

Accessible Manufacturing Equipment, Phase 2

Sponsor: Resource Center for Persons with Disabilities
Stephen Blosser



ECE 480 Design Team 2

Facilitator: Dr. Fang Peng
Nicholas Neumann
Ralph Prewett
Jonathan Brouker
Felix Adisaputra
Li Tian

Contents

Chapter 1: Introduction and Background	5
Introduction.....	5
Background.....	5
Chapter 2: Exploring the Solution	7
FAST Diagram.....	7
Primary and Secondary Function Identification.....	7
Primary Function	7
Secondary Function 1	8
Secondary Function 2	8
Secondary Function 3	8
Secondary Function 4	8
House of Quality.....	9
Design Overview.....	10
Mechanical	10
Hardware	10
Software	11
Initial Budget	11
Project Execution Plan	12
Chapter 3: Technical Description	13
Design History.....	13
September	13
October.....	13
November.....	14
December	14
Hardware Implementation and Photo Documentation	15
Mechanical Systems	15
Hardware	20
LEGO NXT Brick.....	20
I2C I/O Expander PCF8574	21

Circuitry	21
Input	21
Eight Segment LED Display	22
Cutter Relay Control	23
NXT Power Supply	23
Programming	24
Software and Interface Design Requirements	24
Programing Language	25
Chapter 4: Results and Test Data	30
Final Design	30
Test Results	30
Sources of Error	32
Chapter 5: Conclusion	33
Appendix 1: Technical Roles	35
Nicholas Neumann	35
Ralph Prewett	36
Jonathan Brouker	37
Felix Adisaputra	38
Li Tian	39
Appendix 2: References	42
Appendix 3: Technical Attachments	43
GANTT Chart	43
Mechanical Systems	45
Program	47
Audio Circuit 1	48
Audio Circuit 2	48
Average Percent Accuracy Formula	49
LED Schematic	50

Executive Summary

People with disabilities face many challenges in the work place. Keeping pace with their more able co-workers can be difficult at times. A young man at a company called Lettuce Duit in Galien, MI works to cut ribbon that is fashioned into prize ribbons. However, due to both visual and cognitive disabilities this task becomes rather difficult. Stephen Blosser, one of the technical staff from Resource Center for Persons with Disabilities (RCPD), approached ECE 480 Design Team Two at Michigan State University to design a device to solve this problem. This project was a reengineering effort to improve upon an already functioning, handicap accessible ribbon cutting machine. The critical customer requirements included making the machine's processes faster and more precise, implementing an easy to revise programming interface, and solving heat and over current problems within the motor drives. To attain these requirements the previous motor drive was replaced with LEGO Servo Motors, which have better control, and operating characteristics. A capstand feed design was implemented to increase speed and the cutter was replaced to improve the precision of the cuts. Also, the programming interface was updated so that an unfamiliar user will be able to quickly make timing and other machine process adjustments.

Acknowledgements

Design Team Two would like to thank our facilitator Dr. Peng for his advice and guidance throughout the design process. We would also like to thank our sponsor Stephan Blosser. His encouragement and willingness to help proved to be invaluable throughout the semester. We thank the ECE shop staff, Brian Wright and Greg Mulder, for their advice and guidance in part selection. Finally, we would like to thank our friends and family for their continuous encouragement and support throughout the semester.

Chapter 1: Introduction and Background

Introduction

In 2008 a group of engineering students from MSU designed a machine that allowed a physically disabled worker to cut, pleated cloth ribbons used for making awards. There were two major problems with the machine. First, the machine used stepper motors to drive one of the most important machine processes of pulling the cloth to a correct length to be cut. The stepper motors lead to inaccuracy in the cuts, and they caused heat dissipation problems within the control system. The second problem was the programming interface, which was written in C and only a user who is familiar with the machine and language of C can make adjustments to the machine. This became a problem when the machine broke down frequently and Mr. Blosser was the only one capable of repairing it. The machine needed to have a programming interface that is easy to use so that locals near the machine could maintain the machine.

Background

The requested new interface for programming the machine was to be based on Lego's NXT-G programming environment, or National Instrument's Labview. Labview and Lego partnered to create the NXT-G environment so both solutions are similar. Using either programming environment requires the use of hardware purchased from the software supplier to provide the input/output interface for the rest of the design. For this reason the Lego NXT brick was chosen as the microcontroller to be used in the final design. This is because the Lego NXT brick was compatible with the NXT-G software and is also compatible with Labview software designed specifically to expand the Lego programming environment.

The motor drive system used two stepper motors operating in parallel. The previous system had two plastic rollers that were in contact with the rotating bar used for closing the movable clamp. Initial experiments by the group had shown that the system had unnecessarily high friction resulting in the high torque requirement of the stepper motors. In addition the machine was both inaccurate and imprecise in its cuts. By eliminating the

bar and clamp feed system and using the LEGO Servo motors as a feeding device the design becomes much more efficient.

Key components of the system include the cutting mechanism, as well the ability to provide visual and audio feedback of the machines length settings. The NXT Brick will receive the input from buttons mounted on the device and send the output via I²C communication to an LCD display and verbally communicate the length to the user. In the new design a guillotine cutter replaces the old shear cutter, not only does this fit our feed design better but it allows for a cleaner cut which is appreciated by our sponsor.

Chapter 2: Exploring the Solution

FAST Diagram

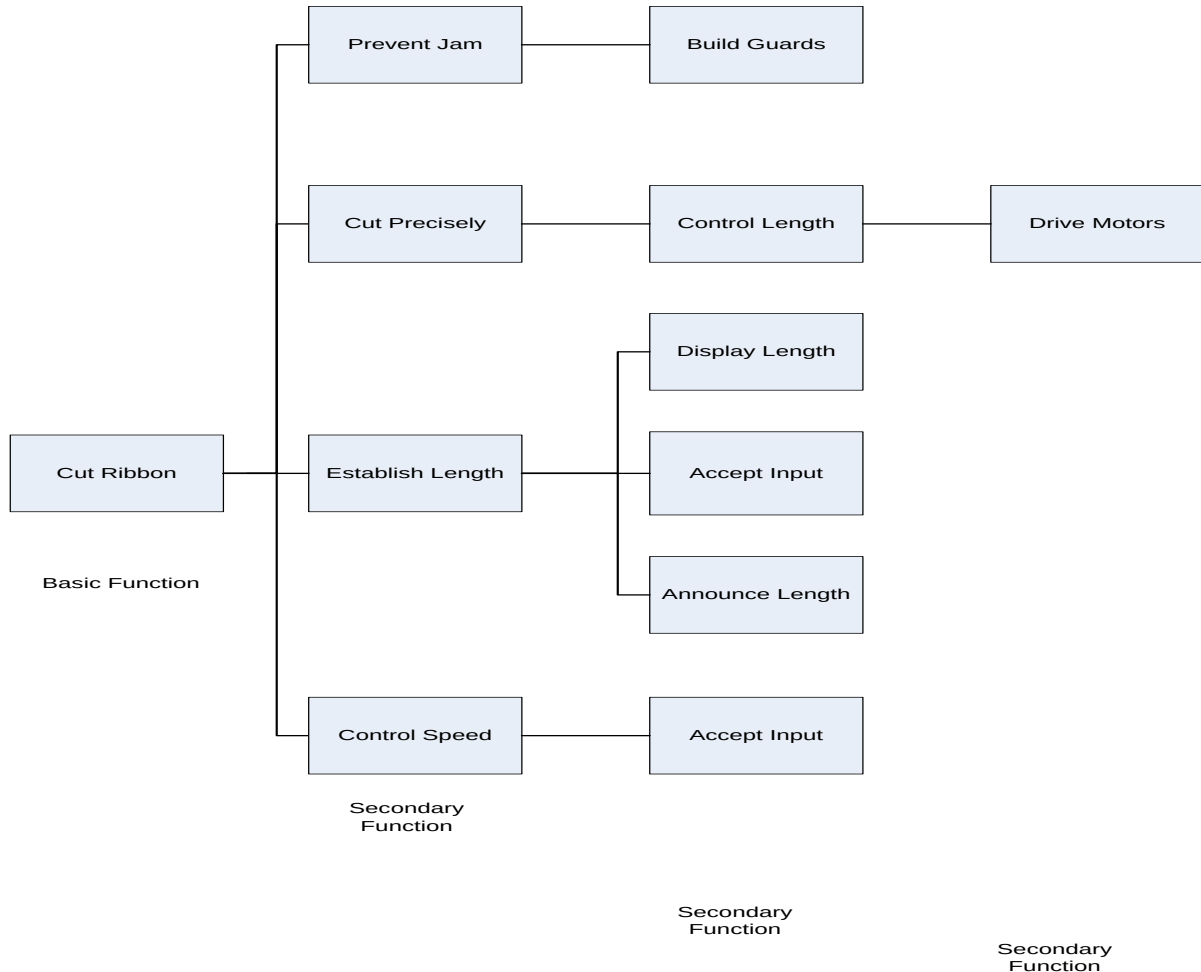


Figure 1: Fast Diagram

The FAST diagram above was created in order to help with identifying the needs and possible ways to design the machine. Defining the goals of the project helped to identify a strategy to successfully complete the design project. Each path in the fast diagram was treated as a specific system then the different design components were implemented as one, leading to a completed design.

Primary and Secondary Function Identification

Primary Function

The primary function of the machine is to cut ribbon. Failure to accomplish this will result in total loss of market value of the finalized product. Besides the primary function there are also several secondary functions.

Secondary Function 1

In order to cut the ribbon, the machine should have the ability to establish the desired length of the ribbon. This specific process could be broken down to more secondary functions as well. In order for the ribbon cutting machine to establish the desired length, it should have the ability to accept length input from the user. This input should be adjusted manually to the desired length by the user. Additional secondary functions could also be implemented such as displaying the length through the LED display and announcing the length by a speaker.

Secondary Function 2

Another essential secondary function is to cut the ribbon to the precise length. This function had also been requested by the customer. The consistency of the ribbon's length is important because the machine will be utilized in the future for mass producing cut ribbon. Therefore, a controller needs to be implemented throughout the initialization and cutting process. The controller should be able to maintain the precision during the whole cutting process. The controller will achieve this through direct control of the motors that are being executed during the cutting process.

Secondary Function 3

The ability to vary the speed of the operation of the machine is another customer requirement. Therefore, the machine should be capable of receiving user input to adjust the speed of the mechanical processes of the machine to suit the needs of the user.

Secondary Function 4

During the overall cutting process, jamming of the ribbon inside the machine is a risk, and could pose a major problem to the user. It is not desired to have the ribbon get stuck in the machine because that would result in the machine needing to be stopped and cost time to fix the jamming issue. Thus, preventing this potential jamming issue is another critical secondary function in designing this ribbon cutting machine. An appropriate tracking

guard should be mechanically implemented in the machine so the ribbon will have clear path throughout the cutting process.

House of Quality

The application of the House of Quality technique was employed to determine the Critical Customer Requirements (CCRs) for our design. The House of Quality technique is a tool to assist in decision making and to reduce design uncertainty. The CCRs are the measurements that can be made during our design process to ensure that we are making progress towards a successful design that will accomplish our customer's goals. A solution selection matrix is then constructed based on the House of Quality technique to decide the conceptual designs that will be implemented throughout the project for further development. A solution selection matrix is a system of matrices that translate the customer needs to engineering design specifications. Below is the complete solution selection matrix for our design.

Engineering Criteria	Importance	Possible Solutions										
● = 9 (Strong) ○ = 3 (Moderate) Δ = 1 (Weak)		Lego Servo Motor	DC Motor	LEGO Mindstorms	LabVIEW	RobotC	C++ Code	Connectors	Safety Guards	New Cutter	New Controller Box	Tracking Guards
Speed of Operation		5	○	●						●		Δ
Accuracy and Precision		5	●	○								
Safety		4								●	●	
Aesthetics		2							●	●	●	
Revisable Software		5			●	○	Δ	Δ				
Noise		2	Δ	Δ						●		
Compatibility		3	●	Δ								●

Mechanical Maintenance	3							Δ				•
Maturity of Technology	4			Δ	○	○	○					
Robustness	4	○	•	Δ	○	○	•					•
TOTALS		101	101	53	39	29	53	21	54	99	18	95

Figure 2: Solution Selection Matrix

Design Overview

Mechanical

The proposed design solution is to build an entirely new mechanical system that will improve the speed and accuracy of the machine. The old clamp based system was replaced by a new capstan system. New brackets were built and placed in the new machine to accommodate all of the new mechanical and electrical components. The new machine is also smaller, and can be considered more aesthetically pleasing. This new machine is expected to maintain all of the basic and secondary functions of the original ribbon cutting machine.

A new cutter was implemented in our new design as well. There were several cutting mechanism options available. The first was using the same electric cutter that the 2008 machine used. Another possibility was using a scissor like mechanism. However, it was decided that the new machine would use a guillotine cutter powered by pressurized air. This cutter will be operated by a valve assembly that turns on when given an electric pulse. To prevent the jamming problem, a tracking guard has been built. Safety will always be an important issue; therefore, proper safety guards were also built and used in an appropriate way to prevent any injuries to the user that might happen in the future.

Hardware

The use of the LEGO NXT robotics system was a customer requirement. Although LEGO servo motors have the same total value in our solution selection matrix with the DC motor, the LEGO servo motor has major advantage because of compatibility with the LEGO Mindstorms software therefore, they were chosen as the motors to be used in the design.

The LEGO Mindstorms System was selected over C++ as the project's programming language because of the customer requirements. Two LEGO servo motors are utilized in our design. These servo motors also came in the same package with the LEGO NXT Brick in the complete LEGO Mindstorms kit which is already provided by our sponsor.

A voltage regulator was used to step-down the 12V DC provided by a computer power supply to 9V DC. This voltage is used to power the LEGO NXT Brick which in turn will power the LEGO servo motors as well. A simple circuit applying a MOSFET was also built to control the valve assembly in order to operate the guillotine cutter.

The design also uses an I²C communication bus to expand the number input and output ports available on the LEGO NXT Brick. In order to do this, I²C circuits will be built using the PCF8574 I/O expansion IC. There are three of these circuits used. One handles the interface for the new LED display. The second circuit is used to run all the buttons/switches which will control the display, speaker, and speed level. The third one controls the MOSFET circuit.

Software

All of the programming is done in the LEGO Mindstorms environment. This means the LEGO NXT Brick performs all of the processing necessary for the operation of the machine and there is no need for an additional microcontroller in the final design. Communication over the I²C bus will be accomplished through downloadable programming blocks that are available through the LEGO Mindstorms programming environment. The final program of the LEGO NXT Brick would be able to automatically command all of the machine processes. Testing was also conducted to ensure that the new system meets all critical customer requirements and expectations.

Initial Budget

Our initial estimated budget is provided below,

Quantity	Parts	Cost
1	Lego Mindstorm NXT 2.0	\$299.99(*)
1	Guillotine Cutter	\$358.16

1	Valves Assembly	\$110.21
1	Computer Power Supply	\$20.00
	Misc. Circuitry	\$15.00
	Misc. Brackets and Metals	\$10.00(*)
	Total	\$813.36

Figure 3: Estimated Budget

Note: (*) items are already provided by our sponsor.

Project Execution Plan

The plan for how the design was executed from week 4 to the end of the semester had been documented in our Gantt chart. Please refer to Appendix 3 for the complete version of our Gantt chart. Throughout the semester, this Gantt chart was not changed significantly as we always put our best effort to stay on track with our Gantt chart. This Gantt chart had been an excellent guide for maintaining the best timeline to meet all the expectations at the end and also it had been tremendously helpful for us to keep up the pace of the design process for this project to accomplish all of our goals.

Chapter 3: Technical Description

Design History

September

We changed the overall design of our machine several times throughout the year. During September we planned on replacing the motors and motor driving circuitry for the original machine with a new DC motor, powerful enough to move the clamps much faster. We would control this motor with a PID controller programmed in C++. We would add a small DC motor with a flipper on it to replace the old machines loud and inefficient fan, and comment the C code making maintenance easy.

October

In October we decided to use LEGO MINDSTROMS NXT as the programming language at the request of the projects sponsor, Mr. Blosser. Because the LEGO programming language did not support the complex math required for a PID controller we had to change our motor control design. Our first choice was to use standard LEGO servo motors. These motors are very accurate but not very strong. Torque testing showed that the LEGO motors could match the old machines in speed if three were run in parallel. In order to increase the speed of the machine we looked into reducing the friction in the clamps used to pull the ribbon out to the correct length. These clamps ran along a rail on plastic rollers. The plastic rollers had a cardboard insert, we replaced this insert with one made of bronze infused with graphite. This met with some success, the rollers turned easier and the speed increased but not by much. We considered using larger stepper motors and replacing their control circuits with LEGO parts but we were not sure it would work well. Instead we opted to redesign the machine entirely. The new cap-stand feed based design was prototyped in LEGO using two LEGO stepper motors. This prototype was not stable and broke easily but it was fast, small, and simple.

The LEGO NXT brick has only three motor driver ports and four general low power I/O ports to interface with other devices. We decided that the machine should have an external speaker, eight buttons, three eight segment displays, as well as two motors. These

external devices would require more than the available seven ports to control. Our first plan was to use a PIC microcontroller to control the buttons and the LED display. These features would be unlikely to need reprogramming so they could be programmed in C++ without making maintaining the code significantly more difficult.

November

We realized that the LEGO NXT brick could use an I²C serial communication bus to interface with peripheral devices. Third party programming blocks even allowed us to use the LEGO NXT's simple graphical programming interface with I²C. This allowed us to program everything using the LEGO NXT environment without running out of I/O ports. We could use a SAA1064 I²C compatible LED driver to control our LED circuit. This chip takes serial input from a I²C bus and translates it into parallel data output for two eight segment displays or four with multiplexing. This chip also allowed us to adjust the LEDs intensity. Our buttons would be connected to the NXT brick through a PCF8574 I/O expander. The I/O expander has a very weak pull up current. When a button is pressed the chip reads the low voltage as a logical zero. The NXT brick does not support interrupt over I²C so the I/O expander had to be polled. For the audio driver we decided to use a ISD 1400. This IC can store voice files and also act as an audio amplifier, allowing it to drive a speaker. This chip would also be controlled using an I²C I/O expander. The I/O expander selects which address the ISD 1400 is reading sound files from and also when to play.

Unfortunately the third party programming blocks we were using to allow the NXT brick to interface with I²C proved incompatible with the SAA1064 LED driver. The LED driver used different registries to control each LED. While we could access the chips control registry by simply not specifying which registry we wanted to access we could not access the registries controlling what LEDs were turned on. We solved this problem by using three additional I²C I/O expanders, one for each digit, instead of the SAA1064 LED driver.

December

We finalized our design. The audio amplifier and memory ICs we had planned on using had to be scrapped in favor of the NXT bricks on board audio because the memory on the chip we were using turned out to be unstable.

Hardware Implementation and Photo Documentation

Mechanical Systems

The Mechanical System carries out the most basic functions of the machine. It must be capable of quickly, and accurately moving ribbon into a cutting mechanism and then cleanly cut the ribbon. The machine must be capable of accommodating ribbon of varying thicknesses, and width. The mechanical portion of the machine must also be robust since it will be used as an industrial machine. The design implemented satisfies all of these requirements.

The mechanical system for moving the ribbon through the machine consists of two capstans, similar to a tape recorder. This design presents a significant risk of jamming due to ribbon becoming caught on an edge, or being mishandled by the machine. Even though the ribbon is pleated, and has a natural curve as a result it was hypothesized that by using brackets it would be possible to pass ribbon through the machine and avoid jamming problems. Conversations with Mr. Blosser revealed that the student group from 2008 chose their mechanical system over the capstan fed system to avoid the jamming problem. Therefore, to prove that capstan design was a viable replacement for the original system a prototype of the ribbon cutter was built from all LEGO bricks using those that had come with the NXT brick.



Figure 4: LEGO Prototype

The prototype showed that the LEGO servo motors could easily provide the necessary torque and power to move the ribbon through the machine, and that a bracket system could be used to guide the ribbon. After being presented with the results of the prototype Mr. Blosser offered to assist in the fabrication of metal parts to build the final design. The original ribbon cutter was built around an adjustable frame that made it easy to adjust and modify. The new machine also features a foundation frame that uses the same type of beams that allow for metal inserts that fit inside the grooves of the beams. This allowed for the precise positioning of components because the metal inserts can slide along the inside of the beams. Precise alignment of the brackets that carry the ribbon is critical since air gaps between them must be kept to a minimum in order to reduce the risk of snag occurring with the ribbon that would cause a jam. Another advantage is that adjustments to the machine can be made with only a screwdriver and wrench.

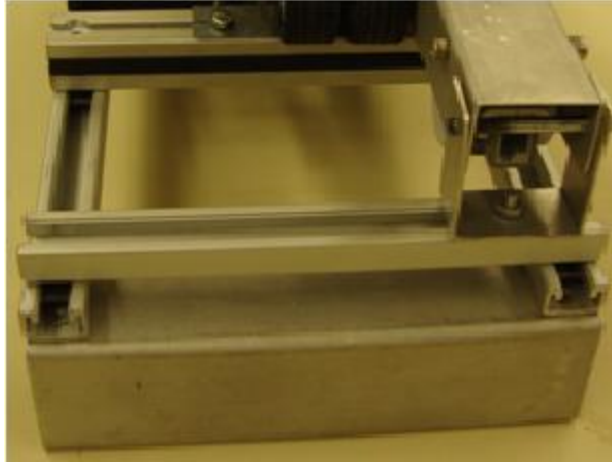


Figure 5: Adjustable Frame

The machine did not initially succeed in passing ribbon through because the ribbon would get stuck on the sharp edge of the blade. It was not possible to easily modify the cutter to have a larger gap when in the open position. This is because the blade and the piston that drives the blade were precisely made so that the thinnest part of the blade would have a minimal overlap with the bottom of the cutting surface. The solution to this problem was to add a guard that would compress the ribbon while inside the cutter so that the ribbon would clear the edge of the blade. This guard also needed to be able to retract while the blade descends; to do this the guard is mounted on a hinge that is spring loaded to the support bracket which is on the right side of the picture below. On the left side is an “L” shaped metal bracket that stops the guard from ascending too high.

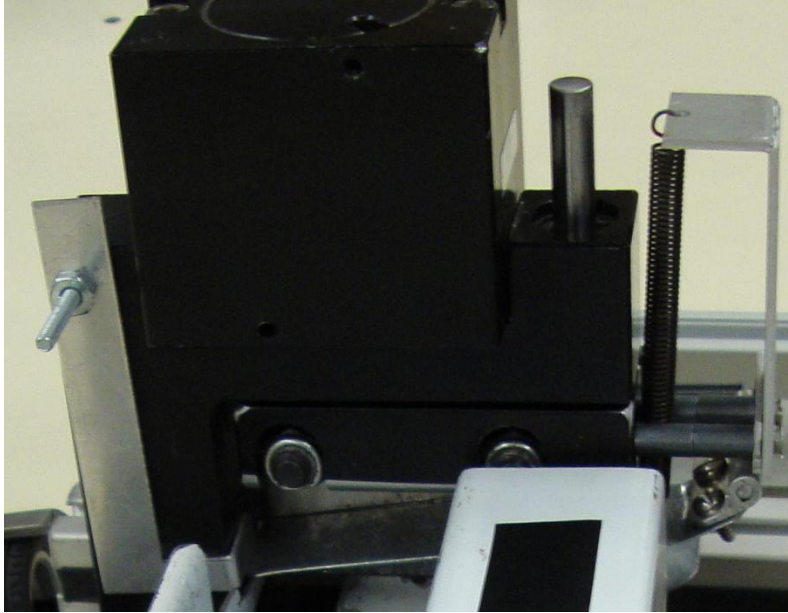


Figure 6: Spring Loaded Ribbon Guard

The NXT servo motors were given their own metal housings that can rotate around the pin that attaches them to the rest of the machine. This allows the rollers to adjust to the thickness of the ribbon being passed through the machine. The weight of the motors and their housings produces enough friction to ribbon through the machine.

The cutter is pneumatic. Pressurized air is available in the factory where the cutter will be employed. The pressurized air is controlled through a valve assembly. The assembly can be seen in the photo below. Following flow of air from left to right the components are: Air Filter, Pressure Regulator, Lubricator, and Control Relay. The components were fitted together and then sealed. A relay inside the control relay moves a piston that releases air to the cutter. The relay requires 24Vdc at approximately 150mA. Appendix 3 contains a section that describes how the relay controls air flow through the cutter assembly.



Figure 7: Pressurized Air Valve Assembly

The final picture in this section is a side view of the mechanical system fully assembled with the exception of the air valve assembly. Ribbon enters the machine on the left side. The white bracket carries the ribbon into cutter. The white bracket is actually two brackets, the side in the foreground overlaps the one in the back. The half in the foreground is mounted onto a metal insert which is inside the beam. By loosening the two screws it is possible to adjust the bracket to the width of the ribbon. It is important that the bracket be as narrow as the ribbon is wide because this keeps the pleated ribbon straight and increases the accuracy of the cut as well as prevent the ribbon from deviating to the side and causing a jam in the machine. The motor on the right side is the exit motor. This motor is used to clear away cut ribbon, and keep the uncut ribbon under some tension. Cut ribbon then falls away from the machine most likely into a basket.

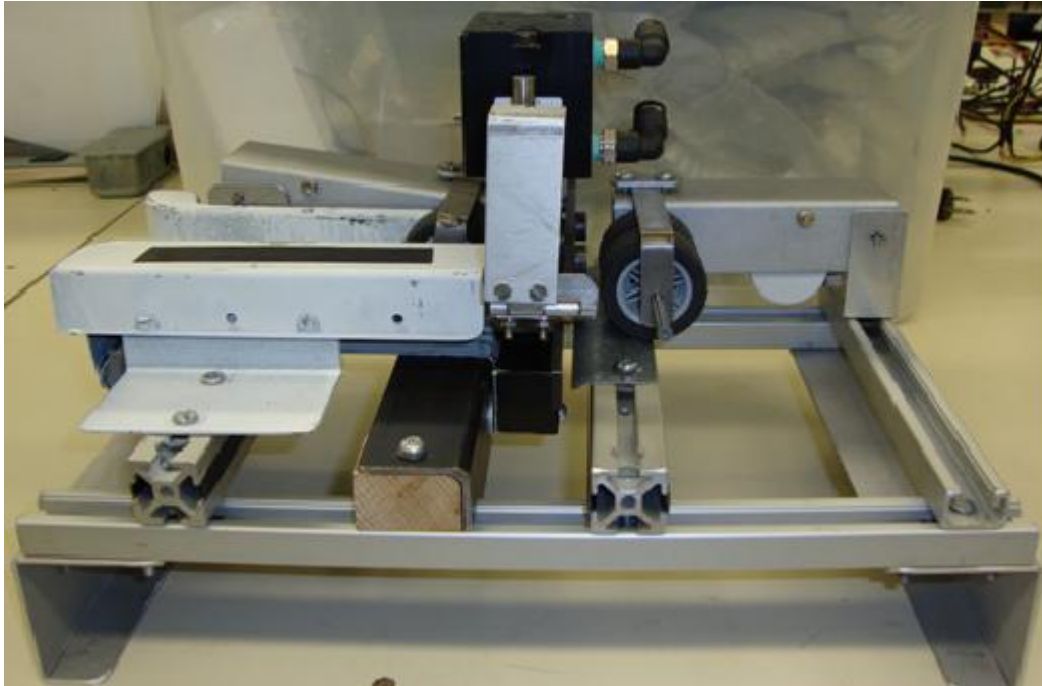


Figure 8: Ribbon Cutter Mechanical Systems

Hardware

The LEGO NXT Brick

The LEGO NXT Brick has three Motor control ports. These ports are primarily used drive 9V LEGO stepper motors. The LEGO NXT brick can control the acutely control the position of these stepper motors to within a degree. This accuracy allows us to cut the ribbons to exactly the correct length. The NXT brick has four general purpose I/O ports. In standard LEGO projects these ports are usually used for one sensor each.

These general purpose I/O pins can provide a current sink capable of switching fast enough to communicate using a slower than standard I^2C serial communication bus.

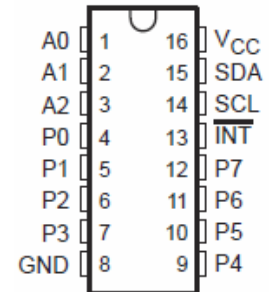
The NXT brick also has some limited interface abilities. The audio abilities of the NXT brick are extremely limited but could be used instead of the audio amplifier we designed in the event of a break down. The brick comes preloaded with an audio library but has a small low quality speaker and poor audio amplification. The bricks LCD screen is programmable



and will be used to show the speed of the motors while the ribbon cutter is running as well as allow for the selection of multiple programs.

I²C I/O Expander PCF8574

The PCF8574 I/O expander takes serial input from two lines and translates it into parallel. SDA is an I²C data line and SCL is an I²C clock line. The pins A0, A1, A2 are used to set the individual chips address, allowing the I²C to use up to eight PCF8574s. P0 through P7 are the parallel output ports. The I/O expander can sink up to 20mA of current but can only supply about 1mA. To use the I/O expander as an input, the ports are all set high and then read by the NXT brick. The I/O expander supports interrupt signals but because the NXT brick does not our design relies on polling to sense the input from I/O expanders.



Circuitry

Input

We used the PCF8474 I/O expander and momentary switches to control the operation of the ribbon cutting machine. The machine has eight momentary switches so it can all be controlled on one I/O expander. When the buttons are pressed it grounds a pin on the I/O expander. The I/O expander reads this ground as a logical low, and when the I/O expander is polled it replies with a binary number indicating which switches are low and which are high. The I/O expander has only a weak pull up so each pin is also connected to a five volt rail.

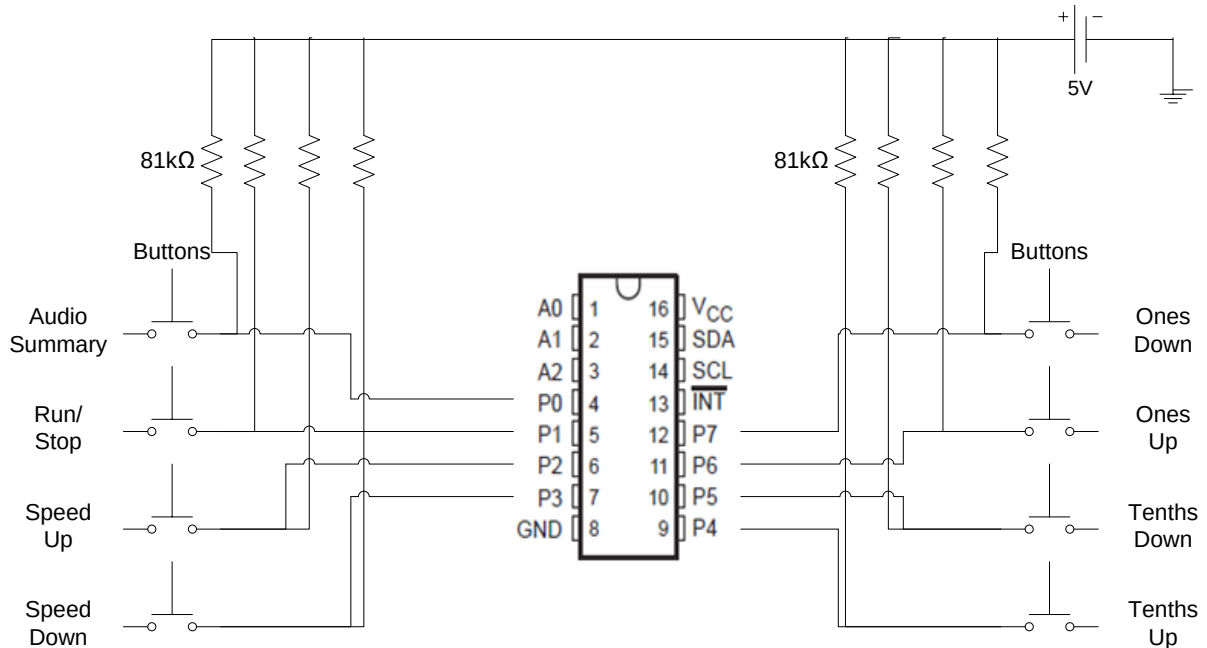


Figure 9: Button Schematic

Eight Segment LED Display

To display the desired cut length from we require three digits, this allows us to display numbers from 0.1" to 99.9". The easiest way to display these digits was a three digit eight segment display. Because using the NXT brick to write to data registries over I^2C proved difficult we used three PCF8574 I/O expanders instead of a dedicated LED driver like the SAA1064. The I/O expander can sink about 20mA of current but can only provide about 1mA, so we must use it as a current sink to drive LEDs. Using a common anode eight segment display we can power the LEDs off of the 5V rail and select which LEDs are on by outputting a zero or a one on an I/O expander. For reference see Appendix 3.

The top row of the table below contains the digits we wish to display. In order to display those digits the ports of the I/O expander must be set to high or low in the order described in the table.

Digit	0	1	2	3	4	5	6	7	8	9	.
Port 0	0	1	0	0	1	0	0	0	0	0	1
Port 1	0	0	0	0	0	1	1	0	0	0	1
Port 2	0	0	1	0	0	0	0	0	0	0	1

Port 3	0	1	0	0	1	0	0	1	0	1	1
Port 4	0	1	0	1	1	1	0	1	0	1	1
Port 5	0	1	1	1	0	0	0	1	0	0	1
Port 6	1	1	0	0	0	0	0	1	0	0	1
Port 7	1	1	1	1	1	1	1	1	1	1	0
Base 10	192	249	164	176	153	146	130	248	128	152	127

Figure 10: 7-Segment Addressing Table

Cutter Relay Control

The guillotine shears are controlled using a MOSFET transistor and another I/O expander as a switch. The I/O expander cannot reliably provide enough current to bias the MOSFET so the gate of the MOSFET is connected to the five volt rail through an 81k Ω Resistor, and the I/O expander is again used as a current sink. When the I/O expander is not sinking current the MOSFET allows a 24V signal through. This signal triggers a relay opening the pneumatic valve which closes the cutter. When the I/O expander is sinking current to ground the MOSFET cuts off the 24V signal and the pneumatic valve closes.

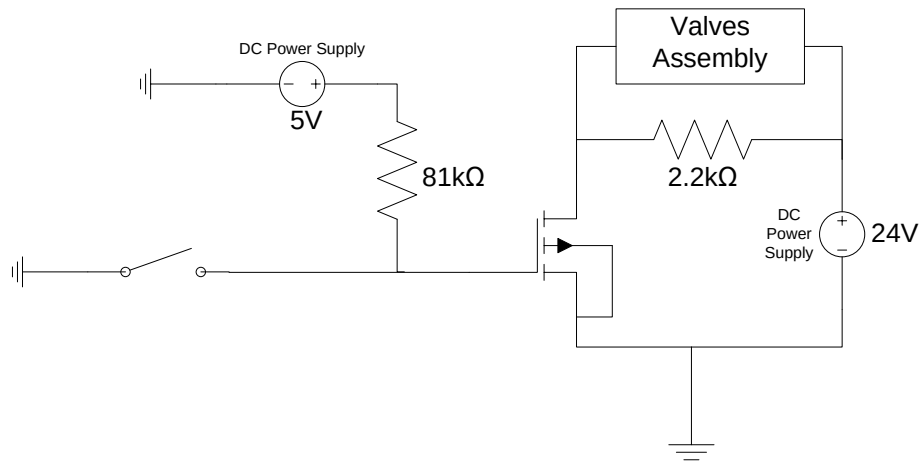


Figure 11: MOSFET Relay Schematic

NXT Power Supply

The NXT brick normally uses batteries to supply its power. To prevent the machine from requiring a change in batteries every few hours of operation, we designed a power supply for the brick. The brick requires 9V and the computer power supply we were using could provide 3.3V, 5V, 12V, and -12V. We used a LM7809 9V regulator to step the voltage

down from the 12V rail. Through testing we found that the brick used enough power to make the LM7809 quite hot. To reduce this heat issue we added a heat sink and further testing showed that the heat problem had been eliminated. The capacitors in the circuit are used to reduce voltage ripple.

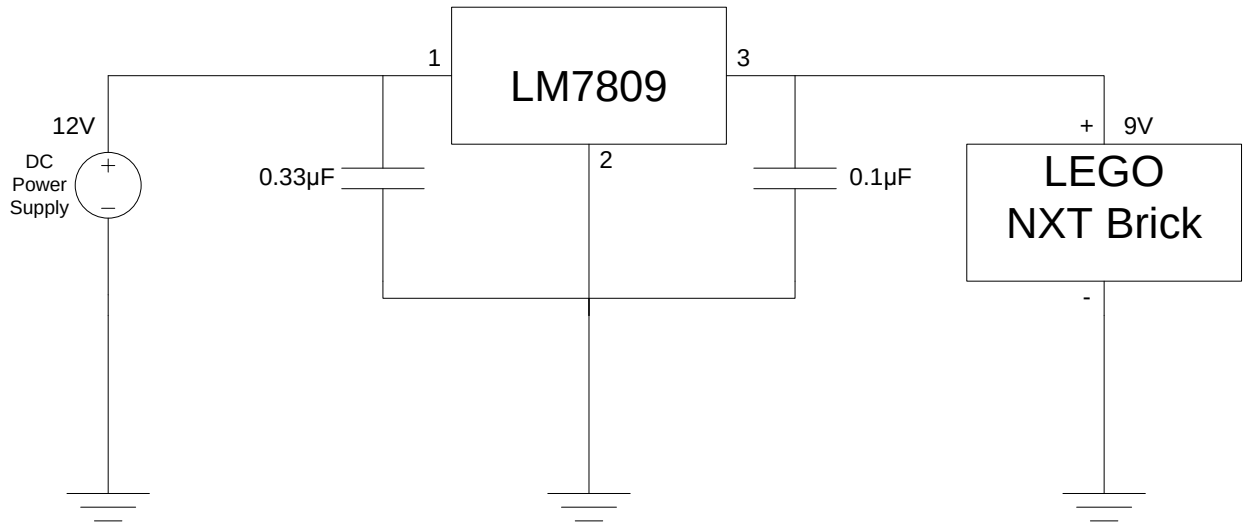


Figure 12: Power Supply Schematic

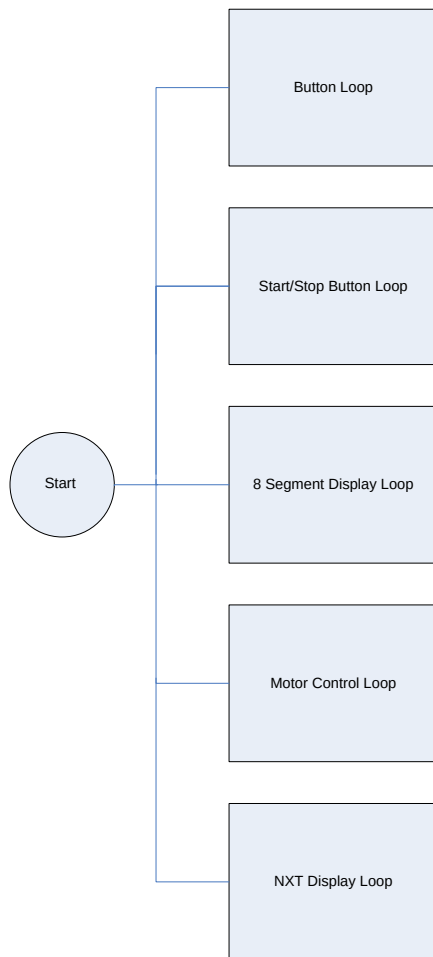
Programming

Software and Interface Design Requirements

The software is required to run the motors, cutter, display information about the length of the ribbon to be cut, and the current speed setting for the motors, provide audio feedback to the user, and accept input from the user and respond quickly. The program also must be able to run fast enough that it does not become a limiting factor in the output of the machine. One of the disadvantages of the 2008 machine is that timing loops were used to execute the next command in code. This was because there was no feedback in the 2008 machine to alter the microprocessors that a task had been completed and that the next command could be issued. The new code was expected to be more efficient by eliminating timing loops wherever possible in favor of an event-driven format that makes use of feedback to determine when the next command should be issued.

Programming Language

The programming language chosen to operate the machine was NXT-G. NXT-G is a graphical programming language; there is no scripted code only blocks that are connected in sequence. The Lego software includes a compiler. The brick executes commands in sequence and in parallel. NXT-G does not support interrupts. This created some challenges while programming because for safety the machine needs to be able to respond to a stop



command quickly if necessary. To overcome this polling was carried out through the use of loops and nested loops.

Every code in NXT-G starts at one starting point that then branches out into parallel sequences of commands. Functions that should be carried out simultaneously are placed in parallel. Functions that take place sequentially are in series. There are five lines of code that are placed in parallel with each other. Each line of code, represented as a box in the drawing left, is series of commands that are executed in sequence. The next in line event can never begin until the previous one has finished.

The line of code called button loop monitors the I2C data bus for data packets that indicate a button has been pressed by the user. Whenever a button is pressed by the user the I2C I/O expander associates the button with a binary number. There are eight buttons

used on the machine therefore each button is represented by one digit of an 8 digit binary number. For example, the button that decreases the length of the ribbon by one tenth of an inch is the least significant bit (LSB). Whenever this button is pressed the I2C I/O expander transmits a data packet that contains the number 254. This is because the binary digits are high by default a pressed button results in a zero. This code can only handle one button being pressed by the user at a time. If more than one button is pressed the data packet will contain a number that doesn't match any of the values associated with one

button being pressed, and the input will be ignored. The code matches the binary numbers to buttons through a switch block. Each “tab” or position that the switch can occupy is given the number associated with one button. When a binary number associated with a button is received through the I2C read block the switch moves to the position occupied by the code that carries out the functions of the button. Below is screenshot of the button loop. The loop is the orange box that surrounds the rest of the code. The loop is set to repeat itself an infinite number of times. Therefore, allowing polling of the I2C bus as long as the machine is turned on. The yellow block reads the I2C data bus. The position that the switch is set to increases the speed of the motor. It uses a variable named speed to contain the speed value of the motor, each time the button is pressed the speed variable is incremented and overwrites the previous speed value. The orange block with the hour glass is a wait block. It has been set to wait .080 seconds. This serves two purposes; first, it is a debouncing method, second it slows down the loop allowing a human to release the button. Whenever a button is pressed it causes some transience. Since the microprocessor executes commands in microseconds it means that one button press might be read as a series of on/off pulses. Adding the wait block freezes the loop long enough to allow the transience to end. Again, since the processing speed is so high a human cannot press and release a button fast enough that it only occurs within one command. This wait block is long enough to allow for a human to increment the speed by only one but also short enough that the program doesn’t seem unresponsive.

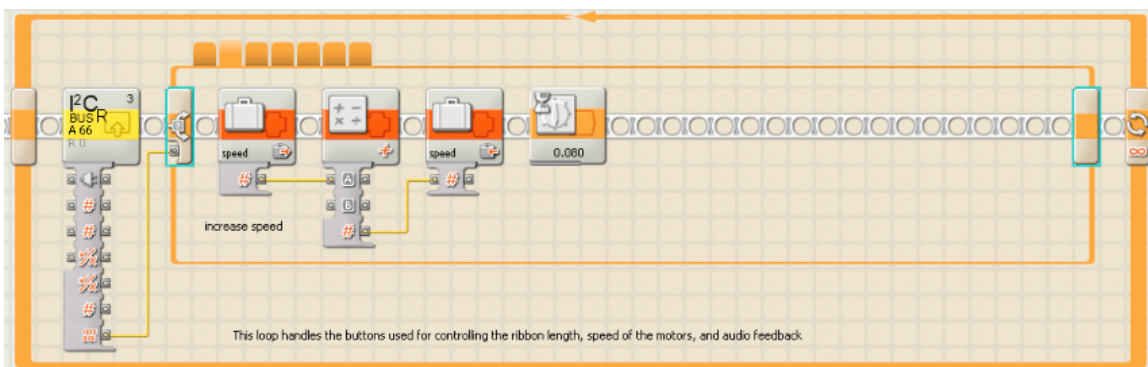


Figure 13: Button Loop

The Start/Stop button gets special treatment because by having its own devoted loop because it is an important safety feature that the machine stops running if commanded to. By having its own parallel loop the Start/Stop button still functions even if another

button's commands are still being executed in the button loop. For example, if the user presses the audio summary button the audio summary of the length will begin to play. This takes some time, and while the summary is being all the other buttons within the button loop would be unresponsive. However, since the Start/Stop button has its own loop it could still be used to stop the machine at any time. Below is a screen shot of the Start/Stop loop. Like the button loop there is a block for reading the I2C data bus and it is enclosed in a forever loop. The Start/Stop button, as the name implies, is used to both start and stop the machine. A XOR logic block is used to switch the Start/stop variable between a true or false value. This variable is later used in the motor control loop. There is a switch in this loop as well as another method of debouncing.

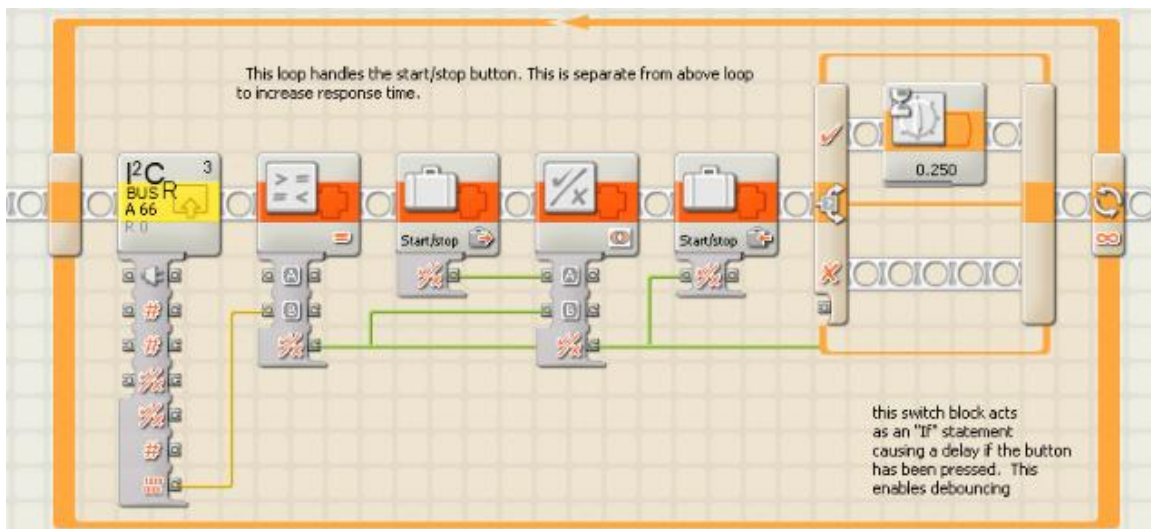


Figure 14: Start/Stop Loop

The next loop is the 8 segment display loop. This is the longest line of code in the program because there is higher amount of data processing that takes place. Since the code is so long its screen shot is featured in Appendix 3 under Program. Each of the three digits has its own address on the I2C data bus. Each digit is given its own switch that writes to the I2C bus binary values that correspond to LEDS within the 8 segment display. To do this first the length variable must have the tens, and ones digits separated. The tenths digit is separate already because of the button loop. Once the digits are separate they displayed. The audio summary functions similarly, expect it results the digit being sounded off instead of displayed.

The Motor Control loop monitors the speed, length, and Start/Stop variables and then controls the enter and exit motors accordingly. Below is a screen shot of the motor control loop. The loop begins by checking the Start/Stop variable. The loop will not progress until the value of the variable has been switched to True, otherwise the small loop the surrounds the variable will loop continuously. Next the program checks that both the length and speed variables are greater than zero. Finally the motor are given the speed and length instructions. The length variable contains a value in inches, so a unit conversion is performed by multiplying it with the number of degrees per inch of a roller. This instructs the servo motors how fast they are supposed to turn in relation to maximum power, and also how many degrees the rotor is supposed to turn. To improve accuracy of the cut being made the motors are also given the instruction to brake at the final position as opposed to coasting to a stop. The circumference of a roller is 5.3 inches; therefore one degree of rotation is .014 inches of ribbon being fed. This is well within the accuracy limits set by the design requirements.

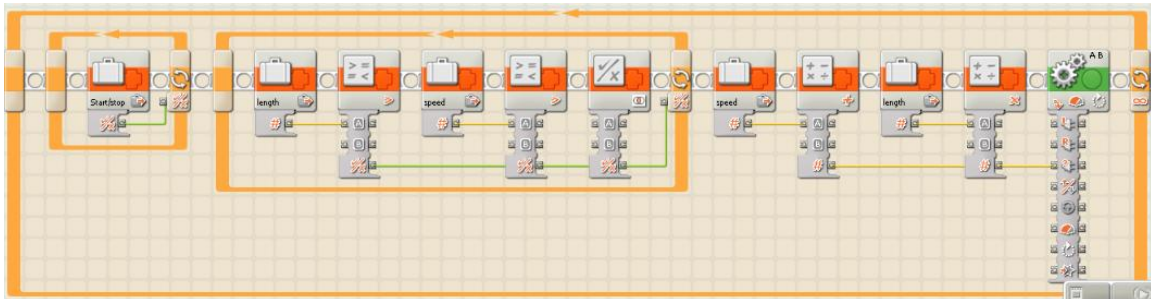


Figure 15: Motor Control Loop

The final loop was originally used for debugging purposes. It contains some dead-end code that doesn't affect the operation of the machine. It converts the length and speed variables to text and displays the text on the NXT brick's display. This means there is some redundancy in the machine because the length of the ribbon to be cut is displayed twice.

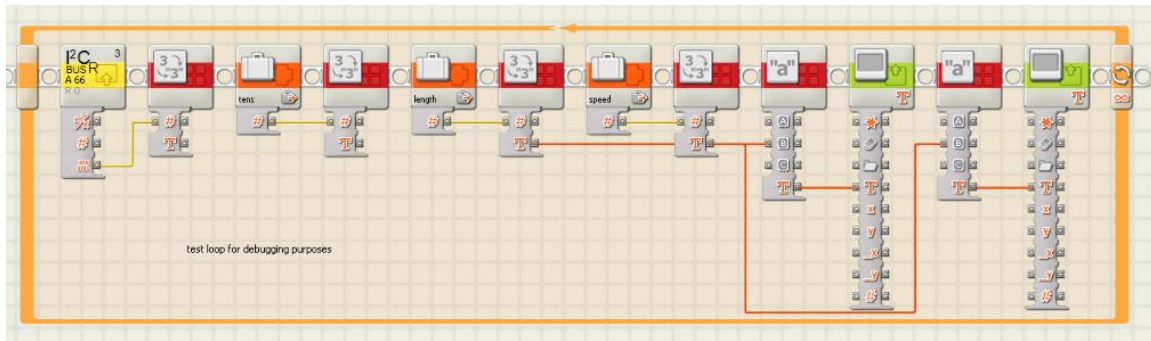


Figure 16: Display Loop

The biggest challenge in creating the program was designing around the lack of interrupts. This meant that loops had to be thought of that would carry out commands, and not get stuck when they are supposed to be carrying out a separate function. The microprocessor appears to carry parallel code segments simultaneously even though this is not really the case.

Chapter 4: Results and Test Data

Final Design

The new machine is 3.19 times faster than the old machine. The new design provides comparable average cutting accuracy and reduced average percent error. As can be seen from the displacement data the 2008 machine did have the ability to accurately extract ribbon to be cut. However, due to the machine's continuous drift problem the average accuracy increases over time. The new machine also continues to provide audio feedback when requested by the user and also displays the length of the ribbon to be cut. The customer requirement of having adjustable speed of operation has been completed. In addition the percent of maximum speed is displayed on the NXT brick's display. Though not requested the mechanical system of the 2010 design is 4.7 times smaller in area than the 2008 machine. The heat buildup as a result of the stepper motors is no longer an issue because of the use of the LEGO servo motors.

Test Results

The two most crucial attributes of a ribbon cutting machine are how fast it can cut ribbon and how accurately it can cut ribbon. Before any changes were made to the original machine two tests were done to benchmark the performance of the new design. For the first test the 2008 machine was allowed to run for one cycle at a time. No actual ribbon was cut during this test instead the displacement of the movable clamp and how far the clamp had drifted away from the stationary clamp was measured. The displacement was measured by using a marker placed on the movable clamp bar. During a cycle the movable clamp would push the marker till it reached its point of maximum displacement. The distance between the marker and the front of the movable clamp was then recorded. An illustration of the test setup is included in Appendix 3. One of the problems with the old machine was that the stepper motors would not always return the movable to its original position. As the drift increases the inaccuracy of the machine increased as well even though the machine had a fairly consistent displacement. Since the ribbon cutter is used in an industrial application this was a factor that needed improvement. During the 20 cycle test

the machine was never manually reset to its original starting position. For the 2010 machine the machine was allowed to run for 20 cuts. Since the machine uses capstans there is no drift factor. The tables below show the accuracy of both the 2008 machine and 2010 machine.

2008 Machine					
#	Expected (in.)	Displacement (in.)	Drift (in.)	Actual (in.)	% Error
1	5	4 15/16	0	4 15/16	1.25
2	5	4 29/32	0	4 29/32	1.875
3	6	5 27/32	5/8	6 15/32	-7.8125
4	6	5 1/2	11/16	6 3/16	-3.125
5	6	5 29/32	11/16	6 19/32	-9.895833333
6	6	5 5/8	11/16	6 5/16	-5.208333333
7	6	5 13/16	1	6 13/16	-13.54166667
8	6	5 15/16	1 1/16	7	-16.66666667
9	6	5 15/16	1 3/32	7 1/32	-17.1875
10	6	5 15/16	1 3/32	7 1/32	-17.1875
11	6	5 29/32	1 3/32	7	-16.66666667
12	6	5 15/16	1 1/8	7 1/16	-17.70833333
13	6	5 15/16	1 1/16	7	-16.66666667
14	6	5 15/16	1 3/32	7 1/32	-17.1875
15	6	5 15/16	1 3/32	7 1/32	-17.1875
16	6	5 29/32	1 3/32	7	-16.66666667
17	6	5 15/16	1 1/8	7 1/16	-17.70833333
18	6	5 15/16	1 1/8	7 1/16	-17.70833333
19	6	5 15/16	1 1/8	7 1/16	-17.70833333
20	6	5 15/16	1 3/32	7 1/32	-17.1875
Avg. Accuracy		98.01%		113.24%	
				Avg %Error	-12.99479167

Figure 17: Data for Previous Design

2010 Machine			
#	Expected (in.)	Actual (in.)	% Error
1	6	5 3/4	4.1666667
2	6	5 3/4	4.1666667
3	6	6	0
4	6	5 1/2	8.3333333
5	6	5 3/4	4.1666667
6	6	6	0
7	6	5	16.666667

8	6	6	0
9	6	5 3/4	4.1666667
10	6	6 1/4	-4.166667
11	6	6	0
12	6	6	0
13	6	6	0
14	6	6 1/2	-8.333333
15	6	6	0
16	6	6	0
17	6	6	0
18	6	6	0
19	6	6	0
20	6	6	0
Avg. Accuracy		98.54%	
		Avg %Error	2.7063

Figure 18: Data for New Design

The second test conducted was how long each machine would take to make 100 cuts. During this test both machines were allowed to run, without ribbon, through 100 cycles. The time was taken through the use of a stopwatch.

Time for 100 cycles	
2008	2010
7min 49.7 sec	2 min 27 seconds

Figure 19: Run Time Comparison

Sources of Error

The new design's accuracy is susceptible to the method that ribbon is being fed into the machine. In the 20 cut accuracy test the first 10 cuts made had the ribbon hanging off the table the machine was placed on top of. This meant that the machine had fight gravity in order to feed ribbon into the cutter. As the results show the increased load resulted in some slippage, which corresponds to decreased accuracy. The final 10 cuts made during the test had the ribbon placed on the table top where the machine was located. As the data indicates, this proved to be a successful means of improving accuracy.

Chapter 5: Conclusion

The goal of this semester was to improve the qualities and efficiency of ribbon cutting machine. Design Team 2 has successfully accomplished this task. The customer requirement of the machine being operated only by LEGO MINDSTORMS NXT has been accomplished. The machine is easily reprogrammed and maintained by high school students with a familiarity to LEGO robotics. A variable speed controller has been added to control the LEGO servo motors. The new design has increased the maximum speed of operation of the machine. The size of the machine has been reduced. Since the step motors and electric cutter have been removed there is little noise generated by the machine. The machine now can produce ribbon with more accurate length and consistency. The new machine is also safer than the old machine because we have added covers to the cutter to make sure when the machine is operating there is no danger to the user.

One major difficulty that was encountered was due to the material the ribbons are made from, it is very compressible so it is very hard to get exact length that user wants. Also, the ribbon has a tendency to curve this caused problems with jamming the machine because of the ribbon becoming stuck. The solution was to use brackets to guide the ribbon through the machine, and build a guard to prevent ribbon from becoming caught inside the cutter.

Throughout the semester the group used a Gantt chart to monitor progress. The first two weeks were spent planning, discussing the design objectives, and researching. By the third week, all the team was better able to understand the project and goals that need to be achieved by the end of the semester. By the end of fourth week, all the functions of the old machine such as speed, length and consistency had been tested. The team then assigned tasks for each team member to be completed.

The overall cost of the machine is listed below. The LEGO MINDSTORMS NXT was provided by our sponsor, and so the total cost to the team is \$518.16, most of the money was spent to purchase the Guillotine Knife. The problem with the old cutter is that it is very noise during operating and it created a large gap in the bracket system of the new design. The main advantage of using guillotine knife is that the user will not have the ability to touch the blade while the machine is operating. It is also a powerful and reliable cutter. The

knife is widely used for cutting various strips of flexible materials. The cutter also has a long blade life because it is made of hardened 0-1 tool steel.

Part	Number	Cost
Guillotine Knife	1	\$358.16
IC LED Driver	1	\$2.80
LED Display	1	\$1.67
4 Digit LED Driver with I2C Bus Interface	1	\$3.20
LED Driver w/I2C	1	\$1.54
0.33uF Capacitor	1	\$0.83
A Voltage Regulator	1	\$0.70
I ² C Interface 8 bit I/O Expander	14	\$25.62
Tubing	1	\$18.75
Valve	1	\$19.75
Filter	1	\$11.25
Regulator	1	\$17.00
Pneumatic Lubricator	1	\$12.25
Male Straight1/4NPT*1/4 Inch	1	\$4.50
Male Straight1/8NPT*1/4 Inch	2	\$5.00
Male Elbow1/8NPT*1/4 Inch	2	\$12.00
Total Shipping		\$32.64
Lego Mindstorms NXT 2.0*	1	\$299.99*
Total Cost		\$831.15

Figure 20: Finalized Budget

*Provided by Sponsor

Appendix 1: Technical Roles

Nicholas Neumann



During the project I was put in charge of the programming and implementing our sensors and buttons during the testing phase. Once we came up with an improved design however I was also put in charge of developing an LED circuit since we would no longer use the existing one. Once testing was completed Jon took over the programming. The programming itself wasn't too complicated, while loops in place the rest of the programming was trivial and mostly consisted of making sure the length adjustments were in place for the motors and that all the I/O expanders were addressed correctly.

In the end we had no need for sensors so those circuits were scrapped. This was because the capstand feed design we came up with no longer needed sensors to detect length since the LEGO motors are accurate enough that it can be programmed in. The buttons were easily implemented using the I/O expanders and I²C programming blocks.

My next step in the design process was designing the Seven-Segment LED Display circuit. This proved to be a bit troublesome though. We had originally intended to use a driver for our seven segment displays that was compatible with I²C, but when trying to program the driver in LEGO the registry values did not match up with those given on the data sheet. While this problem with time could have possibly been debugged I decided not to waste any more time on it and used the I/O expanders to implement it instead. This required 3 I/O expander chips since one was needed for each digit of our display. After wiring I addressed each of the chips and began testing. Afterwards it was programmed in with the I²C programming blocks. Even with some of the setbacks, these tasks were all completed

in a reasonable amount time.

Ralph Prewett



I was responsible for the design and implementation of the input circuitry, eight segment display, and I²C bus. For the input circuitry I chose the switches and momentary switches we were using, designed and tested circuitry to read the switches and communicate with the NXT brick. For the eight segments display I chose the components and designed both the original SAA1064 based driver and the I/O expander based driver. I also designed and implemented the I²C bus, sourcing compatible components and fixing any problems that come up throughout the semester. Making sure that the I²C worked was a challenge because the NXT brick has such a limited programming interface and using a more complicated text based programming environment would have negated the value of NXT brick as an easy to maintain system. This prevented me from using circuits with complicated ICs that use multiple registries. I also wrote example code for all the circuits I designed in order to clear up any potential addressing issues and to provide the team's programmer with a working black box circuit, so that the programming part of the project could be done without the programmer having to worry about minutia like individual ICs pin out diagrams.

I also spent significant time helping other team members with their parts of the project. I redesigned and worked on the 24V relay switch that controls the pneumatic cutter. I also helped Li with the audio circuit and Felix with the power supply and box. I was responsible for the lay out of the final circuitry in the control box. I designed the wiring in the box to allow the lid to be unplugged from the boxes base. I contributed to the teams design for the mechanical portion of the machine as well. I added some details regarding the

feed mechanism, as well as several ideas that were only implemented in some earlier prototypes of the cutter such as using a pair of scissors as a sheer cutter.

Jonathan Brouker



My technical contributions to the team began with me trying to determine the specifications of the motors that would be used to replace the stepper motors in the original 2008 design. I began by taking measurements of the torque and power that would be needed to move the movable clamp system at a speed that would allow for improved performance. I also did research on the possibility of building servo motors using the NXT LEGO servo motor's

rotary encoder and larger, more powerful DC motors. I prototyped the new design for the ribbon cutter using LEGO bricks.

After prototype demonstrated that the new design was feasible I did research to determine which type of cutter would be best suited for the machine based on its ruggedness, and space and power requirements. I assisted Mr. Blosser with the fabrication of metal parts used in the assembly of the machine. After all the necessary parts were fabricated I assembled the machine with Mr. Blosser's help. I then conducted tests on the new machine to verify that the ribbon would pass through the machine consistently without jamming. The machine did not initially succeed in passing ribbon through because the ribbon would get stuck on the sharp edge of the blade. Therefore Mr. Blosser, Felix, and I worked to build a guard that would compress the ribbon to flow through the cutter, and prevent the source of jamming. This guard allows the machine to consistently pass ribbon without jamming. My final contribution to the project was the NXT-G program that controls the entire machine.

I solely designed, and tested the program. This meant that I had to learn the NXT-G programming environment. I had to devise a method for controlling the machine without

the use of interrupts. The solution to this was use of loops that constantly poll the I2C bus for changes. I also had to devise debouncing methods to ensure that the buttons functioned as expected. The rest of the program included data management methods for communicating ribbon length and motor speed, processing 8 segment display commands, and activating the cut sequence after the motors have stopped.

Felix Adisaputra

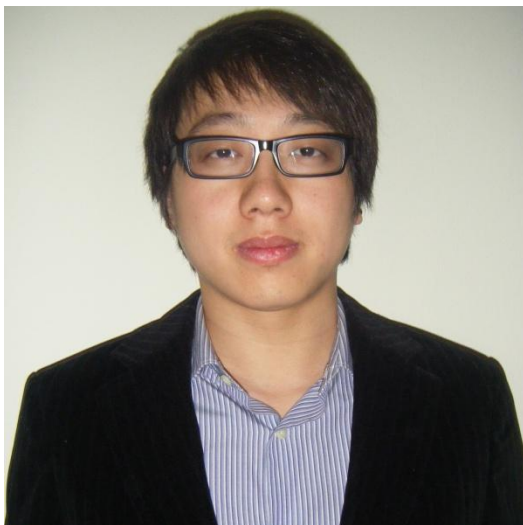


My technical role in this project is to build all of the power supply circuits that are required for the overall machine to run. In addition to those circuits, I was also in charge to replace the old controller box with a newly designed controller box and laying out all components to be fit inside the new controller box.

Throughout the project design and implementation process, I had been able to build the power supply circuits for the whole system successfully. The first power circuit that I built was a Buck converter circuit that steps down 12V DC input from the computer power supply source to a constant 9V DC output voltage to power up the LEGO NXT Brick continuously. This step- down converter circuit used the LM7809 voltage regulator and an appropriate heatsink was attached to the voltage regulator to eliminate the heat issue. Some capacitors were also used in this circuit. The LEGO NXT Brick would also supply power directly to both servomotors in the machine. I helped building the power circuit for the valves assembly of the guillotine cutter as well. This circuit required a MOSFET in order to supply 24V DC pulse in order for the valves assembly to run. The MOSFET acted like a switch to control the cutter to go up and down. The MOSFET then would be controlled by the LEGO NXT Brick through the I/O expander chip. The rest of the circuitry in the system would then utilize the 5V DC provided by the computer power supply directly.

The new controller box was designed to provide space for all the controlling components of the system except the cutting machine. The computer power supply, all the circuits, LEGO NXT Brick, valves assembly, LED display, speaker, buttons and switch would be placed in the new controller box. Some of them would be visible from outside the box, and the rest would be set inside the box that could be opened easily. The required drilling several holes on the box for screws and mounting some components on the box had been successfully implemented by me. Furthermore, during the project, I was requested to take charge of building the safety guard for the cutting machine. As a result, I successfully built all of the safety guards for the cutting machine in order to prevent the users placing their fingers inside the cutter while the machine is still operating.

Li Tian



My technical responsibility was testing the audio recording circuit which was used by the previous cutting machine. We could like to use LEGO MINDSTORMS NTX to control the whole machine and it could send low or high signal to different pins. The voice recording chip we use is ISD 1400p which has 128 addresses and each of them could record 20 second duration. Unfortunately, after we removed the chip, the voice files were erased. I was responsible to test the chip and rerecord the voice file. There is an application circuit (Appendix 3 audio circuit 2) which could use to record voice file. According to the data sheet, if either or both of the two Most Significant Bits of the address (A7, A6) are low, the all inputs are interpreted as address bits. And during playing, the recording pin (pin27) has to connect to high and either of Playback Level-Activated (pin 23) and Playback Edge-Activated (pin24) and to connect to ground. A LED diode indicates that recording is in process. However, after I successfully test the circuit, I realize that there are many memory addresses had been corrupted. I could only hear noise when those addresses reached. However, for our project in particular, we only need minimum 11 addresses which would store numbers

from 0-9 and one word which is “dot”. The reason we need this word is because that we want to do tenth of inch accuracy for the machine. So my next task was to test and find memory addresses which works fine. I developed a 4 Bit truth table which uses the 4 least significant bits and it will have 16 addresses, and set other 4 bits low. I realized the even I recorded different numbers in different addresses, they played the same number which I stored at last and many addresses only play noise and 1111 is not recording. It shows only 1 memory address had been used and while I was recording, it just replaced previous memory. Truth table 1 shows the recorded inputs and results. I could do another test which uses the four Most Significant Bits and it gives me better results but i still did not get the result which I want. If I rerecorded at one address, the voice stored at next address will be erased. So only 5 addresses I can use to store numbers. I set A3 high for next try but the result is I corrupted all the memory address except for all zeros. I can only hear output when all the inputs set to zero and noise when other address reached.

Thus, I went to solution two which is another circuit design (Appendix 3 audio circuit 2) it allows me to save multiply message in one address in 20 seconds. A single momentary press of Playback Edge-Activated (pin24) will play the first message and the second press will play the second message. However, the message stored in the address could only go forward order which means if I save 0-9, the first time I press Playback Edge-Activated, I could hear 0 and the next press, and I will hear 1. It could not achieve reverse order. Another problem is that I cannot use a different address to save the word “Dot”.

After many tests and verifies, I could not successfully build the audio system and fortunately my teammates find there is a build in speak in LEGO brick, we could use it and speak the numbers (0-9) and the word “dot.”

A0	A1	A2	A3	Recorded Number	Output	Correct Output
0	0	0	0	0	14	0
0	0	0	1	1	14	1
0	0	1	0	2	14	2
0	0	1	1	3	14	3
0	1	0	0	4	Noise	4

0	1	0	1	5	Noise	5
0	1	1	0	6	Noise	6
0	1	1	1	7	Noise	7
1	0	0	0	8	Noise	8
1	0	0	1	9	Noise	9
1	0	1	0	10	Noise	10
1	0	1	1	11	Noise	11
1	1	0	0	12	14	12
1	1	0	1	13	14	13
1	1	1	0	14	14	14
1	1	1	1	Not Recording	NA	

Figure 21: Truth Table 1

A4	A5	A6	A7	Recorded Number	Output	Correct Output
0	0	0	0	0	0, 4, 7	0
0	0	0	1	1	1	1
0	0	1	0	2	2,5, 9	2
0	0	1	1	3	3, 6,10	3
0	1	0	0	4	4, 7	4
0	1	0	1	Not Recording	NA	
0	1	1	0	5	5,9	5
0	1	1	1	6	6,10	6
1	0	0	0	7	7	7
1	0	0	1	8	8	8
1	0	1	0	9	9	9
1	0	1	1	10	10	10
1	1	0	0	11	Noise	11
1	1	0	1	12	Noise	12
1	1	1	0	13	Noise	13
1	1	1	1	14	Noise	14

Figure 22: Truth Table 2

Appendix 2: References

References

Gasperi, Michael, and Philippe Hurbain. *Extreme NXT Extending the LEGO MINDSTORMS NXT to the Next Level*. 2nd. New York: Apress L.P., 2009. Print.

Datasheets

Winbond Electronics Corp., 16 Nov, 2005. Web. 7 Nov 2010. <<http://www.winbond-usa.com>>.

"14mm Three Digit Numeric Display." *Datasheet Catalog*. N.p., n.d. Web. <www.DatasheetCatalog.com>.

"Remote 8-bit I/O Expander for I²C Bus." *Datasheet Catalog*. Texas Instruments, July 2001. Web. <www.datasheetcatalog.com>.

PCF8574 Remote 8-Bit I/O Expander for I²C Bus Datasheet. Texas Interments, Revised January. 2004

IRF1404 HEXFET Power MOSFET Datasheet. International Rectifier October 2003

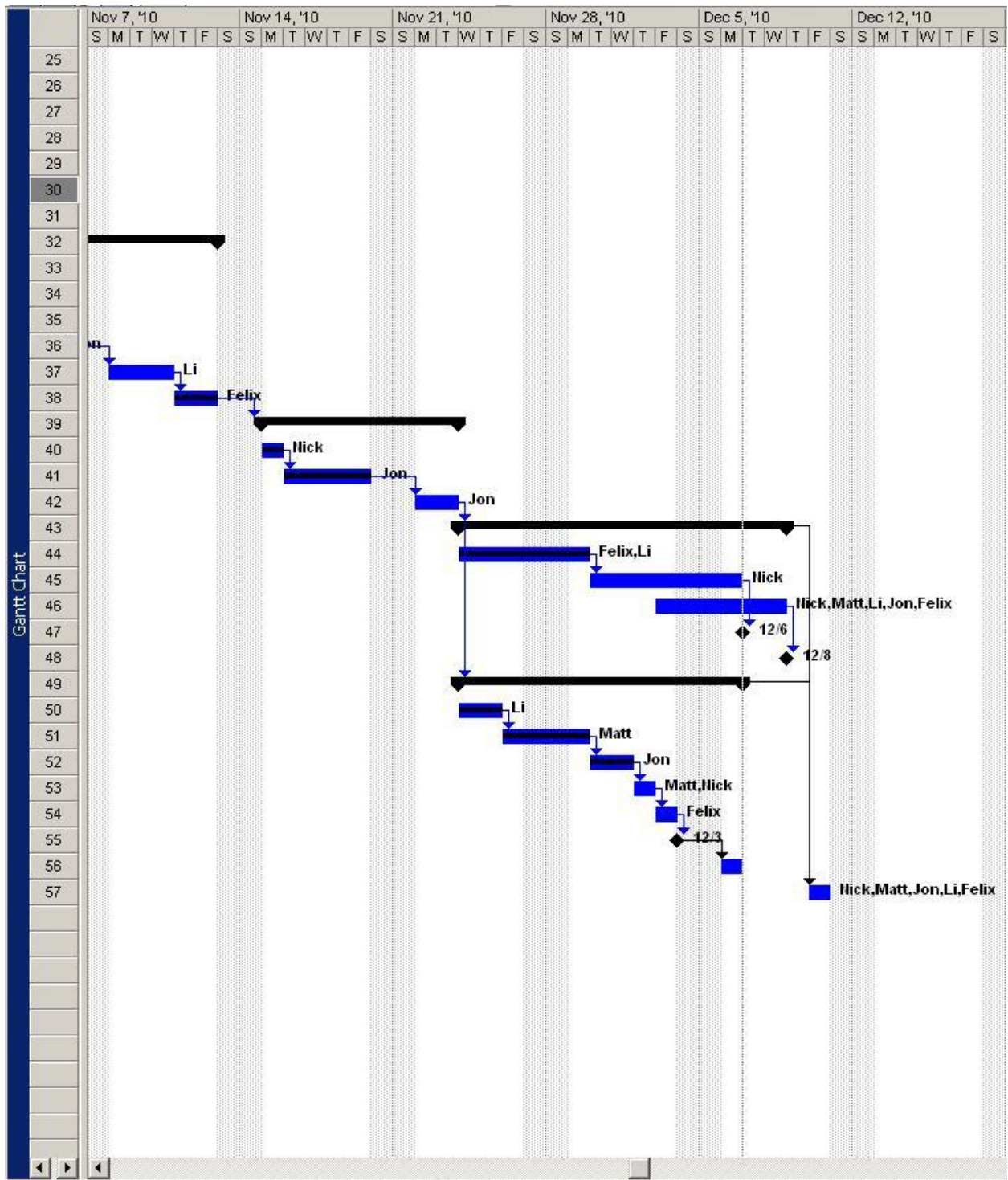
SAA1064 4 Digit LED Driver with I²C Bus Interface. Philips Semiconductors February 1991

LM7809 3-Terminal 1A Positive Voltage Regulator Datasheet. Unisonic Technologies.

Appendix 3: Technical Attachments

Gantt Chart

		Task Name	Duration	Start	Finish	Predecessors	Resource Names
1		Project Definition	4 days	Tue 9/14/10	Fri 9/17/10		b
2	✓	Initial meeting with Sp	1 day	Tue 9/14/10	Tue 9/14/10		
3	✓	Explore and discuss c	2 days	Wed 9/15/10	Thu 9/16/10	2	
4	✓	Initial meeting with Fa	1 day	Fri 9/17/10	Fri 9/17/10	3	
5	✓	Planning done	0 days	Fri 9/17/10	Fri 9/17/10	4	
6	✓	Research possible inte	6 days	Mon 9/20/10	Mon 9/27/10	5	
7	✓	LEGO Mindstorm	2 days	Mon 9/20/10	Tue 9/21/10		Jon
8	✓	LabVIEW	1 day	Wed 9/22/10	Wed 9/22/10	7	Matt,Nick
9	✓	Robot C	2 days	Thu 9/23/10	Fri 9/24/10	8	Felix, Li
10	✓	Pick Interface	0 days	Mon 9/27/10	Mon 9/27/10	9	
11	✓	Aquired Macine	0 days	Mon 9/27/10	Mon 9/27/10		
12	✓	Testing Phase	11 days	Tue 9/28/10	Tue 10/12/10	11	
13	✓	Test Run & Diagnostic	2 days	Tue 9/28/10	Wed 9/29/10		Nick,Matt,Felix, Li, Jon
14	✓	Begin Proposal	2 days	Thu 9/30/10	Fri 10/1/10	13	Jon,Felix
15	✓	Submit Pre-proposal	0 days	Fri 10/1/10	Fri 10/1/10	14	Jon
16	✓	Perchase LEGO NXP	4 days	Fri 10/1/10	Wed 10/6/10		
17	✓	Test LEGO Motors	2 days	Thu 10/7/10	Fri 10/8/10	16	Jon,Nick,Matt,Felix, Li
18	✓	Order Rollers & Touch	2 days	Fri 10/8/10	Mon 10/11/10		Jon
19	✓	Test new rollers	1 day	Tue 10/12/10	Tue 10/12/10	18	Jon
20	✓	Begin Preliminary Inter	3 days	Mon 10/11/10	Wed 10/13/10	17	
21	✓	Learn LEGO Mindstor	1 day	Mon 10/11/10	Mon 10/11/10		Matt, Jon,Nick, Li, Felix
22	✓	Check communication	2 days	Tue 10/12/10	Wed 10/13/10	21	Matt,Nick
23	✓	Presentation & Proposa	14 days	Mon 10/4/10	Thu 10/21/10	14	
24	✓	Prepare slides based	1 day	Mon 10/4/10	Mon 10/4/10		Felix
25	✓	Feedback on Pre-prop	1 day	Mon 10/11/10	Mon 10/11/10	24	
26	✓	Practice Presentation	3 days	Wed 10/13/10	Fri 10/15/10	25	Nick,Matt, Li, Jon, Felix
27	✓	Give Presentation	0 days	Mon 10/18/10	Mon 10/18/10	26	Nick,Matt, Li, Jon, Felix
28	✓	Finish Proposal	3 days	Tue 10/19/10	Thu 10/21/10	27	Felix,Matt, Li, Jon, Nick
29	✓	Final Proposal	0 days	Thu 10/21/10	Thu 10/21/10	28	
30	✓	Work on Brochure	2 days	Tue 10/12/10	Wed 10/13/10		Matt
31	✓	Turn in brochure	0 days	Fri 10/15/10	Fri 10/15/10	30	Matt
32		Begin Project Build	15 days	Mon 10/25/10	Fri 11/12/10	29	
33	✓	Replace PIC with NXT	3 days	Mon 10/25/10	Wed 10/27/10		Matt,Nick
34	✓	Build Prototype Design	3 days	Thu 10/28/10	Mon 11/1/10	33	Jon,Nick
35	✓	Order New Guillotine	2 days	Tue 11/2/10	Wed 11/3/10	34	Li
36	✓	Install and Test Cutter	2 days	Thu 11/4/10	Fri 11/5/10	35	Jon
37		Audio Circuit	3 days	Mon 11/8/10	Wed 11/10/10	36	Li
38	✓	Power Supply and Ne	2 days	Thu 11/11/10	Fri 11/12/10	37	Felix
39		Final Programming	7 days	Mon 11/15/10	Tue 11/23/10	38	
40	✓	LED Circuit	1 day	Mon 11/15/10	Mon 11/15/10		Nick
41	✓	Motors and Drivers	4 days	Tue 11/16/10	Fri 11/19/10	40	Jon
42		Cutter Activation	2 days	Mon 11/22/10	Tue 11/23/10	41	Jon
43		Finishing Touches	11 days	Wed 11/24/10	Wed 12/8/10	42	



Mechanical Systems

The Control Relay of the valve assembly has two modes of operation that are used. In both cases only the pressurized port designated as “P” in the schematic below is

connected to pressurized air. The other two ports, “R” and “S”, are left open to the atmosphere and are used to release air from the cutter. One of the modes shown below is labeled Open Cutter this is the default mode for the cutter assembly because the relay is spring return and therefore always returns to this position when 24Vdc is not being applied to the relay. When 24Vdc is applied to the relay it switches to cut mode this directs pressurized air on top of the piston which forces the blade down. During this mode port “A” releases pressure from inside the piston housing through port “R”.

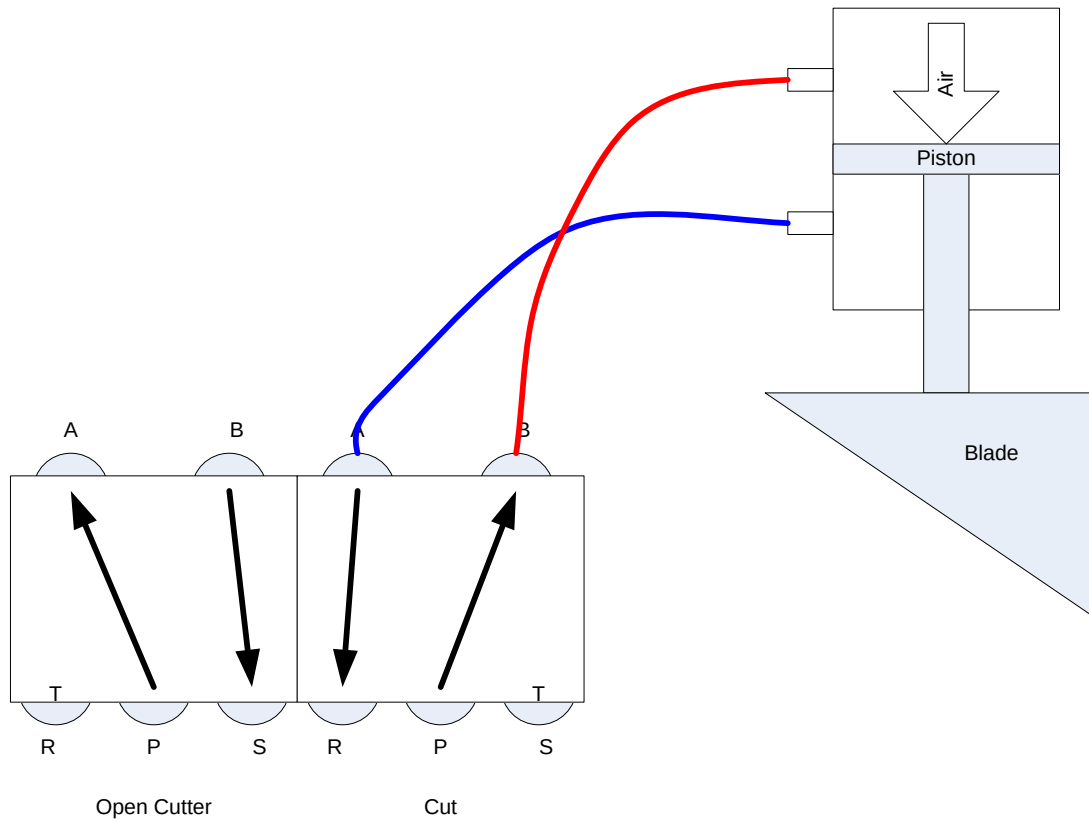
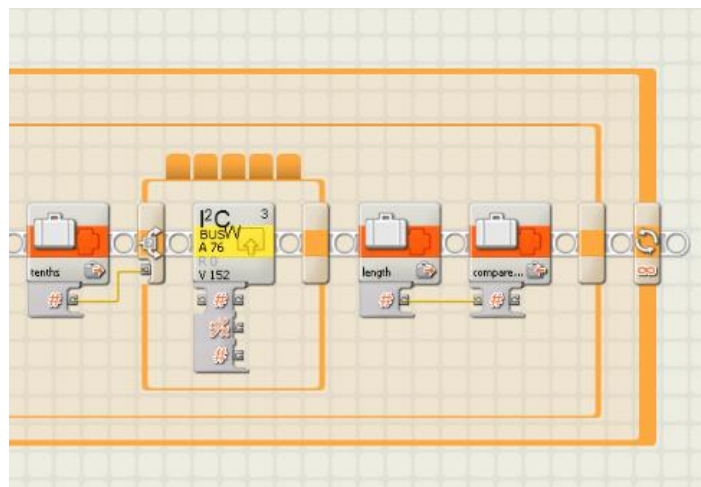
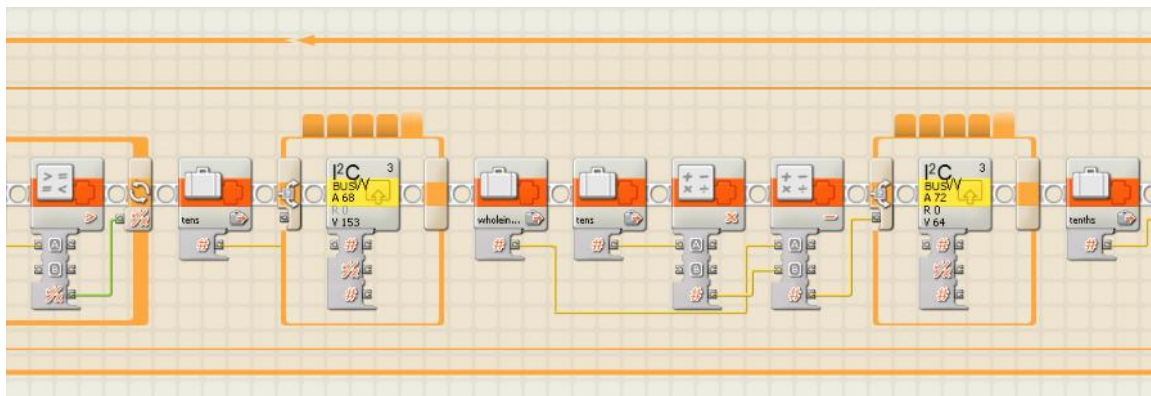
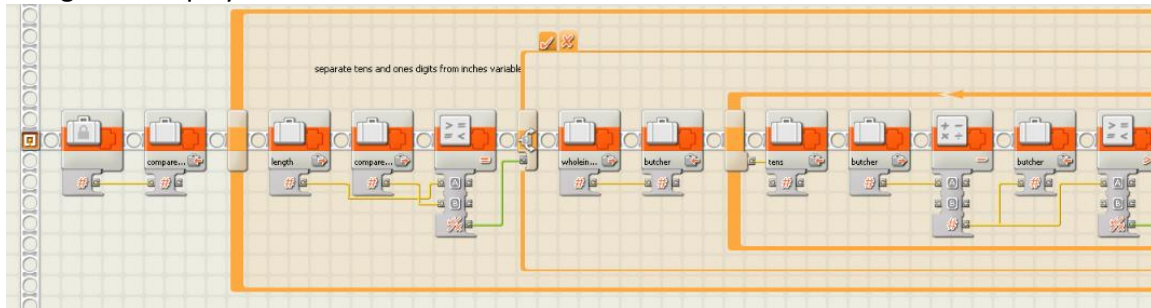


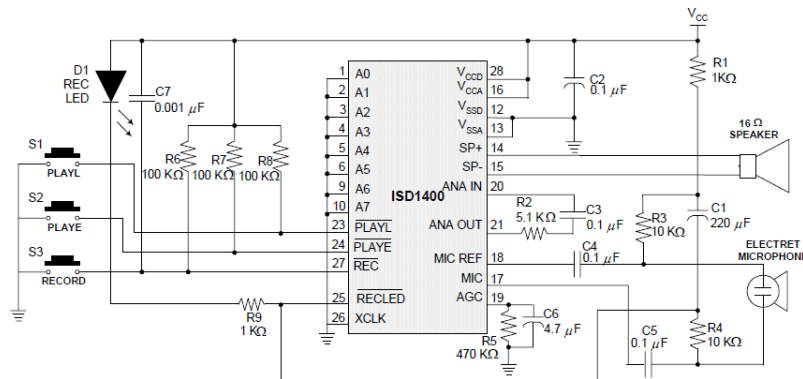
Figure 23: Air Flow Schematic

Program

7-Segment Display Driver Screenshot

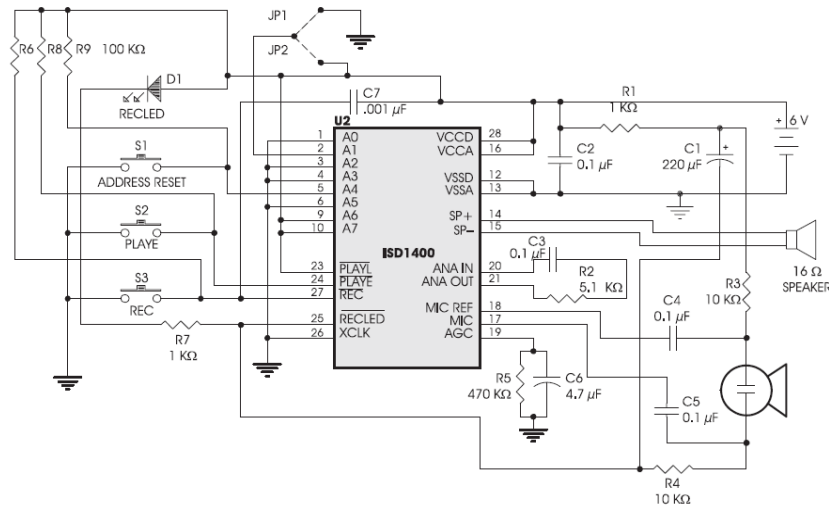


Audio Circuit 1:



When REC is low, recording initiate. Either Edge Activated playback or Level-activated playback low, message will be played while REC must go high.

Audio Circuit2:



Since A7 and A6 going to high, special operation take place. The recording sequence is same as Audio circuit 1. When the record sequence is complete, a single momentary press of the PLAYE button will play the first message stored and a second Press of the button will play the second message stored. Messages will continue to playback in sequence.

Average Percent Accuracy Formula

$$\frac{\text{Sum of Expected Ribbon Lengths}}{\text{Sum of Actual Ribbon Lengths}} * 100$$

Percent Error Formula:

$$\frac{\text{Expected} - \text{Actual}}{\text{Expected}} * 100$$

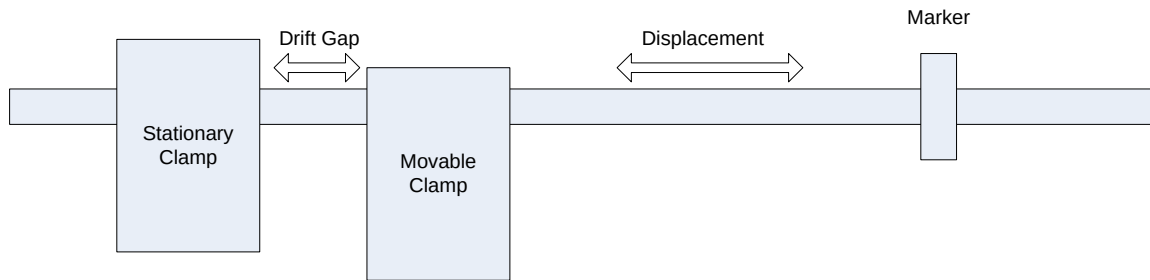


Figure 24: Test setup of 2008 Displacement and Drift Test

LED Schematic

