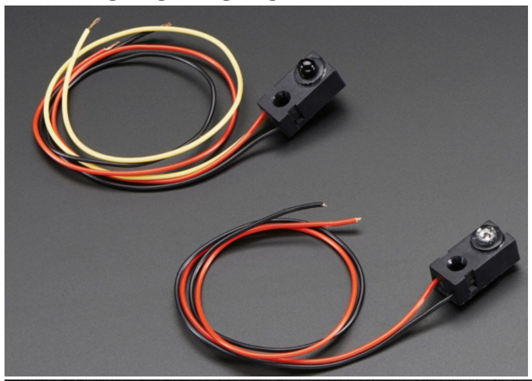
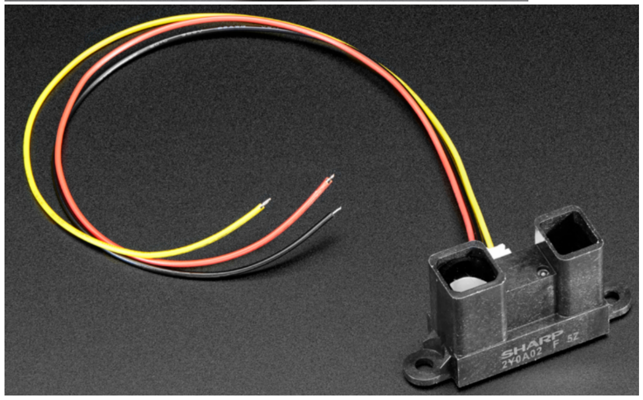
1. Planned and wrote the first draft of 8051 assembly code to control spin dude and LCD screen
    a. Arranging interrupt priority to ensure spin dude motor lags as little as possible
    b. Modularize LCD screen code to make writing simpler(i.e. lcall'ing a function to latch appropriate lines, etc)
2. Received and tested infrared distance and break beam sensors.
    a. First attempts at alignment of break beam sensors indicate possible extension of range if aligned with a laser.
        i. Data sheet in Chinese
        ii. Required pull-down resistor, 10k-15k value.
    b. Distance sensor worked as expected with analog voltage being mapped to distance from nearest object in line of sigh
        i. Minimum distance of 4-6 inches to collected meaningful data. Max at over 1.2 meters
    c. Wire gauge of both significantly lower than expected and thus adjusted by soldering higher gauge wire to each sensor's wires.
    d. 
    e. 

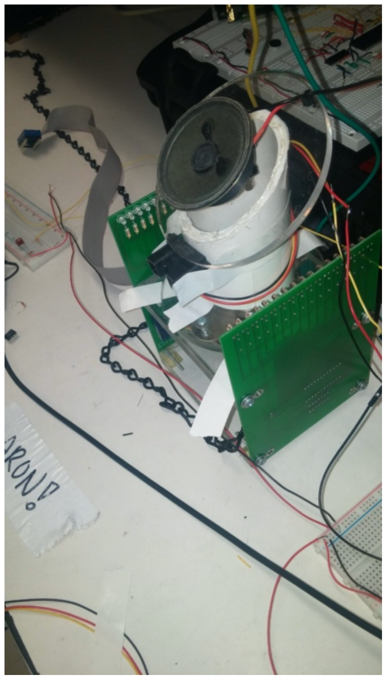
3. Created speaker platform
    a. 4 inch tall, 3 inch diameter PVC pipe, sanded at an angle
    b. 4 inch diameter laser cut acrylic disk hot glued to PVC pipe
    c. Speaker platform to be taped down on to spin dude such that the speaker rises above the photodiodes/resistors on the sides
    d. Speaker held in place against the acrylic disk via magnets on the other side inside of the PVC pipe
4. Got Bluetooth working (sort of)

    a. First attempt involved using HC-06 as a slave module, but I realized the module was designed to be used for serial communication, not audio data as I initially planned.
        i. First Back up plan was switch to using a BLE pioneer kit from cypress
            1. BLE also not capable of audio data transfer (the low energy is a byproduct of the module being off 80% of the time)
        ii. Second back up plan was to toggle between frequency tones in place of 'songs'
            1. Still unable to properly configure the HC-06 module/ view sent data bytes
                a. In hind sign there was activity on Tx line every time I sent data from my phone, so I should have been able to connect that pin to the Rx of a UART block in the PSoC and then access/manipulate the data
            2. Decided to use the BLE pioneer kit, however there is no standard profile of BLE serial communication
                a. Luckily the BLE development community is active and someone created a custom profile of BLE to serial communication similar to the SPP for Bluetooth
    b. After successfully connecting to the Bluetooth module it was a few lines of code to keep track of which value was being sent and using that to determine which frequency tone to play and how many times each tone was played and display that value on the PSoC's LCD screen.
        i. The main issue became keeping track of which tone/how many times via a couple flag variables

5. Distance sensor integration
    a. Attached the distance sensor to the PVC pipe to be in line with the direction that the speaker is pointing
    b. Photodiode/resistor panels on the side of the spin dude limit the range of acceptable motion since they begin to interfere with the distance sensor
        i. I expected 120-150 degrees of free motion, but I can only get ~90 since the distance sensor has a 10-degree scope of view.
    c. Desired behavior was having a lower voltage reading from sensor (further away) result in greater volume from the speaker (greater sine wave amplitude).
        i. Initial approach was to try to construct a voltage controlled amplifier via JFETs. [INSERT CIRCUIT DIAGRAM]
        ii. This caused loading and current issues so I switched to a digital approach
            1. Iterations with op amp buffers didn't help
        iii. Using an ADC to sample the output voltage of the sensor similar to our voltage sensor gave me a reading that I then mapped to amplification levels via a programmable gain amplifier (PGA) which took the un-gained sine wave as an input

            1. Range vs resolution came into play because the PSoC cannot output a sine wave with a Vpp > 4.08V as well as making sure the tone frequency doesn't get shifted too much by the PGA to ensure it stays in the range of human hearing. Also the PGA has discrete gain levels mapped to discrete chunks of ADC outputs in software so a continuous (or at least more finely controlled discrete levels) were not possible

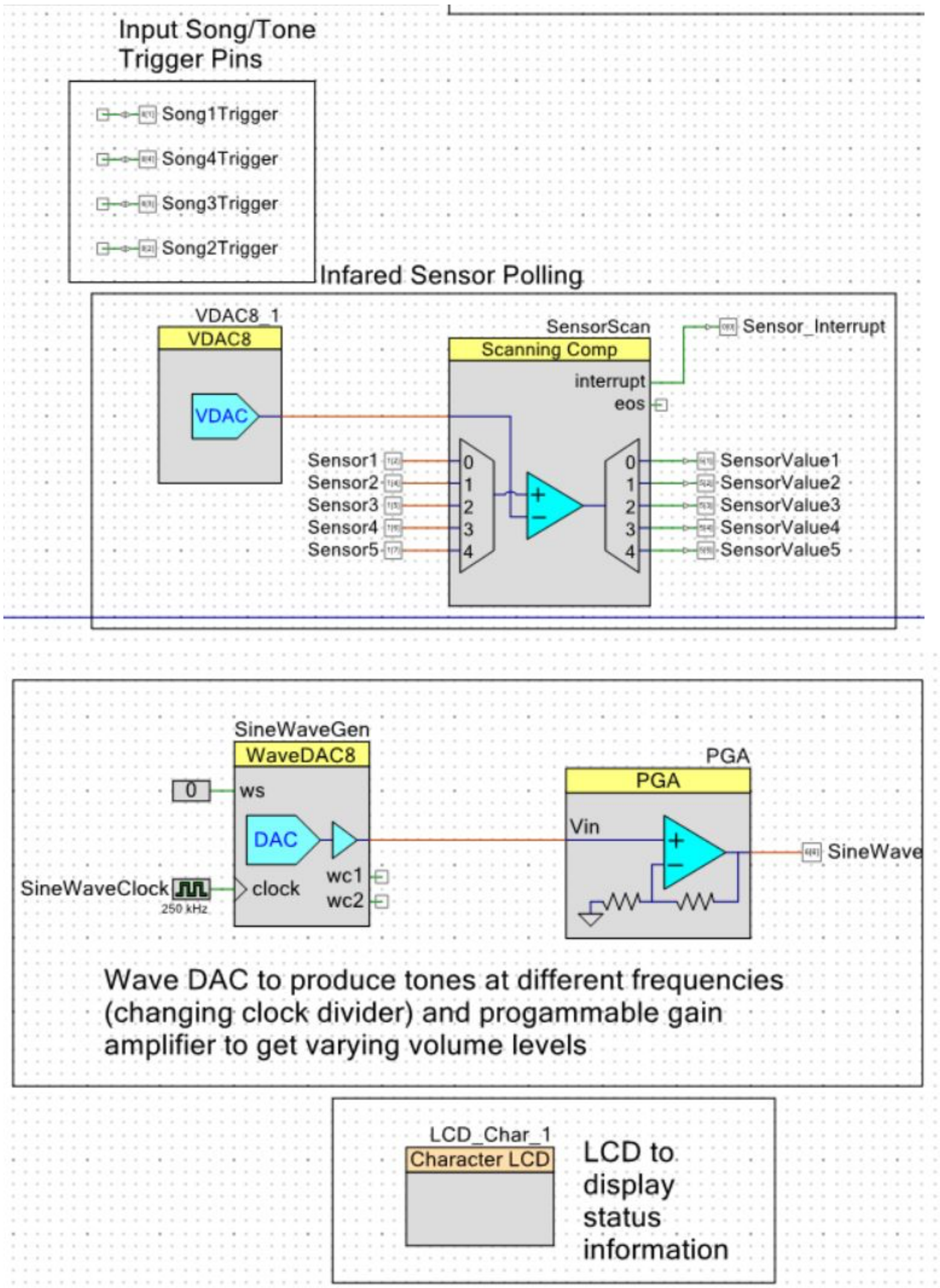
    d. Issue with using the ADC was the value wrapped around from 100 to 999 at about .75 meters, so conditional offsets were required.

    e. 

6. Break beam sensor integration
    a. Designed to work at 50 cm, but I attempted to have them function at close to 1.3m
        i. I initially thought alignment would be the main issue
            1. Used a laser and tape to help align properly
        ii. Voltage output dropped significantly even when properly aligned(<20mV)
            1. Lowered input voltage to receiver
            2. Put signal through low pass filter before reading into PSoC comparator railing the voltage high/low when threshold voltage was crossed
            3. Output of scanning comparator sent to 8051
                a. 5 inputs for the state of each sensor and a 6th input to signal that a change occurred in the state of at least once sensor
7. 8051/R31JP
    a. Chips and components used on R31JP DRAW CONNECTIONS
        i. LCD screen
            1. Display volume reading from output of ADC
        ii. SpinDude w/ stepper motor

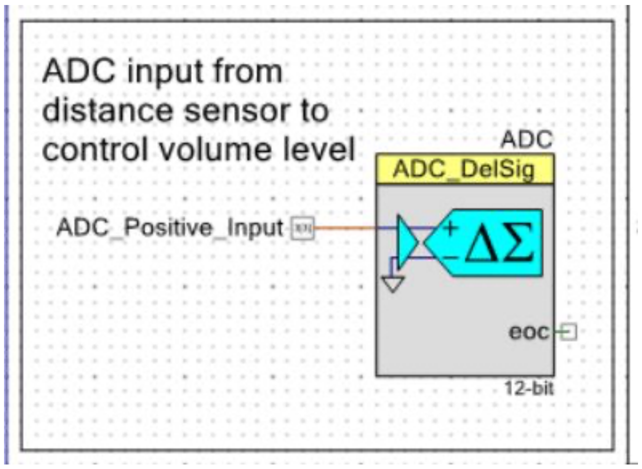
            1. Powered by 293s and rotates to face user
            2. Half stepping used to get finer resolution/control
            3. 7.5 degrees per step
        iii. LM18293
            1. Drive stepper motor on SpinDude
            2. Powered at 10 V from lab supply
            3. 4 inputs from lower nibble of port B of 8255
        iv. ADC0804
            1. Convert microphone voltage output to digital value to display on LCD
        v. Microphone
            1. Pick up audio output for speaker and send analog voltage to ADC
        vi. 8255
            1. Expandable Peripheral interface chip
            2. Port B&C output, Port A input
            3. Port C (and p3.5+p3.4) driving lcd
            4. Lower nibble of port B driving 293s which drive SpinDude's stepper motor
        vii. 74LS138
            1. De-multiplexer
            2. Control 8255 and ADC which are both connected to the data bus
        viii. LM358(on PSoC board)
            1. Active low pass filter before speaker
        ix. 74LS14
            1. Schmitt Inverter
                a. Buffer and invert interrupt from PSoC

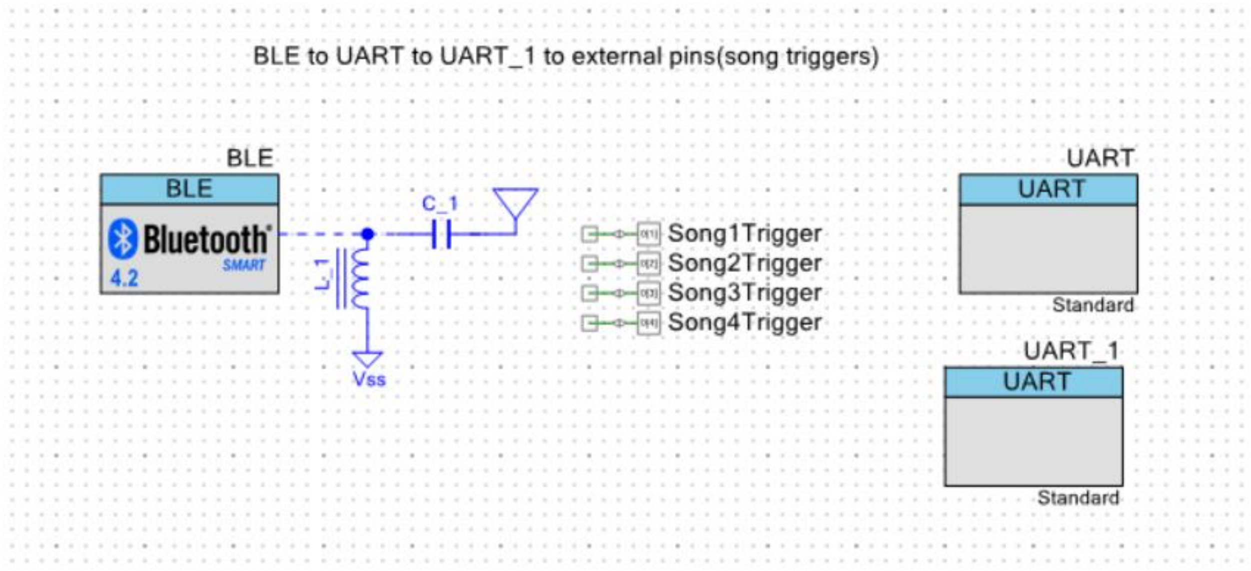
8. Powering Components
    a. Distance sensors powered by 5V rail of PSoC
    b. Break beam sensors powered by DC power supply at ~1V
    c. Stepper motor powered by DC power supply at ~10V
    d. Everything else powered from R31JP with 5V rail
9. Physical set up
    a. Dedicated breadboards for signal filtering and sensor power rails
    b. Stepper motor in the center connected to R31JP
    c. PSoC with LCD screen with song information
    d. 6 input signals from PSoC to R31JP
    e. Smaller LCD screen on the right
10. Position triangulation to stepper motor command algorithm
    a. Based on which sensor(s) is(are) triggered the motor command varies to point toward the user
11. Continuous vs discrete
    a. Continuous was just not possible without significantly more analog portions



Wave DAC to produce tones at different frequencies (changing clock divider) and progammable gain amplifier to get varying volume levels

LCD to display status information


ADC input from
distance sensor to
control volume level
ADC
ADC_DelSig
ADC_Positive_Input
eoc
12-bit


BLE to UART to UART_1 to external pins(song triggers)
BLE
BLE
Bluetooth
SMART
4.2
C_1
L_1
Vss
Song1Trigger
Song2Trigger
Song3Trigger
Song4Trigger
UART
UART
Standard
UART_1
UART
Standard