Authors:
NetIDs:
Date:
Project Name: Lab 5 Fall 21

# Description:
In this lab, you will work in a ** team ** of 3-4 people.

You will use the MPU 6050 accelerometer module and I2C serial communication protocol to read and print the measured X, Y, and Z data accelerometer from the module. You will also detect changes in either Y and Z axis rotation above a predetermined threshold level (~ 45 degrees for each axis). This threshold level will determine the state of the 8x8 LED matrix display and whether a buzzer alarm sounds off.

You will interface with the 8 x 8 LED matrix using the SPI communication protocol. While the MPU 6050 accelerometer is below the threshold level that you determine, you will display a smiley face on the 8 x 8 LED matrix. When the MPU 6050 accelerometer goes above the threshold level, the smiley face turns to a frowny face on the 8 x 8 led matrix display.

You will also implement a buzzer alarm that produces a chirping sound when the accelerometer sensor goes above a certain threshold value.

A push button switch will silence the piezo alarm.

# Instructions:
You will need to create a circuit using your breadboard, jumper wires, accelerometer, 8x8 led matrix, piezo speaker and a push button switch. There should be no special function registers used in the code for main(). Arduino libraries for Serial.println can be used for printing x,y, and z accelerometer data and also debugging. You will need to build your own timer function for delays when needed.

Read the MPU 6050 accelerometer datasheets. You will use the WHO_AM_I register (pg. 46 of register map) as the SLA. The register that you will be reading from are the ACCEL_nOUT registers (pg. 30 of register map). There are two 8-bit registers for X, Y, and Z. These registers will need to be read and combined to gather the necessary information. Finally, you need to obtain the power management register information (pg. 41 of register map). Before developing your threshold values, you should verify that data changes in all orthogonal x,y and z directions.

You will need to determine your accelerometer threshold tilt value (approximately 45degrees from rest position) based on actual readings that you obtain.

Read the 8x8 LED Matrix display datasheet (MAX7221 on how to interface the LED matrix using SPI protocol.

Use the passive piezo buzzer device. The alarm sound should be a chirping sound.

# Requirements
## Overall
1. The project must follow good coding practices and be well commented.
2. The breadboard wiring must be clearly laid out.

## main.cpp
1. Two state machines are used for debouncing the switch and for implementing the display of the 8x8 LED smiley or frowny face.
2. If the accelerometer movement reaches a defined threshold value, a piezo alarm will trigger and produce a chirping sound. Your group will determine the threshold value to trigger the piezo through experimentation.
3. Once the piezo alarm is triggered it will remain on until it is silenced by pressing a button switch.
4. The x,y and z data from the accelerometer will be read out in the while loop in main()
   - Read the datasheet and obtain necessary register numbers (hex)
   - Initialize I2C process
   - sei() <- make sure this is in there so that Serial.println works
   - Start the I2C transmission to the SLA
   - Write to power management -> write wakeup command (all zeros)
   - Inside the while(1)
     - Read from X high register -> Read from X low register
     - Read from Y high register -> Read from Y low register
     - Read from Z high register -> Read from Z low register
     - Combine registers and serial print information
     - Stop the I2C trans

## In a file called spi.cpp
1.All communication with the 8 x8 led matrix must be done over the SPI related pins.
2.Read the 8x8 MAX7219 datasheet for setting up SPI mode.

## timer.cpp
1.Implement a precise millisecond timer using timer 1 for switch debouncing.

## pwm.cpp
1.Use a PWM output signal to change the frequency of the piezo buzzer to generate a chirping sound.

## switch.cpp
1. Uses a switch to silence the audio chirping alarm.

## I2C Functions:
1. InitI2C()
   - Wake up I2C module on mega 2560
   - Set prescaler TWPS to 1
   - Set two wire interface bit rate register TWBR
   - Enable two wire interface

2. StartI2C_Trans(unsigned char SLA)
   - Clear TWINT, initiate start condition, initiate enable
   - Wait for completion

```
        -  Set two wire data register to the SLA + write bit
        -  Trigger action: Clear TWINT and initiate enable
        -  Wait for completion

3.  StopI2C_Trans()
        -  Trigger action + stop condition
        -  Write(unsigned char data)
        -  Set two wire data register equal to incoming data
        -  Trigger action
        -  Wait for completion

4.  Read_from(unsigned char SLA, unsigned Char MEMADDRESS)
        -  Start a transmission to the SLA
        -  Write to the MEMADDRESS
        -  Clear TWINT, initiate start condition, initiate enable
        -  Wait for completion
        -  Set two wire data register to the SLA + read bit
        -  Trigger action + master acknowledge bit
        -  Wait for completion
        -  Trigger action
        -  Wait for completion
        -  Trigger action + stop condition

5.  unsigned char Read_data()
        - Return TWDR
```