) Online ordering (for pricing estimates), www.arrow.com