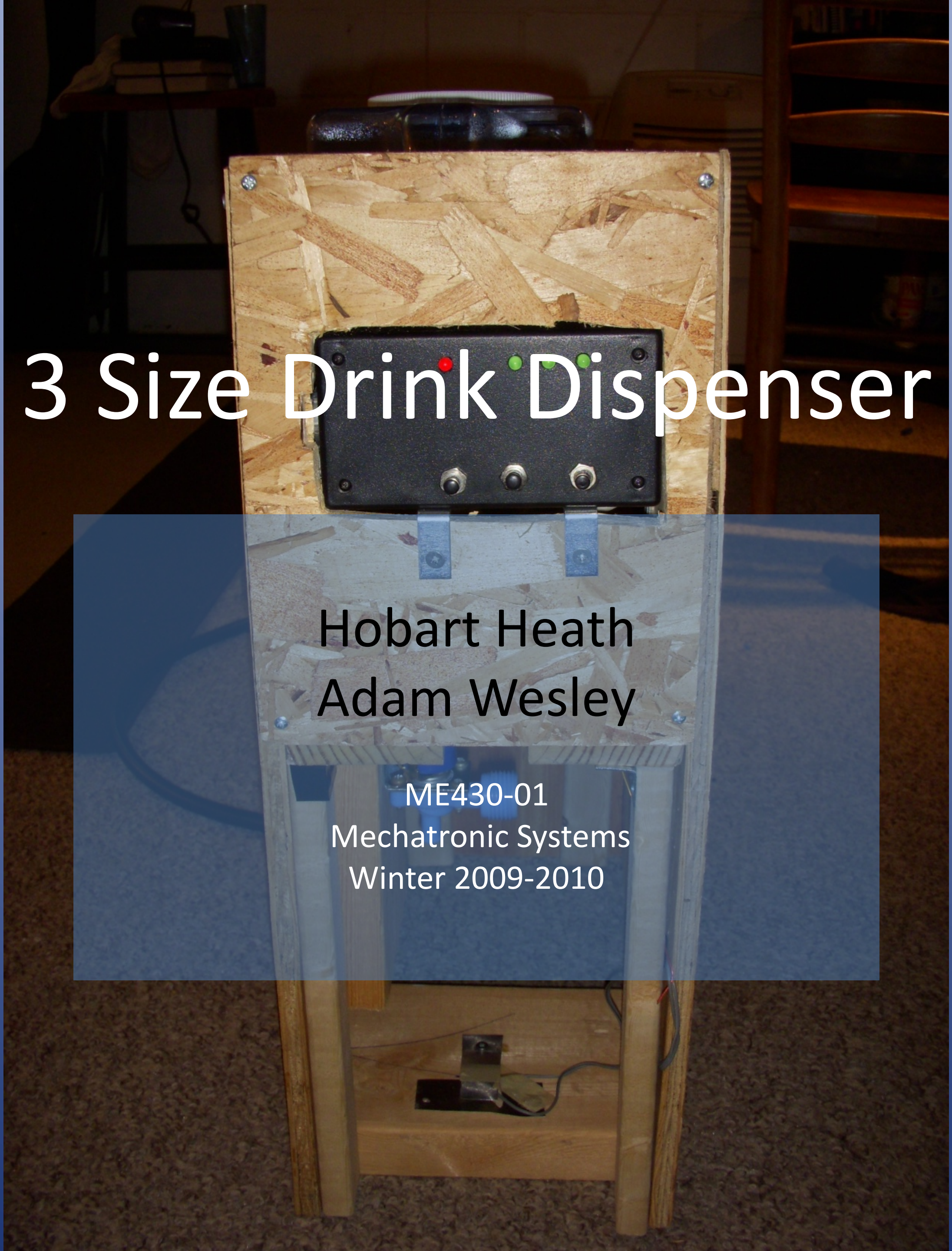


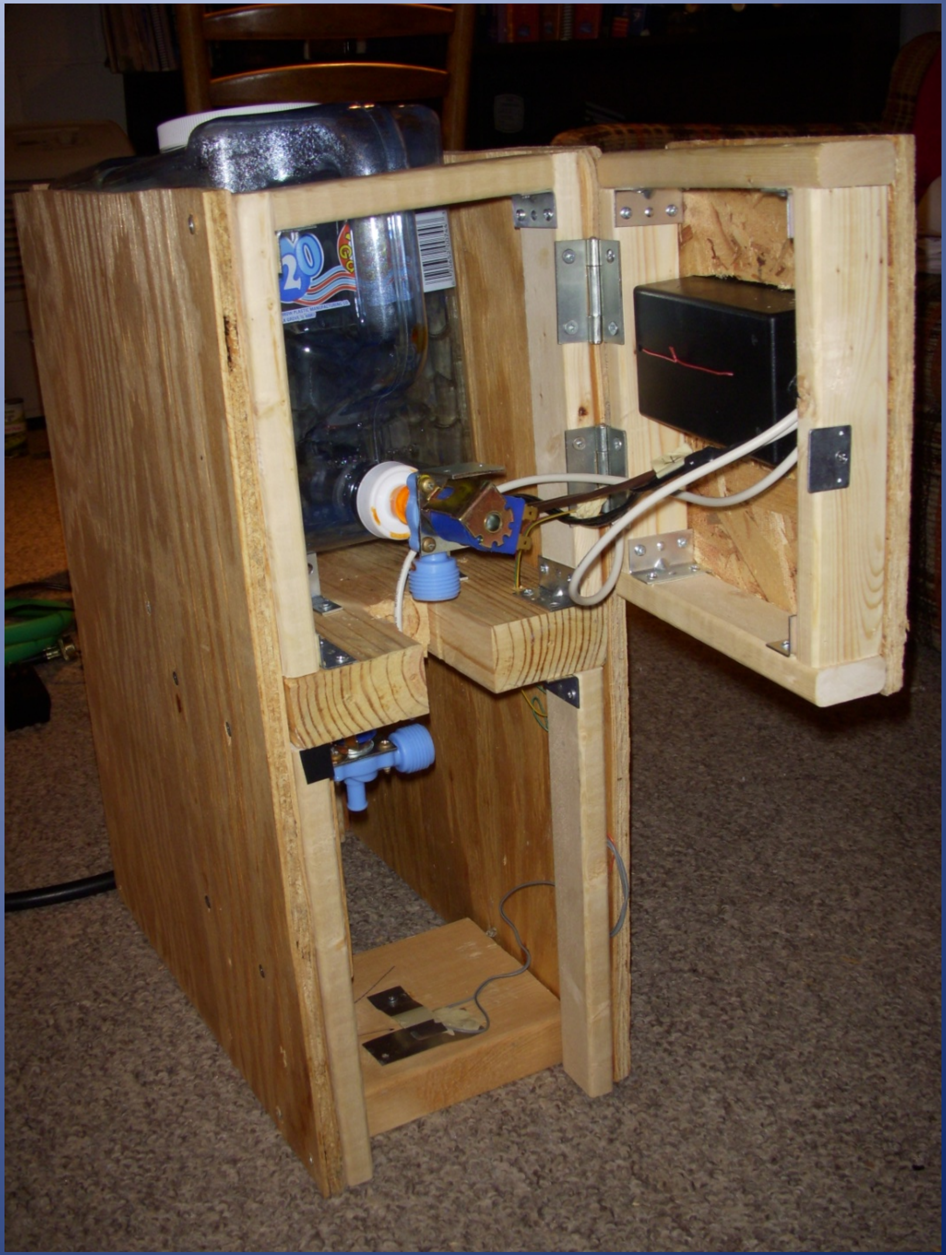
3 Size Drink Dispenser

Hobart Heath
Adam Wesley

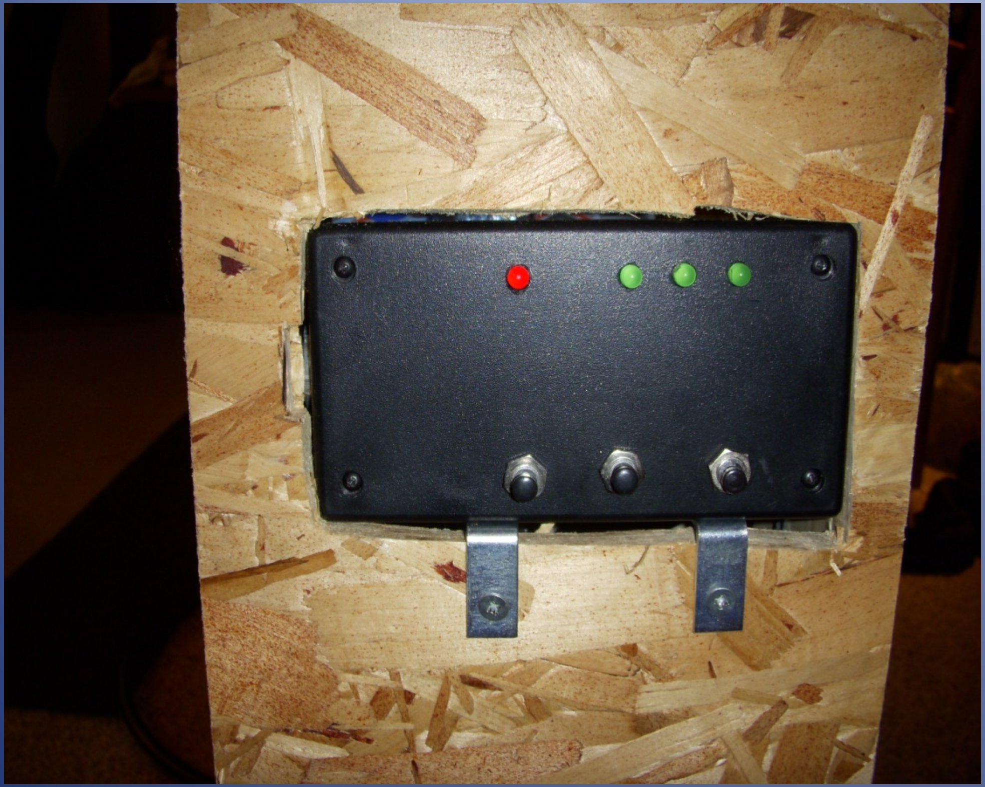
ME430-01
Mechatronic Systems
Winter 2009-2010

Objectives

- Create a drink dispenser that can dispense a small, medium, or large drink for a standard tall drinking glass by use of only touch buttons.



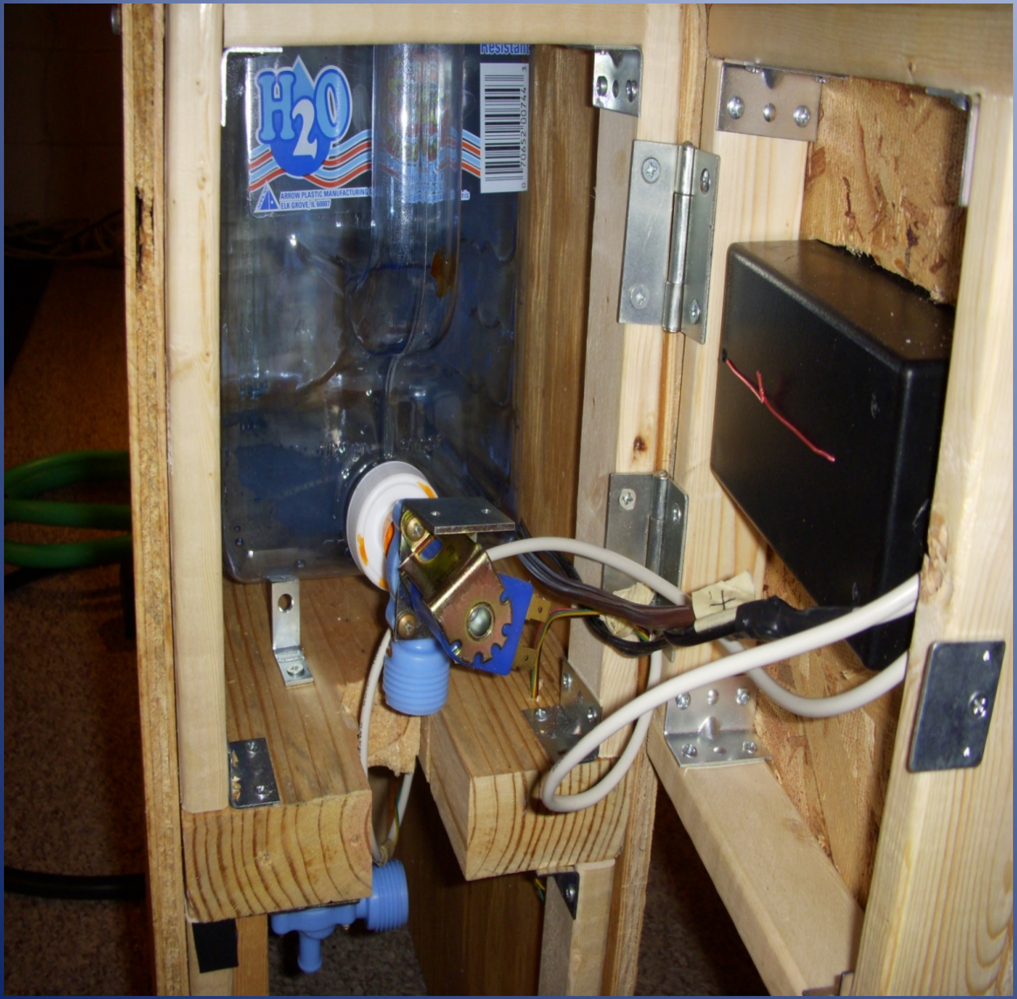
Inputs



- 3 momentary normally open buttons
  - Used to select size
- Pressure Pad Switch to disable the dispenser unless cup is in proper place, and a red indicator light to say the machine is not ready

The user first places their cup in position under the valve. If in the correct place, the pressure pad circuit under the cup will be complete and the red LED indicating a not ready machine will turn off. The user then selects a size from one of the three buttons. (Left to Right: Small, Medium, Large)

Outputs



- Indicator LEDs to let you know what size you have selected
- Valve is triggered and held high on the same timer has indicator LED to let you see what size you selected as it pours

Programming the PIC

- If statement
  - Used in place of interrupts since considerable time was spent trying to get the interrupts to work, but realized a simple 3 case if statement would work just as well.
- Time Delays
  - Used to keep the indicator lights on and the valve open while pouring

Results

- It worked!
- Came out a little quicker than expected, but from a Mechatronics standpoint it worked perfectly.
- Pressure Plate safety switch worked even when wet.

Lessons Learned

- **Build in sections**: Every project is a little more complicated or more involved than you planned. Build in sections to reduce stress
- **Simplicity Rules**: Don't be afraid of too simple. Chances are it won't be.
- **Test**: Build it and test early. Every group fried, blew, or broke something at least once before the class demo.
- **Don't shortcut**: Be sure to cover all wires so they don't ever cross. We lost some time and components because we didn't follow simple wiring practices.

Automatic Basketball Inflator

Matt Szafranski
Grant Walthall

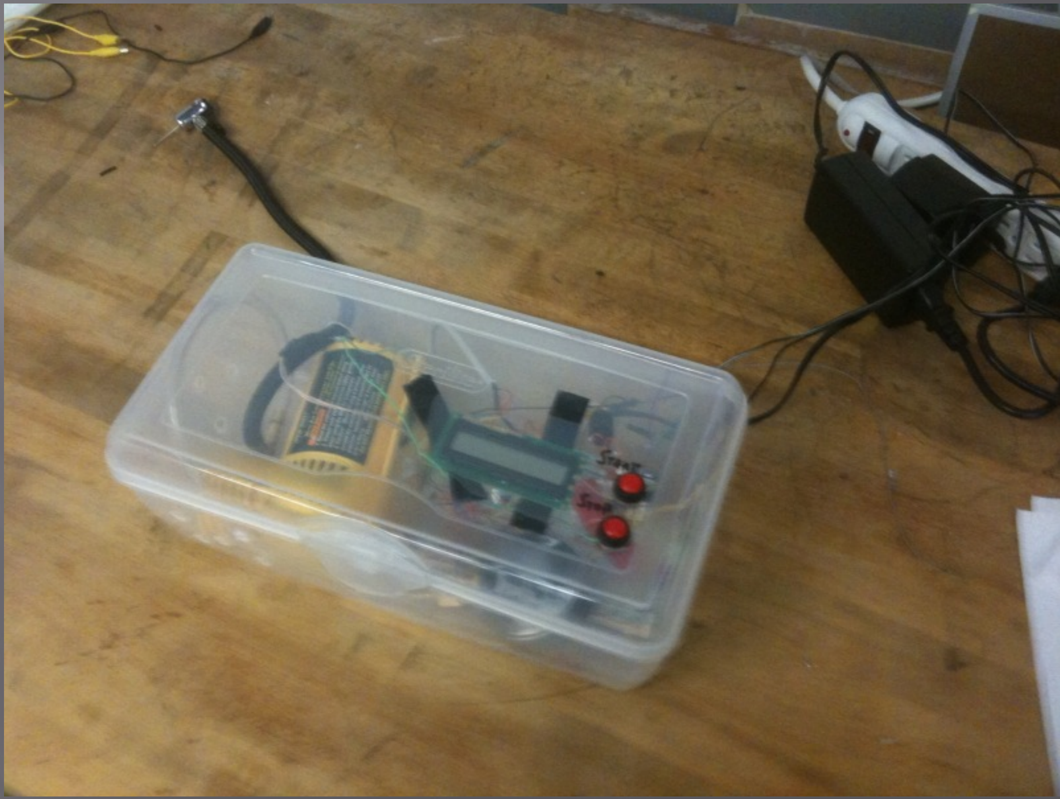
Project Objectives

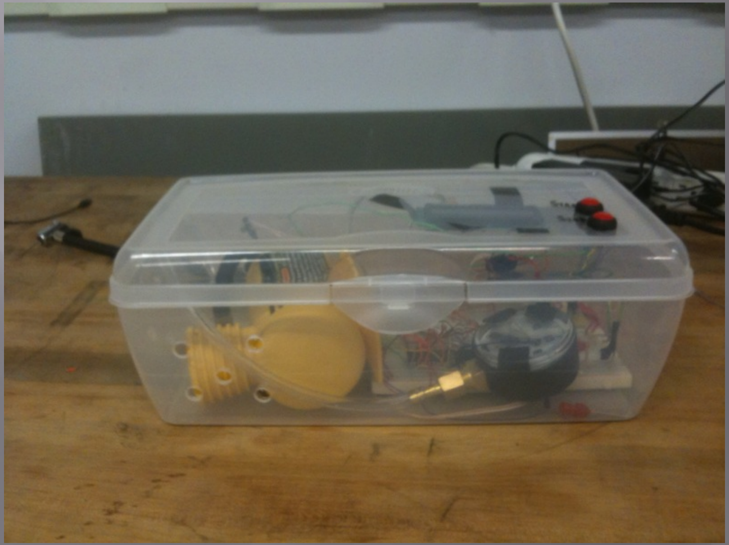
- Create a device that could inflate a basketball to normal PSI rating
- Be able to do everything automatically as well as have a manual stop in case of over inflation

Inputs/Using the PIC/Output

- Start and manual stop, as well as the PSI reading. When it reached the desired PSI it would shut off
- Interrupts were use to sense when a button was pressed or the desired PSI was reached.
- The final output was a fully inflated ball that automatically shut the machine off when the target PSI was reached, working as planned

Final Product

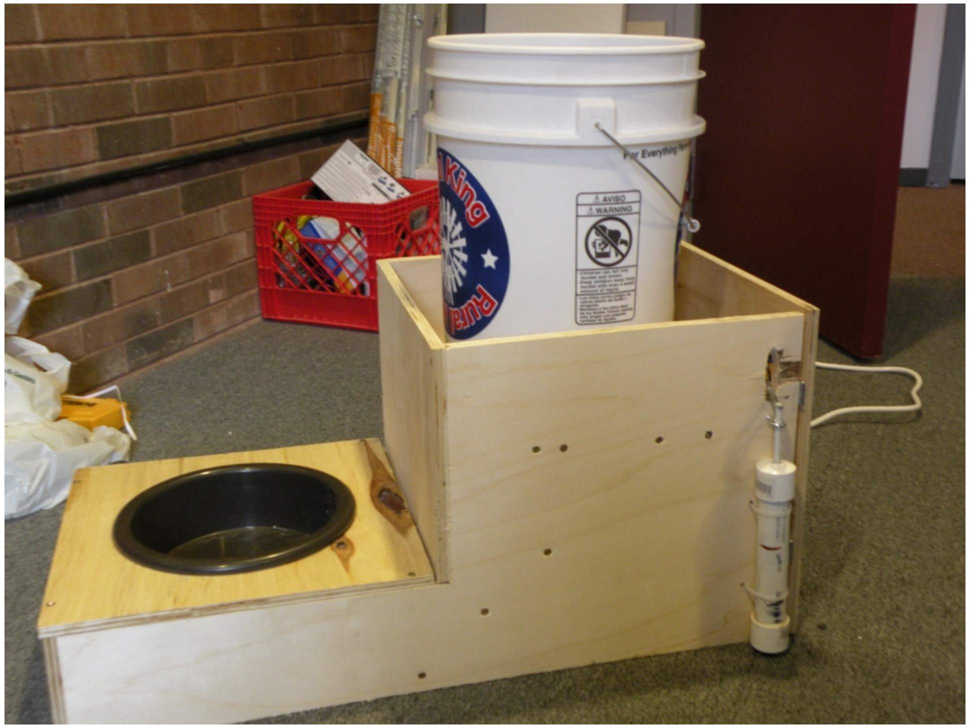




Lessons Learned

- The smallest things cause the biggest problems
- Make sure you have the right size power supplies
- Don’t put off solving problems until the last day. Chances are it’s an easy fix if you ask for help.

Automatic Water Dispenser



A project for ME430 – Mechatronics
by
Kyle Roberts &
John Hollingshead

Project Objectives

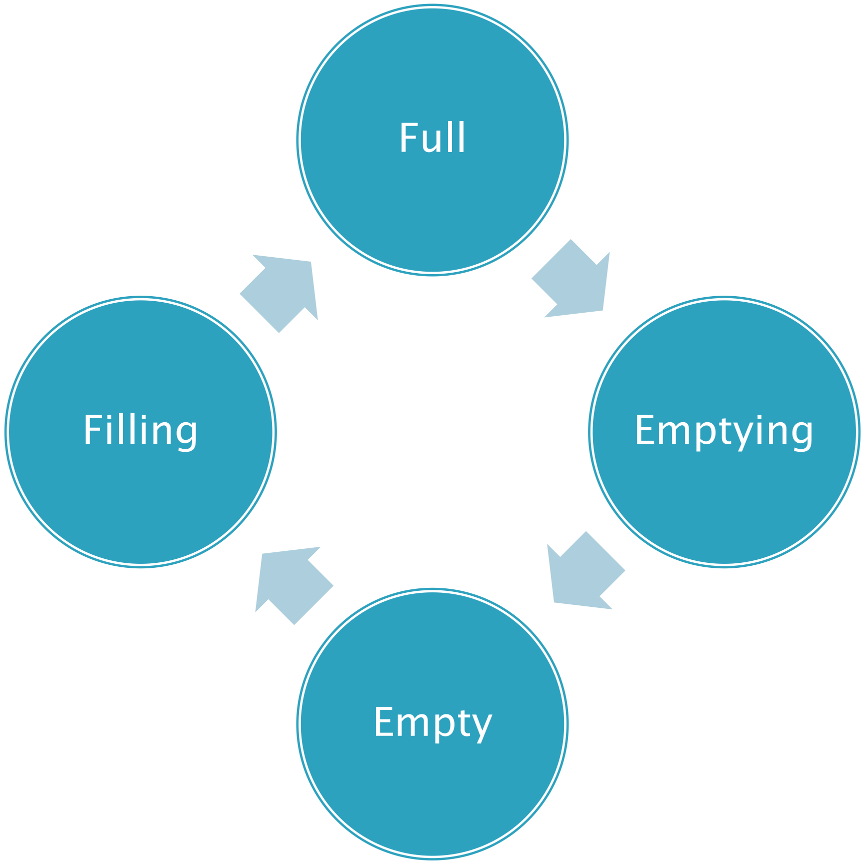
- Prevent pets from running out of water
- Secure pet's water bowl to prevent spills
- Refill bowl when water level is reduced below specified level
- Store 5 gallons of water for several days of consumption by pets

System Design

- Inputs
  - Top button
    - Identifies when bowl is full based on manometer
  - Bottom button
    - Identifies when bowl is empty based on manometer
- Outputs
  - Solenoid Valve
    - PIC used as controller to open or close valve

Device states

- Full
  - Valve closed
  - Top button pressed
- Emptying
  - Valve closed
  - No button pressed
- Empty
  - Valve closed
  - Bottom button pressed
- Filling
  - Valve open
  - No button pressed



Results

- Manometer effectively measured fluid height
- Buttons sensed full and empty states
- Solenoid valve responded to buttons
- Valve control not effective, valve would not stay open sufficient period of time

Advice

- Start early, you never know what will break or malfunction
- Choose manufactured equipment over home-built designs; it is proven to work
- Make something that is useful outside of class
- Pick something fun!

Autonomous Harmonica Player

By: Robb “Skip” Leu & Brandon Duke

Project Objective(s)

- Design a machine which will pass air through a harmonica (‘Mechanical Lungs’)
- Write a program to read a song and then instruct the player (‘Conductor’)
- Run the ‘mechanical lungs’ and ‘conductor’ simultaneously

What/How? → Mechanical Lungs

- What: Plexiglass parts secured with a Plastics Epoxy. Piston created from PVC pipe with rubber air hose attached to cap.
- How: Made drawing in Solidworks and sent it to Ron Hoffman to machine parts with the laser cutter.

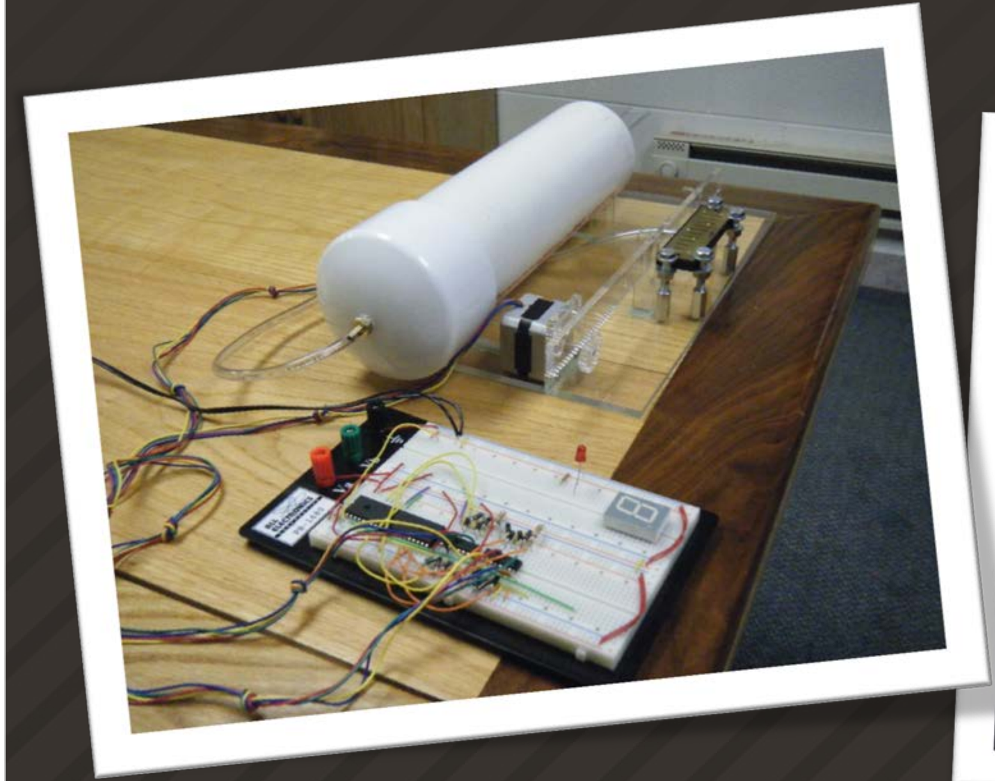
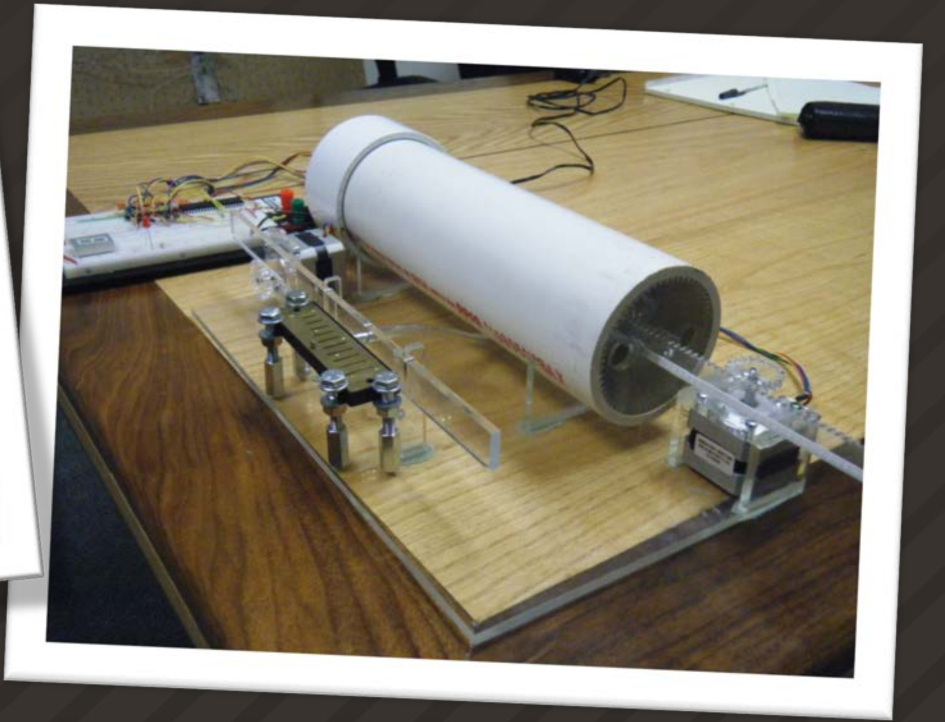
What/How? → Conductor

- What: MPLab Program which translated the notes into a physical position
- How: Program counted “steps” of the stepper motors. Each motor ran individually at different speeds.
- The PIC was used as a controller for both motors. Ports B and C were used as outputs for the different steps.

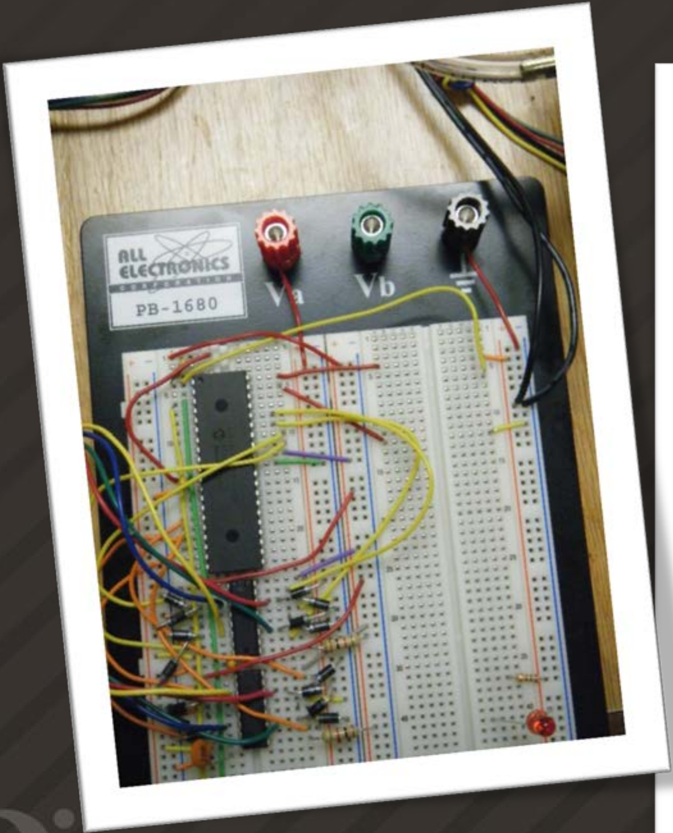
Results

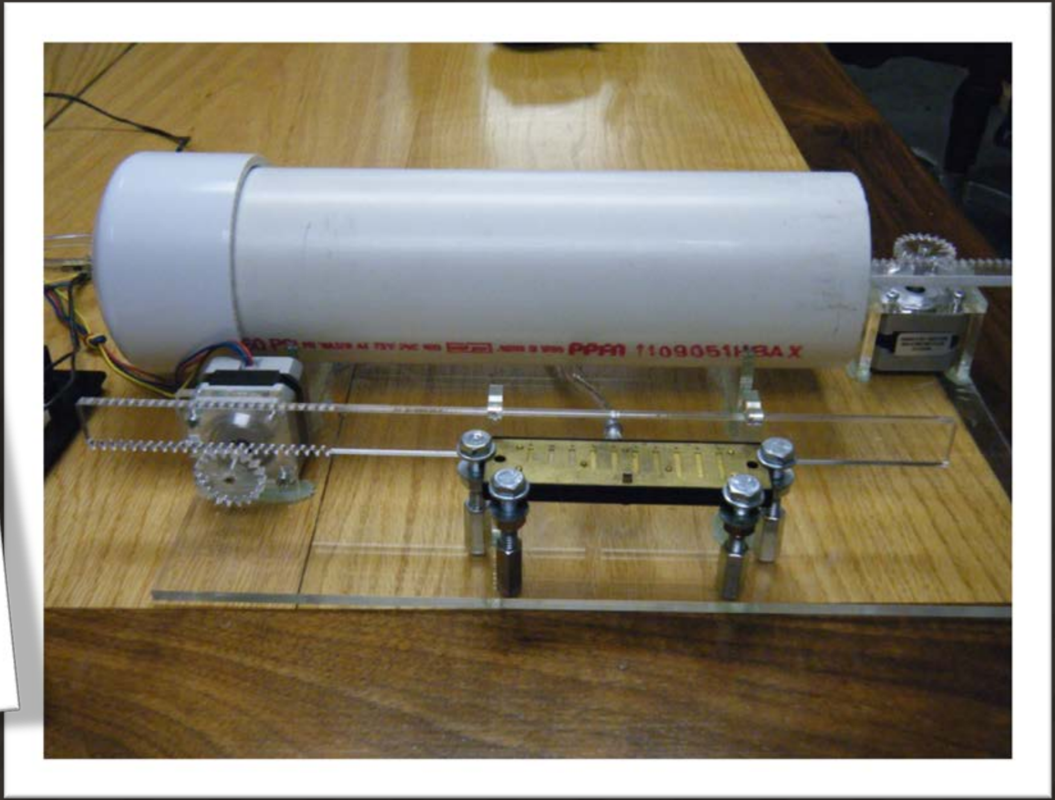
- Success!... Sort of…..
- The motors synchronized and ran well.
- However….
- The piston did not generate enough air to produce audible vibrations in the harmonica.

Autonomous Harmonica Player

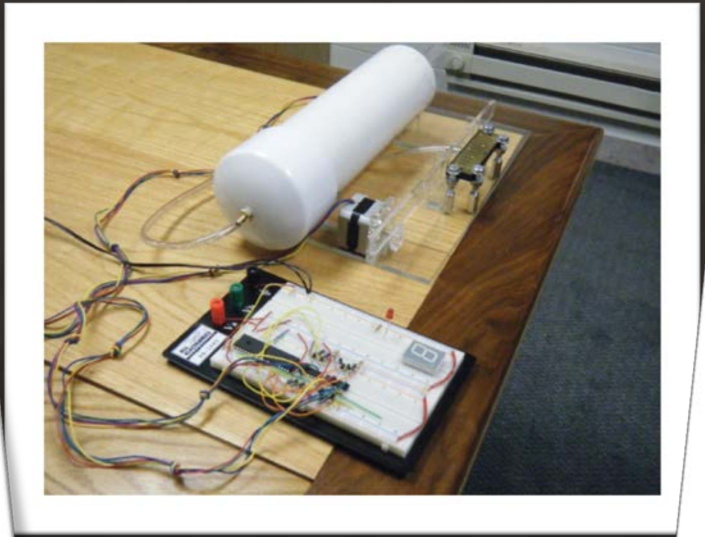



Autonomous Harmonica Player





Music To Our Ears!!



Recommendations

- Be sure to know that a stepper motor will be able to produce the amount of torque necessary to drive the mechanical functions of the project.
- Be sure to plan ahead so as to create an available window to correct any unforseen complications.

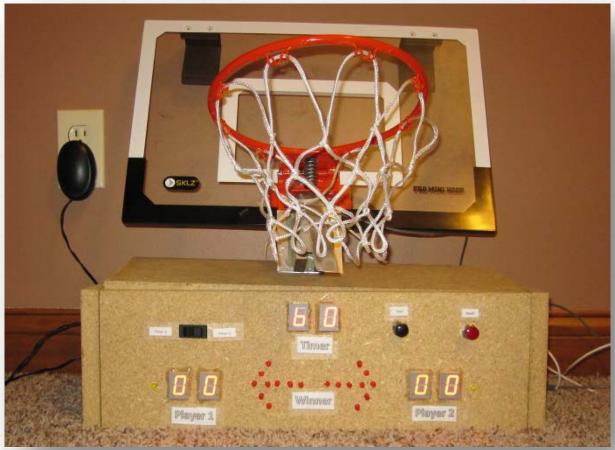
Two Player Basketball

Tyler Johnson

Clayton MacCrindle

Project Objective & Result

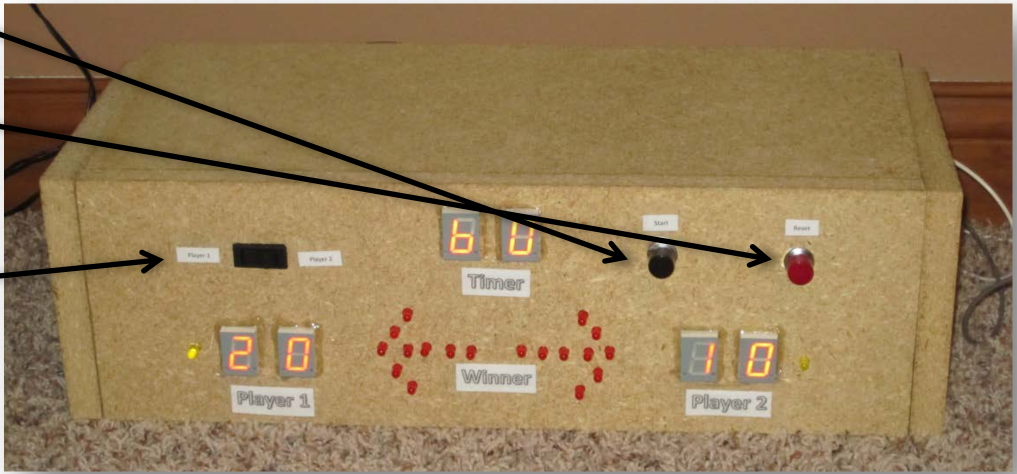
- Construct a two player arcade style basketball game
- Player 1 plays for 60 seconds followed by player 2
- Determine winner and use LED arrow to point to winner
- Be able to reset game at any time without having to unplug it



- Our Project was a success. It accomplished all of our objectives.

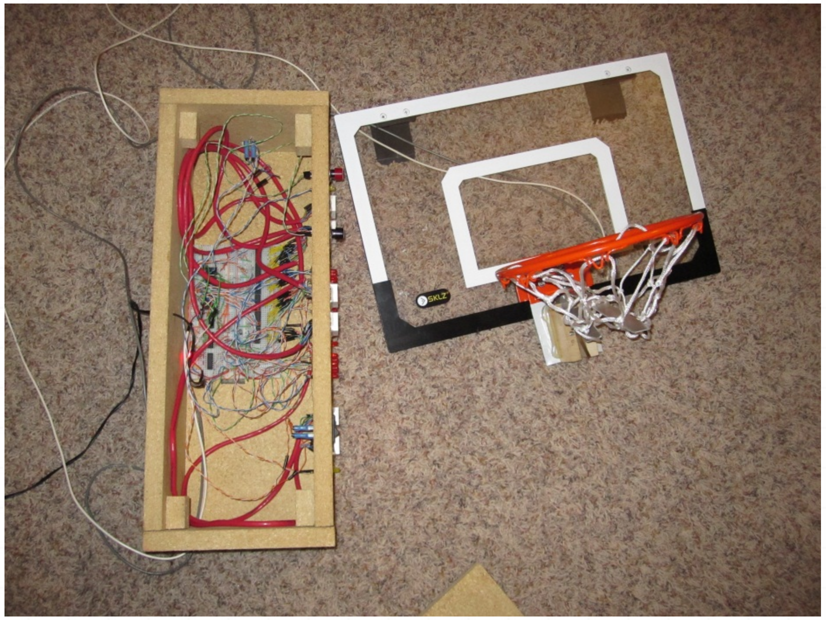
Inputs

- Start
  - Interrupt
- Reset
  - Interrupt
- Player Toggle
  - Non-interrupt
- Basket-Made
  - Interrupt



How Did We Use The Pic?

- Timer set up to flag interrupt every second
- Interrupts for
  - Basket-made
  - Start
  - Reset
- PIC displayed outputs on 6 different 7-segment displays
  - 2: Timer
  - 2: Player 1 Score
  - 2: Player 2 Score



Outputs

- Timer
  - Interrupt
- Player 1 Score
- Player 2 Score
- Player 1 LED
- Player 2 LED
- Winner

Lessons Learned

- Be ambitious but not too ambitious
- Use good wire, stranded Ethernet does not work well
- Test code in pieces, then bring multiple pieces together and test them

BATMOBILE
“IT’S SUPERSONIC”

Dr. Olson
Mechatronics
ME430-01

Todd Deskin
Cody White

Final Product

Batmobile fully operational as intended

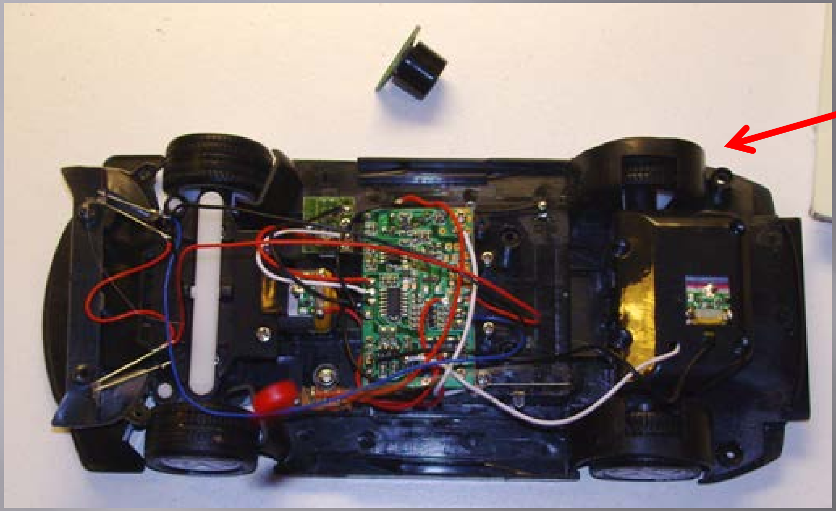
Overview

- Objective
  - Create a self driving car that will detect objects, reverse, and steer away
- Inputs
  - Sensors receive object signal
- Outputs
  - Left turn when right sensor signaled
  - Right turn when left sensor signaled
  - Reverse and turn right when both sensors signaled

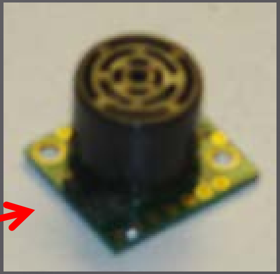
Parts List

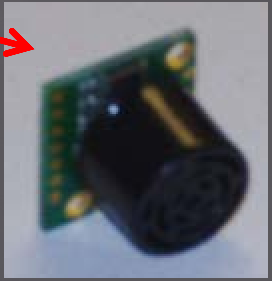
- RC Car
- PIC 18F5520
- Ultrasonic sensor (2)
- AA batteries (4)
- Breadboard
- Potentiometer

The Beginning



RC Car guts

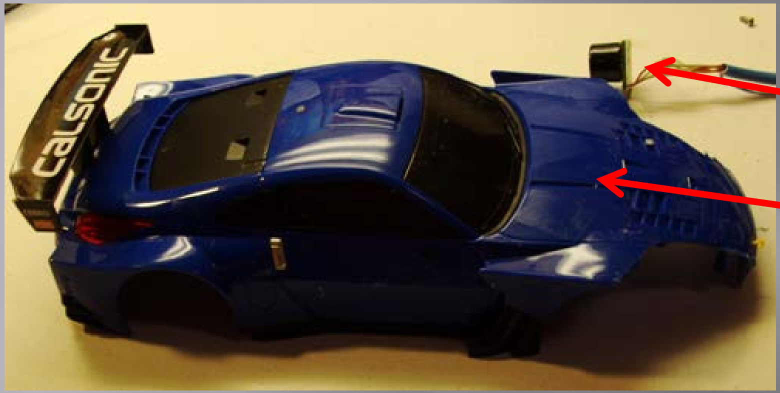
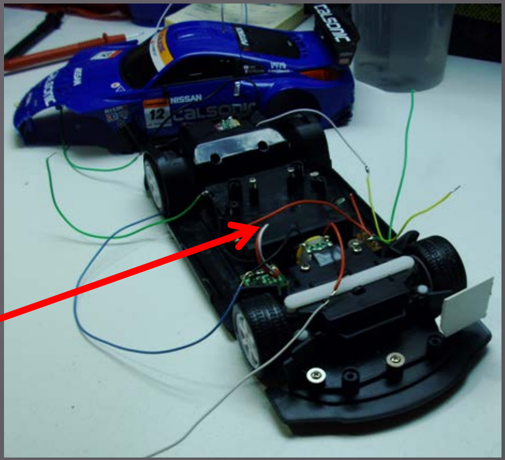




Ultrasonic sensors

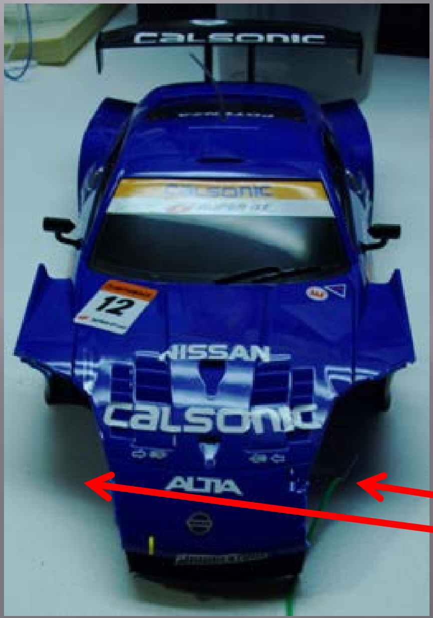
Guts and Shell

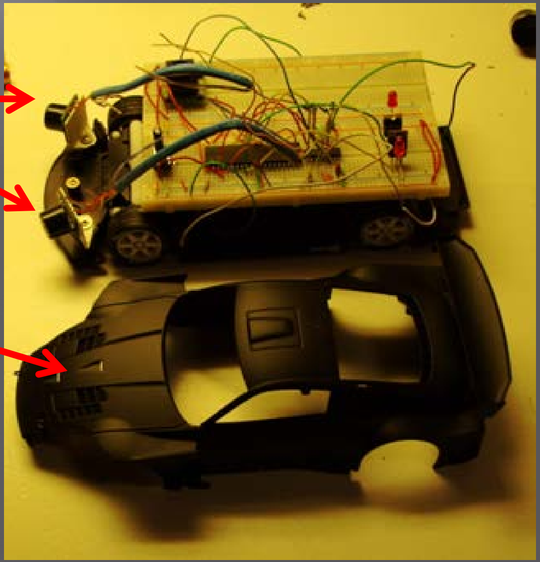
The Transformation



Sensor cable addition

Stickers removed

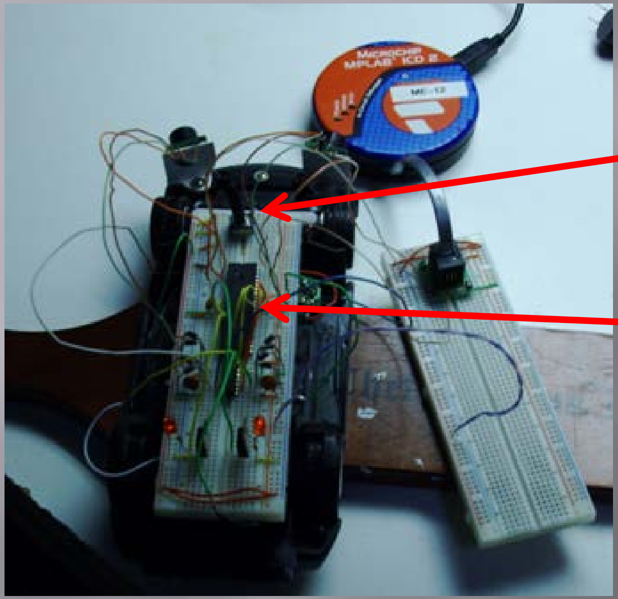
Circuit board removed


Fenders cut


Sensors

Black Shell

Programming the PIC



Potentiometer

PIC 18F5520

- While statement for continuous function

- If statements were used to detect the sensors
  - if no sensor go straight
  - if left sensor turn right
  - if right sensor turn left
  - if both sensors reverse and turn right

- Timer in place to time turns and reversing

Advice

- Order parts early
- Start programming early
- Do not wait until the night before

Buddy Drink

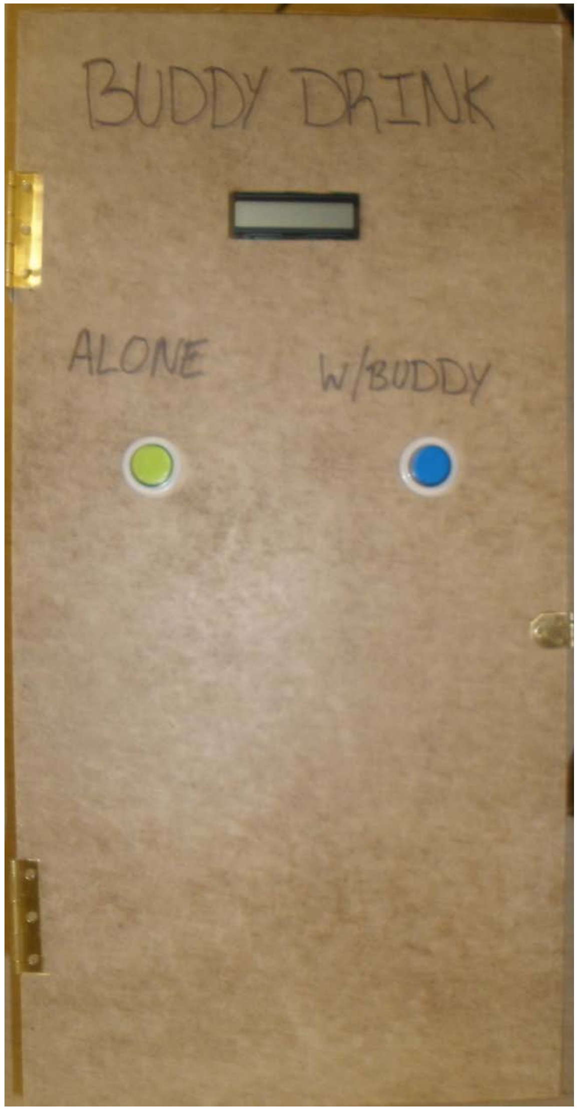
Andrew Meiers

Mehul Patel

Winter 09/10

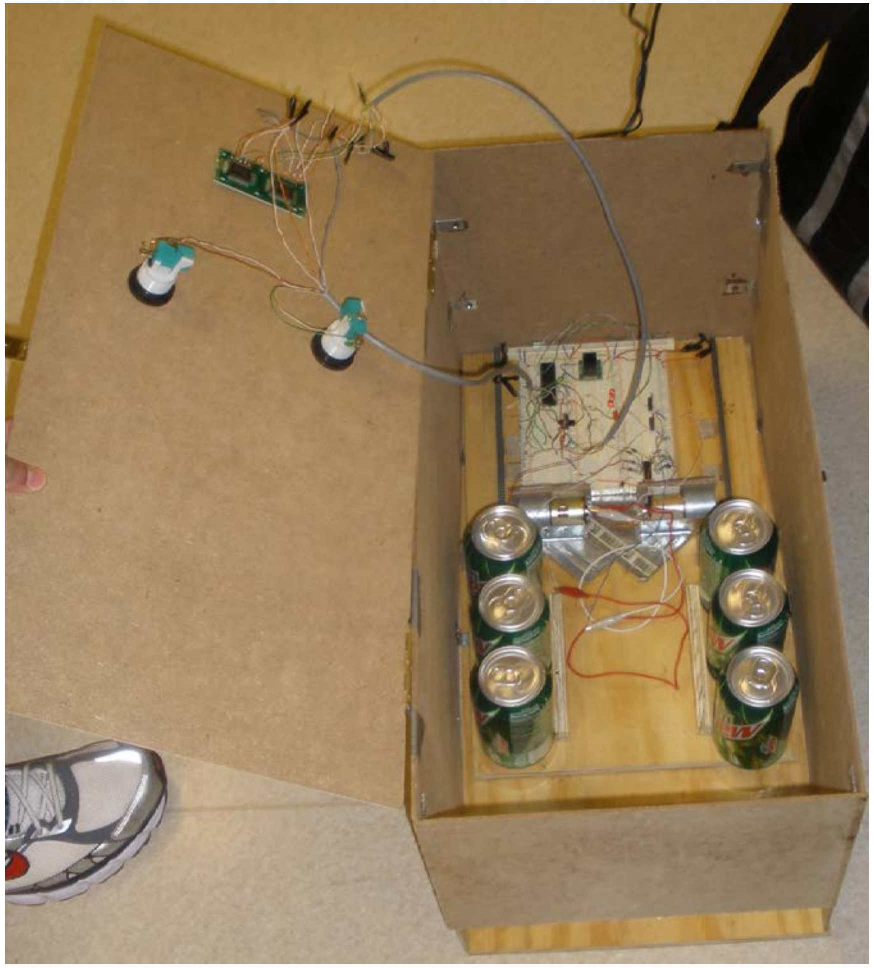
Objective

- You select whether you are alone and want one drink or if you have a buddy with you and want two drinks.
- It will then dispense one or two drinks.
- When the drinks run out it will say “Refill Machine”



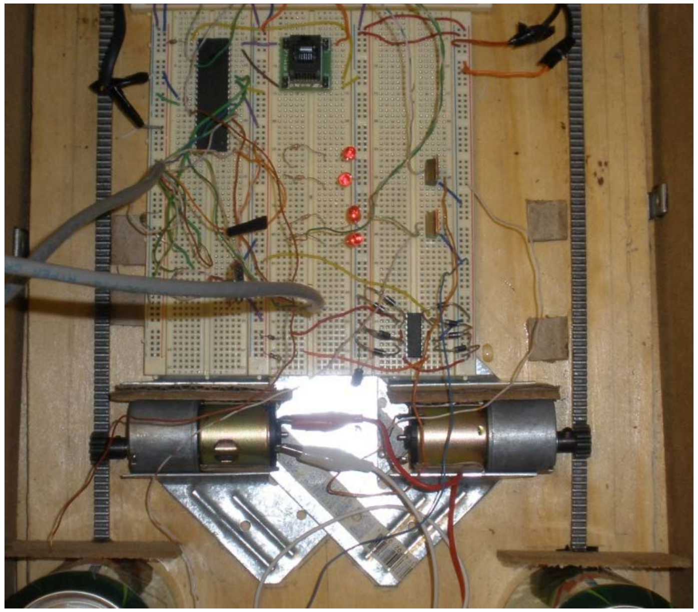
Inputs

- Pushbuttons (RB0 and RB1)
- Alone or w/Buddy



PIC as a Controller

- Within our code we used high priority interrupts with two button pushes and if statements in the while loop.
- Within interrupt
  - Turns on motor
  - Updates LCD display
- Motor Control
  - MOSFETs were used



- Within while loop:
  - Counts the number of times a button is pressed to determine when a refill is needed.

Outputs

- Motor 1
  - Dispense your drink
- Motor 2
  - Dispense friend's drinks
- LCD
  - Displays who's drink is being dispensed.
  - Displays when machine needs refilled.

Results

- Everything worked as planned
  - “Alone” button turns on Motor 1
  - “w/Buddy” button turns on Motor 2 and then Motor 1
- LCD displayed everything
  - “Please select Alone or W/Buddy
  - “Vending your drink”
  - “Vending your friend’s drink”

Advice to Future Students

- The potentiometers given in the lab kit do not stay in the bread boards well at all. Either find a different type of potentiometer to use or solder leads to the ends of the potentiometer so that it can be put in the breadboard
- If you have a high voltage motor you can easily burn out your H-bridge when the motor starts up which can then burn out your PIC.
- When you order the PICs online have both you and your partner buy PICs so that you have enough in case you burn multiple out.

Hot Chocolate Mixer

Team 22
Kyle Stevens
Tina Trivett

ME430-02 : Mechatronics
Dr. Olson

Project Objecvtive

Stir a cup of hot chocolate for a designated time inputted to the PIC by the user while reading the temperature of the drink and buzzing when the process is complete.

Inputs and Outputs

- INPUTS:
  - Buttons to choose the amount of time to stir the hot chocolate
  - Temperature of the beverage

- OUTPUTS:
  - Liquid Crystal display to read the temperature and stirring time
  - Running motor for the desired time

PIC Usage

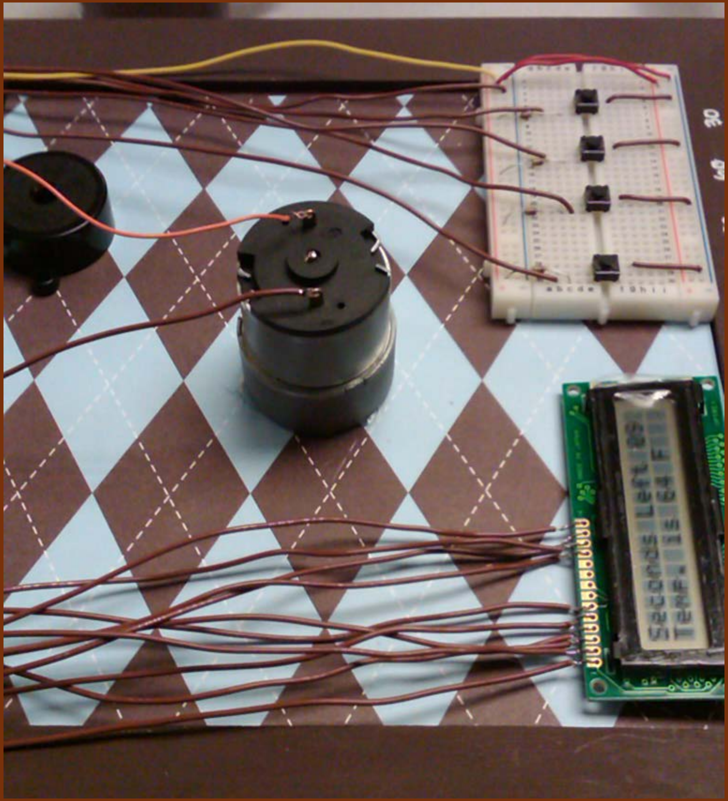
- Interrupts
  - For button presses
  - To control time displayed on the LCD
- Read/Convert ADC
  - To read the temperature
  - Output the temperature to the LCD
- PWM
  - Output to buzzard
  - Output to motor

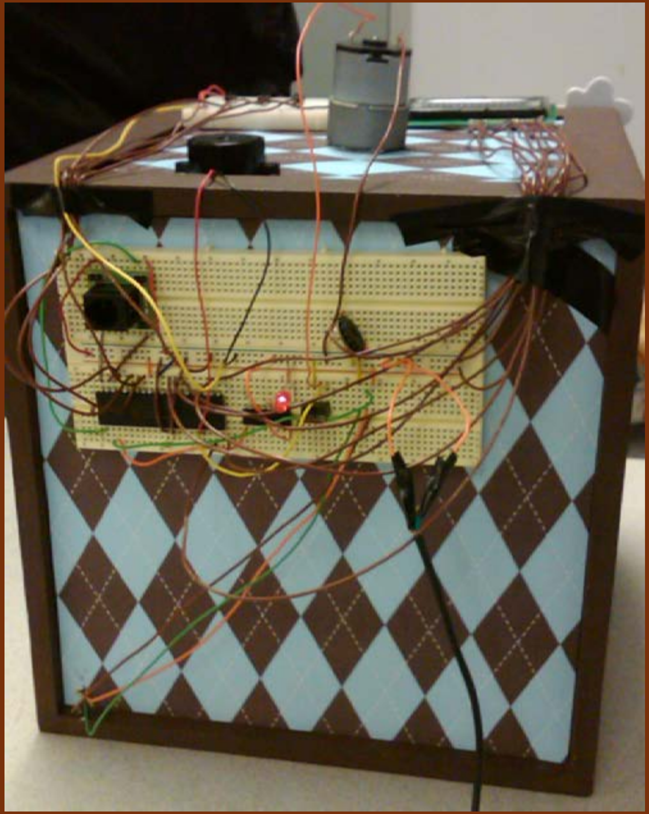
Results

- All worked as planned!
- The hot chocolate stirrer will stir a cup for a desired amount of time inputted by the user.
- The hot chocolate mixer also reads the temperature of the beverage.
- Beeping from the buzzard announces when the beverage is done being stirred.
- The motor spins slightly slower than expected, but can do the job.

PICTURES

More Pictures





- Top View
- Back View

Advice for the Future

- Order parts early incase you encounter a problem with what you ordered.
- Check to make sure the power supply you are using is giving a strong, steady voltage.
- Buy a motor greater than what you need, so you can adjust the speed by programming.
- Pay attention to the gearing on the motor – it is important.
- Spend time on your apparatus and not just coding – presentation is important!
- Ask questions – it will save time and headaches.

ME 430 – Mechatronics Automated Drink Mixer

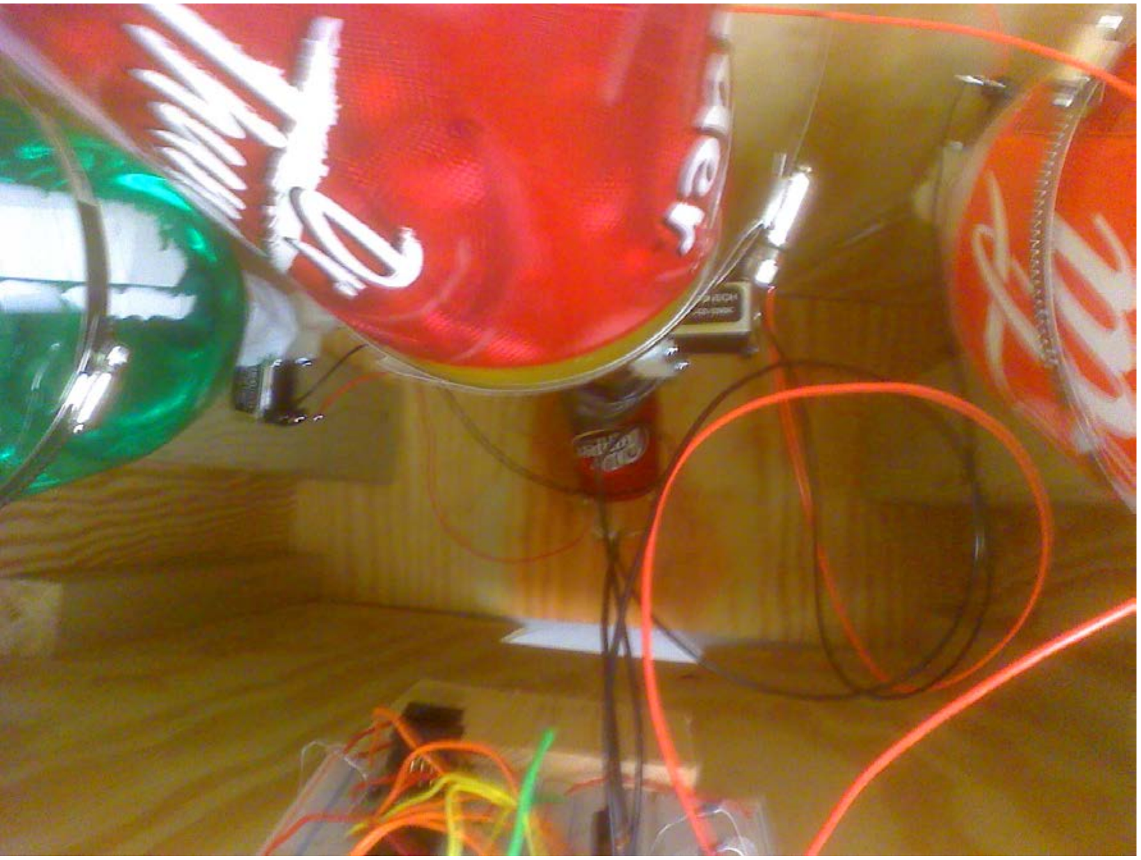
Alex Trent

Shaun Wendel

Project Objective

- Build and program a device to perfectly and consistently measure drink proportions
- Program for three drink combinations with three different ingredients

How did we do it?



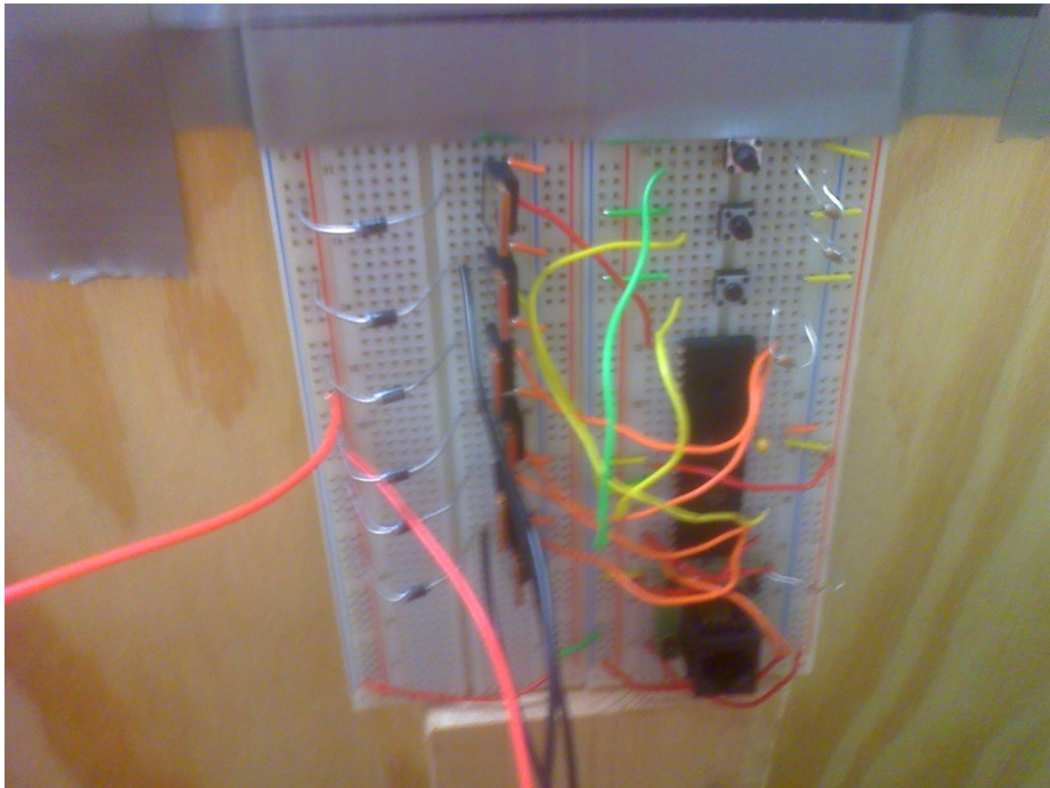
- Inputs were three simple button switches on the breadboard
- These switches were connected to the PIC and designated which drink mix to output

PIC

- The PIC was used to control the valves through the MOSFETs
- We used recent button states and if loops due to initialization issues with interrupts that defied logic
- Delays were used to allow for the physical flowing of the fluids while not overloading the power supply current allowance

Outputs

- Solenoid valves were switched either open or closed based on the PIC controls
- Different combinations of valves created different drinks



Results

- It worked with modification
  - Originally a pair of valves controlled the exact proportions used
  - Pressure differentials without a proper venting scheme resulted in having to downgrade to one valve per ingredient
  - Simple fix to the offset proportions would be to just keep the ingredient containers full

Advice

- Shipping errors can cause snowballed problems
- Do a proof of concept early
- It’s never too early to start

LeBron James/Dwyane Wade Dunk-off

Mike Reis
Nathan Paulsen

Objectives

- Create a two-player game that would encompass topics learned in ME430 while still being fun to play.
- Utilize two PIC chips and create some form of communication between the two during operation.

I/O list

Inputs

- Lever switch from hoop
- LED interrupt from PIC 1 to PIC 2

Outputs

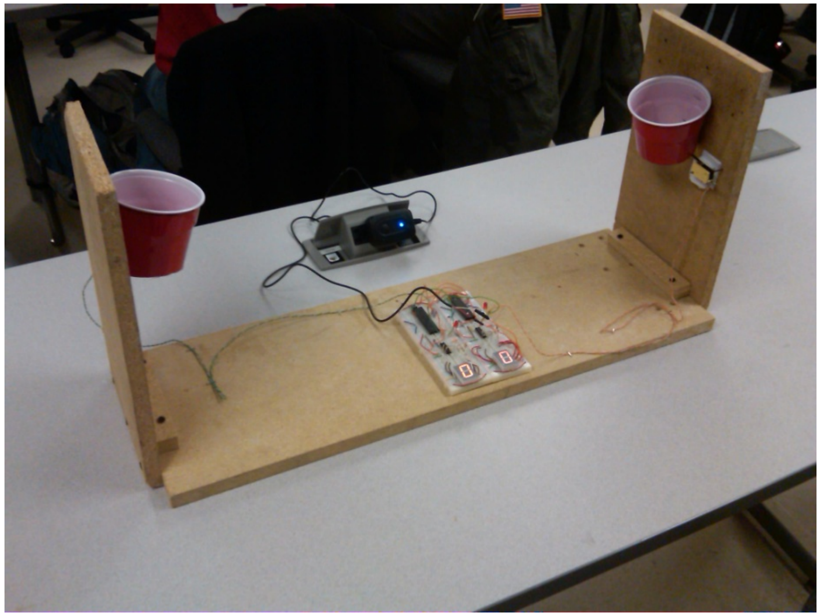
- 7-segment display
- LED’s

Role of the PIC

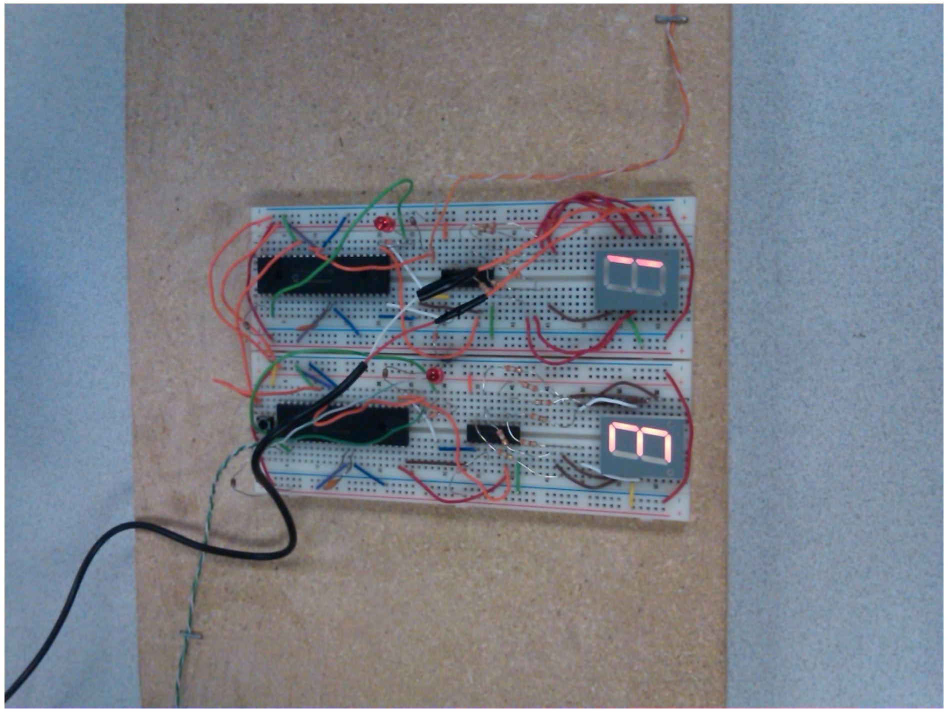
- C functions that were used
  - PORTB
  - Timers
  - Delays
  - Stdio
- The lever switch would cause an interrupt which would increment the 7-segment display.
- Both switches were connected to port B pins.
- Delays were used in the interrupt to protect against faulty switch triggers.
  - They were also used to light up the “win” LED’s for a set amount of time before the reset.
- Send a signal to the other PIC chip to trigger a reset.

Outputs from the PIC

- On each trigger, the code would increment a variable called “score”, signifying that a player had sank a basket. This variable would then be sent to the display.
  - There were two PIC chips on two boards so the code was simply used in both PIC’s.
  - The only wires that will be connecting the two are those for the global reset switch and the wires from RA0 on board 1 to RB2 on board 2 and vice versa on board 2.
- When one player hits “10”, meaning he was previously at 9 and triggered and then triggered an interrupt, the “win” LED will flash for 1.5s, signifying that the player has won. At this time the game will reset and the display will read “0”.
- When one player wins the game, the signal from the LED pin will also be sent to the opponents PIC board. This will trigger an interrupt and reset that game as well.



Pictures



Advice for future projects

- Simplicity is key.
  - Don’t make your project too complicated if it’s not necessary
- Test the code in sections, not all at once.
- If 7-segment displays are bought, there is a chance that the supplier will send CA instead of CC, or vice versa. Don’t be surprised when you blow out a display for no good reason other than the fact that the company shipped the wrong type.

Conclusions

- The lever switches proved to be the most difficult aspect of this project, but once they were debugged the project moved along swiftly.
- The game worked as designed with no problems.
- The construction of the game environment was tricky at times, but ended up working as well.

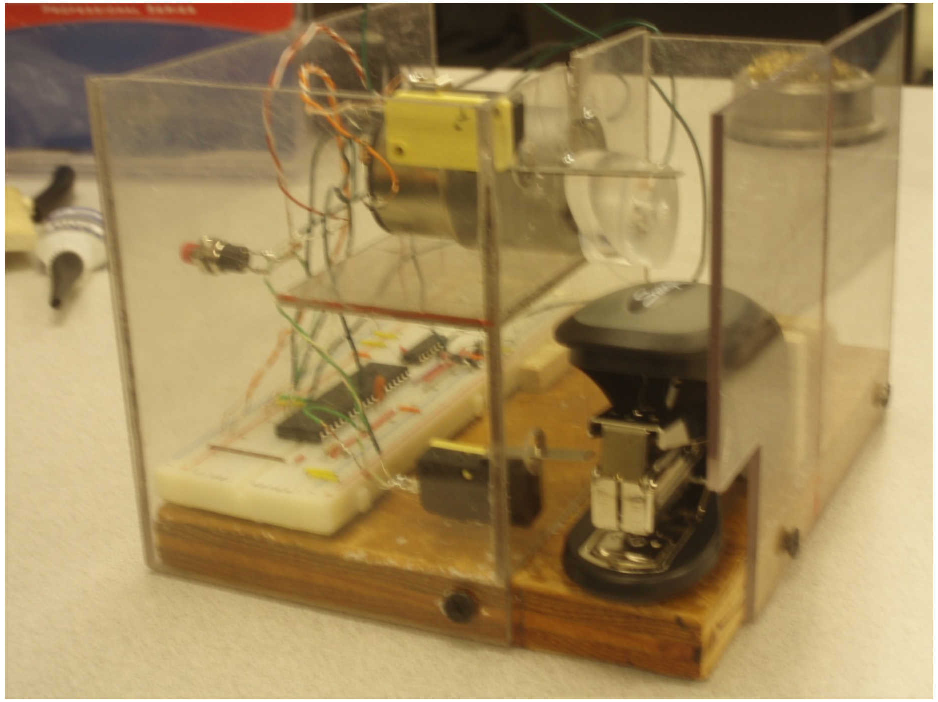
ME 430 - MECHATRONIC SYSTEMS

ELECTRIC STAPLER

By Peter Richards and Shayna Banet

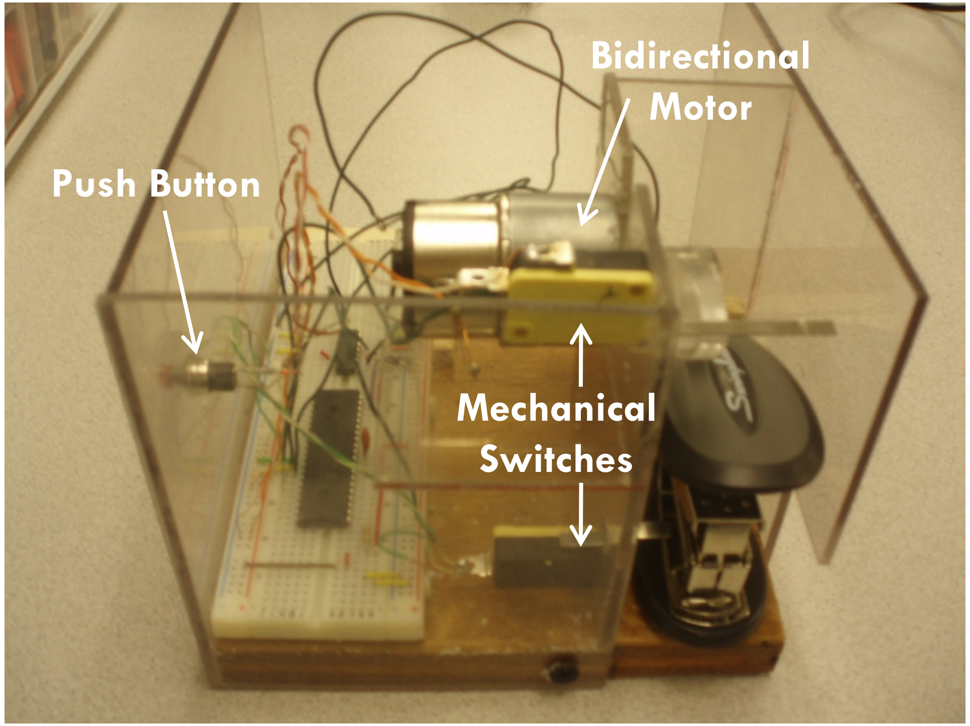
Project Objective

- Build an automatic stapler that runs off of electricity in order to remove any effort required by the user.



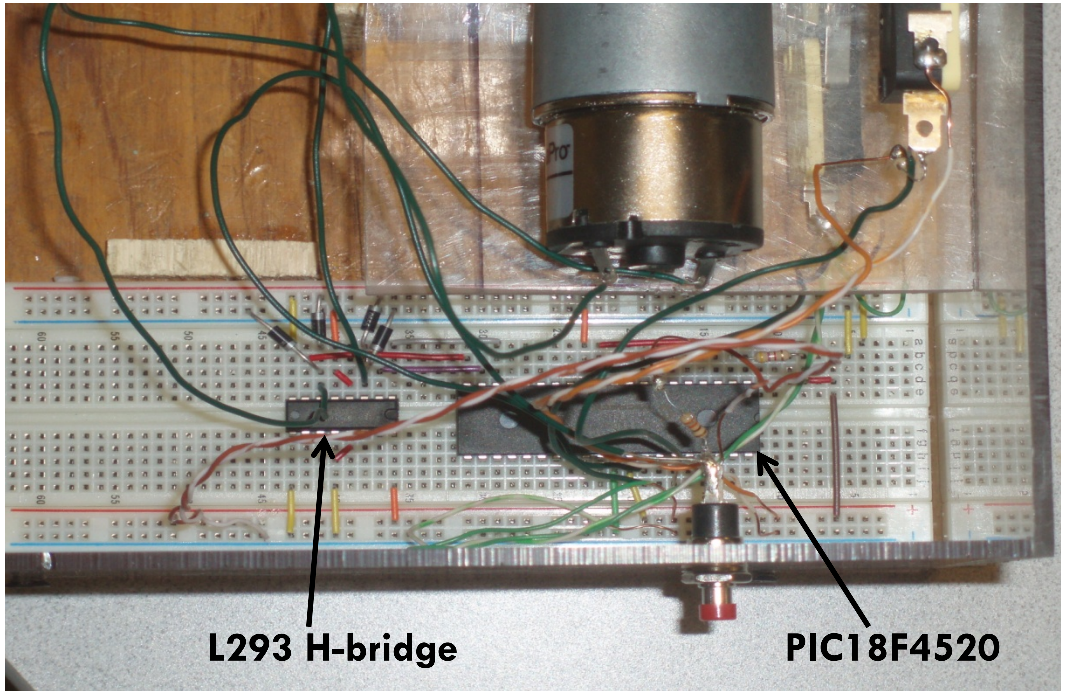
Inputs and Outputs

- Inputs:
  - 1 push button
  - 2 mechanical switches
- Outputs:
  - Bidirectional motor



PIC Usage

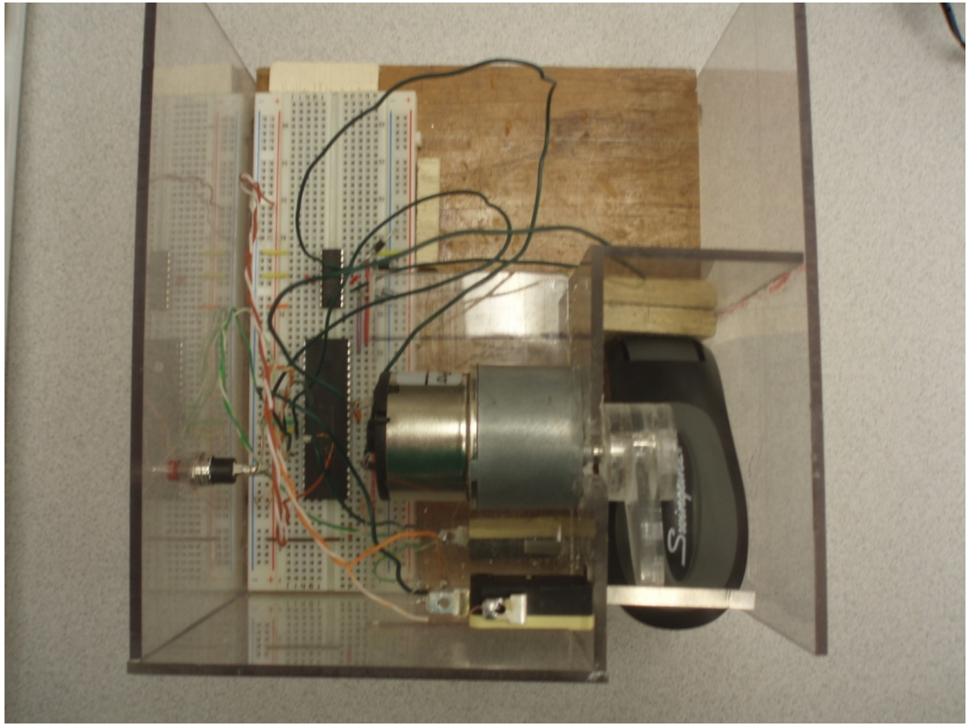
- PIC:
  - 6 digital pins: 3 input pins for switches, 3 output pins for H-bridge
- Code:
  - Series of "if" statements controlling motor in response to sequential triggering of switches



Results

- Troubleshooting:
  - Motor attachment broke and had to be remade, was reinforced the second time
  - Voltage reducer could not handle amps, had to switch to using two power supplies (5V and 15V)

- In the end…
  - It worked!



Advice

- If you get a free part (from the supply closet) make sure you know exactly what it does and how much voltage and current it can handle (look up spec sheet online)
- Be open to adjusting your project ideas. Build, test, fail, repeat.
- Plan to finish the project a week before it is due. This will leave time to troubleshoot.

LockBot

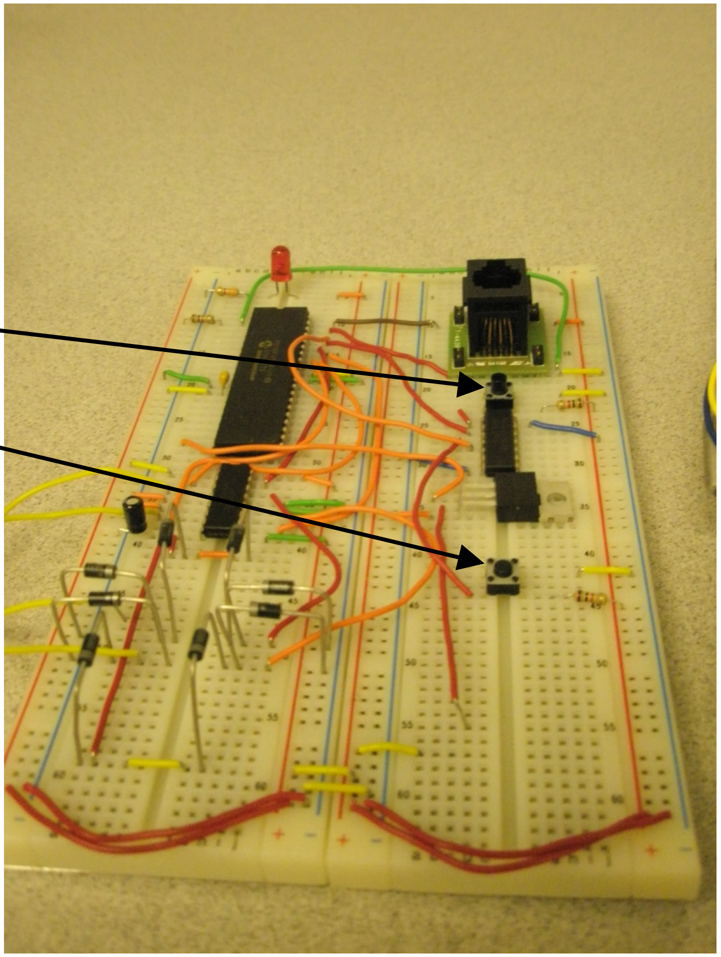
Kristopher Stuckey
Kyle Chmielewski

Objective

- Create something that would put a combination into a combination lock and then pull it open.

Inputs

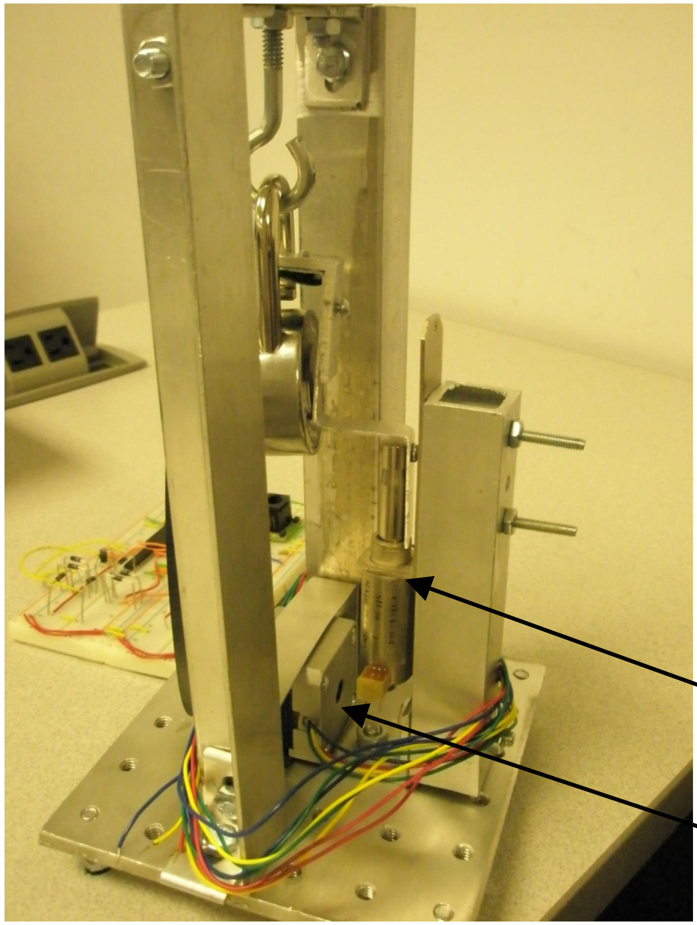
- Inputs:
  - “Go” Button
  - “Reset” Button



Pic

- Pic
  - Controlled a stepper motor (with H-bridge)
  - Timer interrupts stepped motor
  - Count number of steps to put in correct combination
  - Signaled a MOSFET to turn on a solenoid at end of combination

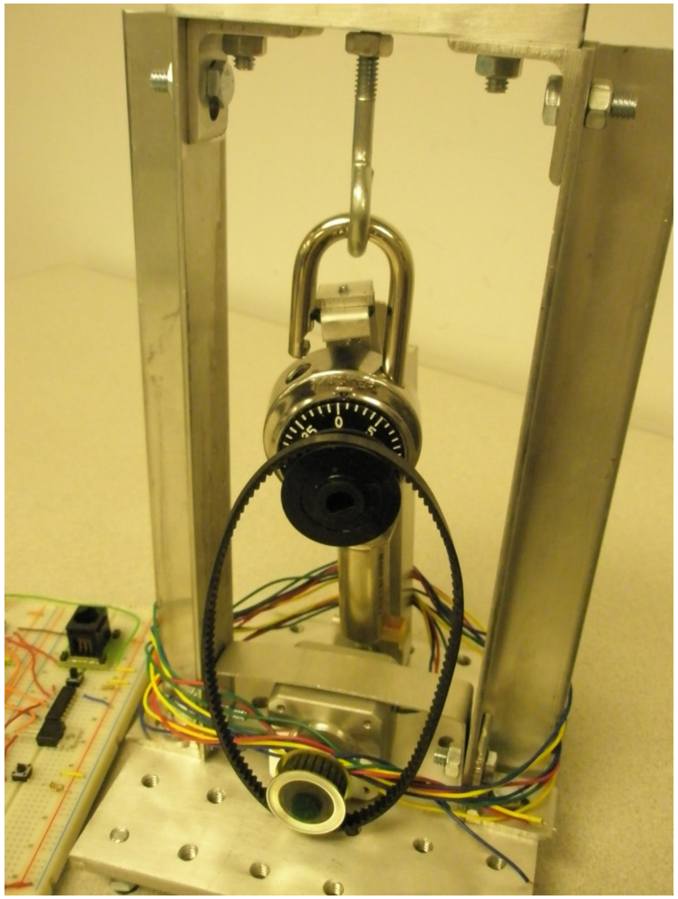
Outputs



- Outputs:
  - Rotation of lock via stepper motor and pulleys
  - Opening of lock via solenoid and custom bracket

Results

- It worked!
  - After several test runs to get the stepper motor the right number of steps for the combination
  - Solenoid required large power supply to open lock



Advice

- If using a solenoid, get power supply with plenty of amps (3 A)
- Use voltmeter to check outputs of pic when things randomly quit working
  - Our pic went bad and we checked everything for two hours before we discovered it was the pic.
- Use snubber diodes - H-bridges can burn

Mini Skee-Ball: A Tabletop Game

ME430: Mechatronics

Charles Roark & Zack Brune

Objective

- Create an apparatus similar to skee-ball, but use a mesh netting to bounce the ball instead of a ramp to roll it
- Must keep track of time and three different levels of scoring
- Incorporate an LED display

Inputs

- Signal from lever switches to trigger an increase in score
- Signal from a flip switch to power device for gameplay

Controller

- PIC 18F4520
- Interrupts looking for rising edges used to indicate score increase from lever switches
- Timer used for specified length of gameplay when flip switch was engaged
- Delays used to make scoring function properly

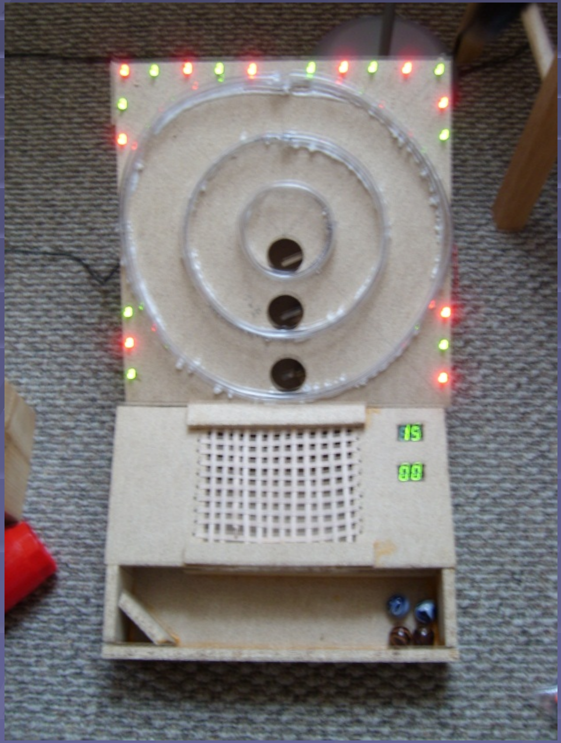
Outputs

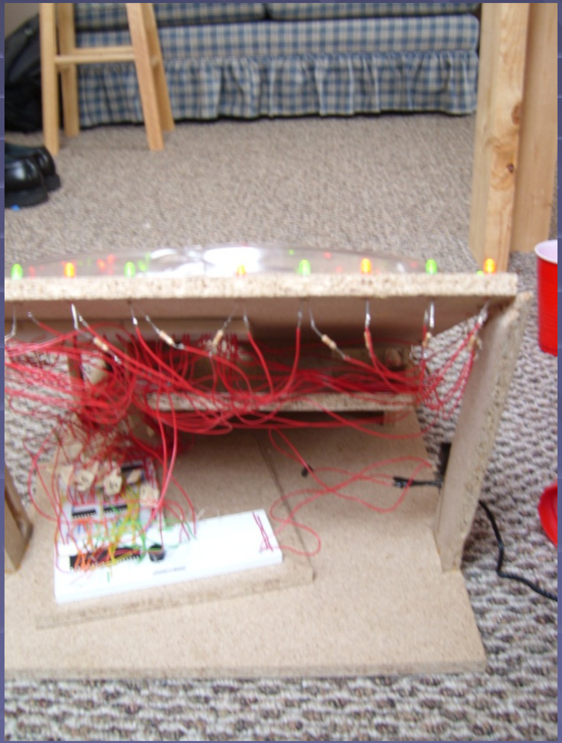
- Score on dual 7-segment display (18 pin)
- Time remaining in game on dual 7-segment display (18 pin)
- LED display around game perimeter (hard-wired)

Results

- The apparatus worked as proposed (score, time, LEDs, switch all functioning)
- Switch for 3 point score stuck a couple times due to use of super glue, but worked enough to trigger a score increase

Photos





Recommendations

- Do not use a delay function in an ISR
- Pick something of moderate difficulty to build and moderate difficulty to code
- Get as early of a start as possible
- Build code in pieces and test circuit and outputs to make certain everything is working as needed
- If particle board is used, be sure to do all cutting on it before mounting

MORSE CODE DECODER

ME 430 – 01 Winter 2010

Rose Bolander
Alex Emley

GOAL:

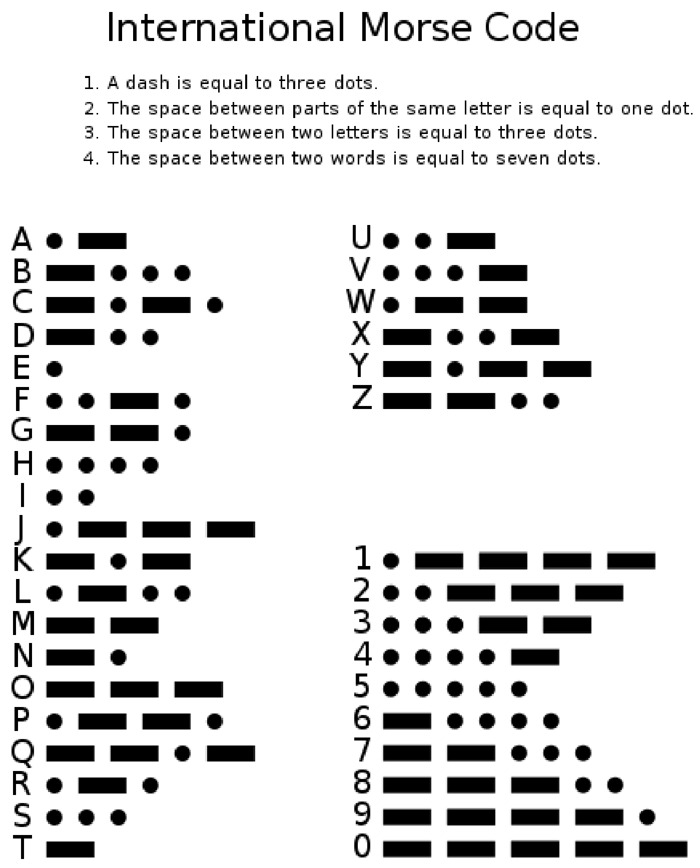
Develop a machine that decodes a basic Morse code transmission from dots and dashes into letters.

Inputs:

- Buttons

Outputs:

- Letters on LCD
- Fail letter LED



International Morse Code

1. A dash is equal to three dots.
2. The space between parts of the same letter is equal to one dot.
3. The space between two letters is equal to three dots.
4. The space between two words is equal to seven dots.

A	● ▬	U	● ● ▬
B	▬ ● ● ●	V	● ● ● ▬
C	▬ ● ▬ ●	W	● ▬ ▬
D	▬ ● ●	X	▬ ● ● ▬
E	●	Y	▬ ● ▬ ▬
F	● ● ▬ ●	Z	▬ ▬ ● ●
G	▬ ▬ ●		
H	● ● ● ●		
I	● ●		
J	● ▬ ▬ ▬		
K	▬ ● ▬	1	● ▬ ▬ ▬ ▬
L	● ▬ ● ●	2	● ● ▬ ▬ ▬
M	▬ ▬	3	● ● ● ▬ ▬
N	▬ ●	4	● ● ● ● ▬
O	▬ ▬ ▬	5	● ● ● ● ●
P	● ▬ ▬ ●	6	▬ ● ● ● ●
Q	▬ ▬ ● ▬	7	▬ ▬ ● ● ●
R	● ▬ ●	8	▬ ▬ ▬ ● ●
S	● ● ●	9	▬ ▬ ▬ ▬ ●
T	▬	0	▬ ▬ ▬ ▬ ▬

INPUTS:

3 push buttons using interrupts:

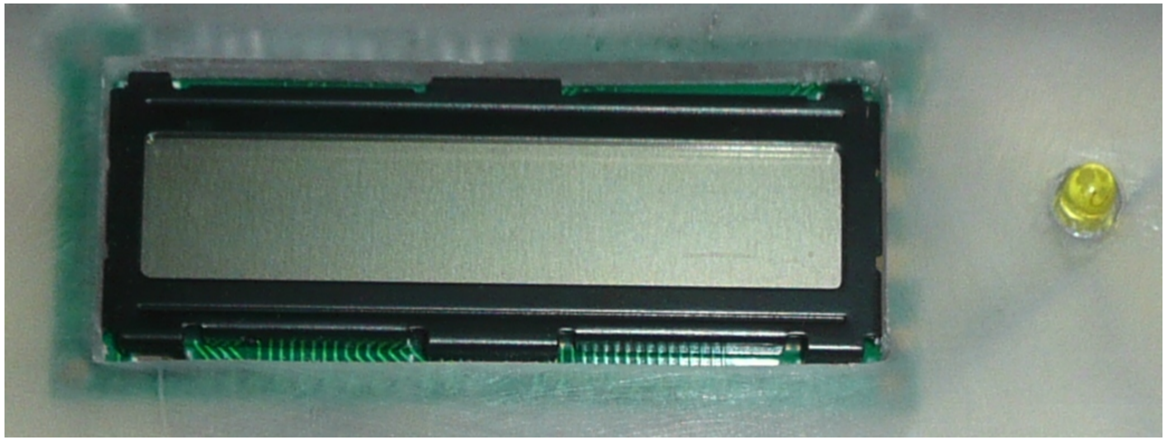
- Dot
- Dash
- Next letter/word

Push button interrupts caused dot or dash to be written to array. Next letter button initiated interpretation of array.

OUTPUTS:

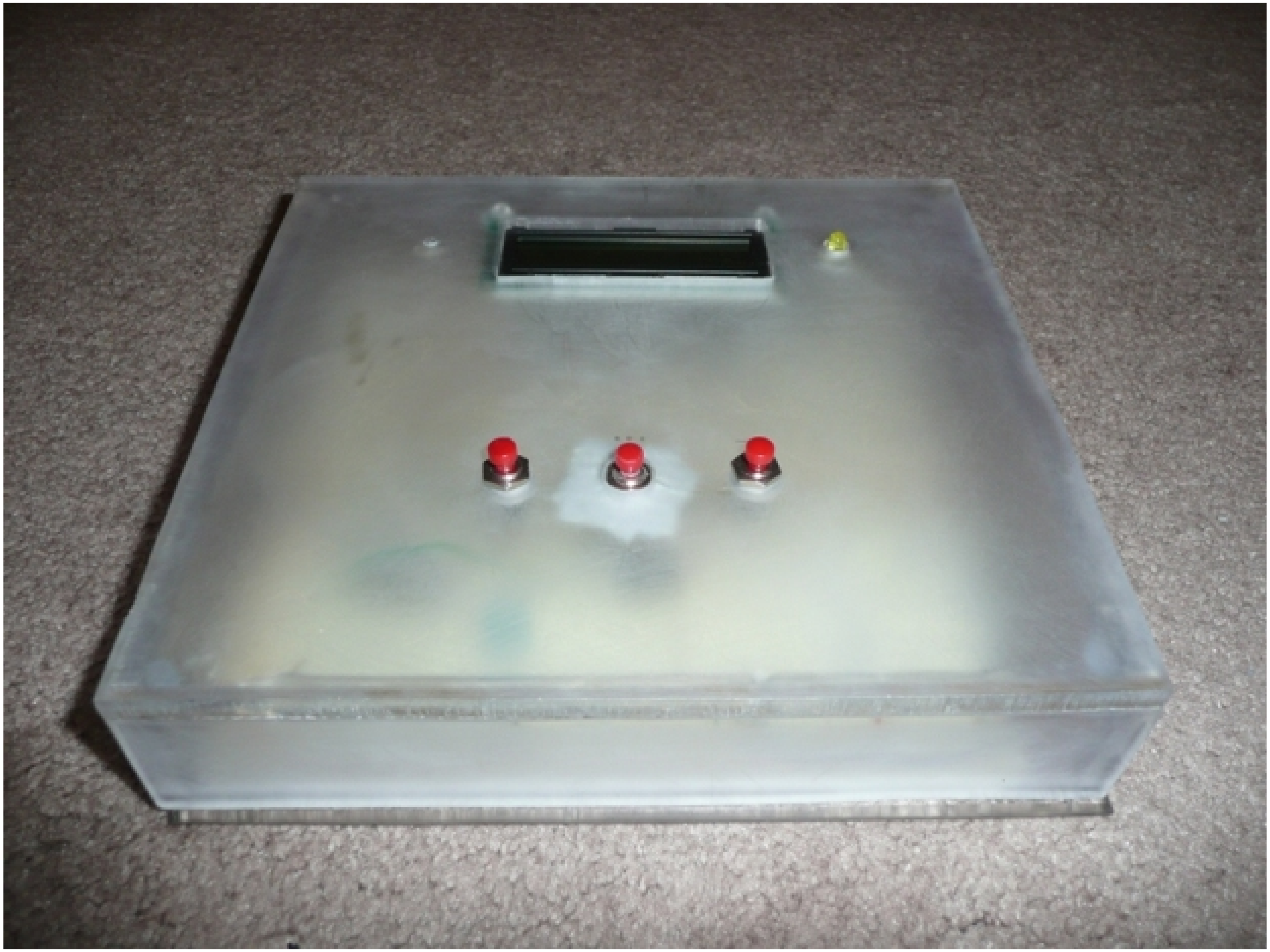
PIC interpreted array of Morse Code into letters and then wrote message onto LCD.

Fail letter LED light allows user to know when an invalid input was entered.



RESULT

- Originally it was planned to only have one button and have timers and interrupts keep track of button presses (1) and no button presses (0). This would be stored in an array and bit shifting would be used to break the message up. Since we learned this was not possible we switched to our current plan.
- Using three separate buttons made the message easier to input for the user. It also eliminated the need to bit shift an int.
- Project worked great after this change was made.



COMMENTS

- DON'T MESS WITH POINTERS IN C CODING!

- The MPLAB software and/or PIC can not bit shift more then 7 bits. (i.e. You can't bit shift into the first 8 bits of an int)

- Frosting (sanding) polycarbonate or acrylic looks really nice.

JOYSTICK CONTROLLED NERF GUN TURRET

Patrick Hudson
Lucas Weir
ME 430-02 Mechatronics
2/21/2010

PRIMARY OBJECTIVES

- To construct a turret consisting of a Nerf gun controlled by two motors
- To program a PIC to control the motors
- T control PIC inputs using a video game controller

INPUTS

- Five total inputs
  - 1 fire button input
  - 4 position inputs
- Inputs are run from a video game controller
- Position inputs
  - Control the location that the Nerf gun is pointed
- Fire inputs
  - Control the firing of the Nerf gun

OUTPUTS

- 7 total outputs
  - 6 motor controlled outputs (3 per h-bridge)
  - 1 fire control
- Outputs controlled by a Microchip PIC 18F4520
- Motor outputs
  - 2 lock outputs (1 per h-bridge)
    - Locks the motor connected to h-bridge from spinning
  - 2 direction outputs (1 per h-bridge)
    - Controls the direction which the motor spins
  - 2 PWM outputs (1 per h-bridge)
    - Duty cycle and frequency of PWM
- Fire output
  - Controls the firing of the gun

PIC 18F4520

- 4 motor position inputs cause interrupts upon signal
  - Interrupts unlock the h-bridge, controlling the spin direction of the motor and the duty cycle of the PWM signal
- 1 fire control input causes an interrupt upon signal
  - Interrupt controls the switch, flipping the switch upon signal
  - The switch completes the firing circuit within the gun, causing the gun to fire

MOTOR POSITION OUTPUTS

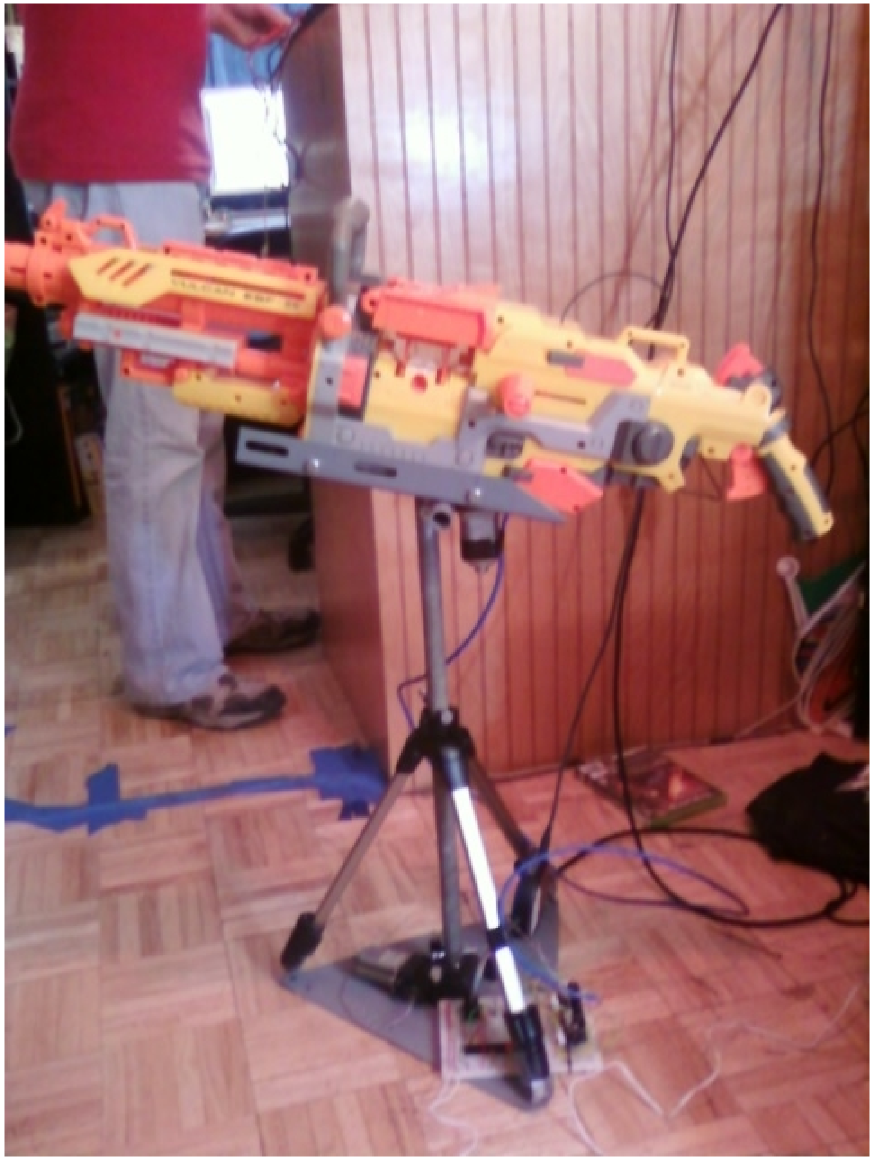
- Outputs connected to a h-bridge
- Lock output for h-bridge
  - Output controls the lock on the motor connected to the h-bridge
    - 1 – Lock the h-bridge
    - 0 – Unlock the h-bridge allowing the DC motor to spin
- Direction output for h-bridge
  - Output controls the direction that the DC motor will spin
    - 1 – Vertical motor upwards, horizontal motor left
    - 0 – Vertical motor downwards, horizontal motor right

MOTOR POSITION OUPTUTS

- PWM output for h-bridge
  - Two components, a duty cycle setting and a frequency setting
    - Duty cycle
      - Range 0-1023 (controls the percentage of the max voltage the motor will see)
    - Frequency
      - Range 1-255 (controls the frequency of the PWM cycle)
- Fire control output
  - Single output controlling a switch
    - 1 – switch is open, Nerf gun is not firing
    - 0 – switch is closed, Nerf gun is Firing

RESULTS

- Outputs worked as intended, except fire
  - Were not able to wire the output to a switch as intended
  - Used a button on breadboard to connect circuit instead
- Inputs did not work as intended
  - Coding issues caused code to be rewritten multiple times to reduce the complexity
  - Ultimately used batch style programming to control outputs



Picture 1 – Completed Turret

Figure 2 – Turret under Construction

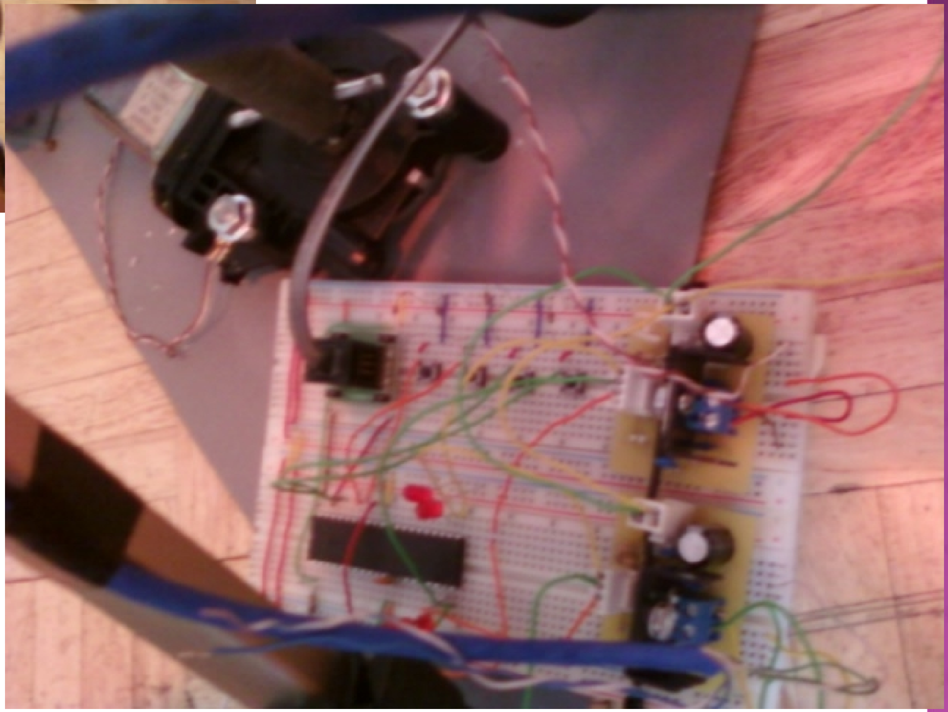
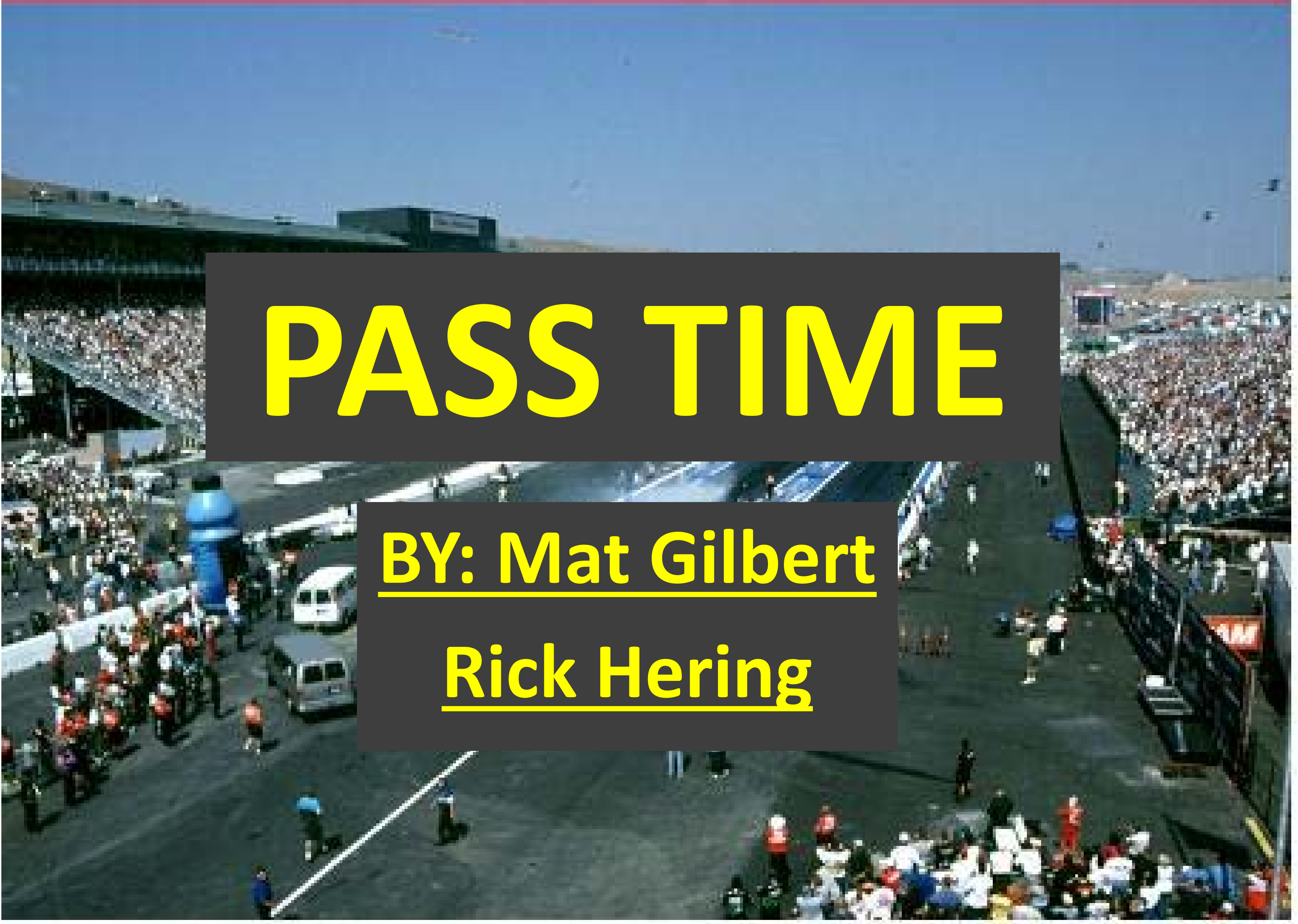


Figure 3 – Wired Bread Board

ADVICE

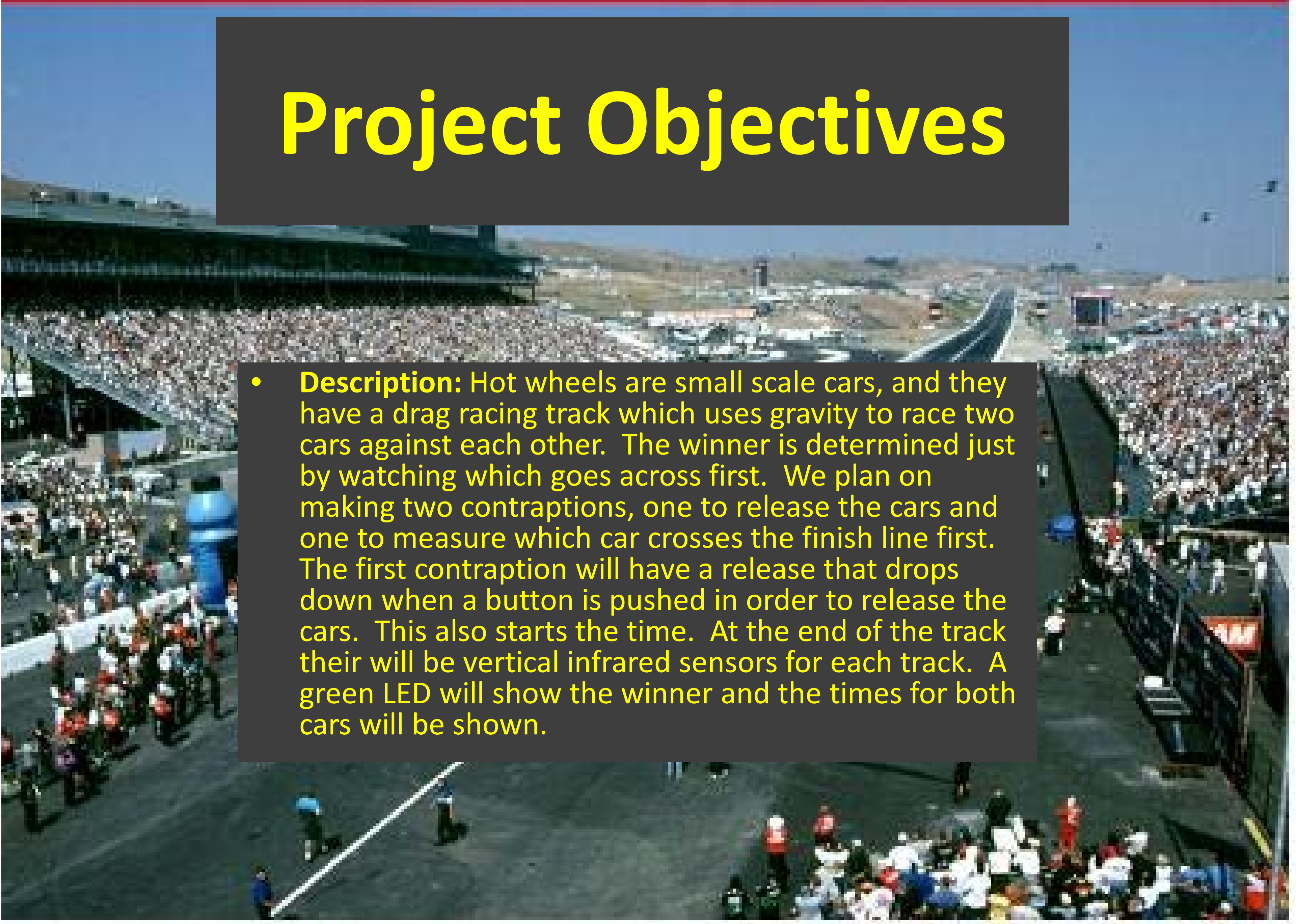
- Start coding early
- Check ranges on settings
- Also start building early
- Don’t assume the hardware cannot be broken
- Have backup components if possible, you will blow up components



PASS TIME

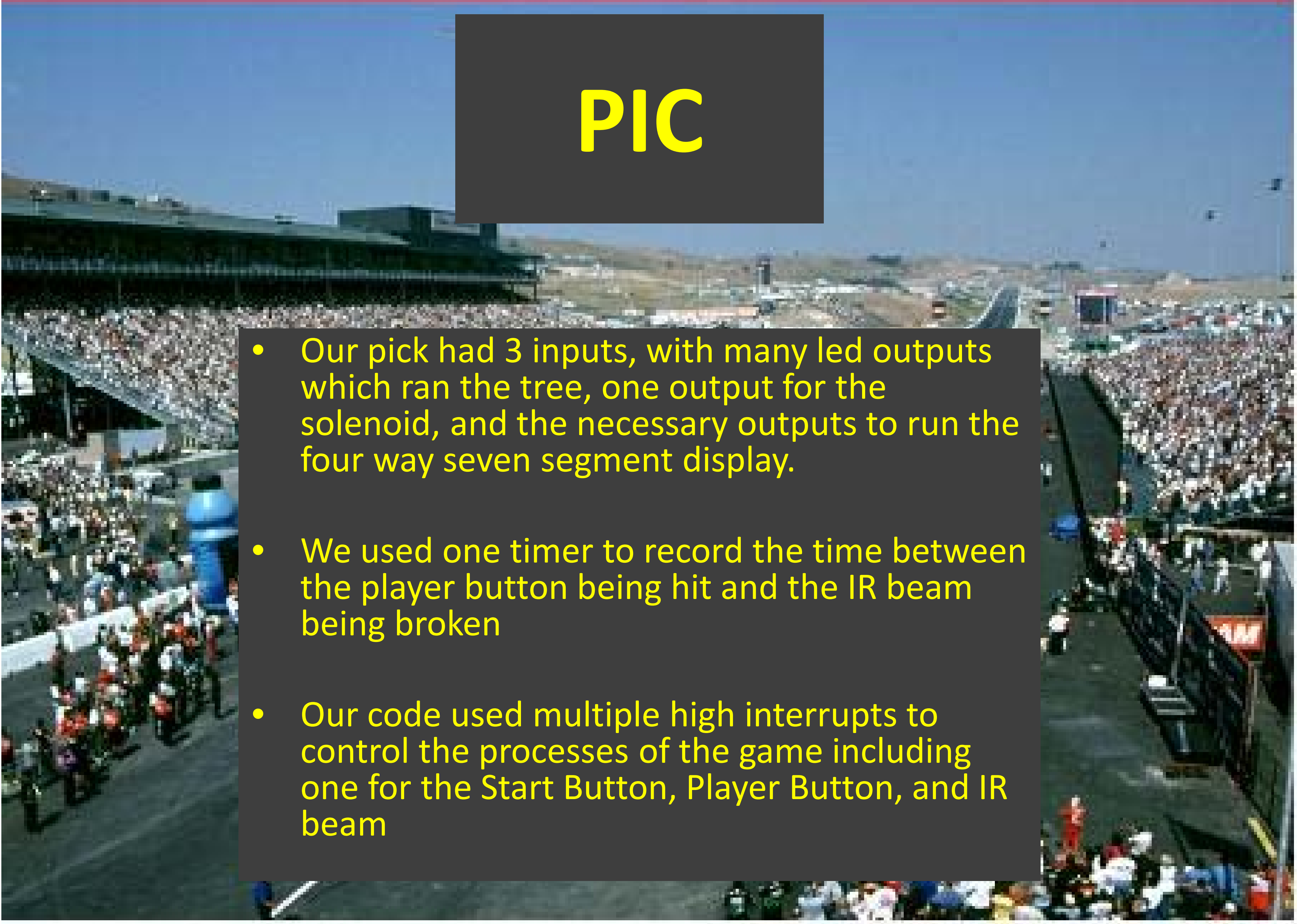
BY: Mat Gilbert

Rick Hering



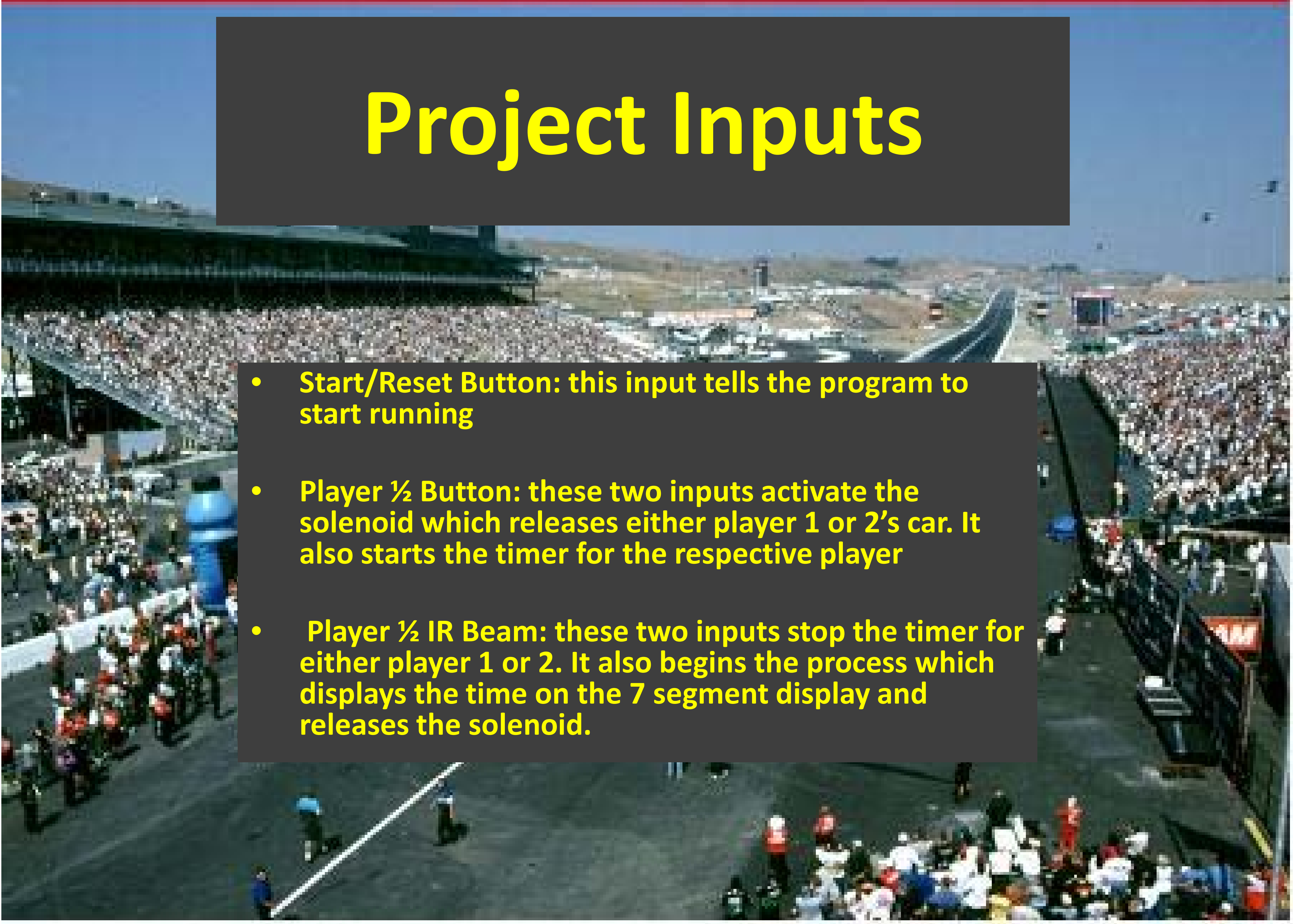
Project Objectives

- **Description:** Hot wheels are small scale cars, and they have a drag racing track which uses gravity to race two cars against each other. The winner is determined just by watching which goes across first. We plan on making two contraptions, one to release the cars and one to measure which car crosses the finish line first. The first contraption will have a release that drops down when a button is pushed in order to release the cars. This also starts the time. At the end of the track their will be vertical infrared sensors for each track. A green LED will show the winner and the times for both cars will be shown.



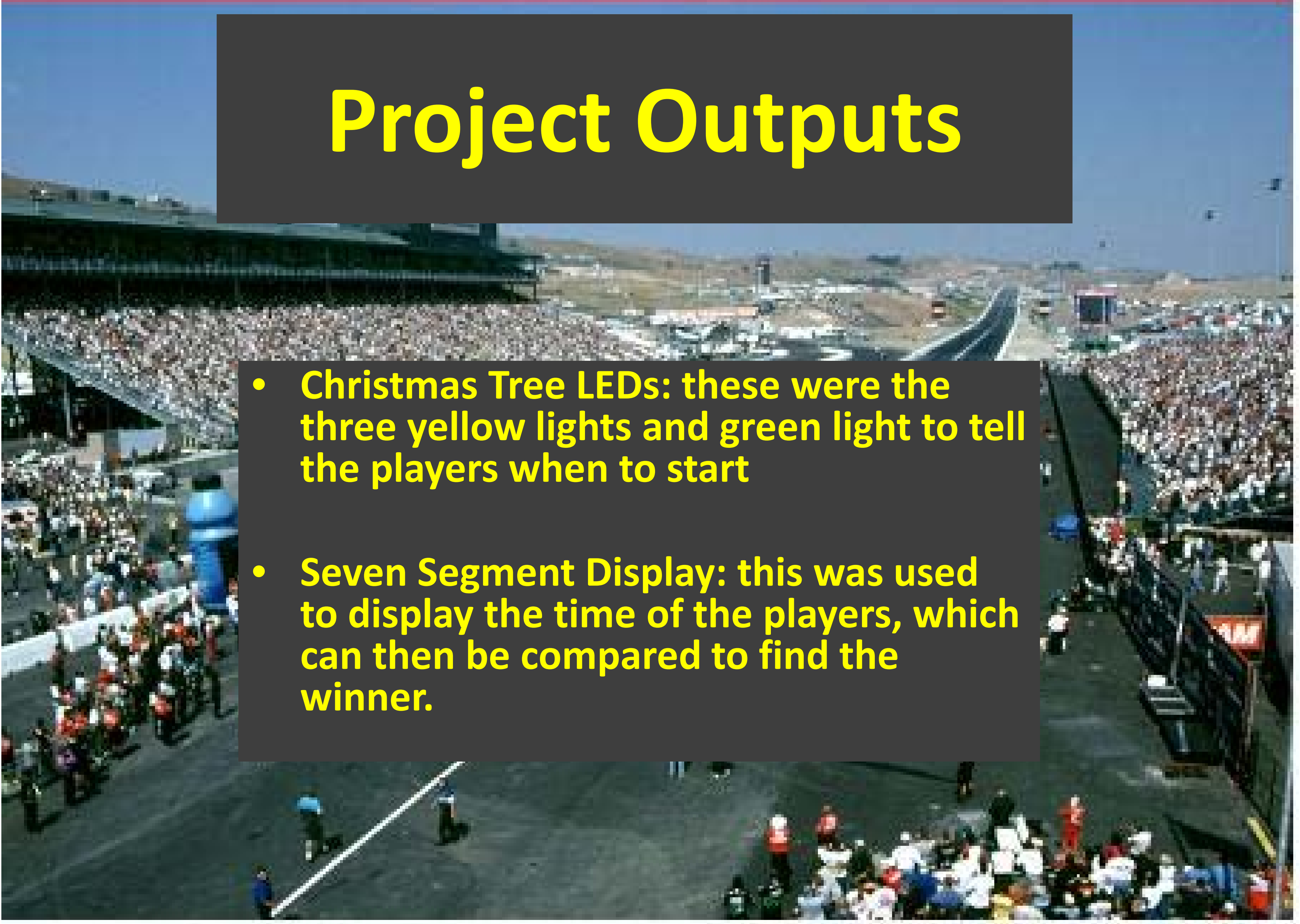
PIC

- Our pick had 3 inputs, with many led outputs which ran the tree, one output for the solenoid, and the necessary outputs to run the four way seven segment display.
- We used one timer to record the time between the player button being hit and the IR beam being broken
- Our code used multiple high interrupts to control the processes of the game including one for the Start Button, Player Button, and IR beam



Project Inputs

- Start/Reset Button: this input tells the program to start running
- Player ½ Button: these two inputs activate the solenoid which releases either player 1 or 2's car. It also starts the timer for the respective player
- Player ½ IR Beam: these two inputs stop the timer for either player 1 or 2. It also begins the process which displays the time on the 7 segment display and releases the solenoid.



Project Outputs

- Christmas Tree LEDs: these were the three yellow lights and green light to tell the players when to start

- Seven Segment Display: this was used to display the time of the players, which can then be compared to find the winner.

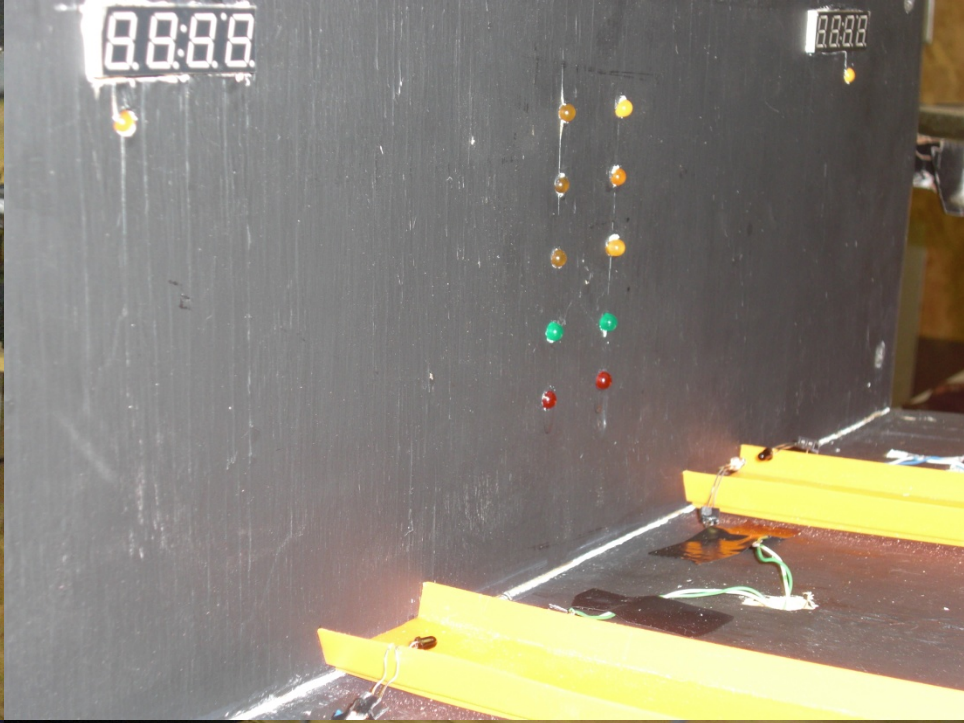
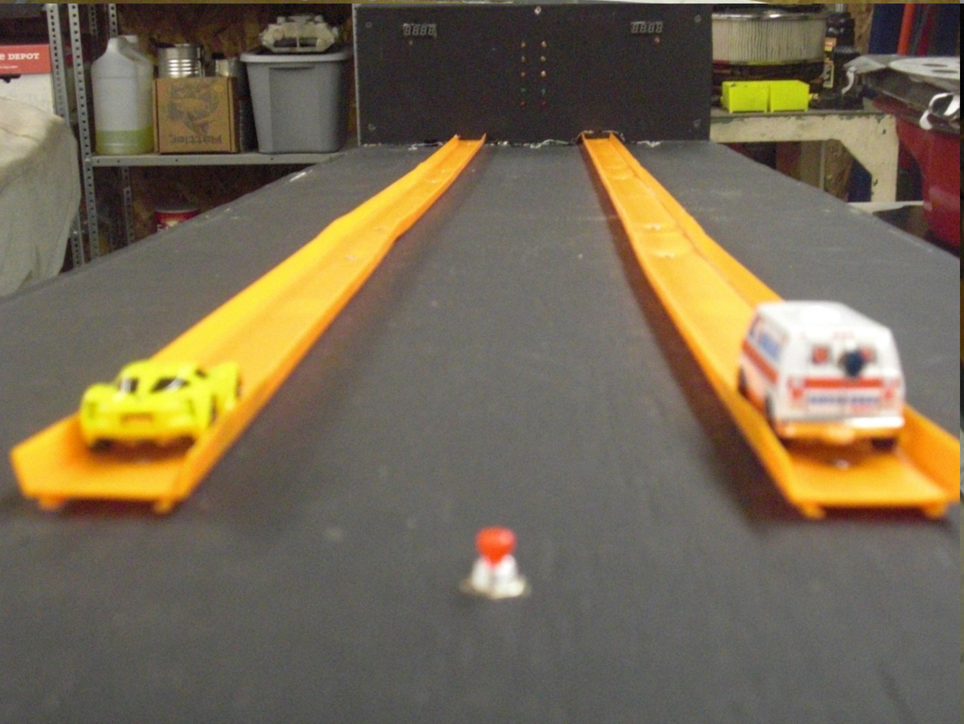
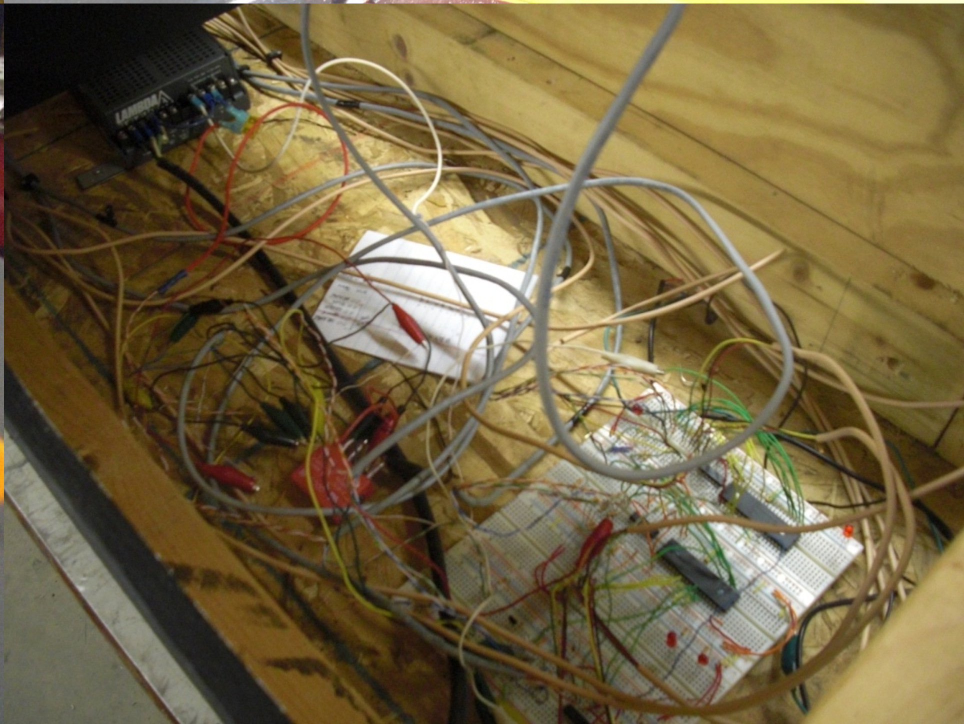
Results

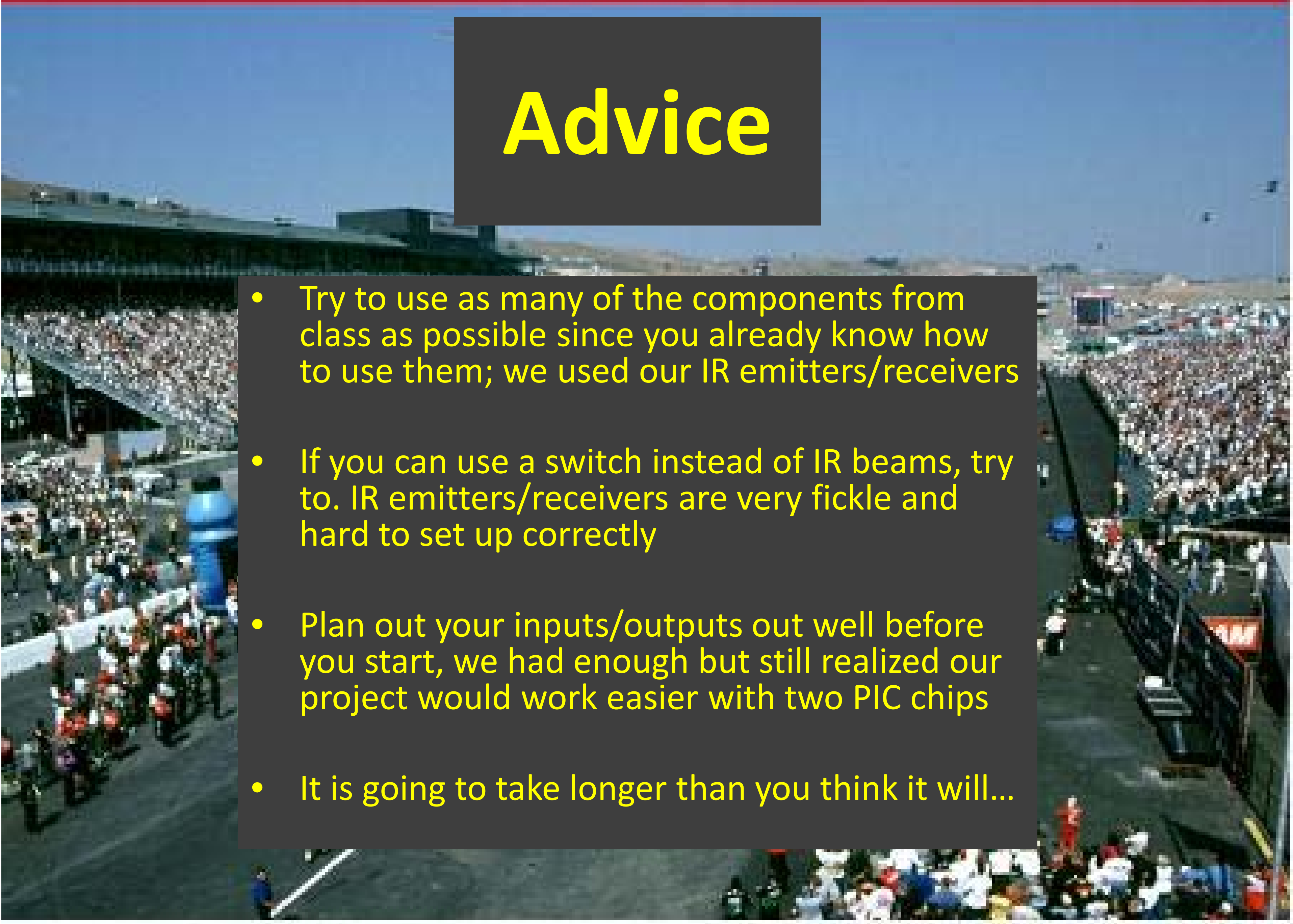
- Our Start Button worked to start the program and the LED countdown on the Christmas Tree correctly
- Our Player Buttons also worked correctly to start the timers and release the cars
- Our seven segment displays worked to read out the times of the runs

Pictures!

VAN HALEN
PERFORMANCE
FLYWHEELS &
FLEXPLATES

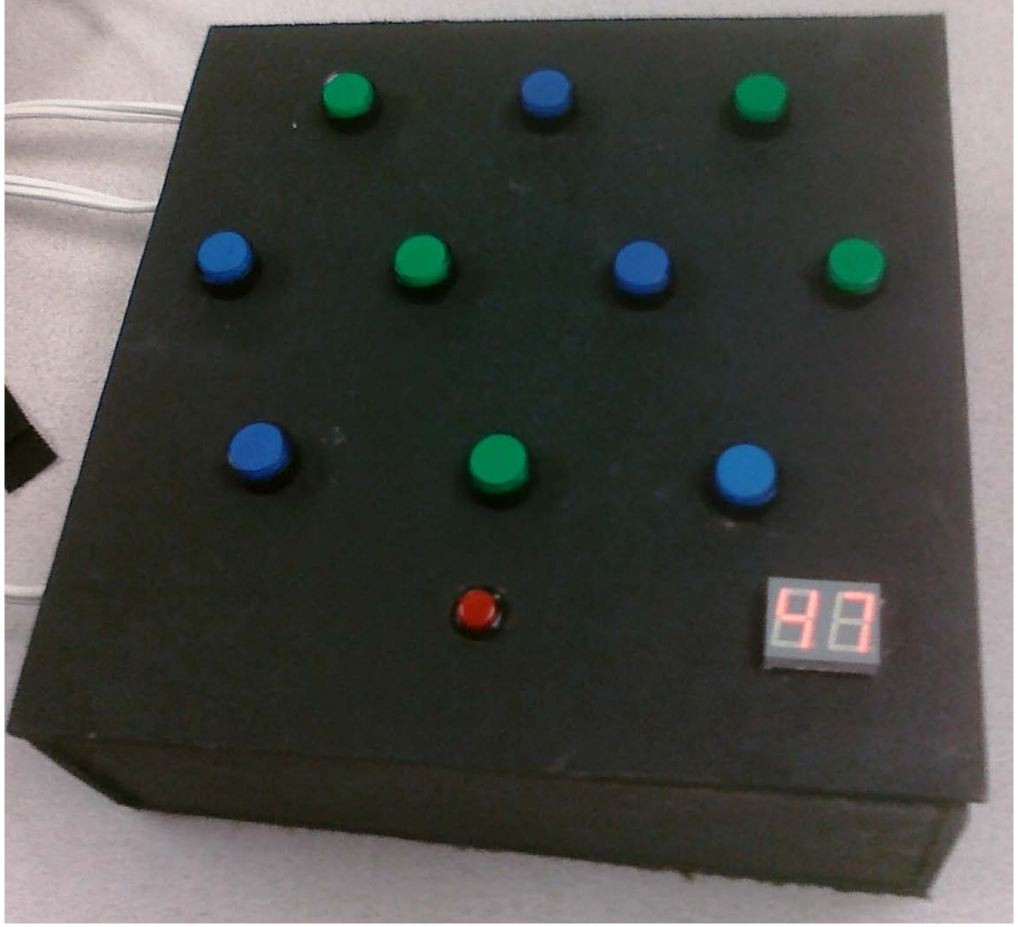
PERFORMANCE FLYWHEELS & FLEXPLATES



DEPOT

LAMBDA



Advice

- Try to use as many of the components from class as possible since you already know how to use them; we used our IR emitters/receivers
- If you can use a switch instead of IR beams, try to. IR emitters/receivers are very fickle and hard to set up correctly
- Plan out your inputs/outputs out well before you start, we had enough but still realized our project would work easier with two PIC chips
- It is going to take longer than you think it will…



Press the Light Game

Mechatronics - ME430-02

Ryan Cummings

Cassie Grammel

Objective

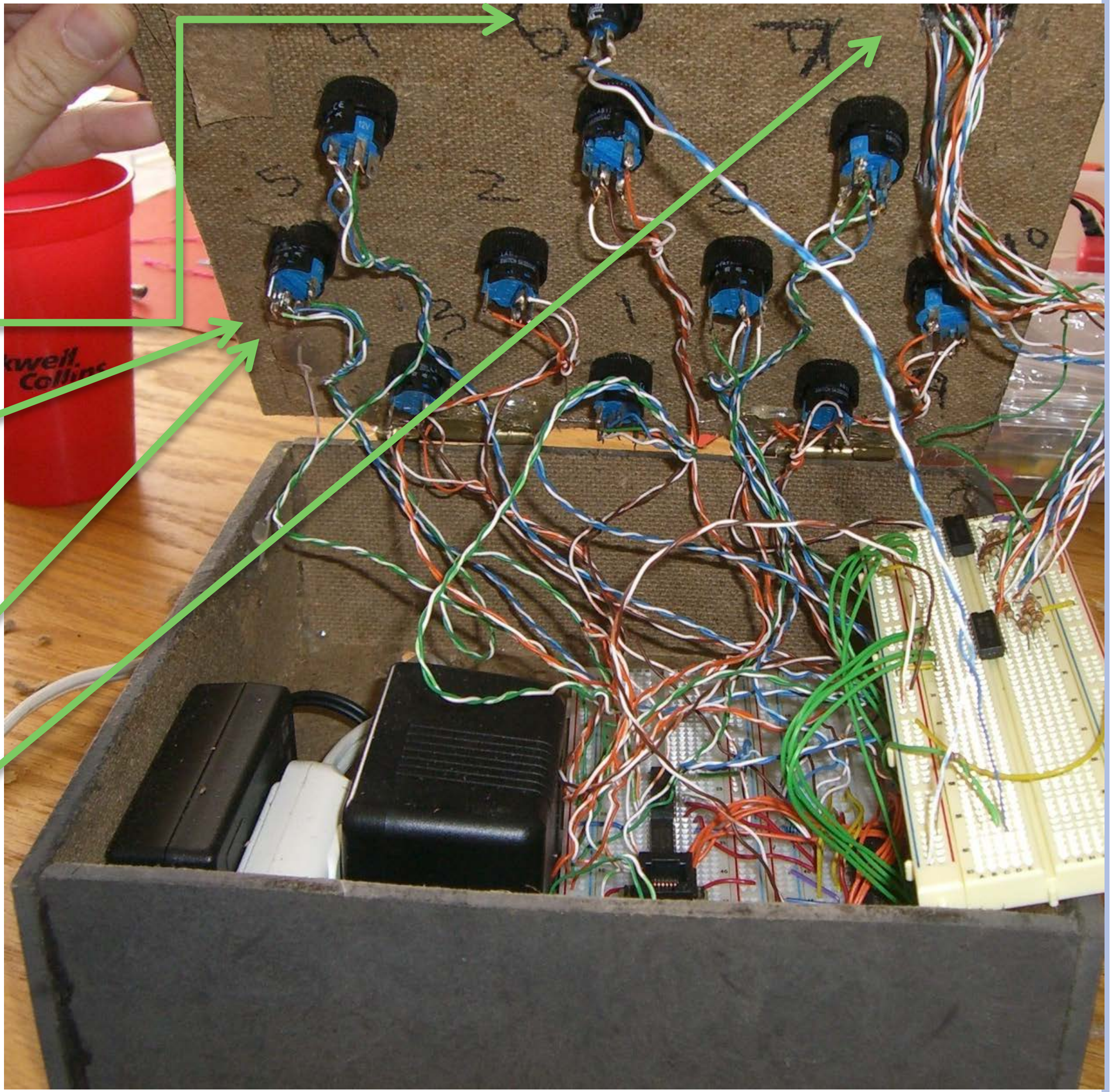
- Make a game where buttons light up and by pressing a lit button you receive points.
- Display the score on a 7 segment display
- Make the game look good

Inputs

- Start button
- Lit button presses

Outputs

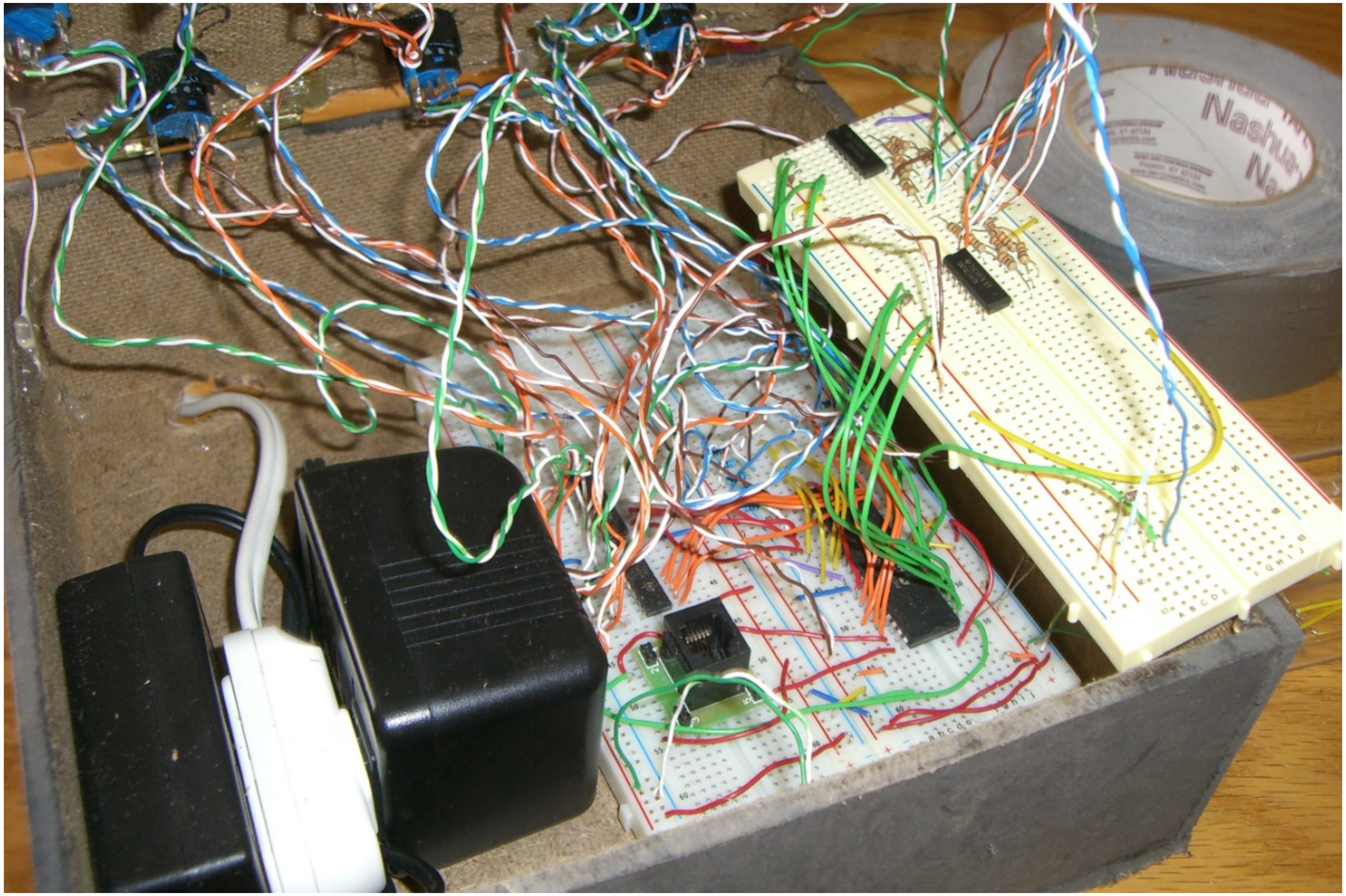
- Light up button
- Score



PIC Programming

- Used a timer with interrupts to generate a random number and light a random button 1-10
- Used a timer for the gamelength
- Input buttons were used in the while loop with our recentButtonState code
- Score was output to 2 7-segment displays

Internal Wiring



Conclusions

- Project worked to our specifications
- We are able to play one game, then press start and play another
- Kept score successfully
- Game is fun and challenging to play

Lessons Learned

- Make the box/housing for your project last
- Make sure you have the right power supply (5v/12v) connected where it should be
- If things don’t work feel the chips and unplug power
- A bigger housing is easier – don’t go for small or fancy
- Order parts early and don’t forget Darlington's

Raccoon Pellet Dispenser

Winter 2009/2010

Mechatronics – Olson

By: Derek Davis and Cody Fogwell

Objective

- In the wild, raccoons are being fed pellets to cure them of ringworm.
- The problem is that one raccoon will eat all of the pellets in one sitting.
- Our device dispenses one pellet, waits a specified time, and then dispenses a second pellet and so on until the specified number of pellets have been dispensed. After that, the system goes into a hibernate period for a specified time.

Inputs

- User specified length between dispensing of each pellet
- User specified number of pellets
- User specified hibernation period after all pellets have been dispensed
- Whether the pellet has been taken from the tray or not

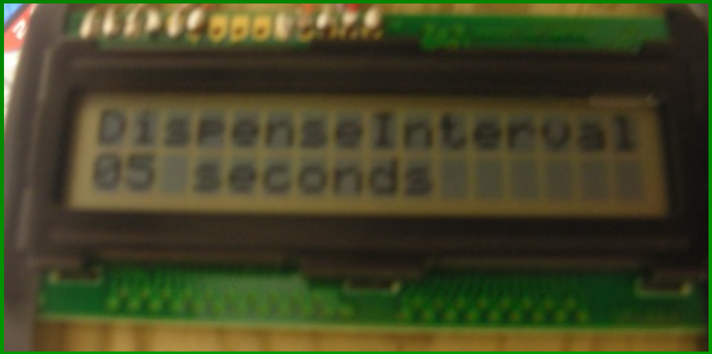
Outputs

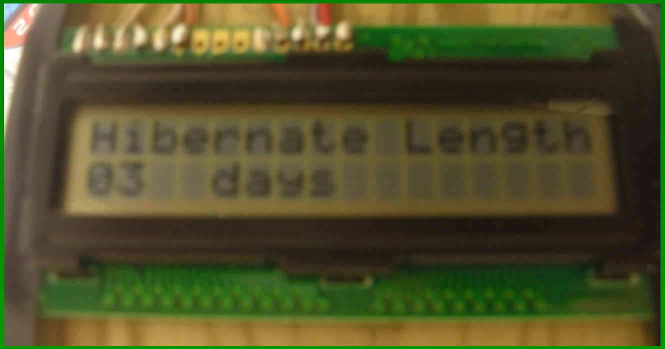
- Display on LCD screen of dispense time interval
- Display on LCD screen of number baits per cycle
- Display on LCD of hibernation time length
- Actual dispensing of the bait pellets

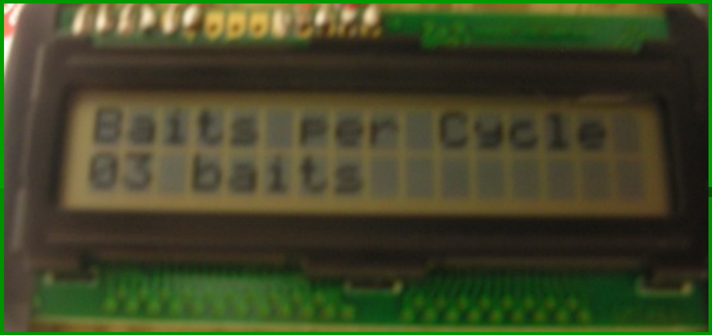
Using the PIC

- The release of a button triggered an interrupt
- Instead of a timer, a while loop with a delay function was used, started by the interrupt
- Other buttons changed what the LCD screen displayed

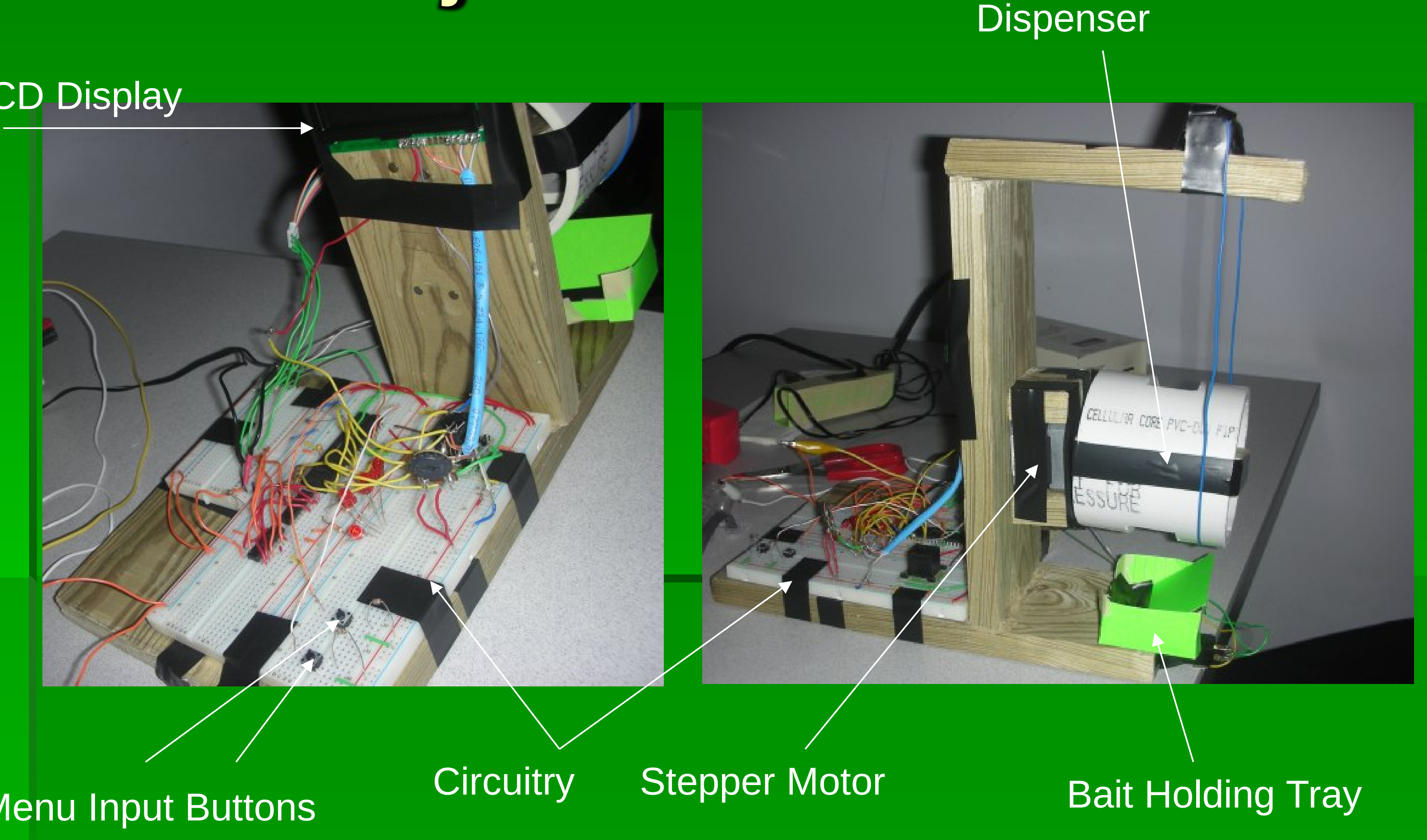
LCD and Menu Options







Final Project



Results

- Our project worked as planned!
- The only slight issue was that the stepper motor was a bit underpowered, causing it to have problems turning at times

Advice

- Avoid intricate machining
- Plan ahead, do not wait to start working until the end
- Leave plenty of time to re-order parts if necessary
- Avoid parts that require high precision to work together

Search and Rescue

Fred Freers
Dan Tomasic

ME430-01
2/16/2010

Project Goals

- Build a robot that will be able to:
  - Locate the “survivor” (highest heat source) [Simulate search and rescue in the snow]
  - Drive towards the survivor while tracking it
  - When the robot bumps into the survivor, a buzzer goes off alerting others that the survivor has been found

Process

Inputs

- Start
- Temperature
- Bumper

PIC

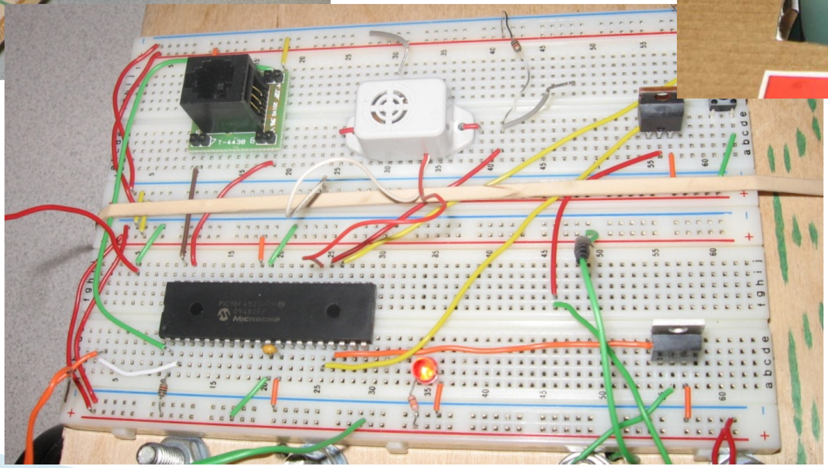
- Waits for start button to be pressed, then starts program
- Reads IR guns and decides where the highest heat source is
- PWM used to drive motors towards the heat source
- Reads bumper and sets off bumper

Outputs

- Motors
- Buzzer

Results

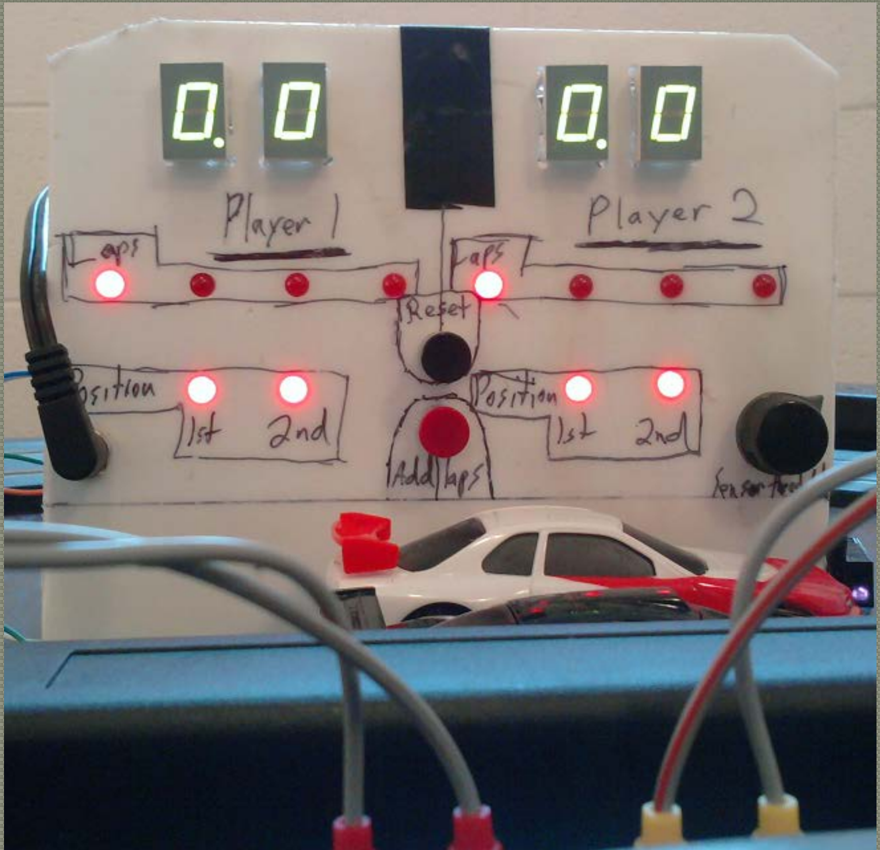
- The robot did successfully find the survivor!



Advice

- Decide on a project that will not involve many ordered parts (Browse the machine shop before you buy)
- As with any project, start early to prevent other projects from getting in the way
- Do something that is fun and engaging, but without too much time input

Slot Car Racing Scoreboard

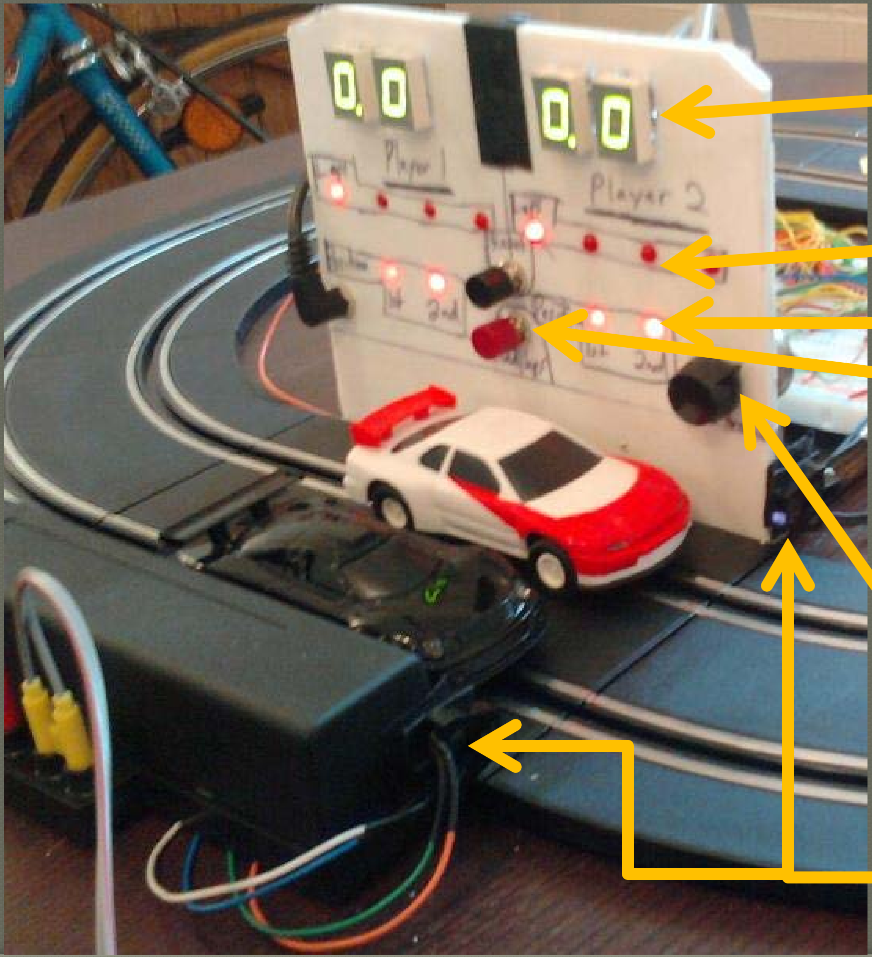


Kenneth Schurer
Matthew Boron
ME430 Mechatronics

Project Objectives

- Allow players to select number of laps
- Provide a countdown for starting the race
- Automatically score second place to any player who jumps the gun
- Count laps and keep track of position during race
- Display last lap time for each racer

Layout of the scoreboard and sensors



Outputs

Lap time and countdown displays

Lap counters

1st/2nd place displays

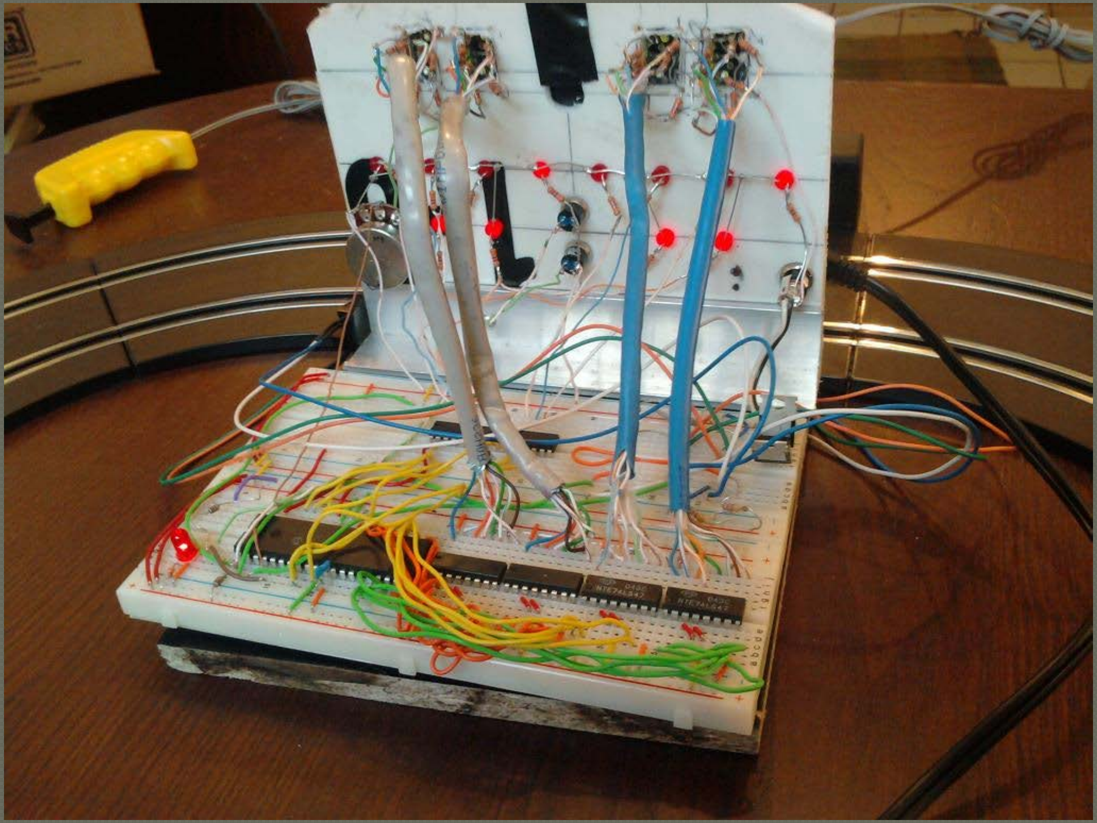
Inputs

Reset and lap set switches

Sensor threshold adjustment

Reflective sensors

Breadboard and wiring



PIC was used to drive four 74LS47 seven segment display drivers and two Darlingtons that power the LEDs.

Inputs were analog from the two car sensors and threshold control and digital for the lap count input. Reset was a switch that brings pin 1 to 0 V.

Timers used interrupts, everything else was in loops.

Three parts to the code

- Lap programming
- Countdown
- Race

Lap programming

- Player presses Add Laps switch until desired lap count is displayed (1-4 laps available)
- If pressed more than four times, lap count resets to 1
- When nothing is pressed for five seconds, countdown begins

Countdown

- Displays a countdown from 9 to 0 on all displays

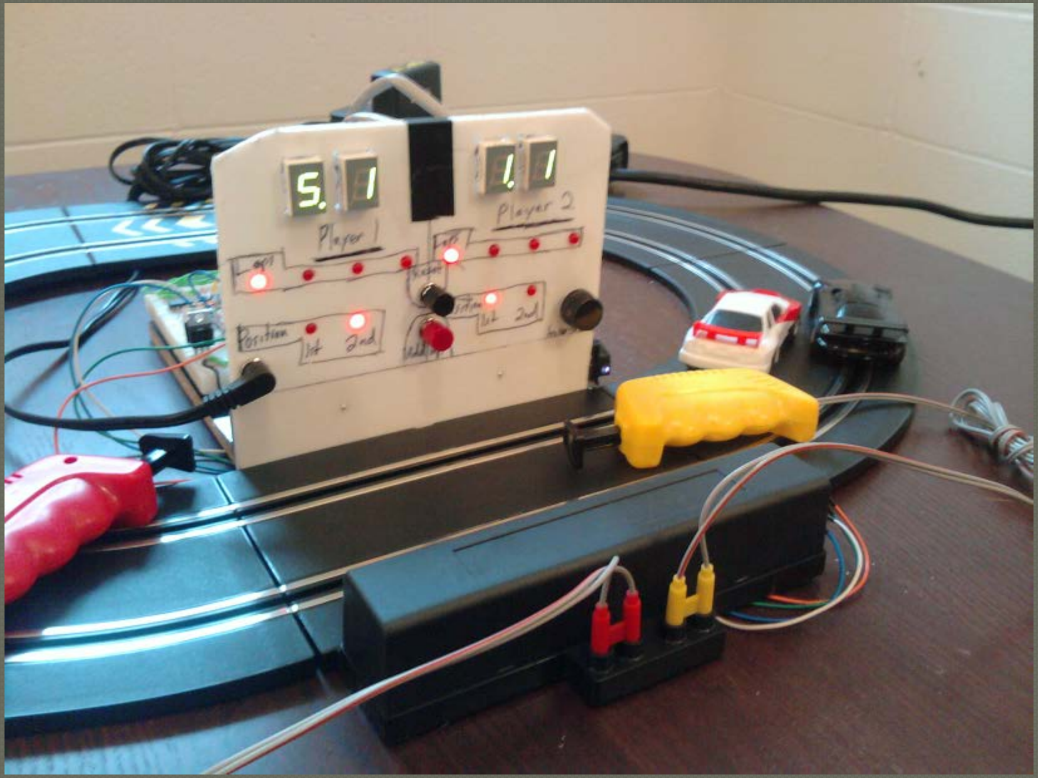
- If a player jumps the gun, he is locked at 2nd place.

Race

- When racers cross the line:
  - Lap timer displays time then resets
  - Lap counter increments
  - If racer’s lap number is greater than the other’s and there were no cheaters on the start, he is scored 1st place
  - If lap number reaches lap limit, displays will lock at the last value

Results

- Everything worked as planned!



Lessons learned

- Always code and test in small parts
- Reflective sensors have a very short range and can be pointed at each other without causing false readings
- Always use a threshold potentiometer with light-based sensors to account for light differences
- Electric guitarists can save money on power supplies by using effect pedal supplies and a $1 regulator chip
- Take advantage of common nodes when wiring so you can run only one wire from the board to LEDs to minimize mess

Teacher Target Practice

Colleen Elliott & Vince Forcellini

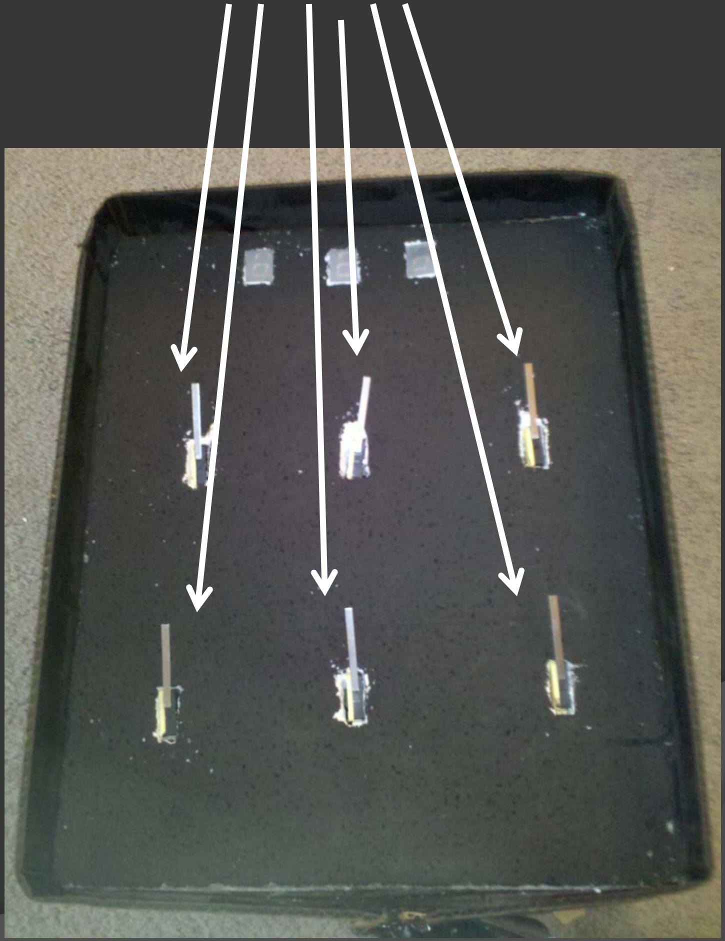
2/22/10

Game Objectives

- Use a Nerf gun and darts to hit the targets
- The favorite teachers get more points
- Repeat until score of 500 or more is reached
- Buzzer sounds and game resets

Game Inputs

- Dart hits target and closes Switch



Game Outputs

- Score displayed on 7-digit display
- Buzzer signals end of game

Used PIC for…

- Delays
- If loops to add up score
- Change values on 7-segment display

Results

- Through diligent work and careful planning, our project was finished 2 days early
- Project was successful and game was demonstrated as planned

Advice for Future Students

- Order parts early in case they don’t work
- Work on project through out quarter
- Pick a partner you can work with all 10 weeks
- Be sure to test more than a day ahead
- Keep the project simple because it will get more complicated

The TrueSpeed 8000

Keenan Long
Dan Prochno
Winter 09/10

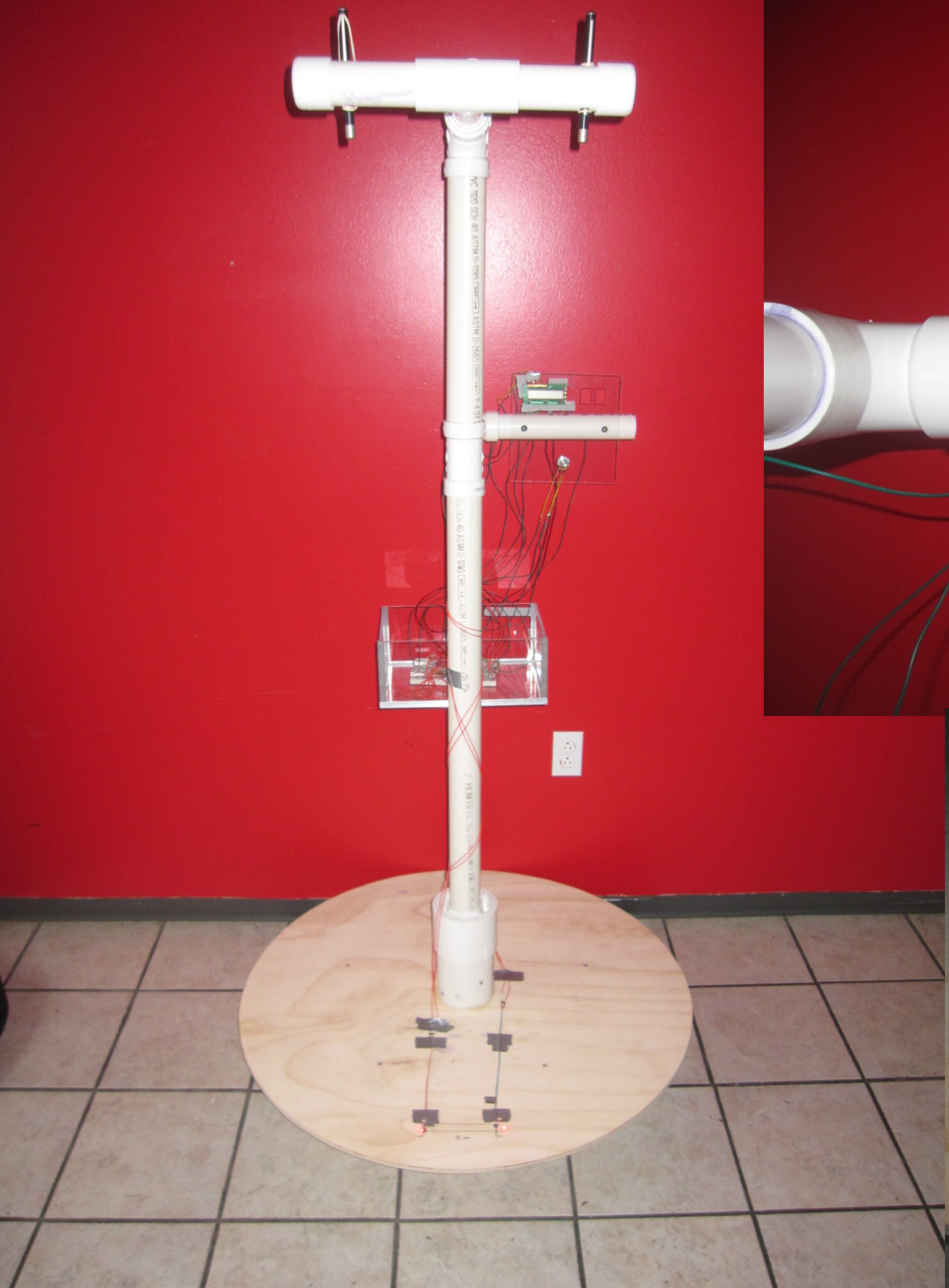
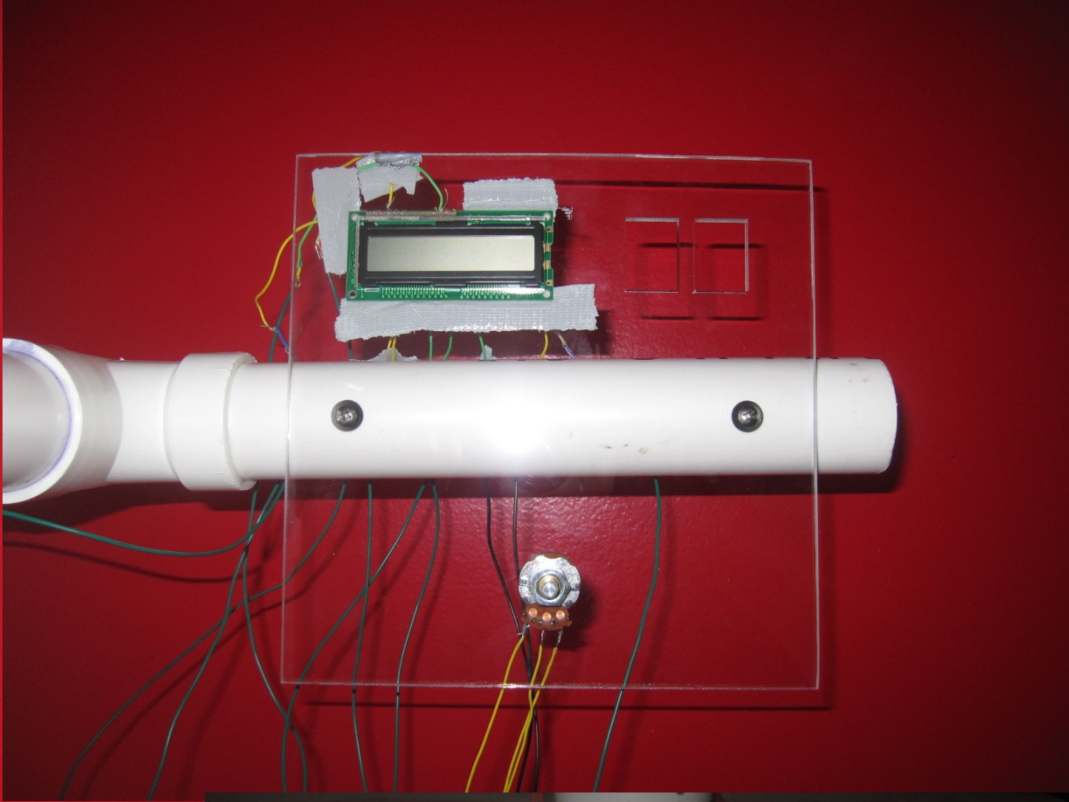
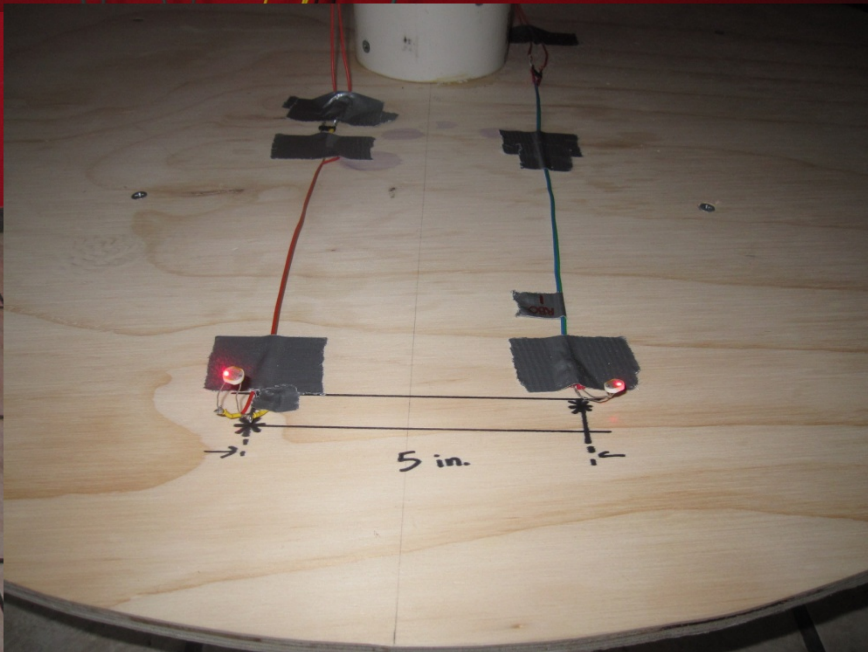
Baseball Bat Swing Speed Analyzer

INPUTS

- User dials in theoretical pitch velocity
  - Poteniometer
- User swings bat through simulated hitting zone
  - Two vertical laser beams aimed at CdS cells
  - 1st beam break starts interrupt and timer
  - 2nd beam break reads timer

OUTPUTS

- LED reads out swing speed in mph
  - Convert timer ticks to seconds and solve for velocity
- Calculates flight of batted ball under perfect conditions
  - uses CoLM in the interrupt and converts to feet
- LCD screen displays…using sprintf command
  - “Pitch speed was xx mph”
  - “Distance of ball was xxx ft”




5 in.

Results

- Hardware worked correctly
  - LEDs displayed swing speed
  - LCD screen printed message with correct variables
- Code most likely worked correctly
  - The math was checked and double checked
- Our code may have over ran the memory of the PIC
  - Had to use floats in order to get accurate calculations of ball distance
  - Advice = avoid long calculations with many variables

Viscosity Tester

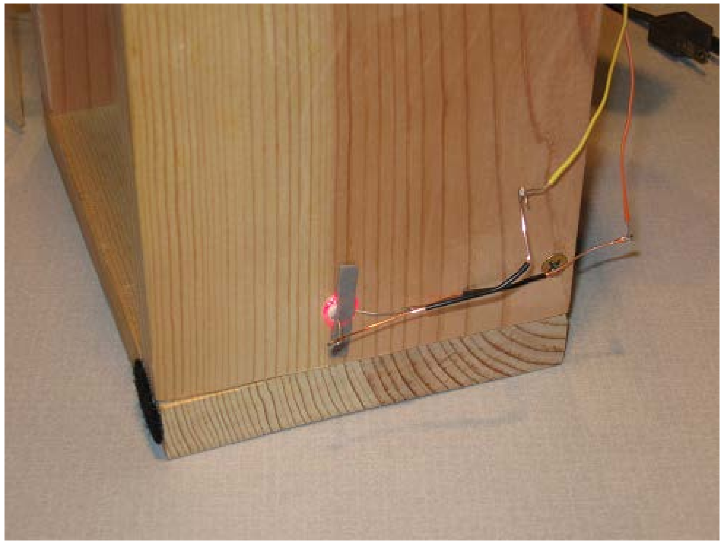
Seth Carter
Dan Stanage
ME-430

Objective

- To determine the viscosity of a fluid and report the results to the user.

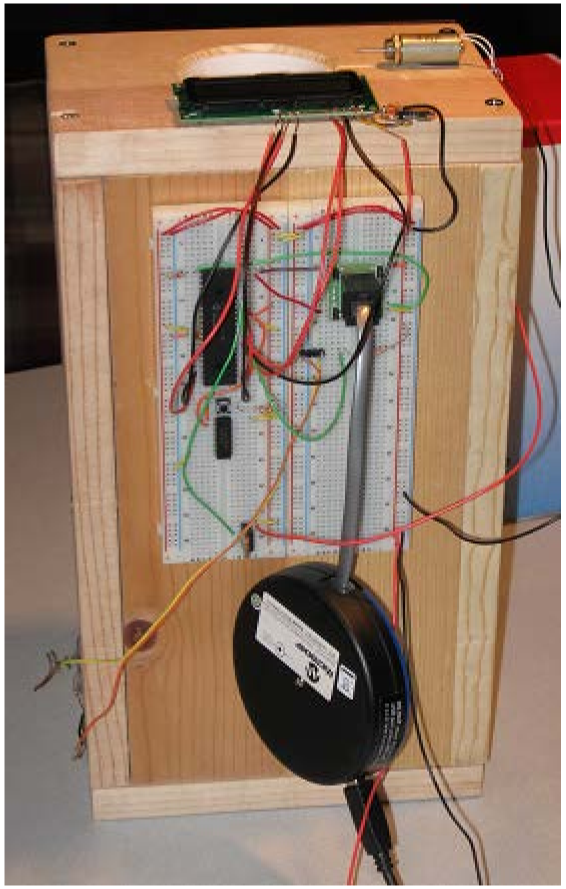
Inputs

- Button
  - Used to start timer and activate the solenoid
- Laser Detector (CdS Sensor)
  - Detects when the metal ball has reached the bottom of the graduated cylinder



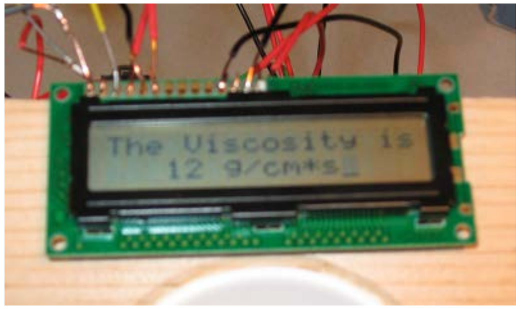
PIC Controls

- Once the signal from the button was received, the PIC sent a signal for the solenoid to activate and started a timer.
- Once the metal ball broke the laser, the CdS sensor signaled the PIC to stop the timer and to print the viscosity to the LCD.



Outputs

- Solenoid
  - When signaled, the solenoid pushes a metal ball down the funnel which leads directly to a graduated cylinder.
- LCD Panel
  - The LCD always displays “The Viscosity is” on line 1 and once the ball trips the CdS sensor, the viscosity value displays on line 2.



Results

- Initial Testing
  - Same viscosity value
    - Misunderstanding of interrupts and timers
  - Solenoid activates before button is pressed
    - No snubber diodes or decoupling capacitors
- Final Results
  - Success! The solenoid will activate and push the ball into the graduated cylinder and once the ball breaks the laser, a viscosity value will display.

Future Advice

- Start early and plan ahead.
- Order parts early, some parts take a while to get.
- Count the number of processes your project will perform and figure in 2-5 hours for each process.

Whack-a-LED 3000

Steven Mardis
Zach Bierhaus

- Description
  - Just like whack-a-mole except:
    - LEDs and CdS cells (light sensors) replace moles
    - Laser pointer replaces hammer
- Output
  - Score
    - Number of hits divided by 6 (to make output one-digit, so it can be displayed on a single 7-segment display)
- Objective: Have a fun, functional game to play

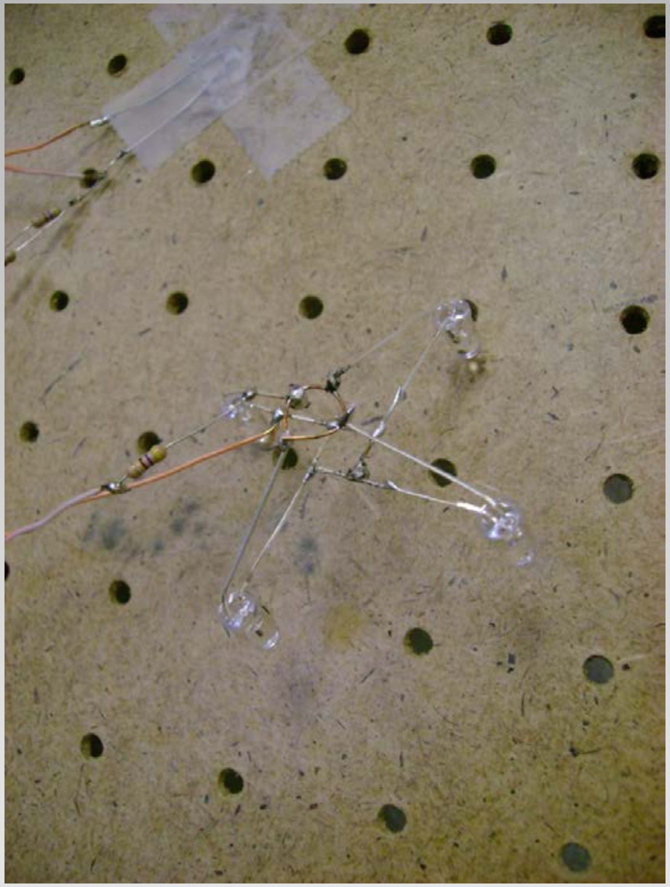
Plan

Programming

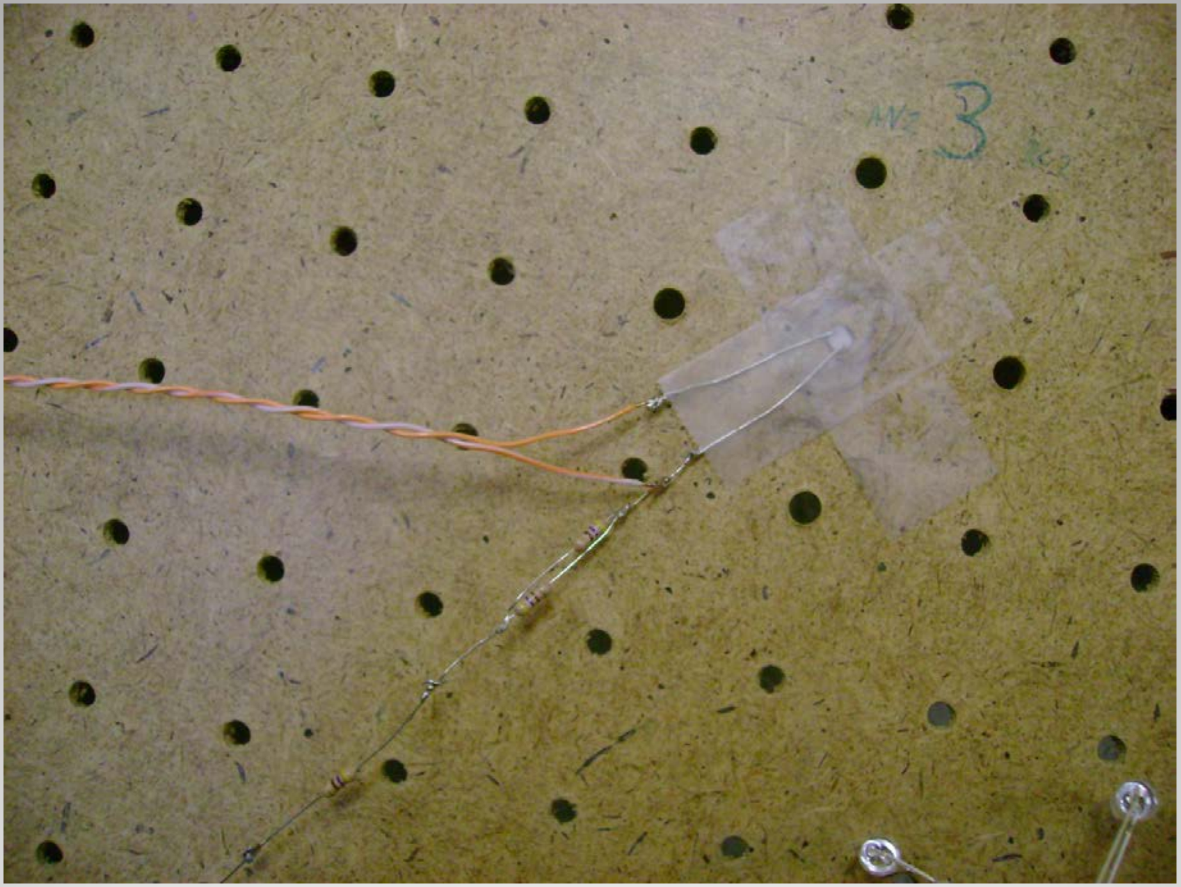
- A timer interrupt was used to end the game
- In the main while loop:
  - A random number from 0-11 was chosen.
  - A switch-case statement chose the lights
  - The lights go on. While the laser isn't on the CdS cell, nothing is done.
  - When the laser hits, the while loop ends, the score is incremented, and the program returns to the main while loop.

- Worked flawlessly on the debugger
- When programmed and removed from the computer, had shorting/groundage problems

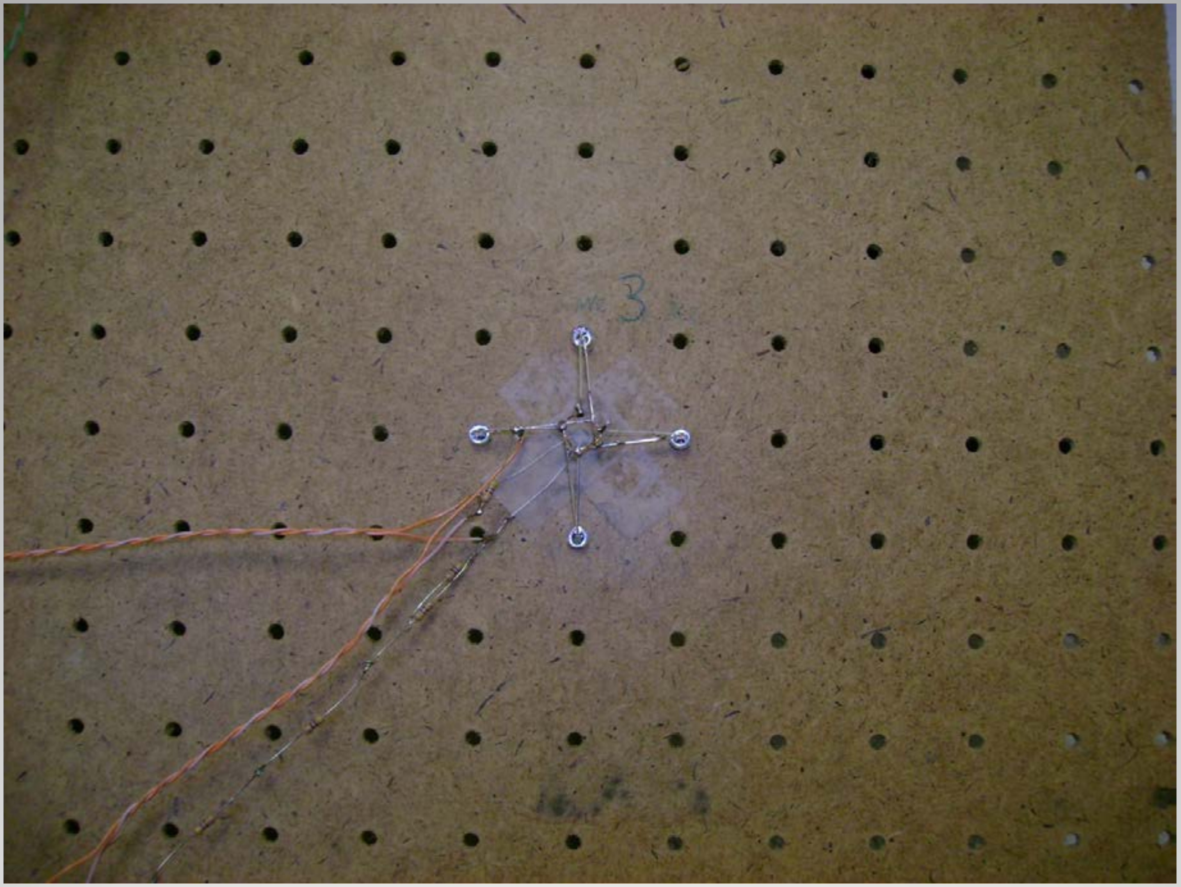
Results



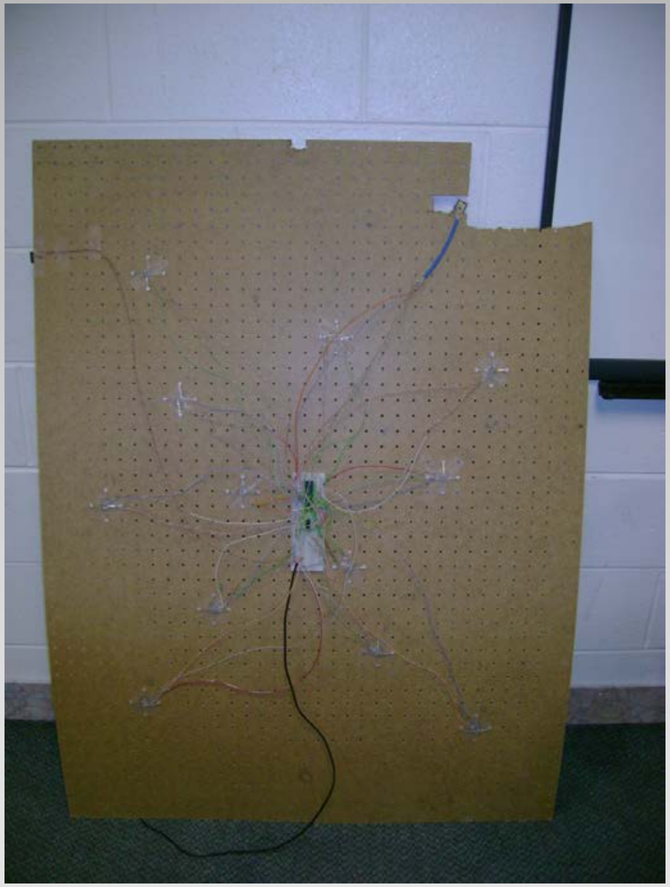
LED Configuration



CdS Sensor Configuration



Both Put Together



Back of Board

Project In Action