VL53L4CD Linux driver

Getting started with Linux Driver and Raspberry PI 3

Introduction

The VL53L4CD is a state of the art, Time-of-Flight (ToF), laser-ranging sensor enhancing the ST FlightSense product family. Housed in a miniature reflowable package, it integrates a SPAD array and physical infrared filters to achieve the best ranging performance in various ambient lighting conditions with a range of cover glass materials.

The purpose of this user manual is to describe how to start using the Linux Driver based on VL53L4CD Ultra Lite Driver (ULD).

Figure 1: VL53L4CD sensor module

References
VL53L4CD User Manual (UM2931)

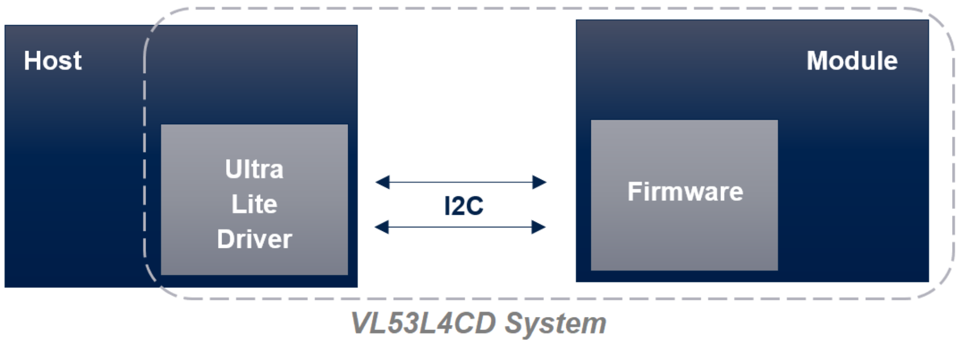
Contents

1 Functional description

1.1 System overview

The VL53L4CD system is composed of a hardware module and the Ultra Lite Driver software (VL53L4CD ULD) running on a host (see figure below). The hardware module contains the ToF sensor. ST delivers the software driver which is referred to in this document as "the driver". This document describes the functions of the driver which are accessible to the host. These functions control the sensor and get the ranging data.

Figure 2: VL53L4CD system overview



1.2 Schematics and I2C configuration

The communication between driver and firmware is handled by I2C, with a capability of operating up to 1 MHz (Fast Mode Plus). The implementation requires pull-ups on the SCL and SDA lines. Please refer to VL53L4CD datasheet for more information.

By default, the sensor is programmed to run on I2C Fast Mode (up to 400kHz). The Fast Mode Plus (up to 1MHz) can be enabled using a platform compilation key in the file platform.h:

```
#define VL53L4CD_I2C_FAST_MODE_PLUS
```

The VL53L4CD device has a default I2C address of 0x52. However, it is possible to change the default address to avoid conflict with other devices or facilitate adding multiple VL53L4CD modules to the system for a greater system Field of View. The I2C address can be reprogrammed using function *VL53L4CD_SetI2CAddress().* A complete description is given into the Ultra Lite Driver User Manual.

2 Package content and data flow

2.1 Driver architecture and content

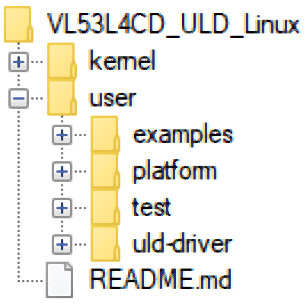
The VL53L4CD Linux driver is based on the VL53L4CD Ultra Lite Driver (ULD). All driver ULD features are described in the User Manual.

The Linux driver is a package composed of several folders:

- kernel / user folder: contain the two supported modes of the driver.
- examples: contain several examples to use the sensor.
- platform: contain mandatory macros and platform functions. It needs to be filled by the customer.
- test: contain the main input file.
- uld-driver: contain the VL53L4CD Ultra Lite Driver (ULD).

The described driver architecture is represented on Figure 3: Linux driver architecture

Figure 3: Linux driver architecture



Note: Platform.h file contains mandatory macros to use ULD. All the file content is mandatory to correctly use Ultra Lite Driver.

2.2 Supported system modes

The VL53L4CD Linux driver supports the User space mode and Kernel mode. The mode can be selected using compilation flag STMVL53L4CD_KERNEL in the test Makefile. By default, the driver is set in User space mode.

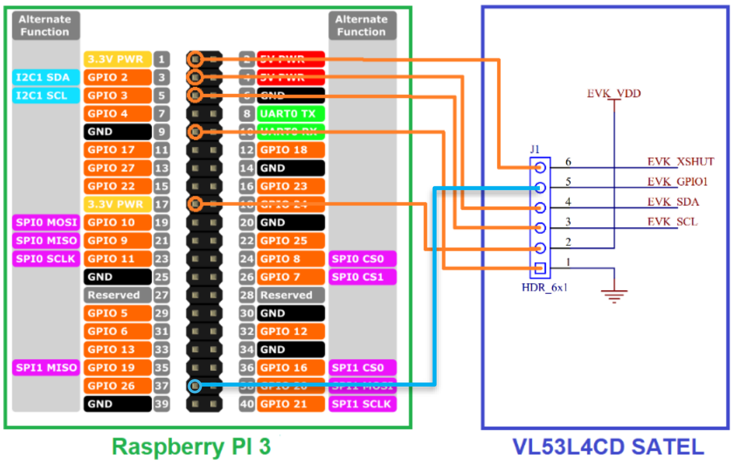
3 Running example on Raspberry PI 3

The proposed Linux implementation is customized to run on a Raspberry PI 3, but can be adapted to run on any Linux embedded application, as far as the VL53L4CD sensor is connected through I2C.

3.1 Hardware connection

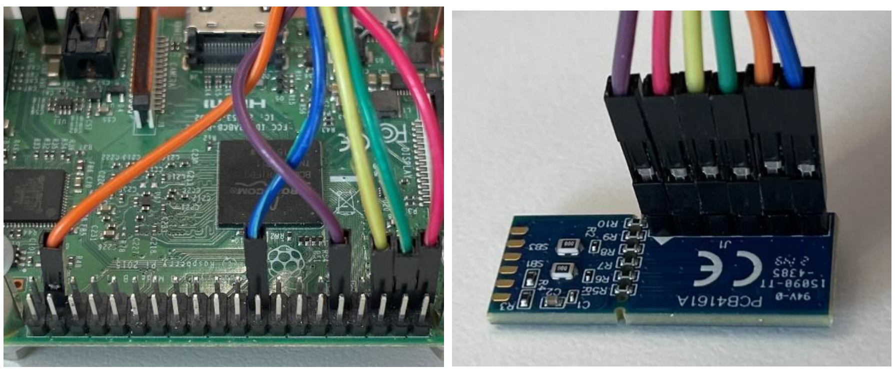
The simplest example to run an example with Linux driver is to use a VL53L4CD SATEL. Pins need to be connected following the schematics Figure 4: VL53L4CD SATEL to Raspberry PI3 schematics.

Figure 4: VL53L4CD SATEL to Raspberry PI3 schematics



Note: In kernel mode, examples require to catch interrupt raised on GPIO1 pin. User needs to connect GPIO1 to the PI3 interrupt input pin, GPIO26 was used by default.

Figure 5 : VL53L4CD SATEL to Raspberry PI3 pictures



3.2 User space compilation

User space mode only requires to compile the *makefile* located into the /test folder.
The following sequence is required:

- Go to test folder
  - cd VL53L4CD_ULD_Linux/user/test
- Build program
  - make

Then user can run the application as described in section 3.4 Run a test application.

3.3 Kernel compilation

Kernel compilation requires installation if kernel source headers and device tree blog to be run. The following sequence is required:

- Install the Raspberry PI Kernel source headers (refer to official documentation to download a version matching with kernel version)
- Update the boot configuration, adding or un-commenting the following lines of file /boot/config.txt
  - dtparam=i2c_arm=on
  - dtparam=i2c1=on
  - dtparam=i2c1_baudrate=100000
  - dtoverlay=stmvl53l4cd
- Compile the device tree blob
  - cd VL53L4CD_ULD_Linux/kernel
  - make dtb
  - sudo reboot
- Go to test folder and enable kernel cflag into the test Makefile
  - cd VL53L4CD_ULD_Linux/user/test
  - CFLAGS_RELEASE += STMVL53L4CD_KERNEL
- Build program
  - make
- Compile the kernel module
  - cd VL53L4CD_ULD_Linux/kernel
  - make clean
  - make
  - sudo make insert

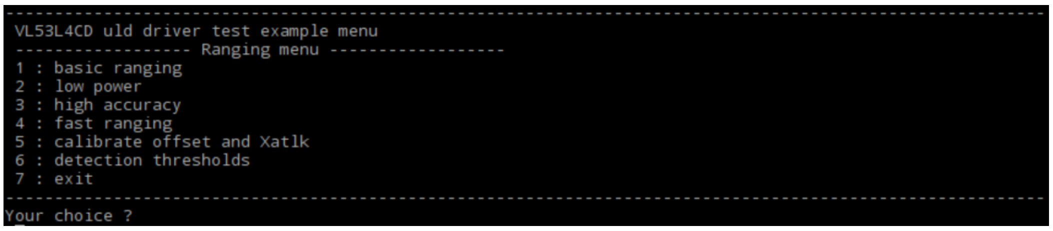
Then the user can run the application as described in section 3.4Run a test application.

3.4 Run a test application

Once the compilation is done, the user can go to the test folder and run the example menu. The menu allows using examples located in the /examples folder.

- cd VL53L4CD_ULD_Linux/user/test
- sudo ./menu

Figure 6: Screen capture of example menu



```
-----------------------------------------------------------------------------------------------
 VL53L4CD uld driver test example menu
 ----------------- Ranging menu -----------------
 1 : basic ranging
 2 : low power
 3 : high accuracy
 4 : fast ranging
 5 : calibrate offset and Xatlk
 6 : detection thresholds
 7 : exit
-----------------------------------------------------------------------------------------------
Your choice ?
```

Note: In kernel mode, Examples requires to catch interrupt raised on GPIO1 pin. User need to connect it to the host interrupt pin and to implement GPIO callback for using such example.

Document revision history

Date	Version	Changes
26-July-2021	1.0	Initial release
20-June-2024	2.0	Update the pictures with interrupt pin