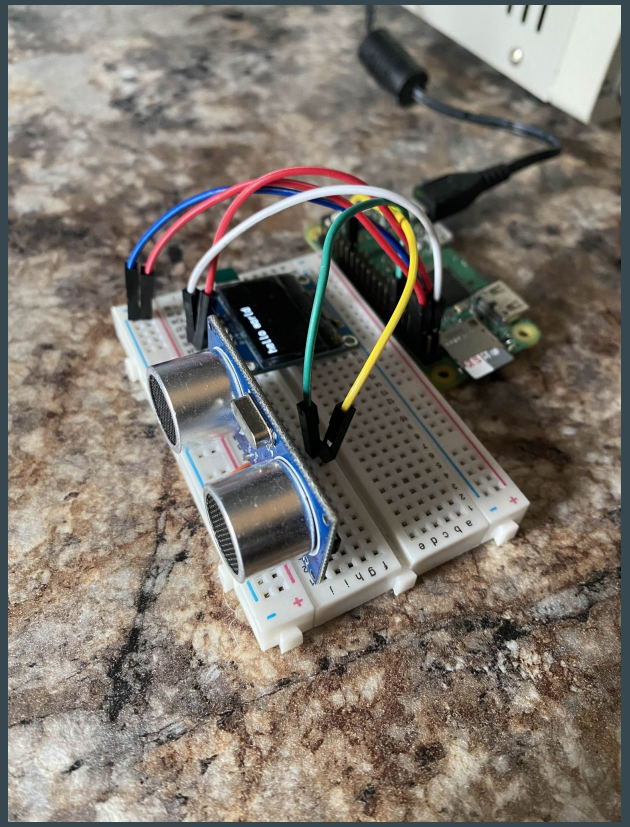
Ultrasonic Range Finder

Siddhartha Bajracharya & Joshua Wilbur

Project Overview

- A short-distance (<4m) rangefinder.
- Ultrasonic sensor uses 40KHz waves to detect distance.
- An OLED display outputs the distance along with other useful metrics.
- Either using PWM or manually triggering the sensor continuously.



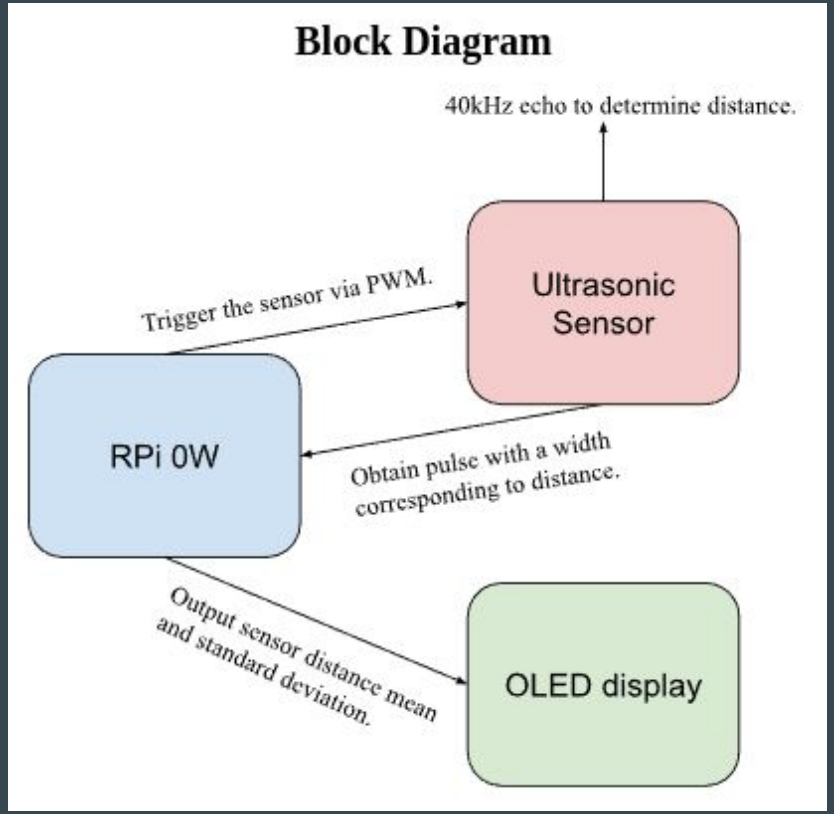
Design & Block Diagram

Hardware:

- Raspberry Pi 0W
- SSD1306 OLED Display
- HC-SR04 ultrasonic sensor module
- USB Power Supply

Software :

- Using the C language to talk with hardware.
- Raspbian OS is running on the Pi.
- Display library used, more on slide 8.

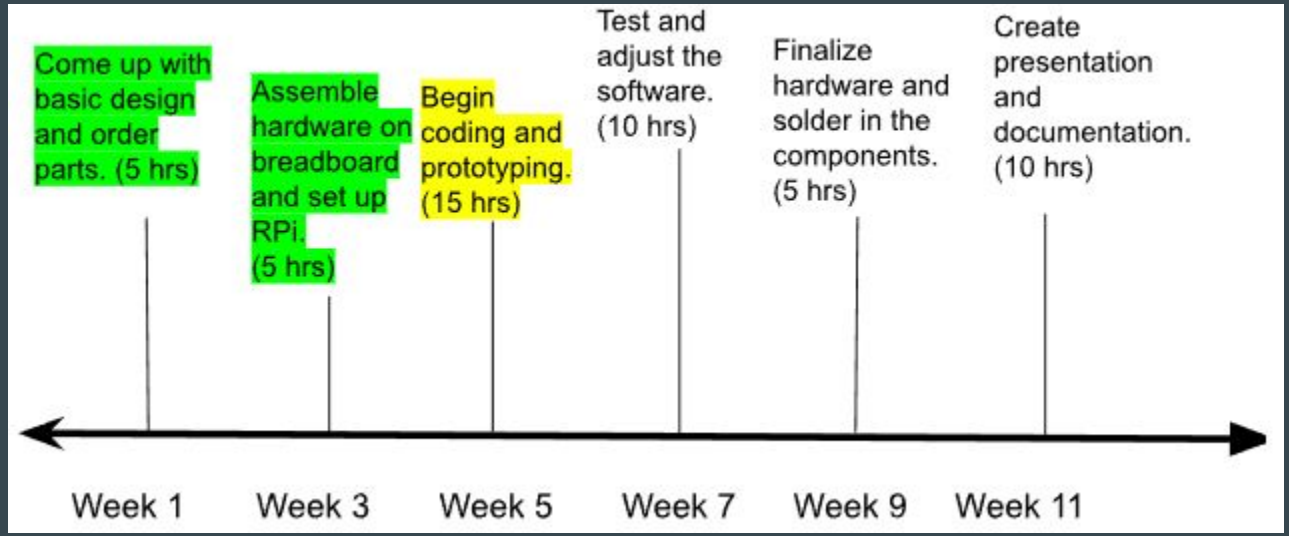


Bill of Materials

Component	Part Number	Unit Price	Supplier
Raspberry Pi Zero W	3400	$15.00	Adafruit
OLED Display	SSD1306	$5.50	Digikey
Ultrasonic sensor module	HC-SR04	$4.50	Sparkfun
USB Power Cable	N/A	$0.00	Hackerspace
Jumper Wires	N/A	$0.00	The team
Solder Board	N/A	$0.00	The team

Current Project Timeline

- Note: It is week 6 of the semester.



Project Progress

What has been completed?

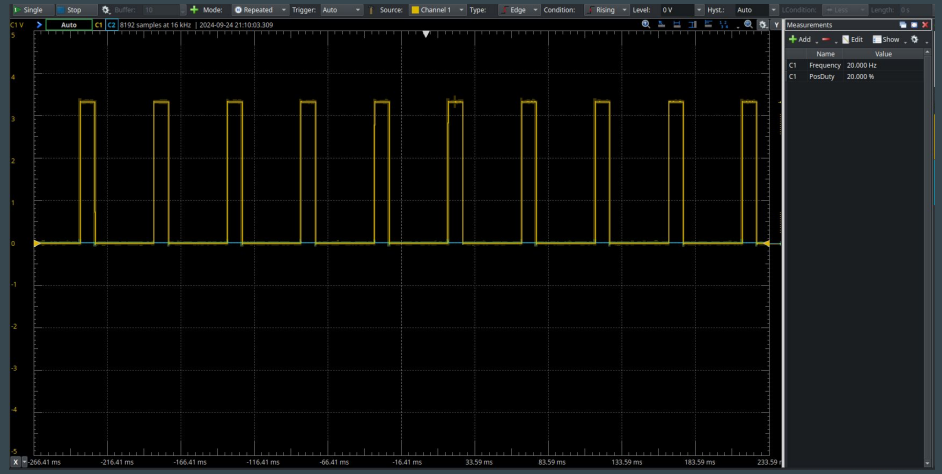
- Basic design and proposal
- Acquired and attached hardware.
- PWM function.
- OLED display code.
- Initialize the Ultrasonic Sensor.
- Basic distance/pulse reading.

What is left?

- Integrate our code together.
- Get a fully working prototype.
- Testing to fine-tune software.
- Solder the components to perfboard.
- 3D print a case, if time avails.

Pulse Width Modulation (PWM) on the Raspberry Pi

- PWM used to trigger ultrasonic sensor.
- Easy setup on Pi compared to STM.
- Wrote values to 4 configuration files.
- Period and duty cycle are integers in nanoseconds.
- PWM set to 20 Hz at 20% positive duty.



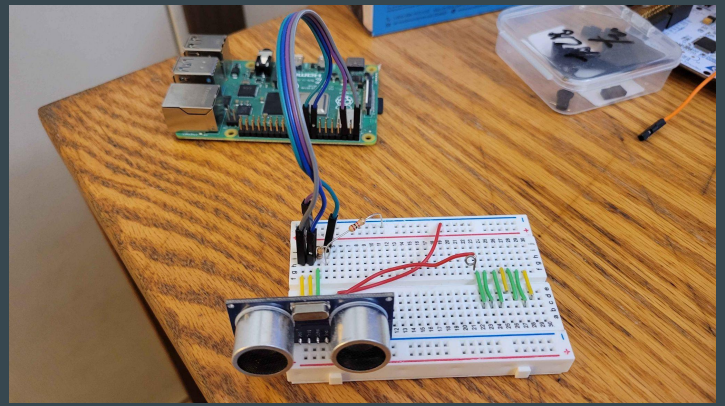

```
// File paths for PWM on my RPi 0W
write_PWM_file("/sys/class/pwm/pwmchip0/export", 0); // Enable PWM channel 0
write_PWM_file("/sys/class/pwm/pwmchip0/pwm0/period", period);
write_PWM_file("/sys/class/pwm/pwmchip0/pwm0/duty_cycle", duty);
write_PWM_file("/sys/class/pwm/pwmchip0/pwm0/enable", 1);
```

Display Library and Setup

- Display library is from GitHub, credit to nopnop2002.
- It is very similar to the one from 271, provides only basic text and shapes.
- Strangely, it works via outputting command line arguments given to a binary file.
- Wrote a program to fork a new process and execute the OLED binary.

Setting up the Ultrasonic Sensor (HC-SR04)

- Found some simple tutorials for it.
- Has Four pins. (VDD, TRIG, ECHO, GND)
- It takes in 5V and the ECHO pin outputs 5V as well
- Used a voltage divider.
- Everything else was basic wiring.



Code for the Ultrasonic Sensor

- Found some basic python code for it (The Pi Hut)
- Translated it to C and added libraries for the GPIO initialization. (pigpio)
- Initially ran the python code but only outputted 0.93cm.
- Ran the translated C code and it had the same output.
- Replaced the sensor the next day and it worked!

Problems Encountered

- Issues writing to PWM configuration files, solved with const *char pathing type.
- Have to run program as root user due to kernel level functions being accessed.
- A defective ultrasonic sensor.
- Getting the PWM code to work with the ultrasonic sensor code.
- Could just stick with manual triggering.

Thank you! Questions?