Links Performance Inc.
School of Engineering Science, Simon Fraser University
8888 University Drive, Burnaby, BC, V5A 1S6
links-440@sfu.ca

April 24, 2003

Lucky One
School of Engineering Science
Simon Fraser University
Burnaby, British Columbia
V5A 1S6

Re: ENSC 440 SmartShifter™ Process Report

Dear Lucky One:

Attached is the Links Performance Inc. (LPI) group's *SmartShifter™ Process Report,* which outlines the process our team went through when designing and implementing our system for ENSC 440. Our goal was to design an automatic transmission, which allowed for automatic and manual shifting for bicycles.

This document details the current state of the device, any deviations from the proposed designs, and future plans for the device. We also talk about any budgetary and scheduling problems we encountered throughout the semester.

LPI consists of four fourth year engineering science students, who are all very dedicated to the SmartShifter™ project. The SmartShifter™ members are Nin Sandhu, Trevor Hawkins, Zafeer Alibhai, and Jack Choi. If there are any questions or concerns regarding the functional specifications, please don't hesitate to contact me at (604) 721-8529 or email the SmartShifter™ team at links-440@sfu.ca.

Sincerely,

Nin Sandhu
President and CEO
Link Performance Inc.

Enclosure: *SmartShifter™ Process Report*

Links Performance Inc.

SmartShifter™

Process Report

Project Team:
Nin Sandhu
Trevor Hawkins
Zafeer Alibhai
Jack Choi

April 24, 2003

Table of Contents

1. Introduction

In the last 13 weeks, the LPI team members have slaved endlessly to create the SmartShifter™ System. The four group members Nin Sandhu, Trevor Hawkins, Zafeer Alibhai, and Jack Choi devoted every spare minute to make this project realizable. This document details the process that took this project from a piece of paper to the end product. Thankfully our demonstration went exceedingly well as out prototype worked quite well and nicely proved the automatic bicycle transmission concept.

2. Current State of the Device

Currently, the SmartShifter™ prototype implements a proof of concept for an automatic bicycle transmission. An automatic bicycle transmission is a worthwhile product as shifting gears is distracting and complicated, especially for beginner and senior riders. As planned, we consider the user's pedaling rate (RPM or cadence) to be the variable of control. The pedaling rate is controlled by the microcontroller via a gear changing mechanism that interfaces with a standard bicycle transmission. Figure 2.1 shows the main functions of the SmartShifter™ system:

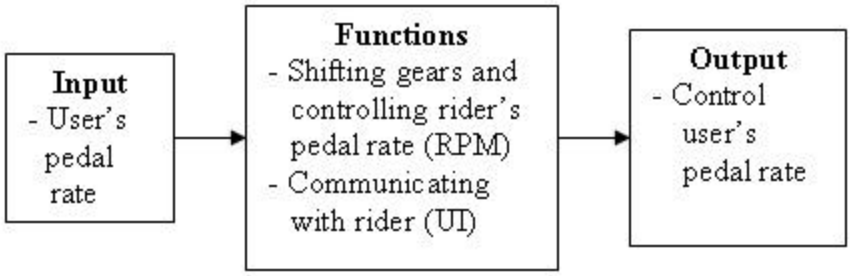


Figure 2.1: Block diagram of SmartShifter™ system.

Overall, very few aspects of the SmartShifter™ design were changed from the design specification. We produced a working prototype by the demonstration deadline that incorporated the main features we required for an automatic bicycle transmission:

- Monitor and calculate user's RPM with Hall-Effect sensors and the microcontroller
- Shift gears manually from the handlebar-mounted buttons when in manual mode
- Shift gears automatically under control when the rider's RPM is outside a user-defined window, for a user-defined number of pedal strokes (when in auto mode)
- The user interface allowed the rider to view the complete status of the system (power, shift mode, shift delay, gear number, RPM and shift occurring) at all times
- The user interface also allowed complete setup of all system options (RPM window, shift delay) and gear positions.

Therefore, the prototype is more or less completely done to our satisfaction and to the specifications laid out in the Functional and Design specifications. We were able to demonstrate that the concept works on a bike track-stand with variable load, and even allow anyone who desired to try the bike. The user interface is fully-featured with all options and settings being adjustable and the complete status of the system being viewable by the rider. The monitoring of the rider's pedaling RPM worked properly within 3% accuracy. The one area where the proof of concept was not satisfactory is that

our open-loop motor/derailleur system would need to be a closed-loop system in a real world product. Luckily the random errors that are introduced because the system is open loop are small enough that were able to successfully demonstrate the prototype. This is the one main area where the prototype is deficient from being a successful demonstration of a real-world prototype, in all other areas we are confident that our chosen design is sound and simple needs to be iterated further.

Overall, we implemented all of the main features that we viewed as being absolute requirements to demonstrate a proof of concept. We also implemented most of the desirable features that we felt were worthwhile doing, such as extra display features for the rider. Therefore, for the most part we followed our Functional and Design Specification documents very closely and this paid off for us as we had expended considerable time in ensuring these documents accurately described the problem that we are solving and the design that would solve this problem. The deviations in design and implementation from these documents is discussed the following section.

3. Deviation of the Device

The prototype produced for the demo deviated slightly from the design specification document, with the largest changes being in the user interface and gear changing mechanism.

3.1. User Interface

The user interface and hardware was exactly as laid out in the design specification. The largest difference is that we did not implement two of the features called for in the functional specification, such as displaying the speed and allowing a shortcut to modify the RPM window. We did not include these extra software features as they were not needed for a prototype and we were very short on memory for the software. In addition, for simplicity we mounted the power electronics of the stepper motor driver in the UI box, which was a late decision to make the wiring easier. This resulted in the UI box being a bulky and very unsexy, whereas we had hoped for a much sleeker and compact user interface. We did make some additions, such as displaying the value of the shift delay counter when the user is outside the RPM window and including three status LEDs beside the LCD (power, shift mode and shift occurring).

3.2. RPM and Speed Sensors

The RPM and speed sensors chosen, which were digital hall-effect position sensors worked exactly as expected. They were very reliable and extremely easy to interface to our microcontroller as they provide a very clean digital output. The only significant change is that we did not actually use the speed sensor, mainly due to constraints on microcontroller memory.

3.3. Gear Actuator Mechanism

The gear changing mechanism had the most significant deviations from the design prototype. Most notably we were not able to mount the motor on the derailleur, or on the bike at all. This occurred as the SFU metal shop was busy in addition to being on strike, so we would have needed to hire an external metal shop to craft the mounts required. As we decided that as the bike would not be made road-worthy due to time constraints and that a demonstration is better done inside, we felt that mounting the motor on the bike was no longer a priority. Therefore, we crafted a simple, but large motor mount and cable mount that allowed us to shift the derailleur via a normal shift cable that runs inside a housing. This was a major change as we had desired to mount the motor directly on the derailleur and remove the need for a cable at all. The motor mount was a large metal box that had sharp edges and was much too large to mount on the bike due to the limited metal working abilities of the four group members.

Next, we ended up building our own stepper motor driver. This was done for cost-effectiveness. We ordered an L297 controller and Fred Heep gave us an L298 driver IC. From these two main ICs we built our own stepper board that took as input signals a clock, direction, enable, reset, home, and step mode. The outputs were the four lead wires to the motor. This device was pretty simple to build and worked reliably throughout our use of it.

The final main deviation from the specification is that we did not use any feedback for the motor and derailleur system. We had discussed and considered feedback in the design specification, but we felt that we could do a proof of concept with the open loop system, after first testing it. Unfortunately due to final exams and time constraints we could not fully test the open-loop system shifting until a few days before the demonstration, so that by the time we did find out that we randomly lost steps, it was too late to obtain any extra parts to solve the problem. Had we tested the shifting of the prototype earlier we could have closed the loop around the motor/derailleur gear changing system relatively easily and cheaply. Thankfully the number of steps lost randomly on the downshifts when the motor must work against the derailleur spring is quite small and does not occur with great frequency. Note that the number of steps lost varies significantly depending on the rider and load, and that we attempted to minimize the lost steps by careful software coding and by running the motor as slowly as possible (t produces more torque at slower speeds). In fact, had we spent the extra time and money on a feedback device to close the loop, it would have produced little if any extra “wow” factor to the demonstration. The observers simply did not notice the random lost steps, because even after a number of cycles throughout the range of gears the error was very small.

3.4. Microcontroller Hardware

We had planned to use a donated Motorola HC11, but it was too ancient and decrepit for us to use. At this point it was early March and we were anxious to get coding and testing our hardware, so we simply ordered an HC12 EVB as they were available online within two days. We were also familiar with this microcontroller and had access to a number of people at school who have extensive experience with these controllers. The interface hardware of the microcontroller to the input and output devices were as mentioned in the design specification, with a number of small changes: we ended up using an 8-line priority encoder to encode our user interface buttons and a CMOS transistor IC to power the LEDs.

3.5. Microcontroller Software

The microcontroller software worked exactly as specified in the design specifications, there were no significant changes. The main differences are that we added small extra pieces of code not mentioned, such as button de-bouncing and motor homing. Overall,

the software development and integration went very well after a short learning curve with was necessitated due to the awful painstaking nature of assembly programming.

3.6. Power Supply

We ended up using a lab power supply for the prototype, mainly as the best demo possible is done on a track stand so that observers can see the shifts occurring easily. An actual road test is very valuable and is needed to continue development of the product, but unfortunately we did not have the time to properly pursue making the prototype road-worthy. Therefore, we felt that the best decision was to use the bench power supply for all testing and demonstration. However, we did research rechargeable power supplies and will of course consider this an absolute requirement for the next iteration of the prototype.

4. Future Plans

4.1. User Interface

- ***Design a better enclosure box***
  The UI unit enclosure box can be redesigned to a smaller size. The current dimension of the UI box is 15cm x 12cm x 15cm (*l x w x h*). The following are the solutions to reduce the dimension of the box more compact.

1. Re-design the circuit board and minimize the circuit space needed.
2. Build the enclosure box out of thin plastic material and minimize the spaces between the boards.
3. Remove the stepper motor board (as it is the power/driver electronics) and make a separate unit mounted near the derailleur.

- ***Larger LCD***
  The current LCD used is only 4 lines in length (Row). Due to the limited space on LCD the attributes displayed on LCD are not easy to see. One solution is to use a larger LCD, and preferably a graphical LCD with color.

- ***Display More Attributes***
  Some attributes designed to appear on LCD, such as speed, were removed due to the memory constraint of MCU, and LCD size. These can be added back on.

4.2. RPM and Speed Sensors

- ***Mounting of RPM Sensor***
  On the prototype, the sensors are mounted using zap straps. An enclosure of RPM needs to be remade with a strong, metallic material and be bolted on the frame of the bike. And the speed sensor needs to be included in the software.

4.3. Gear Actuator Mechanism

- ***Derailleur design/ Motor mount***
  Instead of using a standard derailleur, design a new derailleur that allows the motor to be mounted as described in section 3.3. This is the most desirable gear changing solution, but also the most difficult.

- ***Feedback***
  As prototype to prove the concept, our actuator mechanism worked in a open loop system. This resulted in losing steps between shifts. The problem can be resolved by implementing a close-loop system with feedback. The feedback

could be done on the shift cable, on the derailleur, or with magnetic sensors by determining upon which gear the chain is riding.

4.4. Microcontroller Hardware

- ***Use a Different MCU***
  Instead of using HC12, it is more cost efficient to use a PIC microcontroller, or another MCU that supports larger on-board memories for C/C++ programming.

4.5. Microcontroller Software

- ***Use C/C++ Programming Language***
  The prototype was implemented using HC12 Assembly codes. A problem with HC12 is that one line of code can break the entire program. Also, the debugger software does a poor job of describing the problem. Due to these problems, it is much more time efficient to use C++ language and use Object Oriented Design.

4.6. Power Supply

In any future version of the product, we would wish to power the system by battery rather than Power Supply. In using batteries, we could make the system portable and conduct road tests, bring us one step closer to a final product. We would require two separate battery packs, operating at 4.5 V and 12 V respectively. We would also want both the battery packs to be rechargeable to reduce ownership costs and be environmentally friendly.

The 4.5 volt supply would power everything except the actuator. These components, which include the MCU, sensors and the UI, require 100mA of current. In order to generate 4.5 V we would use three AA batteries connected in series. As AA batteries are rated at 2800 mAh. Therefore, the system would have a runtime of 28 hours.

For the 12V actuator supply, we found that each up-shift requires 400mA while each down-shift requires 1000mA. This discrepancy is due to the fact that the spring in the derailleur aides up-shifts while it hinders the down-shifts. We also found that each shift took 4 seconds to occur. The best solution we found was a rechargeable 12V drill battery with a rating of 2000mAh. At an average of 700mA per shift, we determine that if the battery was to last 28 hours (so that both battery packs could be recharged at the same time) 91 shifts/hour (1.5 shifts/min) could occur. If the shift rate increased to 120 shifts/hour (2 shifts/min) then the battery would last 21.4 hours.

5. Budget

The final budget for construction of the prototype is shown in Table 1, below.

Resources	**Estimated**	**Actual**
MCU – HC12	$200	$268.69
Actuators	$150	$291.72
Sensors / Buttons	$50	$108.28
LCD	$75	Donated
Batteries	$25	N/A
Enclosure	$50	$22.71
(Miscellaneous)	$50	$102.85
Contingency Fund	$100	N/A
Total	$700	$794.25

Originally the ESSEF donated a HC11 EVB; however, the communications protocol was extremely outdated making it almost impossible to use. At this point we were already well into the project, so we choose to use the HC12 (rather than a PIC) which we were already familiar with.

We were able to find an actuator that would fit our needs within our price range, but it was back-ordered 12 to 14 weeks. We found a similar motor from a supplier in the US, but it had to be shipped directly from the UK. Not only was the actuator more expensive, but we also ended up paying over $60 in shipping and duties. In our original estimate we had also failed account for the cost of the electronics to control the motor.

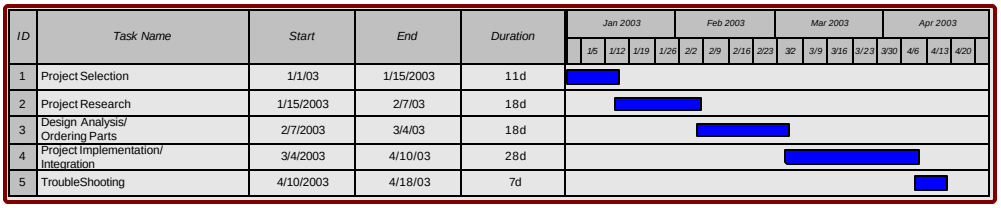
The rocker buttons we used were rather expensive, inflating the Sensors & buttons category. We discovered that we needed many small electronic components and tools which raised the miscellaneous category. Due to our cautious approach of handling and testing equipment, we didn't damage or destroy anything, so the contingency fund wasn't required.

6. Time Constraints

During the early stages of the project our team had come up with an initial schedule outlining when the project milestones and key components would be completed. The initial and updated schedule can be seen in figure 6.1. Throughout the semester we had noticed that our schedule was incomplete and most of our deadlines were being extended. When we first created the schedule we never went into complete details on how things would be done, when they would be ordered, and how would it be tested. These key sections lead to our original schedule being severely flawed and unusable. We believe we could have better planned for the project if we had sat down and made a complete and accurate schedule detailing the construction from the start to the end of the project. A complete schedule would have lead to us discovering flaws in our system at an earlier time and then allowing us to fix them, but in our case we learned this too late. When we began our final testing a weekend before our demo, we concluded our open loop system was not too reliable. If we had scheduled for testing a head of time this flaw could have been prevented and fixed. We also had problems with ordering parts and waiting for them before any work could be done because we hadn't scheduled for the delivery time.

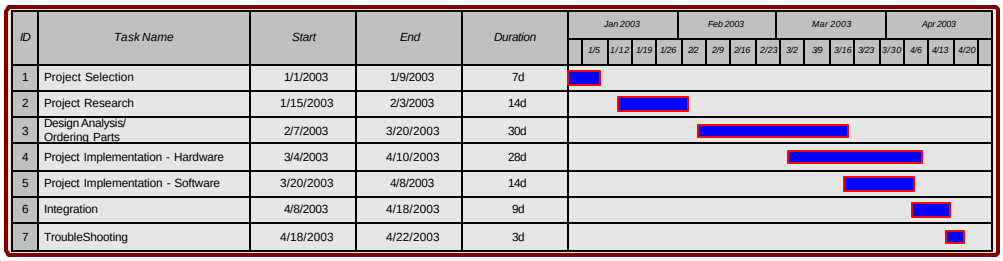
However, even though our original schedule did not provide us with a lot of help, we did learn the value of having an accurate schedule. Planning the project from start to finish in advance allows problems or flaws to appear at earlier dates of the project. Our team learned the value of scheduling the hard way, however, in the end we completed the project successfully and demoed on time.

Proposed Schedule



ID	Task Name	Start	End	Duration
1	Project Selection	1/1/03	1/15/2003	11d
2	Project Research	1/15/2003	2/7/03	18d
3	Design Analysis/ Ordering Parts	2/7/2003	3/4/03	18d
4	Project Implementation/ Integration	3/4/2003	4/10/03	28d
5	TroubleShooting	4/10/2003	4/18/03	7d

Actual Schedule



ID	Task Name	Start	End	Duration
1	Project Selection	1/1/2003	1/9/2003	7d
2	Project Research	1/15/2003	2/3/2003	14d
3	Design Analysis/ Ordering Parts	2/7/2003	3/20/2003	30d
4	Project Implementation - Hardware	3/4/2003	4/10/2003	28d
5	Project Implementation - Software	3/20/2003	4/8/2003	14d
6	Integration	4/8/2003	4/18/2003	9d
7	TroubleShooting	4/18/2003	4/22/2003	3d

Figure 6.1: Gantt Charts

7. Technical Experience and Team Dynamics

7.1. Jack Yoonchan Choi

Working for LPI has given me the opportunity to mature in technical knowledge as well as in team dynamics. Although most of my contribution on the project was implementing the software, I had a chance to experiment and fiddle around with all components used. Some of my contributions include designing and implementing the UI button algorithm, and also other subroutines such as Output capture and RPM control.

Since the beginning of the project we met 3 times a week for 5 hours each. As time proceeded we met almost everyday. All work was done together or in groups of 2. Although it may not be the case in the real world, I think we were only successful because we chose to work together over assigning work to each other and working individually. The reason being, in the real world it takes months or sometime years to plan out the design. However we were given only 4 months to complete the project, and of that 4 months we spent 2-3 weeks for design. New problems came up left and right during the initial phase of implementation but because we had everyone working together, the problems were resolved quickly and most efficiently. Working together also allowed everyone to see the whole picture of the project.

We did a good job of keeping all designs and flow charts on paper. They were kept in a binder in order so if anyone needed to reference back to the design it was easily found. Although it seemed time consuming to write everything on paper, at the end we all agreed that this was probably the smartest thing we did. We had very little problem implementing components and when each components were put together they worked exactly as we expected.

I really enjoyed working with Nin, Trevor and Zafeer. This was my second project course. The experience and knowledge learned from the two projects were quite different. The reason being, my first project course was done by a group of best friends, where as this time I met these guys at the start of the semester for the first time. I can't say which was better because they both have pros and cons. However one advantage worth mentioning is that because we did not know each other well, we tried to be more patient and understanding to each others.

7.2. Nin Sandhu

Throughout the semester, I have learned valuable skills with project design, research, scheduling, and group dynamics.

In the beginning, most of our time was spent detailing how the mechanics of our system would work on paper. The project we had chosen was 50% mechanical and 50% electronic, which is the right blend I wanted. I learned the value of iterative designing is key when designing any mechanical system. Mechanical design requires a complete analysis of all forces, torques, and inputs acting on the device, which made the problem so much harder. As well as designing mechanisms on paper we also had to communicate the ideas to the other group members, so they could provide feedback on the design. I found the design process to be interesting because this is how real world problems would be solved.

The most fun part of the mechanical design was actually going to the machine shop and seeing the designs come to life. The not so fun part was when I cut my hand in the shop, which is fine now.

The low level electronic hardware designing wasn't as bad as I had originally thought. When I sat down and started implementing it, I actually remembered stuff I learned from previous courses. It was nice to know that after all I learned I actually understood the electronics of what I was doing because after all that's what I paid for.

The most painful part of the project I thought was the software. Trying to remember assembly language was a pain because it was so long since I had coded in that language. I swear my hair started to fall out because it was so stressful. I hated the fact that we had such limited space on our MCU because we had to optimize the code to death before it fit in microcontroller. However, by the end of the semester I managed to become a master at coding in assembly, but next time we all decided that we would use a language which is simpler.

The last thing that was equally important to the project was the documentation. The documentation provided non LPI members with information pertaining to the design, functionality, and testing of the device in question. As in the real world we would be expected to create such documents for shareholders, customers, and etc. I thought it was a meaningful process to write these documents because it would be useful in industry.

Working as a group was also interesting because I learned that everyone has completely different views or opinions and everyone has a right to express them. I learned to be patient and respect everyone's views before coming to any decisions. I will admit we had some heated arguments, but I think being brutally honest to each group member made it easy for us to communicate with each other. The team I worked with was great and hopefully we can work on future projects together.

7.3. Trevor Hawkins

During the course of the project I gained valuable experience in fields such as project scheduling and planning, mechanical and electrical design and group dynamics. On the technical side, I learned a great deal about mechanical design, and enjoyed working in the machine shop to build the motor mounts and the UI enclosure. I gained a great deal of knowledge about mechanical design and how it differs from electrical engineering and other fields. I enjoy mechanical design but learned that to build things you need access to a good machinist (and the time to wait) as very few individuals can design and machine the hardware required for complex designs. I also gained a great deal of knowledge about electrical component selection, ordering and assembly as we had to build the user interface and stepper motor drive from scratch and interface them to the microcontroller. Thanks to ENSC387, I was able to use and augment previous knowledge about magnetic sensors and stepper motors in particular. I enjoyed working with all of the hardware a lot as it was a new experience for me, but it was especially cool to do an electromechanical system. On the software side I wish we had gone with a microcontroller that supported C programming easily, but after a steep learning curve I found assembly programming to be quite fast and acceptable for a simple program such as ours. I also gained experience in software planning and design, as I found that by designing all aspects of the software together in a modular way, the testing and integration of each unit was relatively simple. Overall though, I did not find the software development to be as rewarding as the hardware experience, most likely because I had similar previous software experience.

Perhaps more important than the technical skills gained over the project, are the knowledge of scheduling and team dynamics acquired. I learned how important it is to break down a project into all the little tasks that must be done to solve the problem and to schedule them all. I learned about how things such as weekly meetings, reminders and other things can keep everyone up to speed and the project moving along. Especially in the research and design stages it can be easy put the project on the back burner when other school courses are taking up a lot of time. We put serious effort into the functional and design specification documents, and I felt this thorough planning and design really helped us in the later stages of the project once the hardware had arrived. In retrospect we worked much harder in the tail end of the semester and should have put more hours into the project early part of the project. As a result our parts were ordered later than scheduled and this set us back a few weeks. Had we had a couple extra weeks at the end of the project, we could have implemented feedback and made the project significantly better, so really, making a proper schedule and sticking to it is of utmost importance for a real-world project. Leaving the full system-level testing until a couple weeks before the deadline simple wont cut it when you have a design flaw such as our lack of feedback on the gear actuator system. Finally, I gained a significant knowledge of team dynamics and group skills. Keeping a group motivated and on-task at all times can be difficult

especially when all are in 15-18 credit hours of classes. Someone has to send out reminders and organize weekly team meetings. Without weekly meetings to schedule, prioritize and assign outstanding tasks, it is very difficult to keep all group members up to speed and contributing. Overall though, I enjoyed working with my fellow group members, and apart from a few short stress-filled arguments, we had enjoyed very open and free communication with no group member being afraid to express their opinions or dissent from other's views.

7.4. Zafeer Alibhai

Everyone in our group got a chance to work on all areas and stages of the project. As result, I was always well aware of what was going on. By the end of the project I discovered the importance the R & D and Scheduling phases of the project. Some of our design decisions we flawed, especially the choice not to use feedback.

Although we made mistakes, I think it was important for use to make them. I feel that if things had worked out more smoothly, I would have learned less. Inevitably, I would have made a bad design choice or done a poor planning job at some point in my career. At this time, the mistake could be much more costly to both my employer and myself.

During the construction of the prototype, I focused my attention on the LCD display software and software integration. Due to the lack of memory on the MCU and the size of our program, I had to learn to use the flash portion of memory. Due to this memory problem, I also learned to use many of the more advanced commands to optimize our code. During the software process I had to do a lot of debugging, which I feel has enhanced my debugging abilities. As a result, I feel that I have a much better handle of assembly code and would easily be able to TA a course in the area.

The best choice we made during the project was to design all of our functions on paper before we programmed them. We also decided all the variable names (and their types) to be used. Each function was broken down into a flow chart so that it could easily be programmed. Once we had all the flow charts, it was just a matter of turning the pseudo code into assembly. We also planned exactly what the UI display would look like and how the buttons would interface with it on paper, prior to programming. In the end, this made integration a lot easier and the only major problem was memory.

Although we had our differences from time to time, I think we worked well as a team (and we're all better friends for it). I think this was mainly due to the fact that we could all express our opinions and know that they would be listened too. Our idea of voting on issues also helped. Since we worked either as a group or in pairs, it was evident that we were all putting in our fair share of time. Overall, I am pleased with the results of our efforts.

As an afterthought, I think this course would work better over an 8-month time span. The first 4 months could be used for R & D and planning; while the second 4 months could be used for construction. If we had an 8 month time frame we would have been able to order the actuator at a lower cost (and wait for delivery). We would also have time to look at a wider variety of components and learn how to use them. In our case, this would allow us to select a PIC Microchip rather than the HC12, reducing our project costs. Another advantage of a longer time frame is that we would be able to fix design flaws, leading to a better product.