Whac-a-Mole

Karen Escareno

Jeffrey Lee

Group 13

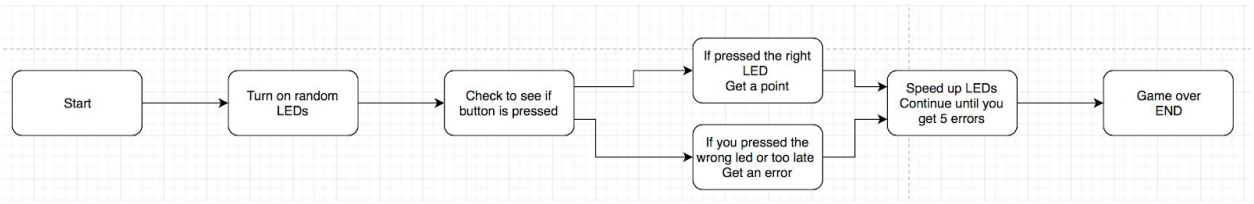
5/8/19

Project description

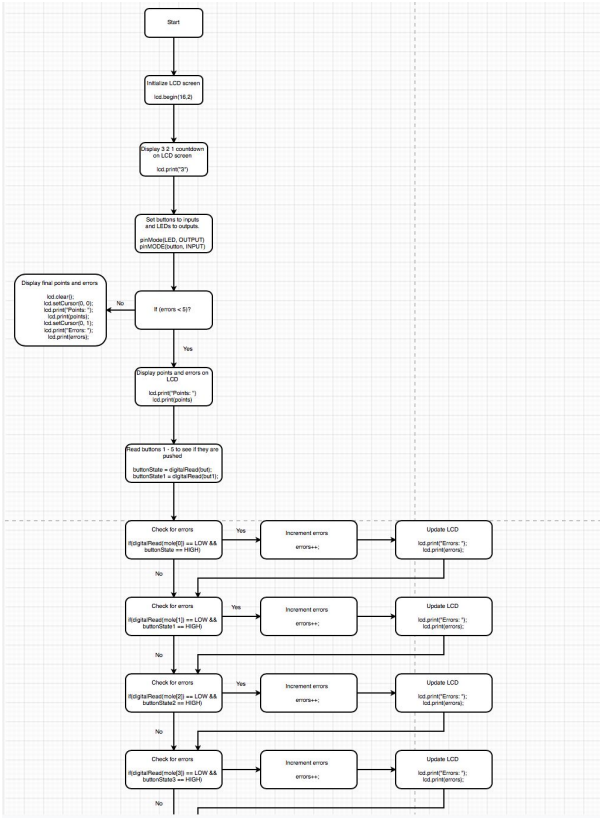
The project is to play whac-a-mole. The moles are LEDs and buttons are used to indicate whether the moles are hit or not. An Arduino Mega is the microcontroller used for this project. The rate the moles pop up increase as more points are earned. In order to lose the game, 5 errors need to be acquired. To get an error, the player must either miss a mole or hit a space where a mole is not at.score and errors are displayed on an LCD screen.

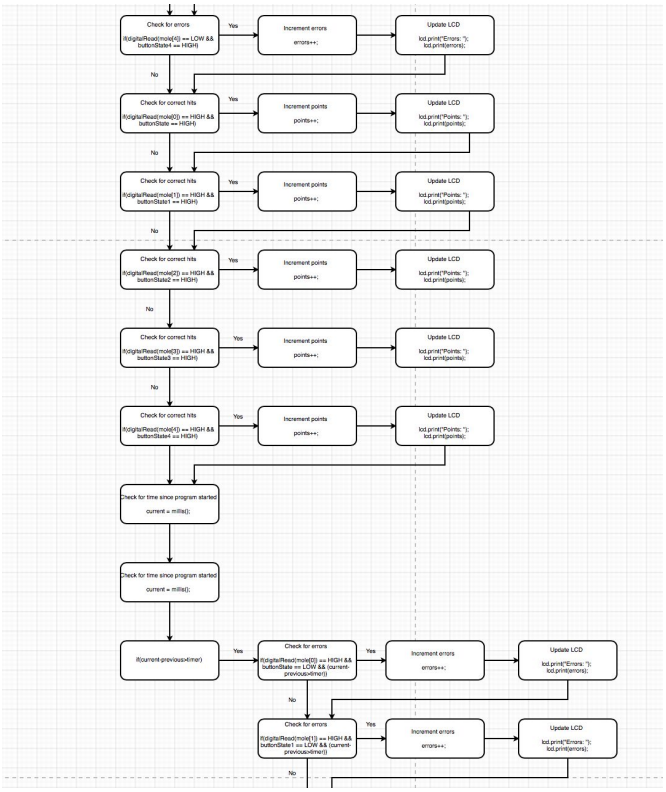
System design

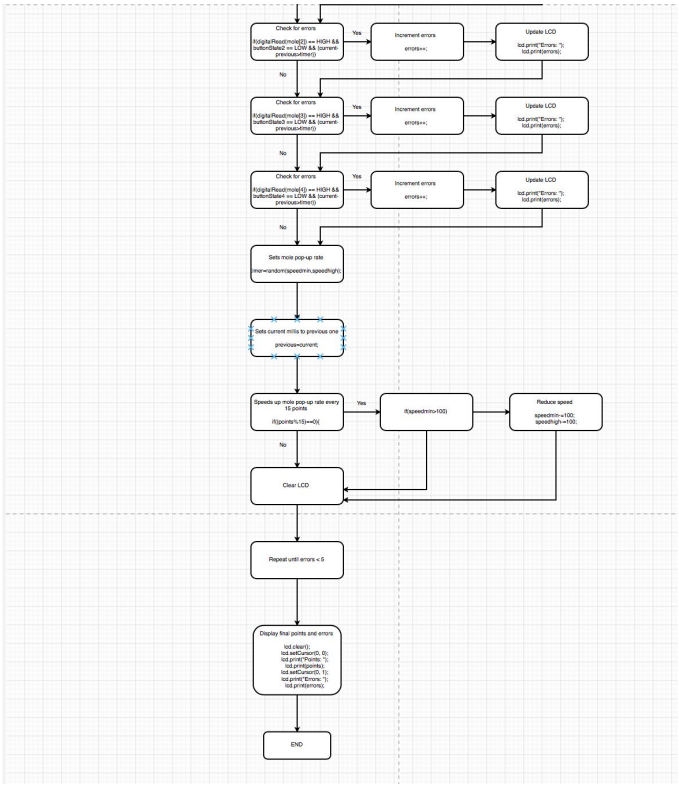
High level block diagram



Flowchart


Start
Initialize LCD screen
lcd.begin(16,2)
Display 3 2 1 countdown on LCD screen
lcd.print("3")
Set buttons to inputs and LEDs to outputs.
pinMode(LED, OUTPUT)
pinMODE(button, INPUT)
Display final points and errors
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Points: ");
lcd.print(points);
lcd.setCursor(0, 1);
lcd.print("Errors: ");
lcd.print(errors);
No
If (errors < 5)?
Yes
Display points and errors on LCD
lcd.print("Points: ")
lcd.print(points)
Read buttons 1 - 5 to see if they are pushed
buttonState = digitalRead(but);
buttonState1 = digitalRead(but1);
Check for errors
if(digitalRead(mod[0]) == LOW && buttonState == HIGH)
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No
Check for errors
if(digitalRead(mod[1]) == LOW && buttonState1 == HIGH)
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No
Check for errors
if(digitalRead(mod[2]) == LOW && buttonState2 == HIGH)
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No
Check for errors
if(digitalRead(mod[3]) == LOW && buttonState3 == HIGH)
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No


Check for errors
if(digitalRead(mole[4]) == LOW && buttonState4 == HIGH)
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No
Check for correct hits
if(digitalRead(mole[0]) == HIGH && buttonState == HIGH)
Yes
Increment points
points++;
Update LCD
lcd.print("Points: ");
lcd.print(points);
No
Check for correct hits
if(digitalRead(mole[1]) == HIGH && buttonState1 == HIGH)
Yes
Increment points
points++;
Update LCD
lcd.print("Points: ");
lcd.print(points);
No
Check for correct hits
if(digitalRead(mole[2]) == HIGH && buttonState2 == HIGH)
Yes
Increment points
points++;
Update LCD
lcd.print("Points: ");
lcd.print(points);
No
Check for correct hits
if(digitalRead(mole[3]) == HIGH && buttonState3 == HIGH)
Yes
Increment points
points++;
Update LCD
lcd.print("Points: ");
lcd.print(points);
No
Check for correct hits
if(digitalRead(mole[4]) == HIGH && buttonState4 == HIGH)
Yes
Increment points
points++;
Update LCD
lcd.print("Points: ");
lcd.print(points);
Check for time since program started
current = millis();
Check for time since program started
current = millis();
if(current-previous>timer)
Yes
Check for errors
if(digitalRead(mole[0]) == HIGH && buttonState == LOW && (current-previous>timer))
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No
Check for errors
if(digitalRead(mole[1]) == HIGH && buttonState1 == LOW && (current-previous>timer))
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No


Check for errors
if(digitalRead(mole[2]) == HIGH && buttonState2 == LOW && (current-previous>timer))
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No
Check for errors
if(digitalRead(mole[3]) == HIGH && buttonState3 == LOW && (current-previous>timer))
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No
Check for errors
if(digitalRead(mole[4]) == HIGH && buttonState4 == LOW && (current-previous>timer))
Yes
Increment errors
errors++;
Update LCD
lcd.print("Errors: ");
lcd.print(errors);
No
Sets mole pop-up rate
timer=random(speedmin,speedhigh);
Sets current millis to previous one
previous=current;
Speeds up mole pop-up rate every 15 points
if((points%15)==0){
Yes
if(speedmin>100)
Reduce speed
speedmin-=100;
speedhigh-=100;
No
Clear LCD
Repeat until errors < 6
Display final points and errors
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Points: ");
lcd.print(points);
lcd.setCursor(0, 1);
lcd.print("Errors: ");
lcd.print(errors);
END

Implementation code

if(digitalRead(mole[0]) == LOW && buttonState == HIGH) **// checks if hits empty space**

if(digitalRead(mole[0]) == HIGH && buttonState == HIGH) **// checks if mole is hit**

if(digitalRead(mole[0]) == HIGH && buttonState == LOW && (current-previous>timer) **// checks if mole is missed**

if(current-previous>timer) **// timer**

current = millis**(); // delay without using delay()**

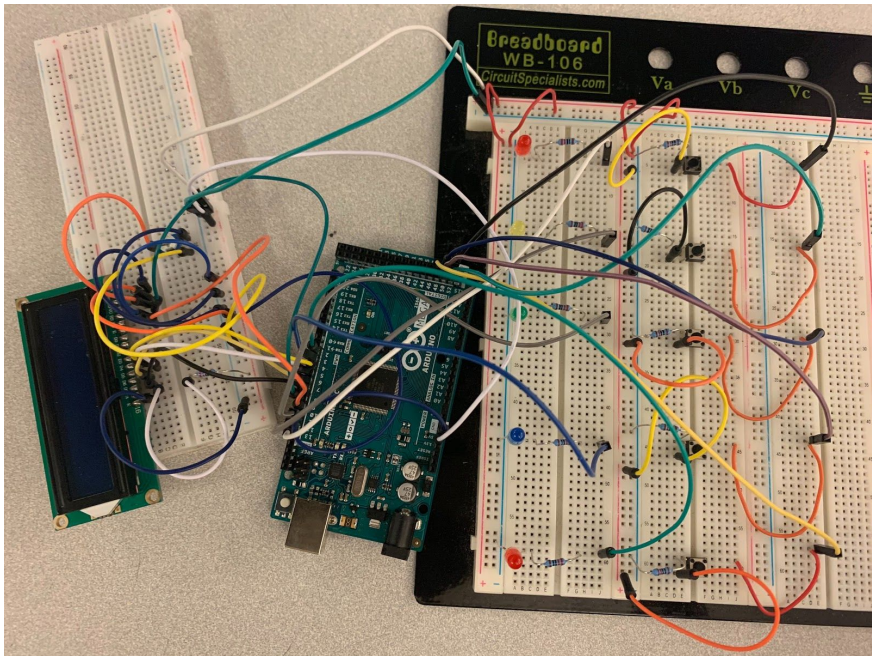
digitalWrite(mole[random(0,sizeof(mole)/sizeof(int))],HIGH); **// randomly displays moles**

timer=random(speedmin,speedhigh); **// sets mole pop-up rate to a variable**

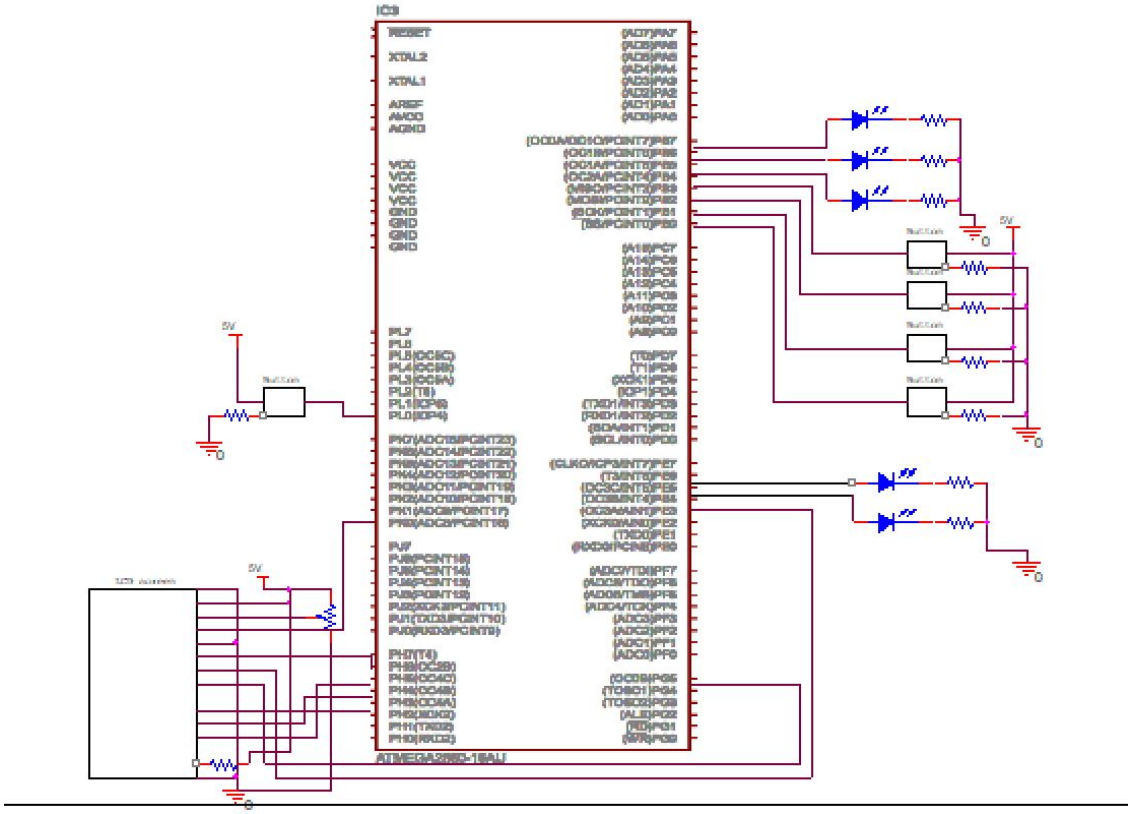
previous=current; **// sets current millis to previous one**

if((points%15)==0) **// speeds up mole pop-up rate every 15 points**

Image:



Schematic:



Testing

Mole hit: hooked up one LED to one button; checked to see if points were incremented correctly

Mole missed: hooked up one LED to one button; checked to see if errors were incremented correctly

Empty hit: hooked up one LED to one button; checked to see if button press on empty space incremented errors

Random pop-up: 5 LEDs; checked to see if LEDs lit up and disappeared at random times

Speed up: 5 LEDs and 5 buttons; checked to see if moles speed up after a certain amount of hits

Discussion

Finding a way to delay without using the delay() function:

If we used the delay() function in Arduino, the whole program stops and the timing of the hits would not work. We needed to find a way to delay while the code still runs. So we used millis() instead of delay(). millis() keeps track of how long the code has been running and we use that as our timer.

A limitation of our design is that only one mole pops up at once. An improvement would be that multiple moles would be up at the same time.

Responsibilities

Jeffrey Lee: debugging code and rewiring hardware

Karen Escareno: Hardware and Code

Conclusion

We successfully achieved the goal for the Whac-a-mole game. Moles pop up randomly and speed up as points are obtained. There are definite improvements that can be made for this game. Overall the project works well.