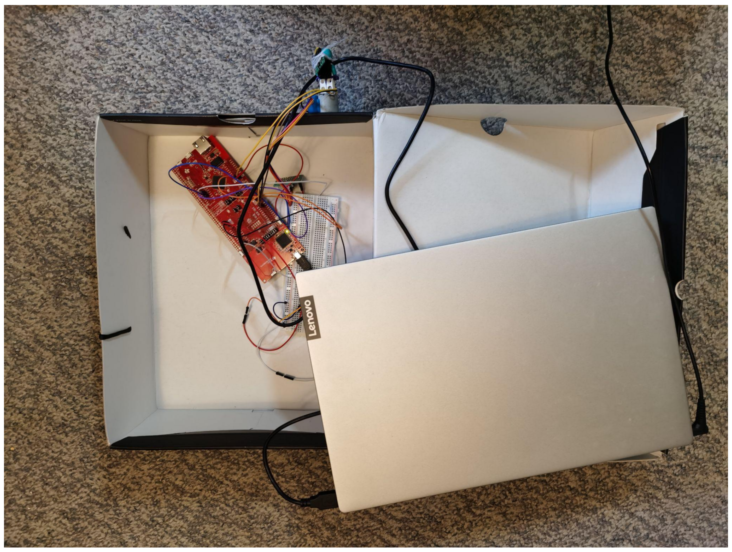
2DX4: Microprocessor System
Final Project

Instructor: Drs.Boursalie, Doyle, Haddara

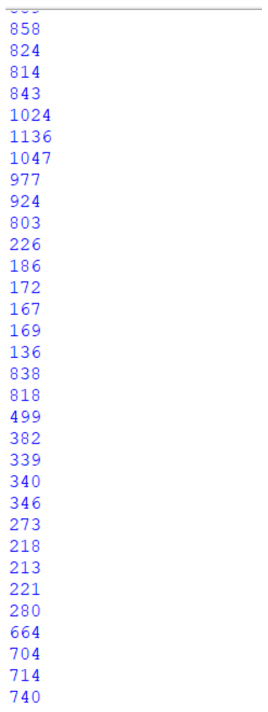
Junbo(Frank) Wang – wangj430 – 400249823 –

2DX4 – Monday Afternoon – L01

1. Device Overview



Entire project system



```
858
824
814
843
1024
1136
1047
977
924
803
226
186
172
167
169
136
838
818
499
382
339
340
346
273
218
213
221
280
664
704
714
740
```

Data displayed on PC

Features

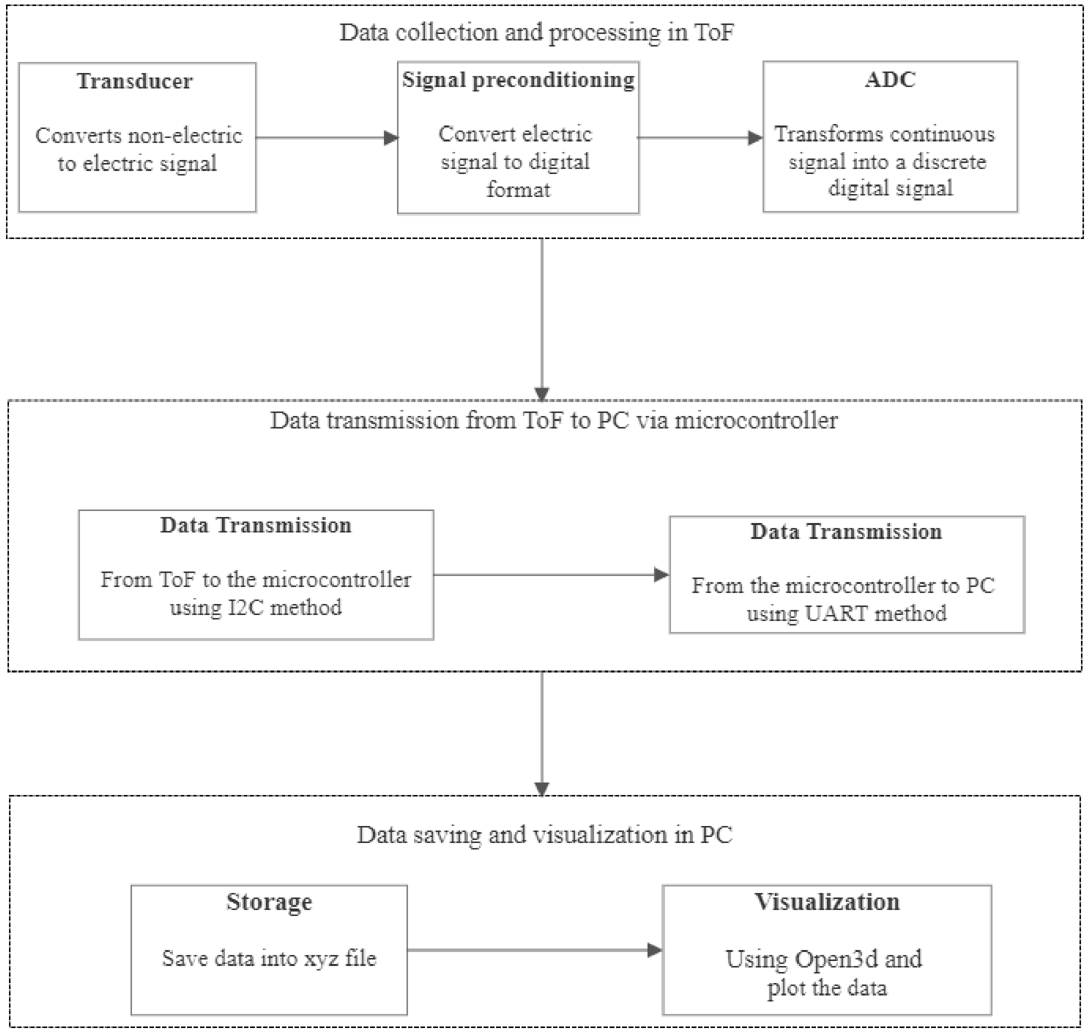
- Hardwares:
    - MSP432E401Y SimpleLink™ Ethernet microcontroller with 120-MHz Arm® Cortex®-M4F Processor Core with Floating-Point Unit (FPU).
    - ULN2003 Stepper Motor
    - Time-of-Flight, laser-ranging sensor VL53L1X
- 12MHz bus speed.
- Operating Voltage
    - 5-12V for the motor
    - 2.6-3.5V for ToF sensor
- Memories
    - 1024 KB of on-chip flash memory
    - 256 KB of bit-banded SRAM
    - 6 KB of EEPROM
    - Internal ROM
- ADC
    - 12-bit precision ADC with 24 shared analog input channels
    - Maximum sample rate of two million samples/second
- The instruction setting is less expensive and requires less memory.
- Programming language
    - C language creates environments and programming microcontrollers.

    - Python for sending, processing and visualizing data.
- Serial communication
    - UART between microcontroller and PC
    - I2C between ToF and MCU
    - Baud rate of 115.2 kbps and stop bit of 1

General description

The ToF sensor is an important part of the system because it obtains distance data by measuring the travel time of the light it emits. The data is processed by the ADC and converted to a discrete digital signal. The data is then transmitted to the microcontroller through the I2C communication method. After that, the data is sent to the PC via UART and displayed on the computer. Finally, the data is stored into an xyz file using the Serial library in Python, which is drawn using the Open3D library.

Block diagram


Data collection and processing in ToF
Transducer
Converts non-electric to electric signal
Signal preconditioning
Convert electric signal to digital format
ADC
Transforms continuous signal into a discrete digital signal
Data transmission from ToF to PC via microcontroller
Data Transmission
From ToF to the microcontroller using I2C method
Data Transmission
From the microcontroller to PC using UART method
Data saving and visualization in PC
Storage
Save data into xyz file
Visualization
Using Open3d and plot the data

2. Device Characteristic Table

Device	**Details**
Pins	PM[0:3], PB[2:3], PF[1,4], PN[0:1]
Bus speed	12MHz
45 degree assigned LED	D3 (PF4)
Motor voltage	5V
Motor input pins	PM[0:3]
Push button voltage	3.3V
Push button resistor	10KΩ
Push button input pin	PF1
ToF voltage	3.3V
ToF SCL pin	PB2
ToF SDA pin	PB3
Serial communication from the ToF to the microcontroller	I2C
Serial communication from the microcontroller to the PC	UART
Baud rate	115200 bps
Stop bit	1
Parity bit	None
Python data storage	Serial
Python visualization	Open3d

3. Detailed Description

Distance Measurement

In distance measurement, there are three main steps to complete the whole process: data collection, data transmission, data conversion, and data storage. At the beginning, load the codes into the microcontroller and push the input button to complete the initialization. The ToF sensor uses a laser to do the distance measurements, which emits light into the surrounding area. After the laser reaches the objects, it bounces back to the sensor and calculates the distance by counting how long the light travels. After each 360-degree rotation, I move the project 30–40 cm to let the ToF sensor detect the whole area. During this process, the signals are converted to digital form and then use ADC to discretize the signals.

Then, the data is transmitted from the ToF sensor to the microcontroller by using the I2C communication method and is kept in the onboard memory. After that, the data is sent to the computer by using the UART communication method. With a stop bit of 1, the data is transferred 8 bits at a time. Serial communication is carried out at a rate of 115200 bps. All the data is transmitted into a list on a PC using Python.

After that, set several values as parameters to be calculated in Python. Here are some parameters and formulas written in Python code:

delta: movement in each rotation
data_rotation: sampling time
num_rotation: number of rotation
idx / i: each element in the list
alpha: angle between y-axis and current position)

alpha = 360/data_rotation * pi/180 * (idx+1) - in clockwise direction
alpha = - (360/data_rotation) * pi/180 * (idx+1) - in counterclockwise direction

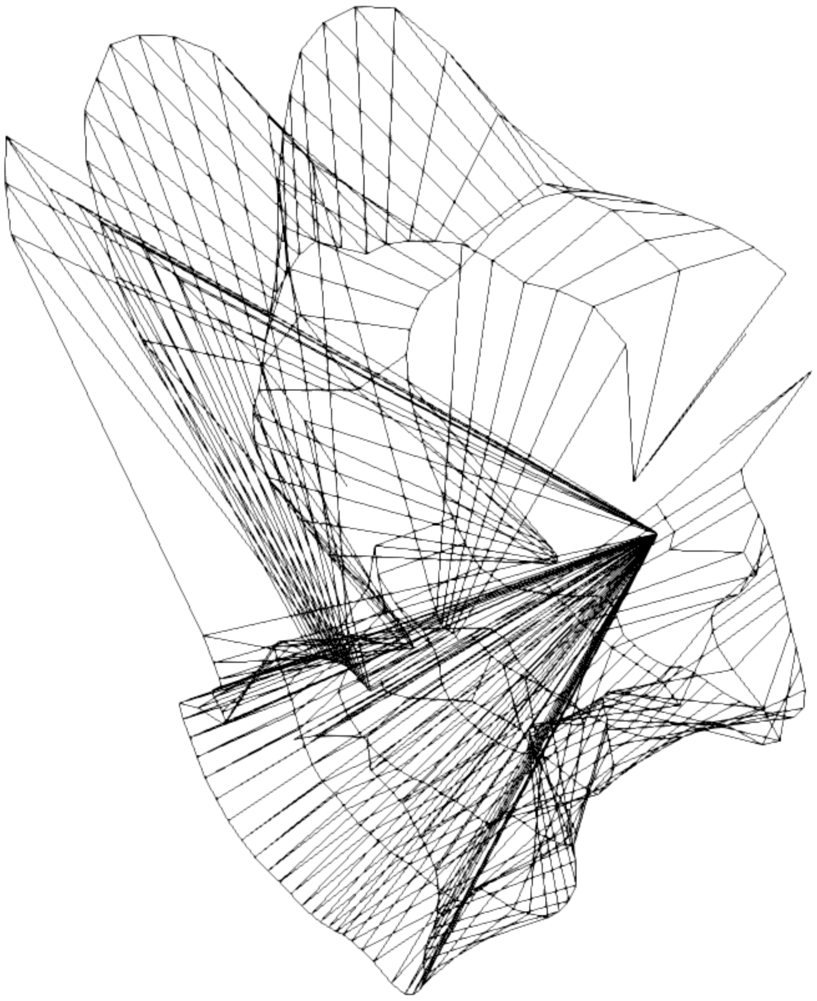
x coordinates: x = (idx // data_rotation) * delta - // is integer division in Python
y coordinates: y = - i * sin(alpha) - trigonometry calculation
z coordinates: z = i * cos(alpha) - trigonometry calculation

Finally, the list containing the distance values is scanned after all the data has been transmitted. All the integer distance values are the x, y, and z coordinates, which are computed through trigonometry, and that data will be stored in an xyz file in the format of x, y, and z coordinates.

Visualization

In this project, all my work is done in Windows. The Python version I used is Python 3.8.7, which can import the Open3d library and do the visualization.

After all the data is stored in xyz files, I import the Open3d and numpy libraries into Python. Python reads the xyz file and creates point clouds of all the data by using the o3d.io.read_point_cloud() function. The point clouds are saved in the list which is generated by the numpy library. Following that, the adjacent vertices are connected in each plane to form indexes. Also, those vertices are linked between the adjacent planes. Therefore, the plot is generated by using the o3d.visualization.draw_geometries function.



Plot of my room

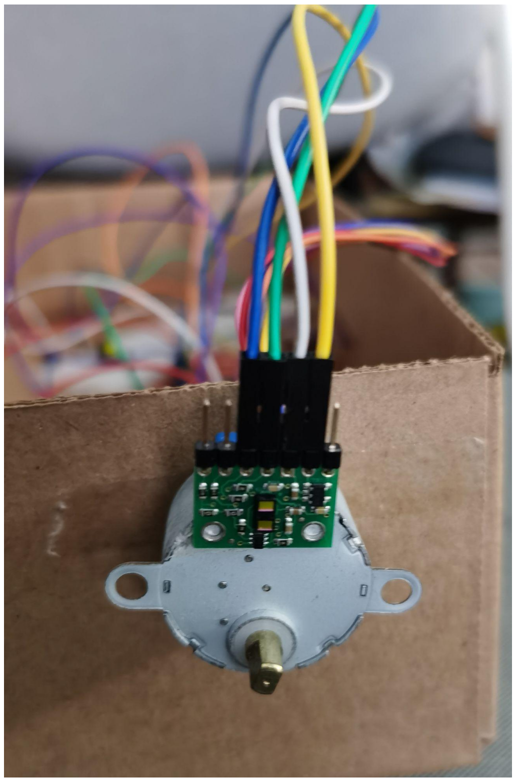
4&5. Application Example and User Guide

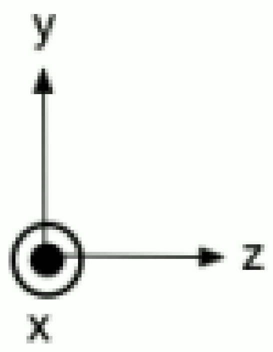
User Guidance

1. Connect the pins and setup the push buttons correctly
2. Attach the motor and ToF sensor in a proper place, so it can rotate smoothly.
3. Set the correct port of PC, Baud rate, and file path in the writedata.py file.
   s = serial.Serial("COM5", 115200)
   filename = "C:\Project_final\\ToF_laser.xyz"

4. Change the file path in the plot.py file.
   filename = "C:\Project_final\\ToF_laser.xyz"
5. Load the code in the Keil into the microcontroller and wait until LEDs stop flashing.
6. Press the push button to start the measurement.
7. Run the writedata.py file.
8. Watch the motor spin and the data is sent and displayed on the computer.
9. Move the project 30-40 cm along the x-axis in every 360-degree rotation.
10. Run plot.py file and generate the 3D plot.

According to the project, I want the sensor to measure in a vertical plane, which refers to the yz plane. As the item moves, it will follow the signal axis. Connect the motors and sensors to the plane of the box, assuming this plane is in the yz plane. When I move the box 30–40 cm per 360-degree rotation, the sensor will measure the distance along the x axis.





6. Limitations

1. Summarize any limitations of the microcontroller floating point capability and use of trigonometric functions?

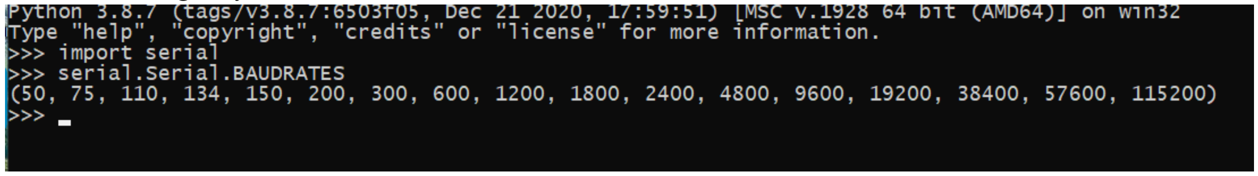
The microcontroller's floating point unit provides 32-bit instructions for single data processing. On the other hand, the ToF's data is the uint16_t type, which indicates that all values are integers and has less effect on range error. The data is transmitted to the PC, where it is processed with Python and the coordinates are determined using trigonometry. However, the ToF sensor's range error will affect the accuracy. In general, Python replaces the microcontroller for floating points processing and trigonometry calculations.

2. Calculate your maximum quantization error for each of the ToF modules?

Maximum quantization error = 3.3V / 2^12 = 8.057*10^-4

3. What is the maximum standard serial communication rate you can implement with the PC? How did you verify?

The PC's maximum standard serial communication rate is 115200 bits per second. It can be verified through Python.



```
Python 3.8.7 (tags/v3.8.7:6503f05, Dec 21 2020, 17:59:51) [MSC v.1928 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import serial
>>> serial.Serial.BAUDRATES
(50, 75, 110, 134, 150, 200, 300, 600, 1200, 1800, 2400, 4800, 9600, 19200, 38400, 57600, 115200)
>>> 
```

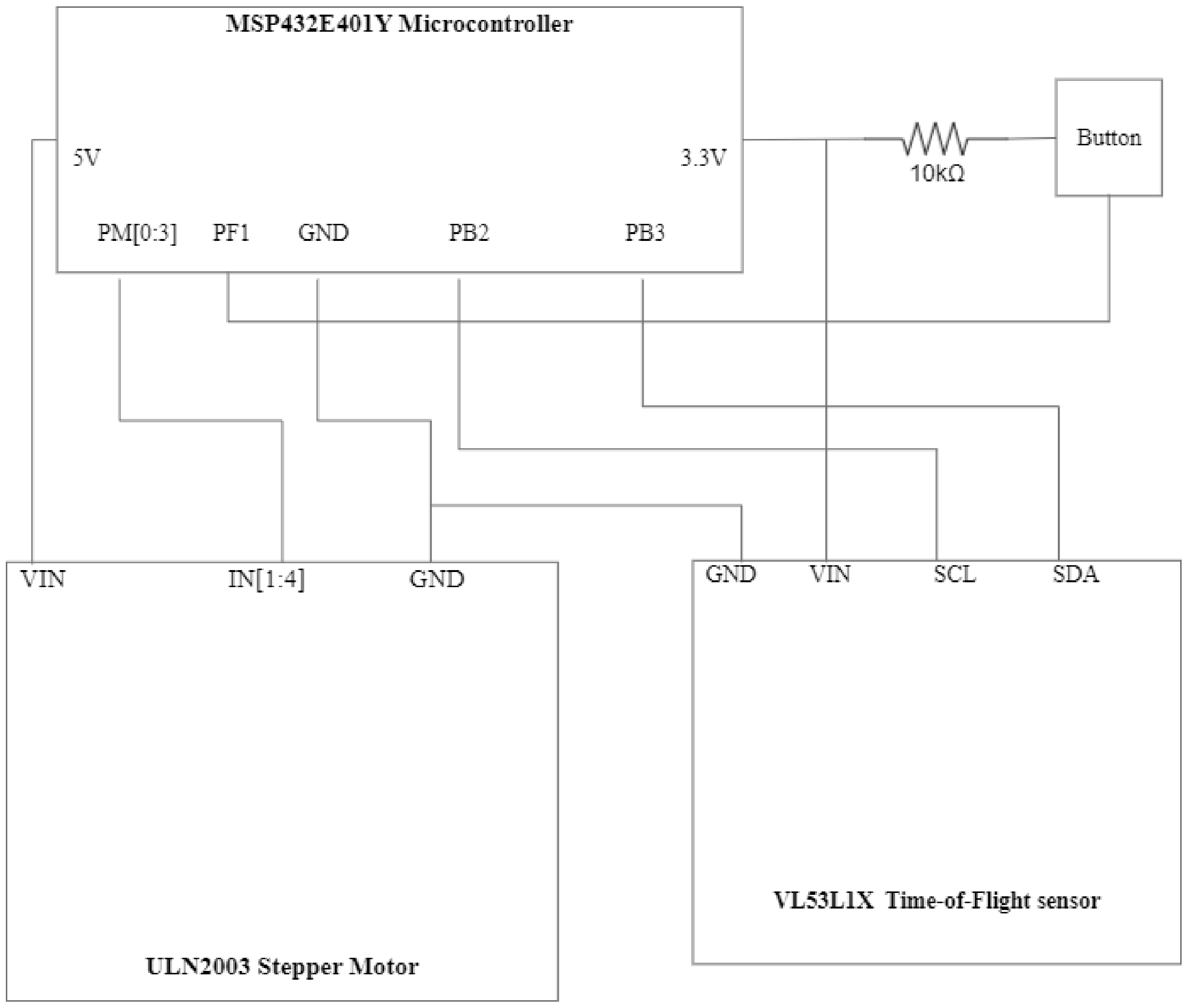
4. What were the communication method(s) and speed used between the microcontroller and the ToF modules?

The communication method is I2C. The binary value in Keil code is 0b00000000000000101000000000111011, which is close to 163899 bps. Therefore, the speed used between the microcontroller and the ToF modules is close to 164 kbps.

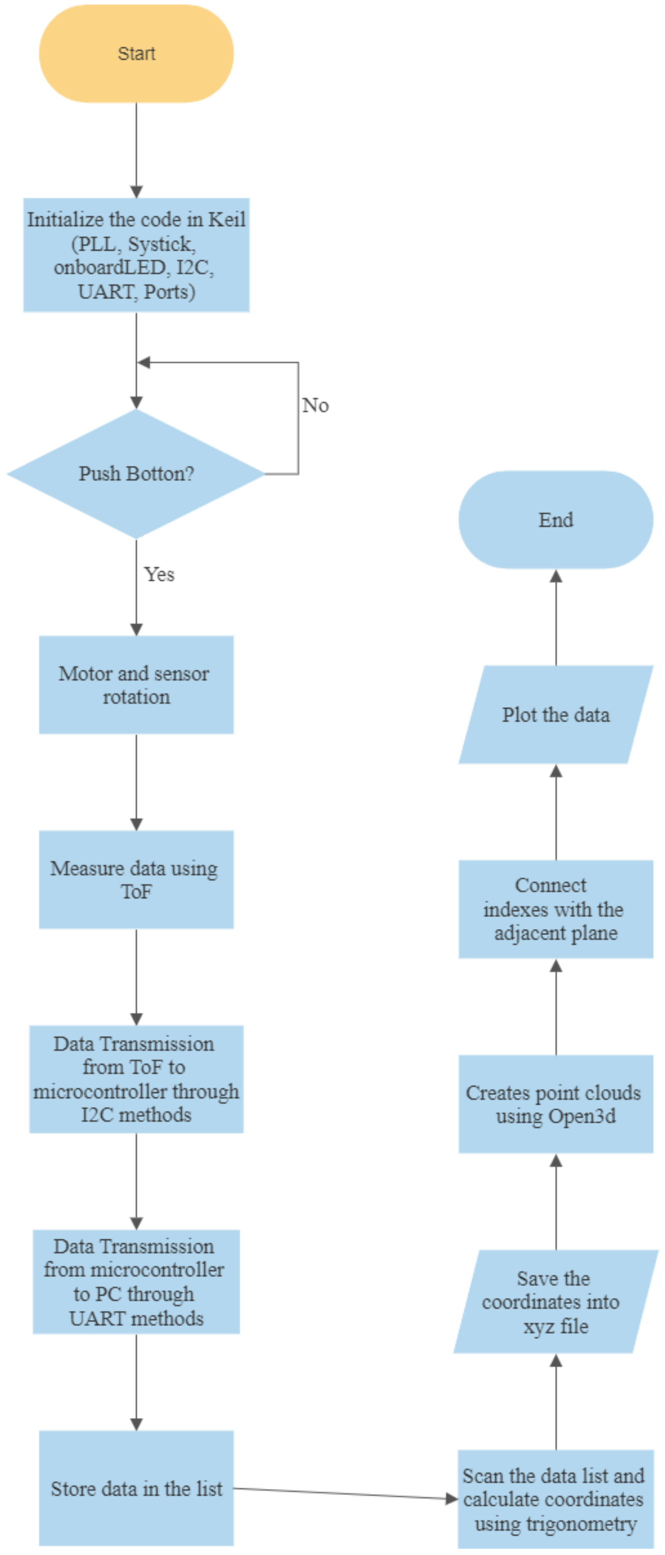
5. Reviewing the entire system, which element is the primary limitation on speed? How did you test this?

According to the VL53L1X Datasheet, the maximum speed for the I2C communication method is 400 kbps. It is faster than the UART communication, which is 115.2 kbps. As a result, the UART connection between the microcontroller and the PC is the primary speed limiter of the two communication methods.

7. Circuit Schematic



8. Flowcharts


Start
Initialize the code in Keil (PLL, Systick, onboardLED, I2C, UART, Ports)
Push Botton?
No
Yes
Motor and sensor rotation
Measure data using ToF
Data Transmission from ToF to microcontroller through I2C methods
Data Transmission from microcontroller to PC through UART methods
Store data in the list
Scan the data list and calculate coordinates using trigonometry
Save the coordinates into xyz file
Creates point clouds using Open3d
Connect indexes with the adjacent plane
Plot the data
End