COMPENG-2DX3

TGN-6909 Datasheet

Tirth Nagar

nagart1

400446909

April 10th, 2024

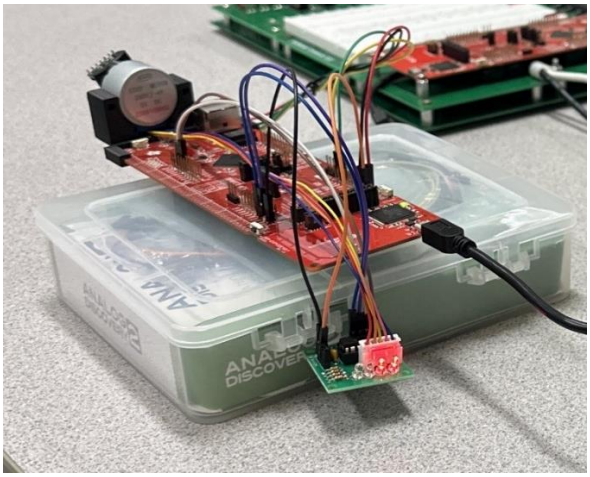
Instructors: Dr. Shahrukh Athar, Dr. Thomas Doyle, Dr. Yaser M. Haddara

TGN-6909 LieDAR for Spatial Mapping

The TGN-6909 is a compact and cost-effective embedded spatial measurement system designed for indoor 3D mapping. Utilizing time-of-flight sensing technology and a rotary mechanism, it provides precise 360-degree distance measurements within a single vertical plane, while integrating fixed distance samples along the orthogonal axis. Unlike bulky commercial LIDAR equipment, the TGN-6909 offers a smaller and more affordable solution without sacrificing performance. With onboard memory for storing spatial data and seamless communication with personal computers, engineers can efficiently acquire and analyze physical phenomena, ensuring precise and accurate measurements at a much lower cost.

Basic Features

- Built-in error checking and on-the-fly error mitigation.
- Single-Button interrupt based operation.
- 4 onboard status LEDs.
- Variable precision with customizable number of measurements per 360º rotation.
- Real-time data transmission and processing.
- Asynchronous data transmission for high-speed applications
- Fully automated 3D Reconstruction



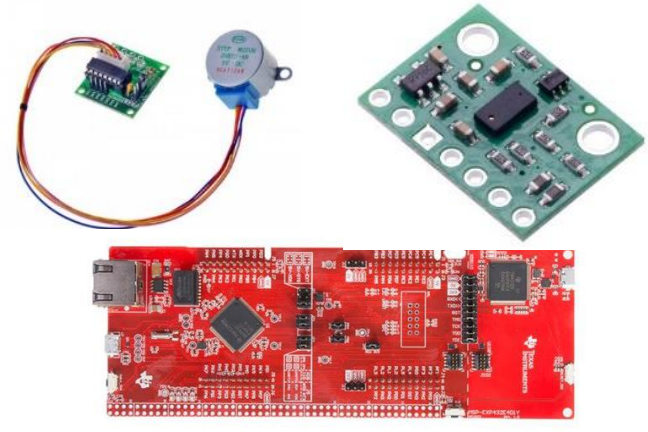
Hardware Setup

MSP432E401Y	
Clock Speed	96Mhz
Baud Rate	115200
Processor	Cortex M4F
Serial Port	COM 4

Digital I/O	
LED 1	PN1
LED 2	PN0
LED 3	PF4
LED 4	PF0
Onboard Button	PJ1

ULN2003 Driver Board	
ULN2003 Pin	**Microcontroller**
IN_1 – IN_4	PH0 - PH3
V_+	5V
V_-	GND

VL53L1X	
VL53L1X	**Microcontroller**
Vin	3.3V
GND	GND
SCL	PB2
SDA	PB3



1 Device Overview

Features

Texas Instrument MSP-EXP432E401Y

- Facilitates seamless data exchange between TOF sensor and computer systems as the primary interface.
- Employs a powerful 32-bit Cortex M4F CPU for efficient processing.
- Operates at an impressive bus speed of 96 MHz for swift processing.
- Offers ample storage with 256 KB of SRAM and 1 MB of Flash Memory.
- Enhances versatility with General Purpose Input/Output (GPIO) Pins.
- Provides clear visual feedback and status indication through integrated LEDs.

VL53L1X Time of Flight Sensor Capabilities

- Capable of measuring distances up to 4m, catering to a wide range of spatial needs.
- Operates efficiently at a frequency of 50Hz, ensuring rapid data acquisition and processing.
- Offers flexibility with an operating voltage range of 2.6V to 3.5V, accommodating various power supply setups.

28BYJ-48 Stepper Motor and ULN2003 Diver Board Features

- Moves precisely with 4 phases per step, ensuring accurate spatial measurements.
- Provides fine granularity with 512 steps per 360-degree rotation, enabling precise positioning.
- Features 4 LEDs on the driver board, offering clear indication of the current phase for enhanced monitoring.
- Offers flexibility with an operating voltage range of 5V to 12V, catering to diverse power supply requirements.

Serial Communication Protocols

- Facilitates seamless data collection via the I2C protocol from the sensor to the microcontroller, ensuring efficient communication.
- Utilizes the UART protocol for data transmission from the microcontroller to the PC, operating at a baud rate of 115200 BPS for rapid and reliable data transfer.

Environment Recreation

- Enables serial communication with the PC via the Pyserial library in the Python programming language, ensuring seamless data reception using UART.
- Utilizes the Open3D library in Python for 3D visualization of spatial measurement data post-processing, enhancing clarity and insight into the gathered information.

TGN-6909

General Description

This state-of-the-art device embodies efficiency and user-friendliness, revolutionizing 3D visualization through seamless integration of an onboard Time-of-Flight (ToF) sensor. Operating with precision, it captures data across the xy-plane at precise intervals of 11.25 degrees. Upon successful distance measurement, LEDs 1 momentarily lights-up, indicating completion before a brief pause in motor movement to process the acquired information.

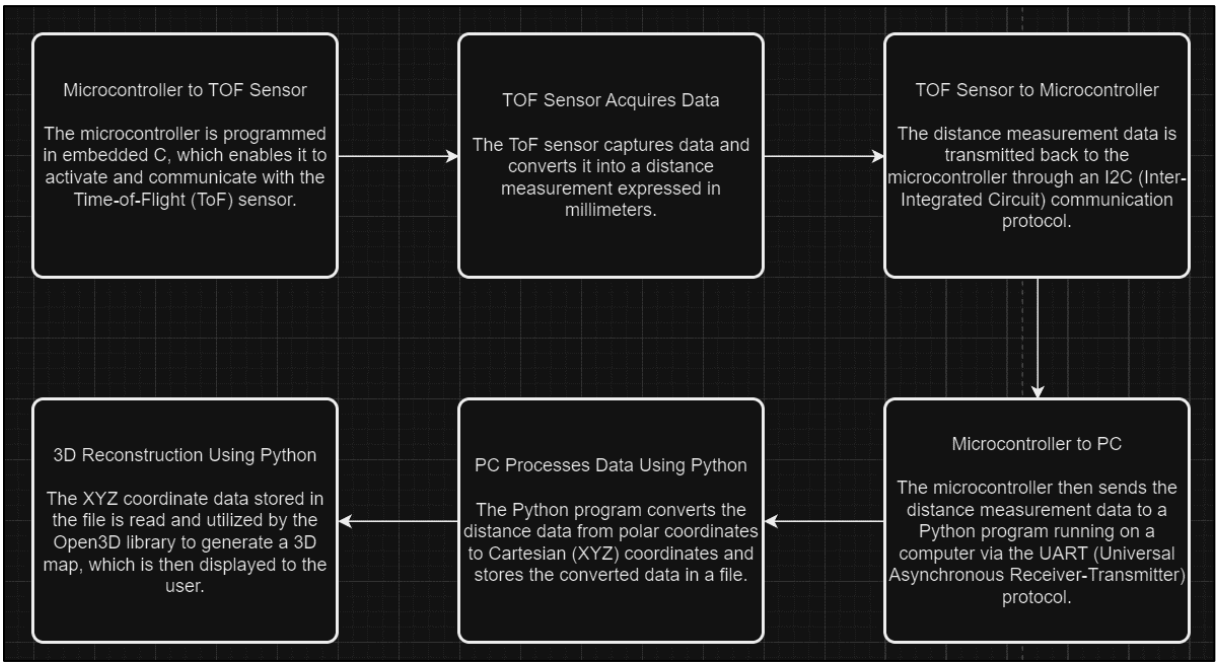
Initiating the scanning process is effortless, initiated by a simple press of the microcontroller's reset button and toggle of the motor using the onboard button PJ1. Subsequently, a designated Python script is executed to commence the operation. Upon system startup and activation of the ToF sensor, the illumination of LEDs 1-4 signifies readiness and operational status of the TOF sensor.

The sensor's data is transmitted to the board via the I2C protocol, then seamlessly relayed to an external PC through UART, facilitating real-time data processing and visualization. At the core of this innovative system lies the MSP-EXP432E401Y microcontroller, orchestrating timing with precision and operating at a swift bus speed of 96MHz.

Using a light beam emitted by the ToF sensor, distances are meticulously measured by calculating the round-trip time, employing the formula for measured distance: half the photon travel time multiplied by the speed of light.

Data processing and visualization are seamlessly executed using Python, coupled with the powerful capabilities of the Open3D library. This intricate process involves transforming polar data into Cartesian coordinates through precise trigonometric calculations. Extracting X and Y components from these calculations, Z coordinates are manually adjusted to accurately represent depth. The processed data is then elegantly plotted on a 3D Cartesian axis, offering a comprehensive and visually striking representation of the scanned area.

Block Diagram



2 Device Characteristics

MSP432E401Y		ULN2003 Driver Board		VL53L1X	
Feature	**Details**	**Microcontroller**	**Board Pins**	**Microcontroller**	**Sensor Pins**
Clock Speed	96Mhz	**IN_1 – IN_4**	PH0 - PH3	**Vin**	3.3V
Baud Rate	115200	**V+**	5V	**GND**	GND
Processor	Cortex M4F	**V-**	GND	**SCL**	PB2
Serial Port	COM 4			**SDA**	PB3
Clock Speed	96Mhz				
Baud Rate	115200				
LEDs	PN0, PN1, PF4, PF0				

3 Detailed Description

Distance Measurements

This cutting-edge 3D mapping device relies extensively on the VL53L1x Time-of-Flight (ToF) sensor, leveraging its advanced capabilities to measure distances with exceptional precision. Operating on the principle of emitting and receiving infrared light beams, the sensor directs these beams towards objects, capturing their reflections to calculate distances. The crucial distance measurement is derived by computing the time delay between emission and reception, utilizing then formula Distance = time of flight/2 * speed of light, which translates time of flight into accurate radial distances.

Post-measurement capture, the sensor undergoes a series of processes including transduction, conditioning, and Analog-to-Digital Conversion. These processes enable the conversion of analog distance measurements into digital format for transmission to the microcontroller via I2C serial communication.

To initiate the sensor's scanning operation, the system employs an interrupt method to toggle the motor state, followed by a polling method executed within the Keil C programming environment. Ensuring synchronization, both the Python script and the Keil C program must share identical variable values, such as total scans and measurements per scan. This alignment facilitates seamless coordination between the Keil program and Python script, ensuring efficient operation.

The scanning process unfolds through a straightforward sequence. Upon pressing the reset button on the microcontroller, LEDs 1-4 flash to signify readiness. Once they extinguish, confirming operational status of the sensor, the motor toggle is activated, and the designated Python script is executed. This script initiates distance measurements by sending a character through UART from the PC to the microcontroller. The Keil code interprets this character as a

signal to commence measurements, eliminating the need for manual intervention via external push buttons, thereby greatly enhancing user experience and simplifying operation.

Data Processing and Visualization

During the visualization phase, Python, in conjunction with the Open3D library, plays a crucial role in converting the polar coordinates obtained from the ToF sensor into precise XYZ coordinates. As outlined in the distance measurement section, all measurements are gathered into an array and transmitted to Python for visualization using the UART protocol. Python parses these values and executes the necessary conversions.

The X and Y coordinates are determined utilizing the following formulas, with values divided by ten to ensure measurements are in centimeters:

$$X = \text{radial distance} \times \sin(\text{angle}) \div 10$$

$$Y = \text{radial distance} \times \cos(\text{angle}) \div 10$$

The Z coordinate functions as a variable dictating the depth of each measurement cycle, offering flexibility for adjustment within the Python program to suit diverse user preferences.

Following each conversion, the Python script arranges these transformed coordinates into a file, ensuring they are sorted for compatibility with the Open3D library for subsequent data visualization. This setup empowers users to modify specific point values if inaccuracies are suspected or to tailor aspects of the final visualization to their preferences.

To generate a 3D map of the space, the file containing all sorted distance measurements is initially opened using file I/O and read by Open3D. Subsequently, the kd-tree function within the Open3D library facilitates automated and efficient point connection based on proximity, resulting in a customizable point cloud. Ultimately, this process yields a mesh-like visualization of the area scanned by the TGN-6909, providing invaluable insights into the environment.

4 Application Example

Startup Guide

This startup guide assumes that the user has all the necessary hardware configured correctly using the connection guide and/or the schematic. It also assumes that the user has downloaded Keil.

1. Download Python version 3.11.7 from https://www.python.org/downloads/.
2. Download zip file which includes both the Keil project, and the Python Script required.
3. Open the Keil project and go to the 'main.c' file, edit the variables "SAMPLES" and "NUM_SCANS" to the desired amount.
4. Save the code, and then click translate, build, and finally load to load the code to the microcontroller.
5. Open the Python script named 'script.py' and change the variables "SAMPLES" and "NUM_SCANS" to match the corresponding values in step 4.
6. Change "Z_INCREMENT" variable to the desired depth (in cm).
7. Change COM port in line 16 of the python script match the UART port the microcontroller is connected to (this can be found from device manager → Ports).
8. Change the output file path on line 27 to the desired location of your save file.
9. Then click the reset button on the microcontroller and wait for all LEDS to flash.
10. Toggle the motor state by pressing the onboard button PJ1.
11. Run the Python script and scan the environment.
12. Follow the steps in the terminal to initiate data transfer once the motor has stopped spinning completely.

    Then finally a 3D reconstruction of the user's environment then open in a separate window.

Expected Output

Below is a practical demonstration of the TGN-6909 in action. The output illustrates the device's scanning of a hallway. Optimal results are achieved when the user remains stationary during the scan and when there are minimal non-reflective surfaces, such as glass, present within the scanned area.

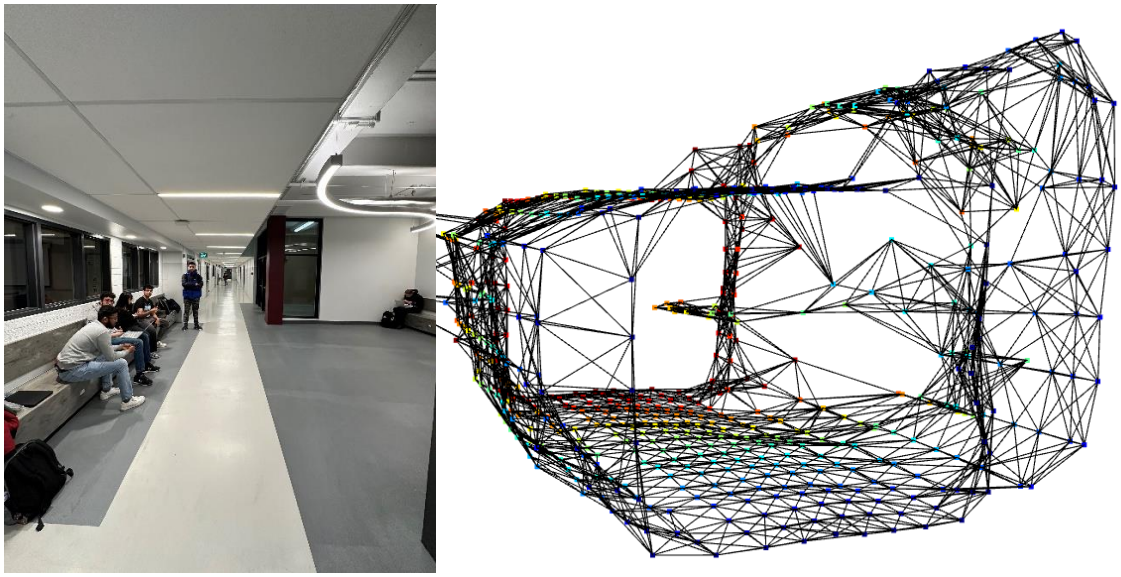


Figure 1 Picture of the hallway on the left, with corresponding scan on the right.

As seen in the above images, this hallway was accurately visualized by the TGN-6909, as it shows all the defining features of the hallway, such as the pillar and the change in height of the room near the gray area.

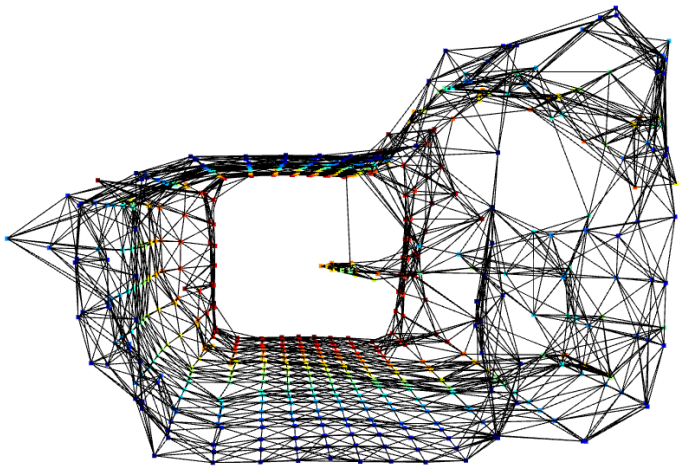


Figure 2 Close-up Front View of the hallway recreated.

5 Limitations

Determining the maximum quantization error for the VL53L1X Time-of-Flight (ToF) module involves employing the ADC error calculation method like so:

$$Error = \frac{Maximum\ Distance}{2^{ADC\ Resolution}} = \frac{4000mm}{2^{12}} = 0.97$$

Furthermore, the data transmission speed of the ToF sensor over I2C is not significantly hindered at the standard protocol speed of 100KHz. This is due to the VL53L1X's maximum capture rate being 50Hz, which does not bottleneck the transmission speed.

To achieve optimal data transmission speed, a baud rate of 115200 BPS was selected. While not the absolute maximum attainable, it represents the highest rate consistently supported across all tested devices via a single UART channel. For the PC utilized in the application of this device, the maximum speed is confirmed to be 128000 BPS, as validated by examining the port properties in the device manager.

The Floating-Point Arithmetic Unit (FPU) embedded within the microcontroller imposes limitations on processing power and capabilities. Direct processing of polar coordinates on the microcontroller is prone to errors and sluggish performance due to these constraints. To overcome this challenge, data is transmitted to Python for processing, capitalizing on its advanced computational capabilities to handle calculations with enhanced accuracy and efficiency.

The primary limitations on the speed of this system are dictated by the time required for the stepper motor to complete all measurements and the duration for the ToF sensor to initialize and begin ranging. While the sensor's operational speed is fixed and beyond user control, adjustments to the motor speed can expedite results. However, this remains the primary bottleneck, as the motor must intermittently pause during rotation to ensure precise measurement readings.

To ascertain the motor's maximum speed without sacrificing accuracy, a comprehensive test was conducted using a binary search algorithm to find a suitable delay. It revealed that the motor can rotate with a delay of 160000 between steps while still maintaining accuracy during measurements to reduce fluctuations due to acceleration.

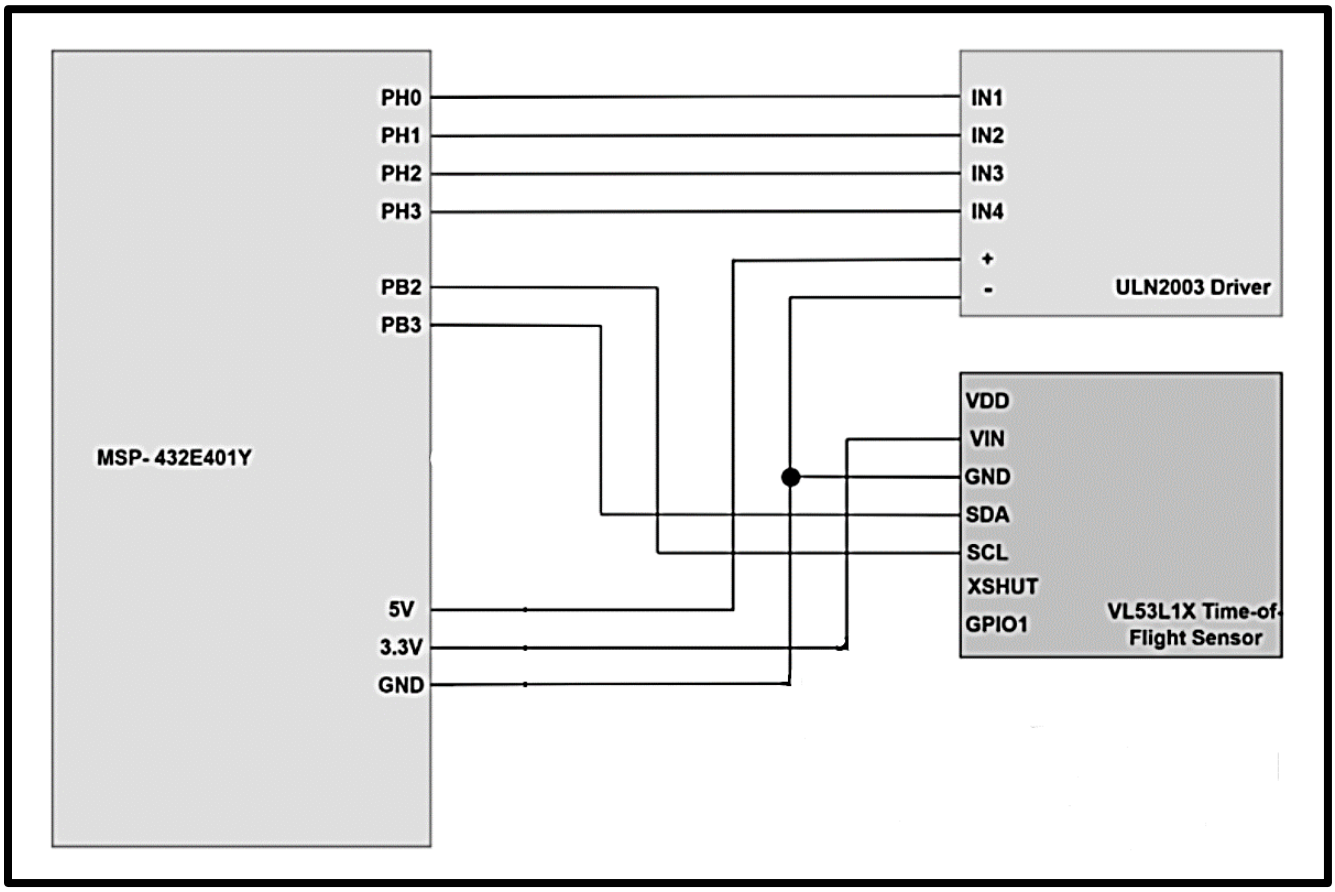
Lastly, the clock speed of the microcontroller may present a slight constraint to this system, although it can be adjusted to reach up to 120MHz, surpassing the requirements for this application. The specific clock speed of this system can be calculated using the formula:

$$Clock\ Speed = 480Mhz \div (PSYDIV + 1)$$

Given the known clock speed of 96MHz, rearranging the equation to solve for the variable PSYDIV reveals that PSYDIV needs to be set to 4. This results in the following:

$$Clock\ Speed = 480Mhz \div (4 + 1) = 96Mhz$$

6 Circuit Schematic



7 Program Logic Flowchart