ECEN 5053 - Embedding Sensors and Actuators
Final Project: Laser Tracking System
12-13-2017

Goal: Our goal for this lab was to implement a laser tracker. We were to do this using closed loop control on a linear slider to follow a laser attached to an open loop controlled stepper motor. The stepper motor homed using an accelerometer.

Equipment and Parts:

- Cypress PSoC CY8CKIT050B development board
- Power Supply (2x)
- Step Motor
- Linear stage (Brush DC motor / Potentiometer / Belt)
- Accelerometer
- Laser
- Motor Driver IC (SN754410 Quadruple Half-H Driver) (2x)

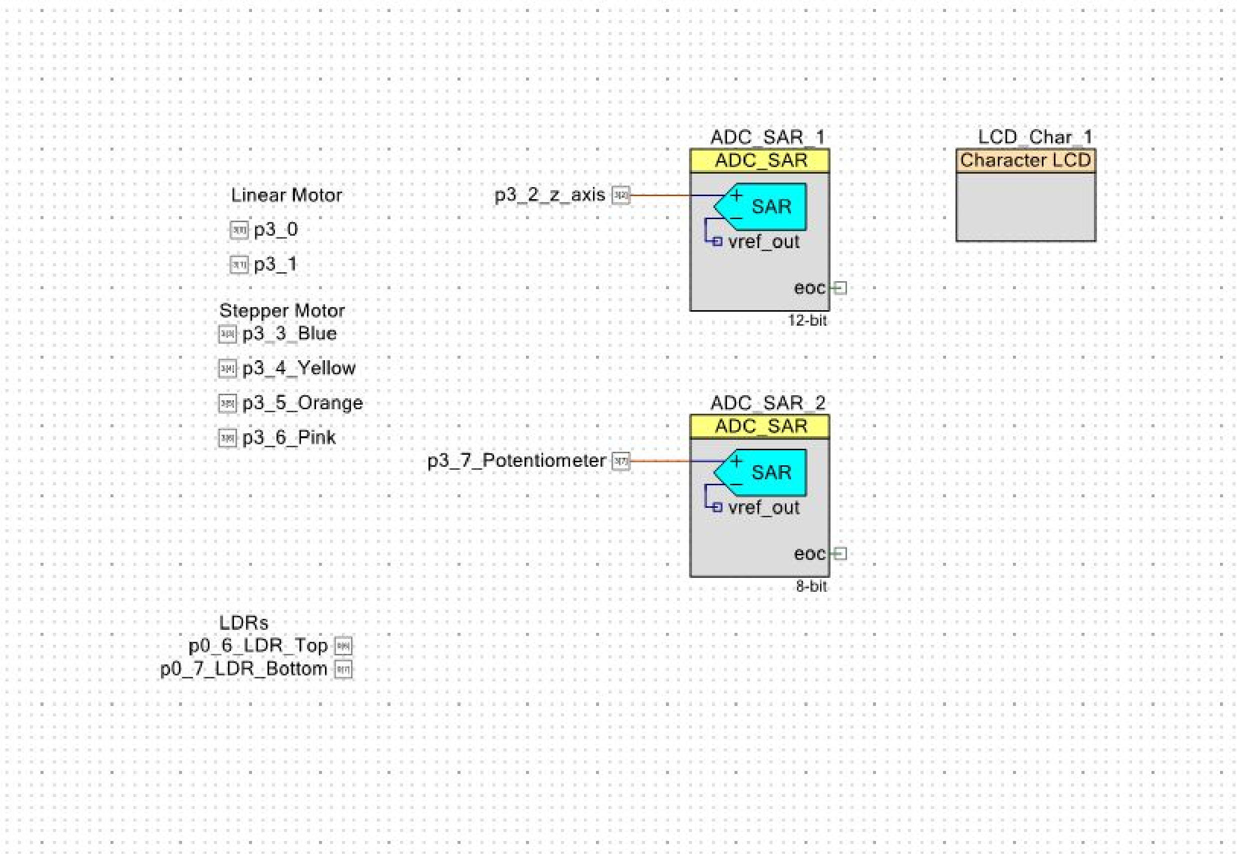

Figure 1: PSoC Schematic, including pins for stepper motor, ADC for accelerometer, ADC for Potentiometer, and Comparators for LDR Feedback

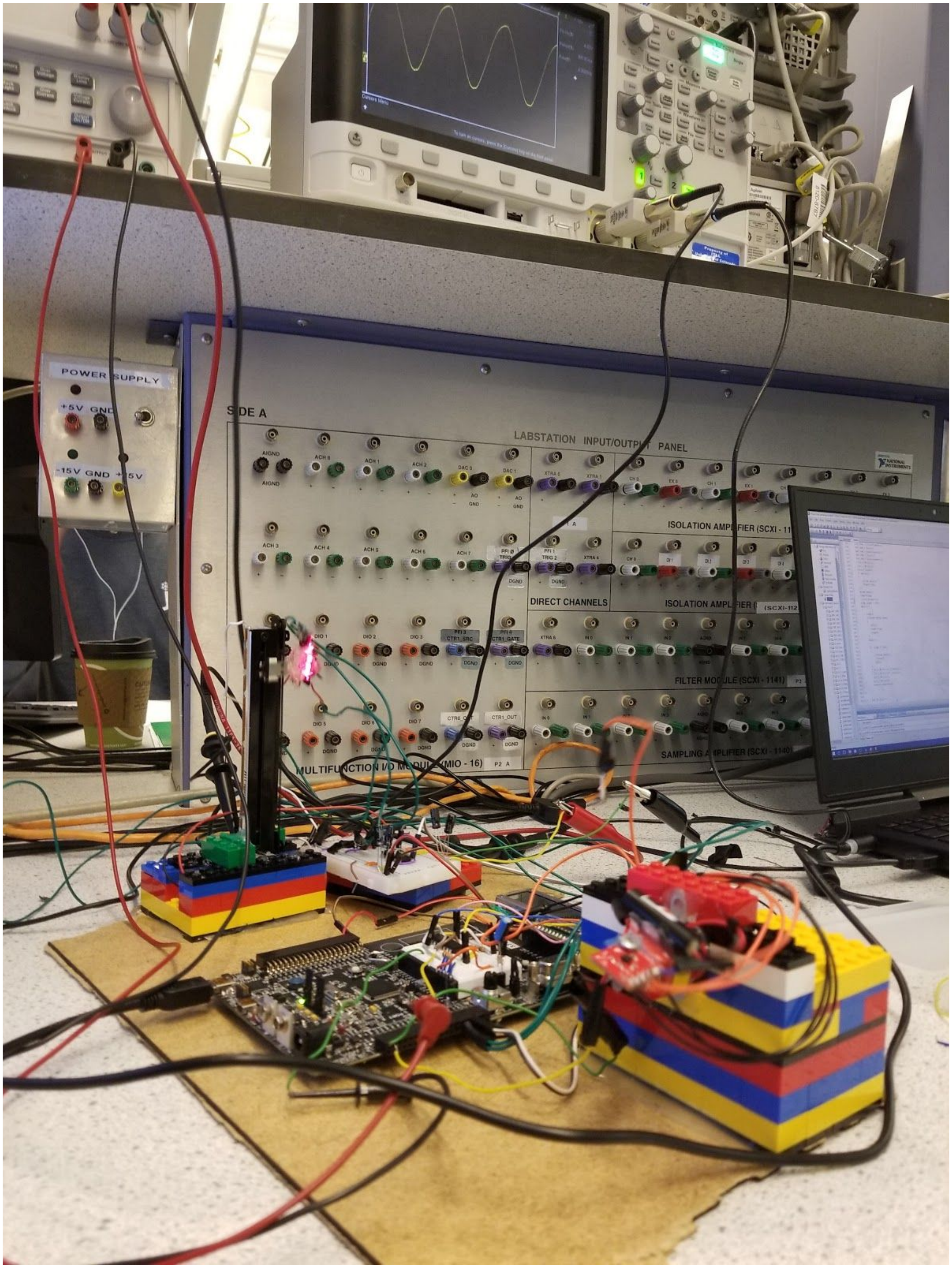

Figure 2: Physical setup including stepper motor, linear stage, and base

Methodology: This lab consisted of two major experiments. The first was to operate the stepper motor in such a manner that we could step to 30 degree, -30 degrees, and level on reset. We accomplished this by setting up the bipolar motor and using 2-dimensional C arrays to hold the value of a step. Each step would increment the current row of the array and set the coils to one phase or another. We operated the motor in a half step pattern: {A, AB, B, ~AB, ~A, ~A~B, ~B, A~B}. After moving the stepper motor, setup up hardware that allowed us to attach the stepper motor to the laser and the accelerometer. We also set up a base to hold the linear slider.

From here we set up code to control the linear stage. We used a state machine with six states to control the linear motor.

This allowed the motor to track the laser as it shifted across the LDRs across most of the range of the linear stage. We used feedback from the potentiometer to make sure the motor was not railing or drawing the maximum current trying to move past the bottom or top of the slider by reversing direction after it reached certain threshold voltages.

Finally, we integrated the system together onto a base of particle board that held the laser and motor aligned, as well as reducing the impact of vibrations from the stepper motor moving parts.

Observations:

1 - Stepper Motor Code:

```
uint8 counter_clockwise_table[8][4]={ {1,0,0,1},  {0,0,0,1},  {0,0,1,1},  {0,0,1,0},
{0,1,1,0},  {0,1,0,0},  {1,1,0,0},  {1,0,0,0}};//fast counter clockwise 8

void step(void)
{
    //CyDelay(2);
    //CyDelayUs(1860);
    CyDelayUs(800);
    static uint8 column=0,row=0;
    if(flag==1)
    {
        if(row==7)
        {
            row=0;
        }
        else
        {
            row++;
        }
    }
    else
    {
        if(row==0)
        {
            row=7;
        }
        else
        {
            row--;
        }
    }
    p3_3_Blue_Write(counter_clockwise_table[row][0]);
    p3_4_Yellow_Write(counter_clockwise_table[row][1]);
    p3_5_Orange_Write(counter_clockwise_table[row][2]);
    p3_6_Pink_Write(counter_clockwise_table[row][3]);
    return;
}
```

2- Question 1: What is the expected distance of the angle value if there are no errors?

Since we are using half stepping the expected angle of our motor is $(5.625°/16)/2 = 0.17578125$ which we round to 0.176 for convenience. This is the ratio based on a reduction gear ratio of 1:16 based on the data sheet.

3- Spreadsheet of Step Motor Angles

Measurement		y1	y2	y	Angle (rad)	Angle (deg)	Percent Error
1	78.5		6.25	-6.25			
2	78.5	6.25	6.5	-0.25	0.003184702609	0.1824700185	3.81%
3	78.5	6.5	6.75	-0.25	0.003184702609	0.1824700185	3.81%
4	78.5	6.75	6.875	-0.125	0.001592355342	0.09123524058	48.10%
5	78.5	6.875	7.25	-0.375	0.004777033726	0.2737038711	55.71%
6	78.5	7.25	7.5	-0.25	0.003184702609	0.1824700185	3.81%
7	78.5	7.5	7.75	-0.25	0.003184702609	0.1824700185	3.81%
8	78.5	7.75	7.875	-0.125	0.001592355342	0.09123524058	48.10%
9	78.5	7.875	8.125	-0.25	0.003184702609	0.1824700185	3.81%
10	78.5	8.125	8.25	-0.125	0.001592355342	0.09123524058	48.10%
11	78.5	8.5	8.625	-0.125	0.001592355342	0.09123524058	48.10%
					MIN	0.09123524058	3.81%
					MAX	0.2737038711	55.71%
					Average	0.1550994926	26.71%

4- Question #2: What are the minimum, maximum, and average step angles in degrees and percent error terms?

Based on the results from our 10 steps the minimum degree step is 0.0912, the maximum degree step was 2.737, and the average degree step was .155 degrees. In percent errors the minimum was 3.81%, the maximum was 55.71%, and the average was 26.71%.

5- Linear Slider Code

In order to get the linear slider to track the laser we set up a state machine with six states. These states were go up, go down, go up pause, go down pause, stop at top, stop at bottom. Each state reacted based on the position of the linear slider sensed by the potentiometer, as well as the status of the LDR's. For instance if the state was go up and the laser hit the bottom LDR it would assume the slider was moving too fast and enter the go up pause state. When the top LDR was reactivated it would then begin going up again, until it hit a threshold value on the potentiometer, set so that the dc motor would not rail the power supply.

```
void control_receiver(void)
{
    potentiometer_value=ADC_SAR_2_GetResult8();
    switch (mode_of_receiver)
    {
        case go_up:
        {
            p3_0_Write(0);
            p3_1_Write(1);
           if(p0_7_LDR_Bottom_Read()==0)
            {
                mode_of_receiver= go_up_pause;
            }
            else if(potentiometer_value >= 250)
            {
                mode_of_receiver= stop_at_top;
            }
            break;
        }
        case go_down:
        {
            p3_0_Write(1);
           if(p0_6_LDR_Top_Read()==0)
            {
                mode_of_receiver= go_down_pause;
            }
            else if(potentiometer_value <= 4)
            {
                mode_of_receiver= stop_at_bottom;
            }
            break;
        }
        case go_up_pause:
        {
            p3_0_Write(0);
            p3_1_Write(0);
            if(p0_6_LDR_Top_Read()==0)
            {
                mode_of_receiver= go_up;
            }
            else if(p0_7_LDR_Bottom_Read()==0)
            {
                mode_of_receiver= go_down;
            }

            else if(potentiometer_value >= 250)
            {
                mode_of_receiver= stop_at_top;
            }
            break;
        }
        case go_down_pause:
        {
            p3_0_Write(0);
            p3_1_Write(0);
            if(p0_7_LDR_Bottom_Read()==0)
            {
```

```
                mode_of_receiver= go_down;
            }
            else if(p0_6_LDR_Top_Read()==0)
            {
                mode_of_receiver= go_up;
            }
            else if(potentiometer_value <= 4)
            {
               mode_of_receiver= stop_at_bottom;
            }
            break;
        }
        case stop_at_top:
        {
            p3_0_Write(0);
            p3_1_Write(0);
            if(p0_7_LDR_Bottom_Read()==0)
            {
                mode_of_receiver= go_down;
            }
            break;
        }
        case stop_at_bottom:
        {
            p3_0_Write(0);
            p3_1_Write(0);
            if(p0_6_LDR_Top_Read()==0)
            {
                mode_of_receiver= go_up;
            }
            break;
        }
    }
}
```

6- Question #3: What does it mean to "crowbar" the power supply?

Crowbarring a power supply would be equivalent to shorting the power supply by dropping a crowbar across its terminal. In an H-Bridge this could occur if the current was not sent across the motor but rather across two terminal on the same side of the circuit leading power to travel directly to ground. To avoid this we made sure that the motor was on one side of the motor driver only.