



# Beverage Carrying Car

Adam Beals

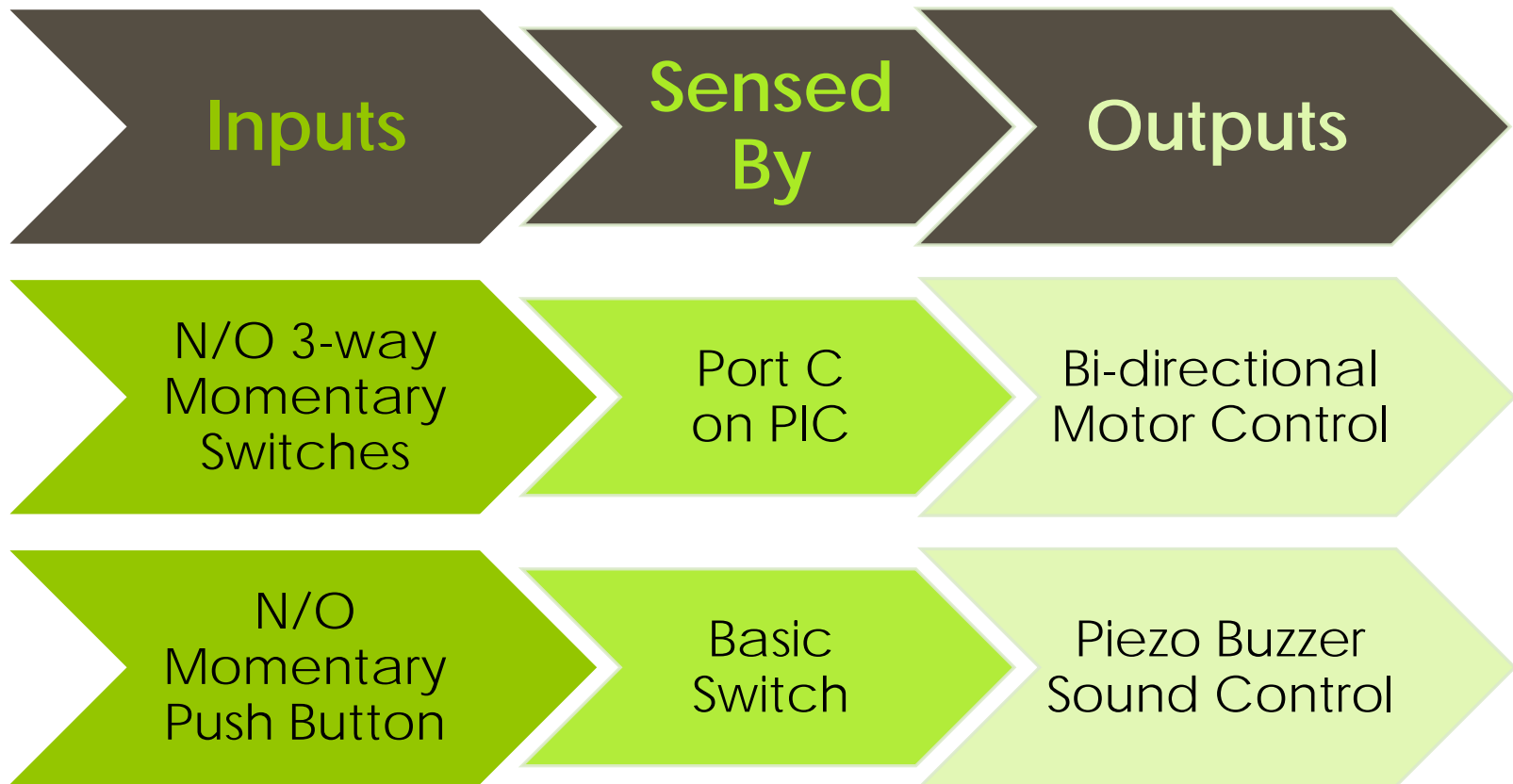
Tony Bowden

---

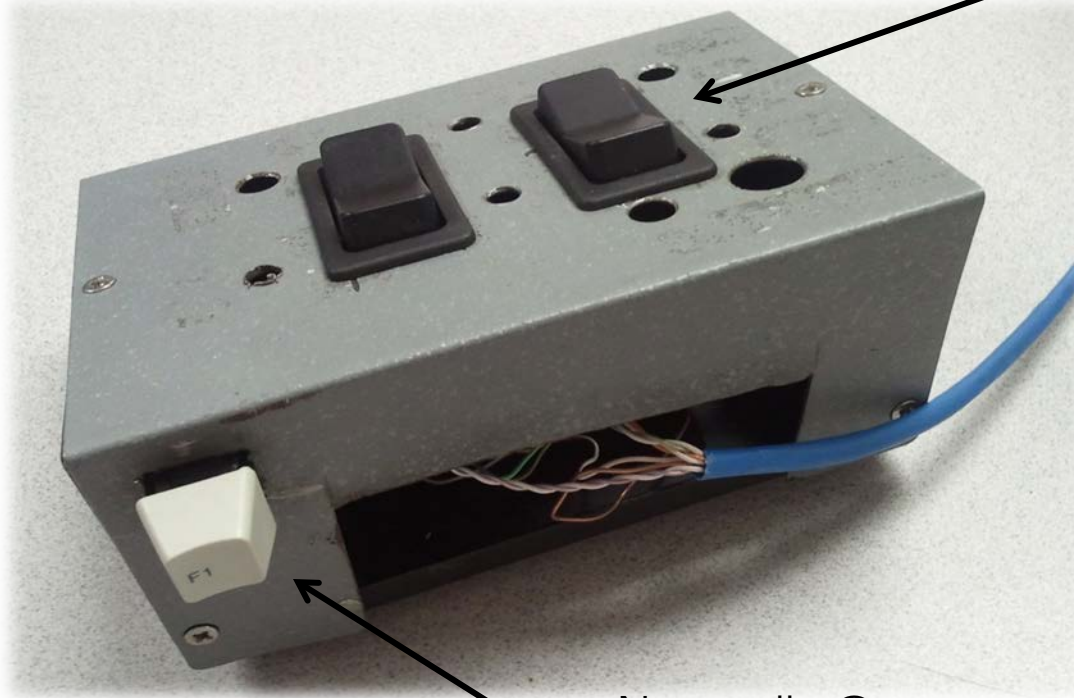
# Objective

- Create a line driven car capable of carrying beverages
- Implement two geared DC motors to drive wheels
- Incorporate a Piezo buzzer to alert passersby of car

# Function Flowchart



# Inputs



Normally Open  
Spring-to-Center  
3-way Momentary  
Switches

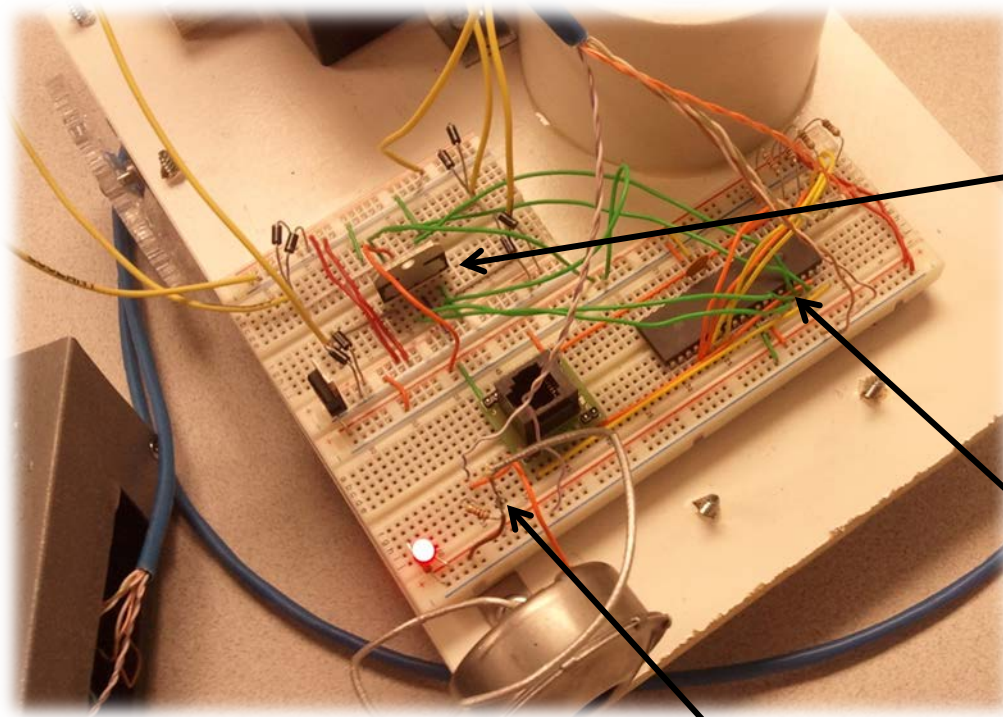
Normally Open  
Momentary Push  
Button



# General PIC Function

- Inputs from all switches read through Port C
- All drive signals were inverted and fed out through Port B
- No PWM was necessary due to mechanical design of our chosen buzzer
- Drive signals were output to an H-Bridge
- No interrupts were necessary

# OUTPUTS



H-Bridge  
(Motor Control)

Drive Signal  
(Port B)

Switch Circuit  
(For Piezo Buzzer)

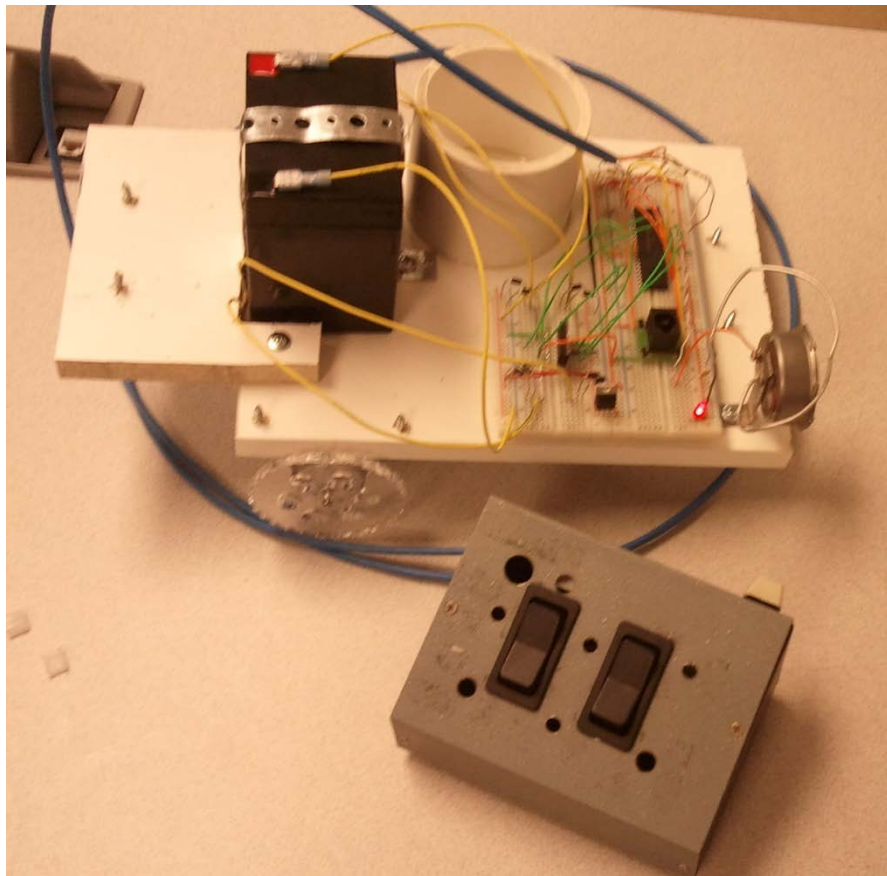
# Results

- Beverage Carrying Car functioned as planned
- Speed of car was acceptable
- Controllability was satisfactory

# Advice

- ◉ Pick a simple yet intriguing project
- ◉ Start work early and finish early
- ◉ Do not underestimate the time necessary to complete a project
- ◉ Establish well defined goals and constraints

# Beverage Carrying Car!!!



# Mini Air Hockey Table

*Cami Beddow & Bailey Wagner*

# Project Description

A miniature air hockey table will be adapted to visually display the score of each team through the use of two 7-segment displays. Once a puck goes through a goal, an IR sensor will be activated, and consequently adding a point to the appropriate team's 7-segment display. Each time a goal is scored, a buzzer will sound once. After a team reaches nine points to win the game, LEDs that are in the shape of a 'W' will flash to show the win. A reset button will also be provided to restart the game at any time.

# List of Hardware

Part Name	Quantity
Miniature Air Hockey Table (with 2 pucks & 2 mallets)	1
18F4520 Microcontroller	1
7-Segment Display	2
7-Segment Driver	2
LEDs	11
330 $\Omega$ Resistors	25
Breadboard (with wires)	1
5V Regulator Chip	1
Infrared Emitters & Detectors	2
Piezo Buzzers	1
MOSFET	1
Push Button	1
1K $\Omega$ Resistor	2
Darlington	2
Decoupling Capacitor	1



# Inputs & Outputs

## ● Inputs

- Puck passing goal
  - This breaks the IR sensor which consequently changes the IR sensor output from being low to high which is connected to Port B and causes an interrupt to be executed
- Reset button
  - When pressed, causes an interrupt to be executed

## ● Outputs

- When the input is sensed of a puck passing through a goal
  - A point is added to the appropriate team's 7-segment display
  - A buzzer is sounded for half a second
- When a team reaches 9 points and therefore winning the game
  - 11 LEDs shaped in a 'W' flash five times
  - After the five flashes, the score boards are reset

# The PIC As The Controller

## ● Interrupts

- The IR Sensors are being used as digital inputs. When the sensor is broken and therefore changing the port from being low to high, a port B interrupt is triggered which adds a point to the 7-segment displays.
  - RB0 and RB1 were used for the IR sensors
- The push button to reset the score is programmed as an interrupt as well
  - RB2 was used for the push button
- All interrupts used were high priority

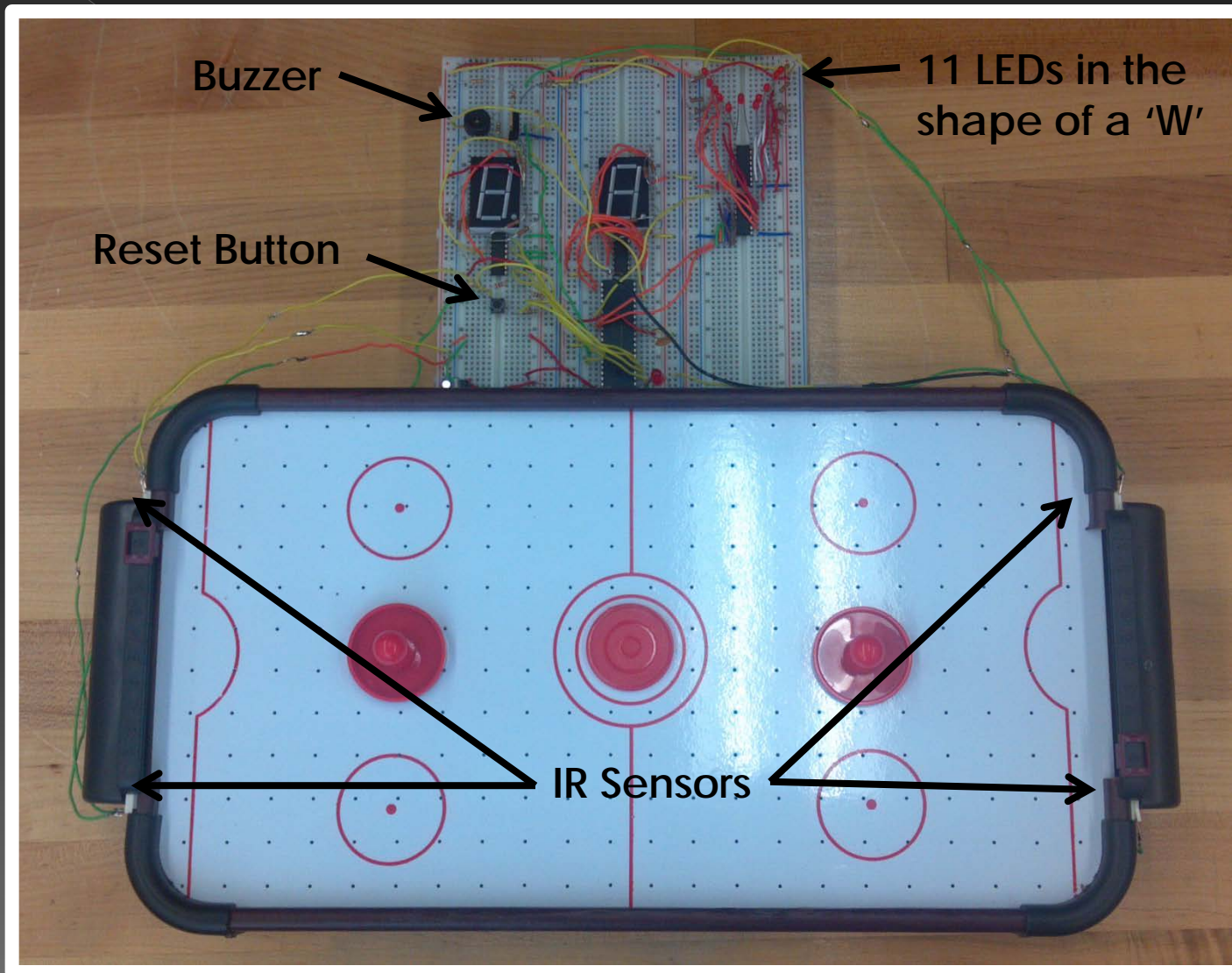
## ● PWM for the buzzer

- A value of 158 was used to set the frequency and a value of 102 was used to set the volume of the buzzer

## ● Delays

- A delay was used in order to sound the buzzer for half a second

# Finished Design



# Results

- Everything worked as planned
- The only difficulty faced was that one of the IR sensors became misaligned during transportation to the presentation, causing it to not detect the puck when it went through the goal

# Advice

- ◉ Instead of using the IR sensors as digital inputs, the scoring system may work better if they were implemented as analog inputs with a threshold set by a potentiometer like the robot. This would account for changes in light intensity and therefore better control when the IR sensors “see” the puck.

# Interactive Beer Pong Table

Jamal Bell

Chad Braun

# Objectives

- Allow the user to select one of seven possible reracks.
- A green LED lights up to indicate the spot a cup should be placed.
- When a cup is removed, the LED changes to red.

# Inputs and Sensing

- Push Buttons
  - Multi-colored push buttons were used as switches to allow the user to select their rack of choice.
- IR Sensors
  - Infrared sensors were used to detect the change of light once a cup was removed.
- Potentiometer
  - A potentiometer was used to set the threshold of the IR sensors. The user would be able to change it depending on the lighting in the room.



# Controlling 3 PICs

- First Attempt
  - At first, 3 buttons were attached to each PIC (1 button on the 3<sup>rd</sup> PIC), and each switch was a PORTB interrupt (INT0-2). The theory was that when one interrupt went off, that PIC would send a signal to the other two PICs to let them know which button was pressed. Unfortunately, communicating between PICs with only HIGH/LOW signals turned out to be very difficult.
  - A series of timers were used to communicate between PICs. A PIC sending a signal would send a HIGH buffer, delay 5ms, then send a HIGH signal followed by an 8ms delay corresponding to the switch pressed. If INT0 was triggered, then there would be one HIGH signal followed by a LOW, if INT1 was triggered then two HIGH signals would be displayed, and as so with INT2. The PIC receiving the signals would be activated by the buffer, delay 10ms, then each time it received a HIGH signal, a variable would increment and then delay for 10ms. When it received a LOW signal, the incrementing would stop and the PIC would know which rack to activate.

# Controlling 3 PICs

- Second Attempt
  - After many attempts at the previous method, we decided to attach all 7 buttons to each PIC; therefore, all PICs would simultaneously know which rack is being selected. Instead of using interrupts, we used if statements to determine if the button was pressed. After this change, the rack selection worked very well.

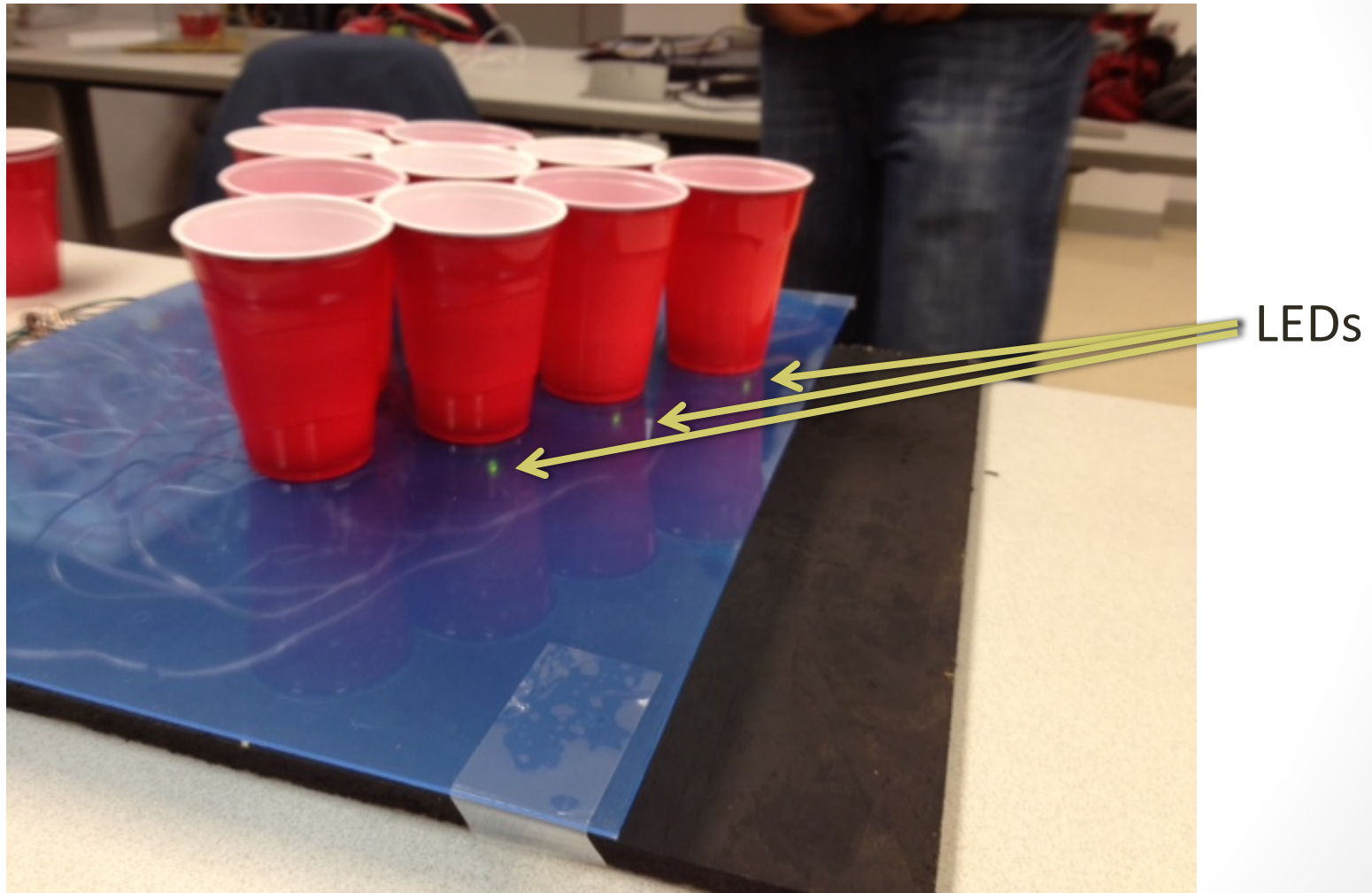
# Outputs

- LEDs
  - Displays the chosen rack
  - Indicates where to place cups with a green light
  - Indicates where a cup was with a red light
  - Indicates which rack was chosen (beside the push button)

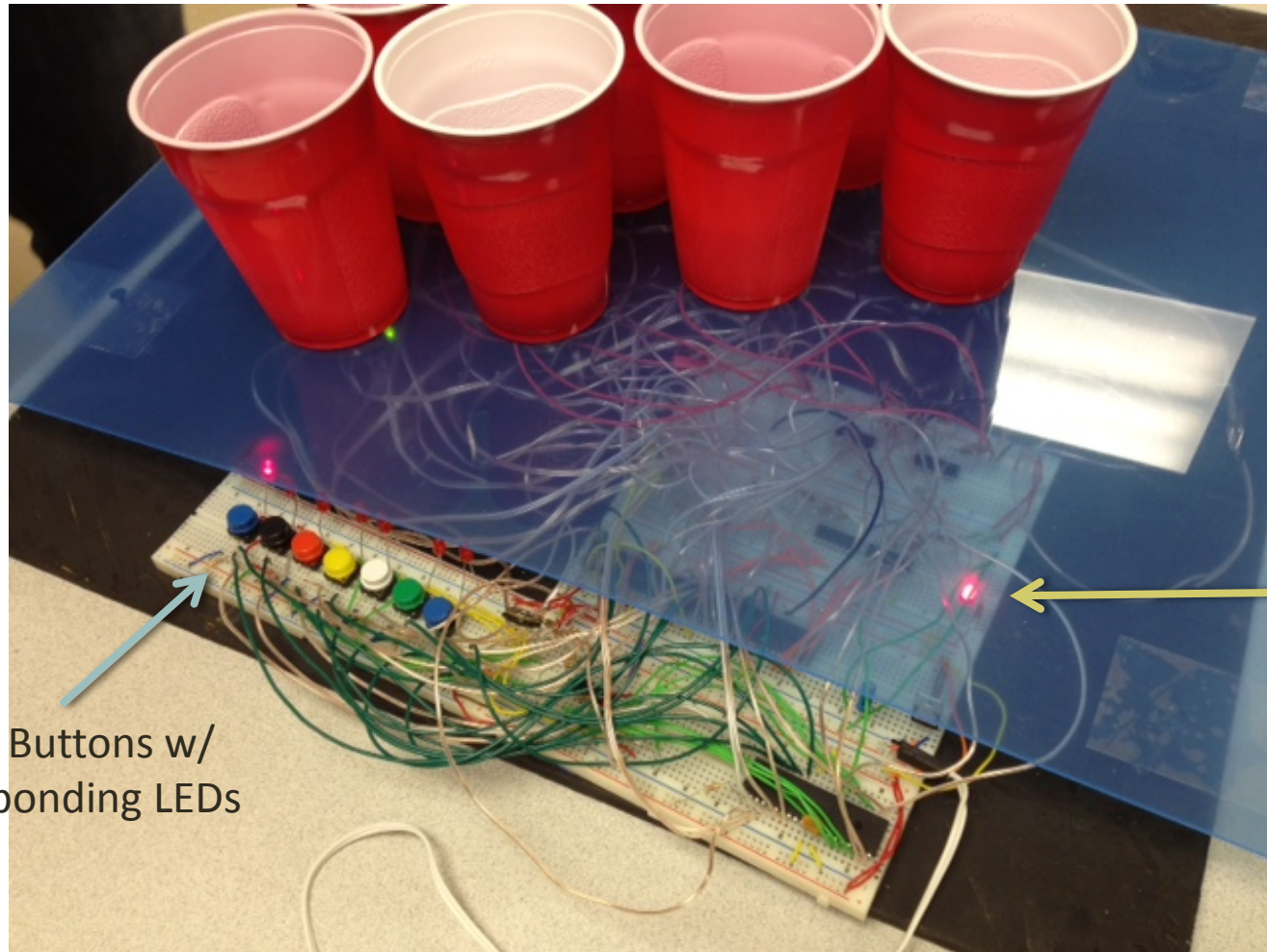
# Results

In the end the project did not work completely as planned. The rack selection buttons worked and so did the LEDs that indicated where to place the cups. However, when we removed the cups the IR sensors were not properly sensing the cup had been removed, which should have changed the LED from green to red.

# Working Model



# Working Model



Rerack Buttons w/  
corresponding LEDs

Power  
Indicator

# Advice

- Pick a project that doesn't have so many inputs and outputs. The wiring becomes a mess.
- Try and have the project finished a couple days early so you have time to work out any kinks.
- Spend time creating your hardware. If your hardware is designed well then the electronics function a lot better or are more organized.
- Have a really neat wiring diagram. Once you gain more knowledge you might want to revise it, which will ultimately help with breadboard space and wiring complications.

# BONSAI BOT

---

Kyle Bristol & Brandon Kmetz



# Overview

- Objective
- Inputs/Outputs
- Implementation
- Results
- Advice

# Objective

- The objective of this project was to create an automated system for taking care of a plant (bonsai tree) over a quarter break
- The system needed to water the plant, provide it with artificial sunlight, and (if necessary) regulate the plant's temperature



# Inputs/Outputs

- Analog Inputs:
  - Ambient light level (sense by a photoresistor)
  - Temperature (sensed by a thermistor)
- Digital Inputs:
  - Water level (full/empty, sensed by a SPDT switch used as a float sensor)
  - Set mode/temperature controls (pushbuttons that let the user choose a minimum allowable temperature for the soil)
- Digital Outputs:
  - Control signal to activate light/heat source
  - Control signal to activate water pump
  - Display signals to 7-segment displays (temperature display)

# Implementation

- The primary feature of the PIC that we used was analog-to-digital conversion.
- ATD conversion allowed the PIC to detect the ambient light level; once the light level passed an adjustable threshold, the light turns on. Similarly, it turns off when there is darkness.
- We used ATD to display the soil temperature as well
- The water level input was checked each time the main loop cycled through, as were all other inputs.
- No timers or interrupt service routines were used

# Water Level Sensor & Full Setup

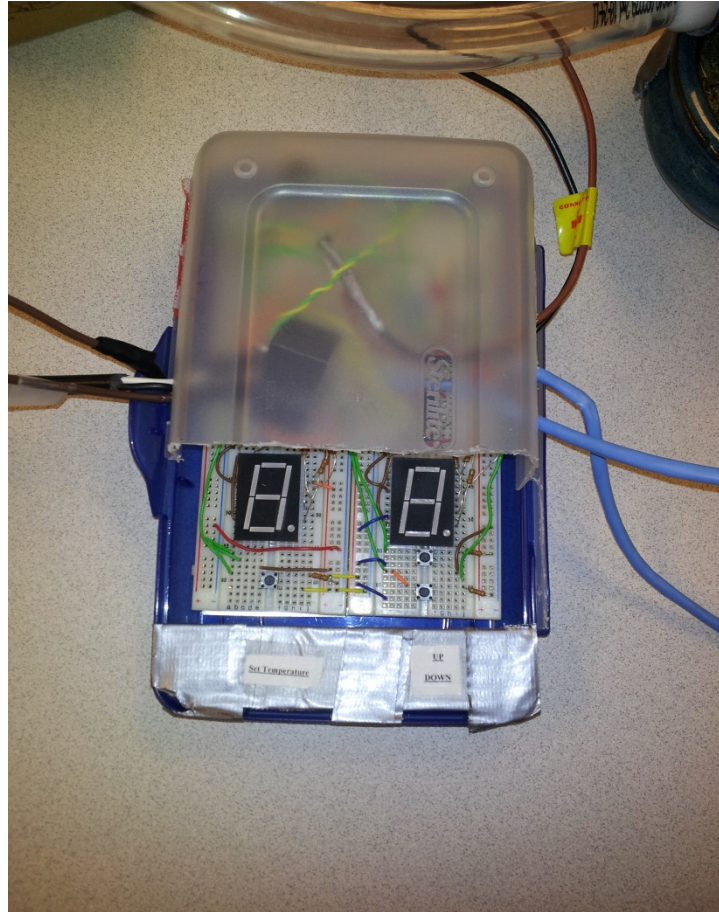


Float Sensor



Fully Connected Device

# Front Panel



Temperature Controls

# Results

- Everything worked as we intended it to! Wiring the project on a breadboard caused some problems that could only be fixed by taking out wires and putting them back in the same hole...



# Advice

- Order parts well in advance and make sure they are what you actually intended to purchase!
- A mislabeled part can literally cost you days of work.
- We had to dismantle our original perfboard project and use the contents of our lab kit because the display we purchased was not common-anode.
- Make sure you test your components individually before assembling the project
- If you use kickback/snubber diodes, be sure to put filter caps across your power rails or the high frequency noise will affect unregulated parts of your circuit





# Drink Bot

Hannah Chapin-Eppert

Mark Herber



# Objective

- Make an automatic drink dispenser to mix pre-determined drinks in various volumes



# Inputs

- The inputs for this device are five push-buttons.
- 4 buttons controlled drink number and drink volume, each using 2 buttons to increment their values
- 1 button sent a “start” signal to begin dispensing a drink



# PIC as a Controller

- 5 interrupts were used to trigger actions on button presses
- Once the dispense signal was given, signals were sent to MOSFETs driving the 3 solenoid valves
- A timer interrupt was utilized to control the period of time that each valve remained open
- The PIC also controlled 7-segment displays, which displayed the drink number and size to the user

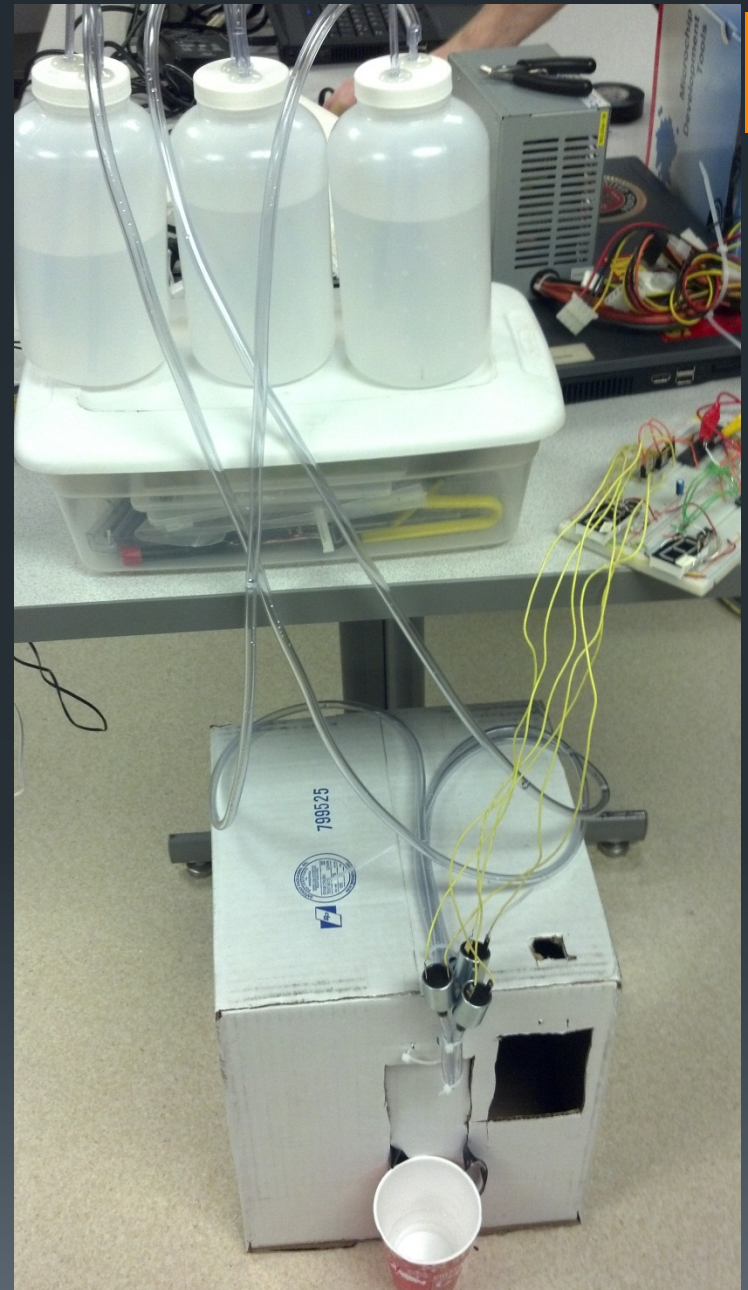


# Outputs

- Outputs from the PIC included:
  - Drink size and drink number on 7-segment displays
  - Signals to MOSFETs to drive the solenoids
- The physical outputs were mixed drinks:
  - Up to 3 liquids can pour simultaneously in various proportions to complete the chosen drink and volume

# Results

- After a frustrating calibration for the amount of time required to dispense one ounce of liquid, the device functioned perfectly



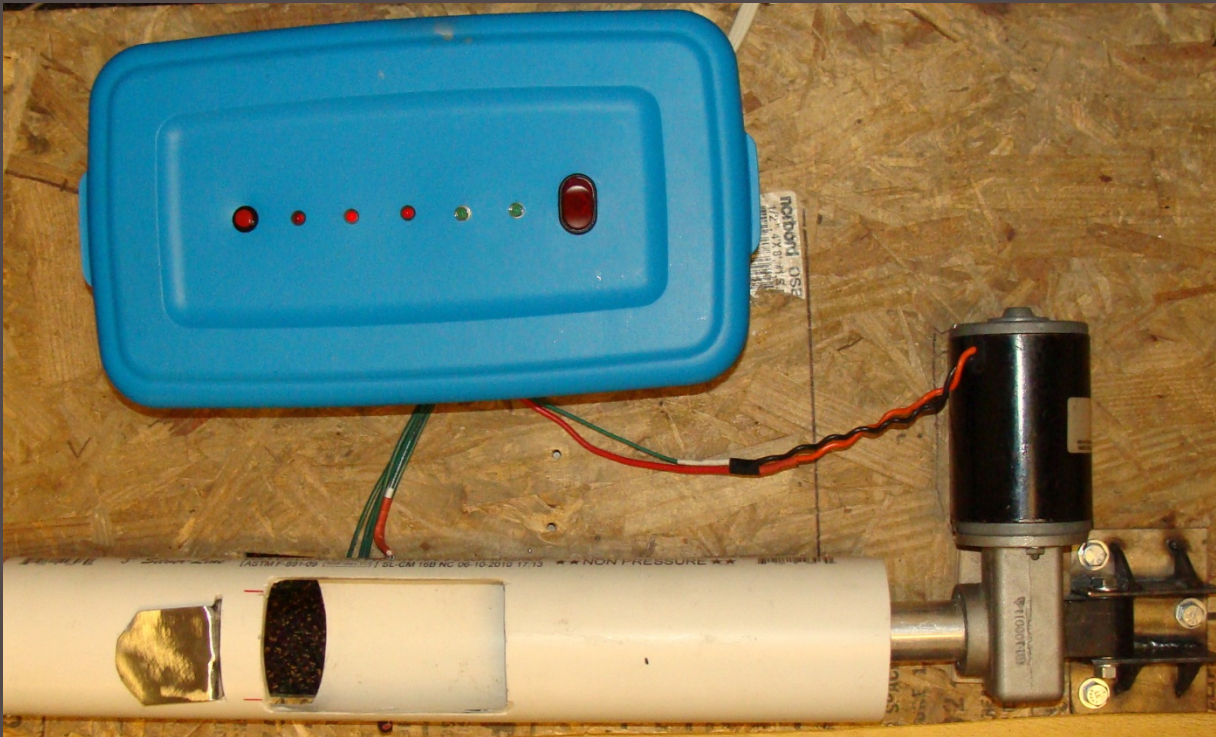


# Advice

- Choose a project that you are excited about, or something that you personally want to own
- Start early to allow time for troubleshooting and finishing your project to make it look nice
- Even if your code builds without errors, that does not necessarily mean it is doing what you expect
- Incrementally build and test your project, otherwise it will become too complicated to easily troubleshoot

# CAN SMASHER

Jep Cohen  
Malhon Godwin  
Daniel Watson





# OBJECTIVES

**Safely**

**smash a can**

**Consistency**

**smash a can**

**Reliably**

**smash a can**

**Automatically**

**smash a can**

# INPUTS

- Power Button – Controlled power to the PIC Chip
- Go Button – Started the can smashing sequence
- IR Sensor – Verified that a can was in the chamber
- 2 Fault Inputs – Outputs from motor controller for debugging
- Potentiometer inside case – Used to calibrate IR detector

# PIC PROGRAMMING

- Analog to Digital - Used to interpret infrared detector and calibrate reading with potentiometer
- PWM output used to control motor controller operating the linear actuator
- RBO interrupt used to start can crushing process
- Extending and contracting controlled with internal timer
  - Continuously checks for fault codes and will stop crushing and enter a standby mode if a fault is detected

# OUTPUTS

## ■ State LEDs

- Power LED
- Can present LED
- Program Running LED
- Two error LEDs

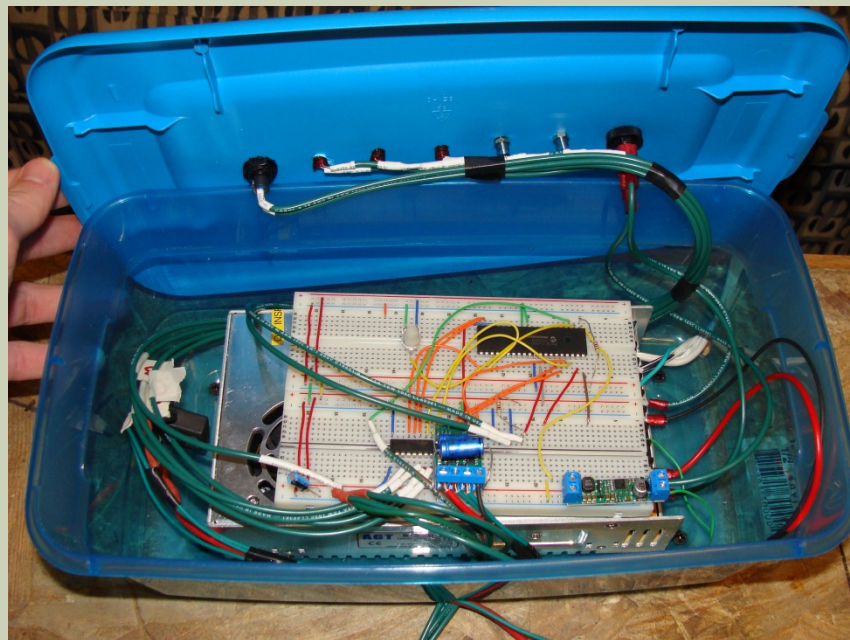
## ■ Linear Actuator

- Digital output to controller to control direction
- PWM output controls speed of motor with duty cycle

# RESULTS



# PICTURES



# ADVICE

- Start Early
- Do not use IR Sensors
- Always use the plasma cutter! And maybe do some welding too...

# ME430 Mechatronics Project: Memory Challenge!

By: Chad Conway and Blake Vermillion

18FEB12

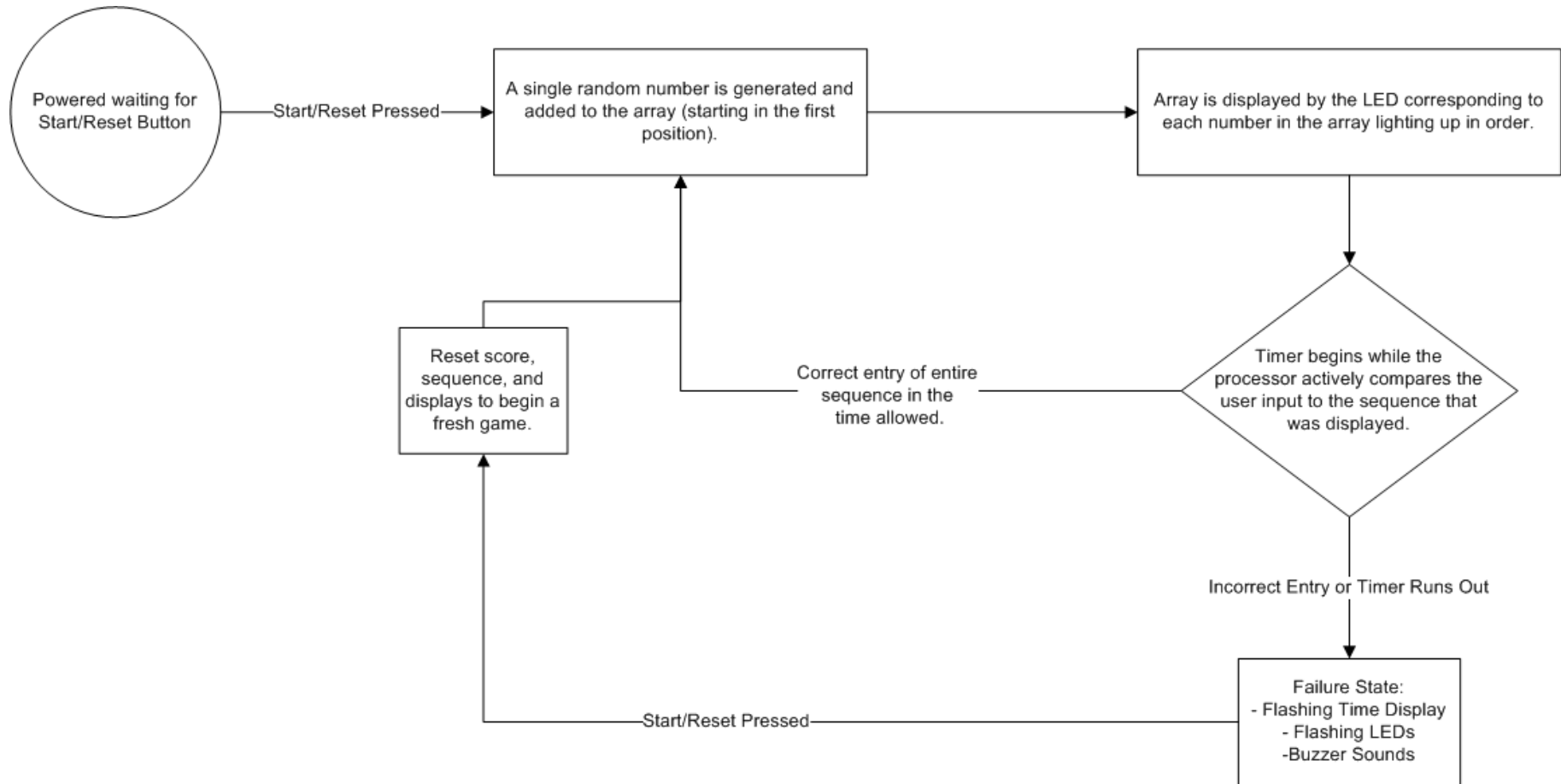




# Project Objective:

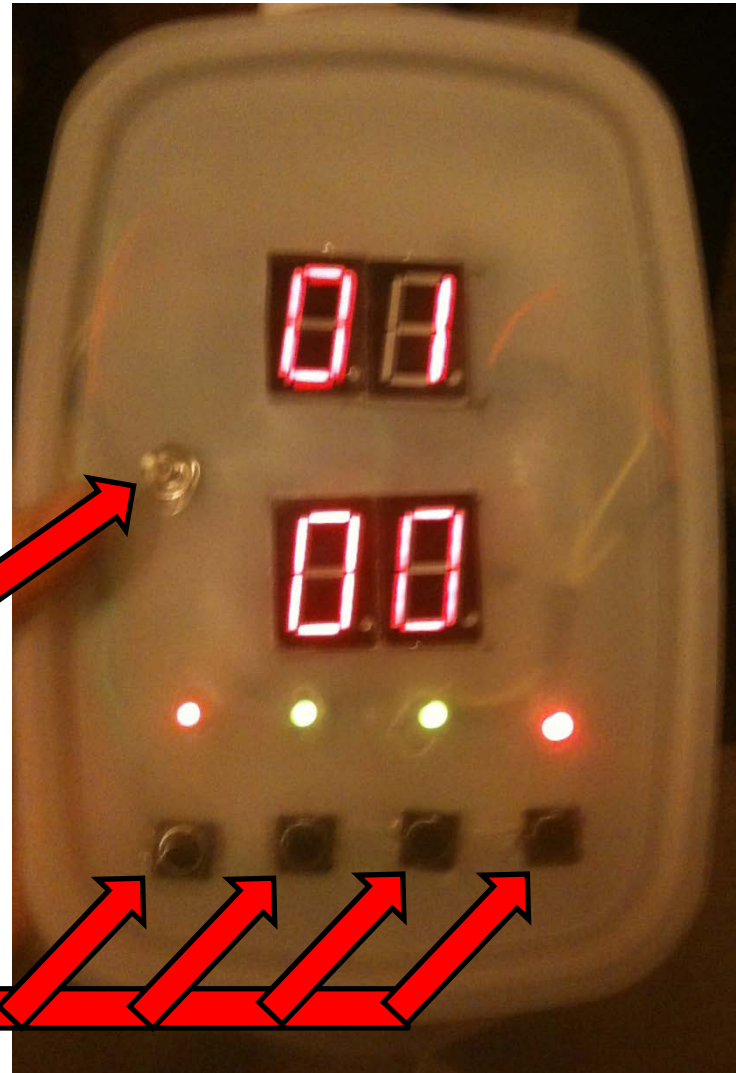
This is a fun game that will challenge your memory skills. It consists of a set of four lights and a button to go with each light. Once the start/reset button is pressed the processor will generate a random pattern starting with a single step. Then the operator will repeat the pattern. If the pattern is repeated correctly in the time allowed then the processor will add one more step on the end of the pattern. Then the process will be repeated. If the pattern is repeated incorrectly or the time runs out, the time and LEDs flash until the start/reset button is pressed while the buzzer sounds will

# Project Flow Chart



# Inputs

- 5 Push Buttons
- All sensed on falling edge (when the signal voltage dropped to 0)
- Start/Reset was set up as an interrupt
- Four others were normal inputs

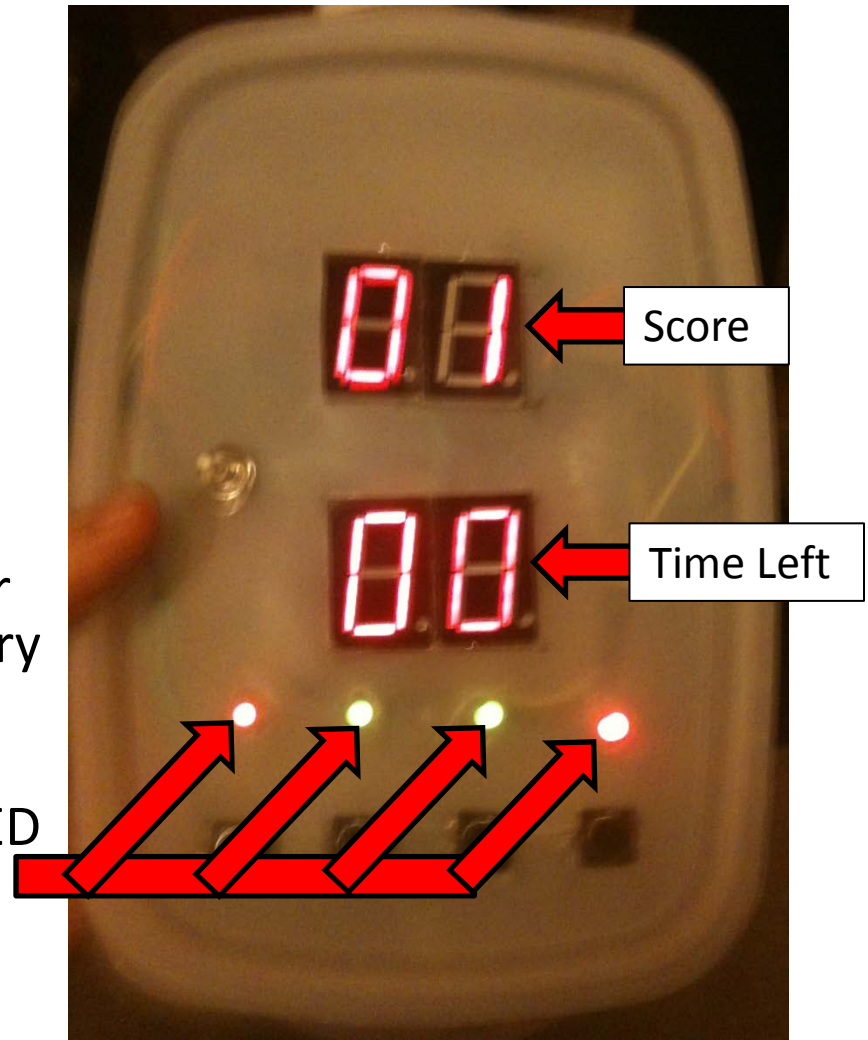


# PIC Controller Use

- Random Number Generator
- Array storage
- Two Timers
  - one for time count down (when active it subtracts from time at each overflow low priority interrupt)
  - second for PWM control for buzzer
- Compares input to the generated array
- Start/Reset high priority interrupt to resets displays, score, time, and sequence on falling edge and begin the game again.

# Outputs

- 4 Seven Segment Displays
  - 2 for Score
  - 2 for Time
- Score is a global variable that gets added to and displayed each time the user successfully inputs the array.
- Time counts down during the user input. It is reset and added to every second successful sequence.
- Four LEDs that are used to display the sequence. In the code each LED is associated as 1-4 and based on the number generated the corresponding LED lights.



# Results = Success!

- The entire game runs consistently without problems.
- The buzzer was not part of the design, however it was added as an extra as suggested as an option by Dr. Olson.

# Advice

- Find a project that is relatively easy in concept to ensure that it is doable.
- If you complete it quickly, add extras! (like the buzzer we added)
- Get started on it right when it is assigned that way there is plenty of time to debug throughout the quarter.

# Hole-in-One Counter



By Derek Dauenbaugh and Nick Mohr



# Project Objectives

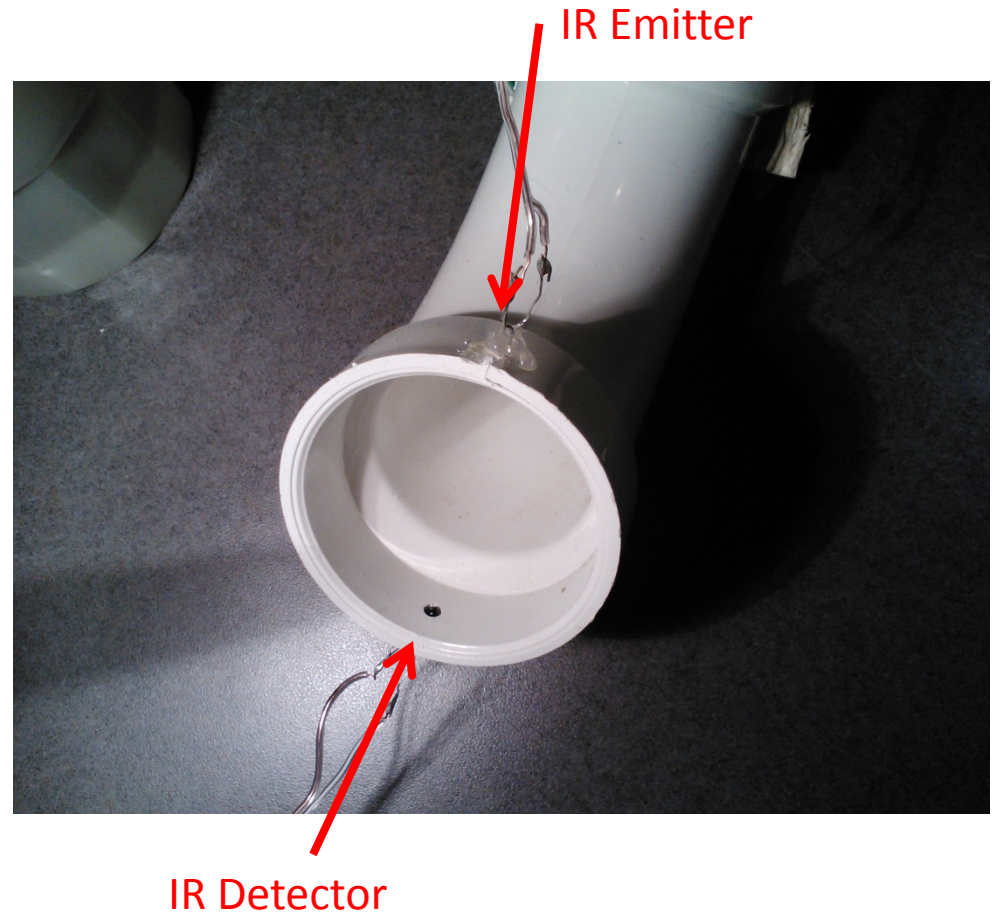
- Create a putting machine that would return a MADE or MISSED putt back to the putter
- Provide a scoreboard using 7-segment displays that would count the amount of MADE putts and MISSED putts
- Have a reset button that would reset the scores of MADE and MISS to zero

# Used digital inputs controlled by IR sensors

High all the time

Low when ball  
passes through

Triggers internal  
counter

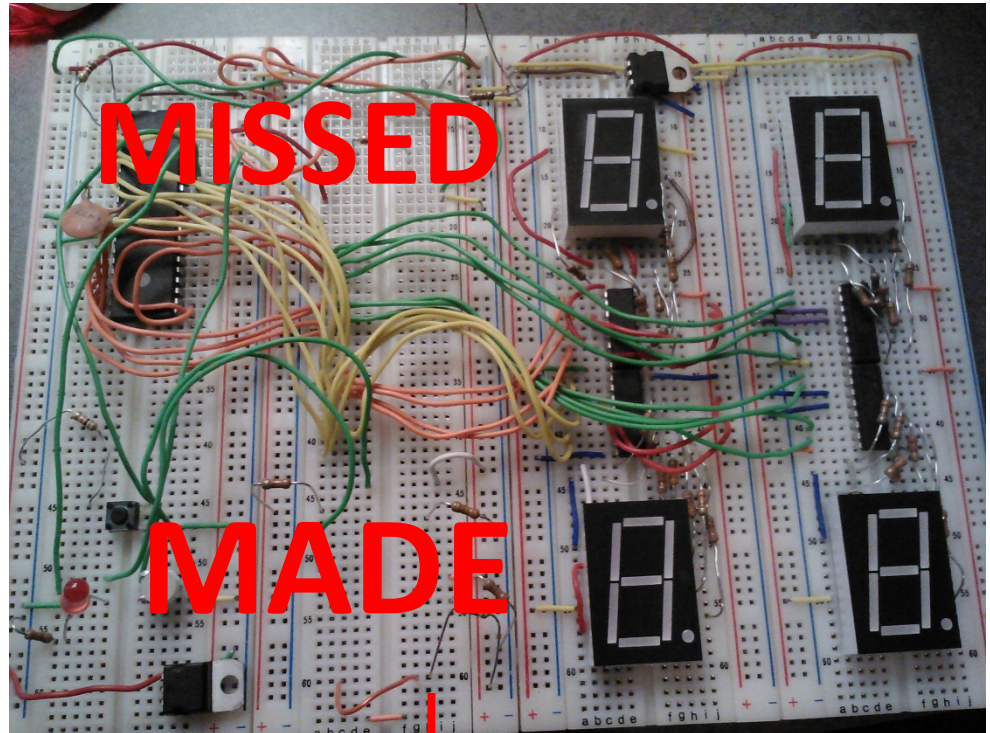


# PIC controlled 7 segment displays

- PORT A controlled 10s digit for both MADE and MISSED values
- PORT C controlled 1s digit for MADE
- PORT D controlled 1s digit for MISSED
- PORT B was inputs from IR sensors and reset button
- Ran electric motor constantly
- While loop ran that called to MADE/MISS functions when a PORTB sensed that it was low
- MADE/MISS functions incremented higher the number of putts made and missed and programmed to respective port to display the counted value

# Outputs

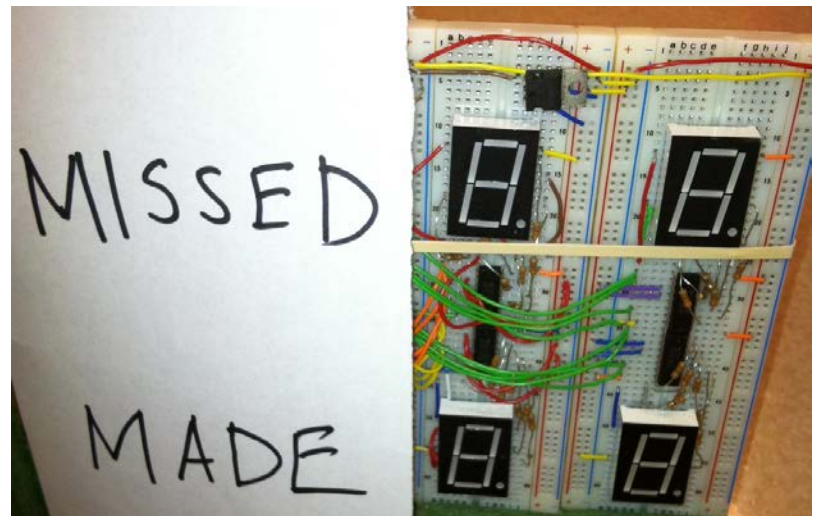
- MADE and MISSED displays reset to 00 when reset button pushed
- Displays increment when ball interrupts MADE or MISSED sensors
- Electric motor turns wheel that propels golf ball back towards golfer



Output to motor

# Results

- Motor spun disk that returned ball at all times
- MADE and MISSED displays worked and would count up to 99
- Reset Button worked every time



# Future Advice

- Begin the Project Before 8<sup>th</sup> Week. It will make life easier towards the end.
- Don't try and do too much. Pick a feasible project that is doable along with the work for the class.
- Make a plan and schedule and try to stick to it as much as possible.
- Avoid too much light when dealing with IR sensors.



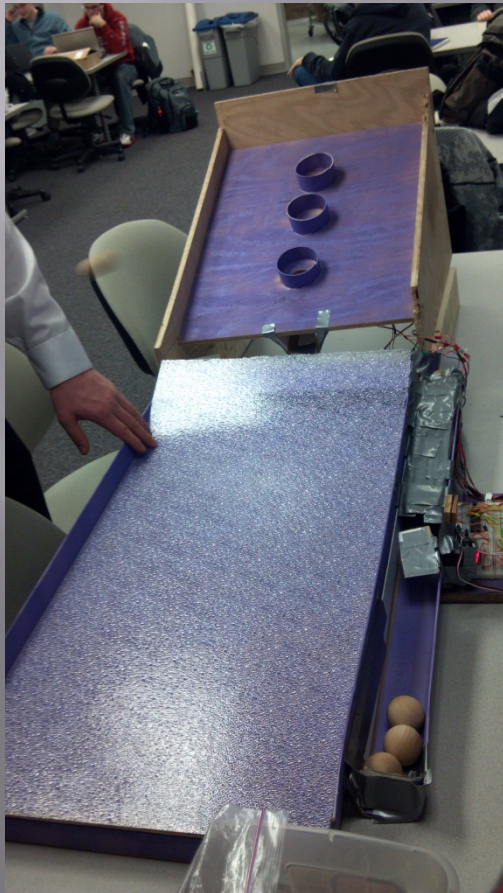
# MINIATURE SKEEBALL MACHINE

ME430-01

Matt Green

Ethan Nash

Ben Parker



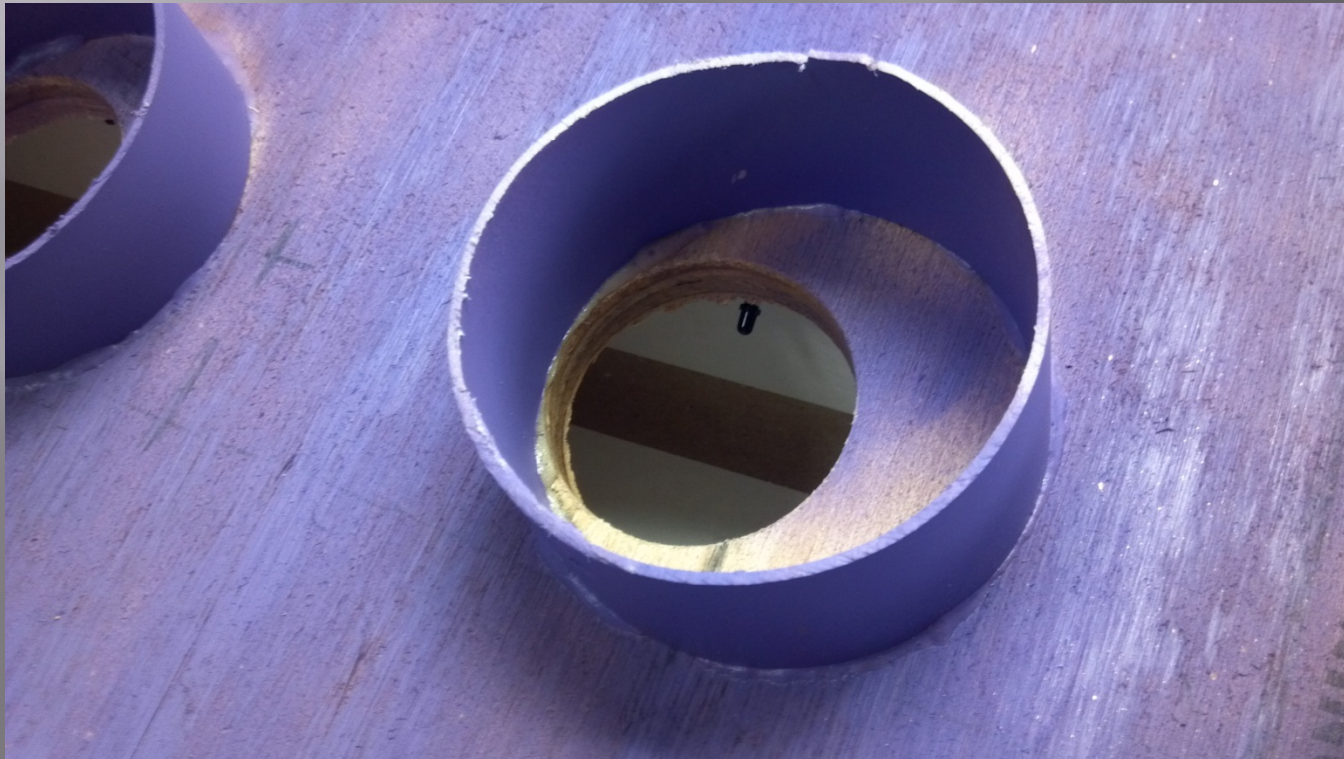
# Goals

- ▣ The goal of the project was to make a functional skeeball game that would have three different “holes” to shoot for, each varying in point values. The game would keep track of your tallied points until you run out of skeeballs. Upon hitting the reset button, the balls would be released and the score will be reset.



# Inputs

- ▣ Reset Push Button
- ▣ IR Sensors to detect skeeballs going through the holes.

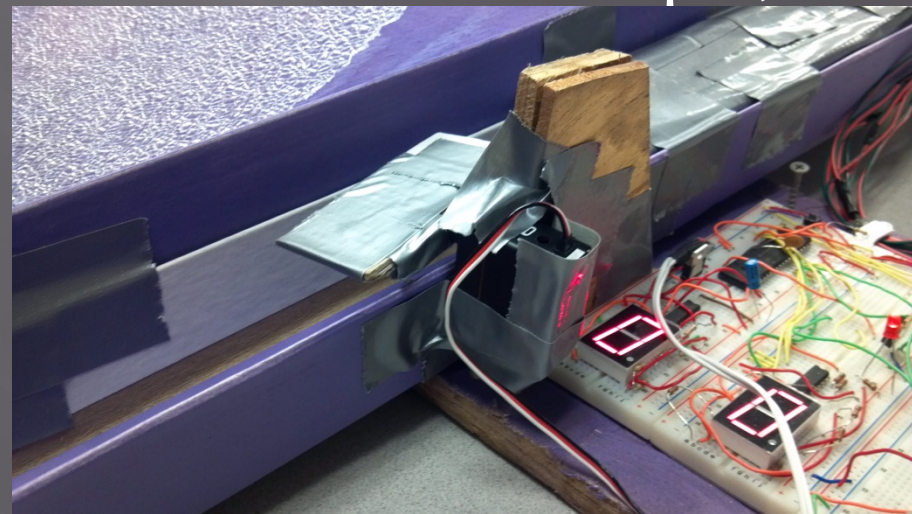


# The PIC

- ▣ Each input signal was used as an interrupt.
- ▣ The IR inputs were used as inputs that would increment the score by a certain amount
- ▣ The push button input was used as an interrupt that not only reset the score but it also output a PWM that was used to turn the servo motor approximately 90 degrees, to allow the balls to roll down.

# Outputs

- ▣ Seven segment display – These would count by 1's, 2's, and 3's depending on which hole you could make the shot in.
- ▣ Motor – Upon pressing the reset button, this would release the skeeballs back to the user. A servo motor was used to make the low torque, fixed rotation turn.





# Results & Recommendations

- ▣ The miniature skeeball machine worked as planned.
- ▣ Device should be built in place because due to size, it is very difficult to move.
- ▣ The IR Sensors are difficult to deal with and require time to calibrate.



A project by JT Hibbard

# SIR MIX-A-BOT

# Automatic Drink Mixer

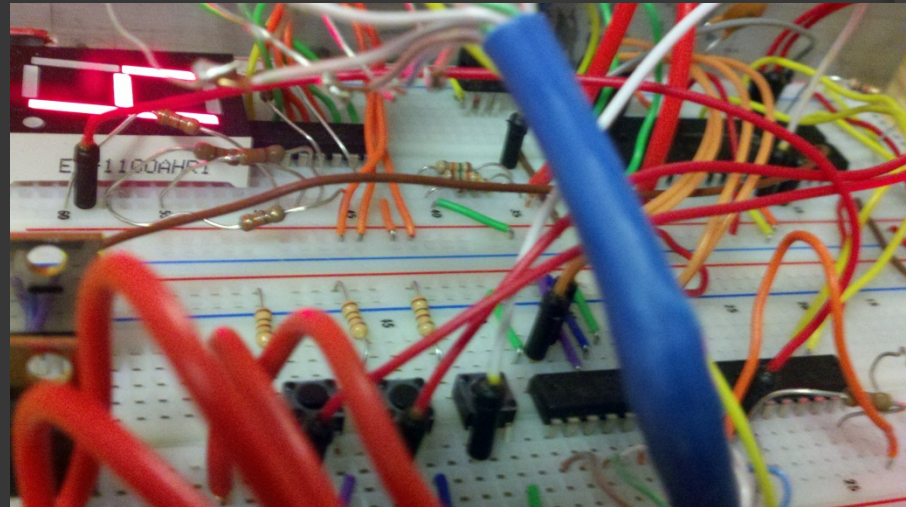
- Mixes your beverage until it reaches the perfect temperature





# Inputs

- 3 Press Buttons
  - Controls the set temperature
  - Start/Stop Button
- Temperature
  - ADC reads voltage across a thermistor
  - Voltage reading is converted to a temperature



# PIC as a Controller

- ⦿ Interrupts
  - Start/Stop button was an interrupt
- ⦿ On/Off switch for the motor
- ⦿ Timers used to prevent infinite loops



# Outputs

## ⦿ Seven Segment Displays

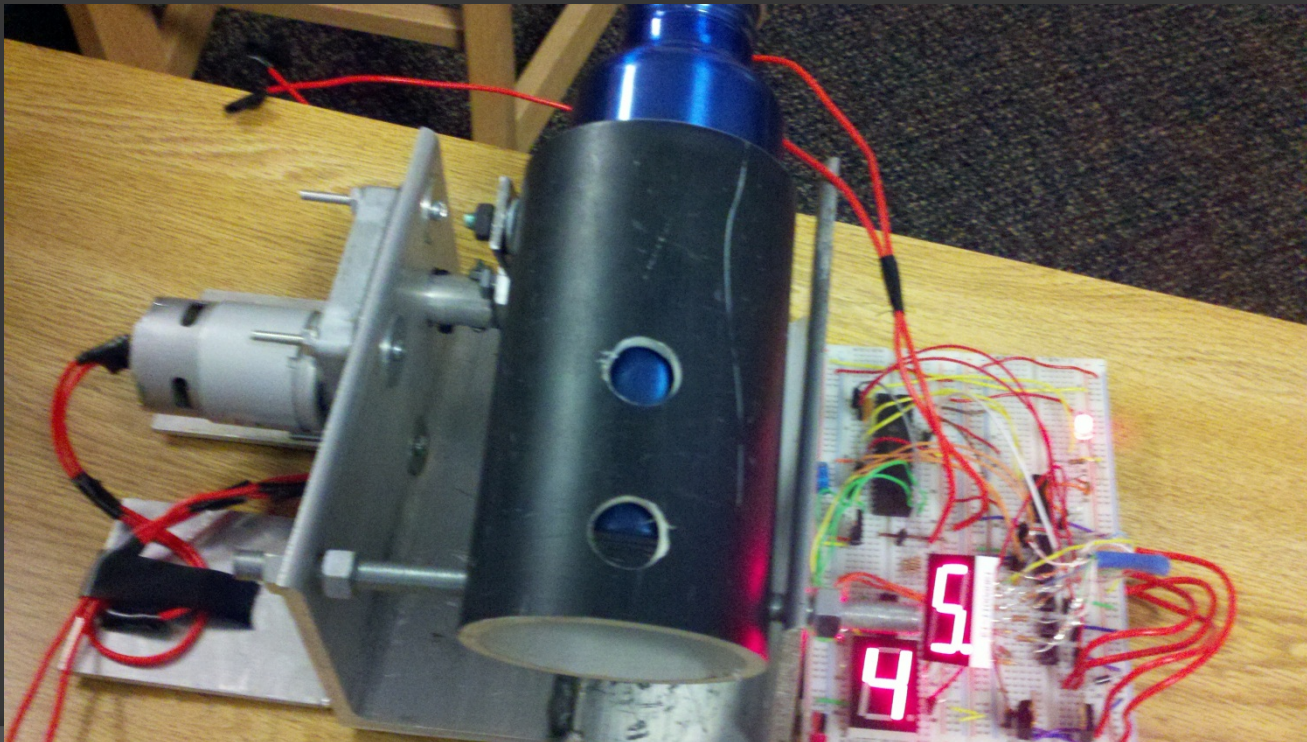
- Shows the set temperature
- Shows the current temperature during mixing sequence
- Sends binary signal to seven segment driving chip

## ⦿ Motor On/Off

- Sends a signal to a MOSFET to switch the motor on or off

# Results

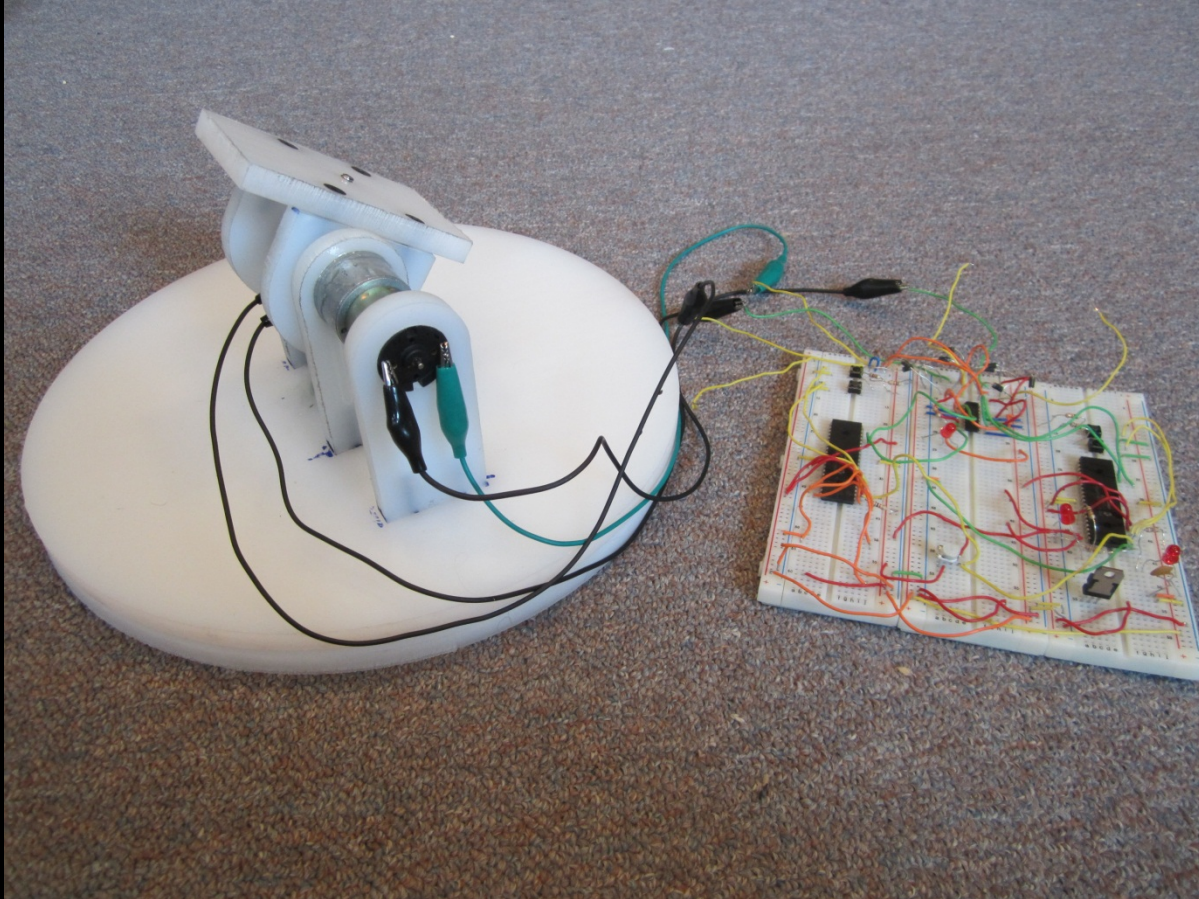
- Works exactly as planned
  - Mixes beverage until the correct temperature has been reached



# Advice for Future Generations

- ⦿ Order all components early and test them immediately
- ⦿ Avoid using IR sensors
- ⦿ Think carefully about the simplest way to do something
- ⦿ You only have a limited amount of spots on the PIC, don't waste them on a bunch of seven segment displays

# Controllable Camera Mount



Kenton Hochstetler, Andrew Schmidt, Adam Reke

# The objective was to create a camera mount

that would

- have a controlled panning motor
- have a controlled tilting motor
- have variable speed for pan and tilt
- be universal



# Inputs to accomplish objective

Button  
Press/Hold

Tilt Down

Button  
Press/Hold

Tilt Up

Button  
Press/Hold

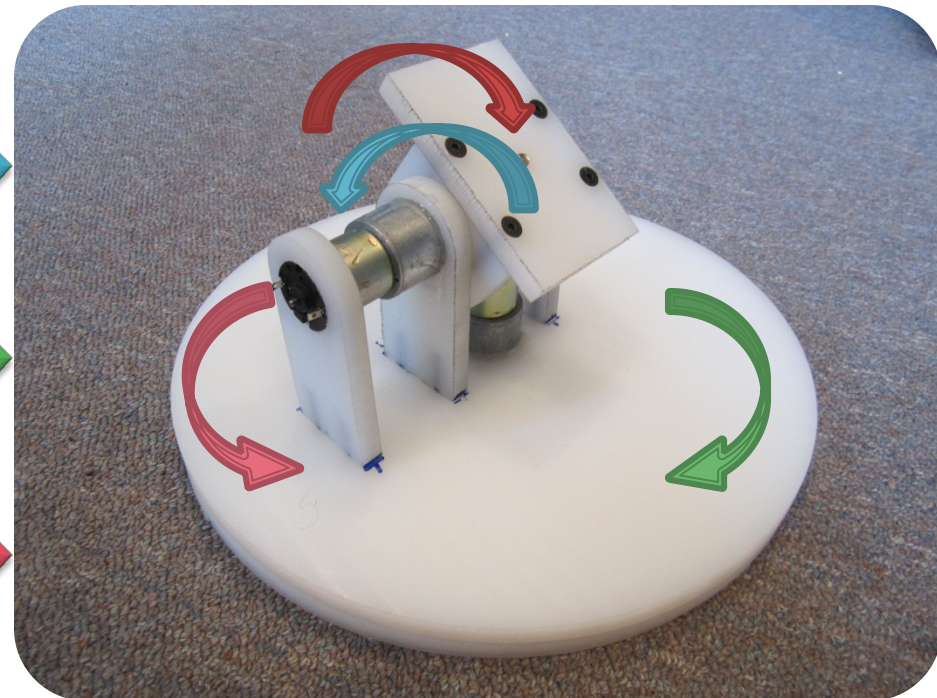
Pan Right

Button  
Press/Hold

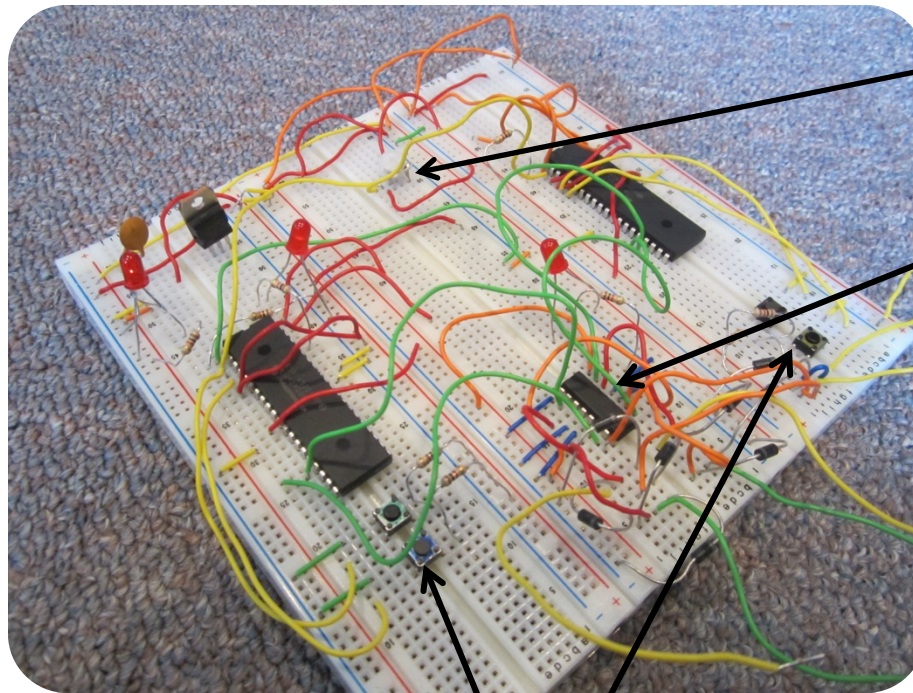
Pan Left

Potentiometer  
Setting

Pan/Tilt  
Speed



# How the PIC was used

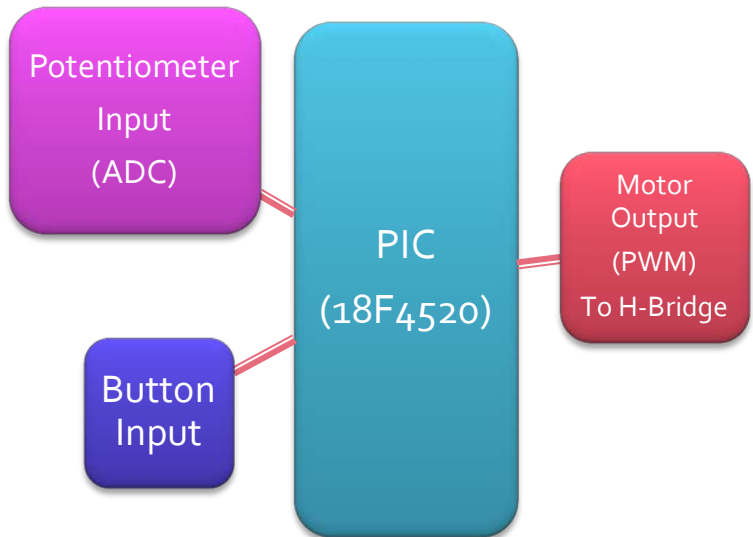


Potentiometer

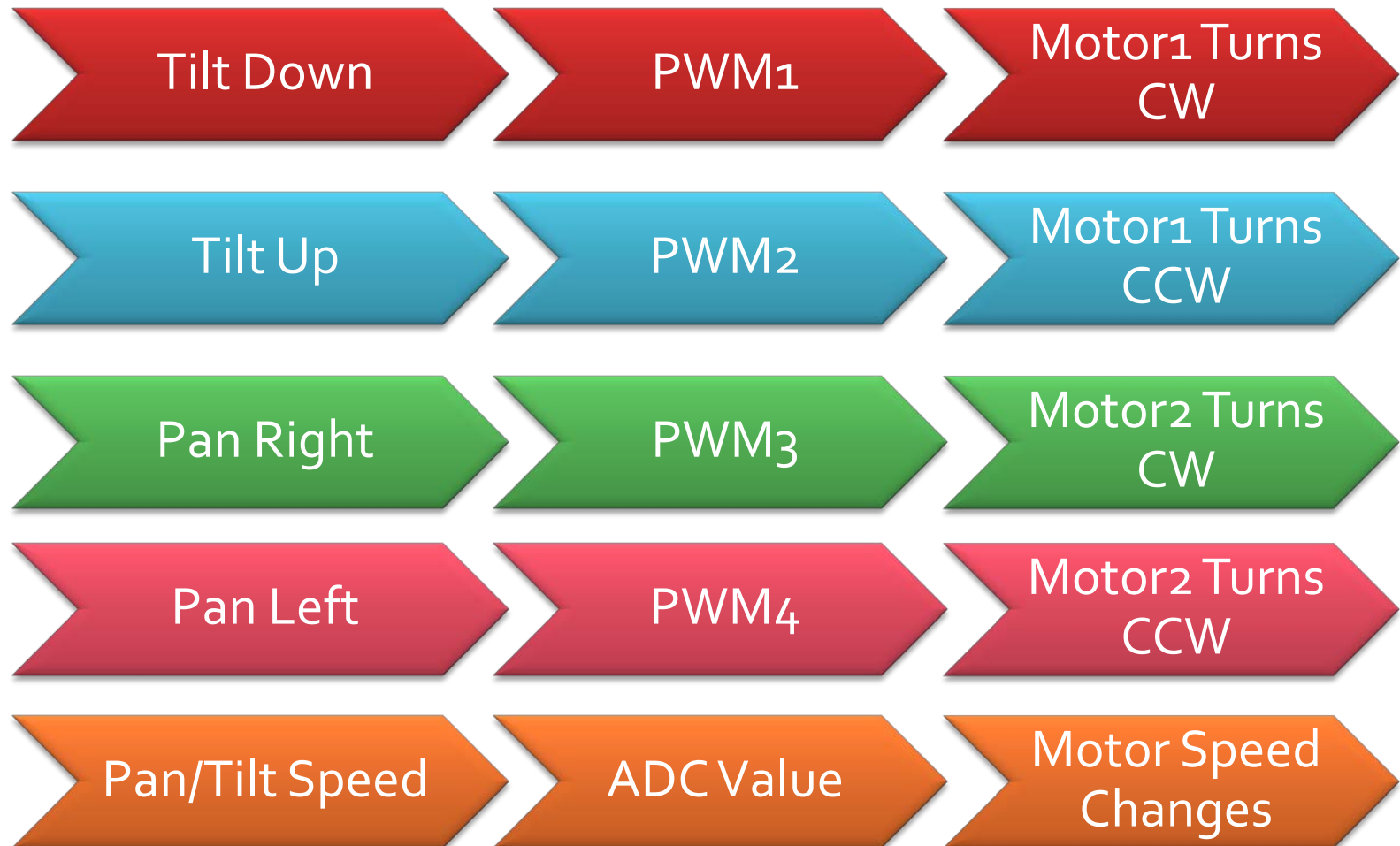
H-bridge that received PWM

Two PIC's were used  
with identical programming

Button Inputs



# Outputs to accomplish objective





# The results of our work

- Our camera mount worked well and did everything that we proposed
  - Bi-directional Pan/Tilt with speed control was accomplished



# Future Generation Advice

---

- Don't pick a project beyond your means and time.
- Think Simple
- A good project does not need to have four or more PICs
- Start the project early and work on it in manageable chunks

# Falling Plate Rack

Elaine MacAslan

Jordan Oja

ME430

# Objectives

- Modify a falling plate rack to automatically reset fallen plates



# Inputs

- 7 contact switches composed of copper tape and grounding the rack

Switch open  
(plate down)



Switch closed  
(plate up)

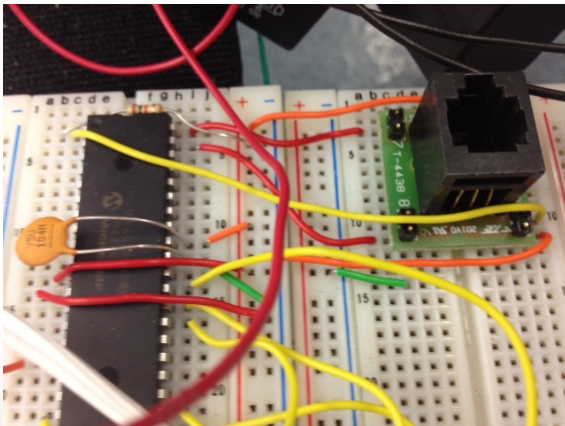


Electrical connection  
to the plate: wire and  
conductive adhesive,  
plus JB Weld to  
anchor the wire

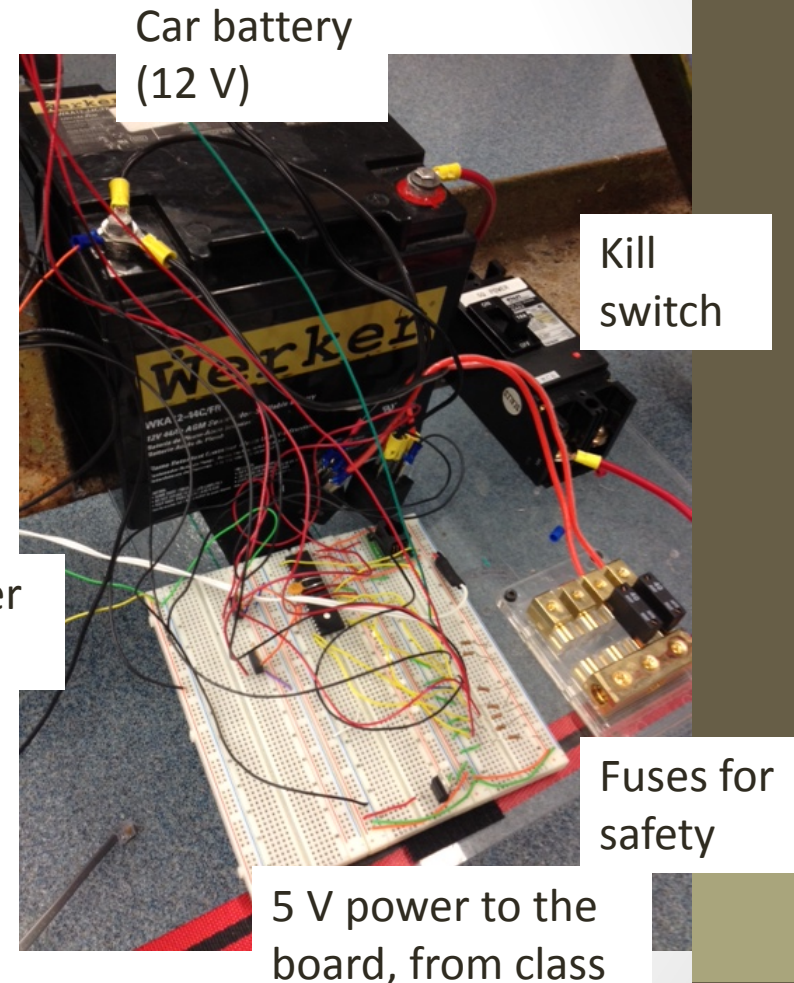


# Inputs

- Power
  - 5 V: PIC, switches
  - 12 V car battery with kill switch: motor
  - Separate 12 V: Turn on and off relay
- Phone jack to program PIC

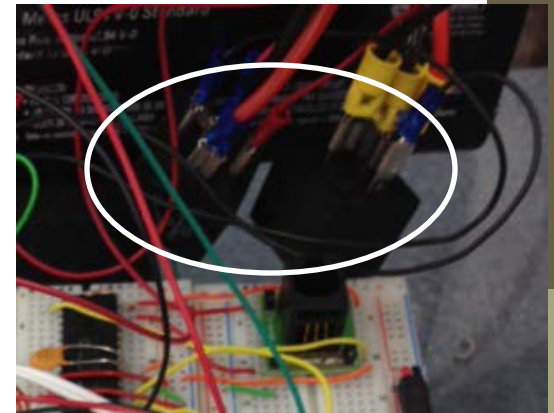
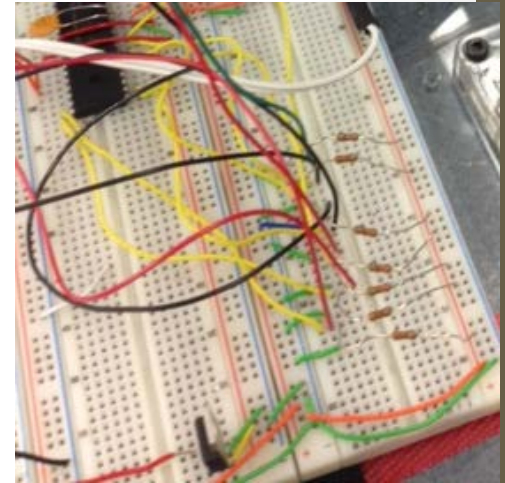


12 V power  
for relay



# PIC as the Controller

- PIC had 7 input pins (resistors, red & black wires, and yellow wires)
  - High when plate was up
  - Low when plate was down
- PIC had 2 outputs (relays circled)
  - To one relay, to make the motor run clockwise
  - To the other relay, to make the motor run counterclockwise
  - Use of delays to allow the motor to finish its motions
- Puck connections to program the PIC



# Outputs

- Two 12 V relays
  - One ran the motor clockwise
  - One ran the motor counterclockwise
- Truck windshield wiper motor
- The motor rotated a shaft with welded brackets that lifted each plate





# Results

- The electronics worked
- The mechanical portion worked, but lifted only three plates when powered with a car battery
- The connection between the motor shaft and pipe was not robust, so the plates did not finish lifting

# Advice

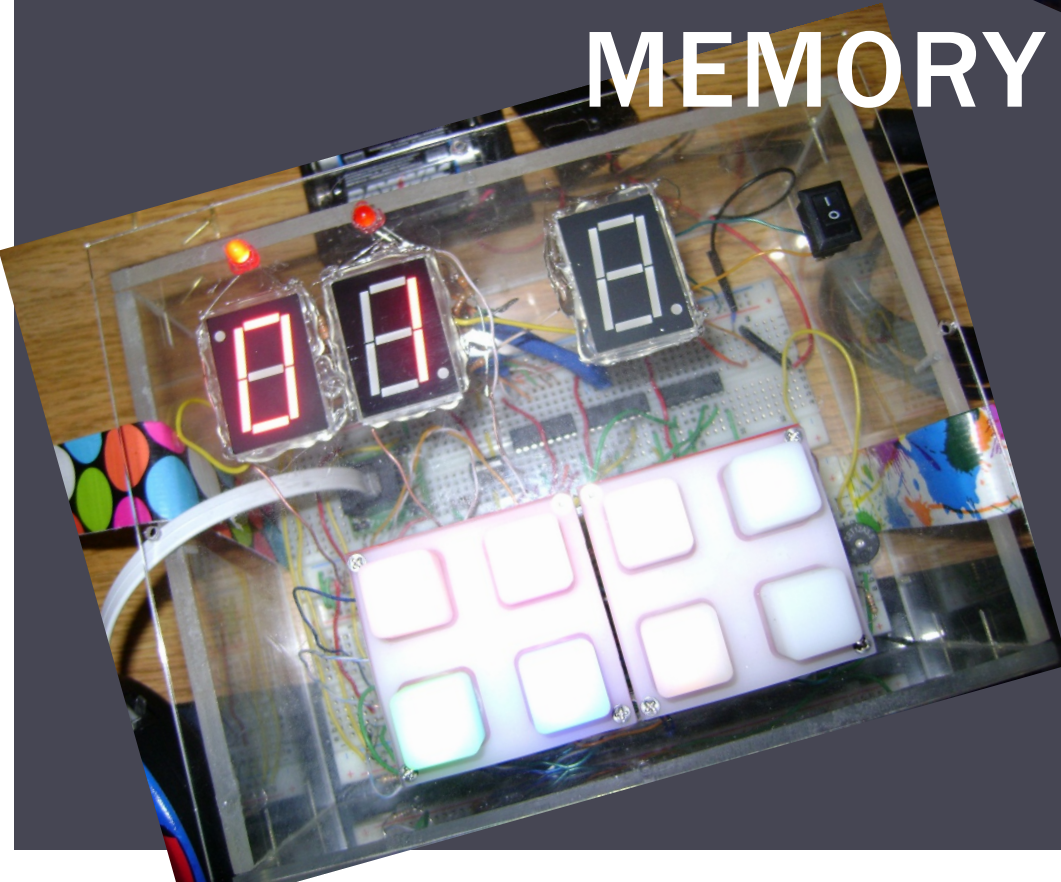
- Start machining early, as the machine shop hours are not convenient
- Acquire spec sheets—ensure your components are robust enough for your application
- Leave time to redesign

# MEMORY GAME

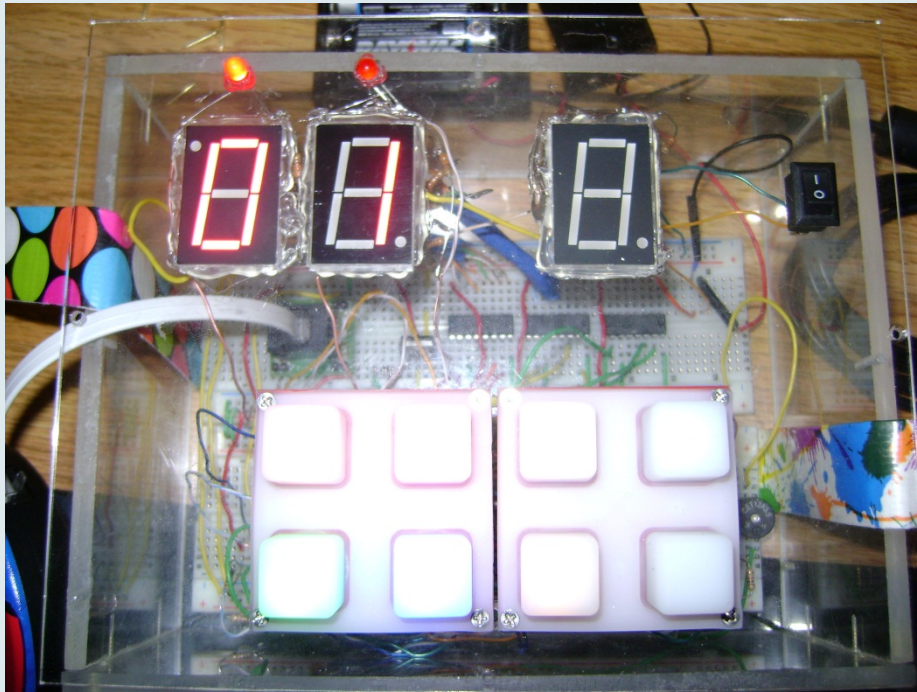
Margaret Modic  
Kayla Pence

ME 430

Winter 2012



# OBJECTIVE



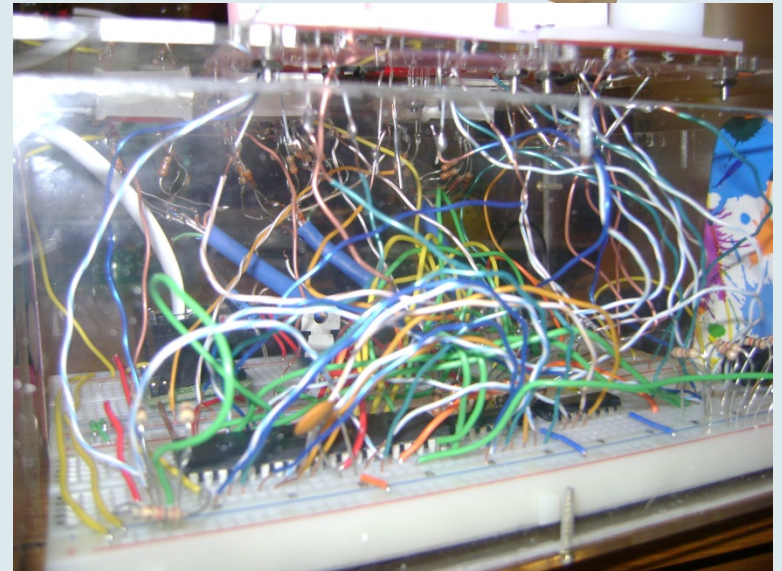
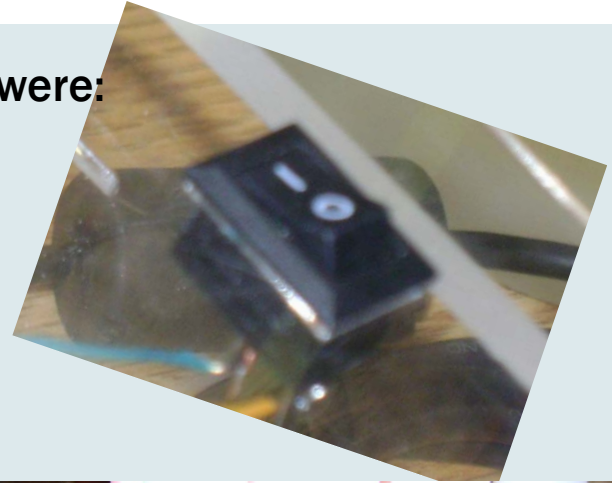
The goal of our project was to:

- accept inputs from the user to decide difficulty and repeat pattern
- display two different random patterns for two players
- output player turn and score

# INPUTS

The inputs of our project were:

- power button
- color buttons

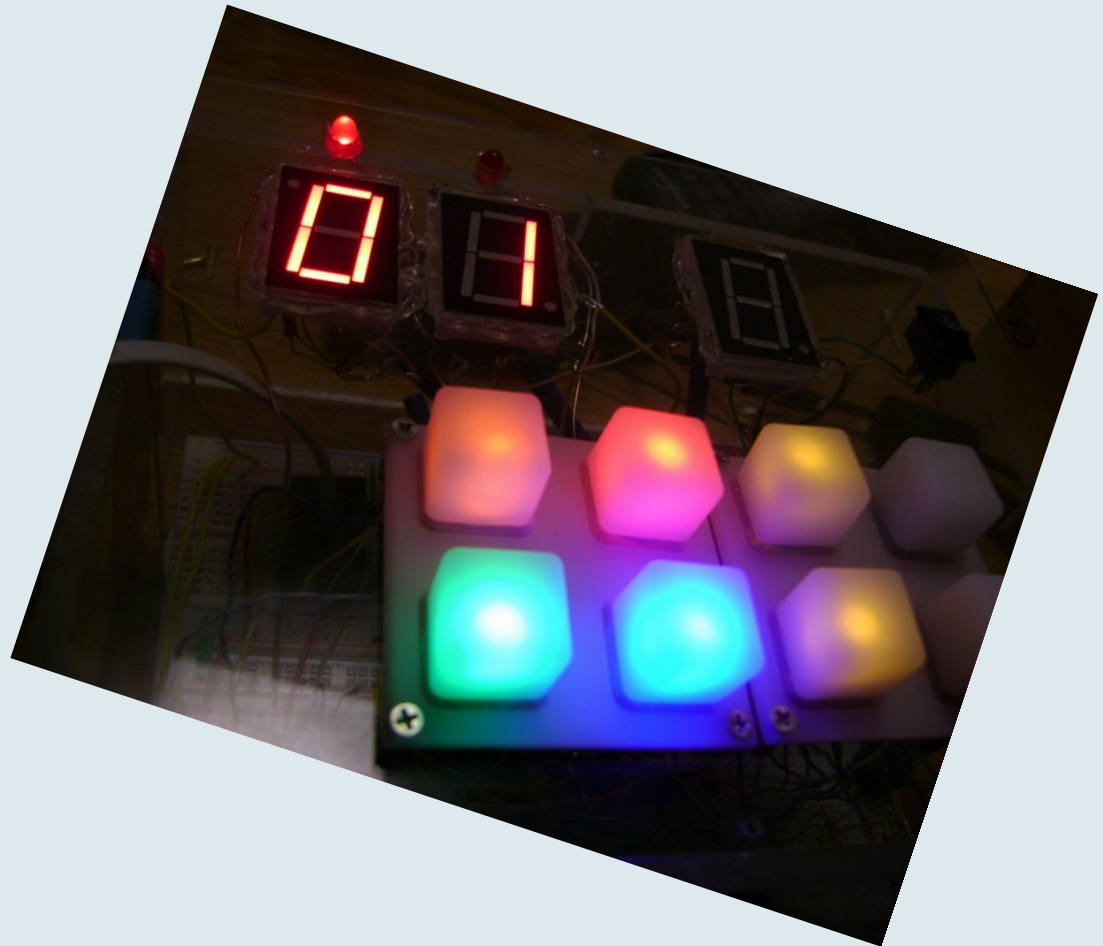




# PIC CONTROL

For this project our PIC:

- accepts inputs from buttons
- uses timer to seed random generation of patterns
- outputs LED's for patterns, player turn, and seven-segment displays



# OUTPUTS

The outputs of our project were:

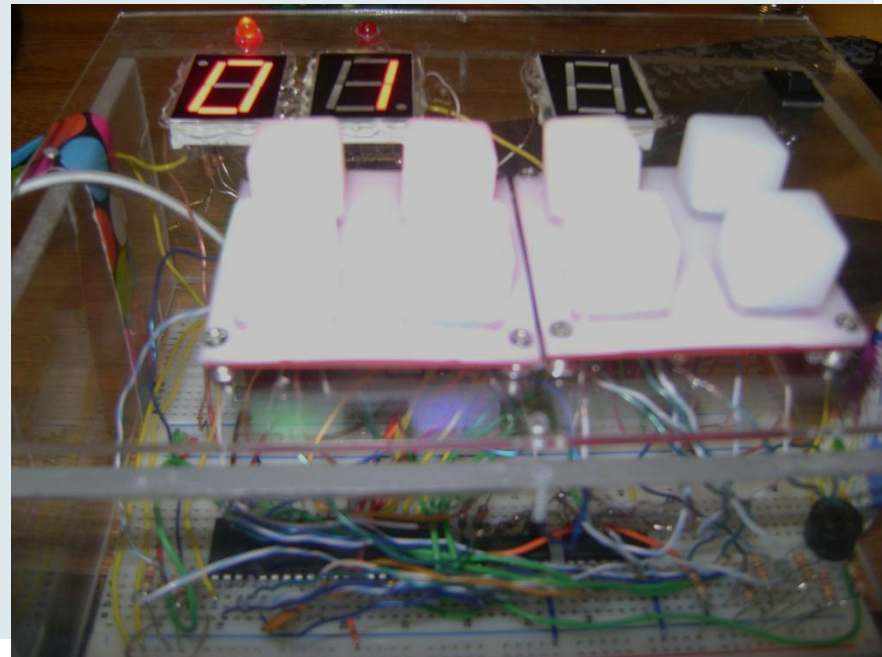
- button LED's
- seven-segment displays
- player turn LED's



# RESULTS

Our project worked like we said it would. It:

- took inputs from the buttons to decide a difficulty
- lit LED to display player turn
- displayed two random patterns for players, that were added to when gotten correctly
- counted correct number of patterns with seven-segment displays
- ends after one or both players get through 9 turns





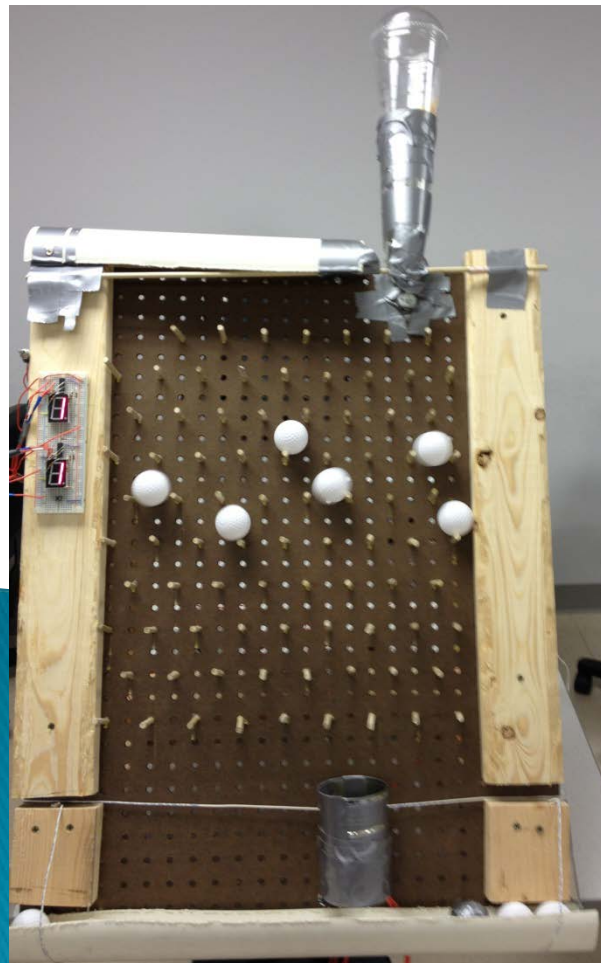
# ADVICE



Advice for the future is to:

- make sure all components work before putting it together
- try little parts of code to make sure they are all working before putting it together

# Mechatronics Project Plinko Board



Donnie Quamme  
Bharat Yalamanchili

# Objectives

- ▶ Create a simple modified version of Plinko.
- ▶ Two modes
  - Use a motor to dispense all the balls at once and catch as many as possible.
  - Pick up balls from the bottom and put them at the top of the board manually. Catch a ball before the buzzer goes off.
- ▶ Have FUN!!!

# Inputs

- ▶ Switch (momentary) to start the game
  - It was a basic switch that was controlled using the PIC. It just turned the game on and activated the code that looked for a ball.
- ▶ Switch (position) to choose game mode
  - Since we decided to have two game modes, we needed a way to select the game mode. The easiest way was to have a two way switch.
- ▶ IR sensors
  - We used a tape sensor although a break beam was easier to set up. The break beam didn't have the range we required. So we used a tape sensor and they worked perfectly by reflecting the light off of the white plastic golf balls.

# PIC controlled

## ▶ DC motor

- We were having some trouble running the motor off the analog signal. Hence we decided to use one of the other pins (digital) and used a delay to turn the motor on and off.

## ▶ Buzzer

- The Buzzer was run using a Timer 0.

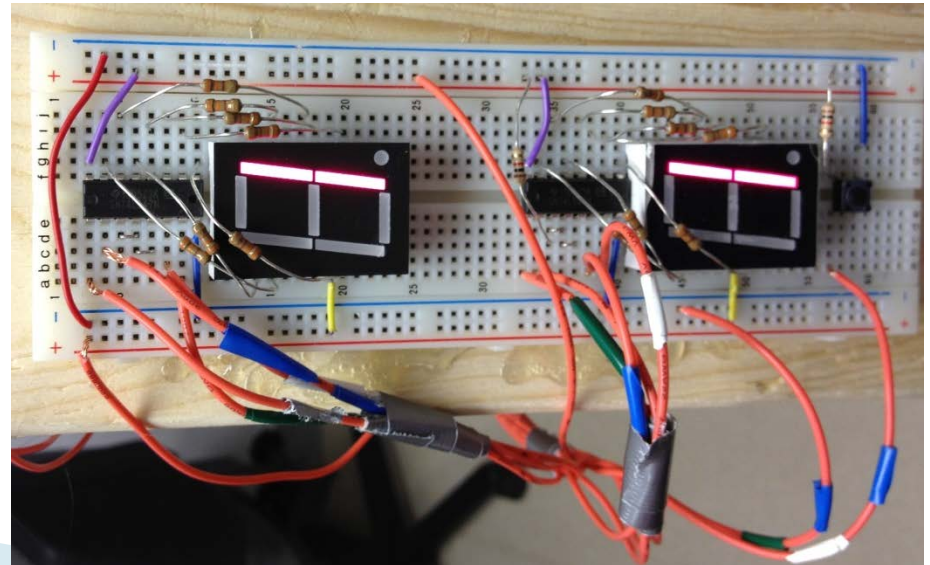
# Outputs

- ▶ Two 7 segment displays
  - We chose standard 330 Ohm resistors to drive the LEDs. We used two 7 segment driver chips one for each 7 segment.
- ▶ DC motor
  - We used a unidirectional DC motor and run it using a MOSFET and used delays to turn it on and off.
  - We added snubber diodes to avoid inductive kick.
- ▶ Piezoelectric Buzzer
  - We used a Piezoelectric Buzzer to indicate that in one of the game modes, the players turn was over as he/she failed to catch a ball within 3 seconds.

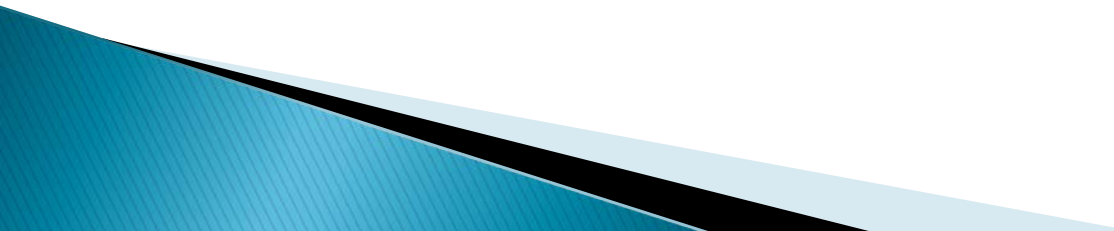


# Results

- ▶ It worked perfectly well.
- ▶ The game mode chose correctly.
- ▶ The motor turned on fine and the score updated as per required.
- ▶ The buzzer went off when the game was over.



# Advice

- ▶ Always figure out ways to attach your pieces together. During crunch time, all one can think of is Duct tape and hot glue. These work fine but it won't make the product look good.
  - ▶ Code and wire simultaneously as opposed to doing one after the other. Debugging becomes a huge problem if you do one or the other first.
- 

# ME 430 Project: Automatic Dog Feeder



Jordan Ralph  
Daniel Wiegand

# Objective

- ▶ Create an automatic dog feeder that will:
  - Allow user to control 2 feeding times
  - Allow user to select the quantity to feed their dog
  - Sense if dog is under bowl as to not drop food on them

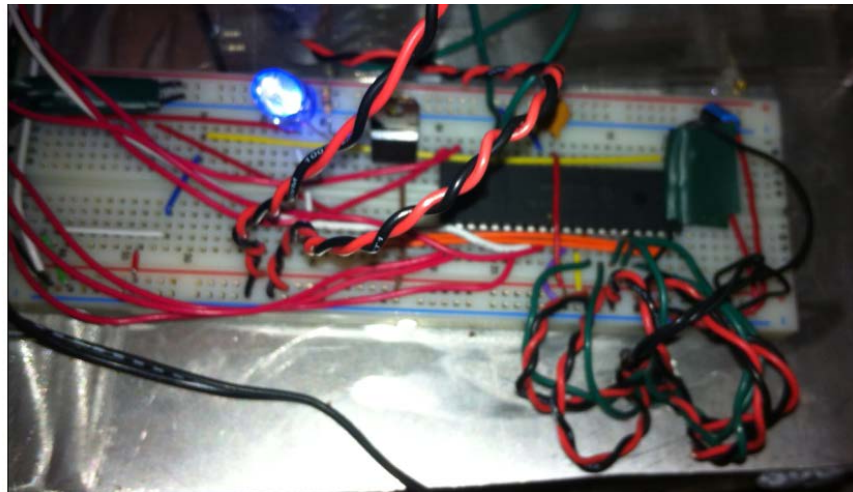
# Inputs

- ▶ 5 Buttons:
  - Two to set current time (hours and minutes)
  - One to set the morning feed time (30 minute increments)
  - One to set the night feed time (30 minute increments)
  - One to set the number of cups to feed
- ▶ PIR sensor to detect if dog is in front of dispenser



# PIC Control

- ▶ PWM – used to control servo motors
- ▶ Interrupts – button and time control
- ▶ Timers – used to keep current time






# Outputs

- ▶ Servo motor control
  - Dispense the dog food
- ▶ LCD display
  - Display current time, feed times, and number of cups to be fed



# Results

- ▶ Everything worked as planned
  - ▶ User was able to easily control feeding variables which were displayed on the LCD
  - ▶ Servo motors dropped a consistent amount of food into bowl each cycle
- 

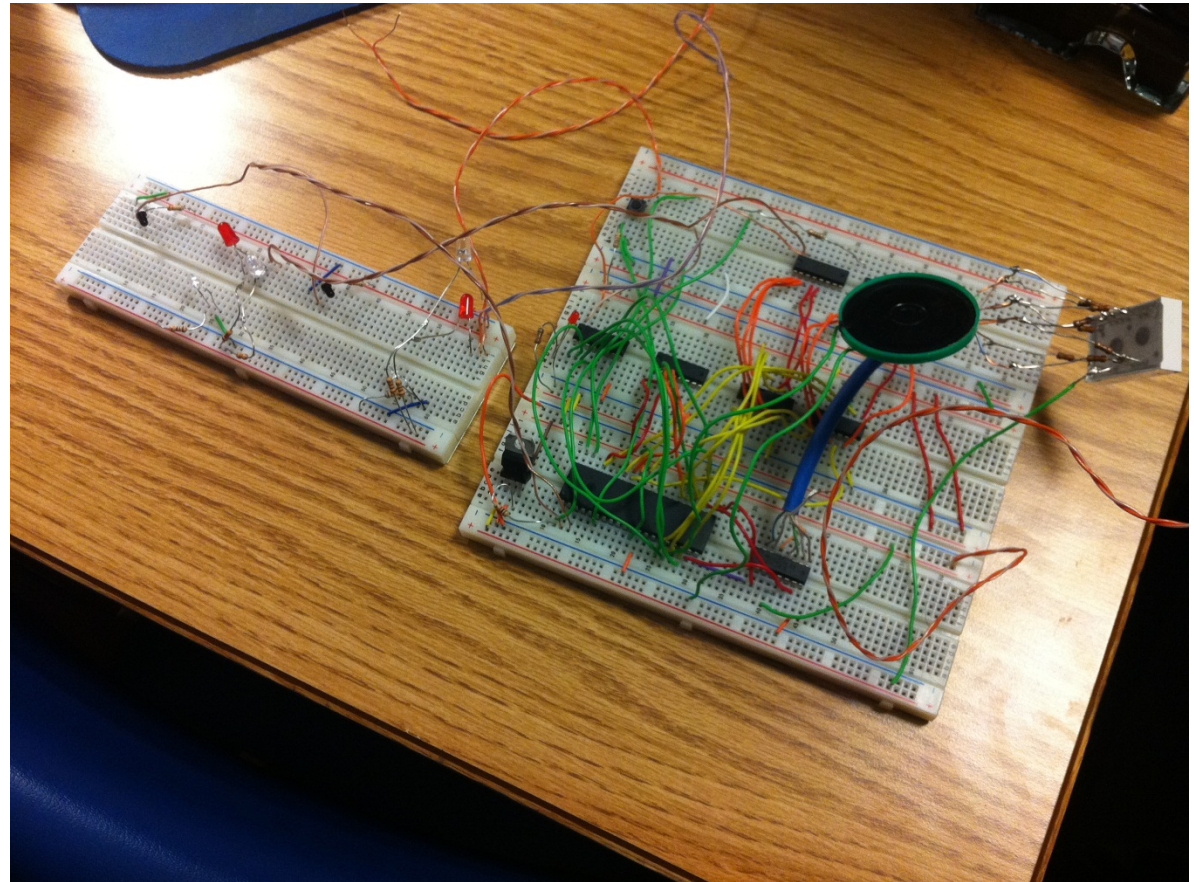
# Advice

- ▶ Don't procrastinate, start early to ensure project success
- ▶ Utilize Rose-Hulman services
  - Laser cut parts
- ▶ Have a plan of attack, divide the work and everything will come together much easier

# Beer Pong Module

Alexander Ray  
Aleksander Yoder

ME430 Mechatronics



# We sought to create a module which allowed the user to play a version of a popular drinking game

- Ten cups are placed in a triangle shape
- A ping pong ball is thrown into the cups in order to “score”
- A team wins when all cups are made

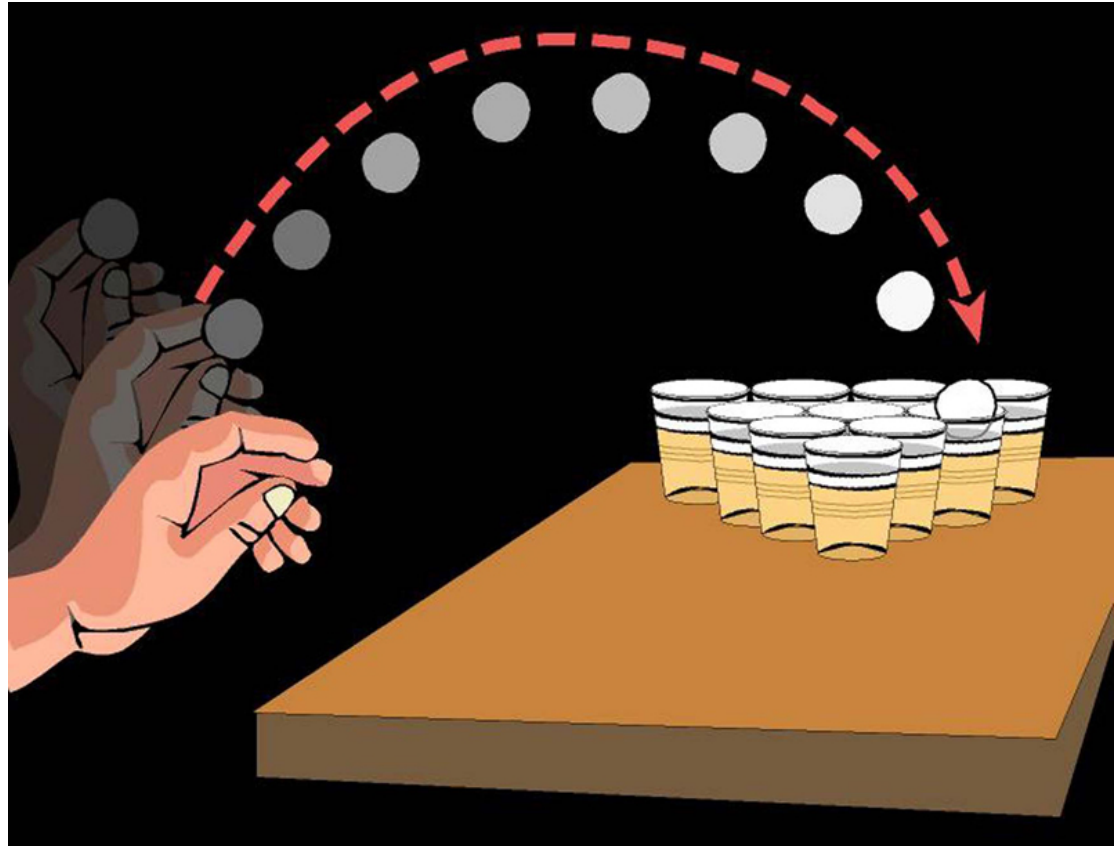
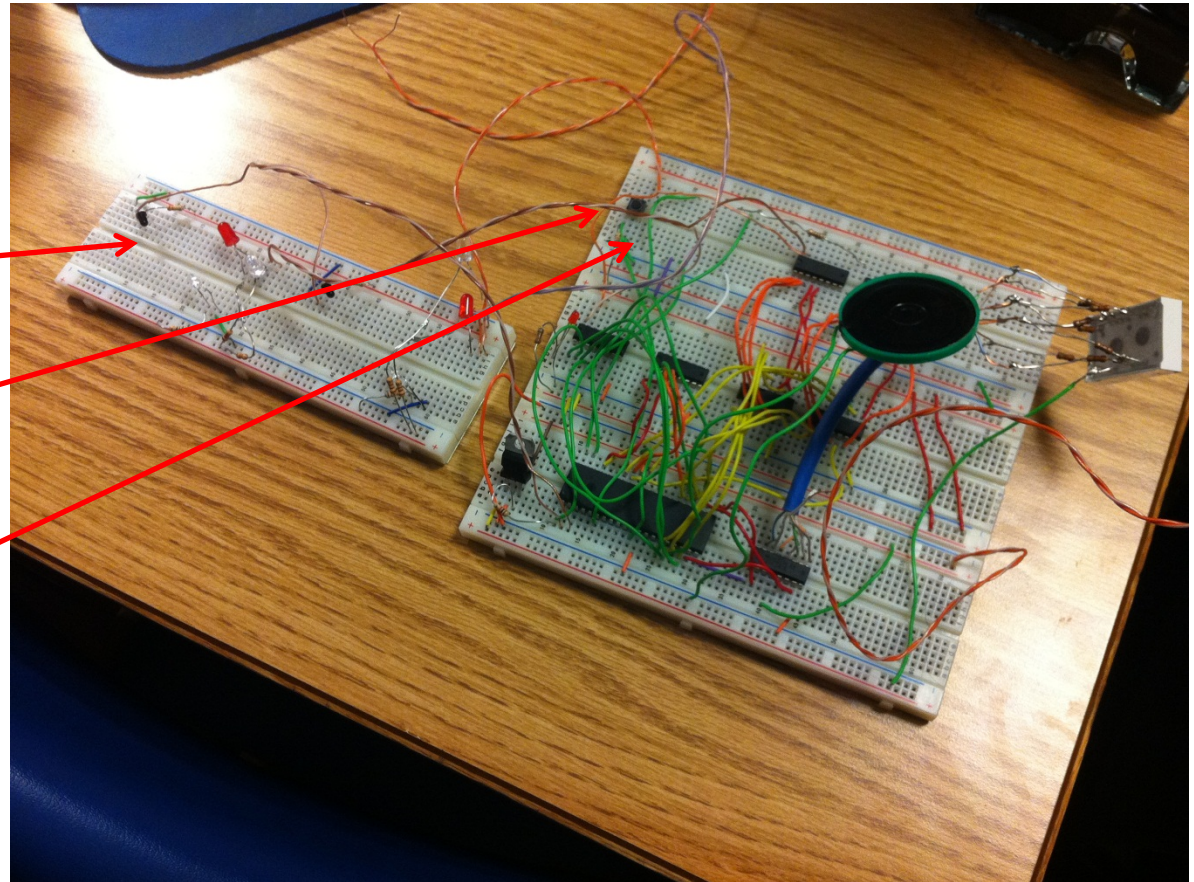


Figure 1: Beer Pong setup



# Expected Inputs for the module

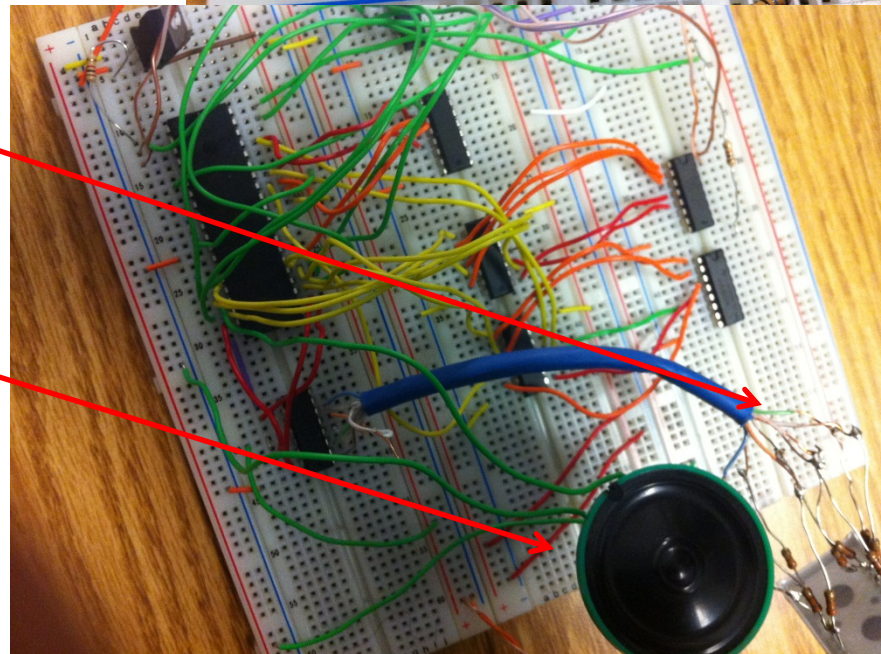
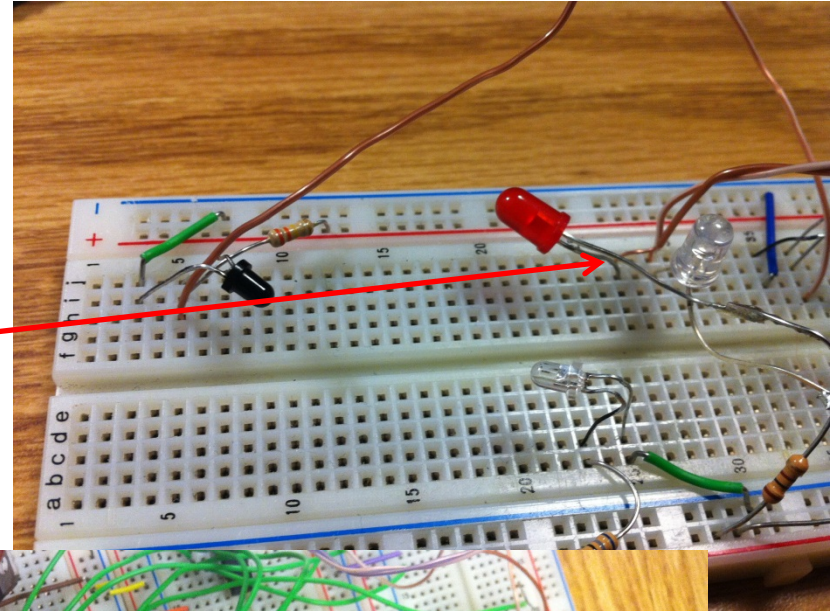
- Each cup requires a sensor to show when a cup is made
- A game start button
- The start button also functions as a reset button





# Expected outputs

- Two LEDs for each cup to show in play or out of play
- One Seven-Segment Display to show number of cups made
- Piezo buzzer to indicate end of game



# Using the PIC as a controller

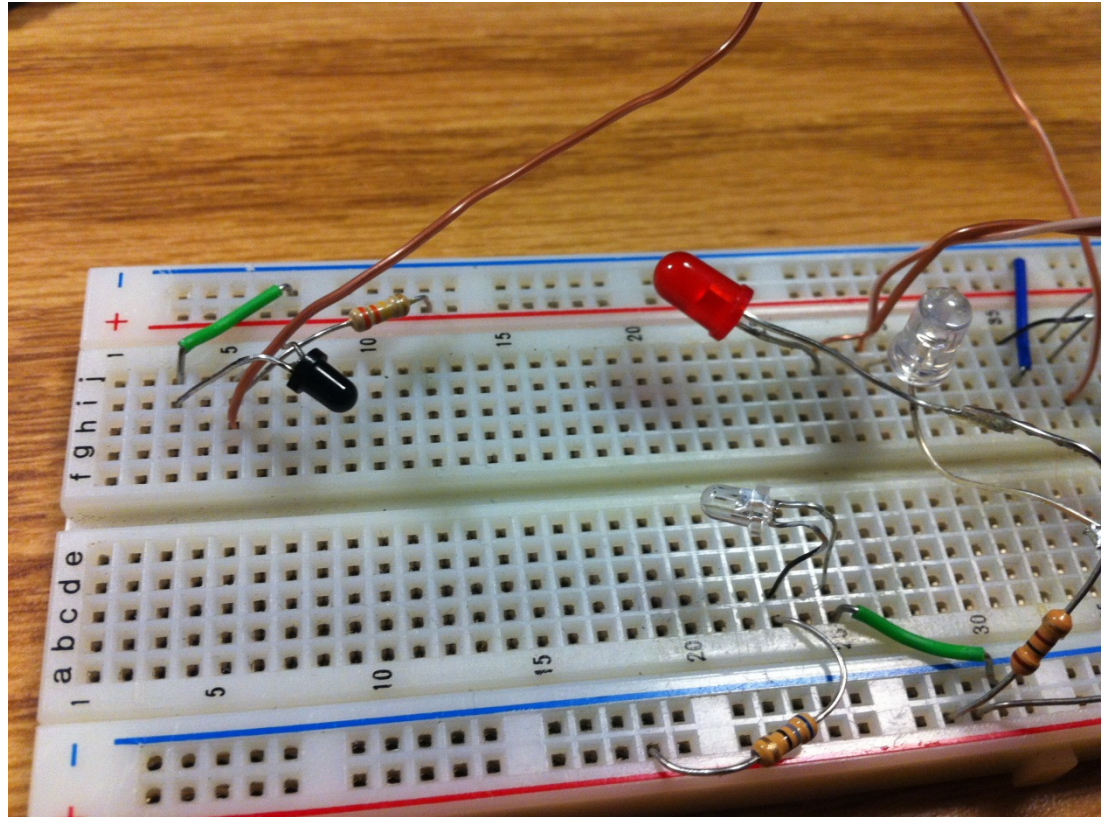
- Senses the analog value from the IR sensor
- If the break-beam is triggered the LED color changes
- PWM is used to control the Piezo buzzer
- Delays are used to control the time between reading LED color change

# Results

- When the full wiring setup was complete the sensors would no longer read
- The base was not a sufficient size to house the wiring
- The system was simplified to two cups which worked as specified
- The Seven-Segment display and Piezo buzzer worked as described

# Advice to future classes

- Using break-beam sensors is not the way to go
- Aligning the emitter and detectors were difficult





# AUTO DEFENSE TURRET



Nolan Roberts  
Zach Runyon  
Zhen Wei

# Project Objective

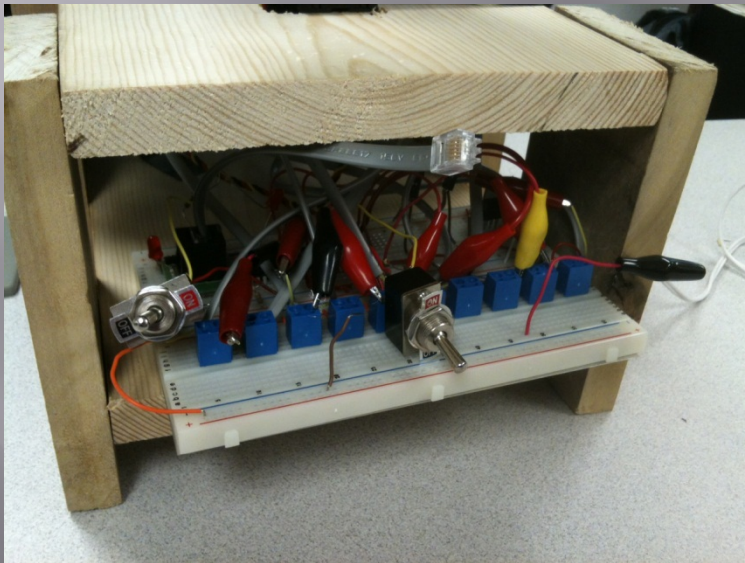
- ▣ Design and build a fully functioning Auto Defense Turret
  - Track target for 180 degrees
  - Aim an airsoft gun at a detected target
  - Pull trigger after sensing target; fire until the target is no longer in the field of view



# Inputs

▣ On/Off switches

▣ PIR Motion detectors

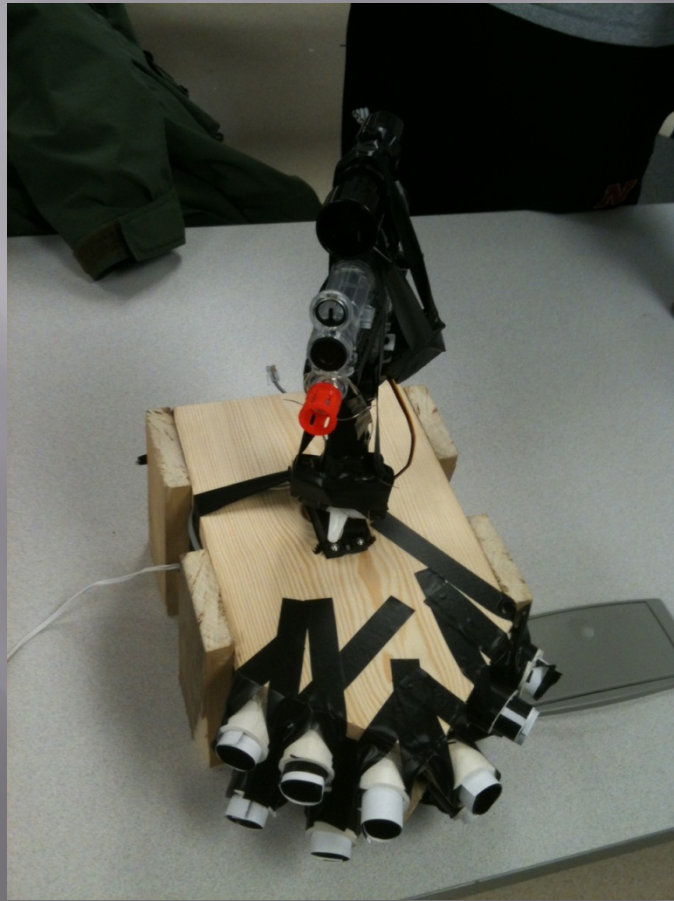


# PIC Control

- ▣ On/Off switch controls the activation of interrupts and PWM
  - Two outer PIR sensors each view half of the field of view and control a program interrupt
    - ▣ When triggered, the interrupt function uses if statements to determine where the target is
- ▣ The “safety” switch
  - Every time the target moves, the position of this switch is checked
    - ▣ If on, a servo is used to pull the trigger

# Outputs

- ▣ Proper Target Region  
(Gun points to target)



- ▣ Trigger is pulled





# Results

## WORKED

- ▣ The gun tracked target with mostly consistent results in half of field\*
- ▣ The trigger was pulled

## BROKEN

- ▣ Mostly ineffective in the other half of the field\*
- ▣ Force to pull trigger broke servo mountings

\*Result of poor sensor performance. Some did not work at all. The ones that did had an inconsistent interpretation of motion.

# Advice for the future

- ▣ Order early and test equipment upon arrival
  - Cheap parts have a tendency not to work. It takes time to get replacements
  - Even if you got spares, you might still not have enough that work
  - You'll waste a lot of time de-bugging working code if the problem is really bad hardware
- ▣ Be realistic
  - Take your expectations and divide them by two.
    - ▣ You can always add more
    - ▣ A simple working product is better than a shiny object that doesn't move
- ▣ Divide and conquer
  - Two people staring at code is not productive
  - Soldering takes a takes time



# AK-47 BULLET LOADER

ME430 Mechatronics

Eddie Scott

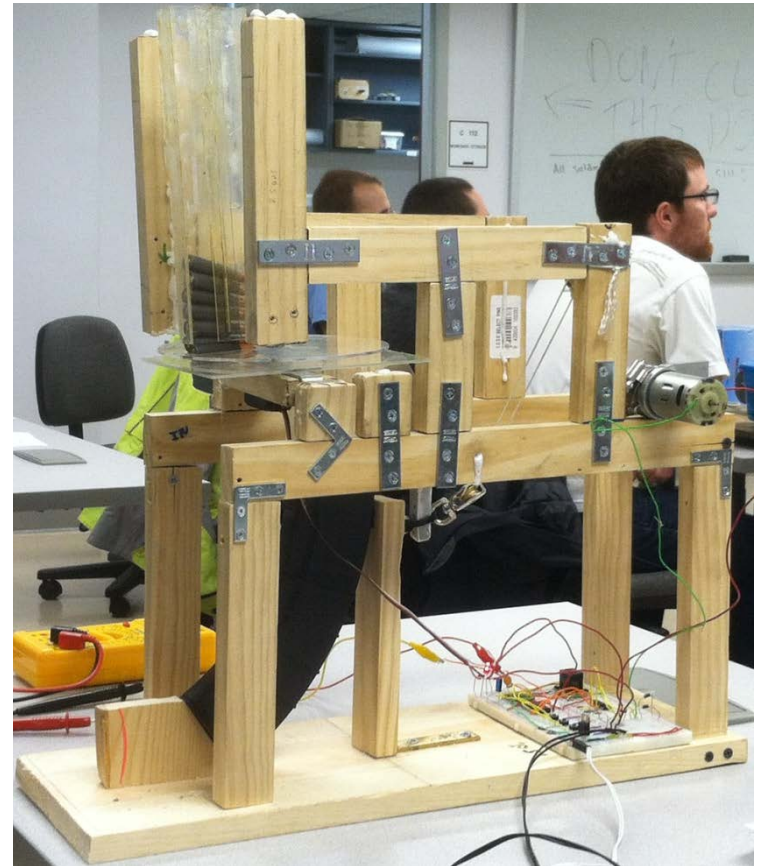
Matthew Hogue

Ryan Parker



# PROJECT OBJECTIVES

- Easily and automatically reload a AK-47 magazine that will be able to accept a digital input of how many rounds you want to load into the magazine.
- It then should put the bullets in one at a time and keep track of how many it has inserted.
- Once it reaches the prescribed limit it will shut off.



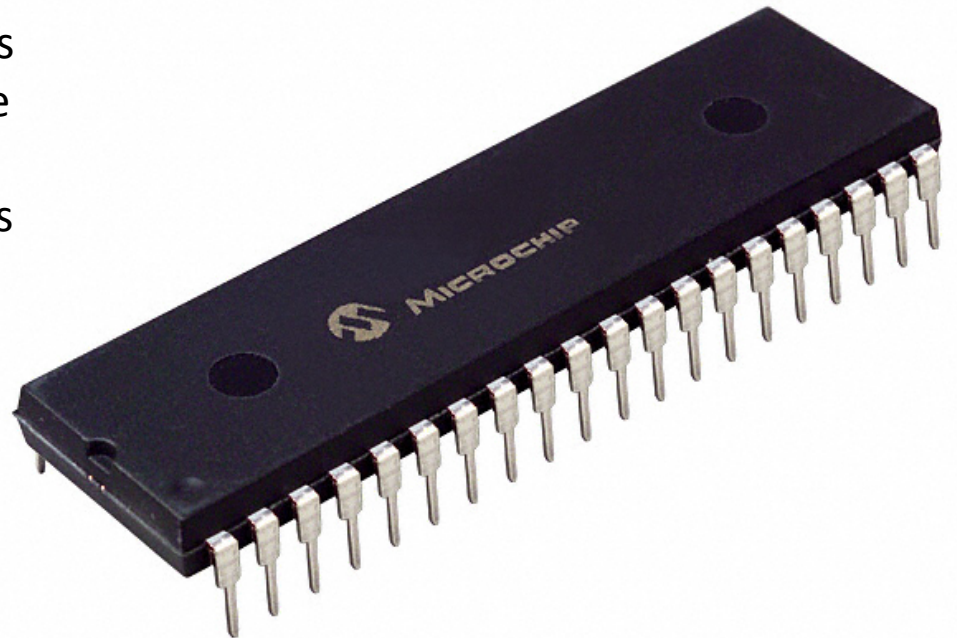
# INPUTS

- Our project inputs consists of three pushbuttons that will tell the PIC to load either 1,5 or 10 bullets into the magazine.



# PIC COMPONENTS

- Interrupts were used to run the servo and the pushbuttons. Using interrupts for the pushbutton it made it possible to continuously add to the current queue of bullets to load in increments of 1,5, or 10.
- Timer interrupt was used to trigger the servo.
- Delays were used to start / stop the servo and DC motor.



# OUTPUTS



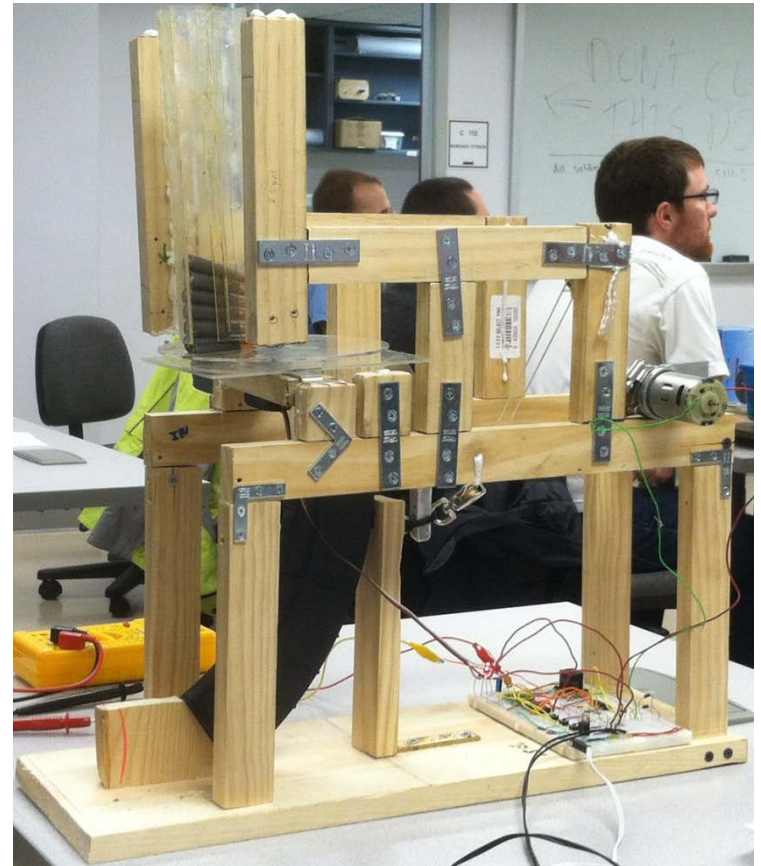
A servo motor was used to control the progression of the bullet from the hopper to the loading chamber.

A DC motor was used to actuate the loading arm.



# RESULTS

- The project worked as planned the night before demo day but that morning the servo motor did not respond and we were forced to manually advance the bullet.
- Even without the servo motor the project worked with no other complications.





# ADVICE FOR FUTURE CLASSES

- Make hardware easily interchangeable in case of future complications.
- Start project early and manage time effectively.
- Order parts early and share shipping orders with other teams to cut down on costs.





**GAME OVER**

# Shoulder Rehabilitation Device

ME 430

Dr. Olson

February 2012

Bobby Short

Ethan Kestenman

# Background

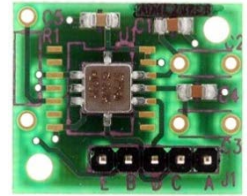
- The idea for this device was suggested to us by a physical trainer

During shoulder rehab exercises, it is common practice to have the patient lift a light bar, either directly in front of them or behind them. The patient is supposed to try to match the flexibility of his/her healthy arm. Patients have a tendency to “cheat” by keeping their injured arm lower, especially when they do this exercise behind their back and they cannot see the bar.

# Project Objective

- Create a prototype rehabilitation device used to increase shoulder flexibility in early stages
  - Indicate level and direction of tilt relative to horizontal
    - LED display
    - Vibration motors

# Project Sensor



- One input: accelerometer (analog)
- Analog output of the accelerometer gets higher or lower depending on which way you tilt the device
- Our device was actually a 2-axis accelerometer with optional digital duty cycle modulated output (ADXL213EB)
- We just used analog output of one axis

# PIC Programming

- One analog input, all digital outputs
- Read ADC
- If/else statements to determine whether or not the input crosses thresholds
- Turn outputs on or off accordingly



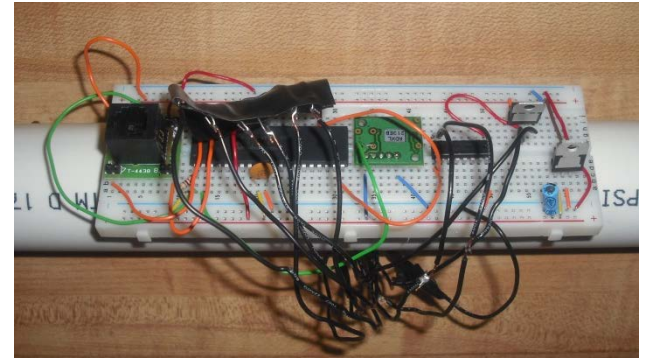
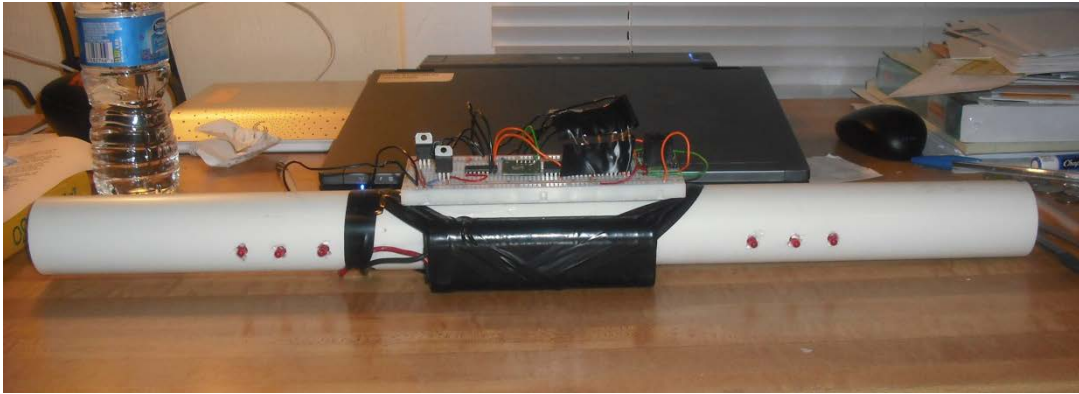
# Outputs

- 3 LEDs and 1 vibrating motor on each side of the device
- Motors vibrate if the device is not held level
- LEDs show level of tilt
  - Slight tilt = 1 LED lit
  - Medium tilt = 2 LEDs lit
  - Large tilt = 3 LEDs lit
- LEDs driven directly from PIC outputs
- Motors run on 3.3V (needed a voltage regulator)

# Results

- Our final product worked exactly as we planned
  - During the building process, we ran into a few problems finding where to read the analog output on the accelerometer chip, but we were able to resolve these issues

# Results



# Advice

- Try to create a project with as few moving parts as possible. The complication involved with creating moving parts proved to be a challenge to many teams in our class.
- Troubleshoot code by using an equivalent model on a breadboard if possible. Wait until everything works before integrating the circuit board with the physical device.
- Generate project ideas that have a specific and practical use.

# Automatic Shooting Gallery

*Created by:*

Andrew Roth

Evan Wilson

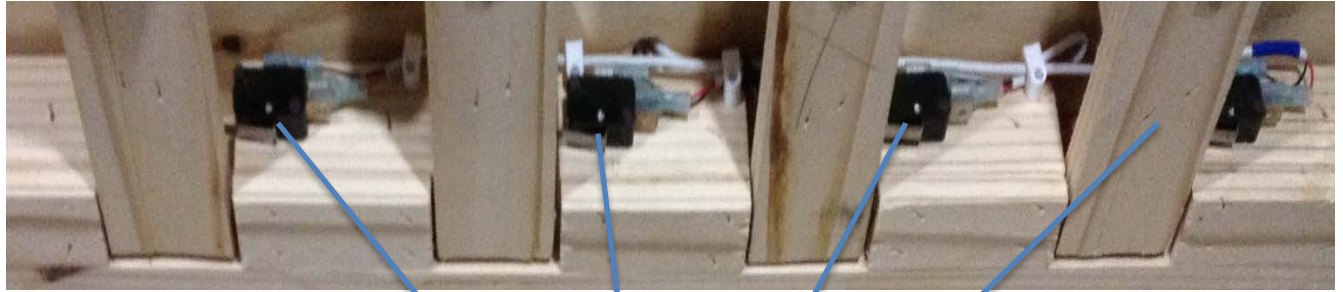


# Objective

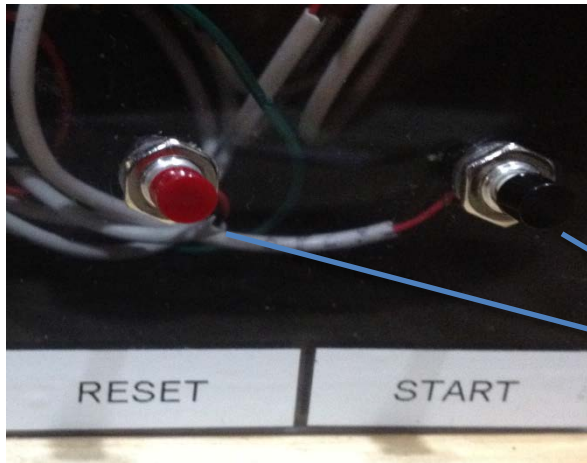
- Create a platform that would:
  - Register targets being knocked down
  - Have a timer that starts and stops with the game
  - Have 2+ game types with varying scoring systems
  - Reset targets to upright position



# Inputs



Target Status Switches



Start/ Stop  
Momentary  
Buttons



Motor  
Status  
Switch



6-state Game-type  
Switch

# PIC Utilization

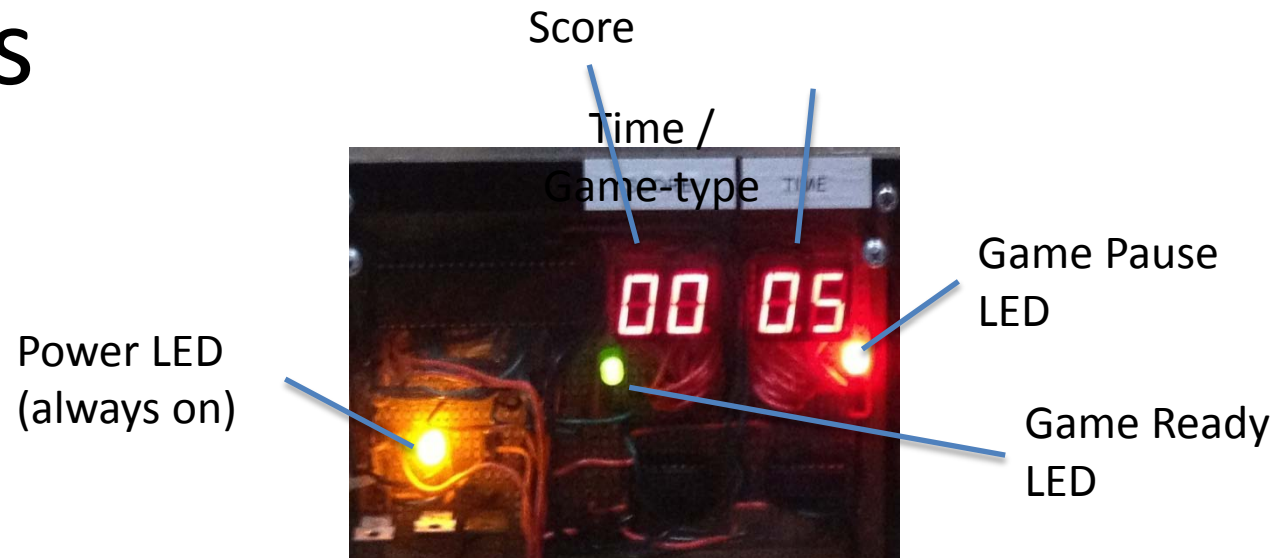
- ADC: Used to determine game-type based on position of the 6-state switch
- Timers
  - Timer0: Used to increment time variable during gameplay
  - Timer1: Used to drive Multiplexing (see next slide)
  - Timer2: Used to drive PWM

# PIC Utilization

- PWM: Used to run DC motor
- Interrupts: Used interrupts on Timer0 and Timer1 to complete their respective tasks
- Multiplexing (outside of curriculum): Used to run 4 7-segment displays using only 2 Darlington chips

\*Our project only used one PIC

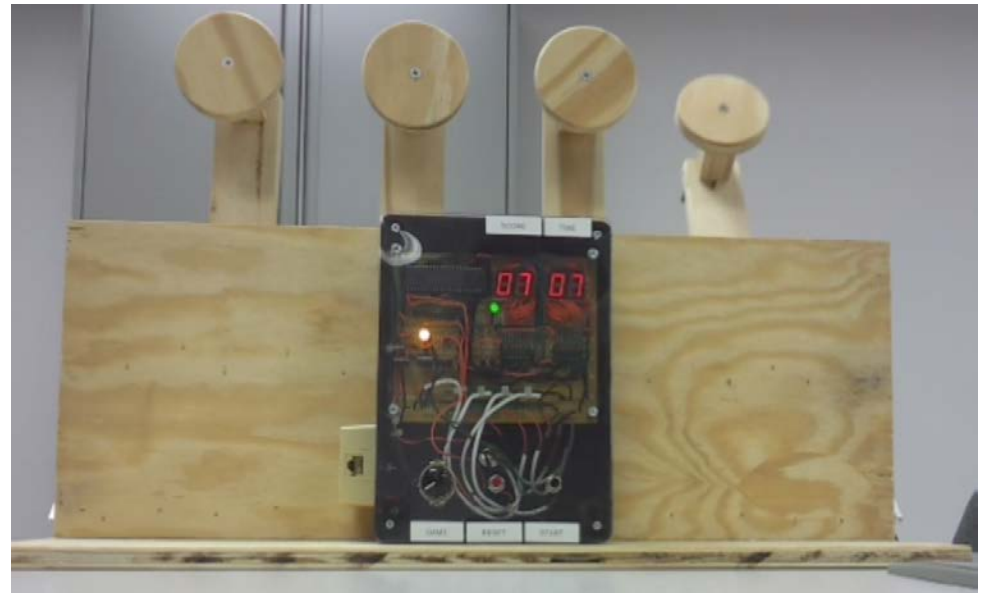
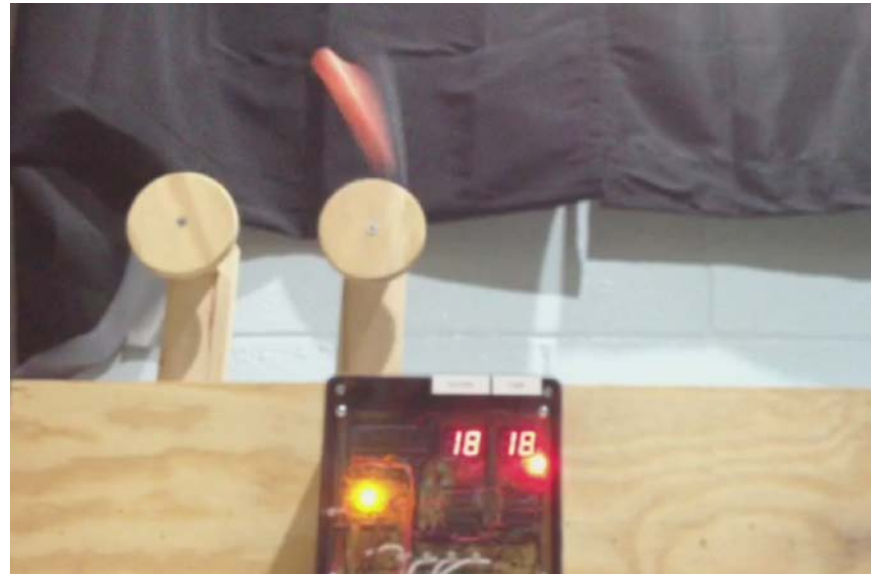
# Outputs



DC Motor connected to Linkage Mechanism

# Results

- It worked as expected
  - Had 6 playable game-types (more than originally planned)
  - Motor reset targets as specified by game-type
  - Printed score and time to displays



# Advice

- Start build hardware as soon as you feel you have learned enough to begin
- Keep it simple
  - a lot of simple systems together can do cool things
- Avoid using methods not discussed in class, or be prepared to ask a EE professor for help
- Leave time to troubleshoot and replace parts
  - Know where you can get replacement parts last minute or order spares