2DX4: Microprocessor Systems Project
Final Project

Instructor: Drs. Bruce, Haddara, Hranilovic, and Shirani

Xinyu Chen–L09

Project Demo:
https://drive.google.com/file/d/1Vlz_Qf- y0-Y41gM3tREe60K1Oog4AV/view?usp=sharing

Question 1 Response:
https://drive.google.com/file/d/1GrKnC-rB26cyQsAUr3lvwrB-frBxlTiv/view?usp=sharing

Question 2 Response:
https://drive.google.com/file/d/1DbKD816y87e64cBwgsS1P8o9XpioaIfN/view?usp=sharing

Question 3 Response:
https://drive.google.com/file/d/1l_o1ExsU1PRh99r5ur0M4z60yn_h7PdF/view?usp=sharing

Overview

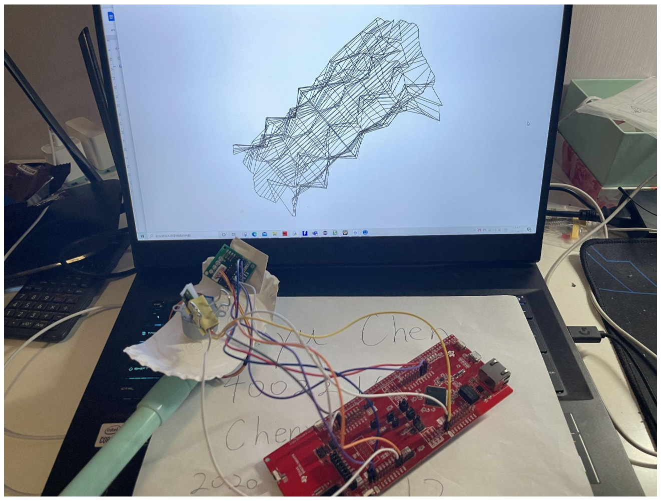


Figure 2.1 The lidar device with the PC and visualization display

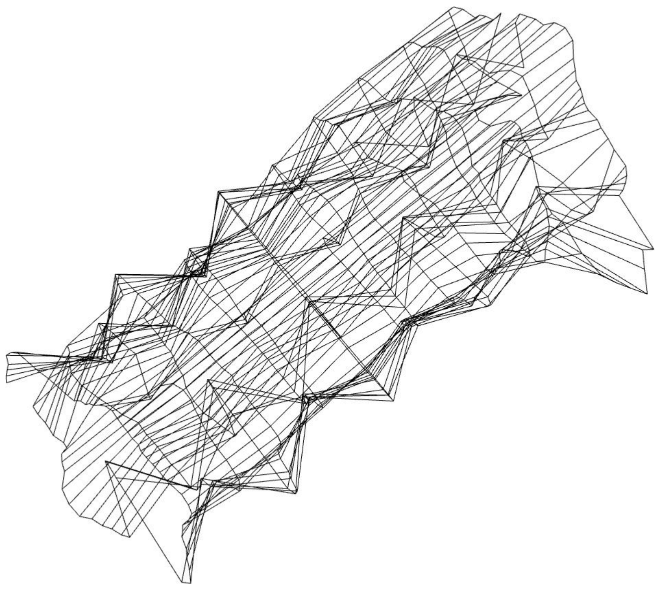


Figure 2.2 The Screenshot of Visualization

Features

- Texas Instrument MSP432E401P Microcontroller
    - 20-MHz Arm® Cortex® -M4F Processor Core With Floating-PointUnit (FPU)
    - 120 Hz clock speed
    - 1024KB of flash memory, 256KB of SRAM and 6KB EEPROM
    - Using C/C++ to Program
    - 3.3-5V operating voltage
- VL53L1X Time-to-flight Sensor
    - Theoretically longest measurement range is to 4m

    - 16-bit distance reading for millimeters
    - Baud rate is 115200 between the microcontroller and PC
- ULN2003 Stepper Motor Driver PCB and 28BYJ-48 Stepper Motor
    - 512 Steps per rotation
    - 5V operating supply
    - adjust the rotation speed by changing the delay
- Data
    - Using Python 3.88 and import serial and open3d package
    - I2C serial communication between microcontroller and sensor
    - UART serial communication between the microcontroller and PC
    - Process the data in xyz format file to generate 3D diagram
- Cost
    - The pole used to carry the step motor and sensor
    - Tapes used to combine components together
    - Box used to be measured

General Description

The lider is an embedded system including microcontroller, step motor and the sensor.The microcontroller controls the system and it provides power to all the components. It also transmits the measurements data from the sensor to PC. The LED onboard(decided by student specific requirements) should blink every time the step motor rotates 45 degrees. The step motor carries the sensor rotate together to make it finish the 360-degree measurement, which can get the coordinates to generate a y-z plane. The ToF sensor measures the distance and transmits the data to the microcontroller via I2C. And the data will be transmitted to the PC through UART via USB. On Pc, all the measurement data will be processed by the python file. The receive file will collect and store the data. And the visualization file will read the data stored to generate 3D visualization.

Block Diagram

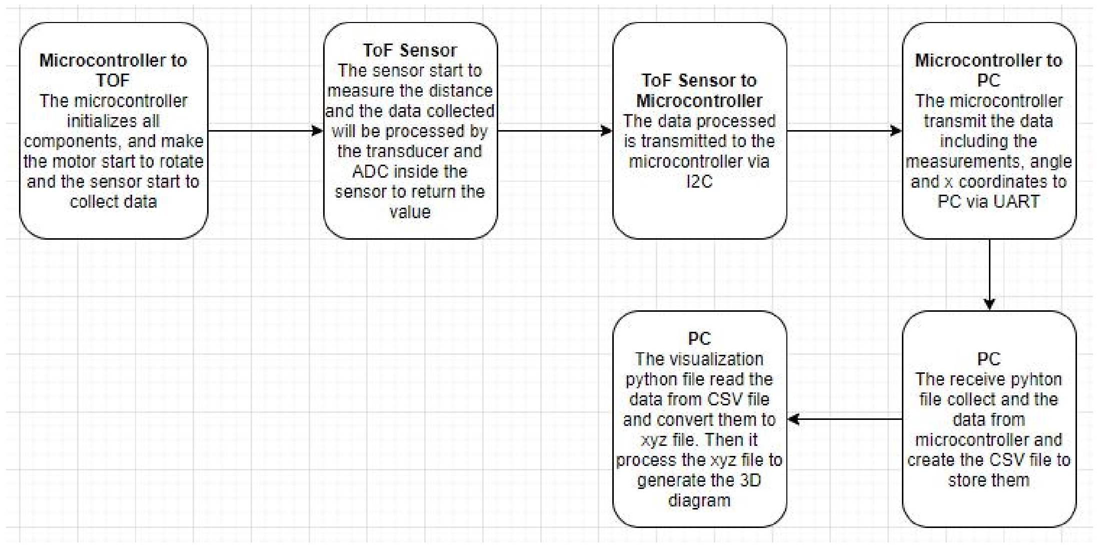

Microcontroller to TOF The microcontroller initializes all components, and make the motor start to rotate and the sensor start to collect data
ToF Sensor The sensor start to measure the distance and the data collected will be processed by the transducer and ADC inside the sensor to return the value
ToF Sensor to Microcontroller The data processed is transmitted to the microcontroller via I2C
Microcontroller to PC The microcontroller transmit the data including the measurements, angle and x coordinates to PC via UART
PC The receive pyhton file collect and the data from microcontroller and create the CSV file to store them
PC The visualization python file read the data from CSV file and convert them to xyz file. Then it process the xyz file to generate the 3D diagram

Figure 2.3 The Screenshot of Block Diagram

Device Characteristics Table

Feature	Detail	
Bus Speed	80 MHz	
Serial Port(UART Port)	COM3	
Baud Rate	115200 bps	
Pins for VL53L1X	Microcontroller	VL53L1X
	3.3V	VIN
	GND	GND
	PB3	SDA
	PB2	SCL
Pins for ULN2003	Microcontroller	ULN2003
	5V	V+
	GND	V-
	PH0	IN1
	PH1	IN2
	PH2	IN3
	PH3	IN4
Button	PJ1	
LED onboard	LED1(PN1)	

Detailed Description

Distance Measurement

The device with the function of measuring distance is composed of the microcontroller, the step motor and the sensor. The major component to measure distance is the VL53L1X sensor. It emits a pulse of invisible light (940 nm) to the target that we want to measure the distance with, and waits for the light to hit the object and returns back to the sensor. Then it measures how long it takes for emitted pulses of light to reach the nearest object and be reflected back to the detector. As we know, the equation of distance is that distance equals the product of time and speed. However, there are duplicated distances if we use this equation directly because the light gets back after it arrives at the target which means it experiences two distances. In order to get the correct measurement, we have to divide the product of time and speed of light by 2. So the equation to get the distance for sensor is $x = t*c/2$, where x is the distance, t is the time measured and c is the speed of light. The calculation will be finished inside the sensor, so the output which is returned is the distance from the sensor to the target directly.

After we connect the sensor with the microcontroller and combine it with the step motor together, we load the C code on the board and run the receive python file. When the C code is successful to be loaded and the “Open: COM3” appears on the screen, it means that it is ready to turn on the measurement system.

We set the PJ1 button and the input of the system. When we push the button PJ1 on the microcontroller, the microcontroller provides power which is obtained from the PC to step motor and the sensor which require 5V and 3.3V voltage separately. And the step motor will start to rotate because it receives the instruction from the microcontroller through GPIO. The step motor carries the sensor to rotate for 10 revolutions, which can help the sensor finish 360-degree measurement. Then the measurement could measure the distance fomt itself to all the objects around it and output the distances which can be converted to y and z coordinates to generate a plane. The step motor should experience 512 steps per revolution and the LED on board should blink every time when the motor gets 45 degrees which is equivalent to 64 steps. The step motor starts to rotate in the reverse direction after every revolution because if it keeps the same direction, the wires will get tangled up to block the rotation of the step motor and the measurement of the sensor.

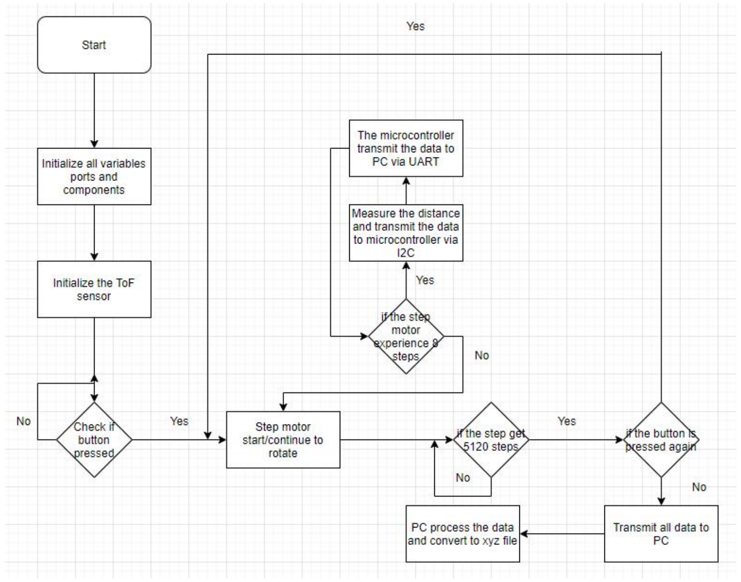
Also the sensor gets the instruction via I2C from the microcontroller and starts to measure. It is controlled by a series of predefined application programming interfaces(API) which is also used to receive data. It emits the laser to hit on the surrounding objects and return the output distance with the unit of millimeter from the calculation finished inside. We need 64 measurements per revolution so the sensor should measure once every time when the step motor gets 5.625 degrees which is equivalent to 8 steps. To decrease the measurement time, we reduced the delay time.After the device finches all the 10 revolutions, the sensor has outputted 640 measurements and the data is transmitted to the microcontroller via I2C.

When the microcontroller receives the distance data, it will transmit the data including distance, motor angle and the x-displacement to the PC via UART. All data will be collected and stored by the receive python file which has imported the serial and xlsxwriter package. We can see all of the distances are listed in the output shell of python. At the same time, a xlsx file named dataset.xlsx is created to store all of the measurements. The data stored in the dataset are the 640 measurements transmitted from the sensor via UART. The visualization python file concerts the measurements data to y and z coordinates through the equation below

$$= \qquad * \quad (\qquad)$$

$$= \qquad * \quad (\qquad)$$

The x coordinate is the displacement we move the device and it keeps the same in a revolution. Every time after the motor experiences one circle, the displacement will be increased in 10cm. These measurements will be converted to xyz coordinates. The x coordinate is the displacement we move the device and it keeps the same in a revolution. Every time after the motor experiences one circle, the displacement will be increased in 10cm. All of the coordinates values are stored in the xyz format file to be prepared to generate the 3D diagram.


Start
Initialize all variables ports and components
Initialize the ToF sensor
Check if button pressed
No
Yes
Step motor start/continue to rotate
if the step get 5120 steps
No
Yes
if the button is pressed again
Yes
No
Transmit all data to PC
PC process the data and convert to xyz file
if the step motor experience 8 steps
Yes
No
Measure the distance and transmit the data to microcontroller via I2C
The microcontroller transmit the data to PC via UART

Visualization

I use the laptop 17’ MSI GP76-427 running Windows 10 Family version with the Intel Core i7-10870H and Nvidia Geforce RTX 3070.To finish the visualization part, I installed the open3d package in python. Firstly we should download Python earlier than Python 3.9 because the edition after python 3.9 does not support open3d any more. The edition I use is Python 3.88. And I also import the package of xlrd because I need to use it to read the data store in the dataset.xlsx created by receive before.

When the microcontroller receives the distance data, it will transmit the data including distance, motor angle and the x-displacement to the PC via UART. All data will be collected and stored by the receive python file which has imported the serial and xlsxwriter package. We can see all of the distances are listed in the output shell of python. At the same time, a xlsx file named dataset.xlsx is created to store all of the measurements. The data stored in the dataset are the 640 measurements transmitted from the sensor via UART. The visualization python file concerts the measurements data to y and z coordinates through the equation below

= * ( )
= * ( )

The x coordinate is the displacement we move the device and it keeps the same in a revolution. Every time after the motor experiences one circle, the displacement will be increased in 10cm.

After getting the values of the three types of coordinates, the program will store them in the xyz format file line by line. As we know a point can be determined by a group of single x, y, and z coordinates. After all x, y and z coordinates are stored, a line represents a point.The y and z coordinates generated from the distance measurements in the same revolution are in the same plane because they share the same value of the x coordinate. The program links every point in the plane to generate the plane. After all planes are established. It links every point in the plane to the corresponding point in the next plane. Since all the planes are linked with the next one, a 3d diagram is completed. From the screenshot below, we can observe the lines before planes which make the diagram from 2D to 3D.
In the visualization python file. I use the for loop to read every column in the xlsx file because the data in every column belong to the same plane and with the same x displacement. It will be very convenient to link lines between the points in the same plane if we put the data in one column tiger in the xyz file. After all the data is processed. I use a for loop which will run 10 times. In every loop, I take 64 lines of coordinates which represent 64 points and link them to generate a plane. Because these 64 lines of data have the same x displacement and belong to the same plane. When the for loop is done, there are 10 planes at present. I use the last nested for loop to link the points with the corresponding points in the next plane. At last, every point has the link with the points in the same plane and the points in the next plane. The 3d Visualization is finished.
The room I recreated is the image shown below. It has bias with the room inside a box I measured. That may be caused by the wires or something else.

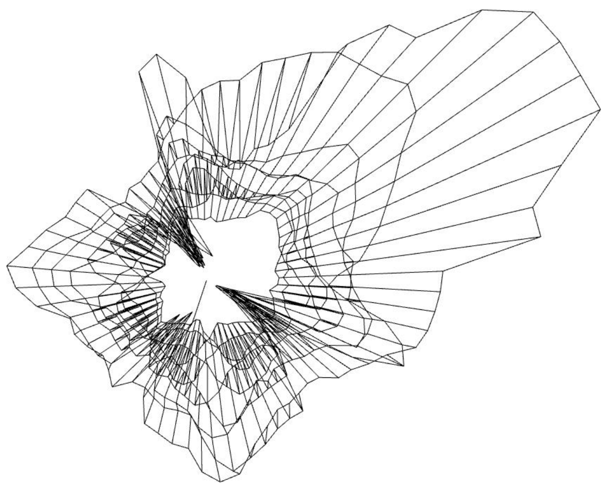


Figure 4 Screenshot of Visualization

Application Example with Expected Output

The guidance of how to use the lidar system

- Software
  - Download Python 3.88 to run the program which is included in the project file. You can download that from Python Release Python 3.8.8 | Python.org and choose the edition allowed on your PC. Note: You can choose the python in lower edition but you cannot download the python above 3.9 because it doesn't accept the use of the open3d package.
  - Install 4 python packages:
    serial: serial · PyPI
    xlsxwriter: XlsxWriter · PyPI
    xlrd: xlrd · PyPI
    xlrd-compdoc-commented · PyPI
    open3d: open3d · PyPI
  - Download Keil to run or setup the C code for the microcontroller. You can down keil from Keil Downloads
- Connection and Usage
  - Use the USB to connect your PC with the microcontroller. All the wires are connected with the ports on the microcontroller with the components set well. Do not change that without altering the code.
  - Run the C code in Keil and load it on the microcontroller. Run the python file receive.py and observe the screen. If there is a window showing"Opening: COM3", it means the PC is ready to receive data from a microcontroller via UART.
  - Push the wake button to initialize and active the microcontroller. When you are ready, push the button PJ1 to turn the entire system on. You will observe the senator start to rotate with the step motor. You can hold it at rest to measure the data of a plane, or move with the device to measure the size of a room.
  - After the sensor finishes 10 revolutions, all of the measurements data are transmitted from the sensor to the microcontroller via I2C and transmitted from the microcontroller to the PC via UART.
  - You will see all the measurements listed on the screen, and a xlsx.file is created to store all of the distance data. Now open the Visualization.py and run it. The python file will read and process the data stored in the xlsx.file and generate a 3D diagram.
- Setup
  To set up the device, you can change the code in C or Python.

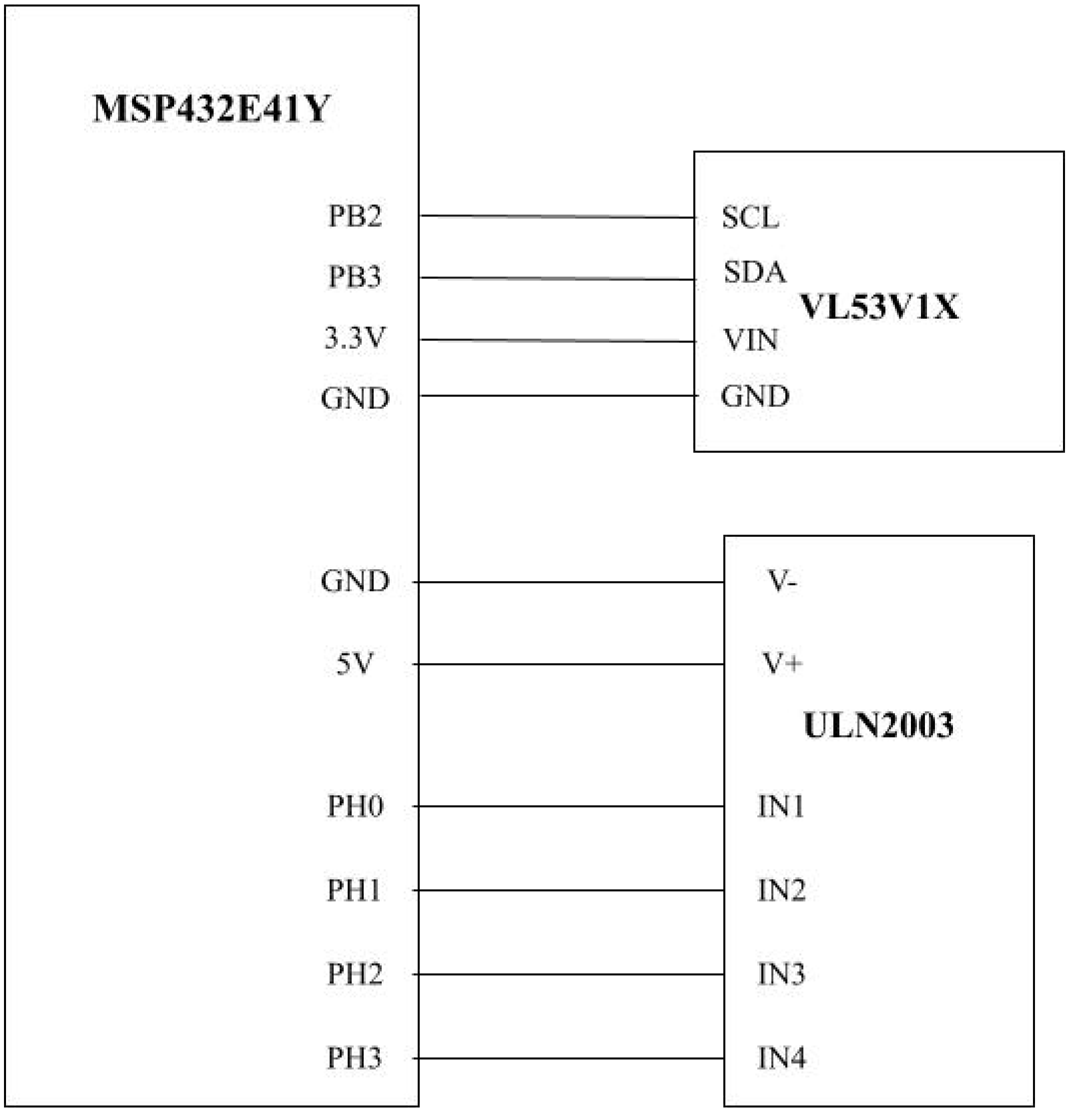
- If you want to set the bus speed, you can go to the file PLL.h to change the PSYSDIV in line 29. The table of the PSYSDIV and the corresponding bus speed is on the bottom in this file.
- To speed up or slow down the rotation, you can decrease or increase the delay time. The default delay period is 1ms and you can set it in the SysTick.c in line70. To alter the number of the delay for the step motor, go to the 2dx4_dx90-1. c and change the number in the bracket from line 40 to 58.
- If you want to set the COM port or the baud rate by yourself, open the receive.py and ho to the line 23. You can see the COM3 and 115200 in the bracket. The COM3 represents the COM port and the 115200 represents the baud rate. Change any one of them by yourself to set different COM ports or baud rates.

● Measurement
- For my lidar device, I measured the horizontal plane with the sensor. I set the vertical direction as x, and the horizontal plane as the y-z plane. So I raise the device up vertically in 10cm every time when it finishes one revolution to complete the x-direction displacement.

Limitations

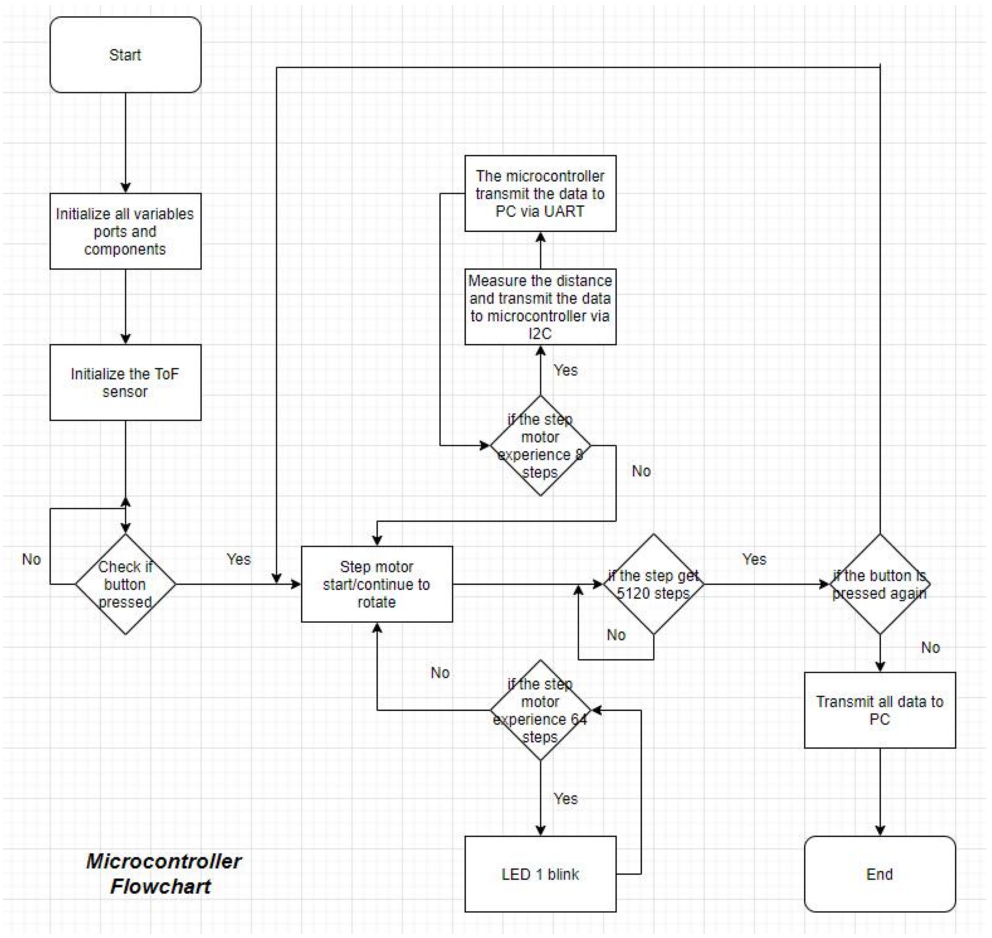
1) The microcontroller has the Floating point Unit(FPU) which can support 32-bit(single-precious) mathematical operations like addition, subtraction, multiplication,division and square root. To deal with trigonometric functions found in math libraries, FPU should split the 64-bit(double precious) operations into 32-bit words because it is 32-bit wide.
2) The maximum quantization error for the ToF module: $4000 \quad /2^{16} = 0.061$
   The maximum quantization error for the IMU module: $32g/2^{16} = 4.79 * 10^{-3} m/s^{2}$
3) The maximum standard serial communication rate that can be implemented with a PC is 128000 bits/s. It is verified by checking the port settings for the XDS110 UART Port.
4) They communicate by using the I2C serial communication. The clock speed is 100 KHz and the transfer rate is 100 kbps
5) The primary limitation on speed should be the ranging on the ToF sensor. The timing budget of the sensor is between 20ms and 1000ms. However, 140 ms should be the minimum budget because the max distance allowed to be measured is 4000mm for the sensor. The sensor also has the delay which minimum value is 5ms between 2 ranging operations. According to this, the minimum time cost on one measurement is 285ms, we can't assign less delay to get the faster speed. To test this, we can set the delay less than the minimum delay we get above. The sensor won't give any response to us the the entire system will not work.

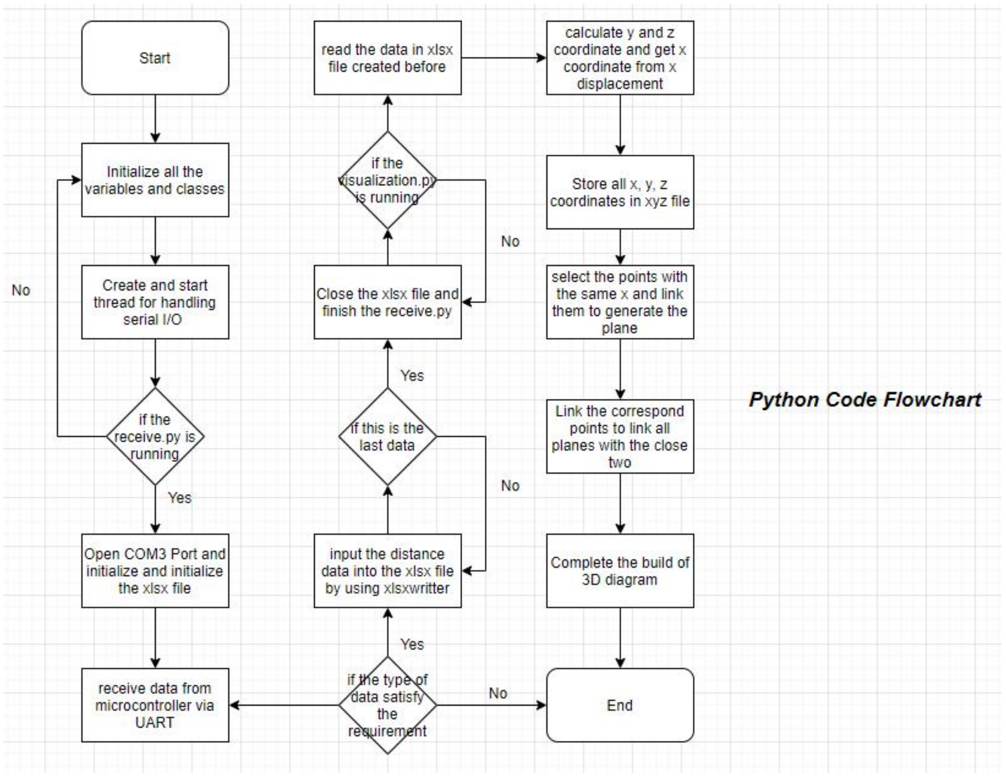
Circuit Schematic



Programming Logic Flowchart

For microcontroller


Start
Initialize all variables ports and components
Initialize the ToF sensor
Check if button pressed
No
Yes
Step motor start/continue to rotate
The microcontroller transmit the data to PC via UART
Measure the distance and transmit the data to microcontroller via I2C
Yes
if the step motor experience 8 steps
No
if the step get 5120 steps
No
Yes
if the button is pressed again
No
Transmit all data to PC
End
No
if the step motor experience 64 steps
Yes
LED 1 blink
Microcontroller Flowchart


Start
Initialize all the variables and classes
Create and start thread for handling serial I/O
No
if the receive.py is running
Yes
Open COM3 Port and initialize and initialize the xlsx file
receive data from microcontroller via UART
if the type of data satisfy the requirement
Yes
No
input the distance data into the xlsx file by using xlsxwritter
if this is the last data
No
Yes
Close the xlsx file and finish the receive.py
if the visualization.py is running
No
read the data in xlsx file created before
calculate y and z coordinate and get x coordinate from x displacement
Store all x, y, z coordinates in xyz file
select the points with the same x and link them to generate the plane
Link the correspond points to link all planes with the close two
Complete the build of 3D diagram
End
Python Code Flowchart