

# Automated Can Chiller



Brian Yount  
Jonathan Rogers

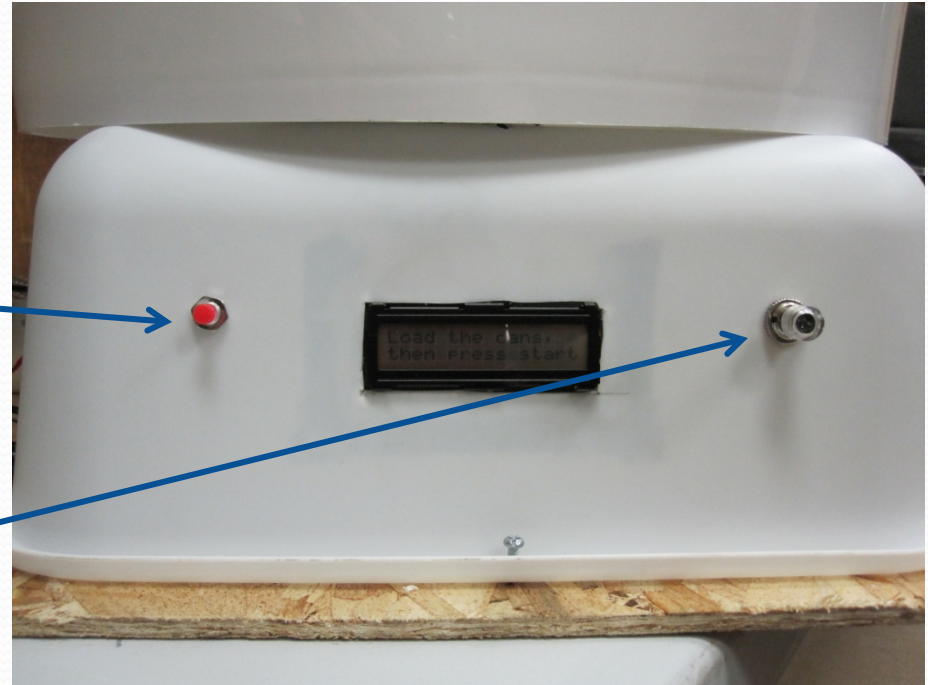
# Project Objective

- To quickly chill a 12 oz. can at the press of a button.



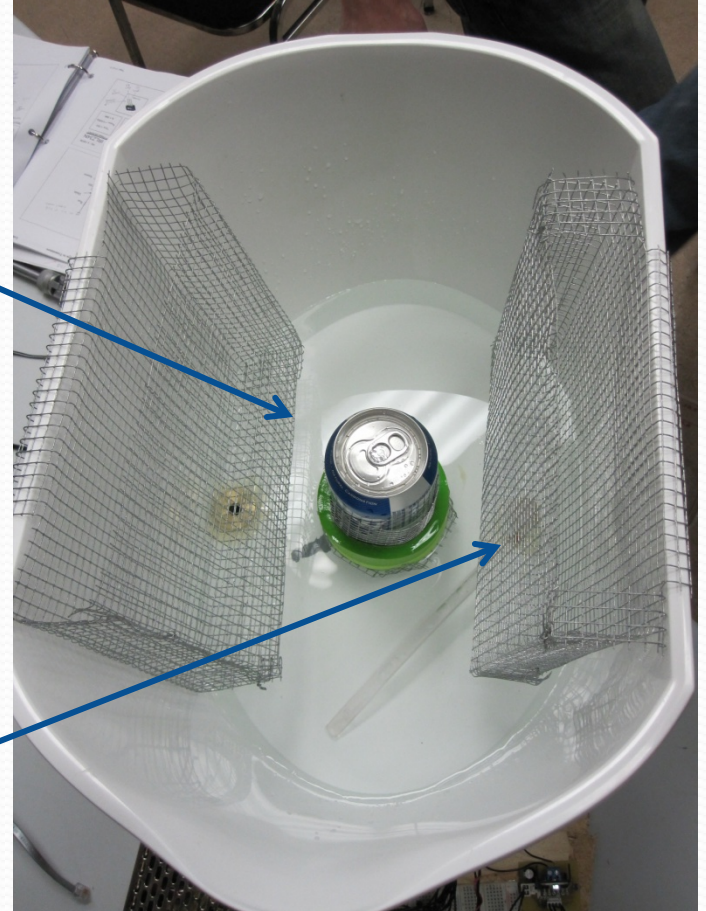
# Inputs

- Two Buttons
  - Momentary-Closed Push Button
    - Used to cycle through different LCD displays.
  - Toggle Button
    - Used as a master On/Off switch for chilling.
- Air Temperature
  - Sensed by Thermistor
    - Displayed on one of the LCD display options.



# Outputs

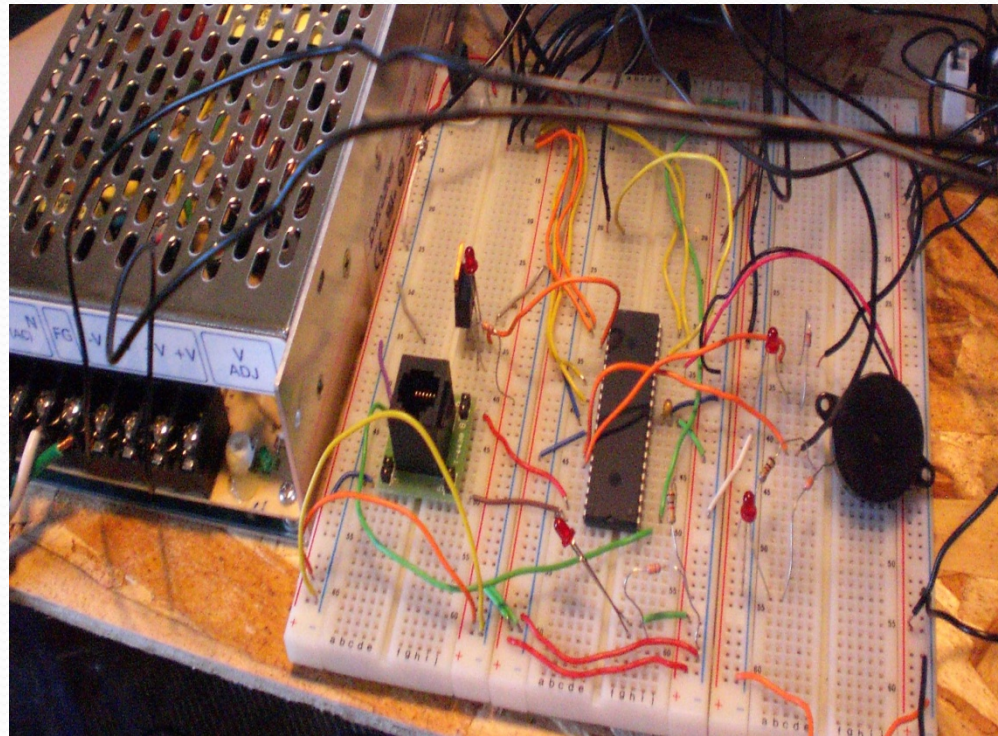
- Spinning Can
  - Spun by 12V Motor
  - Run by a PWM signal from PIC
- LCD Display
  - Used combination of delays, interrupts, and timers to display one of 4 displays
    - Air Temperature
    - Chill Time Remaining
    - Instructions to User
    - Creators (Brian and Jonathan)
- Circulate Water
  - Pump
    - Run by PWM Signal from PIC





# PIC Controller

- Used a p18f4520 PIC Controller
  - Delays
    - Used to overcome button-bounce
  - Timers
    - Used to run motor for a specified amount of time
    - Used for PWM
  - PWM
    - Used for Motor and Pump Control
  - Interrupts
    - Used to sense button presses & timer overflows.
  - Function to change display
    - Displayed LCD Contents based on a global variable that could be changed by a button press (interrupt).



# Advice

- Get started early!
- Test all electronics when you get them, just to make sure they work!
- Work out a plan early for the 'look' of the project
- Code and test individual items first, then put it all together.
- Don't make it too complicated.



# Automated Card Dealer

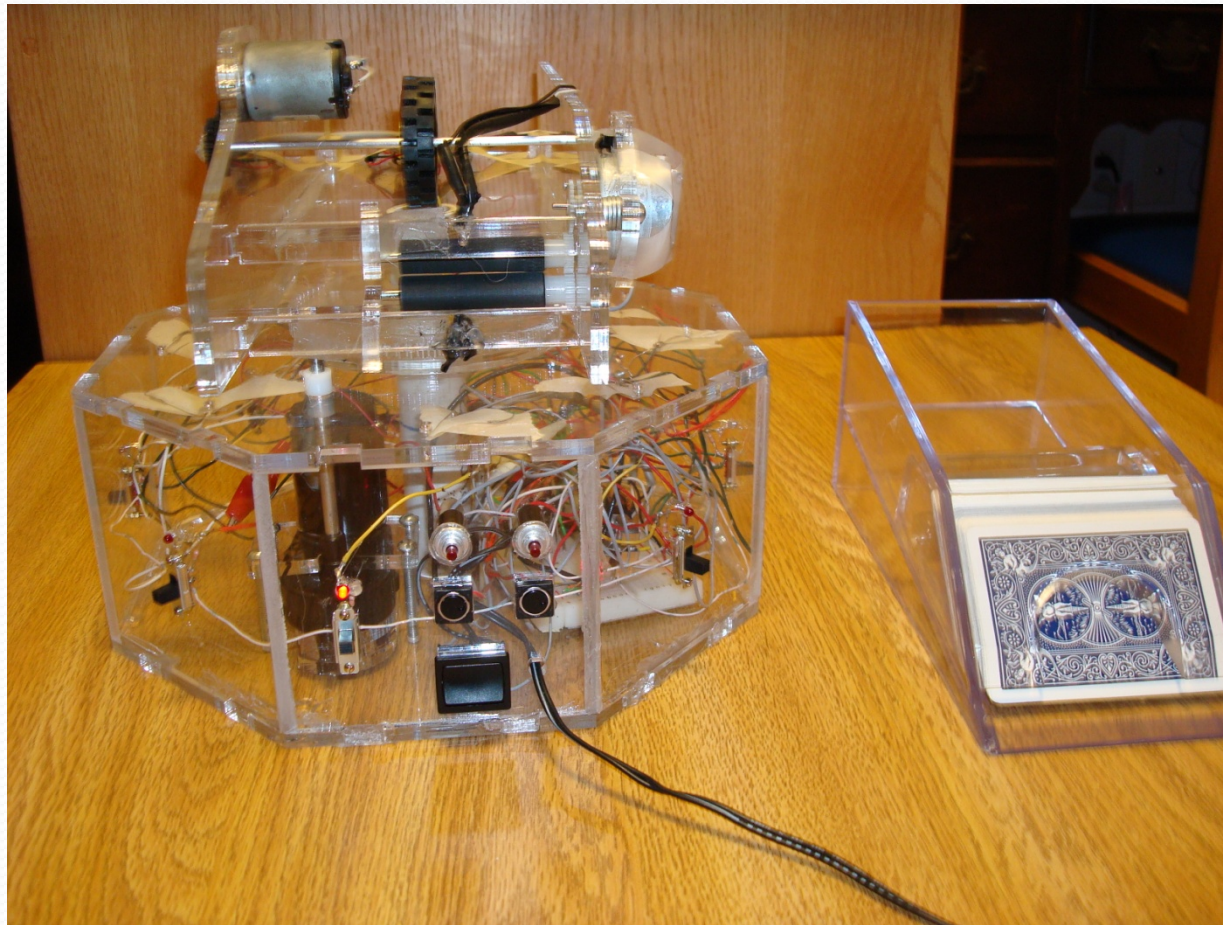
ME 430 Project Completed By:

Jeremy Goodsitt

Nathan Wendt



# Basic Layout



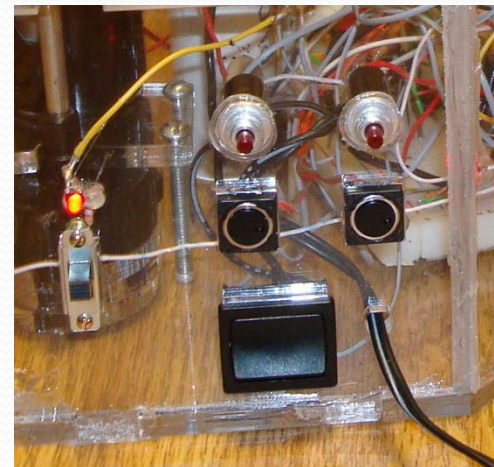


# Inputs

- 3 Infrared Detectors
  - Position Detector
  - Player Present
  - Card Dealing
- Start Button
- Reset Button
- Deal Single Card Button
- Rotate to next player Button



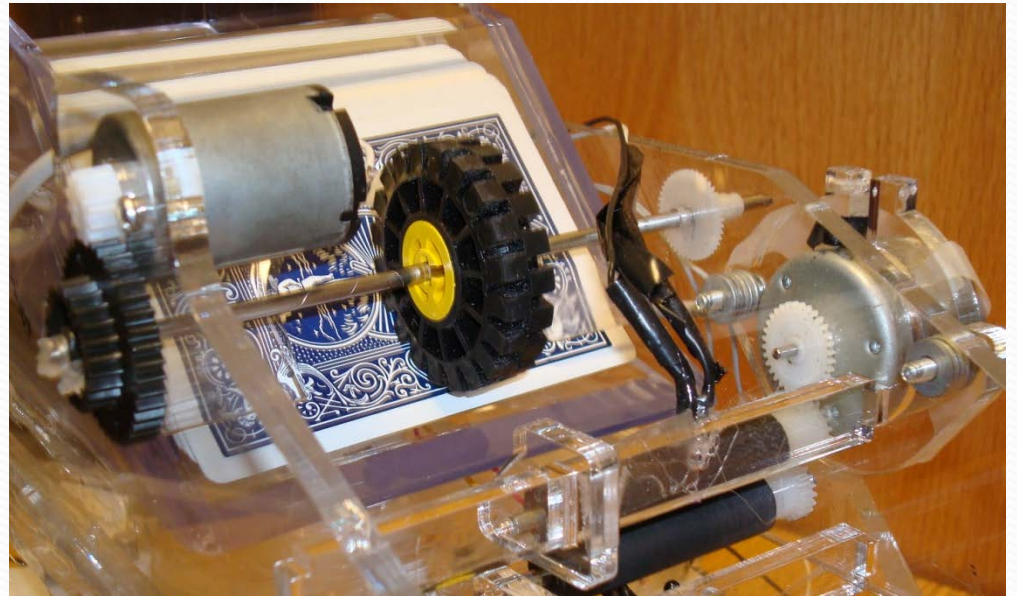
The Card Dealing IR Pair



Dealer Control Panel

# Outputs

- Main Drive Motor
  - Included enable bit and forward and reverse motion
- Deal Card Motor
- Shoot Card Motor

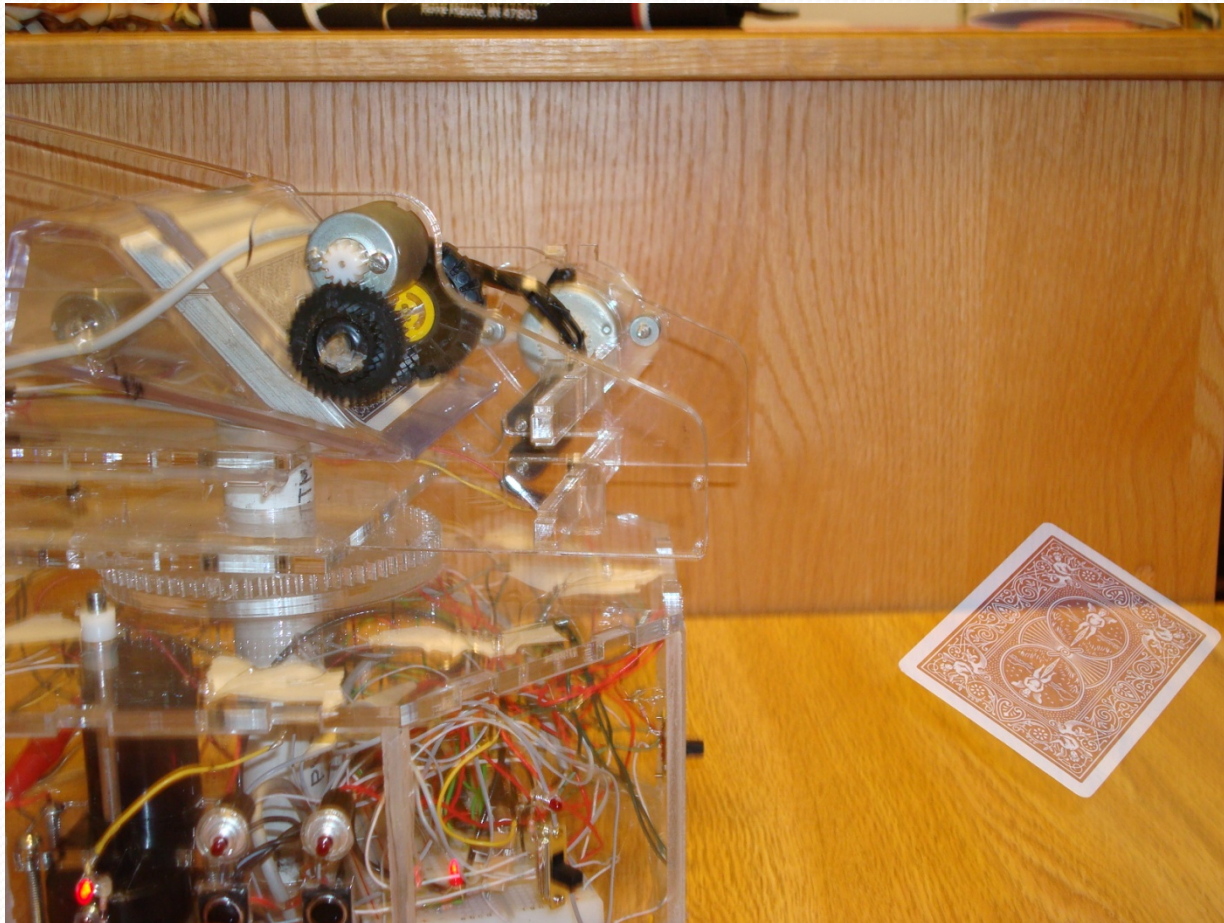


# Basic Operation

- After the start button is pressed, the machine first rotates to find its position and then deals a card to every player with their light turned on. After the machine has rotated 5 times (i.e. dealt 5 cards to each player) the machine rotates in reverse 5 times to untwist the internal wires and then stops.
- The deal card button will deal a single card at the current position
- The rotate to next player button will rotate the machine to the next available player
- The Reset button will stop the machine



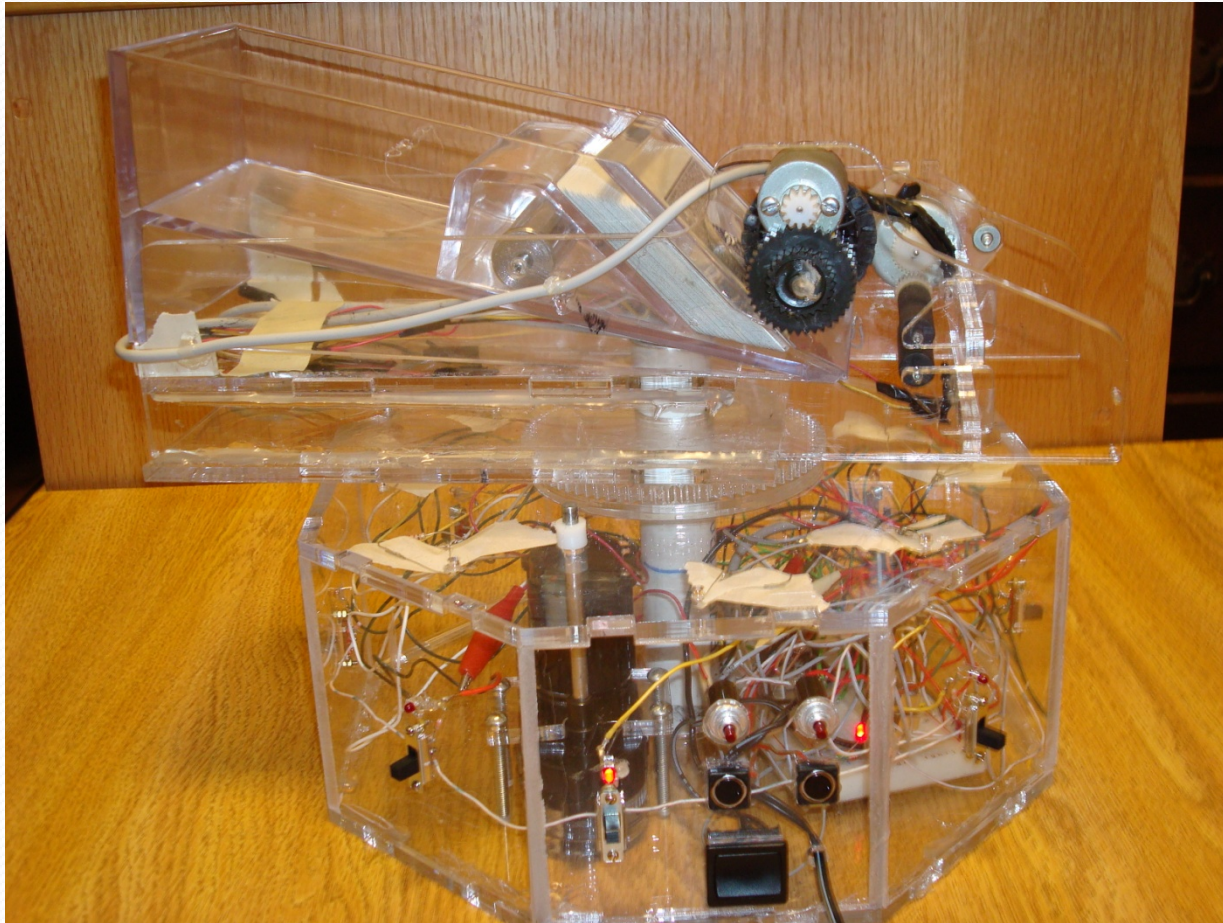
# Pictures



Machine running and shooting. This picture is kind of neat...



# More Pictures



# Lessons Learned

- Start early on the project. You do not really get class time to work on the project and the deadline has a tendency to sneak up on you.
- Plan early. If using the laser cutter, model early.
- Ask for help!
- When working with faculty provide plenty of lead time and information when requesting help or services. Be respectful of their time and efforts to help you!

# Lessons Learned (Cont'd)

- Avoid floating inputs!
- Put your project together in parts.
- Make sure the simple things are working before the more complex things. (i.e. make sure I/O's are working before trying to drive actual hardware)

# Automatic Light Switch

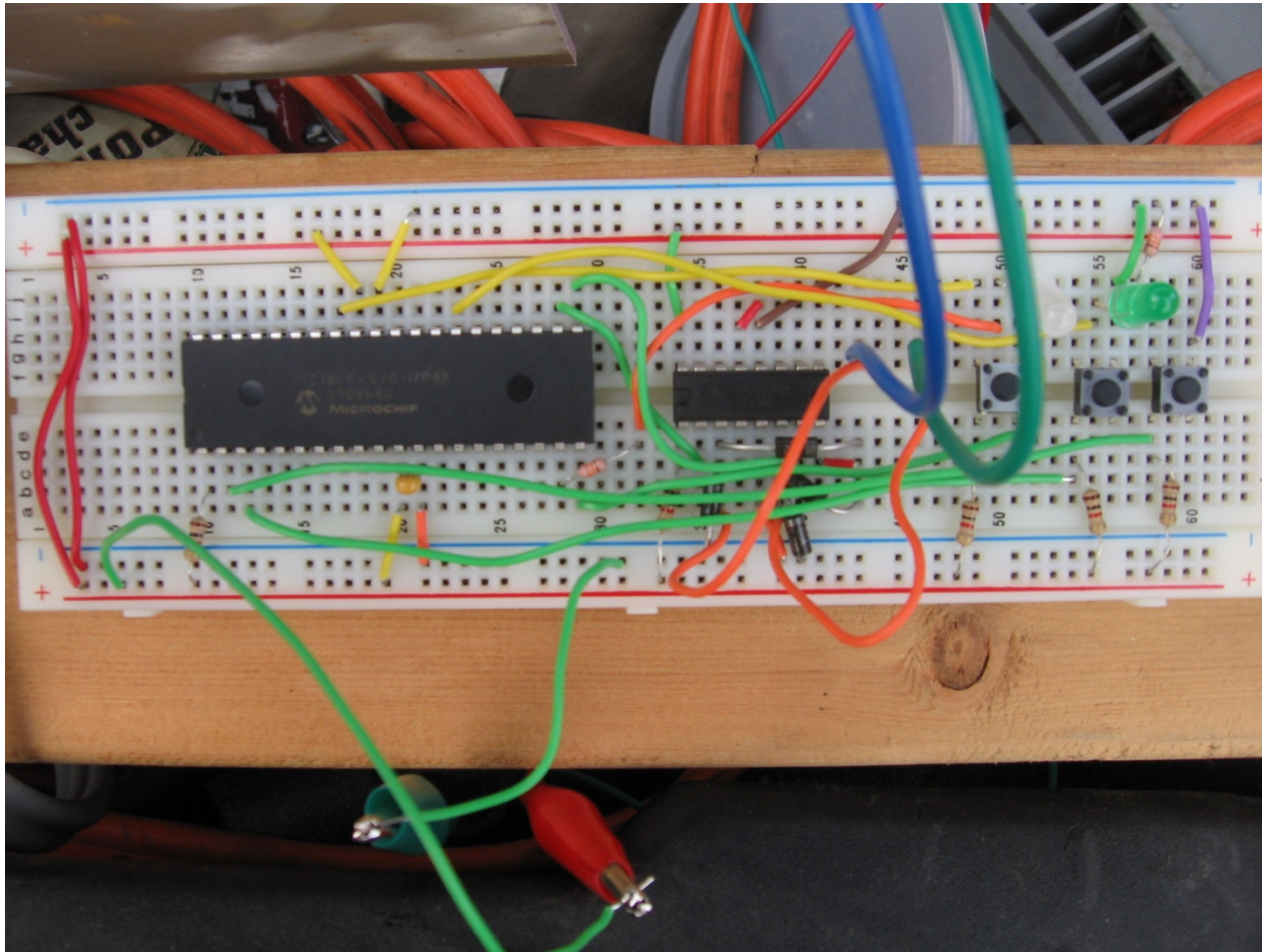
By:

Casey Collins

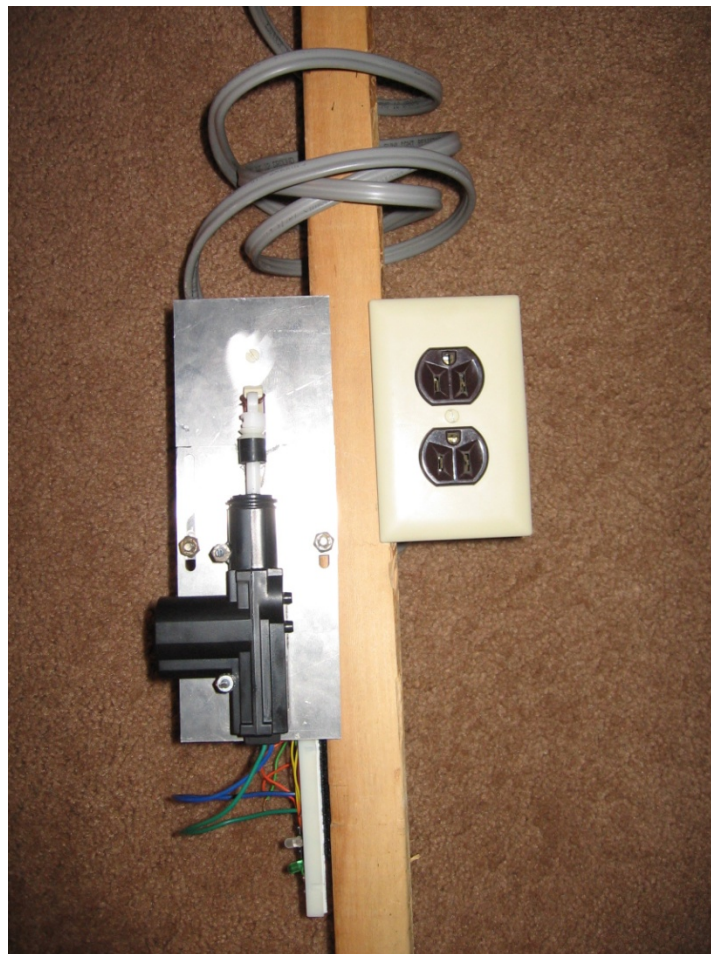
Ryan Bormann



# Circuit with 3 buttons on right



# Automatic Light Switch Setup



# I/Os

- Inputs
  - Switches that help to select the desired mode
  - Switch that tells circuit if door is open or not
- Outputs
  - Manual mode bicolor LED
  - Automatic mode green LED
  - Signal to H-Bridge that turns on/off actuator

# Completed Objective

- Circuit has a manual button that will permanently turn on/off light switch
- Circuit also has an automatic button that turns on light for several seconds every time the door is opened (door trigger is a button on the circuit for convenience of showing functionality)



# Displays current mode of operation

- For the user to know what mode they are in there is a bicolor LED when in manual mode
  - When light is on current runs one way and LED is green
  - When light is off current runs other way and LED is red
- When in automatic mode, bicolor LED is green and automatic LED is also lit Green

# Lessons Learned

- Use interrupts to stop timed functions and begin other functions
  - This was done as a means to exit the automatic timed mode and enter manual mode
- Cheaper is sometimes better
  - Used cheaper low amperage 12V power supply and that helped to protect our H-Bridge from overloading

# Bicycle Speedometer, Tachometer and Gear Display

Jeff Dovalovsky and Jonathan Lyu

# Project Objective

- ▶ Create an attachment to a bicycle that shows the following information:
  - How fast the rider is travelling – Speedometer
  - The revolutions of the wheel per minute – Tachometer
  - Which gear he or she is currently in.





# Inputs

- ▶ Our main input was a magnetic switch that is placed on the wheel of the bicycle. Every time the wheel rotated, the switch was triggered as it passed the sensor.
- ▶ The gear status will be calculated from the speed and rotation of the wheel to decide which gear the bicycle is in.



Magnetic switch

# Hardware/Software

- ▶ In our apparatus, 2 PICs, 18LF4520, were used. Also, 5 ULN2803 Darlingtons were used to control the LED outputs.
- ▶ In our programming code, we used timers, interrupts, and delays.



18F4520 PIC

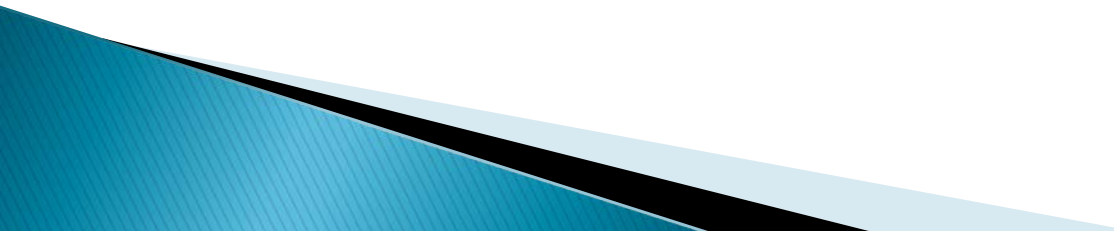


ULN2803 Darlington

# Outputs

- ▶ The outputs of our apparatus are rows of LEDs which indicate the current speed, RPM, and gear.
- ▶ They are controlled by the two PICs which receives the data from the magnetic sensor and displays the desired output.

# Advice to Future Generations

- ▶ One of the most important aspects of this project for us was time management. Making sure we met all the checkpoints was key.
  - ▶ During the course, always examine what you have learned and see how you could apply that to your project. Write down ideas as soon as they come, instead of having to look back.
  - ▶ Good communication with your partner is essential. Scheduling meetings or dividing up the workload makes the project go much more smoothly and successful.
- 



# Dog's Best Friend



Chris Huizenga  
Preston Render  
Jarrett Waters

# Project Objective

---

- Allow the machine to run autonomously once user inputs are set
- Throw a ball for a dog to fetch
- Once ball is returned, machine throws again
- Repeat above cycle until user inputs are met

# Inputs

- Use PIC p18f4520 to control inputs
- Time/Cycle Input Pushbuttons
  - Allows user to set number of minutes/cycles for dog to play
- Set/Play Input Switch
  - Toggles between 'play' mode and 'user-input' mode
- Ball Returned Switch
  - Senses return of ball



# Outputs

---

## ● Cycle Display

- Use delays to wait 3 seconds to activate 7-segment display once Ball Return Switch is triggered

## ● Timer Display

- Use Timer0 Interrupt to calculate minutes on PIC
- Update timer display

## ● Motor

- Use delays to wait 3 seconds to ensure proper loading of ball once Ball Return Switch is triggered



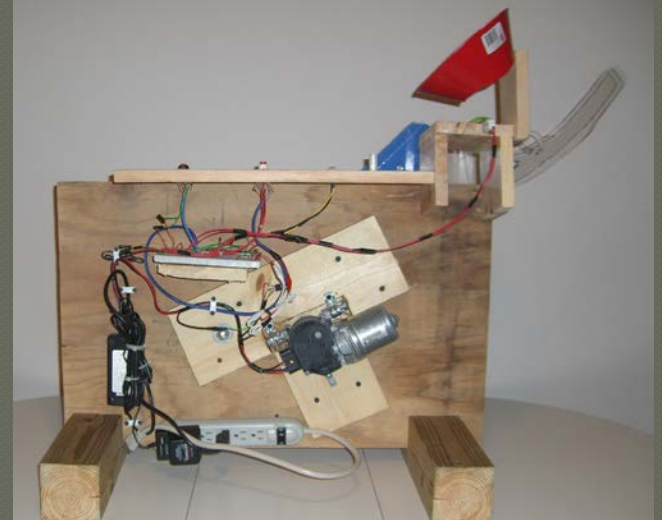
# Machine Design

## ● Four-Bar Linkage

- Designed in Working Model
- Allows for launch distance with compact design

## ● Motor

- 12 V DC Windshield Wiper Motor
- Resetting feature with loss of ground



# Advice

---

- Mosfet Transistor

- Use the Mosfet to control ground for higher voltage devices

- CAD Preliminary Design

- Allows for testing of model

- Be Simple and Creative

- Balanced combination of mechanical and electrical/programmable components

Nathan Hazard and Quintin Coppola

# FINGER FOOTBALL

# Objectives

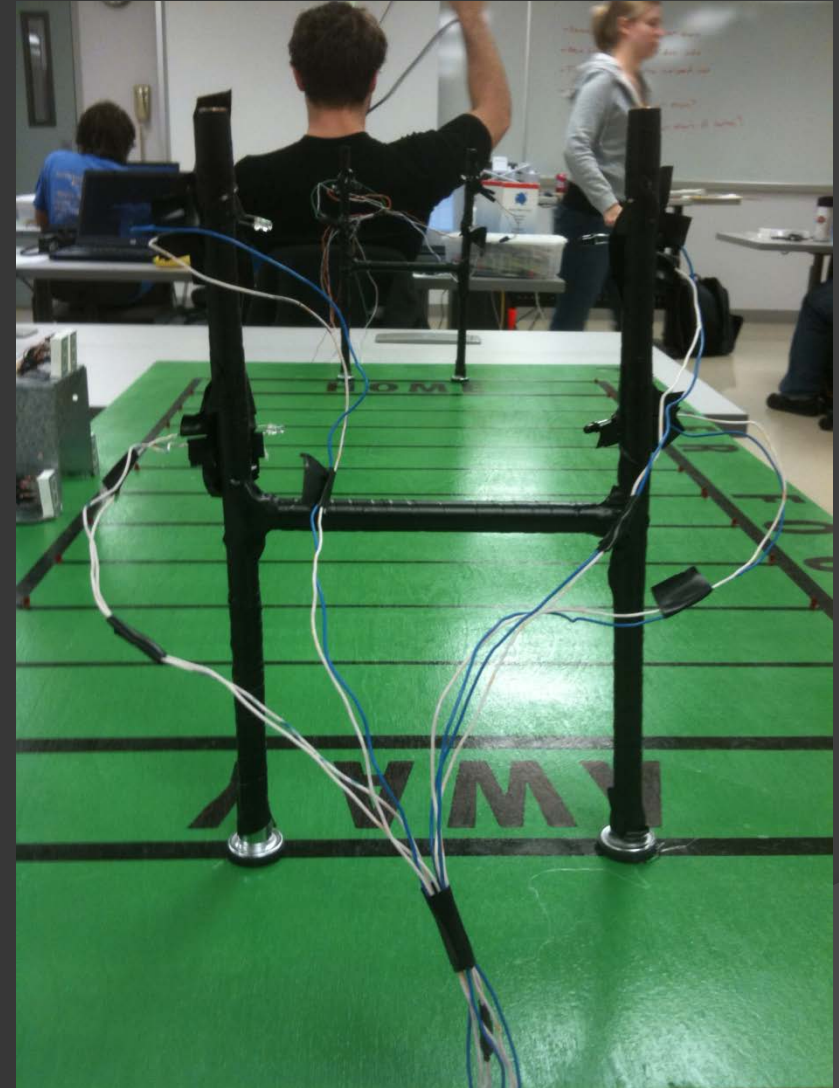
- Create a two person Paper Football game that would keep score/time while also remaining visually attractive and user friendly





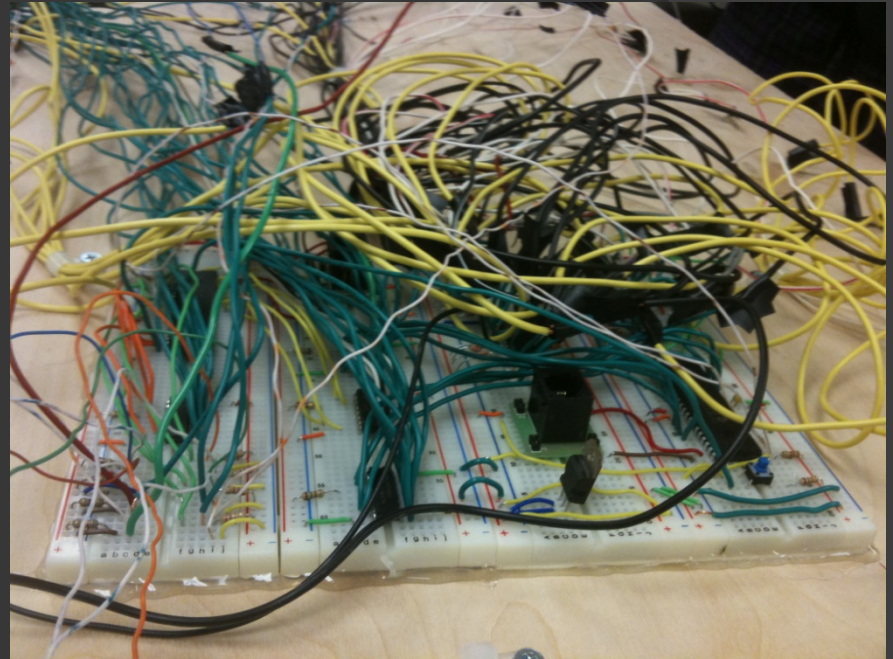
# Inputs

- Start / Restart Button
- 4 IR Sensors
  - Detects if Football flies between goal posts
  - 2 sensors for each side



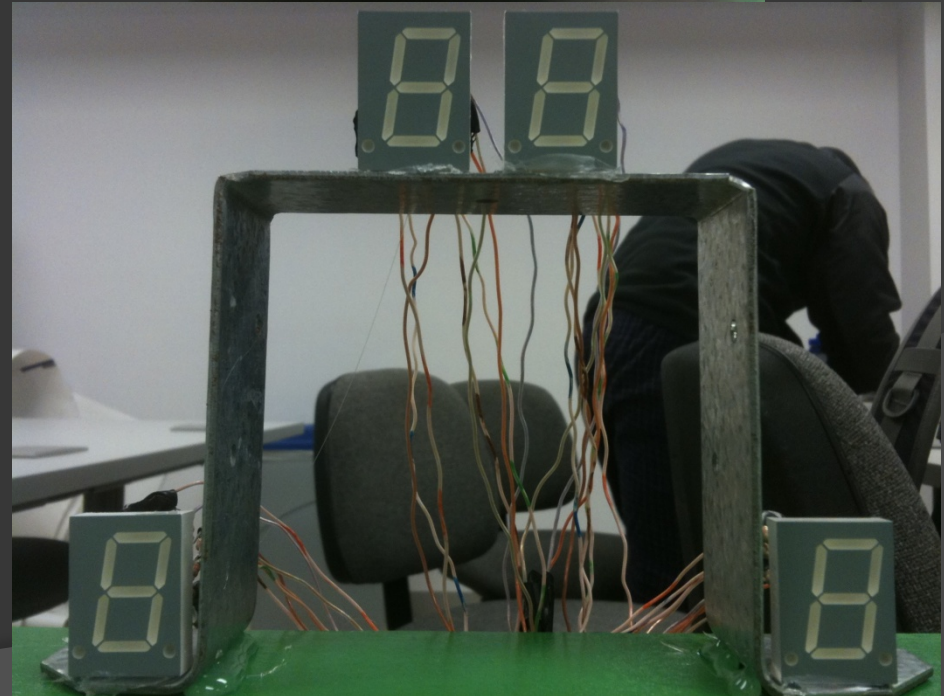
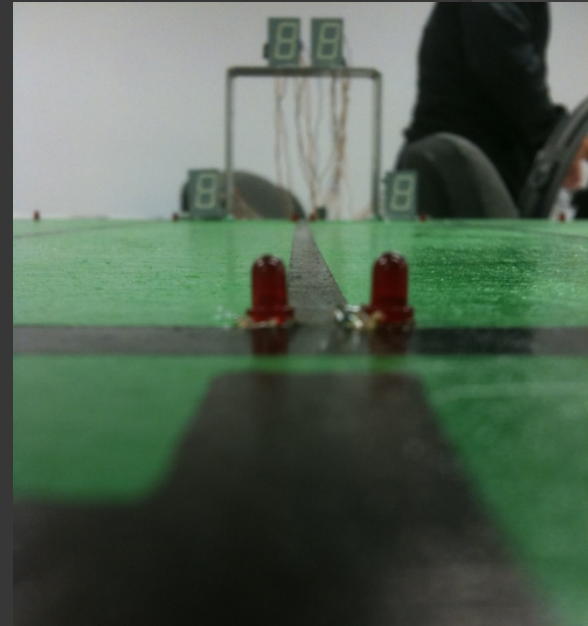
# Controllers

- PIC 1
  - Time 7-segments
    - Time counted w/ Interrupt
  - Touchdown LEDs
  - Detect start/restart
  - Detect IR inputs
    - Pressed/Unpressed code
- PIC 2
  - Received 3 inputs from PIC 1
    - Goal Home
    - Goal Away
    - Reset
  - Score 7-segments
  - 10-50 yard LEDs

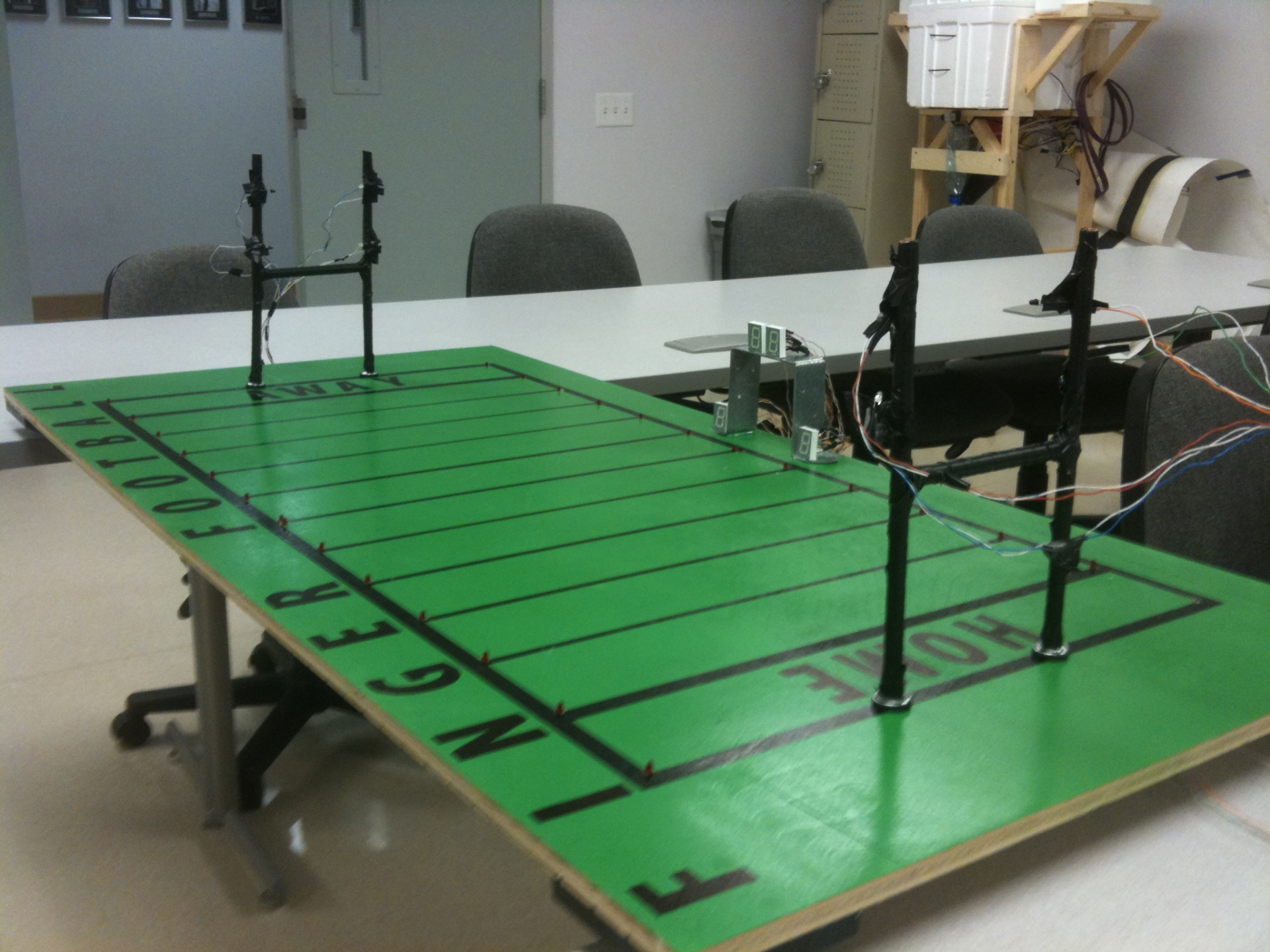


# Outputs

- Time (2 7-segment displays)
- Score (2 7-segment displays)
- 26 LEDs with 3 flashing modes
  - Normal game play
  - Crazy score mode
  - End of game







# Advice

- ⦿ Create reasonable objectives for project
- ⦿ Build incrementally!!! Don't try to integrate everything at once
- ⦿ Invest in quality wire and solder
- ⦿ Obtain part documentation before you order the parts
- ⦿ Be aware of build time and don't wait until the last minute
- ⦿ Switch lab duties with partner. This will make it easier to trouble shoot and prepare you for the exams



# MAKE-A-SHOT 3000

By: David Pope  
Travis Vanderberg

# What is Make-a-Shot?

Basketball Arcade Game!

Includes:

- Countdown Timer

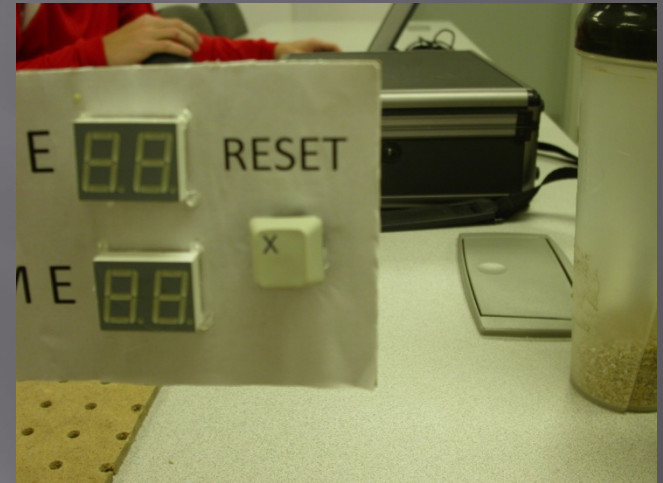
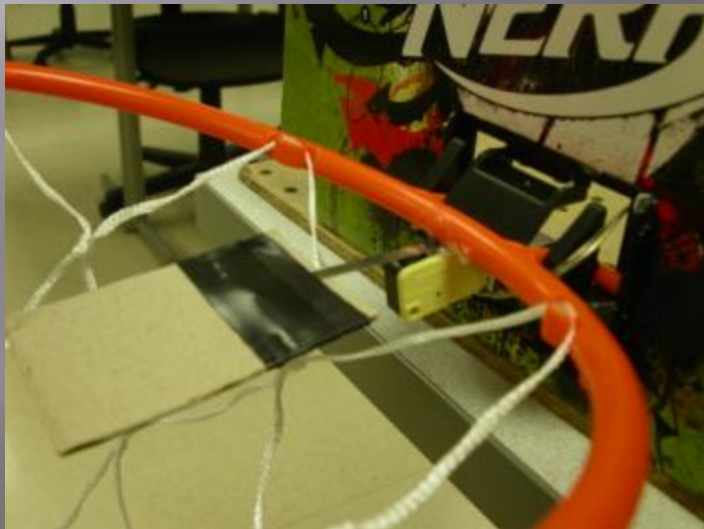
- Keeps Track of Points

- Manual Reset Button



# Inputs

- ▣ Paddle Bit
  - Used with interrupt
  - Works similar to button
  - Sensed rising edge



- ▣ Reset Button
  - Used inside main loop
  - Sensed for when button was high (1), or pressed

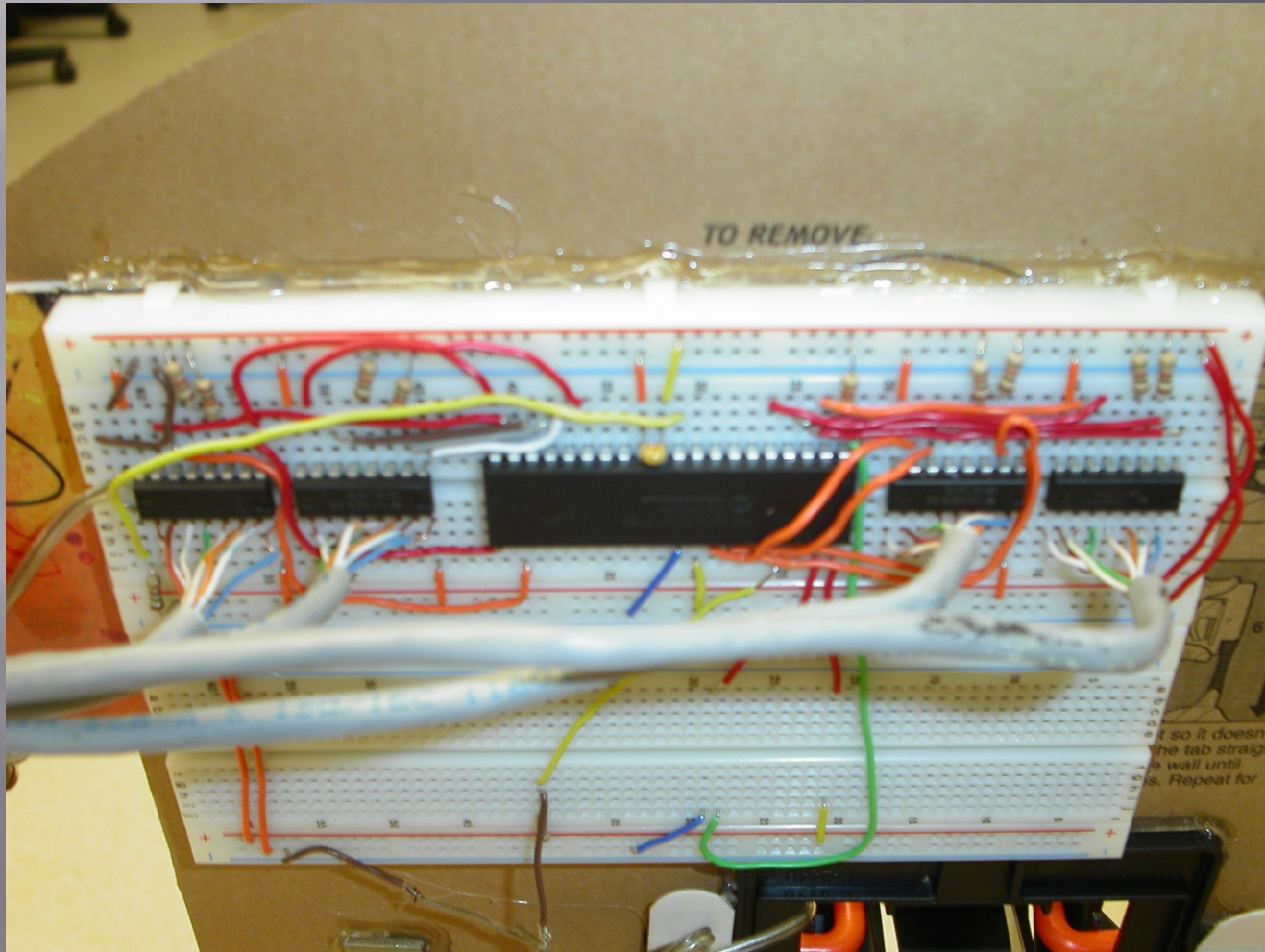
# Outputs

- ▣ Timer
  - Starts at 30
  - Counts to 0
  - Used timer.h program in MPLAB
- ▣ Scoreboard
  - Counts up by 2
  - Coded inside Paddle Bit interrupt





# PIC Used





# Recommendations

- ▣ START EARLY!!!!!!
  - Something WILL go wrong
- ▣ Test parts as soon as they come in
- ▣ Have a couple ideas in mind on how to code your project
- ▣ Choose something you'll enjoy building

# PLINKO

By:

Brittany Bernthold

Cassie Pfeilsticker

ME430-02





# Project Objective:

## What were we trying to do?

We wanted to recreate the classic television game PLINKO. In the game, a disk is dropped down a board and falls into a slot. Each slot corresponds to a certain score. Then the score is displayed on a screen. Another disk is dropped and the score is added. We wanted our PLINKO board to behave similarly; it would have several slots that would correspond to a score. We also wanted to display a score on a seven segment display as well as add the score correctly. A reset button was also wanted so a new player could start their score at zero.

# INPUTS

- Seven switches to sense the disk in a slot
- One on/off switch to control power to the board
- One reset button to clear the score

# How did we use the PIC as a Controller?

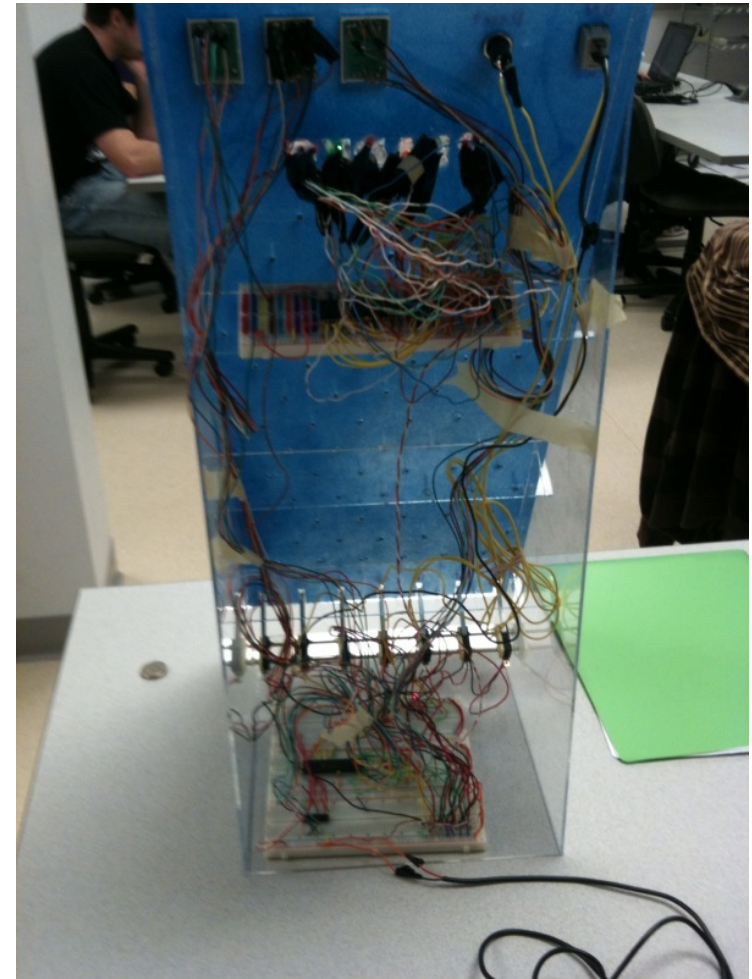
- We used two PICs
  - The first PIC was used to translate the switches being pressed into a value that was displayed on the seven segment displays
  - The second PIC was used to control the led display for extra appeal
- Delays, the recentButtonState method, and a helper function were used

# OUTPUTS

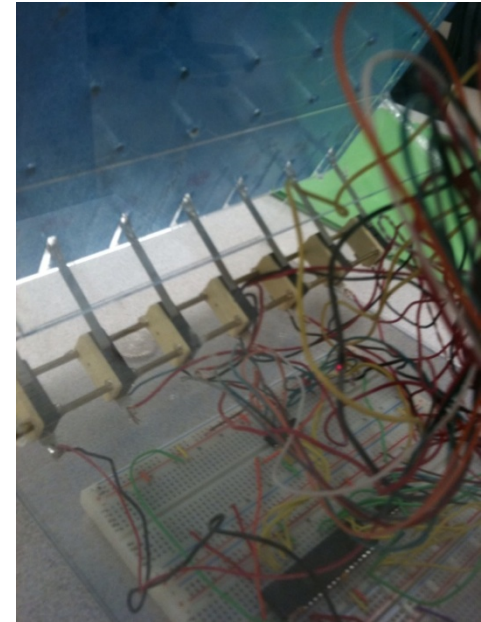
- 3 seven segment displays that displayed the score
- LEDs that flashed for a cool lighting effect



# PICTURES



# More Picture



# Advice to Future “Generations”

- Stripping wires takes forever so start early
- Soldering material runs out early, so either be the first to use the guns or go buy your own material
- If you can use solid edge to build your project, it makes making parts a lot easier and you can make precise adjustments
- Raid the storage room!!! There are lots of useful parts for projects in there.

# Thanks

- Dr. Olson
- Ron Hoffman
- Chris Wlezien



# “Quick Cooler”

Ryan Schalk

Michael Burris



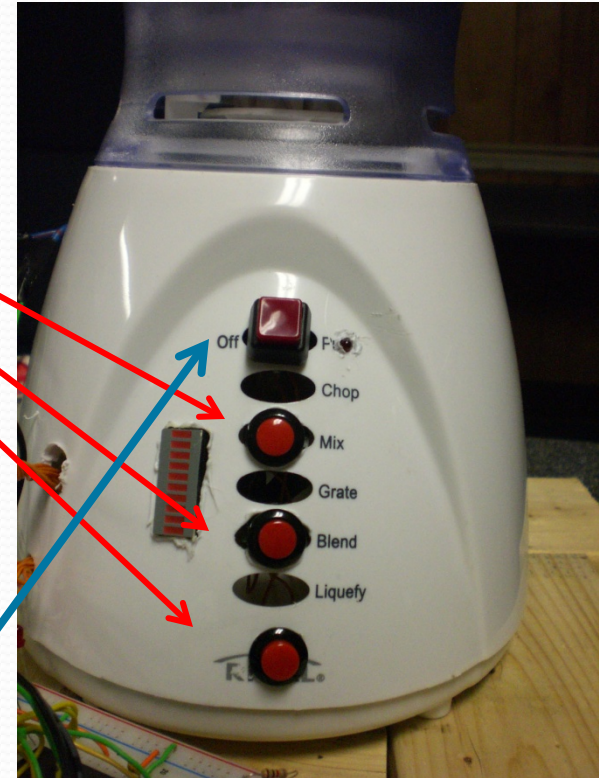
# Objective

- Make a device that quickly cools a room temperature 12 ounce can/12 ounce bottle/16 ounce can, to a cool drinking temperature.



# Inputs

- Button Inputs for different size containers
  - Each button controls the amount of time the motor runs, which is a different cooling time for each input/button.
  - Also a kill switch to shut off the motor during operation. This also resets the time of operation for each input



# Using the PIC

- The three buttons caused an interrupt which chose the motor speed and time of activation
- A timer was used which overflowed every second to time the operation of the motor
- The motor was run using pulse width modulation and a MOSFET
- The kill switch was a high priority interrupt which stayed in the interrupt until the switch was deactivated

# Outputs

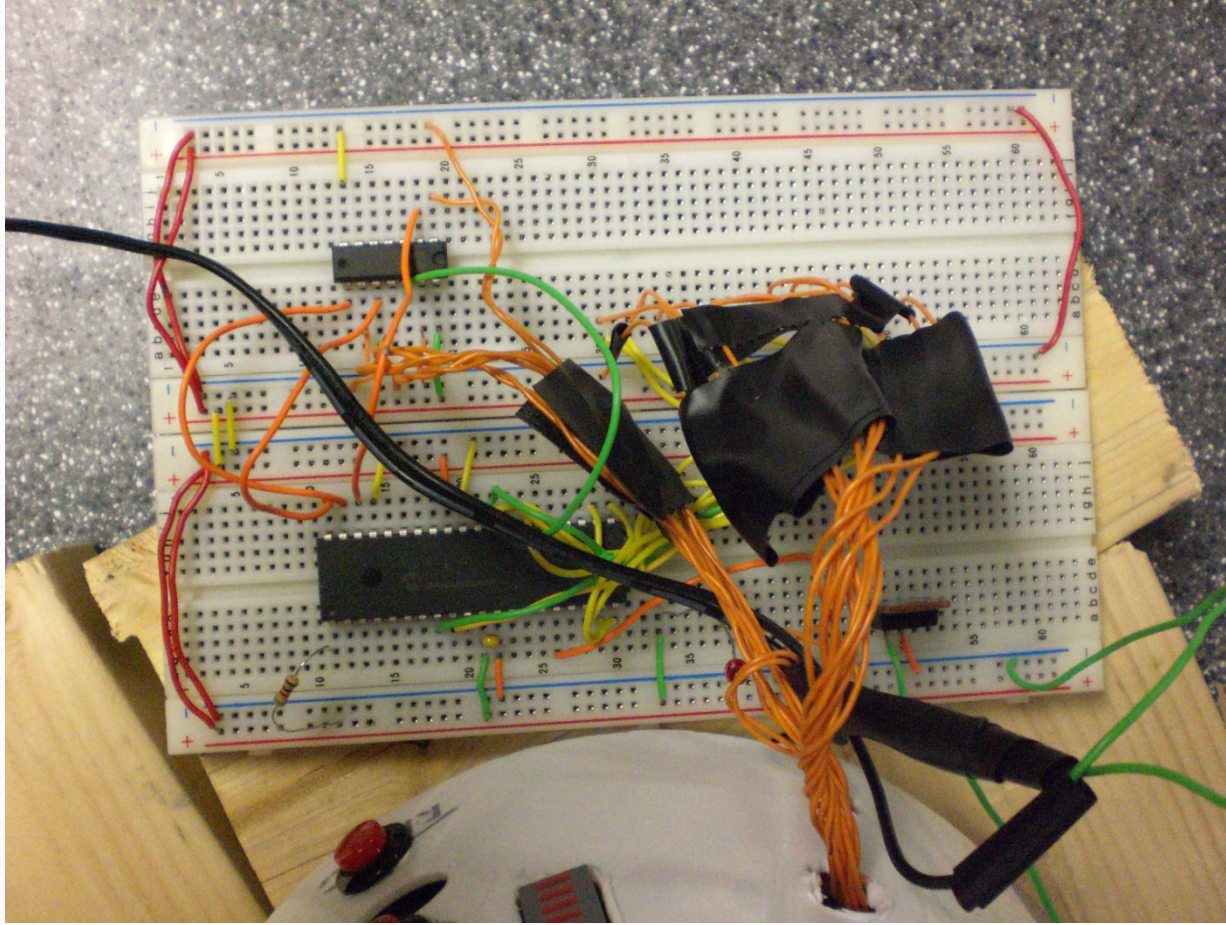
- Motor speed and time
  - A button press activated an interrupt in the program which choose motor speed using PWM and a number of timer overflows to run through
- Led timer display
  - A series of 10 LED's were turned off in order to display the time remaining. Each LED turned off when a certain fraction of timer overflows were met













# Advice for Future Classes

- Start early
- Check to see if current supplies are available to save money
  - Power supply
  - Motors
  - Buttons and LED's
- When using switches make sure they are grounded and set up correctly (use a not gate)



ME430-Mechatronics

# TAB COUNTER




By: Bethany Brisco & Joe Giacoletto



# Objective

To count the number of Tab cans that remain in a 12 can dispenser and display that number to the consumer





# The Device

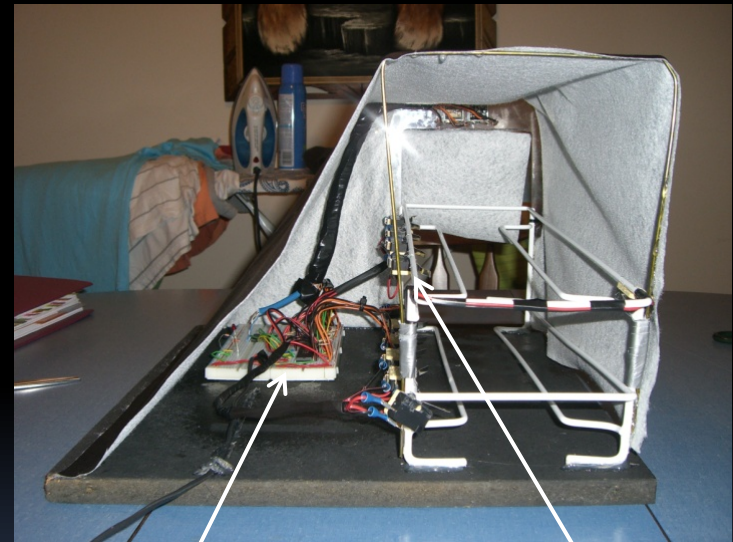
## Front



Seven Segment  
Displays

Wire Tab can  
dispenser

## Back




Motherboard  
/PIC

Switches  
(12 - one for each can)

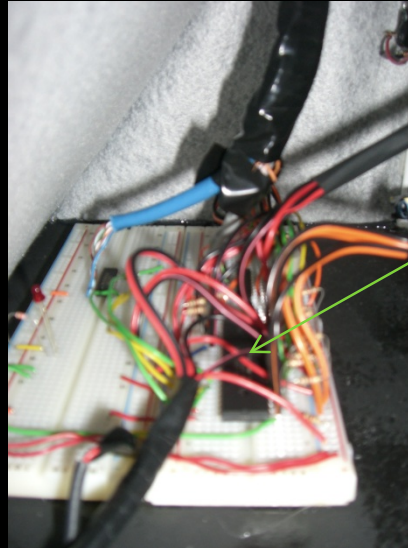


# Inputs & Outputs

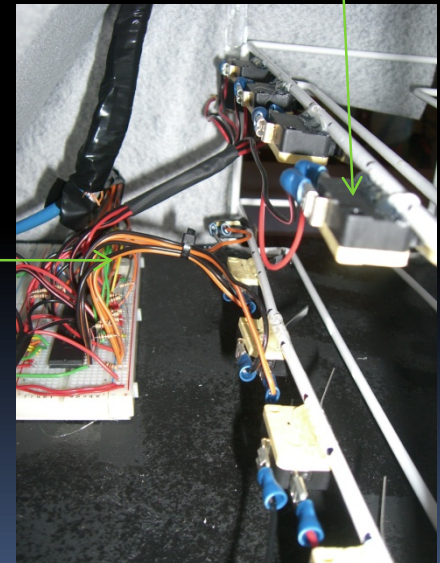
- Inputs
    - Tab cans
    - Switch Sensors
  - Outputs
    - Seven segment display showing number of cans remaining
    - One warning buzz and flash of the six red LEDs when one Tab can is remaining
    - Three warning buzzes and three sequenced flashes of the six red LEDs when out of Tab
- 

# PIC, Buzzer, & Switches

- PIC used to control the entire Tab Counter
- Delays were used to flash the LED's
- A small buzzer run by PWM was used to create the warning buzzes
- Twelve switches were used, one for each can, to count the cans of Tab in the dispenser



PIC




Buzzer

Switch



# Advice

- Order supplies early
  - Find a glue that will adhere to desired surface
  - Use wires that will easily mate with breadboard
  - Test parts when they come in to determine if you will need to reorder something
  - Allow plenty of time for use of laser cutter in case it is already in use
- 

# Tabletop Console Switch

Nobutoshi Hiro  
Adam Lusk  
Fall 2009



# Objective

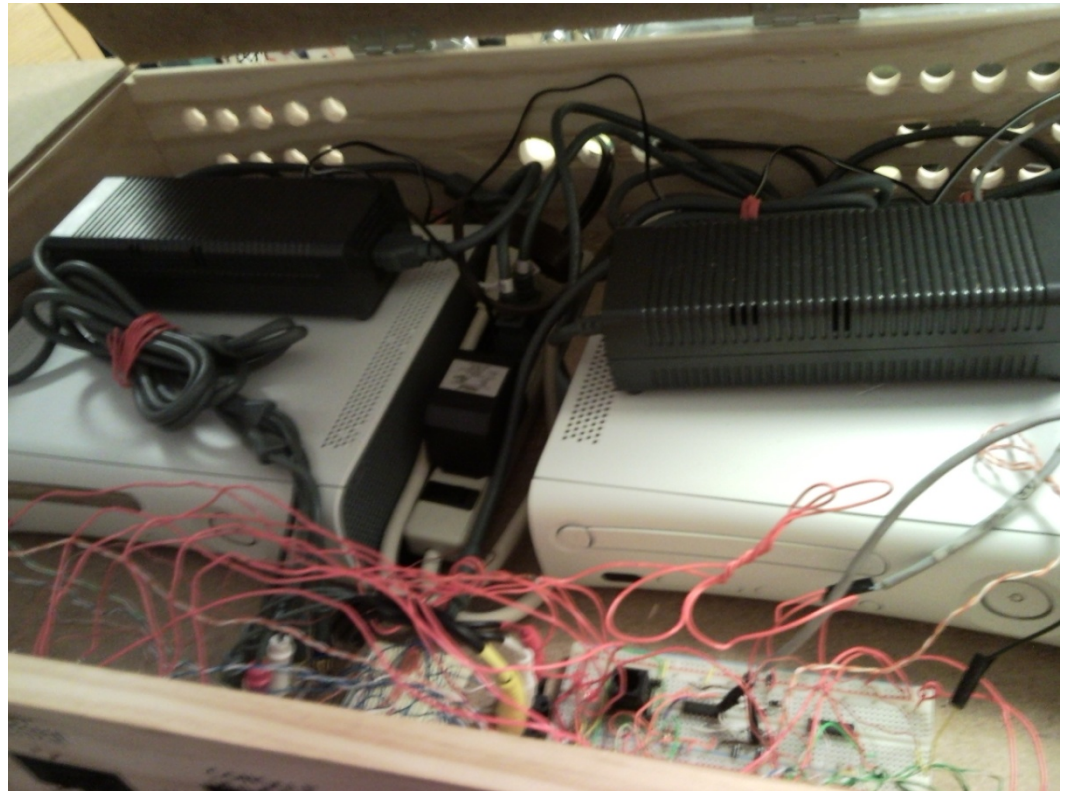
- ▶ Provide easy setup of multiple gaming platforms with specified automated controller cabinets



XBOX 360

PS3

Wii™



# Device

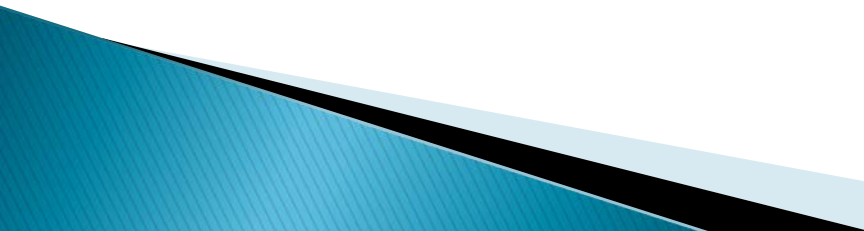
## ▶ Control

- Motors and Switches controlled by PIC using interrupts
- Video and Audio controlled by custom A/V switch

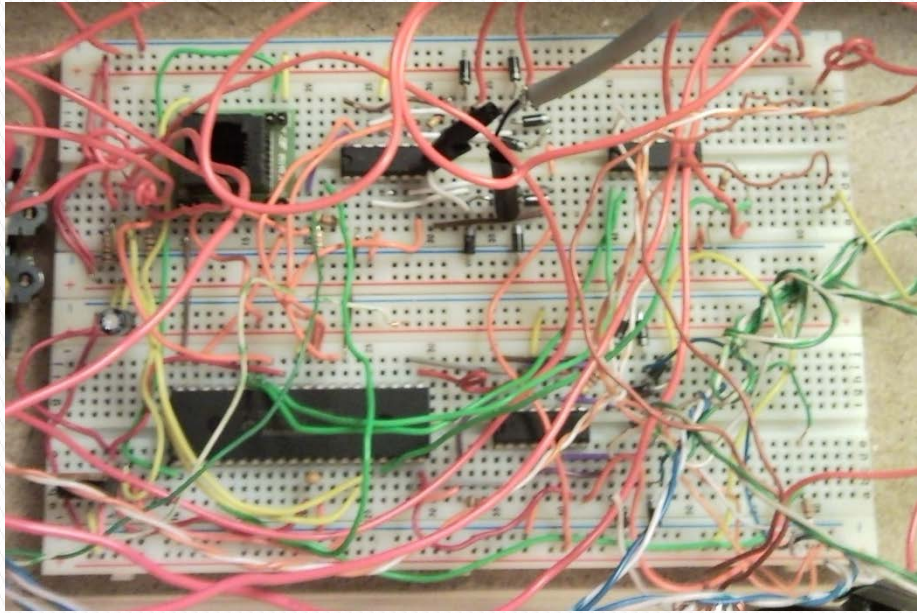
## ▶ Inputs

- Power switch
- A/V switch for selecting active game console
- Two gaming consoles
- Controller cabinet open/close switch
- Controller cabinet selection switch
- Cabinet door sensor

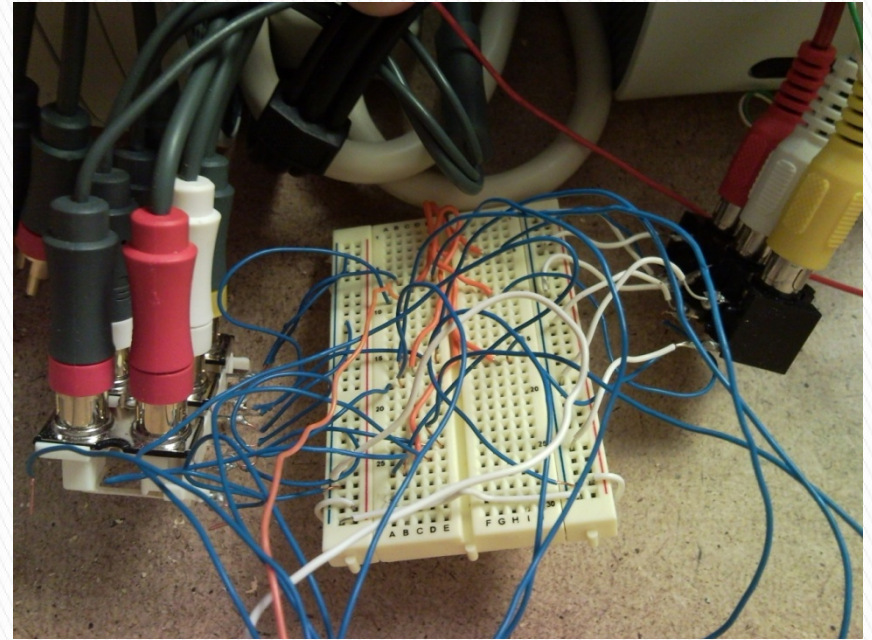
## ▶ Output

- Analog Video and Stereo Audio
  - Stepper Motors for Opening Cabinet
  - DC Motors for closing Cabinet
- 

# Wiring Layout



PIC (18f4520)



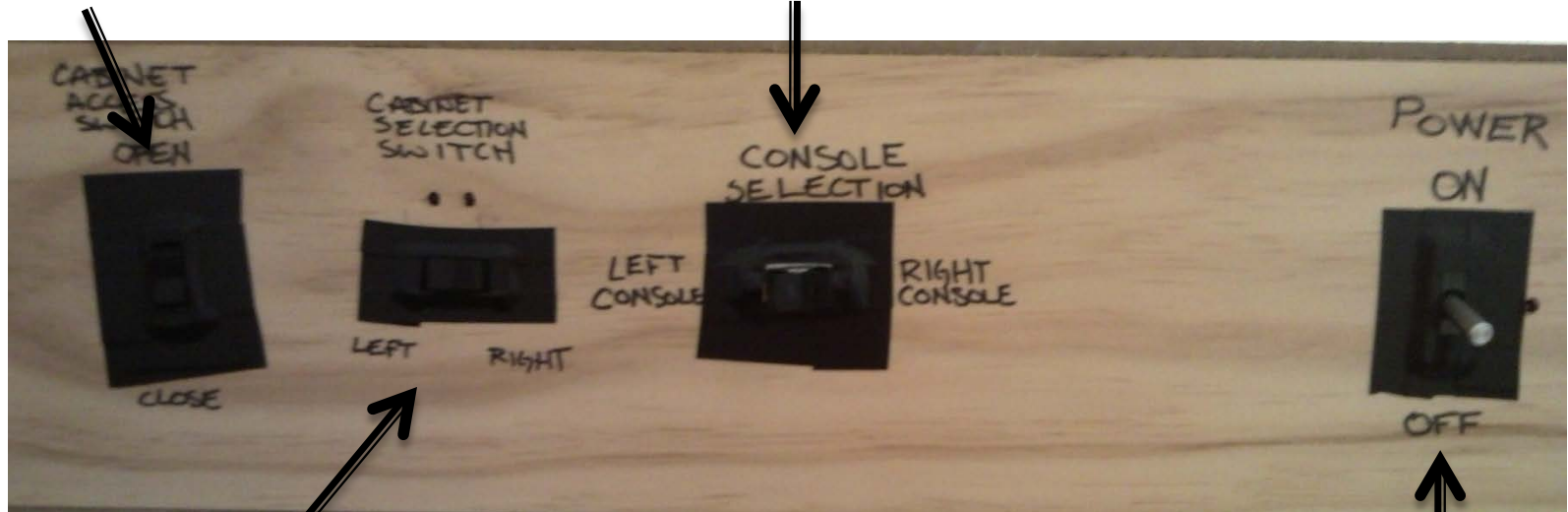
A/V Switch Board



# Switch Layout

Cabinet  
Access  
Switch

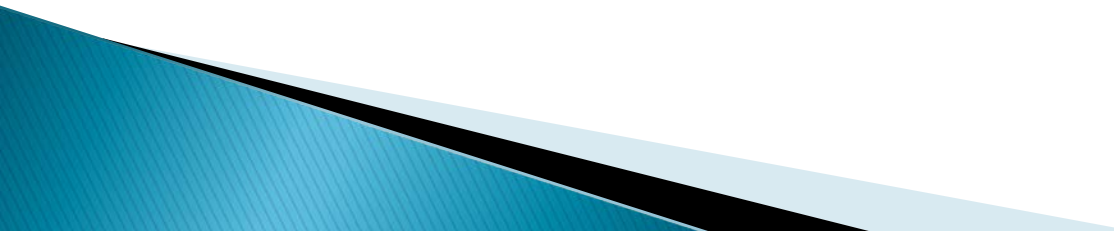
Console  
Switch



Cabinet  
Selection  
Switch

Power  
Switch

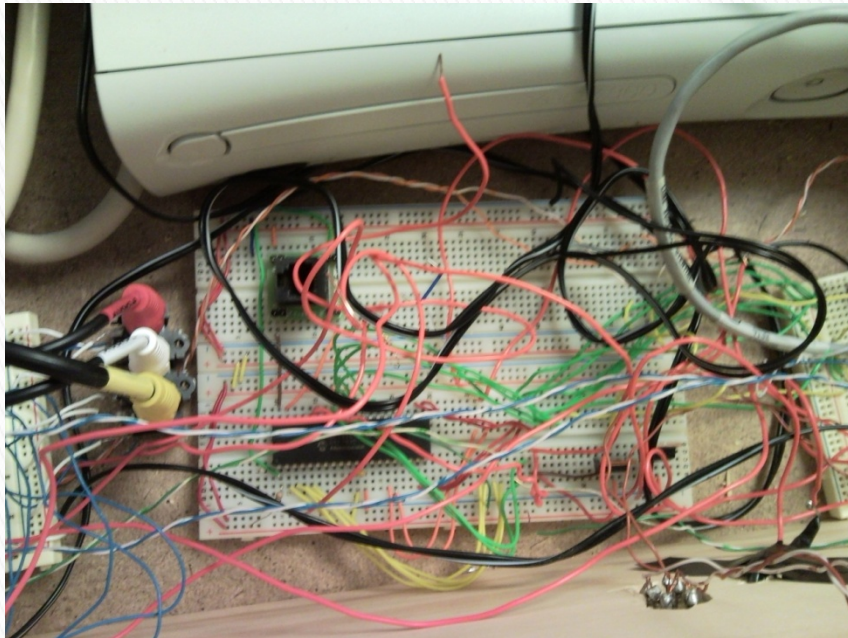
# Advice

- ▶ Clearly map out logic and design before ordering parts
  - ▶ Use materials from lab kits
  - ▶ Keep extra wire close by
  - ▶ Invest in a good wire stripper
  - ▶ Cut and paste from previously written code
  - ▶ Stay away from mechanical parts (ie. Motors)
  - ▶ Use one power supply
  - ▶ Account for more space than needed
  - ▶ Double and triple check wiring for switches
  - ▶ Do not send a voltage to a pin higher than the voltage powering the PIC
- 

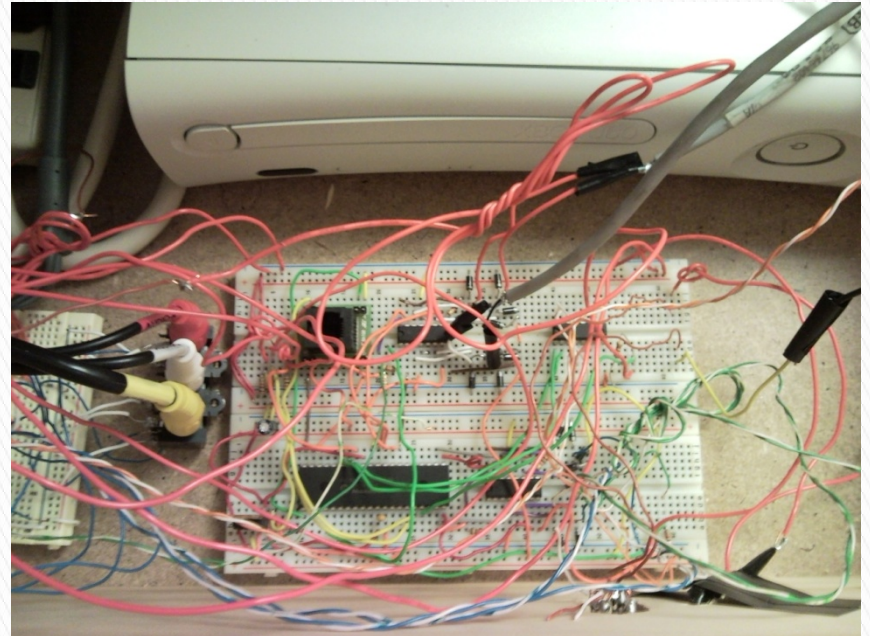


# Most Important Tip

Control wire mess early

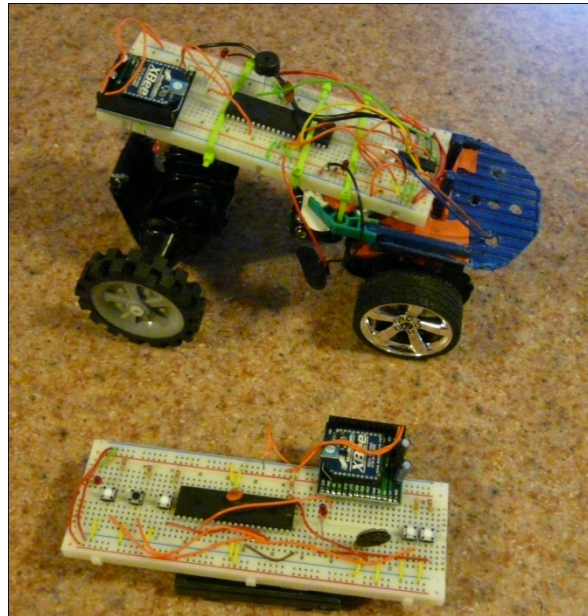


After cleanup (not much better)



# The Thomas Electric Car

An Experiment In Wireless Control

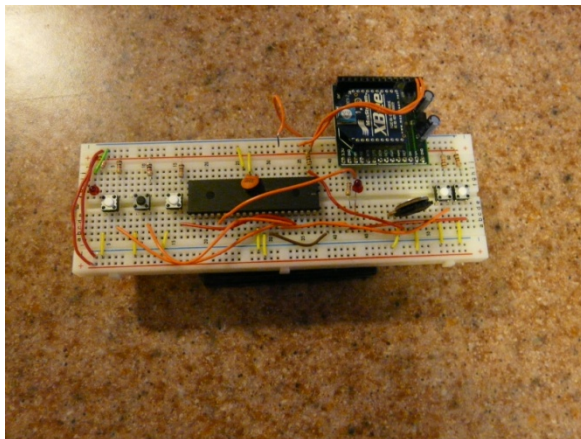


Mitchell Thomas


Ian Thomas

# Objectives

- To learn how to wirelessly communicate with a microcontroller
- To control motors with a PIC microcontroller
- To use those wireless controls to make a fun little car to drive around



# Inputs

- Controller
  - 5 Simple Buttons into the Controller PIC
    - Horn, 
  - 1 Potentiometer with A/D conversion
    - Converted to just upper 3 bits of the conversion value for an 8 position throttle
- Car
  - Received Byte From Xbee Wireless Module on USART Rx line-Receiver Interrupt
    - 3 Throttle bits, 2 Drive bits, 2 Steering bits, 1 Horn bit




# Outputs

- Controller

- Transmitted byte to Xbee Wireless Module on USART Tx line every ~10 ms using delays for timing
  - 3 Throttle bits, 2 Drive bits, 2 Steering bits, 1 Horn bit

- Car

- PWM output to Piezzo buzzer enabled by Horn bit
- PWM to enable drive motor on the H-Bridge
  - Duty cycle calculated from throttle bit value
  - 4 Motor Controls to an H-Bridge-2 Bi-Directional Motors
  - 



# Observations and Advice

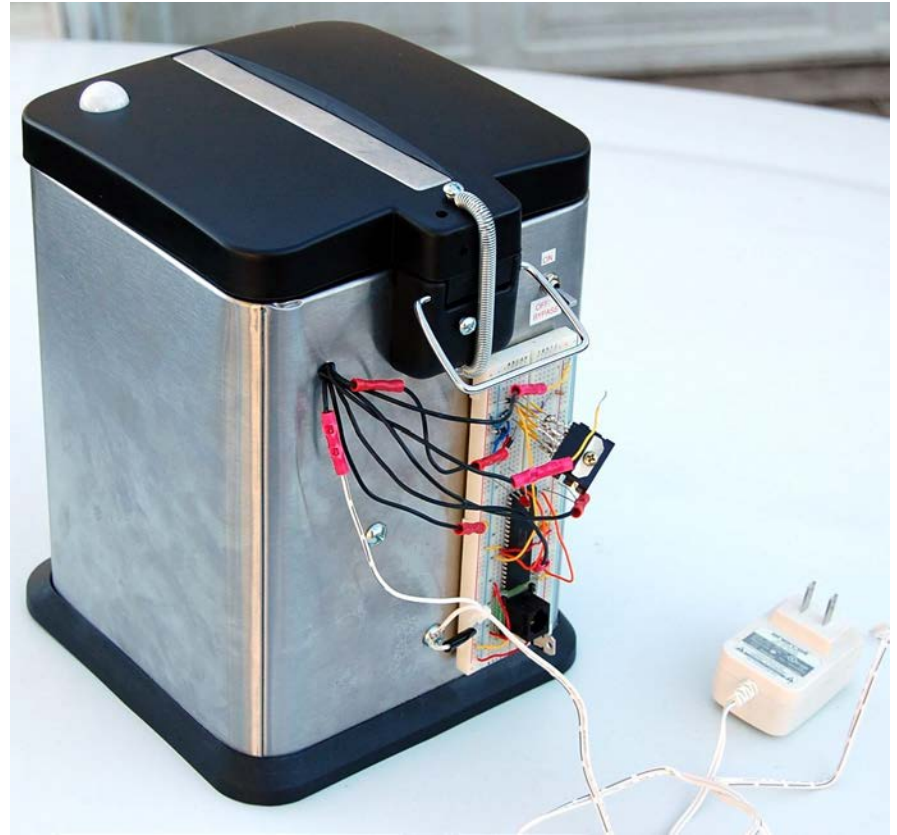
- It was a lot easier to work on a project that we wanted to have fun with when it was done-choose a fun project.
- Computers are much more predictable than mechanical things-give yourself plenty of time for tinkering and repairs.
- Think about new ways to use the class material. A familiar concept used in a new way can be rewarding.

# ME430 Project Documentation

Braxton Anderson  
CM203  
Group 9

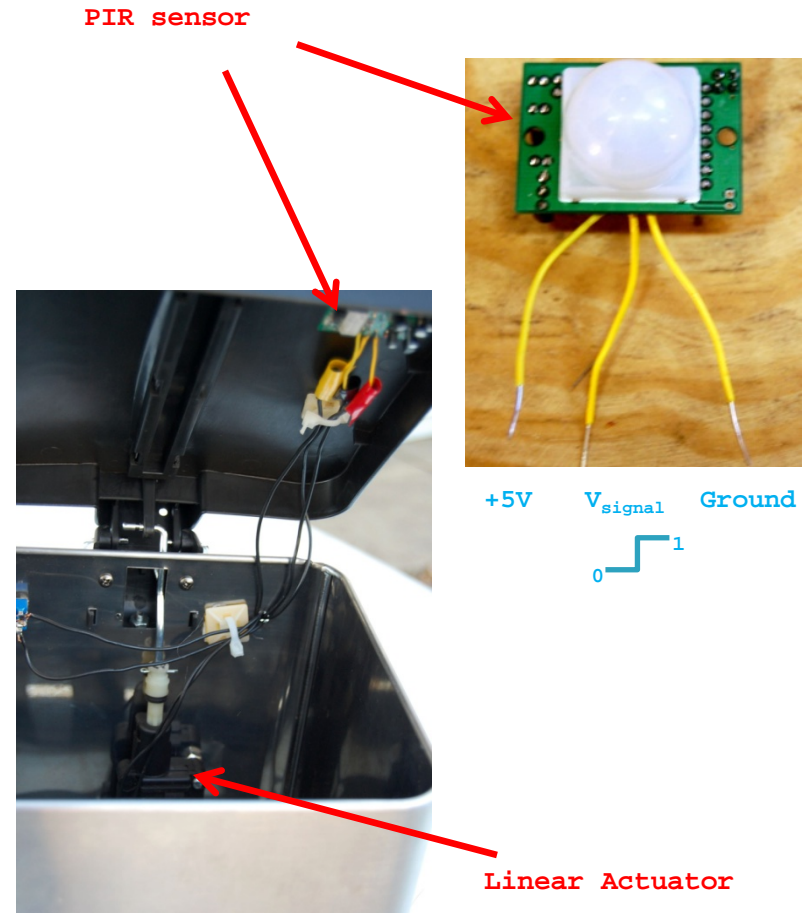
# The Automatically Opening Trash Can

- Obviously, the purpose of this project was to create a trash can that would sense a user and open automatically without the need to touch it.
- The intention was to build a can that could be operated sanitarily. No one likes to wash and dry their hands and then touch a dirty trash can lid.

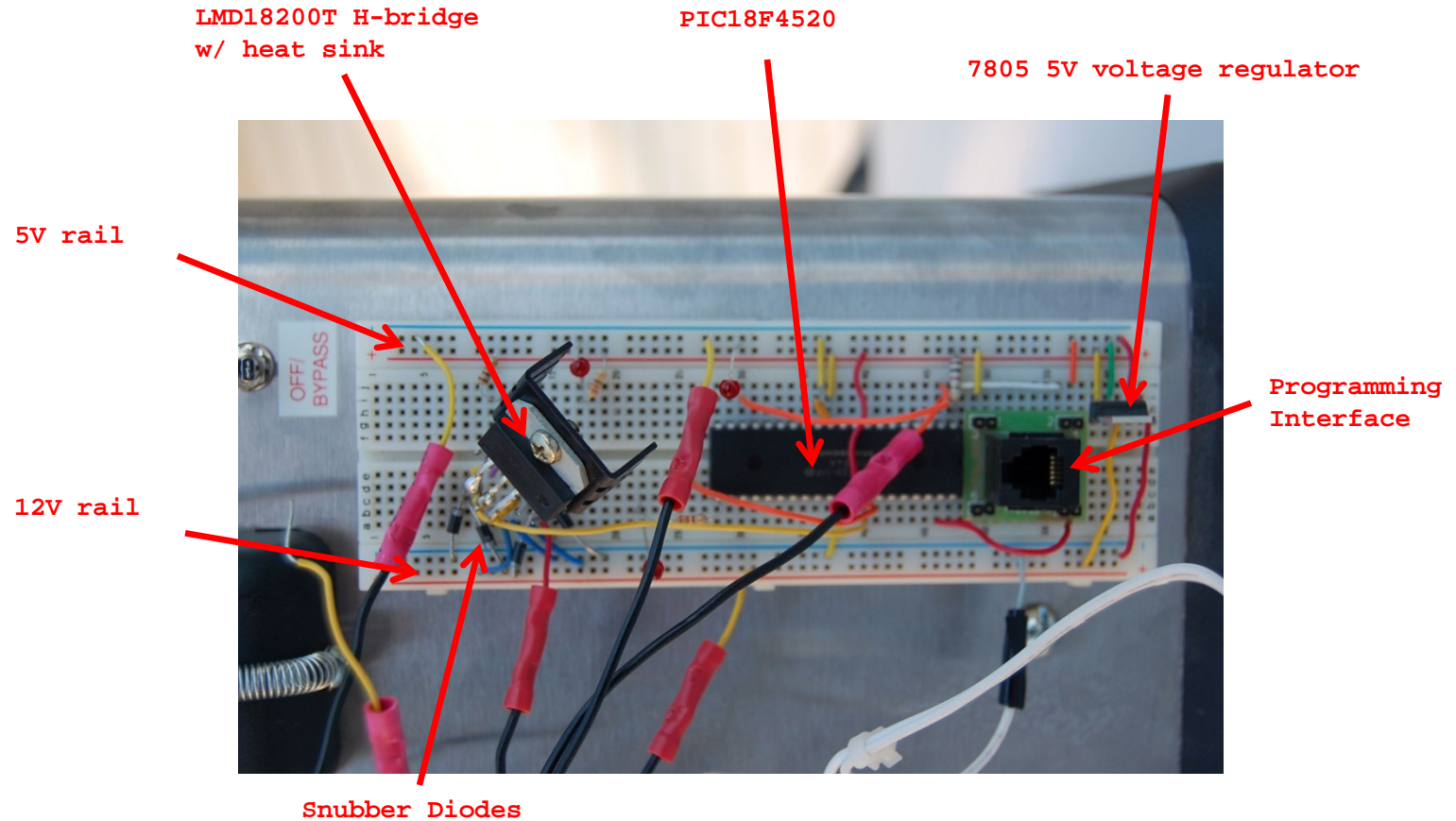


# I/O

- The only operational input to the PIC is a PIR (passive infrared) motion sensor.
- When motion is sensed, the PIR sends a digital signal to the PIC.
- Output from the PIC is routed through an H-bridge that is used to drive a linear actuator.
- Basically, the linear actuator opens or closes the can lid, depending on the signal from the PIC.



# Circuitry



*Large wires are board I/O*



# Logic

- The microcontroller used is the PIC18F4520.
- Input from the PIR goes to pin RA0.
- Output from the PIC to drive the motor (simple HIGH - LOW PWM) through the H-bridge comes from pin RB1 to H-bridge pin 5.
- Output from the PIC through the H-bridge to switch directions of the linear actuator comes from pin RB0 to H-bridge pin 3.
- Pins 2 and 10 of the H-bridge are connected to the linear actuator.
- That is it. The bulk of the code, which is very simple, is seen on the next slide.

# Code

```
/** Header Files *****/
#include <pl8f4520.h>
#include <delays.h>

/** Configuration Bits *****/
#pragma config OSC = INTIO67 // EC = External 4MHz Crystal for PICDEM board only
#pragma config WDT = OFF
#pragma config LVP = OFF
#pragma config BOREN = OFF

/*****
* Function:      void main(void)
*****/
#pragma code
void main (void)
{
    OSCCONbits.IRCF2 = 1; // 4 MHz internal clock
    OSCCONbits.IRCF1 = 1;
    OSCCONbits.IRCF0 = 0;

    ADCON1 = 0x0F; //Set all pins digital
    TRISA = 0x0F;
    TRISB = 0x00; //Set PortB as outputs
    PORTB = 0x00; //clear PORT B

    // initialize components for use

    Delay10KTCYx(250); // wait for PIR to acclimate after power on
    Delay10KTCYx(250);
    Delay10KTCYx(250);
    Delay10KTCYx(250);
    Delay10KTCYx(250);
    PORTBbits.RB1 = 1; // enable motor power
    Delay10KTCYx(40);
    PORTBbits.RB1 = 0;
    Delay10KTCYx(250);
    Delay10KTCYx(250);
    Delay10KTCYx(250);
    Delay10KTCYx(250);
    Delay10KTCYx(250);

    // run device with PIR (passive infrared) sensor input

    while (1)
    {
        if (PORTAbits.RA0 == 1) // PIR triggered
        {
            PORTBbits.RB1 = 1; // enable motor power
            PORTBbits.RB2 = 0;

            PORTBbits.RB0 = 1; // open lid
            Delay10KTCYx(40);

            PORTBbits.RB1 = 0; // disable motor power
            PORTBbits.RB2 = 1;

            Delay10KTCYx(200); // time delay while lid is open before close
            Delay10KTCYx(200);
            Delay10KTCYx(100);

            PORTBbits.RB1 = 1; // enable motor power

            PORTBbits.RB0 = 0; // close lid
            Delay10KTCYx(40);

            PORTBbits.RB1 = 0; // disable motor power
            PORTBbits.RB2 = 1;

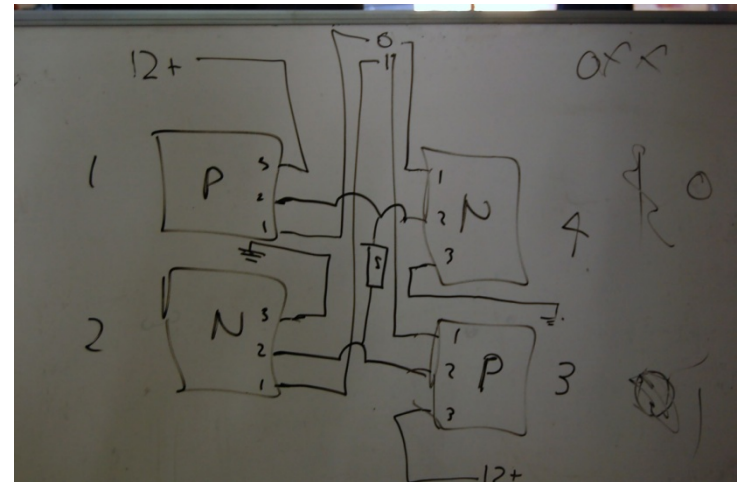
            Delay10KTCYx(250); // blind to PIR input for 5s
            Delay10KTCYx(250);
        } // end if
    } // end while

    // end program
}
```

*Output from pin RB2 is latent and unnecessary;  
for purpose of brake (H-bridge pin 4 - unused)*

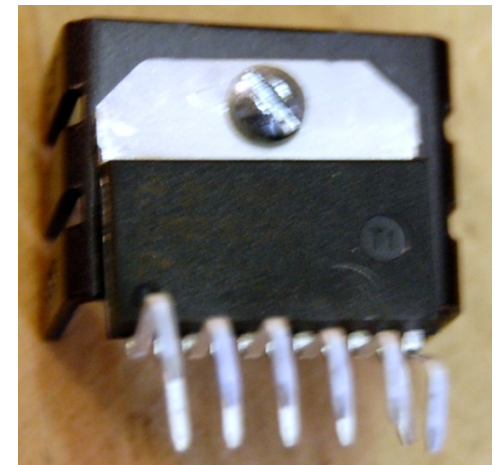
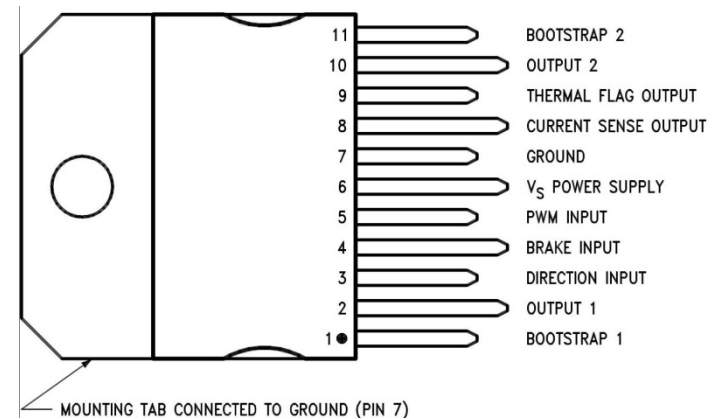
# Complications

- Originally, I thought this project would be extremely easy. But after I built it I notice that my original H-bridge, the L293, couldn't sustain current output from the 12V source to the linear actuator.
- Then I moved on to the idea of using transistors to build an H-bridge.
- N-channel and P-channel MOSFET's were used.
- The crude system schematic in the image worked for a small 9V motor, but not the 12V linear actuator.
- As it turns out, the P-channel MOSFET's need a 12V signal to turn on and off; however, standard 5V logic is good enough for a small 9V motor.



# Solution

- Hours were spent trying to get the MOSFET system to work (turns out 6 MOSFET's are required)
- Hairs were pulled, parts thrown, so I moved on.
- I decided to give the LMD1800T H-bridge a try, although I thought it had a slim chance of working.
- Miraculously the H-bridge did the job, although the soldering was tedious.



# Pricing

• Trash Can	\$15.00
• Linear Actuator	\$55.00
• PIR	\$10.00
• Soldering Work (PIR)	\$37.50
• 12V Power Supply	\$25.00
• Misc. Components	\$10.00
• Est. Gas (driving around)	\$25.00
• <u>Total</u>	<u>\$177.00</u>
• Total Hours Spent	30



# Advice for Others

- Get started as soon as possible. I waited until 10<sup>th</sup> week to put together the project. Even though I'm sort of proud that I completed the it in two days, this resulted in exorbitant spending (retail instead of eBay) and more stress than necessary.
- Be sure to know every detail about the circuit components. For example, the P-channel MOSFET's need a 12V signal to pass a 12V current source. Knowing this would have saved me hours of frustration.
- Take care to use snubber diodes, and not to touch leads together. I fried many components, including a PIC, because of this.
- Don't touch hot components. I didn't think that these things could get so hot. I still have a brand on my finger from touching a mis-wired MOSFET.
- Have a lab partner (I worked alone - not as fun as I thought).

Tim Harreld

Sean Hannon

# ME430 Mechatronics Project

Launching a Model Rocket

## Inputs

- Button 1
- Button 2
- Button 3
- Button 4
- Laser Pointer  
/CDS Cell

## Outputs

- Raise Rocket
- Lower Rocket
- Start Countdown, at 0 ignite rocket
- Interrupt and reset countdown
- Keep Countdown from starting



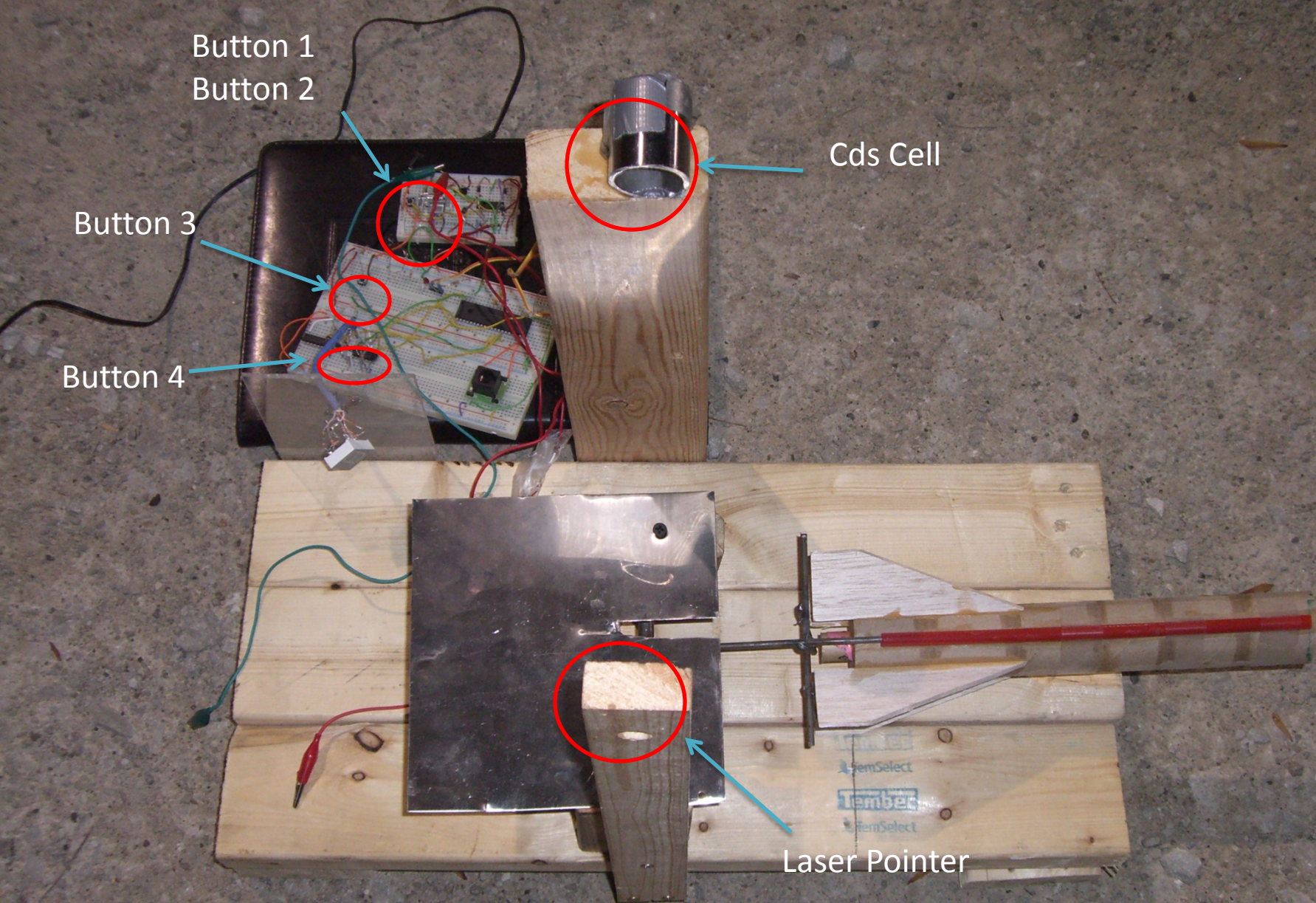
Button 1  
Button 2

Cds Cell

Button 3

Button 4

Laser Pointer



# Using the PIC

- The microcontroller was responsible for managing the input of the Cds cell, ignition button (Button 3) and reset button (Button 4)
- Analog to digital conversion (ADC) was used to define the input of the Cds cell. A value of 600 was used as the threshold for “seeing” the laser pointer and refusing to launch
- Button 3 was a digital input
- Button 4 threw the flag for an interrupt command which reset the countdown



# Closer Look at Interrupt

- Originally when the interrupts flag was thrown the state was reset to 10
- This was inadequate because after returning to the “for” loop, “state” would be reset again, and countdown would resume
- Adding new variable “reset” and nesting the “if” statements solved the problem

```
if (state == 9)
{
    PORTD = 9;          //
    Delay10KTCYx(100); //Delay for 1 second
    state = 8;
    if (reset ==1)
    {
        reset = 0;
        state = 10;
    }
}
else if (state == 8)
{
    PORTD = 8;
    Delay10KTCYx(100);
    state = 7;
    if (reset ==1)
    {
        reset = 0;
        state = 10;
    }
}
}
//*****
if(INTCON3bits.INT2IF)
{
    INTCON3bits.INT2IF = 0;
    //PORTD = 9;
    //PORTBbits.RB1 = 1;
    //state = 10;
    reset = 1;
}
```

# For Future Classes

- Plan to buy heat sinks and voltage regulators
- When in doubt buy the bigger power supplies and make sure parts being purchased are the correct size aka. Bread boards.
- The NPN transistors provided are useless! Go with the MOSFET, they can handle greater currents.

# For Your Viewing Pleasure

