Data logging with Raspberry Pi and Arduino

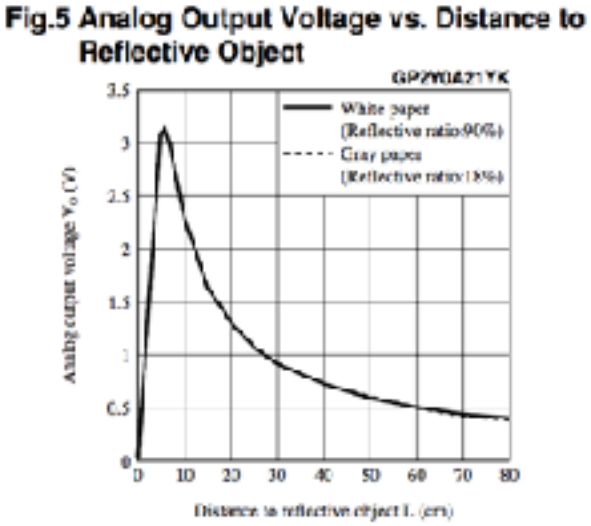
Yating Zhan (yz2237)
Collaboration with: Lannie Miao (ym232), Shanshan Zhang (sz438), Betty Chou (fc332)

Part A. Writing to the Serial Monitor

a. Based on the readings from the serial monitor, what is the range of the analog values being read?
0-1023
b. How many bits of resolution does the analog to digital converter (ADC) on the Arduino have?
10-bit analog to digital converter

Part B. Voltage Varying Sensors

1. IR Distance Sensor
a. Describe the voltage change over the sensing range of the sensor. A sketch of voltage vs. distance would work also. Does it match up with what you expect from the datasheet?
The reading starts from 100, jumps up to 900 and start decreasing. Convert reading to voltage, the graph matches up with my expectation from the datasheet.



2. Accelerometer
a. Include your accelerometer read-out code in your write-up.

```
// include the library code:
#include <LiquidCrystal.h>

// these constants describe the pins. They won't change:
const int groundpin = 18;            // analog input pin 4 -- ground
const int powerpin = 19;             // analog input pin 5 -- voltage
const int xpin = A3;                 // x-axis of the accelerometer
const int ypin = A2;                 // y-axis
const int zpin = A1;                 // z-axis (only on 3-axis models)

// initialize the library by associating any needed LCD interface pin
```

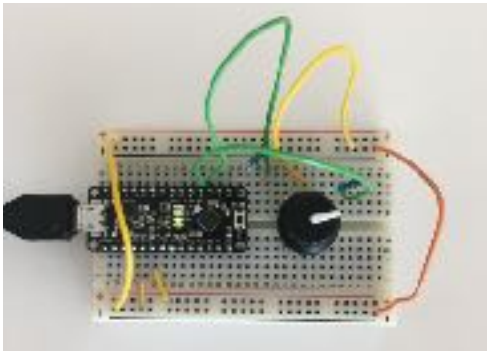
```
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
void setup() {
  // initialize the serial communications:
  Serial.begin(9600);
  lcd.begin(16, 3);

  // Provide ground and power by using the analog inputs as normal digital pins.
  // This makes it possible to directly connect the breakout board to the
  // Arduino. If you use the normal 5V and GND pins on the Arduino,
  // you can remove these lines.
  pinMode(groundpin, OUTPUT);
  pinMode(powerpin, OUTPUT);
  digitalWrite(groundpin, LOW);
  digitalWrite(powerpin, HIGH);
}

void loop() {
  // print the sensor values:
  Serial.print(analogRead(xpin));
  lcd.setCursor(0,0);
  lcd.print(String("x: ") + analogRead(xpin));
  // print a tab between values:
  Serial.print("\t");
  Serial.print(analogRead(ypin));
  lcd.setCursor(0,1);
  lcd.print(String("y: ") + analogRead(ypin));
  // print a tab between values:
  Serial.print("\t");
  Serial.print(analogRead(zpin));
  Serial.println();
  lcd.setCursor(8,1);
  lcd.print(String("z: ") + analogRead(zpin));
  // delay before next reading:
  delay(100);
}
```

Part C. Count/Time-Based Sensors

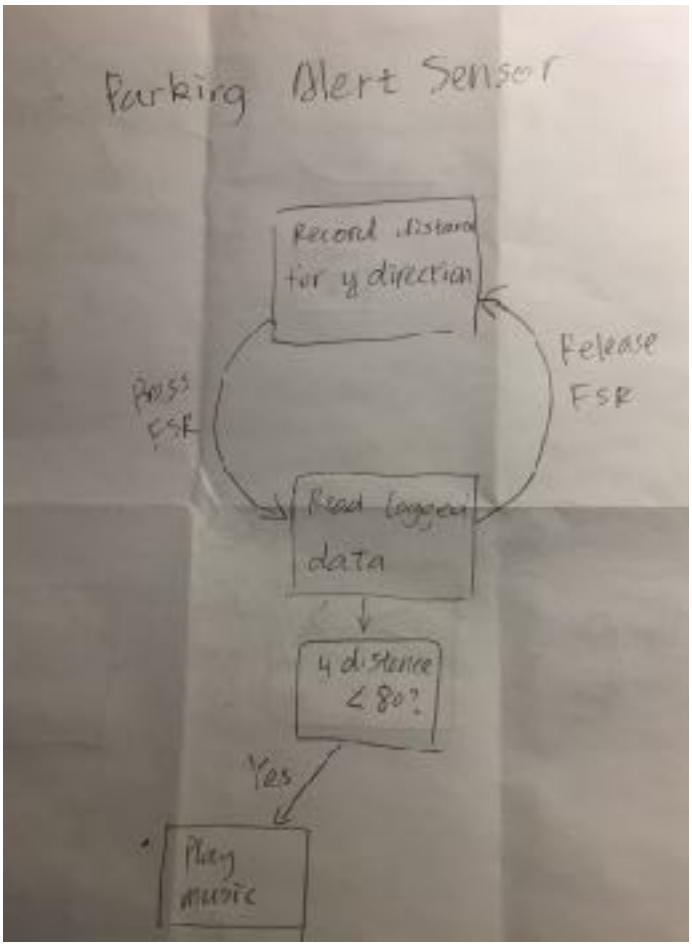
1. Rotary Encoder



Part D. Logging values to the EEPROM and reading them back

1. Design your logger

a. Turn in a copy of your final state diagram.



2. Reading and writing values to the Arduino EEPROM

a. How many byte-sized data samples can you store on the Atmega328?

1024 samples

b. How would you get your analog data from the ADC to be byte-sized?

Map 0-1023 to 0-255

4. Create your data logger!

a. Use the lab camera or your own camera/cell phone to record and upload a short demo video of your logger in action.

link: https://youtu.be/FBj6LLd4EGM

Description: We use a accelerometer to record the moving distance along the y-axis. When the user doesn't touch the FSR, it's in logging state and when the user press the FSR, the device switch to data-reading state. It will read the previously stored data and play music when the data is smaller than 80.

Code:

```
#include "pitches.h"
```

```
// include the library code:
#include <LiquidCrystal.h>
#include <EEPROM.h>

// initialize the library by associating any needed LCD interface pin
// with the arduino pin number it is connected to
const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

// notes in the melody:
int melody[] = {
  NOTE_C4, NOTE_G3, NOTE_G3, NOTE_C4, NOTE_G3, NOTE_G3
};

// note durations: 4 = quarter note, 8 = eighth note, etc.:
int noteDurations[] = {
  4, 8, 8,  4, 8, 8
};

// these constants describe the pins. They won't change:
const int groundpin = 18;            // analog input pin 4 -- ground
const int powerpin = 19;             // analog input pin 5 -- voltage
const int xpin = A3;                 // x-axis of the accelerometer
const int ypin = A2;                 // y-axis
const int zpin = A1;                 // z-axis (only on 3-axis models)

// start reading from the first byte (address 0) of the EEPROM
int address = 0;
byte value;
int counter = 0;

int sensorPin = A0;
int sensorValue = 0;

void setup() {
  // initialize the serial communications:
  Serial.begin(9600);
  lcd.begin(16, 2);
  pinMode(groundpin, OUTPUT);
  pinMode(powerpin, OUTPUT);
  digitalWrite(groundpin, LOW);
  digitalWrite(powerpin, HIGH);
}

void loop() {
  sensorValue = analogRead(sensorPin);
  if (sensorValue > 10) {
    // read a byte from the current address of the EEPROM
    value = EEPROM.read(address);
    lcd.clear();
    lcd.setCursor(0,1);
    lcd.print(value);
    lcd.setCursor(0,0);
    lcd.print("Printing distances");
```

```
  if (value < 80) {
    // iterate over the notes of the melody:
    for (int thisNote = 0; thisNote < 6; thisNote++) {
      // to calculate the note duration, take one second divided by the note type.
      //e.g. quarter note = 1000 / 4, eighth note = 1000/8, etc.
      int noteDuration = 1000 / noteDurations[thisNote];
      tone(8, melody[thisNote], noteDuration);
      // to distinguish the notes, set a minimum time between them.
      // the note's duration + 30% seems to work well:
      int pauseBetweenNotes = noteDuration * 1.30;
      delay(pauseBetweenNotes);
      // stop the tone playing:
      noTone(8);
    }
  }
  address = address + 1;
  if (address == EEPROM.length()) {
    address = 0;
  }
  delay(500);
} else {
    //write mode
    lcd.clear();
    lcd.setCursor(0,0);
    lcd.print("Writing distances");
    int val = analogRead(ypin) ;
    EEPROM.write(address, val);
    Serial.print(address);
    Serial.print("\t");
    Serial.print(val, DEC);
    Serial.println();
    address = address + 1;
    if (address == EEPROM.length()) {
      address = 0;
    }
    delay(5000);
  }
}
```