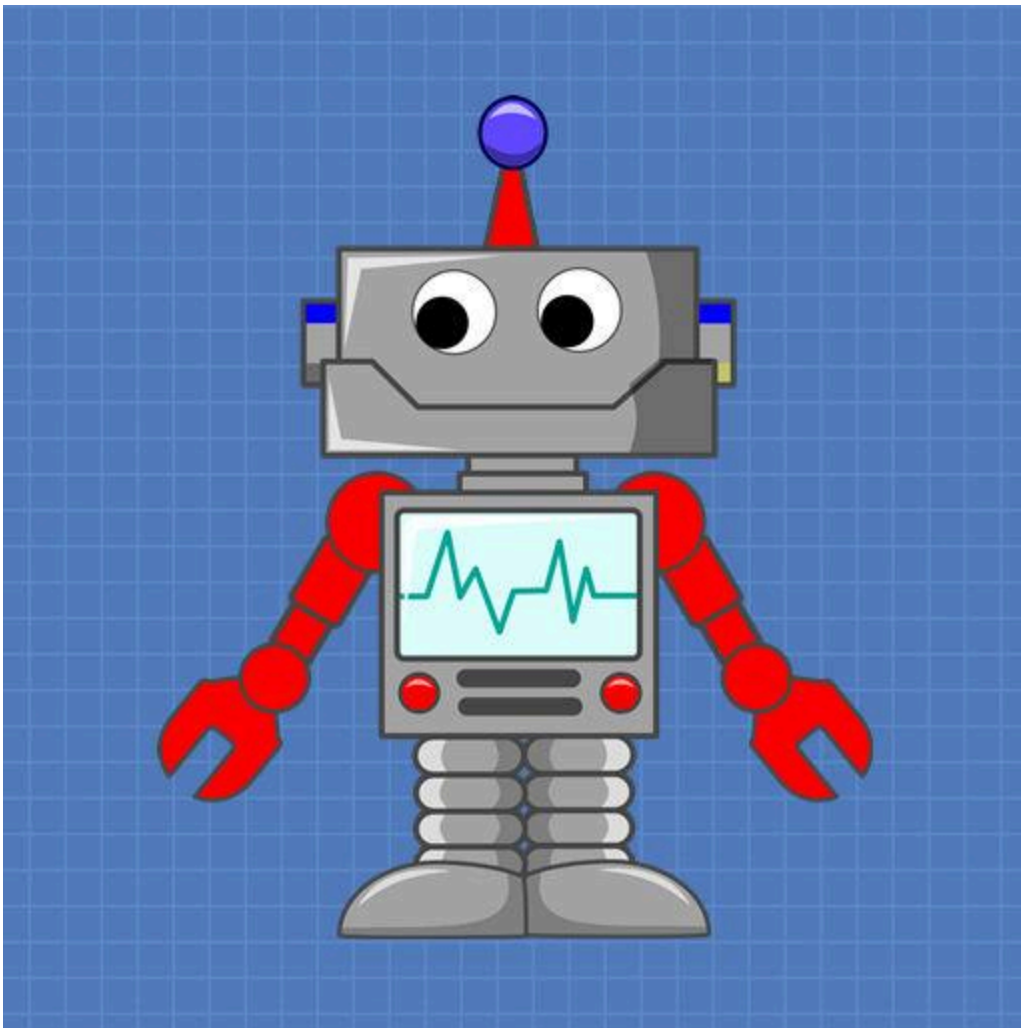


LD2410C Human Presence Detector

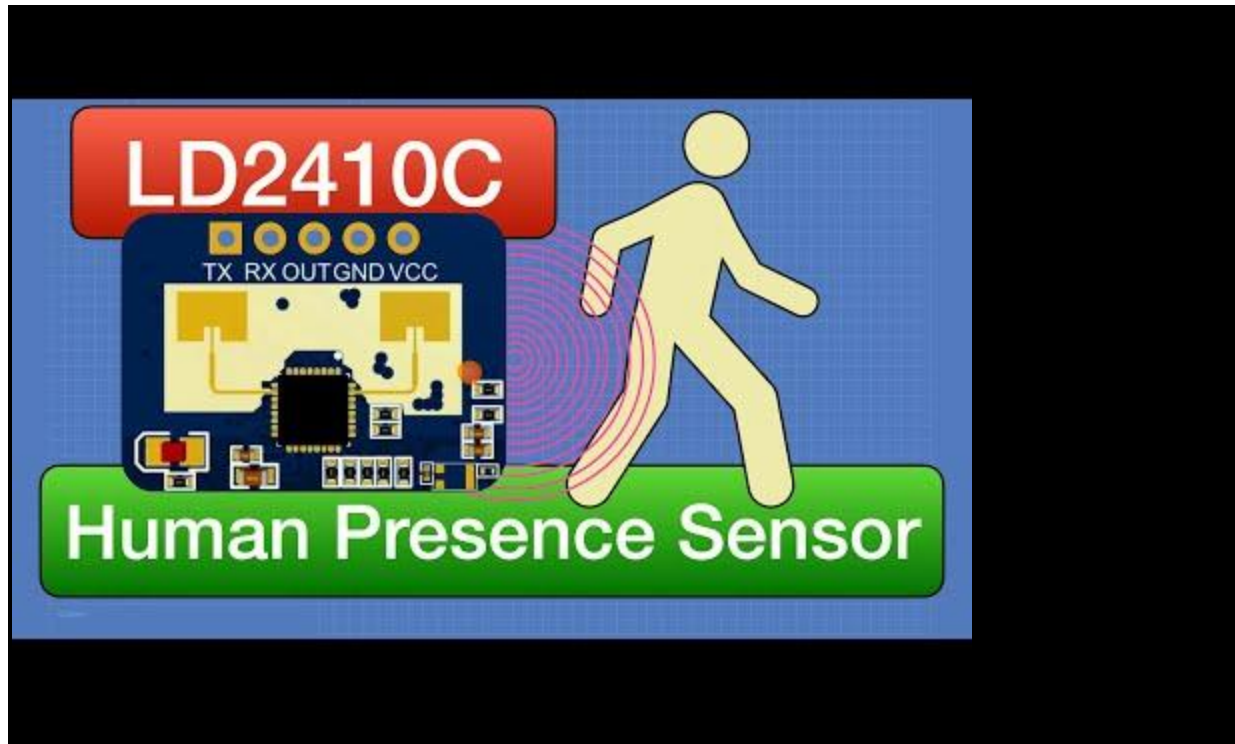


DroneBot Workshop Tutorial

<https://dronebotworkshop.com>

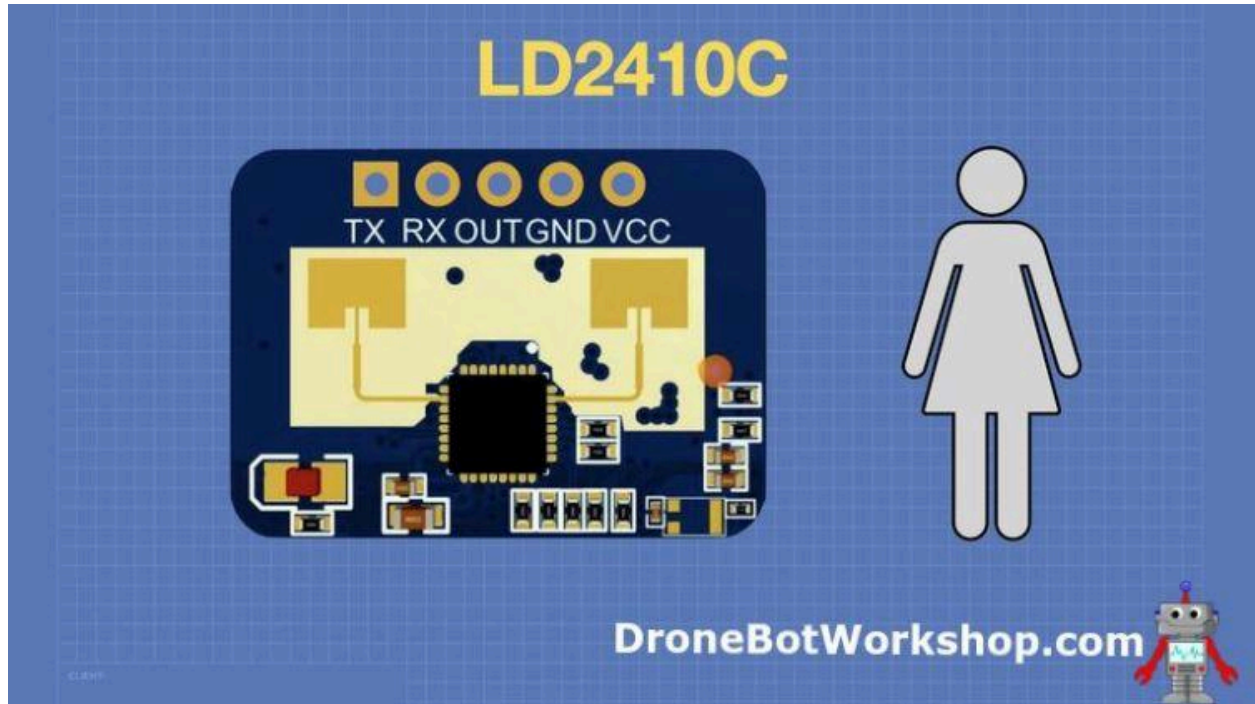
For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

Today, we will be working with a Human Presence Detector, a sensor capable of detecting (what else?) human beings. Unlike PIR sensors, these microwave-based devices are capable of distinguishing humans from animals and other false triggers.



The sensor we will be working with, the LD2410C from Hi-Tech, is an inexpensive device with some very impressive features. You can use it stand-alone, with a Windows or Bluetooth app, or you can connect it to a microcontroller. I'll show you how to do both.

<https://dronebotworkshop.com>



Introduction

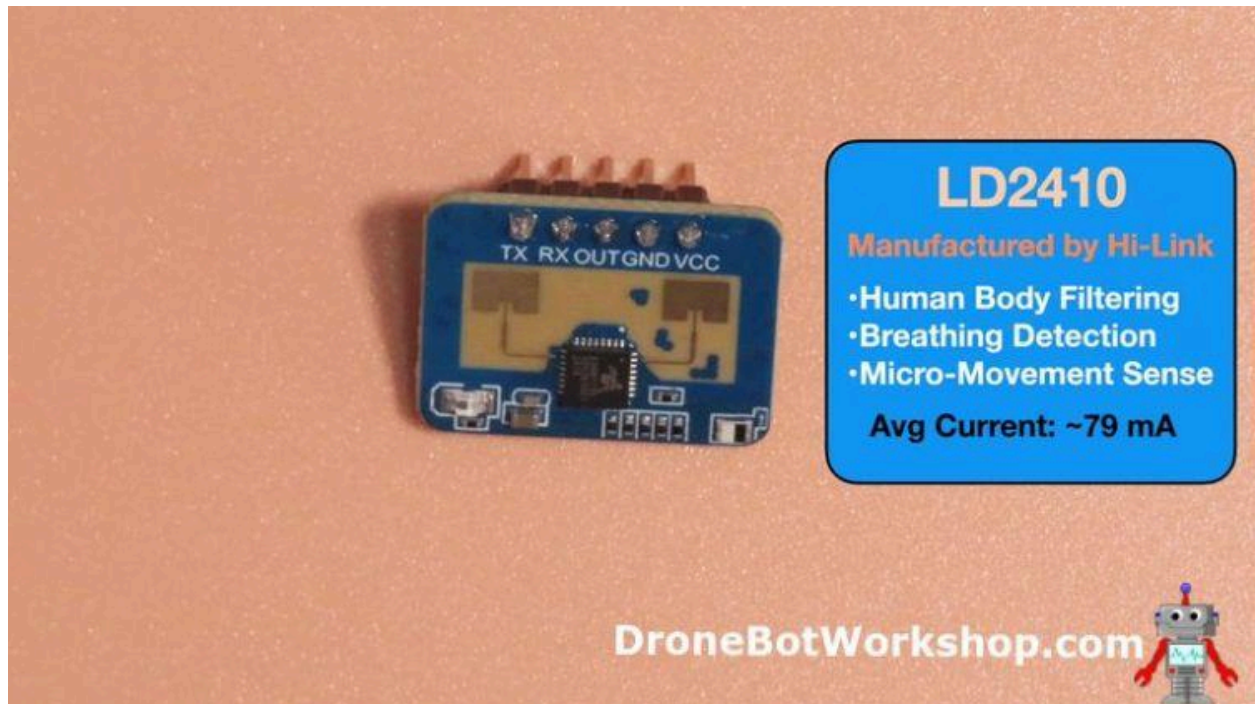
Human presence sensors have a lot of practical applications:

- **Smart Home Automation** – Lighting and Climate control, as well as security systems.
- **Office and Commercial Spaces** – Desk and Meeting Room Occupancy, Energy Management, and Digital Signage Activation are just a few uses for human presence sensors in the office.
- **Automotive Applications** – Cabin Occupancy Detection and Child Presence Detection are new applications for this technology.
- **Healthcare and Elder Care** – Fall Detection Support and tracking patient movement without the invasiveness of a camera are two potential applications.

Unlike PIR (Passive Infrared) sensors, microwave-based Human Presence Sensors don't rely upon body heat. Instead, they can make microscopic measurements of movement and use sophisticated algorithms to recognize human characteristics.

LD2410C Human Presence Detector

The LD2410C is a cost-effective, compact, and low-power human presence sensor. It utilizes 24GHz **Frequency-Modulated Continuous-Wave** (FMCW) radar technology, enabling it to detect both movement and subtle human presence, such as the breathing of a stationary person.



Operating in the 24GHz ISM band, this versatile millimeter-wave radar module boasts a wide detection angle of ± 60 degrees and can sense human presence up to 5 meters away. Unlike PIR sensors, which rely on changes in infrared radiation for motion detection, the LD2410C's FMCW radar technology enables comprehensive human presence sensing, making it easy to integrate into various security and home automation applications.

Here are the key specifications of the LD2410C Human Presence Sensor:

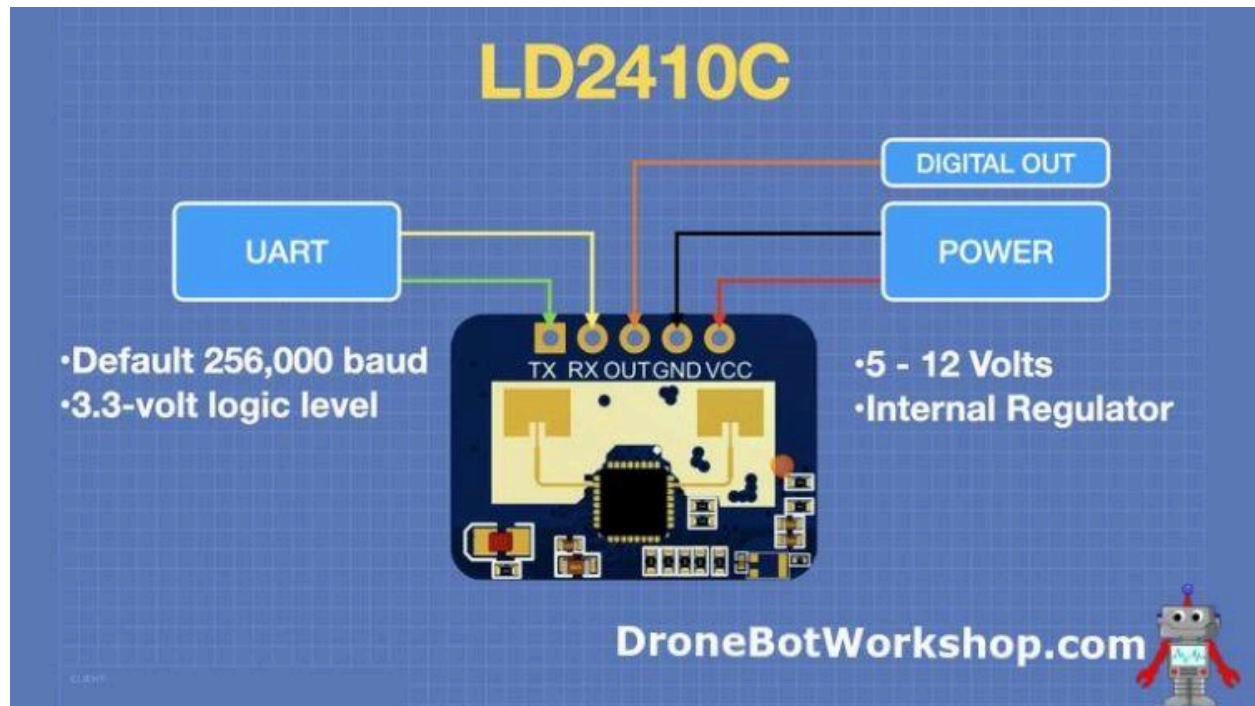
For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

- Operating Frequency: 24GHz (ISM band)
- Detection Range: 0.5m to 6m (adjustable)
- Operating Voltage: 3.3V to 5V DC
- Current Consumption: <100mA
- Communication Interface: UART (256,000 baud default)
- Detection Angle: 60° cone
- Operating Temperature: -40°C to +85°C
- Dimensions: 22mm × 16mm × 3mm

This inexpensive module is available from Amazon as well as many electronics distributors.

LD2410C Pinouts

The LD2410C is a small module, measuring just 22mm x 16mm. It has five connections, as illustrated here:



- TX – UART Transmit
- RX – UART Receive
- OUT – Digital Output
- GND – Ground
- VCC – 5 to 12 VDC

Note that the UART pins are 3.3-volt logic and are not 5-volt tolerant. The device features an onboard voltage regulator and accepts power ranging from 5 to 12 DC volts.

The UART default speed is 256,000 baud, with 8 bits, one stop bit, and no parity.

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

The Digital Output is a 3.3-volt logic GPIO pin. It will go HIGH when a human is detected within a preset distance.

LD2410 Detector Family

The LD2410C is part of the LD2410 family developed by Hi-Link (HLK), which offers several radar-based sensors tailored for different presence-detection scenarios.

The LD2410 family has three members:

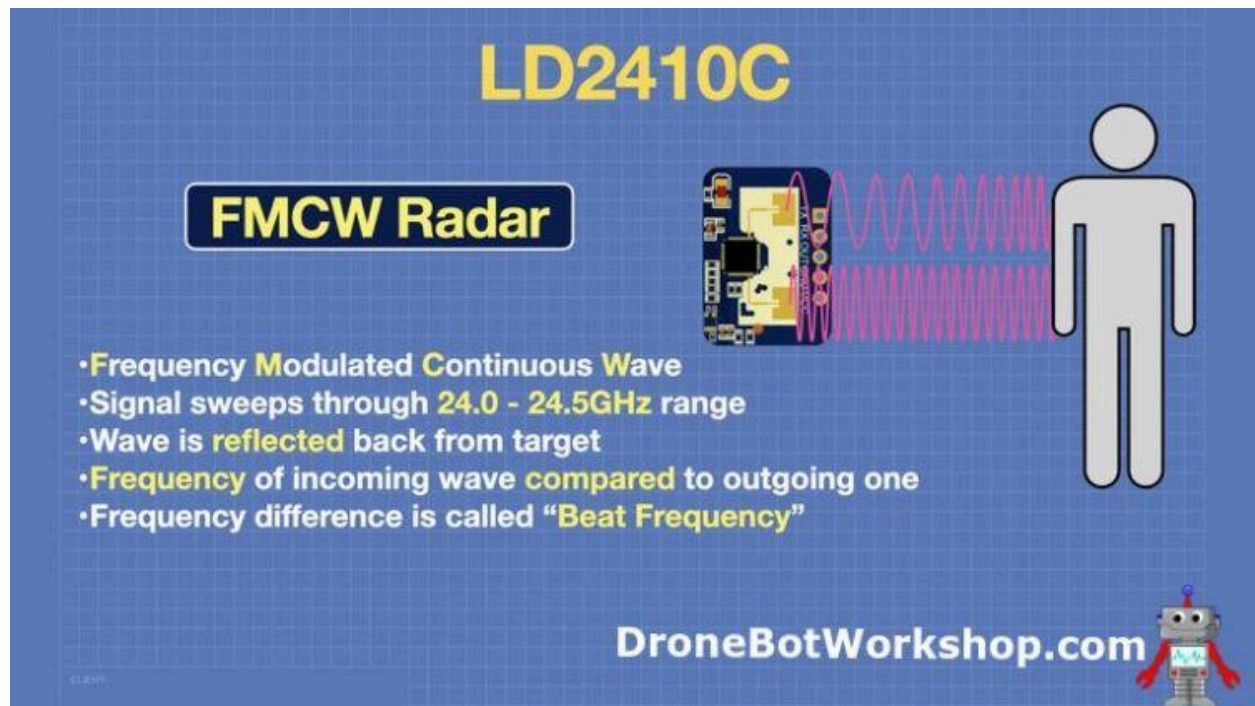
- **LD2410** – Original version with basic UART interface.
- **LD2410B** – An improved version with Bluetooth built in.
- **LD2410C** – Enhanced model combining UART and Bluetooth with more stable firmware and better configuration options.

The original LD2410 and its successor, the LD2410B, laid the groundwork for the technology. A key physical difference in the LD2410C is its use of a standard 2.54mm pin pitch, a welcome change for hobbyists who often found the 1.27mm pitch of the earlier models challenging to work with on standard breadboards and perfboards.

All three models are equipped with UART communication capabilities, allowing them to interface with a host computer or microcontroller. Additionally, the LD2410B and LD2410C models feature Bluetooth connectivity, complemented by Hi-Link's HLKRadar app, available for both Android and iOS devices.

FMCW Radar

FMCW (Frequency-Modulated Continuous-Wave) radar is a type of radar technology that transmits a continuous wave signal with a frequency that varies linearly over time. This frequency modulation allows the radar to measure the distance and velocity of targets.



The diagram illustrates the LD2410C FMCW Radar system. It features a blue background with a grid pattern. At the top center, the text "LD2410C" is displayed in large yellow letters. Below it, on the left, is a black box with the text "FMCW Radar" in yellow. To the right of this box is a small image of the LD2410C radar module. Further right, a series of pink wavy lines represent the radar signal being transmitted towards a grey human figure, which represents the target. Below the radar module, there is a list of bullet points in yellow text: "•Frequency Modulated Continuous Wave", "•Signal sweeps through 24.0 - 24.5GHz range", "•Wave is reflected back from target", "•Frequency of incoming wave compared to outgoing one", and "•Frequency difference is called 'Beat Frequency'". At the bottom right, the text "DroneBotWorkshop.com" is written in white, with a small robot icon next to it.

LD2410C

FMCW Radar

- Frequency Modulated Continuous Wave
- Signal sweeps through 24.0 - 24.5GHz range
- Wave is reflected back from target
- Frequency of incoming wave compared to outgoing one
- Frequency difference is called "Beat Frequency"

DroneBotWorkshop.com

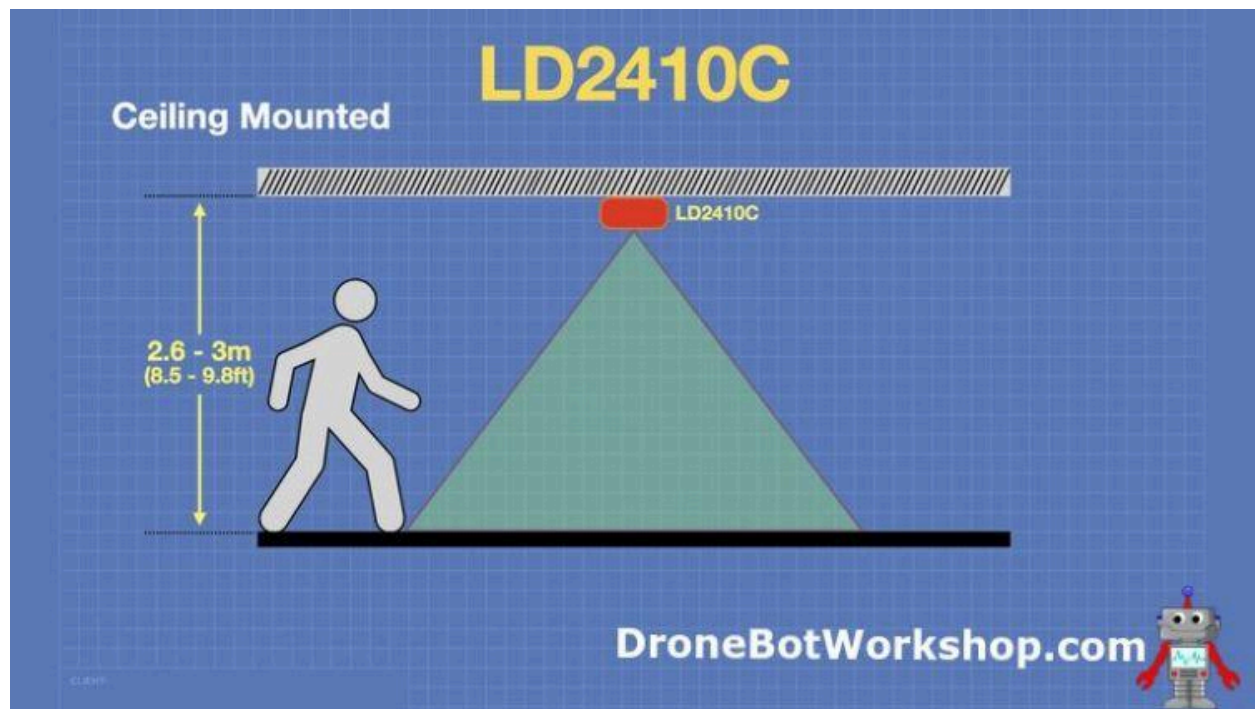
The basic principle of FMCW radar is as follows:

- The radar transmits a frequency-modulated signal towards the environment. The frequency varies from 25 GHz to 25.25 GHz.
- The signal reflects off objects in the environment and returns to the radar.
- The returned signal is mixed with the transmitted signal, producing a *beat frequency* that is proportional to the distance and velocity of the target.

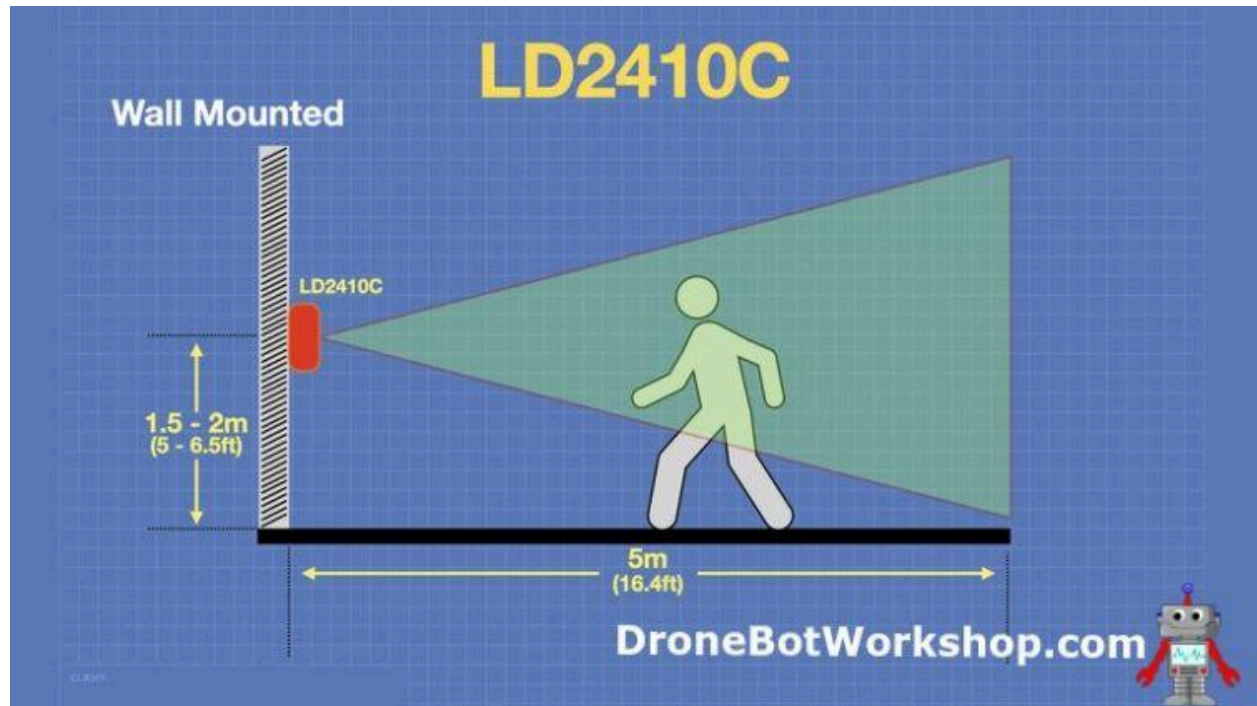
- By analyzing the beat frequency and time delay between the transmitted and reflected signals, the sensor calculates the distance to the target.

By varying the transmitted frequency, we gain many advantages over conventional radar, which uses a fixed carrier frequency.

Continuous-wave Radar without frequency modulation only detects moving targets, as stationary targets will not cause a Doppler shift.



However, FMCW Radar is also capable of determining distance, which increases reliability by providing both distance and speed measurements. So we can pick up both moving and stationary humans.



For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

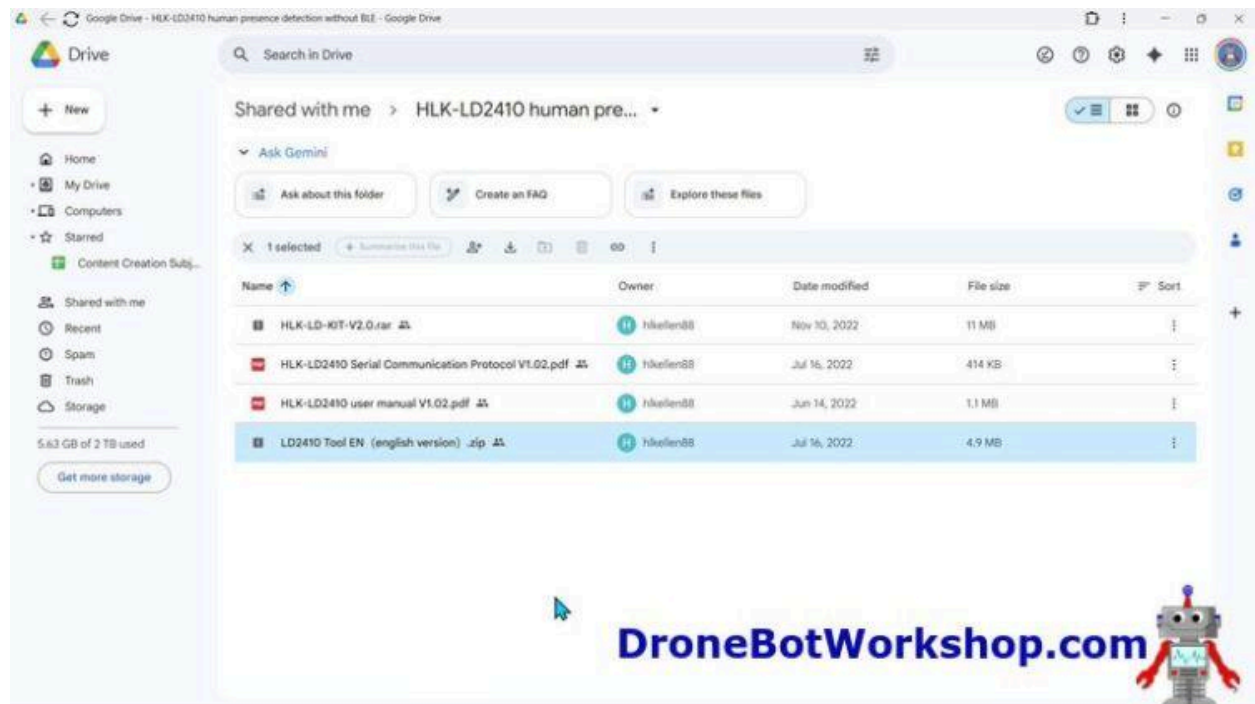
LD2410 with LD2410-Tool Windows GUI

The LD2410-Tool by Hi-Link is a Windows GUI application used for programming and testing the LD2410 family of Human Presence Sensors. It is only available for Windows.

To use the tool, you'll need to accomplish two things:

- Download the tool from Hi-Link and install it on your Windows computer.
- Connect the LD2410C to the Windows computer.

The first step, downloading the tool, involves [accessing a shared Google Drive folder](#) and downloading the ZIP file. After that, you can uncompress the ZIP file and run the tool from the EXE file. There is no installation required.



On one of my Windows 11 computers, I encountered an error indicating a missing DLL when attempting to run the EXE file. I resolved this by downloading the [Microsoft Visual](#)

<https://dronebotworkshop.com>

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

[Studio 10 Runtime](#) and installing it. It resolved the DLL error, and the GUI loaded properly.

LD2410-Tool Windows GUI Hookup

We will need to connect our human presence sensor to the Windows 11 computer using a USB cable; however, the LD2410C only has a serial UART connection. So we'll need a method of converting the UART connection to USB.

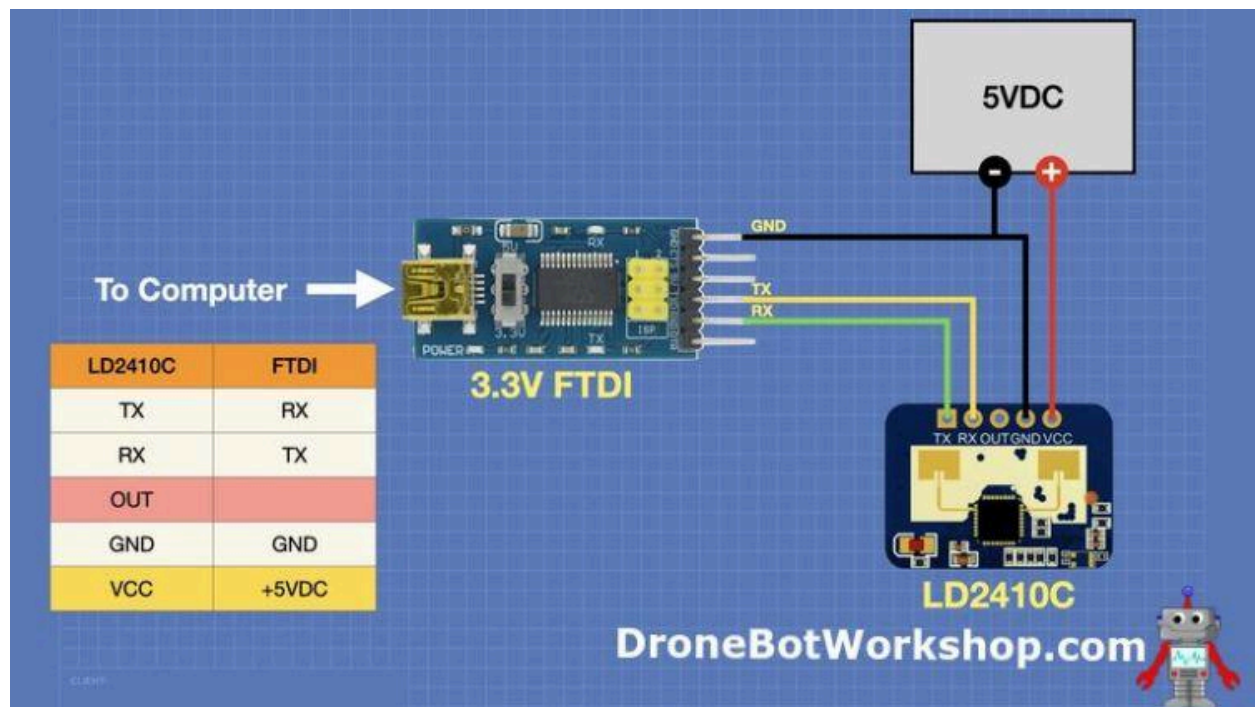
This is a common task, and there are several ways to accomplish it. We will be using an FTDI adapter, as most people will likely have one or more of these in their parts drawers.

The hookup is pretty basic, but one thing we need to contend with is logic and power levels. Most FTDI adapters can be configured (through a strap, switch, or jumper) to work at either 3.3 or 5 volts. For the LD2410C, we want to use 3.3-volt logic, so **the FTDI adapter must be set to 3.3 volts**.

However, the LD2410C requires a power supply of 5 to 12 volts DC. If we set our FTDI adapter for 3.3 volts, then it will only output 3.3 volts. So it won't work.

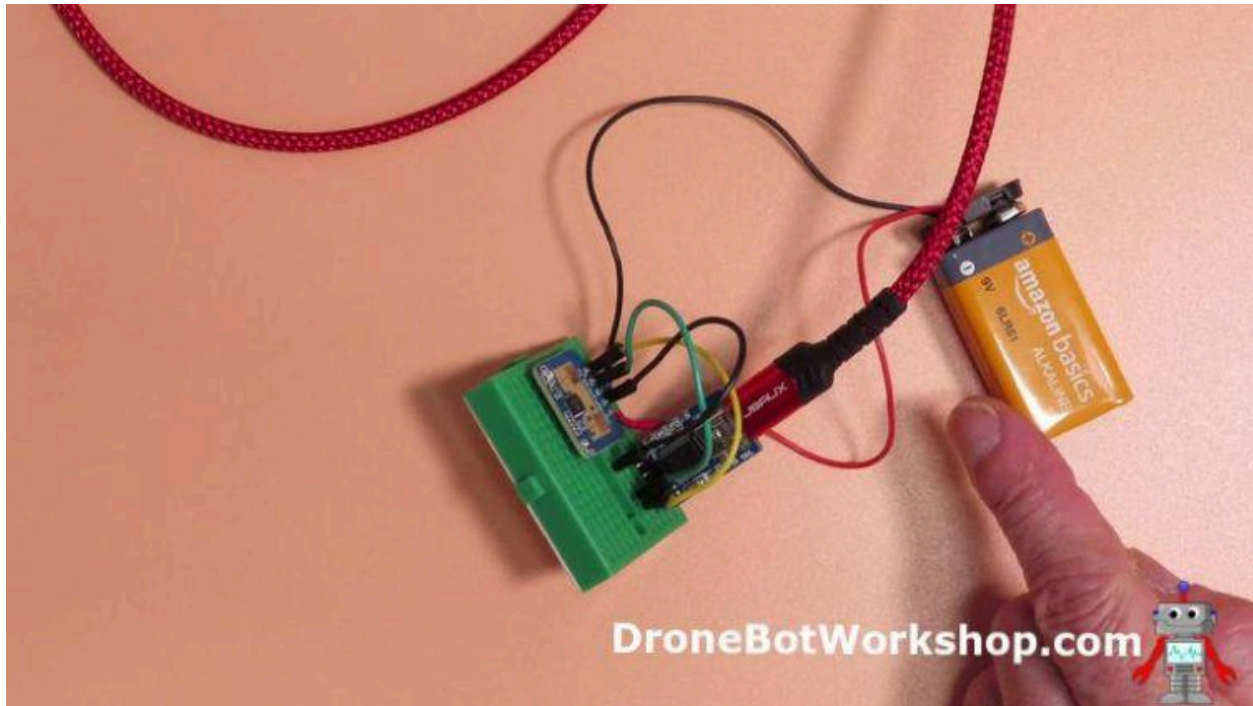
The solution is to use a separate power supply, anything from 5 to 12 volts with at least 100ma of current capability will work. In my experiments, I used a 9-volt battery; however, as the LD2410C consumes a significant amount of current, this is not recommended as a permanent solution.

Here is the hookup, with an independent 5-volt power supply shown:



Ensure that the ground of the power supply is connected to the LD2410C ground. The transmit and receive are reversed between the FTDI adapter and the human presence sensor.

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

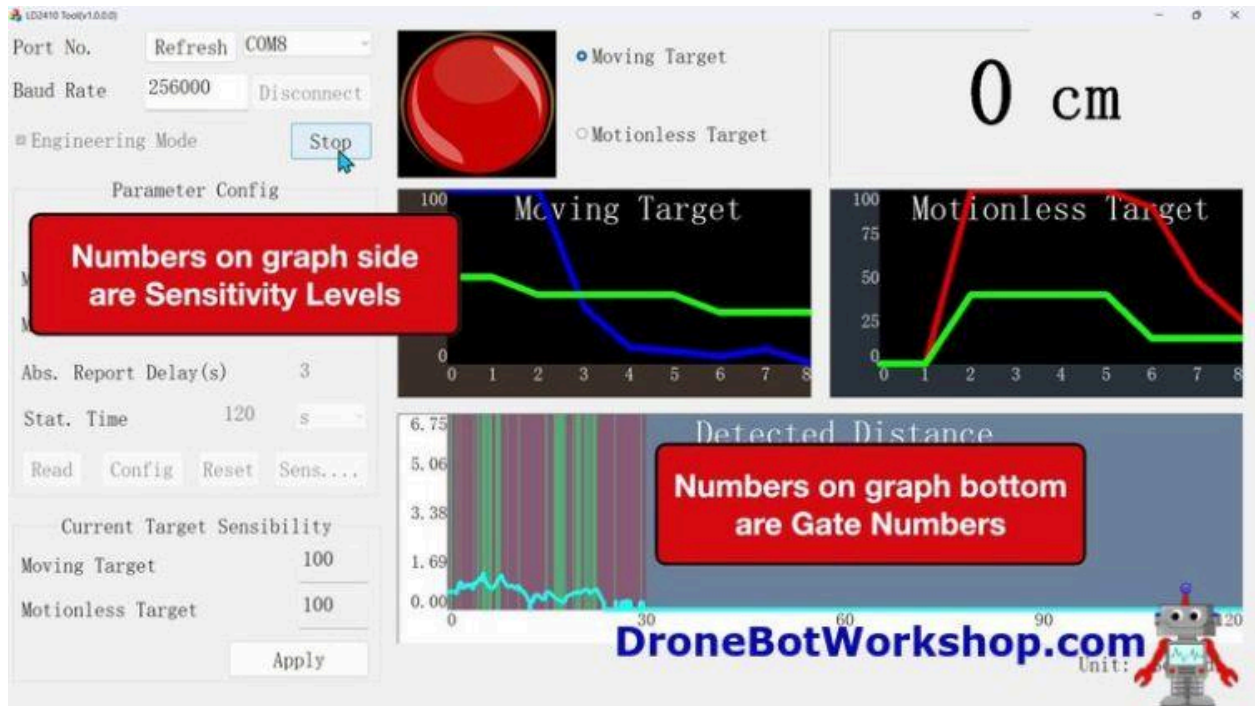


LD2410-Tool Windows GUI Demo

With the LD2410C & FTDI connected to the computer and the GUI files extracted, launch the GUI by clicking on the EXE file. The GUI display should launch.

<https://dronebotworkshop.com>

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>



Look in the upper left corner and note the COM Port. This is a drop-down menu that you must use to select the correct COM port for your LD2410C. If you have multiple choices, then you can unplug and plug in the LD2410C to determine which one to use. Another way to determine the correct COM port is to use the Windows Device Manager.

If your FTDI COM port does not appear, try clicking the *Refresh* button.

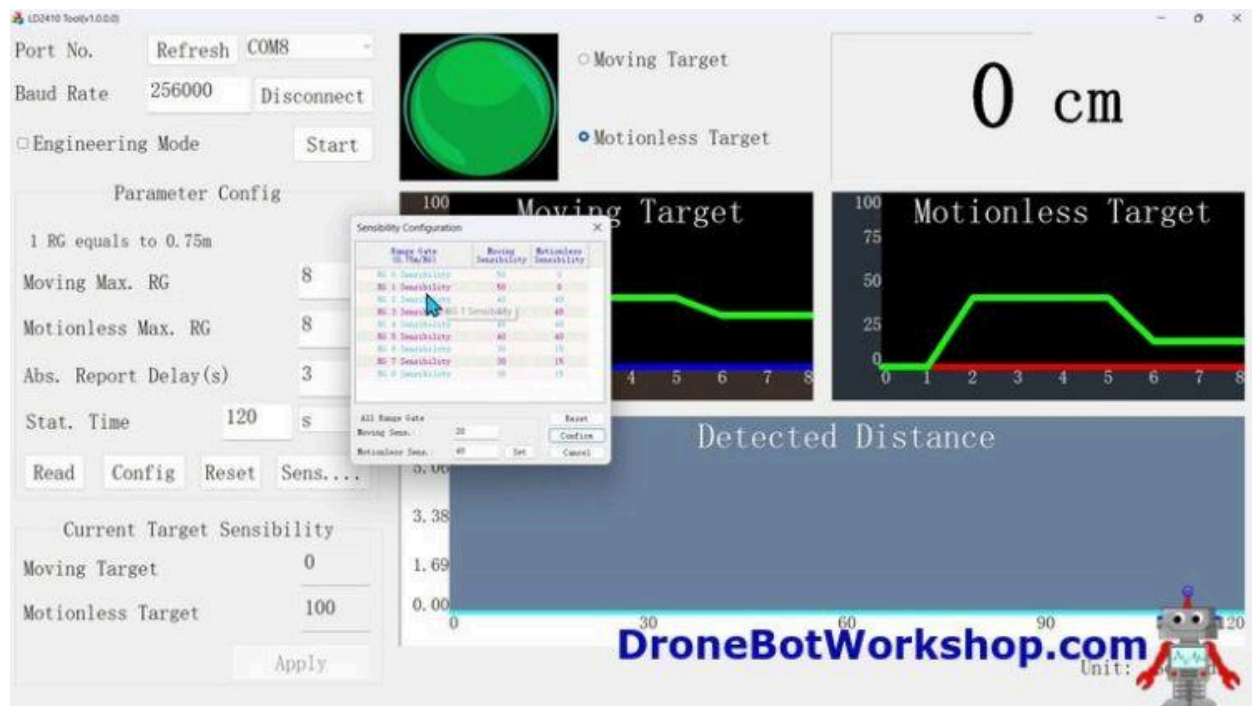
Set the Baud Rate to 256,000, which is the default setting. Now click *Start* to connect. You should see live distance and presence data in the real-time panel.

Note the checkbox labeled “Engineering Mode”. If this is checked, you can modify the LD2410C parameters using this GUI tool. If you exit Engineering Mode, then you can only view the sensor parameters; you can’t change them. You can view the sensor data in both modes.

<https://dronebotworkshop.com>

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

In the *Parameter Config* area, you can adjust several LD2410C parameters. You need to be in Engineering Mode to perform this action, and you must remember to save your settings using the *Apply* button located at the bottom of the interface.



The Sensitivity Configuration is probably the most common adjustment to make. It allows you to set the sensitivity of the nine gates. Remember, each gate reacts to a specific distance, so by adjusting the sensitivity of the gates, you can “fine-tune” the LD2410C detection area to suit your requirements.

The GUI has two graphs, one for a Moving Target and the other for a Motionless Target. You can see the numbers 0 to 8 at the bottom of each graph, which represent the nine gates in the LD2410C.

Experiment with the sensor out of Engineering Mode before making any settings changes, and remember what you have set so you can reset it later if necessary.

<https://dronebotworkshop.com>

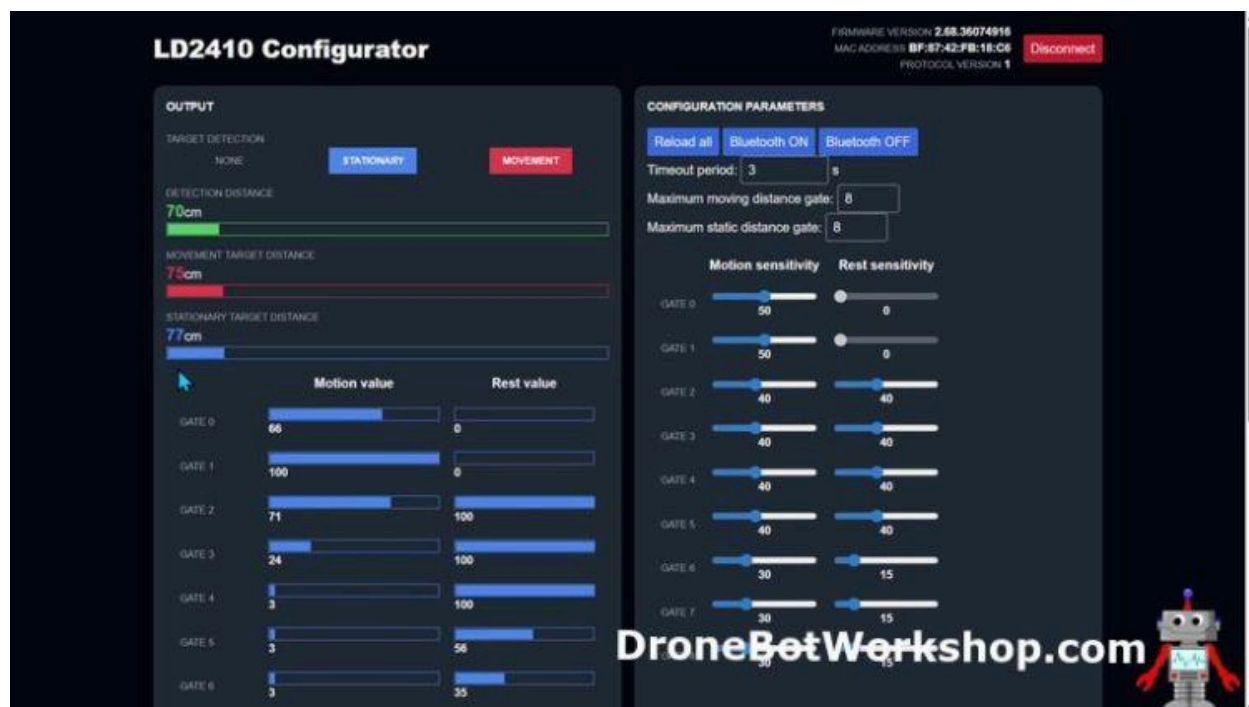
For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

LD2410 Configurator

Another method of testing and adjusting the LD2410C is to use an online utility called the *LD2410 Configurator*. This tool can be used from any computer, regardless of operating system.

To use the LD2410 Configurator, you will need to connect the LD2410C to the computer via an FTDI adapter, just as we did with the Windows GUI. Once you have done that, just visit the application at <https://ld2410.albert.nz/>.

The interface you are greeted with couldn't be more straightforward: a blue button that prompts you to select a port. Click it and select the USB port to which your human presence sensor is attached. The "advanced connection options" are just the baud rate, which defaults to the standard 256,000 baud.



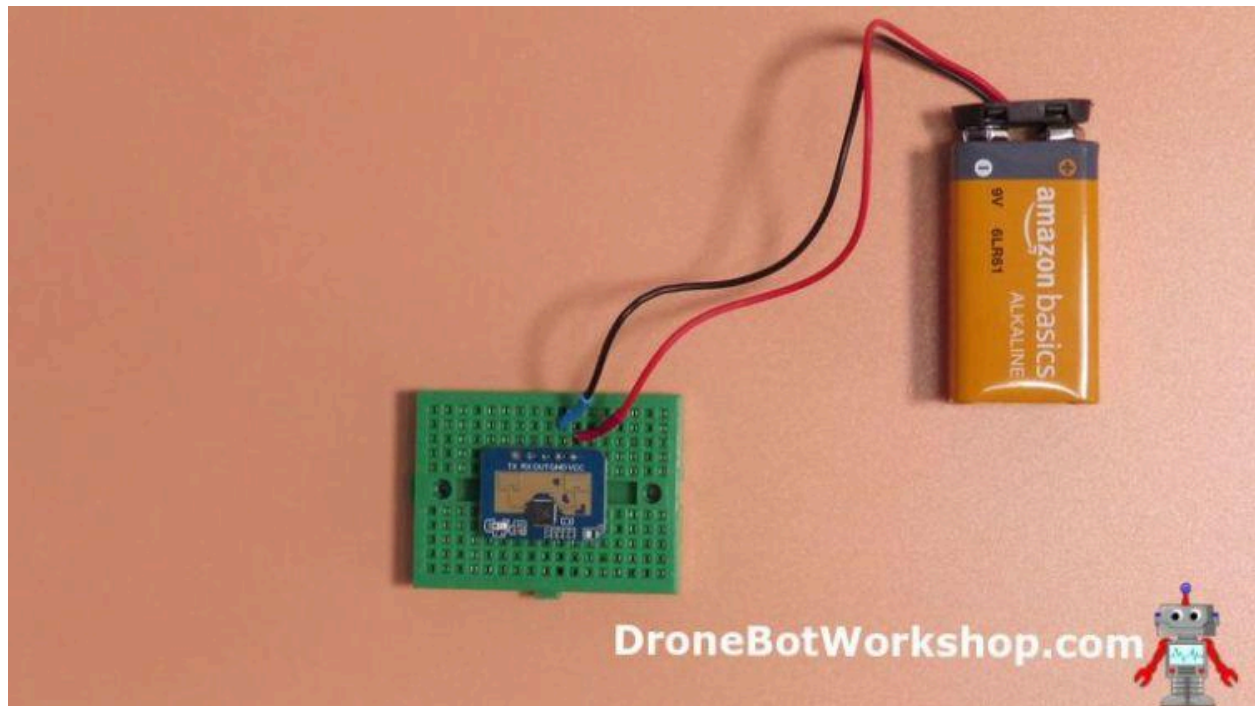
For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

Once you have established a connection to the LD2410C, you will be greeted with a display of the activity on each gate. You also have options to set gate sensitivity, just as with the Windows GUI tool.

The LD2410 Configurator is a fine example of a useful open-source utility. Check it out, for simply configuring the sensor, it is very easy to use.

LD2410C with Bluetooth

The LD2410C sensor features built-in Bluetooth Low Energy (BLE) capability, enabling wireless connection using the *HLKRadarLayout* Android or iOS app. This is especially useful for configuring or monitoring the sensor after it has been installed in a fixed location (such as inside a wall, ceiling fixture, or enclosure) where wired access may be inconvenient or impossible. With Bluetooth enabled, you can adjust settings and view live radar data without needing to disconnect the sensor from your project.



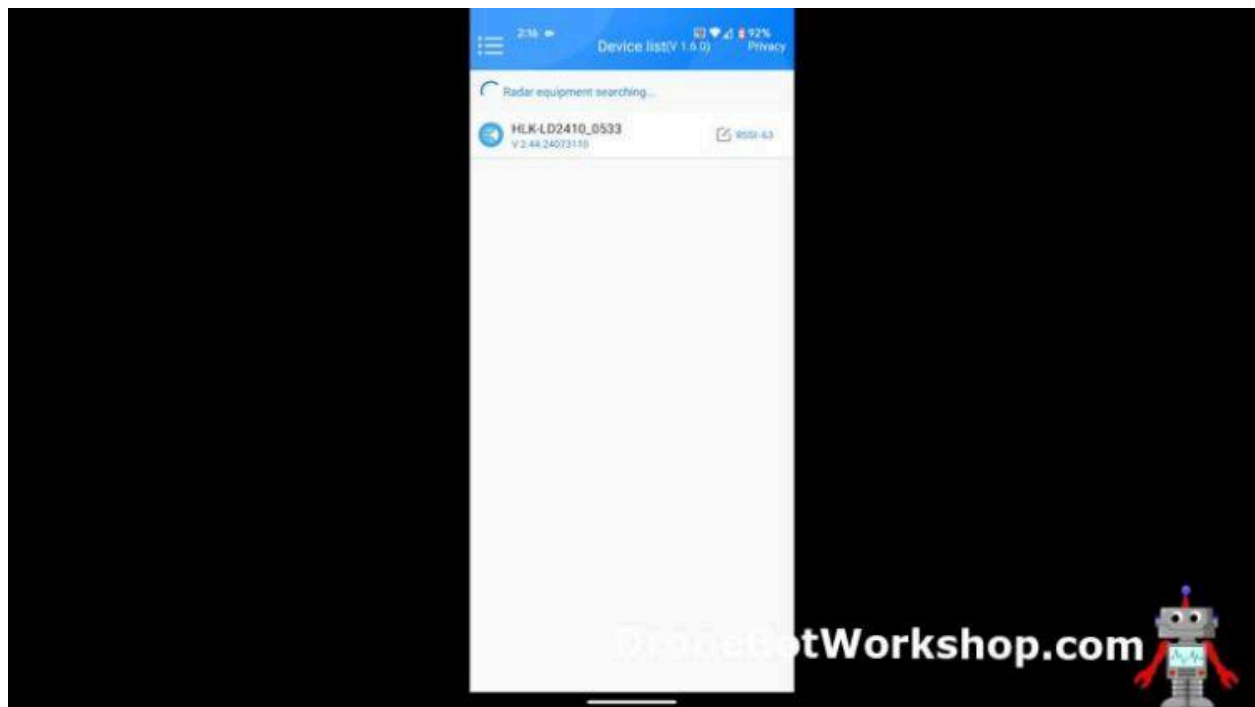
To use the LD2410C with the HLKRadarLayout App, simply provide it with 5 to 12 volts of power. For testing, I used a 9-volt battery; however, this is not intended as a permanent power solution, as it has inadequate current capability.

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

HLKRadarTool App

The HLKRadarTool App is available for both Android and iOS. Visit your app store and download the version appropriate for your phone or tablet.

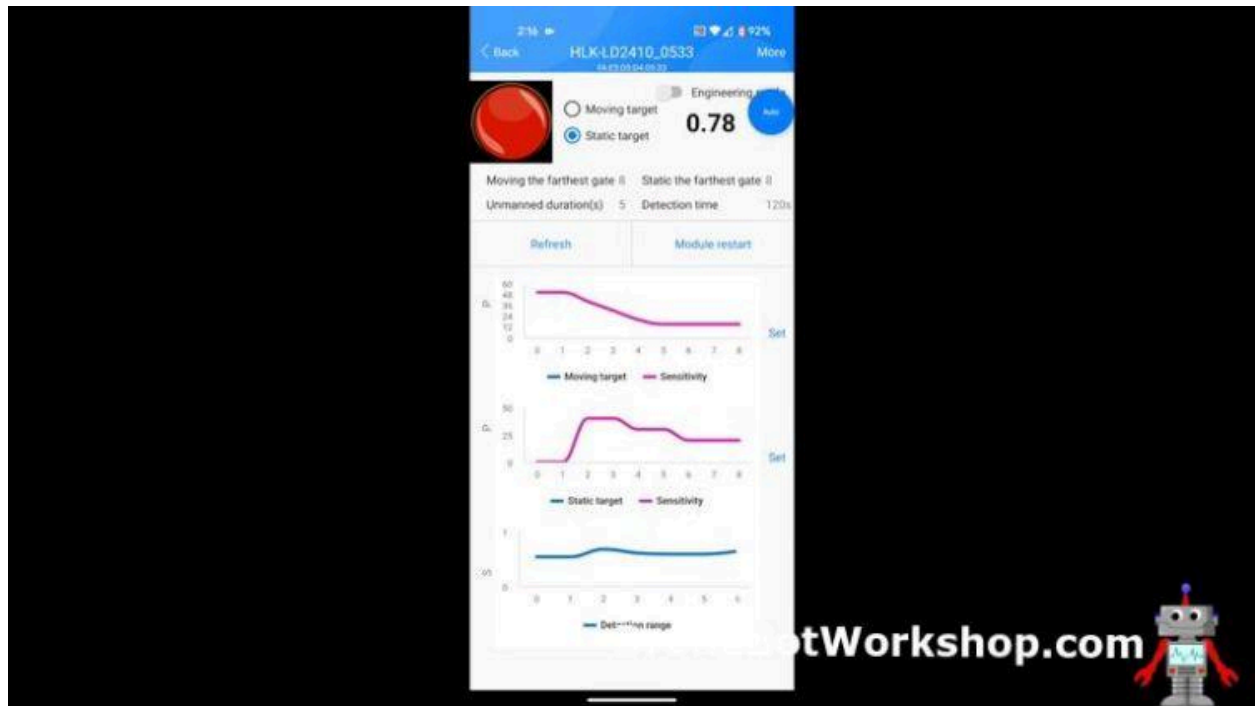
Once you have installed the app and powered up the LD2410C, you are ready to proceed. Open the app, and you should see it scanning for compatible radar devices. After a few seconds, it should find the LD2410, which will be listed in the format *"HLK-LD2410_xxxx"*.



Select the device and you'll be taken to the main screen. This provides a display similar to the Windows GUI, and you can use it to test the sensor.

<https://dronebotworkshop.com>

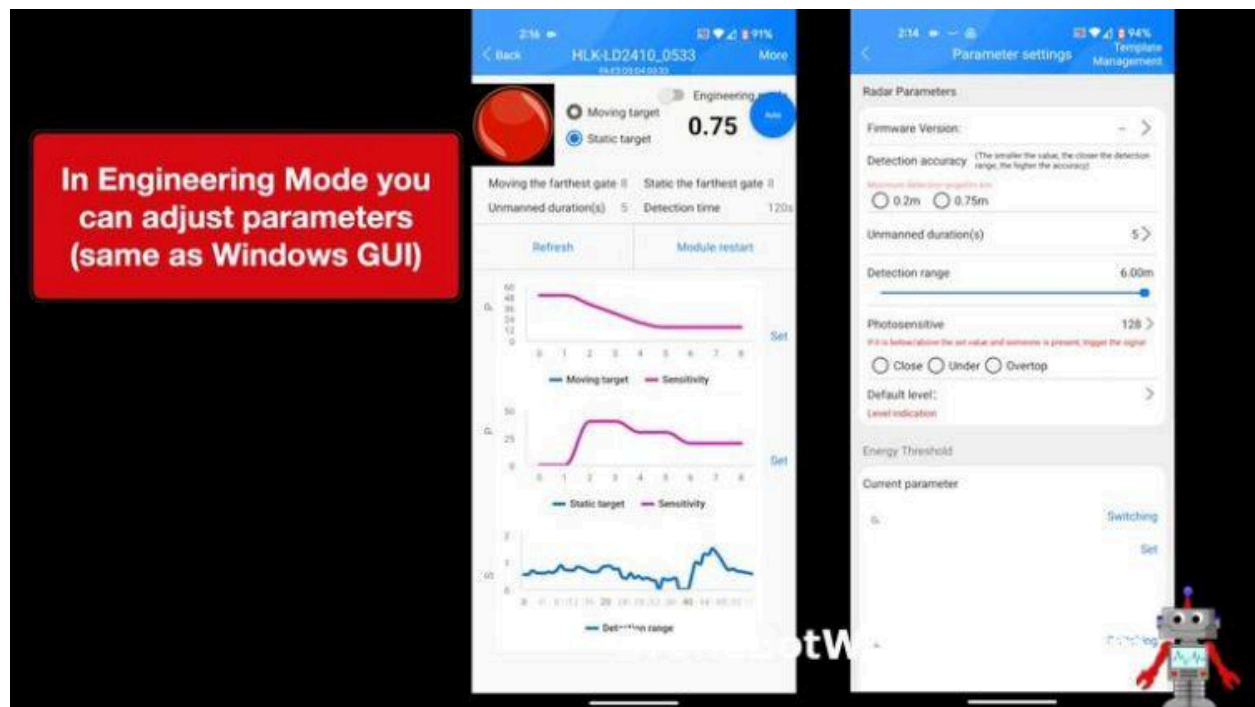
For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>



The app also features a switch to enable Engineering Mode. Similar to the Windows GUI tool, this mode allows you to adjust the configuration settings of the LD2410C human presence sensor.

<https://dronebotworkshop.com>

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>



The HLKRadarTool provides a seamless way to configure and monitor the LD2410C wirelessly, making it a great companion for smart home, security, and automation projects.

<https://dronebotworkshop.com>

The image displays two screenshots of the HLK-LD2410_0533 application interface, which is used for configuring and monitoring the LD2410 module.

Left Screenshot: The interface shows the 'Engineering' mode. A large red button labeled 'Auto' is visible. The 'Moving target' option is selected, and the value '0.60' is displayed. Below this, there are two tabs: 'Moving the farthest gate' and 'Static the farthest gate'. The 'Unmanned duration(s)' is set to 5, and the 'Detection time' is 120s. There are 'Refresh' and 'Module restart' buttons. Three line graphs are shown: 'Moving target' vs 'Sensitivity', 'Static target' vs 'Sensitivity', and 'Detection range'.

Right Screenshot: The interface shows the 'Parameter settings' menu. The 'Initialize' option is selected, and the value '0.30' is displayed. Below this, there are two tabs: 'Moving the farthest gate' and 'Static the farthest gate'. The 'Unmanned duration(s)' is set to 5, and the 'Detection time' is 120s. There are 'Refresh' and 'Module restart' buttons. A 'System prompt' dialog box is displayed, stating: 'Bottom noise detection is turned on successfully, the module will automatically enter the detection after 10S, please make sure that there is no moving target in the detection environment, so as not to affect the detection results!'. The dialog has 'Cancel' and 'Confirm' buttons. A watermark 'otWorkshop.com' and a robot icon are visible at the bottom right.

<https://dronebotworkshop.com>

LD2410C with ESP32

The LD2410C features a single GPIO pin, making it suitable for stand-alone applications that require switching on or off when a human is detected. But if you want to do something more advanced than that, you'll need to interface the LD2410C with a microcontroller (or microcomputer).

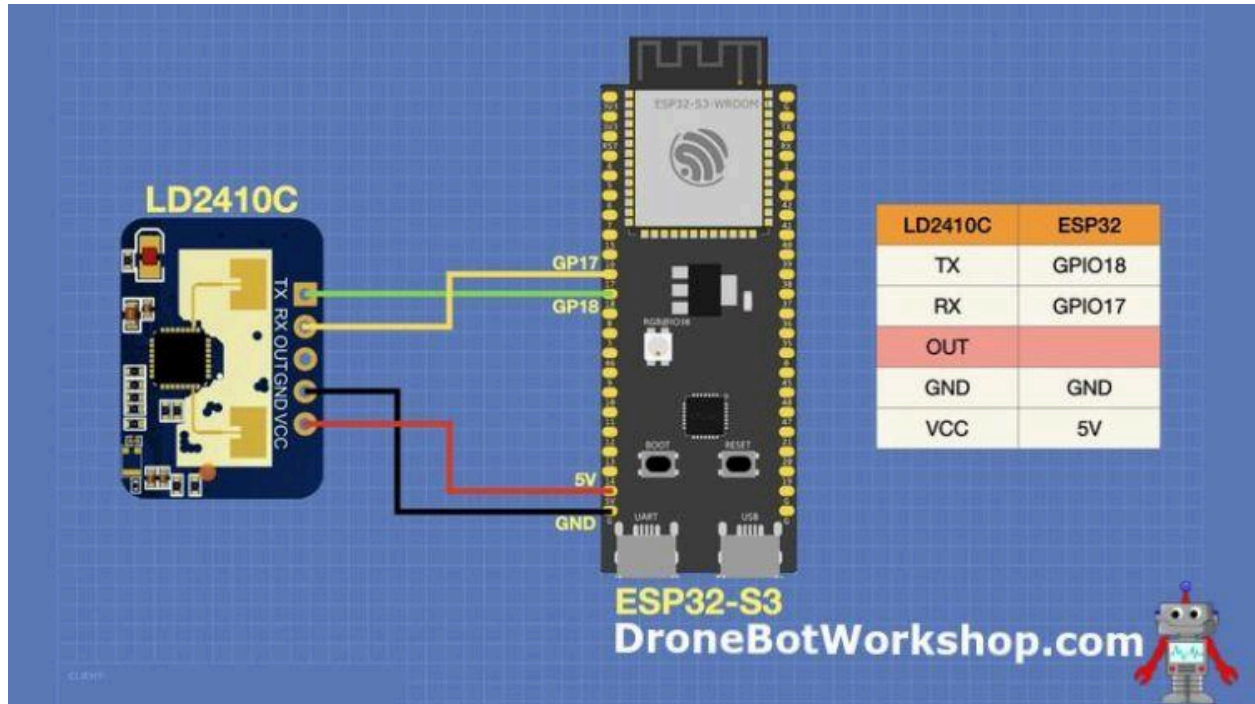
By connecting a microcontroller to the UART we can extract the data from the LD2410C and use it in our code.

We will be experimenting with an ESP32, but several other microcontrollers are also suitable for the job.

LD2410C with ESP32 Hookup

I'll be using an ESP32-S3 Dev Kit as my microcontroller, but feel free to substitute a different ESP32 board if you prefer. If your chosen ESP32 board has different available pins than the ones I'm using, you'll need to adjust the code accordingly.

Here is how I hooked up the LD2410C Human Presence Sensor to an ESP32-S3 Dev Kit:



The ESP32's 5 volt output is used to power the human presence sensor.

MyLD2410 Library

We will be using a library to simplify coding, [the MyLD2410 Library](#). This library offers a straightforward and efficient method for interfacing with the LD2410C presence sensor from an ESP32 or other Arduino-compatible microcontroller.

One great thing is that all of the functions and methods of the library have been [documented on GitHub](#). This makes coding with this library a breeze.

The library is available in the Arduino IDE Library Manager, so head over there and search for "MyLD2410". Install the library and then take a look at the selection of example sketches included with it. You'll find examples for reading data, as well as for programming the LD2410C's internal parameters.

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

Let's run one of those examples with our ESP32 hookup.

LD2410C with ESP32 Code

The example we will be using is the “*sensor_data*” example. This sketch exposes many of the library functions.

This sketch illustrates how to use many of the library functions to retrieve data from the LD2410C human presence sensor, including the value of the module's onboard light sensor.

```
/*  
  
  This program reads all data received from  
  
  the HLK-LD2410 presence sensor and periodically  
  
  prints the values to the serial monitor.  
  
  
  Several #defines control the behavior of the program:  
  
  #define SERIAL_BAUD_RATE sets the serial monitor baud rate  
  
  #define ENHANCED_MODE enables the enhanced (engineering)  
  mode of the sensor. Comment that line to switch to basic mode.  
  
  #define DEBUG_MODE enables the printing of debug information  
  (all received frames are printed). Comment the line to disable  
  debugging.  
  
  
  Communication with the sensor is handled by the  
  "MyLD2410" library Copyright (c) Iavor Veltchev 2024  
  
  
  Use only hardware UART at the default baud rate 256000,  
  or change the #define LD2410_BAUD_RATE to match your sensor.  
  
  For ESP32 or other boards that allow dynamic UART pins,  
  modify the RX_PIN and TX_PIN defines  
  
  
  Connection diagram:  
  
  Arduino/ESP32 RX  -- TX LD2410  
  Arduino/ESP32 TX  -- RX LD2410  
  Arduino/ESP32 GND -- GND LD2410  
  
  Provide sufficient power to the sensor Vcc (200mA, 5-12V)  
  
*/  
  
#if defined(ARDUINO_SAMD_NANO_33_IOT) || defined(ARDUINO_AVR_LEONARDO)
```

```
//ARDUINO_SAMD_NANO_33_IOT RX_PIN is D1, TX_PIN is D0

//ARDUINO_AVR_LEONARDO RX_PIN(RXI) is D0, TX_PIN(TXO) is D1

#define sensorSerial Serial1

#elif defined(ARDUINO_XIAO_ESP32C3) || defined(ARDUINO_XIAO_ESP32C6)

//RX_PIN is D7, TX_PIN is D6

#define sensorSerial Serial0

#elif defined(ESP32)

//Other ESP32 device - choose available GPIO pins

#define sensorSerial Serial1

#if defined(ARDUINO_ESP32S3_DEV)

#define RX_PIN 18

#define TX_PIN 17

#else

#define RX_PIN 16

#define TX_PIN 17

#endif

#else

#error "This sketch only works on ESP32, Arduino Nano 33IoT, and Arduino Leonardo (Pro-Micro)"

#endif

// User defines

// #define DEBUG_MODE

#define ENHANCED_MODE

#define SERIAL_BAUD_RATE 115200

//Change the communication baud rate here, if necessary

//#define LD2410_BAUD_RATE 256000

#include "MyLD2410.h"
```

```
#ifdef DEBUG_MODE

MyLD2410 sensor(sensorSerial, true);

#else

MyLD2410 sensor(sensorSerial);

#endif


unsigned long nextPrint = 0, printEvery = 1000; // print every second


void printValue(const byte &val) {

    Serial.print(' ');

    Serial.print(val);

}


void printData() {

    Serial.print(sensor.statusString());

    if (sensor.presenceDetected()) {

        Serial.print(", distance: ");

        Serial.print(sensor.detectedDistance());

        Serial.print("cm");

    }

    Serial.println();

    if (sensor.movingTargetDetected()) {

        Serial.print(" MOVING    = ");

        Serial.print(sensor.movingTargetSignal());

        Serial.print("@");

        Serial.print(sensor.movingTargetDistance());

        Serial.print("cm ");

    }

}
```

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

```
if (sensor.inEnhancedMode()) {  
    Serial.print("\n signals->");  
    sensor.getMovingSignals().forEach(printValue);  
    Serial.print(" ] thresholds:");  
    sensor.getMovingThresholds().forEach(printValue);  
    Serial.print(" ]");  
}  
Serial.println();  
}  
  
if (sensor.stationaryTargetDetected()) {  
    Serial.print(" STATIONARY= ");  
    Serial.print(sensor.stationaryTargetSignal());  
    Serial.print("@");  
    Serial.print(sensor.stationaryTargetDistance());  
    Serial.print("cm ");  
    if (sensor.inEnhancedMode()) {  
        Serial.print("\n signals->");  
        sensor.getStationarySignals().forEach(printValue);  
        Serial.print(" ] thresholds:");  
        sensor.getStationaryThresholds().forEach(printValue);  
        Serial.print(" ]");  
    }  
    Serial.println();  
}  
  
if (sensor.inEnhancedMode() && (sensor.getFirmwareMajor() > 1)) {  
    Serial.print("Light level: ");  
    Serial.println(sensor.getLightLevel());  
}
```

<https://dronebotworkshop.com>

```
    Serial.print("Output level: ");

    Serial.println((sensor.getOutLevel()) ? "HIGH" : "LOW");
}

Serial.println();
}

void setup() {
    Serial.begin(SERIAL_BAUD_RATE);

    #if defined(ARDUINO_XIAO_ESP32C3) || defined(ARDUINO_XIAO_ESP32C6) ||
    defined(ARDUINO_SAMD_NANO_33_IOT) || defined(ARDUINO_AVR_LEONARDO)

        sensorSerial.begin(LD2410_BAUD_RATE);
    #else

        sensorSerial.begin(LD2410_BAUD_RATE, SERIAL_8N1, RX_PIN, TX_PIN);
    #endif

    delay(2000);

    Serial.println(__FILE__);

    if (!sensor.begin()) {
        Serial.println("Failed to communicate with the sensor.");

        while (true) {}
    }

    #ifdef ENHANCED_MODE

        sensor.enhancedMode();
    #else

        sensor.enhancedMode(false);
    #endif

    delay(nextPrint);
}
```



```
}  
  
void loop() {  
  if ((sensor.check() == MyLD2410::Response::DATA) && (millis() > nextPrint)) {  
    nextPrint = millis() + printEvery;  
    printData();  
  }  
}
```

The sketch begins by including the library and defining the hardware serial port (Serial1) that the LD2410C is connected to. It then creates an instance of the MyLD2410 class.

In Setup, we start both the primary serial monitor (for displaying output) and the sensor's serial port. The key step is `mySensor.begin(SENSOR_SERIAL)`, which instructs the library to listen to the specified serial port for sensor data.

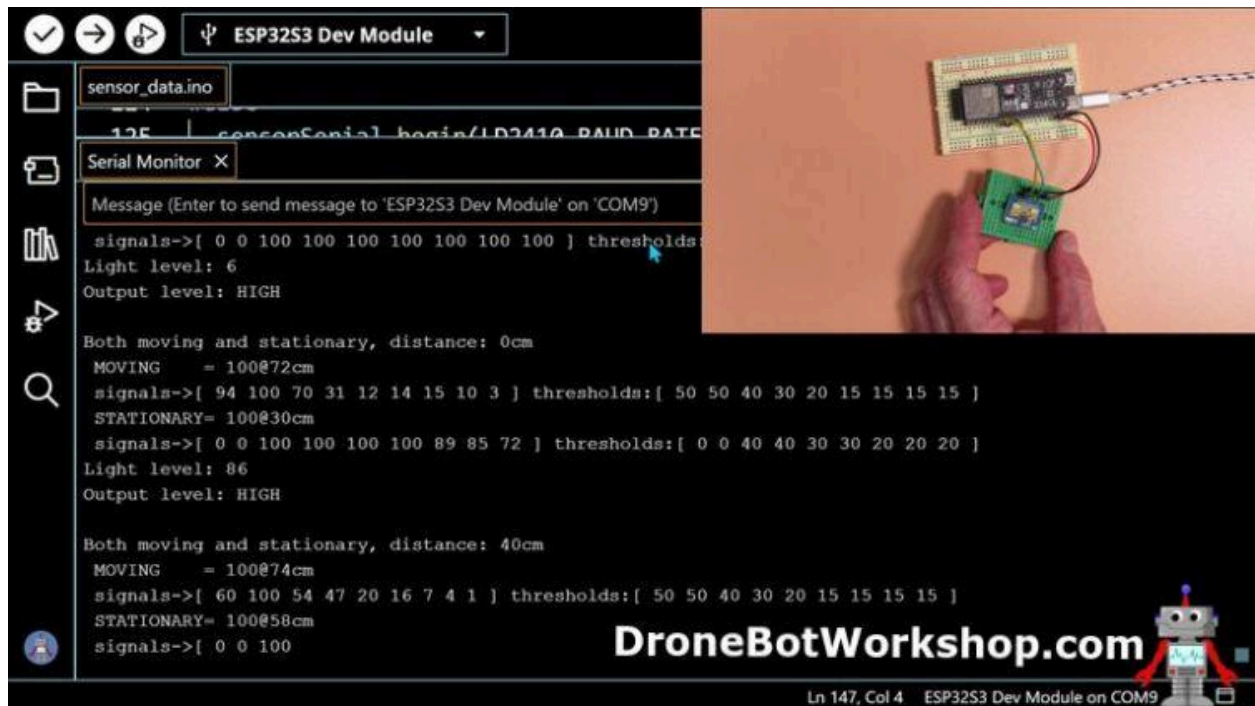
In the Loop, we call the `mySensor.loop()` command. It must be called on every iteration to allow the library to check for and parse incoming bytes from the sensor. The `mySensor.available()` function returns true only when a complete, valid data packet has been received.

Inside the `if (mySensor.available())` block, the sketch calls various functions to retrieve specific pieces of information. It checks for moving and stationary targets using `mySensor.isMoving()` and `mySensor.isStationary()`.

If a target is detected, it then retrieves its distance and energy (signal strength) using functions like `mySensor.movingTargetDistance()` and `mySensor.stationaryTargetEnergy()`. This data is then printed to the serial monitor, giving you a clear, real-time view of what the sensor is detecting.

For more projects and tutorials visit the DroneBot Workshop - <https://dronebotworkshop.com>

Run the sketch and observe the output as you walk towards and away from the sensor. You can also change the light level (or block it with your hand, as I did in the video) to see the light levels change.



You can use this sketch as the basis for your own LD2410C code.

<https://dronebotworkshop.com>

Conclusion

The LD2410C Human Presence Sensor is an impressive 24 GHz FMCW radar module designed for makers, students, and experimenters. Unlike traditional PIR sensors, it excels at detecting not just motion, but also the subtle presence of a stationary person. Understanding the principles of FMCW radar and how the LD2410C analyzes reflected signals for distance and energy measurement reveals the advanced technology within this compact sensor.

In this article (and its accompanying video), we've seen how versatile this little module is. We learned how to connect it to a computer to visualize its data using the LD2410-Tool GUI and LD2410 Configurator. We also used the HLKRadarTool app to communicate with our human presence sensor over Bluetooth.

Finally, we connected the LD2410C to an ESP32 and utilized the versatile MyLD2410 library to retrieve data from it.

I encourage you to get one of these sensors and experiment with it. Its combination of low cost, high performance, and ease of use makes it one of the most exciting new components for the modern electronics workshop, and an ideal part of a Smart Home installation. I'm excited to see what you create with it!