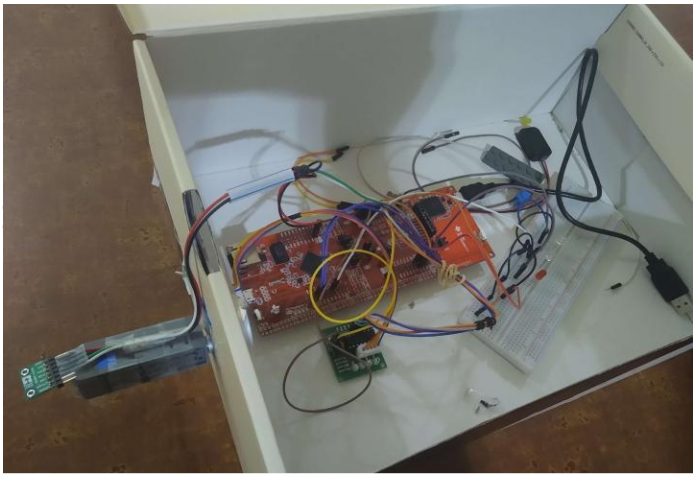
PRODUCT DATASHEET

3D Room Scanner

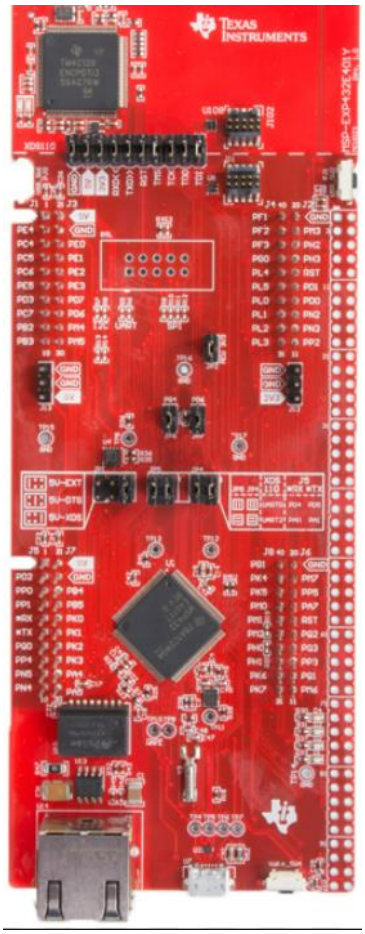
Abdurahman Bade - 400247875



Applications

As a future member of the engineering profession, the student is responsible for performing the required work in an honest manner, without plagiarism and cheating. Submitting this work with my name and student number is a statement and understanding that this work is my own and adheres to the Academic Integrity Policy of McMaster University and the Code of Conduct of the Professional Engineers of Ontario. Submitted by [**Abdurahman Bade, badea, 400247875**]

Features



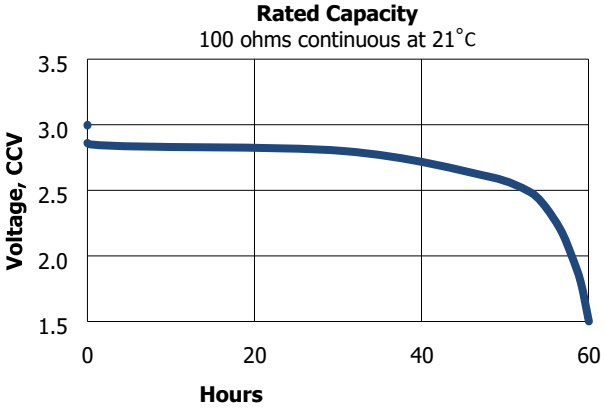
MSP432E401Y

COMPENG 2DX3

Microprocessor Systems Project

Lab Section: L02

Typical Discharge Characteristics



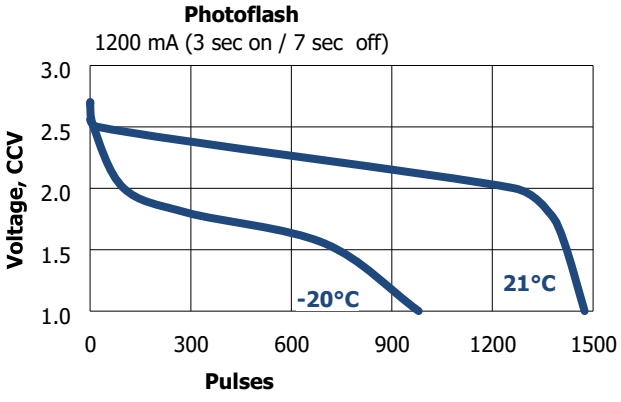


Table of Contents

Device Overview

Features

MSP-EXP432E401Y Microcontroller

- Bus Speed: 120MHz
- Operating Voltage: 2.5 – 5.5V
- 2x 12-Bit SAR-Based ADC Modules
- Cost: $40.00 (USD)
- Serial Communication: 8 independent UART's, 10 I2C's
- Baud Rate: 115200
- Programming Language: C/C++
- Flash Memory: 1024 KB
- SRAM: 256 KB
- EEPROM: 6KB

VL53L1X Time-of-Flight Sensor:

- Operating Voltage: 2.6 – 3.5 V
- Cost: $24.95 (CAD)
- Max Distance (Measurement): 4 metres
- Max Frequency: 50 Hz
- Lazer Emitter: 940 nm

Definition/Description

- Bus Speed: how quickly can a system move data from one component to the other.
- Operating Voltage: the amount of voltage needed to operate said component.
- Baud Rate: rate at which information is transferred in a channel.
- Flash Memory: memory that holds its data without a power supply.
- Static Random Access Memory (SRAM): memory that retains data bits as longs as power is supplied.
- EEPROM: erasable read-only memory.

General Description

The main components that were used in this project were the MSP432E401Y microprocessor and the VL53L1X Time-of-Flight sensor. The sensor uses a stepper motor to detect distances in its 4-metre surroundings, which are then converted into coordinates and plotted using 3D graphing software. After pressing an on-board button, the microcontroller was programmed in C to spin the shaft of the motor, which would output a 3D render of a room from an external Python code. The process of obtaining tangible signals from the environment using sensors is known as signal acquisition. These signals can be electrical, mechanical, auditory, or optical in nature, and they can transmit information about the world around us. Preprocessing is the procedure that follows signal capture and involves using different ways to clean, sort, and improve the signal before it is further processed. Analog-to-digital conversion (ADC) is the process of turning a constant-time and amplitude analogue signal into a discrete-time and amplitude digital signal. This is accomplished by sampling the analogue stream in certain intervals and allocating each sample a digital value. Serial communication is a way of sequentially delivering data between two devices, one bit at a time, across just one communication link or network. The practise of putting facts or information in a visual manner to assist people comprehend and analyse complicated material more readily is known as visualisation.

Block Diagram

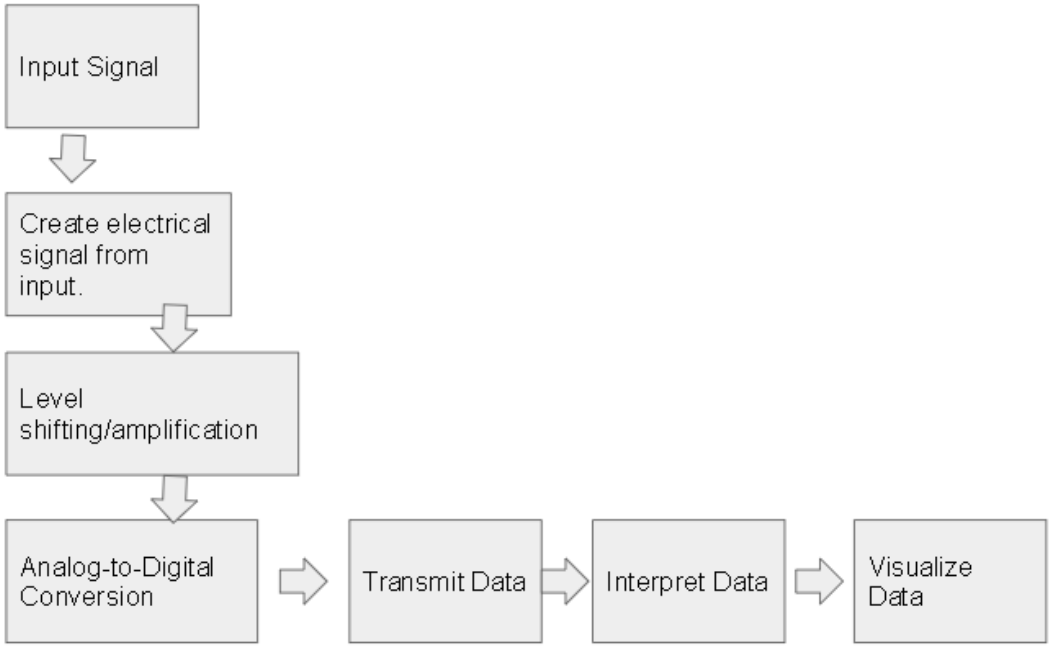


Figure 1: Block Diagram of the Project

Device Characteristics Table

Component	Pin Value
ToF Sensor	SCL (PB2), SCA (PB3)
Stepper Motor	PH0:3
Serial Port Communication Speed	115200 BPS
Bus Speed	16 MHz
LED's	PL0, PE0, PM0, PF4
Push Button	PJ1

Detailed Description

Distance Measurements

To begin the distance measuring process, the system initialises and configures all essential ports, including Port H for triggering the stepper motor, Port L, E, M, and F for the LED indication lights, Port J for the onboard push button, and Port B for I2C. Interrupts are then enabled after establishing the appropriate values for each individual port, as the system relies on interrupts for Port J to detect button pushes. All necessary steps for configuring interrupts are done, such as arming the interrupt source, activating the interrupt, enabling global interrupts, establishing the priority level, and developing a suitable interrupt service routine (ISR). After completing all of these processes, the system can now measure distance using an invisible laser at the push of an onboard button. In order to accomplish this, the system employs a 940 nm invisible laser transmitter to emit a wavelength of light into its surroundings, which is then reflected off of the nearest object, returning to the ToF being viewed by the receiver. The microprocessor on the ToF sensor's board then calculates the distance between itself and this item using simple calculations on the time it just recorded for the signal in question to return and the known value for the speed of light, which is approximately 2.997 * 10 ^ 8. Using the speed light, we can derive an equation to measure distance, which is: Distance = (Photon Travel Time/2) x Speed of Light x 10-3 mm. The resulting value was then multiplied by 10 to the power of -3 since, as previously indicated, the sensor provides the distance measurement to the microcontroller through I2C in millimetres. Ten different measurements were used in this project to obtain a 3D render.

Visualization

The PC handles the data visualisation aspect of the procedure via a Python program. This procedure starts by importing the necessary extra libraries, which include serial, math, open3d, and numpy. The program then sets the serial port to 'COM5' with a baud rate of 115200 bps, which is the same as the microcontroller's baud rate. The port is then opened and a '.xyz' file gets outputted and is set up for writing. Variables for the total number of steps taken by the motor and the beginning position in the X direction are then set. The program subsequently waits for the user to choose the total amount of complete scans they want to do. After receiving a number from the user, the system is ready to begin receiving data from the microcontroller After that, the program receives the data from the microcontroller, decodes it, and transmits the data. The motor's angle of rotation is then computed by dividing the total number of steps performed by the stepper motor by the number of steps necessary to complete a full 360-degree rotation and multiplying this result by 2pi. The vertical y and horizontal z coordinates are derived by multiplying the transmitted distance measurement by the sin and cos of the angle.

Theta = (Num. of Step / Num. of Steps in Full Rotation) * 2pi

Y = d*sin(Theta), Z = d*cos(Theta)

The program now checks to see if the motor has completed a full revolution, and if so, it resets the step count to zero, increases the X value by the predetermined amount, and increases the number of scans taken by one. This operation is repeated until the number of scans specified by the user is reached, at which time the file is closed, preserving the data, and eventually outputting an "XYZ" file. Using the open3d addon in Python, the final result of the render would then display after closing the point cloud render.

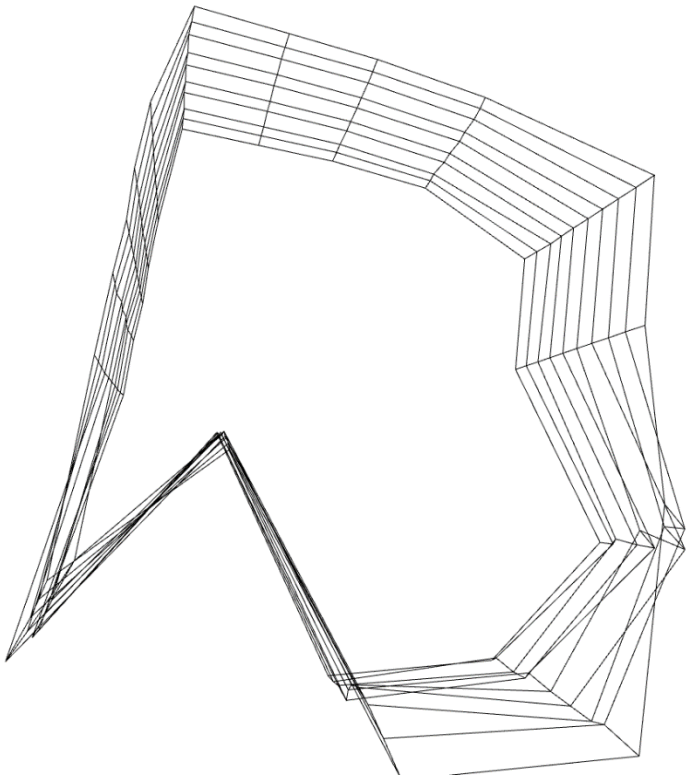


Figure 2: 3D Graph of ETB First Floor Hallway (Connected)

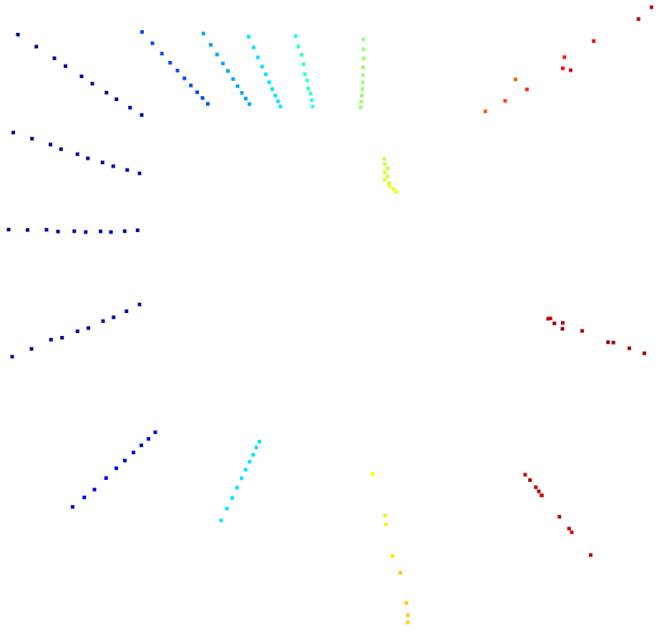


Figure 3: 3D of ETD First Floor Hallway (Point Cloud)

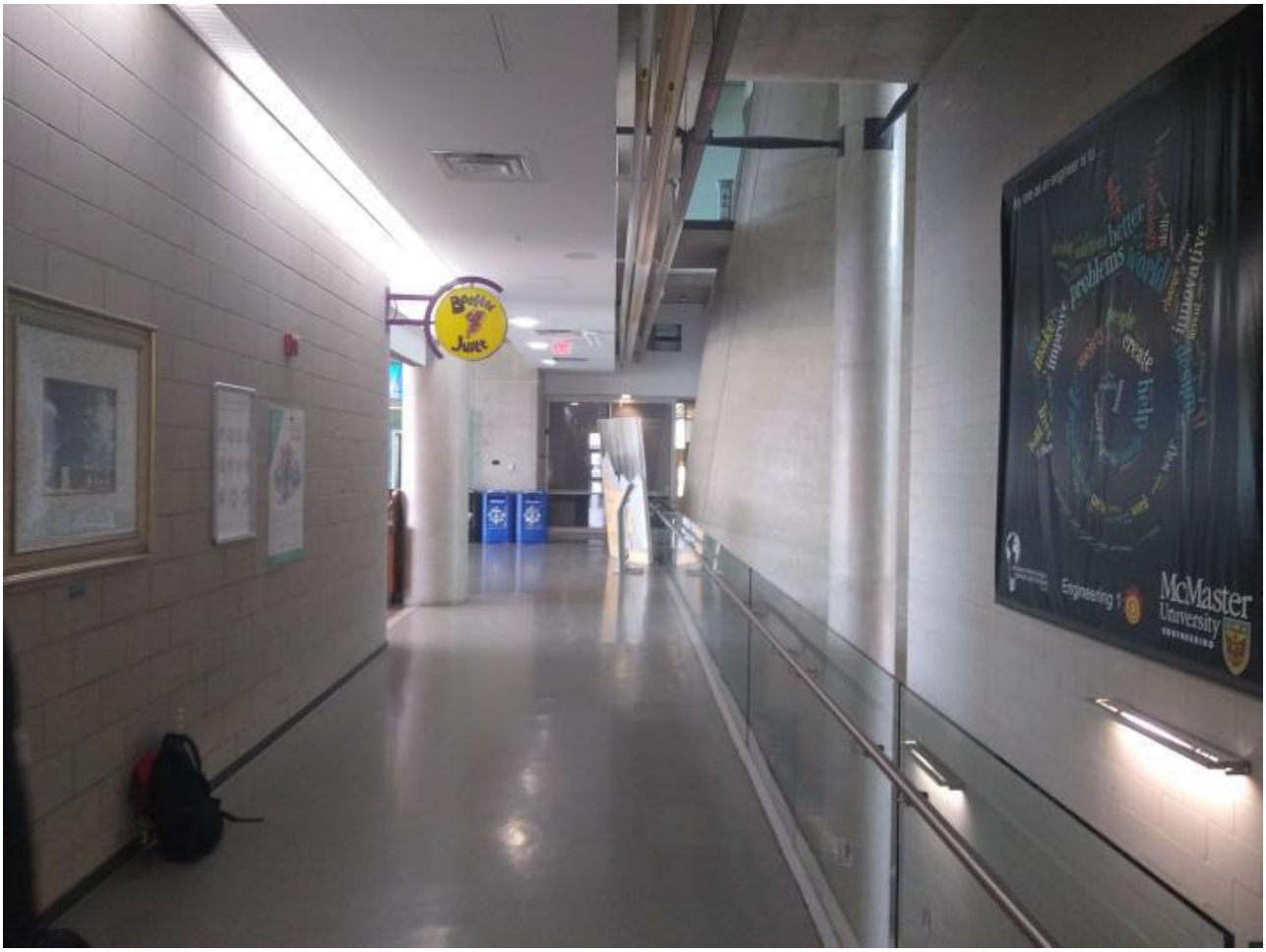


Figure 4: Picture of ETB Hallway

Usage Examples/Instructions

1. Connect all the components to their respective pins, as illustrated in the circuit schematic below.
2. Using Keil uVision, flash the C project into the microcontroller.
3. Ensure you reset the microcontroller upon flashing the code.
4. In order to effectively use UART and I2C, you must identify what COM port your device uses by searching “Device Manager” on Windows Start, then go to “Ports” to determine you COM value.
5. Ensure that the baud rate is set to 125200 and that the COM value matches the port value in UART and the Python code.
6. Run the Python code and enter the number of steps you want to make.
7. Commence the scanning by pushing PJ1 button.
8. After a full rotation, take one increment of steps until you reach your desired number of steps.
9. Both the point cloud and the final 3D graph will display upon a successful reading.

Limitations

<u>Max Quantization Error for microcontroller:</u>

Resolution = V_fs / 2^n = 5 / 2 ^ 12 = 1.22 mV

Max Quantization Error for ToF sensor:

Resolution = 3.3 / 2 ^ 8 = 12.9 mV

- The MSP432E401Y microcontroller is powered by a 32-bit Arm Cortex-M4F CPU core with a floating-point unit, or FPU. As a result, floating point computations will be restricted to the 32 bits available.
- The highest conventional serial communication rate that the PC can support is 115200 bps. This was validated when the baud rate was increased to 230400 bps and an error occurred while attempting to open the UART port. This happened because the microcontroller can only withstand a baud rate of 115200 bps.
- The microcontroller and ToF sensor communicated using I2C, which means there was a data line as well as a clock signal line. The serial data line (SDA) is used for data transmission, while the serial clock line (SCL) is used to synchronise data transfer. The speed between the two devices is equivalent to the peak transmission speed of the ToF, which is 400 kbps.
- The baud rate of the UART transmissions between the microcontroller and the PC is the fundamental constraint on system speed. This limits the system to 115200 bps, which is much less than the ToF and micro's 400 kbps connection bandwidth.

Circuit Schematic

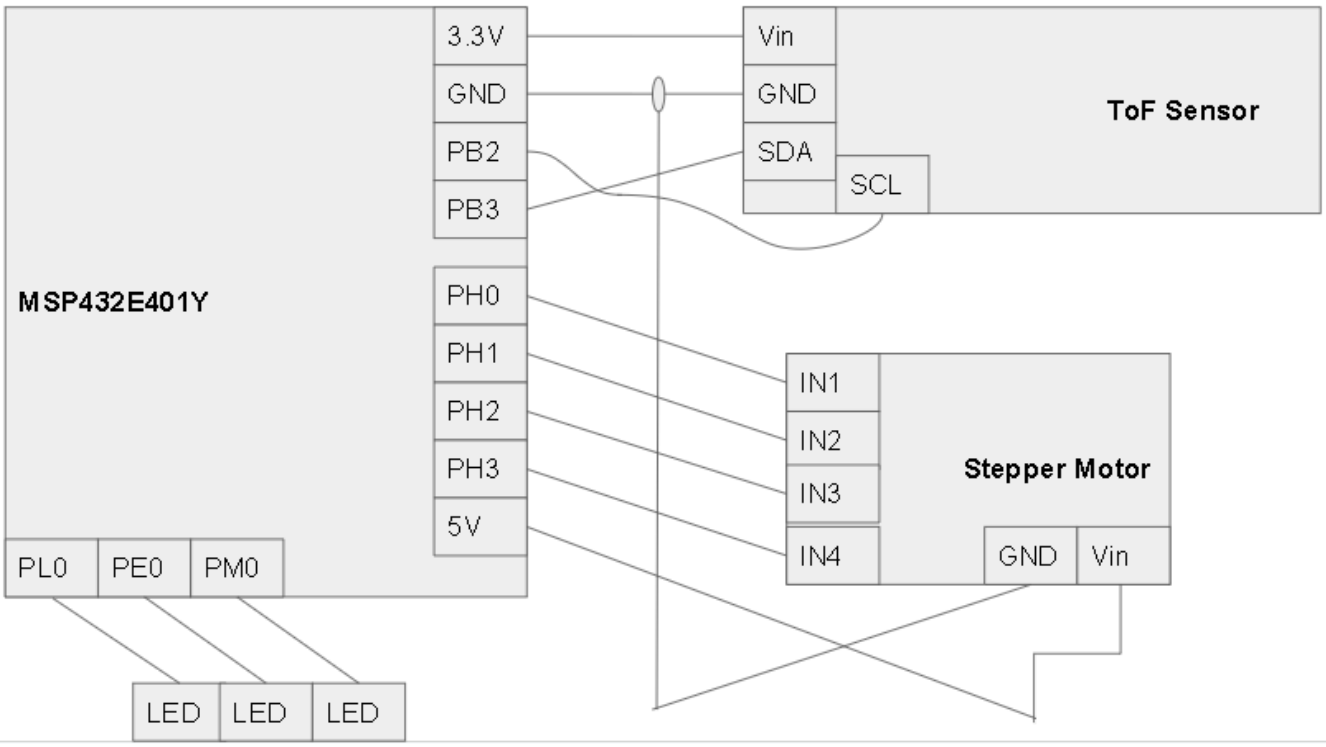


Figure 5: Circuit Schematic of Project

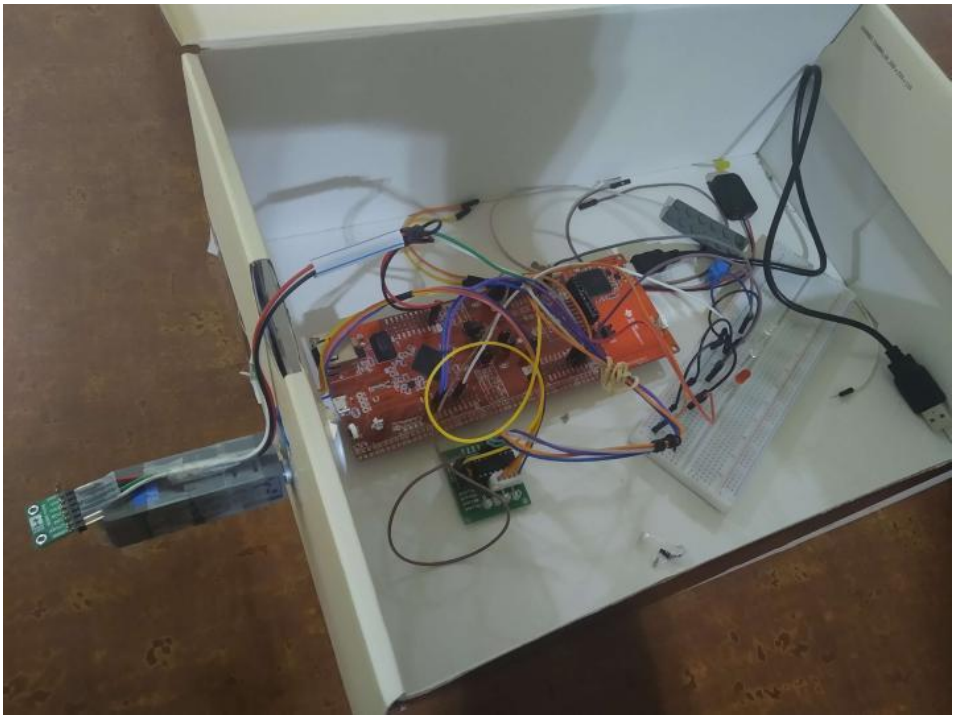


Figure 6: Photo of Project Setup

Programming Logic Flowchart

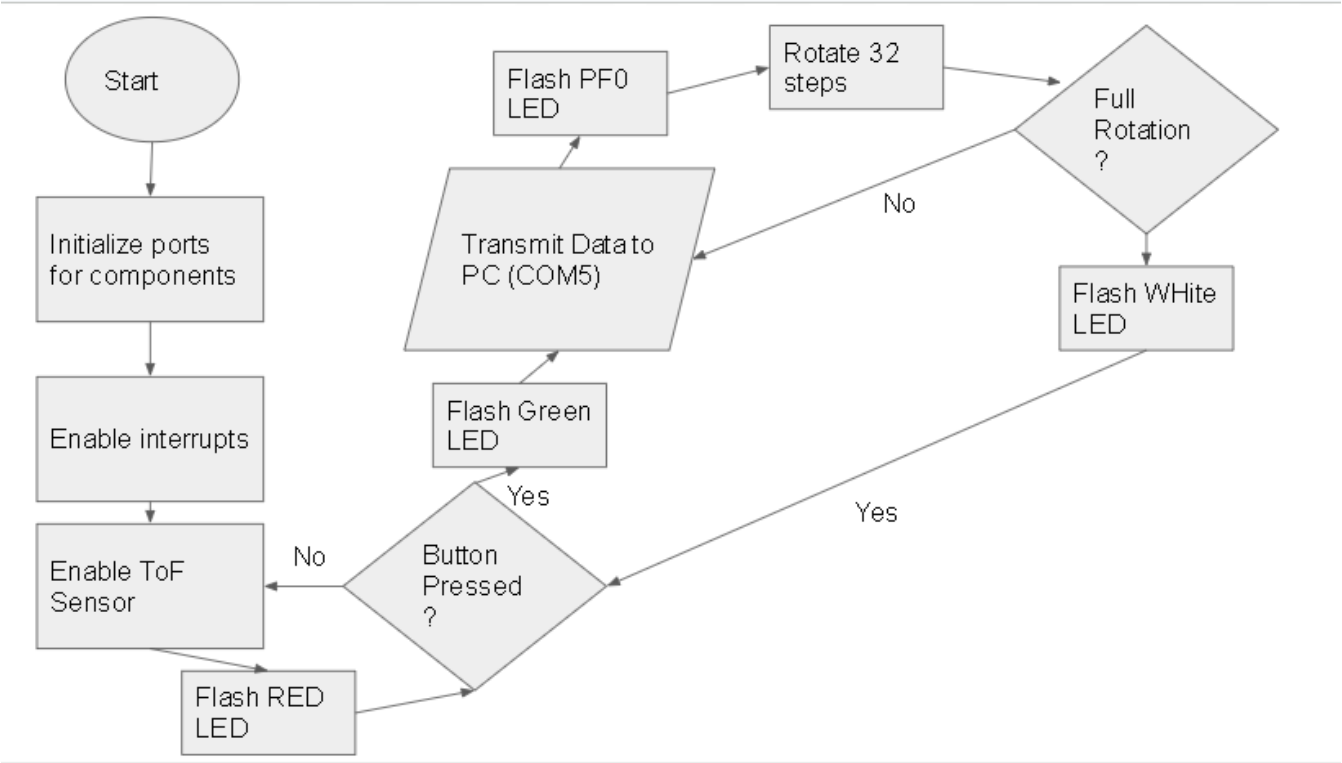


Figure 7: Microcontroller Code Flowchart