Digital-Pong

School of Engineering and Applied Sciences
Harvard University
33 Oxford Street
Cambridge MA.

The Ultimate Automated Beer Pong Table

By: Student A, Student B and Student C

Class: ES50
Instructor: Marko Loncar & Christopher Lombardo

December 2014

ABSTRACT

The Digital-Pong table aims to create a fun and unique beer pong experience that minimizes clean-up, digitally keeps score, re-racks cups on behalf of inebriated people and dispenses the exact same amount of beer for each player in the spirit of fairness. We set out to create a digital beer pong table which would count score and display cups which were hit by changing an LED ring surrounding the cups from green to red. Not only did we build the circuits and write the code ourselves, but we also built the table ourselves. Although we had envisioned many more features that did not get implemented, we worked very hard and the experience was a rigorous and educational application of the material we learned through the semester.

INTRODUCTION

All around the world, beer pong is a time-honored tradition that has entertained young people for centuries. But playing the game comes with some mild irritations, specifically having to clean up a particularly messy game and fights that can range from arguments over keeping score to how much beer each person is consuming (aka if people are copping out). We chose this project because we wanted to build something that was fun and unique, but also helpful. We originally envisioned a new Digital-Pong table with hopes to avoid these conflicts and keep the game clean for everyone involved. Instead of the classic solo cups full of liquid (that inevitably someone will spill), there will be similarly sized holes that detect when a ping-pong ball goes through. A ring of LEDs around each cup will help players identify which cups are still in play - green if it is, red if you already scored in that "cup" and a digital display will help you keep track of your score. Of course, we can't forget the other integral part of the beer pong experience, which is ensuring that players are consuming a safe amount of liquid when the other team scores. Instead of drinking from a cup that you just had to fish a dirty ping pong ball out of, a convenient drink dispenser on both sides of the table will dispense 3oz of a liquid of your choosing in a clean cup. This way the amount a player drinks is consistent and fair. This also minimizes chances of spilling since there will only ever be at most two cups with liquid on the table. One other bonus is the added empowerment a team will have with regards to "re-racking" - the strategy of rearranging the remaining cups. Instead of fights breaking out over whether a team properly executed a "diamond" arrangement, players will be able to rearrange their cups using a convenient display of illuminated push buttons that represent each hole/cup and the table will automatically reconfigure the LED rings around to each hole to reflect the new arrangement.

DESIGN

To accurately reflect the traditional beer pong experience, we sought to maintain the classic 10-a-side game; we needed to decide how each cup unit would detect whether a ball had landed. The first option we considered was using pressure sensors on the table surface that would detect the additional weight of a ping-pong ball should it land in a cup. However, we were concerned that without liquid in the cups to hold them down, any sudden movement or even touch by a poorly tossed ping pong ball might knock over the practically weightless cups and require frequent rearrangement. In addition, due to the extremely light weight of a ping pong ball, the pressure sensor would have to be so sensitive that it may give more false-positives.

Instead, we thought about utilizing a motion detector-type system that would report whether a ball had passed through a cup. Using pairs of IR LED emitters and binary output photodetectors, the photodetector would return a 0 in the presence of IR light and a 1 once the light was sufficiently interrupted by a ping pong ball. Once a ball was detected (1), a ring of LEDs around the cup that was tripped would turn from green to red indicating that you had already scored in that cup. Once tripped, any consecutive ping pong balls that fell in the cup would not be registered until the game was reset or re-racking put the cup back in play (green).

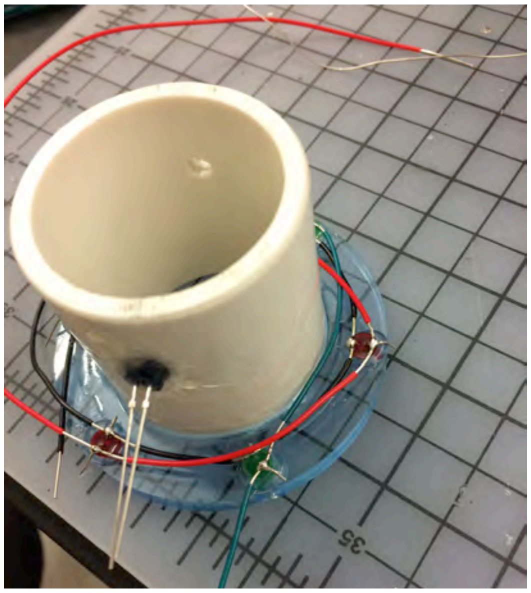
The next step was designing the cup units. We knew that the ball needed to sufficiently block the IR light emitted from the LED and so decided on using small pieces of PVC that were slightly larger in diameter than a ping pong ball. By placing an IR LED and photodetector opposite each other inside the PVC, the snug fit would cause the ball to completely block any IR light from reaching the detector. An arduino could then read in the values from the detector and change the color of the LEDs inset into rings of acrylic at the surface of the table. This would require usage of Solo cups that had holes in the bottom to allow the ball would pass through the PVC with the sensor, but given that we had removed the liquid element from the cups, it would still work.

Creating a Prototype

To build our prototype, we first started with a horizontal bandsaw to cut twenty, two-inch long pieces of 1.5" diameter PVC. We then used a drill to drill holes just large enough for the infra-red LED and photodetector in each PVC unit. Using AUTO-CAD, we designed small acrylic rings that had an inner diameter the width of the PVC and an outer circumference the size of the circular saw we would use to drill holes into our table. In these rings were six equidistant 5mm diameter holes that would hold the alternating red and green LEDs. We also designed rings of slightly larger outer diameter that would sit on top of the table and an inner diameter equal to that of the bottom of a typical Solo cup.

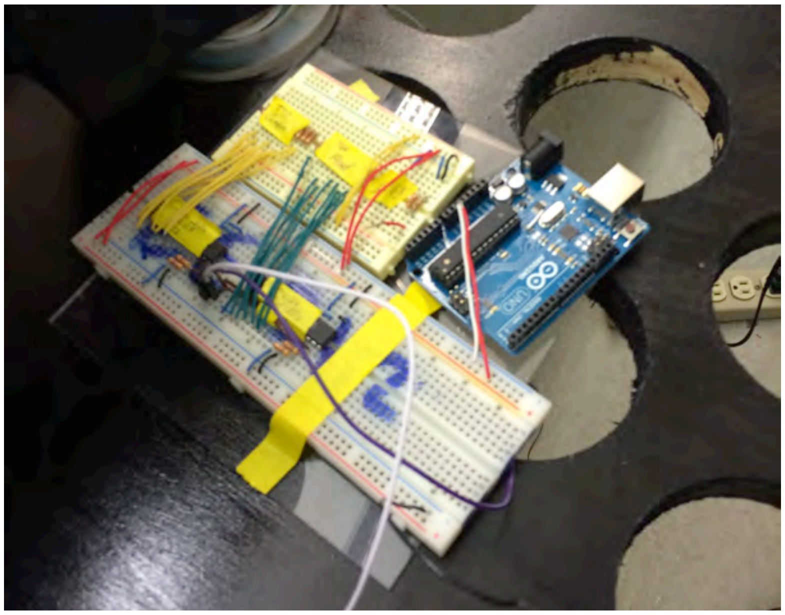
Moving forward from a prototype

We then laser cut twenty of each ring type in ¼" clear acrylic. We used cement to glue the PVC into the smaller ring and used hot glue to glue three green LEDs and three red LEDs into the holes in the smaller ring. The larger rings were painted with frosted glass spray paint to better diffuse the light from the color LEDs underneath and hot glued to the smaller ring. We made one prototype first, with the infrared LED inserted into one hole and the photodetector taped onto the other side of the LED. After building the appropriate circuit, we passed a ping pong ball through the prototype to see if it would trip the photodetector and change the lights from green to red, which it did, so we proceeded to make all 20 cup units. To reduce the number the wires coming from each cup unit, we soldered all the grounds for the LEDs and photodetector together into a single wire, all the red LEDs to a single wire, and all the green LEDs to a single wire.


Soldering one of the cup units

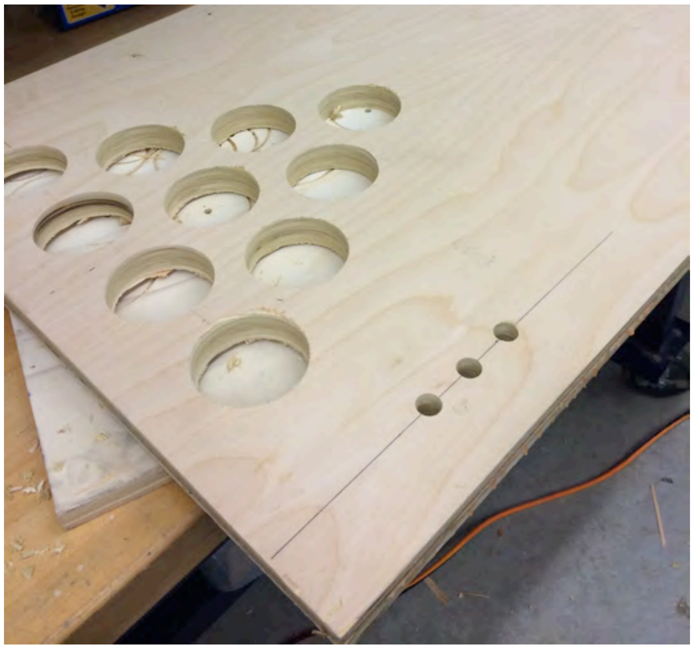
More of the cup units

With this design we realized we had a large number of inputs and outputs coming from each cup unit. In order to know which cup specifically had been tripped, each needed to have its own unique sensor (which consisted of a ground pin, voltage source pin, and output pin to the arduino), IR LED, red and green LEDs (alternating, 3 of each that were controlled by the arduino), and ground resulting in no fewer than 2 outputs (red/green LEDs) and 1 input (photodetector) that needed to be controlled by the arduino per cup unit. 20 cups would thus require 60 pins on the arduino, much more than available. Therefore we used port expanders that would functionally add up to 128 extra I/O pins (16 per expander). We decided against solely using shift registers because they only allow for output, not the inputs we needed to read each cup's photodetector.

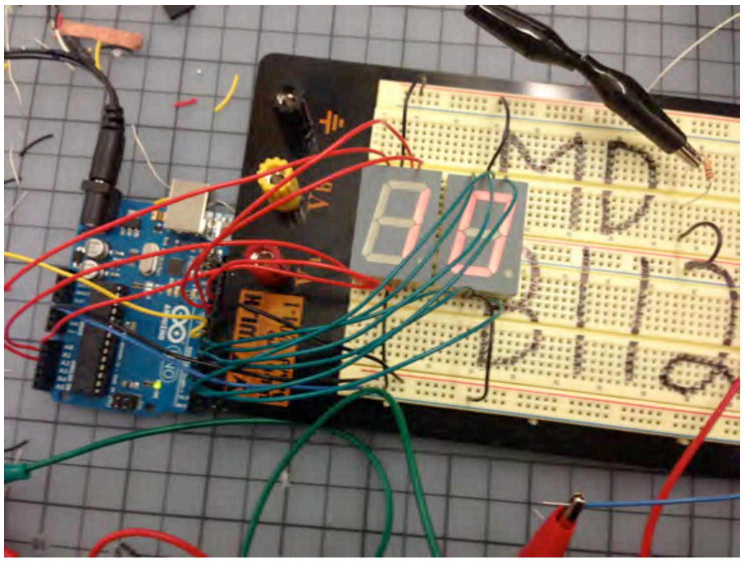

One of the final breadboard circuits with two port expanders

Designing the Table
For the physical table itself, we obtained a 4' x 8' x ½" sheet of birch plywood and cut it to have two 2' x 8' x ½" sheets. The sheets were glued together with wood glue to make a thicker more stable table and then 10 holes were drilled in each side using a circle saw to hold each of the cup units. We used a laser cut template that allowed us to drill the holes accurately and far enough apart that wide mouths of the Solo cups would have enough space. Three smaller holes were drilled on either side to hold the start/reset and score up and score down buttons and the table was stained with waterproof paint.

Gluing the plywood together


Post-drilling

With the main components of the table complete, the rest of the design consisted of elements that would respond to the input given by the photodetectors. Once a cup was tripped, it would increase the score given by two seven-segment digital displays by one; although there also would be physical buttons that would allow score manipulation to add/subtract points according to fouls or any other special rules decided on by the players.



Digital display circuit

Future Improvements
While there was unfortunately no time to implement the following, the plan was that once a cup was tripped the arduino would light up an illuminated push button on the opposite side of the playing team to signal that they needed to push the button to collect their drink. Once the button was pushed, it would send the appropriate voltage through a 12V gravity-feed solenoid valve, opening the valve for a predetermined amount of time and let the water/beer/drink-of-your-choosing flow into a cup. Other design elements also included sets of 10 illuminated push buttons on either side of the table that would represent the 10 cups and allow re-racking. Once the rerack button was pressed, the team could push and therefore light up the buttons that would represent the new configuration of cups they wanted in play.

PARTS LIST

Provide a table that summarizes all parts and materials you used. Include part numbers and links (assuming that you ordered them on-line), quantities used, and the price (including the grand-total). Also indicate the parts that you found in the lab, but also try to identify their costs and on-line sources.

Part	**Part Number**	**Link**	**Price per unit**	**Quantity used**	**Total cost**
Birch Plywood	833185	http://www.homedepot.com/p/Project-	$41.95	1	$41.95

		Panels-Whole-Piece-Birch-Domestic-Plywood-Price-Varies-by-Size-833185/100020218?N=5yc1vZbqm7Z25egxhZ1z0z6htZ1z11605			
16mm Illuminated Pushbutton - Blue Momentary	1477	http://www.adafruit.com/product/1477	$1.95	6	$11.70
Varathane Ebony Wood Stain	266266	http://www.homedepot.com/p/Varathane-1-2-pt-Ebony-Wood-Stain-266266/203310388	$5.26	1	$5.26
Red LEDs	COM-09590	https://www.sparkfun.com/products/9590	$0.35	60	$21
Green LEDs	COM-09592	https://www.sparkfun.com/products/9592	$0.35	60	$21
¼” Acrylic (clear and clear blue)	S1-12x12-.25	http://www.amazon.com/Source-Premium-Acrylic-PlexiGlass-S1-12x12-25/dp/B00844SOSE	$12.99	4	$51.96
IR LED Emmitor 950nm, 5mm	TSUS5202	http://www.digikey.com/product-detail/en/TSUS5202/TSUS5202-ND/4073538	$0.34	25	$8.60

IR Photodetector w/ Schmitt Trigger Circuit	SFH 5140 F	http://www.digikey.com/product-detail/en/SFH%205140%20F/475-1476-ND/1228129	$0.58	25	$14.55
Gorilla Wood Glue	6200	http://www.homedepot.com/p/Gorilla-8-fl-oz-Wood-Glue-62000/100672167	$3.83	1	$3.83
Frosted Glass Spray Paint	1903830	http://www.homedepot.com/p/Rust-Oleum-Specialty-11-oz-Frosted-Glass-Spray-Paint-1903830/100195608	$3.76	1	$3.76
1" Digit Size, 7-segment display (red)	SA10-21EWA	http://www.digikey.com/product-detail/en/SA10-21EWA/754-1682-5-ND/3084477	$2.94	4	$11.76
i2c 16 input/output port expander	MCP23017	http://www.adafruit.com/product/732	$2.95	4	$11.80
				Grand Total:	**$155.21**

PROJECT IMPLEMENTATION

We bought two pieces of plywood at standard beer pong table size from Home Depot, sanded one side of each piece of wood and used wood glue and clamps to glue them together. We also painted the table with ebony-coloured waterproof paints.

We used Auto-CAD to create a drawing of all the cups laid side by side in a beer pong formation with the centres of each circle drawn as holes. We lasercut the formation, and used it to figure out the centre of the drill of a circular saw drill bit. We used the circular saw to drill through the plywood. We then used larger drill bits to drill in holes for the button, and a combination of larger drill bits and a jigsaw to cut out a large rectangular box for the seven-segment display units.

We wrote the code for the seven-segment displays and for the cups. We then built the circuits for the cups and the seven-segment displays on breadboards first to make sure our code worked and our circuit diagrams were not faulty. Once we had figured out how to make the circuits work on the breadboard, we re-built less cumbersome versions of the circuits that would be fixed under the table.

While one of us was constructing the two seven-segment display circuits, we began testing the other 19 cup units and realized that by soldering the infra-red sensors, we had accidentally killed them. We did not have enough infra-red sensors to re-do all 20 cups, so we quickly rebuilt only two of them, one cut unit for each side, and hooked up the LEDs in the other cups to the cup units which were working on each side for display purposes. Instead of our original goal, in which we would have a workable game of digital beer pong, only the first cup would sense the motion of the ping pong ball, causing all the cups on that side to turn from green to red for demonstration purposes. Ultimately, due to running out of time, we had each side of the table controlled by two arduinos - one for the cup units and one for the digital displays. We would advise those who would like to replicate our results to avoid soldering around the photodetectors - we only used tape on our prototype, which is why we had not noticed that soldering in the sensor would break it.

Once we had finished with each other circuits, we disconnected and reassembled them to fit them into the table, using duct tape and hot glue to fix them to under the table.

Wiring the underside of the table

TEAM MANAGEMENT

We did all the “heavy” mechanical work together, often with supervision with one of our TFs, including gluing the table together, drilling through the holes in the table with the circular saw, cutting out the parts necessary to build the cup units.

We then split up the tasks that did not require three people to work on between us. Student A was in charge of soldering each cup unit, Student B wrote the code and built the circuits for the cup units, and Student C wrote the code and built the circuits for the seven-segment displays. Student A and Student B, who had more experience with code, were excellent at troubleshooting our code. Student C also worked on the implementation front in the final stages, figuring out how to place all the circuits and Arduinos under the table.

OUTLOOK AND POSSIBLE IMPROVEMENTS

First, after getting more of the IR photodetectors, we would replace all the sensors in each of the cup units and modify the code to constantly monitor each sensor and switch an individual cup unit’s LEDs from green to red if a ping pong ball was detected. In addition, we had originally hoped to build a beer dispenser, which would automatically dispense beer when a cup unit was triggered (i.e. when someone earned a point). We unfortunately did not have enough time to build the dispenser, but we hope to keep working on the project next semester and are considering adding the dispenser in. Similarly we would also implement some kind of rerack function, whether it’s in the form of the 10 illuminated push buttons per side or having preset reracks that an individual could cycle through and choose with a single button. Hopefully, we would also be able to better condense all our circuitry and code to minimize the number of separate arduinos controlling to the table.

ACKNOWLEDGEMENTS:

We would like to acknowledge the help given to us by Felipe Da Cruz, our TF, who taught us how to use all the power tools in Pierce. He was instrumental in helping us use AUTO-CAD to design the cup units, how to create pieces needed in making the cup units, and cutting out the holes in the tables. He dedicated an unreal number of hours at sometimes horrible hours of the night/morning to help us out and we truly appreciated it.

REFERENCES:

List any references you cite in your report. This includes web site, software that you downloaded, Youtube videos that were helpful, various on-line tutorials, etc.

http://arduino.cc/en/Tutorial/Button
https://github.com/mwilliams/barduino/blob/master/barduino.rb
http://www.hacktronics.com/Tutorials/arduino-and-7-segment-led.html
i2c Tutorial Part 1: http://tronixstuff.com/2010/10/20/tutorial-arduino-and-the-i2c-bus/
i2c Tutorial Part 2: http://tronixstuff.com/2010/10/29/tutorial-arduino-and-the-i2c-bus-part-two/
MCP23017 Tutorial: http://tronixstuff.com/2011/08/26/tutorial-maximising-your-arduinos-io-ports/

APPENDIX

Attach any extra drawings/diagrams that could be useful in this report or are referred to in the report but are too bulky to include in the text … e.g. extra photos, blueprints etc.
Also include the codes (arduino, processing, matlab) that you wrote/ modified.

NOTE: if you have substantial code (> 200 lines or multiple files) please save your code and this report in a zip archive titled TeamX_ES50_Final_Project_Report_2014.zip and submit that to Canvas. Each team should submit only one file - a zip or a pdf.

Submit as PDF OR zip archive including report and code

```
#include "Wire.h"

int sensorLevel1;
int sensor1 = 2;
int set1 = 0;
int mem1 = 0;

int sensorLevel2;
int sensor2 = 3;
int set2 = 0;
int mem2 = 0;

int reset1 = 4;
int reset2 = 5;
int begin_game1;
int begin_game2;
int start_game = 0;

void setup()
{
  Serial.begin(9600);

  pinMode(sensor1, INPUT);
  pinMode(sensor2, INPUT);
  pinMode(reset1, INPUT);
  pinMode(reset2, INPUT);

  Wire.begin();

  // make port B1 all outputs (T1 green LEDs)
  Wire.beginTransmission(0x20);
  Wire.write(0x01);
  Wire.write(0x00);
  Wire.endTransmission();

  // make port A1 all outputs (T1 green LEDs)
  Wire.beginTransmission(0x20);
  Wire.write(0x00);
  Wire.write(0x00);
  Wire.endTransmission();

  // make port B2 all outputs (T1 red LEDs)
  Wire.beginTransmission(0x21);
  Wire.write(0x01);
  Wire.write(0x00);
  Wire.endTransmission();

  // make port A2 all outputs (T1 red LEDs)
  Wire.beginTransmission(0x21);
  Wire.write(0x00);
  Wire.write(0x00);
  Wire.endTransmission();
```

```
// make port B1 all outputs (T2 green LEDs)
Wire.beginTransmission(0x23);
Wire.write(0x01);
Wire.write(0x00);
Wire.endTransmission();

// make port A1 all outputs (T2 green LEDs)
Wire.beginTransmission(0x23);
Wire.write(0x00);
Wire.write(0x00);
Wire.endTransmission();

// make port B2 all outputs (T2 red LEDs)
Wire.beginTransmission(0x24);
Wire.write(0x01);
Wire.write(0x00);
Wire.endTransmission();

// make port A2 all outputs (T2 red LEDs)
Wire.beginTransmission(0x24);
Wire.write(0x00);
Wire.write(0x00);
Wire.endTransmission();

// make sure all lights are off
// T1
  Wire.beginTransmission(0x20);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x20);
  Wire.write(0x12);
  Wire.write(0);
  Wire.endTransmission();

  Wire.beginTransmission(0x21);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x21);
  Wire.write(0x12);
  Wire.write(0);
  Wire.endTransmission();

 // T2
  Wire.beginTransmission(0x23);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x23);
  Wire.write(0x12);
```

```
        Wire.write(0);
        Wire.endTransmission();

        Wire.beginTransmission(0x24);
        Wire.write(0x13);
        Wire.write(0);
        Wire.endTransmission();
        Wire.beginTransmission(0x24);
        Wire.write(0x12);
        Wire.write(0);
        Wire.endTransmission();

}

void loop()
{
  // start or reset game if start button pressed
  if (start_game == 0)
  {
     begin_game1 =digitalRead(reset1);
     begin_game2 =digitalRead(reset2);
    if (begin_game1 || begin_game2   ==HIGH)
    {
      start_game = 1;
    }
  }
  else
  {

  // check if reset (start game) button pressed
   begin_game1 =digitalRead(reset1);
   begin_game2 =digitalRead(reset2);

  // reset game if pressed
  if (begin_game1 || begin_game2 ==HIGH)
  {
    mem1 = 0;
    mem2 = 0;
    set1 = 0;
    set2 = 0;
  }

  // check cup 1 sensor
   sensorLevel1 =digitalRead(sensor1);
  Serial.println(sensorLevel1);

  // if ball went through cup one, flash and turn red
  if (sensorLevel1 == 1 && mem1 == 0)
  {

   for (int blink =  0;blink  <  3;blink++)
   {
```

```
 // turn off green lights
  Wire.beginTransmission(0x20);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x20);
  Wire.write(0x12);
  Wire.write(0);
  Wire.endTransmission();
  delay(500);

   // turn on green lights
  Wire.beginTransmission(0x20);
  Wire.write(0x13);
  Wire.write(0xFF);
  Wire.endTransmission();
  Wire.beginTransmission(0x20);
  Wire.write(0x12);
  Wire.write(0xFF);
  Wire.endTransmission();
  delay(500);
 }

  // turn off green lights
  Wire.beginTransmission(0x20);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x20);
  Wire.write(0x12);
  Wire.write(0);
  Wire.endTransmission();

  // turn on red lights
  Wire.beginTransmission(0x21);
  Wire.write(0x13);
  Wire.write(0xFF);
  Wire.endTransmission();
  Wire.beginTransmission(0x21);
  Wire.write(0x12);
  Wire.write(0xFF);
  Wire.endTransmission();

  // remember sensor was tripped
  mem1 = 1;

}

// check cup 2 sensor
 sensorLevel2  =digitalRead(sensor2);
Serial.println(sensorLevel2);
```

```
// if ball went through cup two, flash and turn red
if (sensorLevel2 == 1 && mem2 == 0)
{

  for (int blink =  0;blink <  3;blink++)
  {
 // turn off green lights
  Wire.beginTransmission(0x23);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x23);
  Wire.write(0x12);
  Wire.write(0);
  Wire.endTransmission();
  delay(500);

  // turn on green lights
  Wire.beginTransmission(0x23);
  Wire.write(0x13);
  Wire.write(0xFF);
  Wire.endTransmission();
  Wire.beginTransmission(0x23);
  Wire.write(0x12);
  Wire.write(0xFF);
  Wire.endTransmission();
  delay(500);
  }

  // turn off green lights
  Wire.beginTransmission(0x23);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x23);
  Wire.write(0x12);
  Wire.write(0);
  Wire.endTransmission();

  // turn on red lights
  Wire.beginTransmission(0x24);
  Wire.write(0x13);
  Wire.write(0xFF);
  Wire.endTransmission();
  Wire.beginTransmission(0x24);
  Wire.write(0x12);
  Wire.write(0xFF);
  Wire.endTransmission();

  // remember sensor was tripped
  mem2 = 1;
```

```
}

// if sensor1 not tripped yet, turn on green lights
if (mem1 == 0 && set1 == 0)
{
  // turn on green lights
  Wire.beginTransmission(0x20);
  Wire.write(0x13);
  Wire.write(0xFF);
  Wire.endTransmission();
  Wire.beginTransmission(0x20);
  Wire.write(0x12);
  Wire.write(0xFF);
  Wire.endTransmission();

  // turn off red lights
  Wire.beginTransmission(0x21);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x21);
  Wire.write(0x12);
  Wire.write(0);
  Wire.endTransmission();

  // make it so it doesn't go into this loop again unless reset
  set1 = 1;
}

// if sensor2 not tripped yet, turn on green lights
if (mem2 == 0 && set2 ==0)
{
  // turn on green lights
  Wire.beginTransmission(0x23);
  Wire.write(0x13);
  Wire.write(0xFF);
  Wire.endTransmission();
  Wire.beginTransmission(0x23);
  Wire.write(0x12);
  Wire.write(0xFF);
  Wire.endTransmission();

  // turn off red lights
  Wire.beginTransmission(0x24);
  Wire.write(0x13);
  Wire.write(0);
  Wire.endTransmission();
  Wire.beginTransmission(0x24);
  Wire.write(0x12);
  Wire.write(0);
  Wire.endTransmission();
```

```
      // make it so it doesn't go into this loop again unless reset
      set1 = 1;
    }

    }

  }
```