Class:	CPE301L	Semester:	Spring 2024

Points		Document author:	Yanai Avila
		Author's email:	avilay1@unlv.nevada.edu
		Document topic:	Postlab 1: C and Assembly Programming

Instructor's comments:

1. Introduction / Theory of Operation

This lab allows one to work with both the C language and assembly. One uses an Arduino online environment to review the basics of C programming, and OnlineGDB to review the basics of assembly. It tests one's abilities by implementing 6 different circuits on Tinkercad using variables, loops, switch statements, and other conditional statements. Additionally, it allows one to learn how to work with registers in assembly using the debug tools on OnlineGDB.

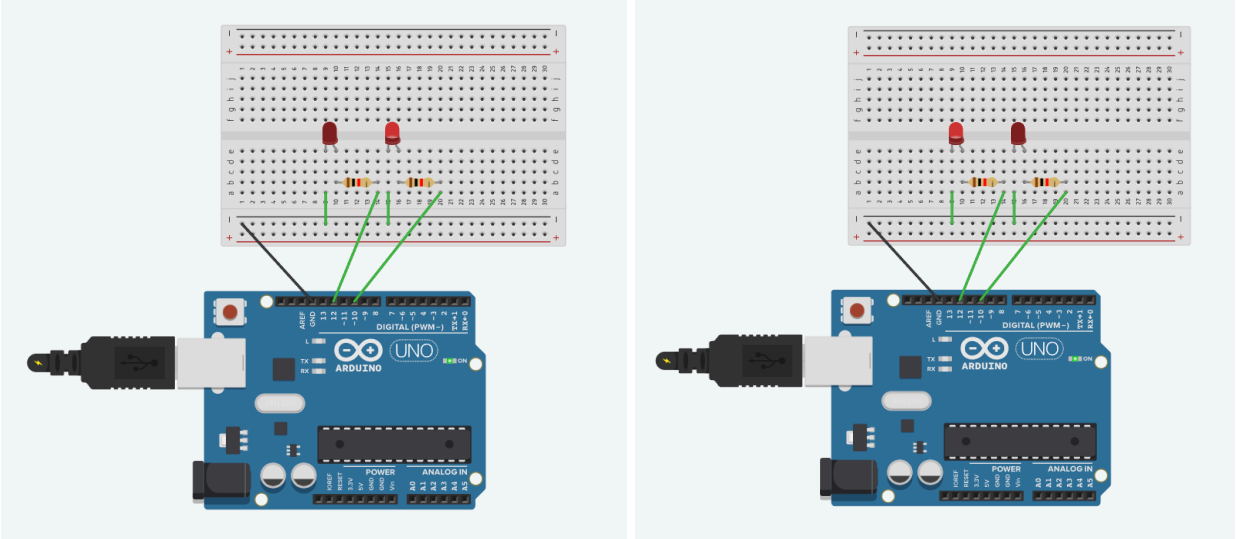
2. Prelab

No prelab

3. Results of Experiments

Experiment 1: LEDs inverse blinking (Link to Tinkercad circuit and code)

Screenshot of the circuit:

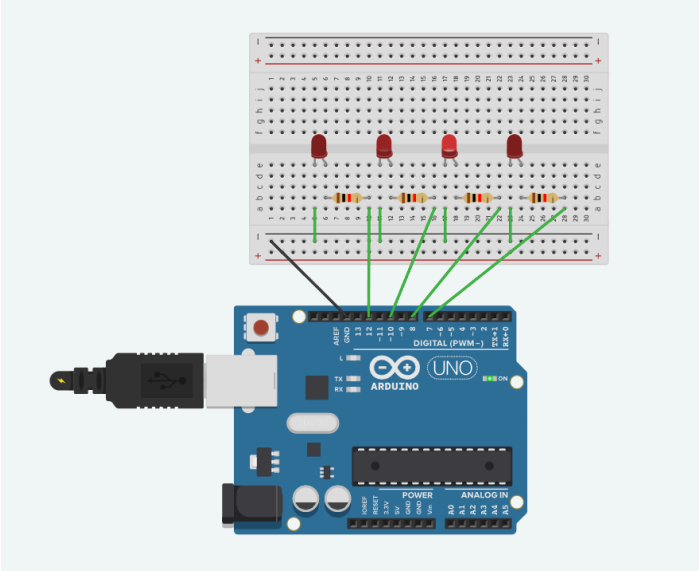


Code:

```
// Experiment 1: LEDs blink in inverse

int LED1 = 10;
int LED2 = 12;


void setup()
{
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
}

void loop()
{
  // LED1 on, LED2 off
  digitalWrite(LED1, HIGH);
  digitalWrite(LED2, LOW);
  // Wait for 1000 millisecond(s)
  delay(1000);
  // LED 1 off, LED 2 on
  digitalWrite(LED1, LOW);
  digitalWrite(LED2, HIGH);
  // Wait for 1000 millisecond(s)
  delay(1000);
}

```

Experiment 2: LEDs blinking in sequence (Link to Tinkercad circuit and code)

Screenshot of the circuit:



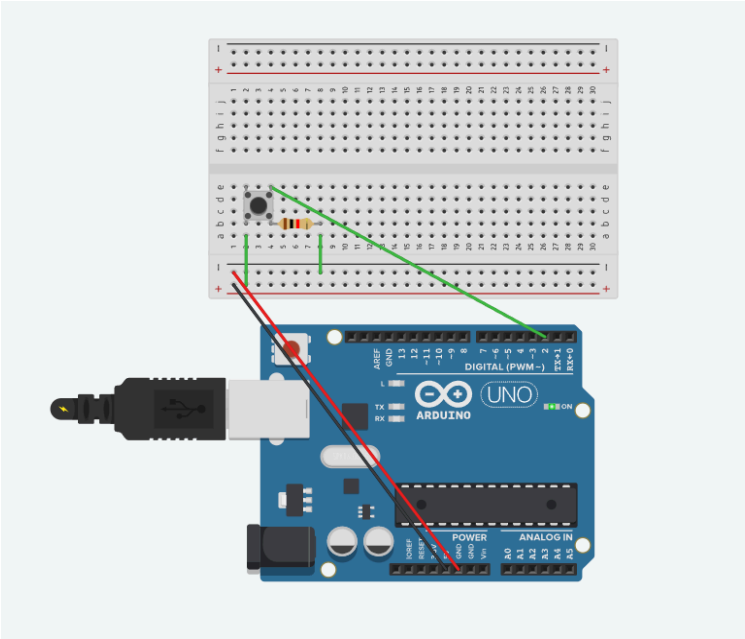
Code:

```
// Experiment 2: One of many LED operation

int LED0 = 12;
int LED1 = 10;
int LED2 = 8;
int LED3 = 7;

// Will indicate which of LEDs is on
int pos = 0;

void setup()
{
  pinMode(LED0, OUTPUT);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
}

void loop()
{
  // Switch statement to make LEDs light up one by one
  switch(pos++ % 4)
    {
    case 0:
        digitalWrite(LED0, HIGH);
        digitalWrite(LED1, LOW);
        digitalWrite(LED2, LOW);
        digitalWrite(LED3, LOW);
        break;
    case 1:
        digitalWrite(LED0, LOW);
        digitalWrite(LED1, HIGH);
        digitalWrite(LED2, LOW);
        digitalWrite(LED3, LOW);
        break;
    case 2:
        digitalWrite(LED0, LOW);
        digitalWrite(LED1, LOW);
        digitalWrite(LED2, HIGH);
        digitalWrite(LED3, LOW);
        break;
```

```
    case 3:
        digitalWrite(LED0, LOW);
        digitalWrite(LED1, LOW);
        digitalWrite(LED2, LOW);
        digitalWrite(LED3, HIGH);
        break;
    default:
        digitalWrite(LED0, LOW);
        digitalWrite(LED1, LOW);
        digitalWrite(LED2, LOW);
        digitalWrite(LED3, LOW);
        break;
    }

    delay(500); // Wait
}

```

Experiment 3: Pushbutton (Link to Tinkercad circuit and code)

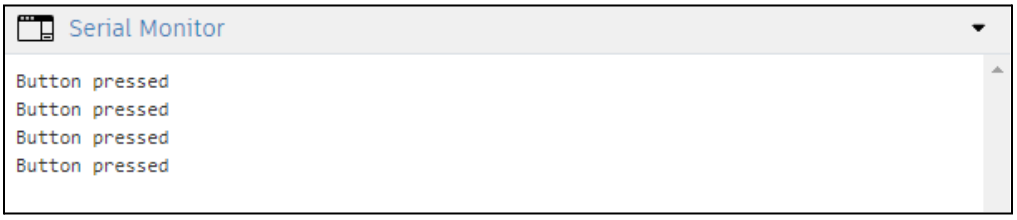
Screenshot of the circuit:



Code:

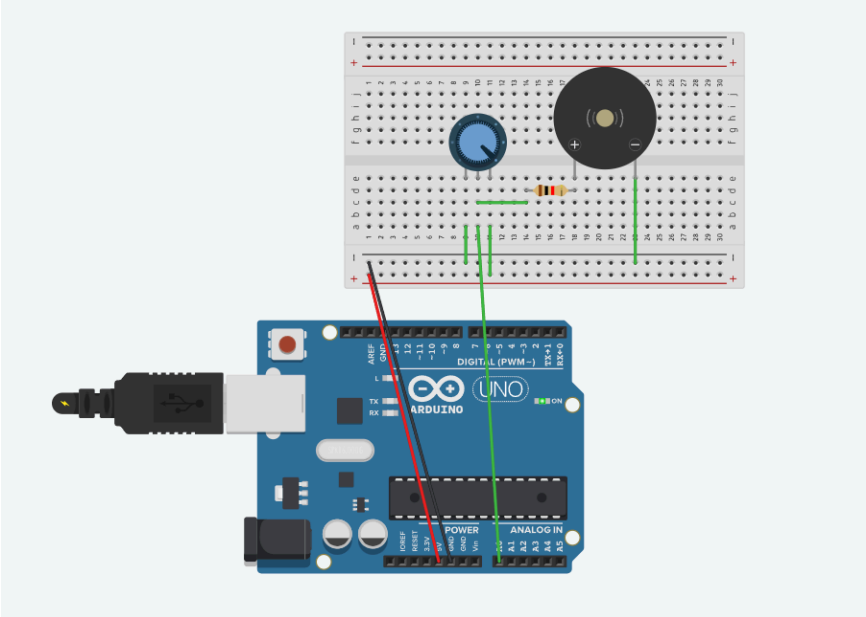
```
//Experiment 3: Pushbuttons

void setup()
{
  pinMode(2, INPUT);
  Serial.begin(9600);
}

void loop()
{
  // If button is pressed, then display
  // "Button pressed" in serial monitor
  if(digitalRead(2) == HIGH){
    Serial.println("Button pressed");
    delay(100);
  }
}

```

Serial monitor output:



Experiment 4: Changing buzzer with potentiometer (Link to Tinkercad circuit and code)

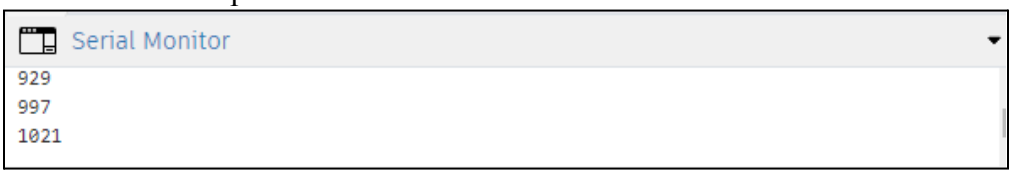
Screenshot of the circuit:



Code:

```
// Experiment 4: Analog read

void setup()
{
  pinMode(A0, INPUT);
  Serial.begin(9600);
}

// To store previous/current values of potentiometer
int pot_prev = 0;
int pot_cur;

void loop()
{
  pot_cur = analogRead(A0);

  // Display potentiometer value in the serial monitor,
  // but only if the potentiometer value changes
  if(pot_cur != pot_prev){
    Serial.println(pot_cur);
    pot_prev = pot_cur;
  }
}
```

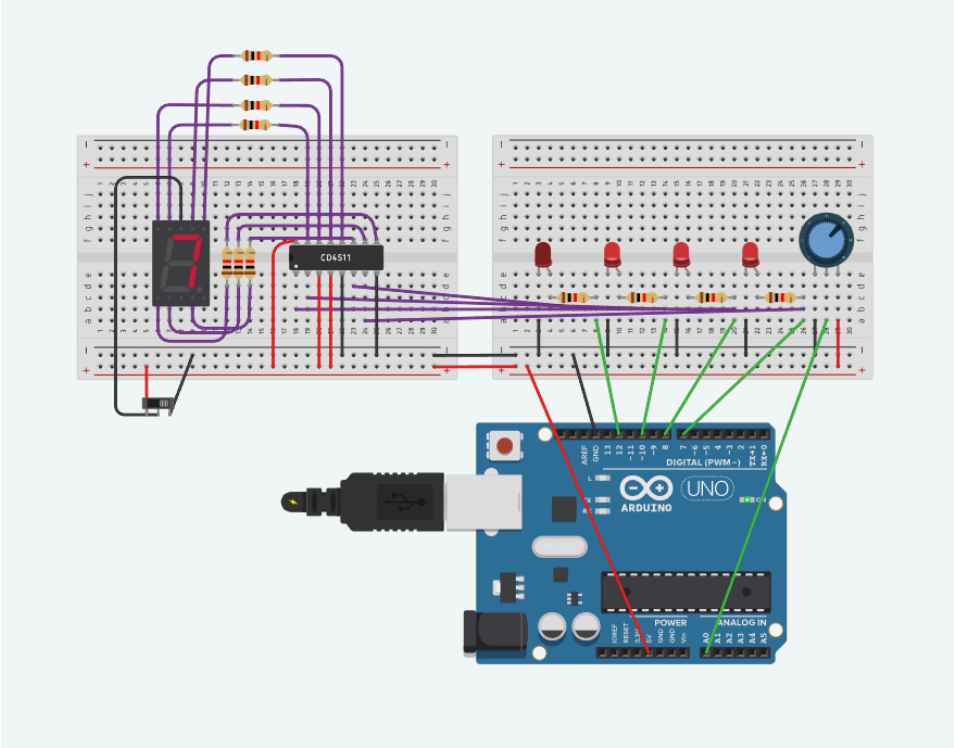
Serial monitor output:



NOTE: Serial monitor shows highest adc value. This is where the buzzer was the loudest.

Experiment 5: 7-segment display (Link to Tinkercad circuit and code)

Screenshot of the circuit:



Code:

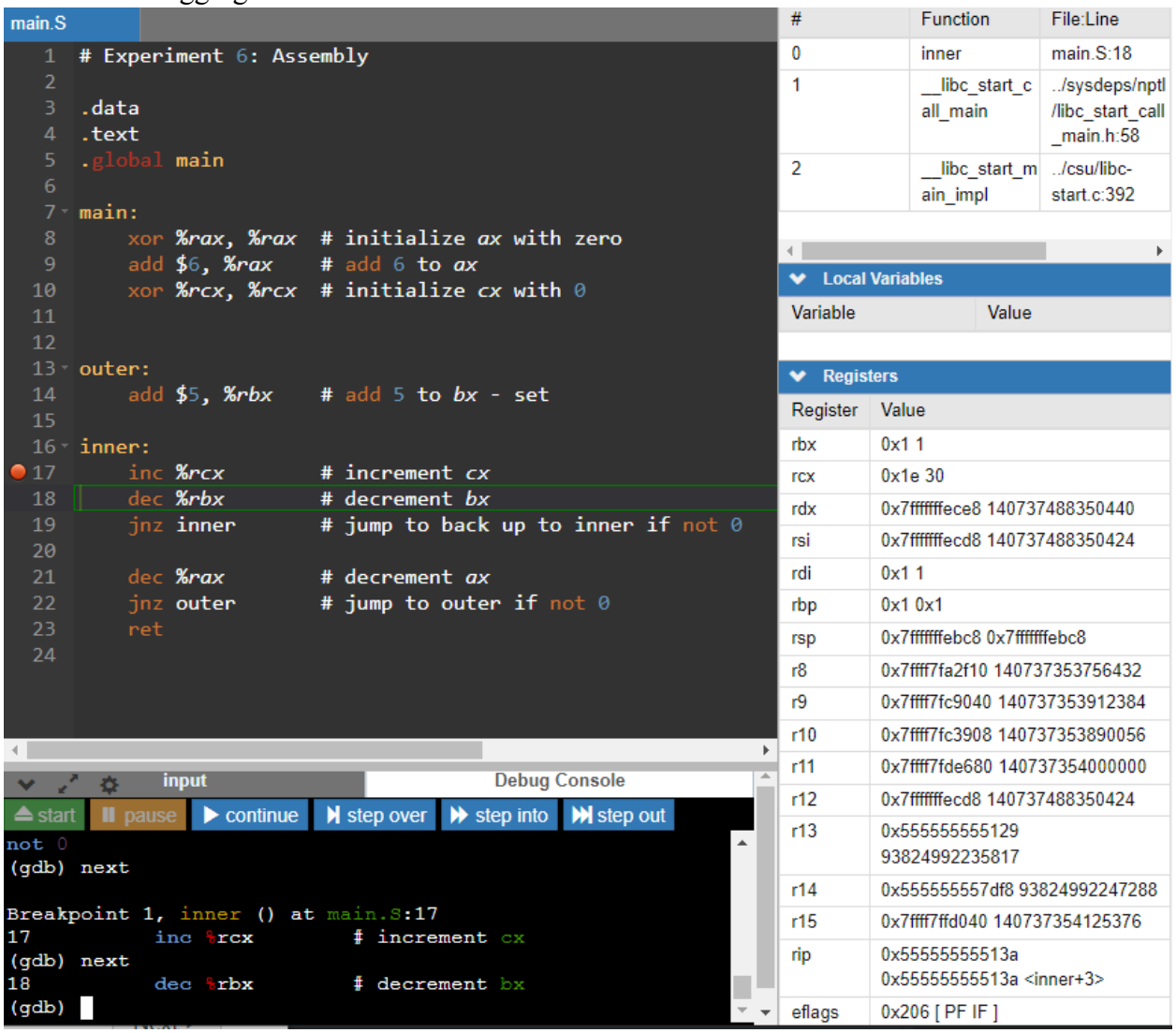
```
// Experiment 5: 7-segment

int LED0 = 12;
int LED1 = 10;
int LED2 = 8;
int LED3 = 7;

void setup()
{
  pinMode(LED0, OUTPUT);
  pinMode(LED1, OUTPUT);
  pinMode(LED2, OUTPUT);
  pinMode(LED3, OUTPUT);
  pinMode(A0, INPUT);
  Serial.begin(9600);
}

// digit variable
int digit = 0;

//// To store previous/current values of potentiometer
int pot_prev = 0;
int pot_cur;

```

```
void loop()
{
  // Divide the range of potentiometer into 10 ranges

  if (analogRead(A0) > 0 && analogRead(A0) < 110) {
    digit = 1;
  }
  else if (analogRead(A0) > 109 && analogRead(A0) < 220) {
    digit = 2;
  }
  else if (analogRead(A0) > 219 && analogRead(A0) < 330) {
     digit = 3;
  }
  else if (analogRead(A0) > 329 && analogRead(A0) < 440) {
    digit = 4;
  }
  else if (analogRead(A0) > 439 && analogRead(A0) < 550) {
    digit = 5;
  }
  else if (analogRead(A0) > 549 && analogRead(A0) < 660) {
    digit = 6;
  }
  else if (analogRead(A0) > 659 && analogRead(A0) < 770) {
    digit = 7;
  }
  else if (analogRead(A0) > 769 && analogRead(A0) < 880) {
    digit = 8;
  }
  else{digit = 9;}

  pot_cur = analogRead(A0);

  if(pot_cur != pot_prev){
    Serial.println(digit);
    // Passes digit into 7 seg function
    control7SEG(digit);
    pot_prev = pot_cur;
  }
    delay(100);
}

```

```
void control7SEG(int num){
  // Set LED0-LED3 with the value of current number in BCD
  digitalWrite(LED0, bitRead(num, 3));
  digitalWrite(LED1, bitRead(num, 2));
  digitalWrite(LED2, bitRead(num, 1));
  digitalWrite(LED3, bitRead(num, 0));
}

```

Experiment 6: Assembly

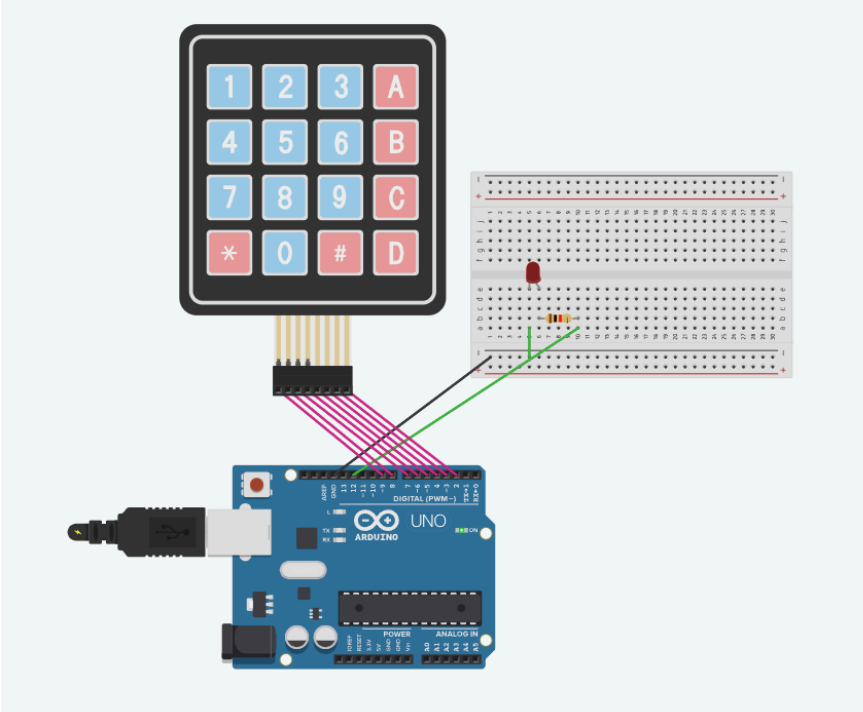
Code and debugging:



NOTE: RCX register contains the value of RAX*RBX right before program termination. In the list of registers, RCX is 30, which is RAX(6) x RBX(5).

Experiment 7: Keypad (Link to Tinkercad circuit and code)

Screenshot of the circuit:

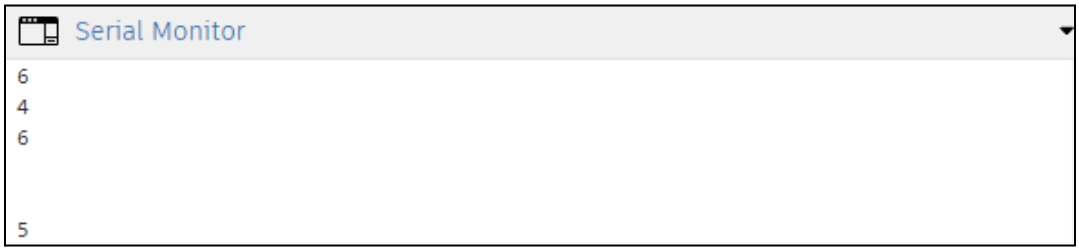


Code:

```
//Experiment 7: Keypad

#include <Keypad.h>

// # of rows and colunms for keypad
const byte r = 4;
const byte c = 4;

// Keypad connections -> arduino terminals
byte rpins[r] = {9,8,7,6};
byte cpins[c] = {5,4,3,2};

// map
char keys[r][c]=
{
{'1', '2', '3', 'A'},
{'4', '5', '6', 'B'},
{'7', '8', '9', 'C'},
{'*', '0', '#', 'D'}
};

// Instance of keypad
Keypad myKeypad = Keypad(makeKeymap(keys), rpins, cpins, r, c);


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

```

```
31  void loop()
32  {
33    char keypressed = myKeypad.getKey();
34    KeyState state = myKeypad.getState();
35
36    // when key is pressed, display its value in
37    // the serial monitor.
38    if (keypressed != NO_KEY && keypressed != 'D')
39    {
40      Serial.println(keypressed);
41    }
42
43    // When D is pressed, apply new line to the serial monitor
44    if (keypressed == 'D')
45    {
46      Serial.println("\n");
47    }
48
49
50    // light up LED for the time the key is pressed,
51    // off when not
52    if(state == HOLD)
53    {
54      digitalWrite(12, HIGH);
55    }
56    else{
57      digitalWrite(12, LOW);
58    }
59  }
60
```

Serial monitor output:



NOTE: The space in the serial monitor is due to the newline when pressing “D”

4. Answers to questions

1. What is the technical difference between analog and digital input of Arduino board?
An analog input can be any number of values within a range (0V to 5V in most cases) while a digital signal can only be two values (HIGH and LOW). Arduino has a built-in analog-to-digital converter (ADC) that can convert analog voltage into digital values between 0 and 1023.

2. What is a bitwise operation?
A bitwise operation is an operation that involves working with and manipulating individual bits. Examples include bitwise AND, bitwise OR, bitwise XOR, left shift, right shift, and binary ones complement.

5. Conclusions & Summary

This lab was a needed review of the C language and a good introduction to assembly. I was able to implement 6 circuits on Tinkercad which allowed me to practice using variables, loops, and conditional statements in C. It allowed me to work with LEDs, pushbuttons, potentiometers, and even a keypad, all while learning about analog and digital signals. I was also able to use OnlineGDB to practice my assembly and get a feel for how the debugging process works.

The only experiment I had problems with was Experiment 5 with the 7-segment display. At one point, my potentiometer and LEDs were working fine, but the 7-segment display was not. I knew my connections with the display were correct because it displayed a 0, but it would not change with the potentiometer. I did some troubleshooting by messing around with the 4 wires connected to the inputs of the CD4511 decoder. I realized that I had the wires on the wrong side of the resistor, so it was not reading the values. Once I made that switch, the 7-segment display worked as desired.