VL53L0X GUI Installation

Quick Start Guide

Apr 2017

life.augmented

VL53L0X eco-system

Documentation

VL53L0X GUI
User Manual

VL53L0X X-CUBE
User Manual

X-NUCLEO-53L0A1 HW
User Manual

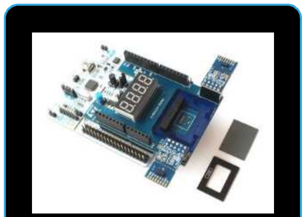
VL53L0X Eval GUI - STSW-IMG006

Get ranging live curves on your PC
Change key settings of the device
Data logging capabilities

STM32 Cube
STM32 Nucleo
STM32 Open Development Environment

X-CUBE-53L0A1 package

Full integration in STM32 MCU (real-time)
All source code provided
Full access to product settings
Data logging capabilities
Ranging and Gesture detection demo



Hardware

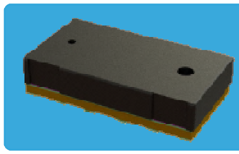
P-NUCLEO-53L0A1
X-NUCLEO-53L0A1

VL53L0X API Data Brief
VL53L0X API User Manual

VL53L0X C API package – STSW-IMG005

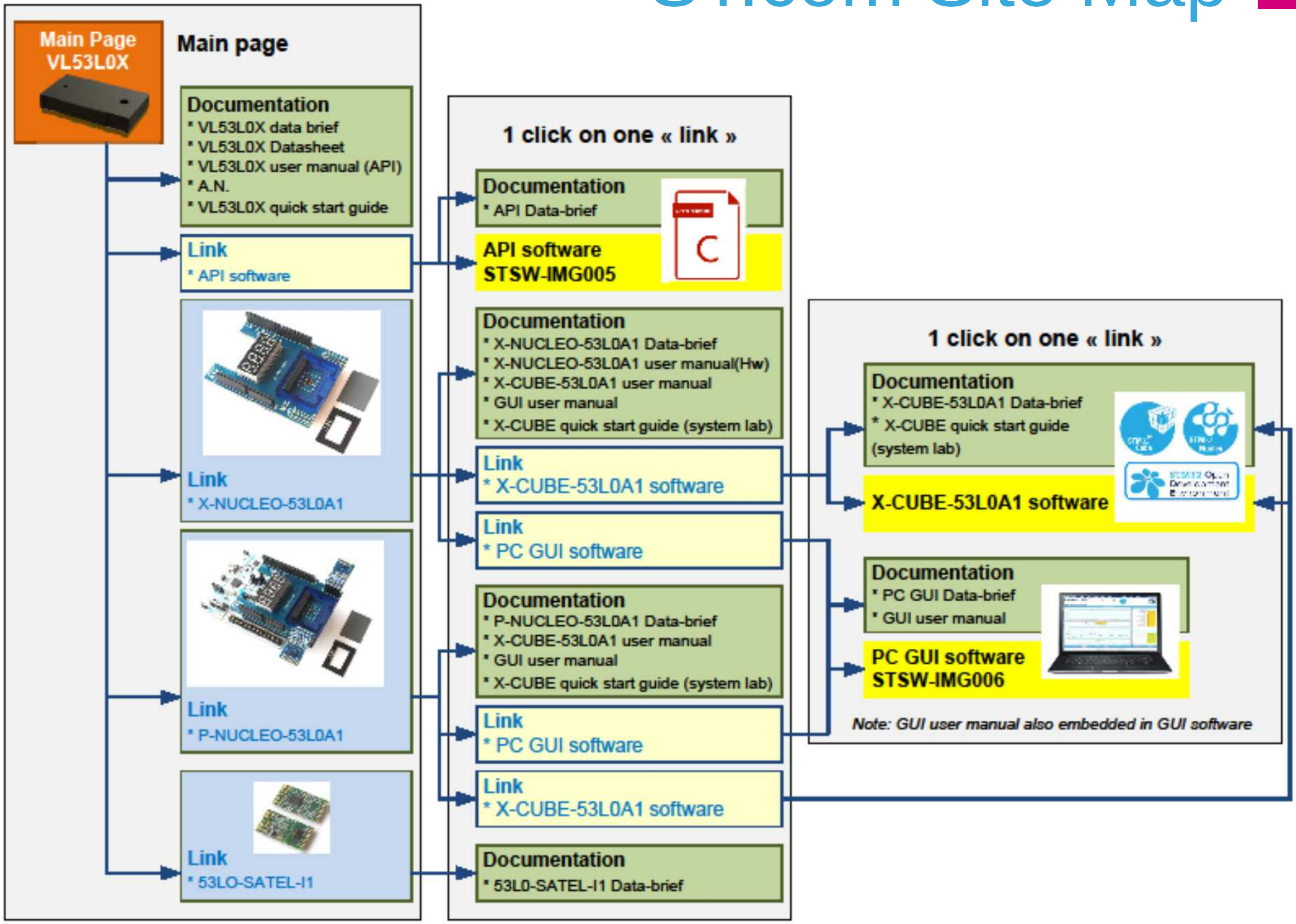
Discover API Source code

VL53L0X Data Brief
VL53L0X Datasheet
VL53L0X Quick Start Guide
Applications Notes



VL53L0X : Miniature ToF Ranging & Gesture Sensor

ST.com Site Map



Main Page VL53L0X

Main page

Documentation
- VL53L0X data brief
- VL53L0X Datasheet
- VL53L0X user manual (API)
- A.N.
- VL53L0X quick start guide

Link
- API software

Link
- X-NUCLEO-53L0A1

Link
- P-NUCLEO-53L0A1

Link
- 53LO-SATEL-I1

1 click on one « link »

Documentation
- API Data-brief

API software STSW-IMG005

Documentation
- X-NUCLEO-53L0A1 Data-brief
- X-NUCLEO-53L0A1 user manual(Hw)
- X-CUBE-53L0A1 user manual
- GUI user manual
- X-CUBE quick start guide (system lab)

Link
- X-CUBE-53L0A1 software

Link
- PC GUI software

Documentation
- P-NUCLEO-53L0A1 Data-brief
- X-CUBE-53L0A1 user manual
- GUI user manual
- X-CUBE quick start guide (system lab)

Link
- PC GUI software

Link
- X-CUBE-53L0A1 software

Documentation
- 53L0-SATEL-I1 Data-brief

1 click on one « link »

Documentation
- X-CUBE-53L0A1 Data-brief
- X-CUBE quick start guide (system lab)

X-CUBE-53L0A1 software

Documentation
- PC GUI Data-brief
- GUI user manual

PC GUI software STSW-IMG006

Note: GUI user manual also embedded in GUI software

Hardware Description

- VL53L0X Evaluation tools are all based on the same hardware pack composed of
  - Nucleo F401RE board
  - X-NUCLEO-53L0A1 Nucleo Expansion board
  - Optional two VL53L0X satellites
  - Several gap spacers and cover glass

- Search for ***P-NUCLEO-53L0A1*** on st.com to order the pack and get documentation

X-NUCLEO-53L0A1
Nucleo Expansion board
VL53L0X
TIME-OF-FLIGHT (ToF) RANGING SENSOR, BASED ON FLIGHTSENSE™
• Evaluation board Compatible with STM32 Nucleo boards & Arduino™
• Delivered with:
- Airgap spacers (1, 0.5, 0.25mm)
- Cover glass
- 2x VL53L0X satellites
- 2x 10 pin connectors
ST part number: X-NUCLEO-53L0A1
www.st.com/VL53L0X

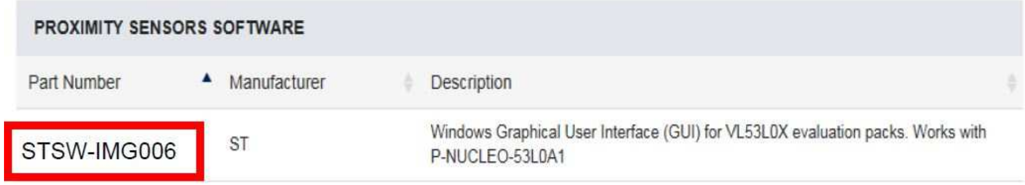
VL53L0X GUI : Purpose

- PC Graphical User Interface to
  - Display (in live) key ranging data (distance, signal rate)
  - Change key parameters of VL53L0X
  - Perform calibration phases (offset and xTalk with cover glass)
  - Get data logging (.csv file)
- GUI is running on the PC connected to a P-NUCLEO-53L0A1 pack
  - VL53L0X API running on the PC side
  - Run simple .exe programs on the PC to do ranging from VL53L0X
- Download from st.com searching for “STSW-IMG006”
  - Run the installer with Admin privileges

VL53L0X GUI : Installation(1/2)

- Software Link
  - Search for STSW-IMG006 on www.st.com

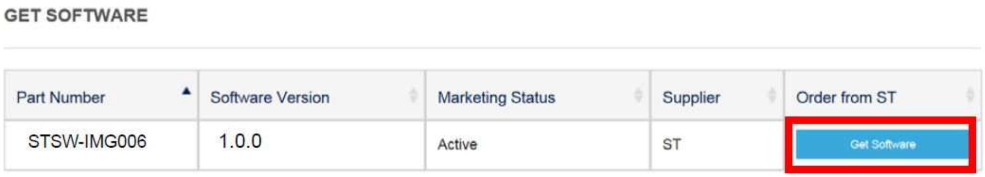
- Download step
  - Click on “STSW-IMG006”.



PROXIMITY SENSORS SOFTWARE

Part Number ▲	Manufacturer	Description
STSW-IMG006	ST	Windows Graphical User Interface (GUI) for VL53L0X evaluation packs. Works with P-NUCLEO-53L0A1

  - Click on “Get Software”



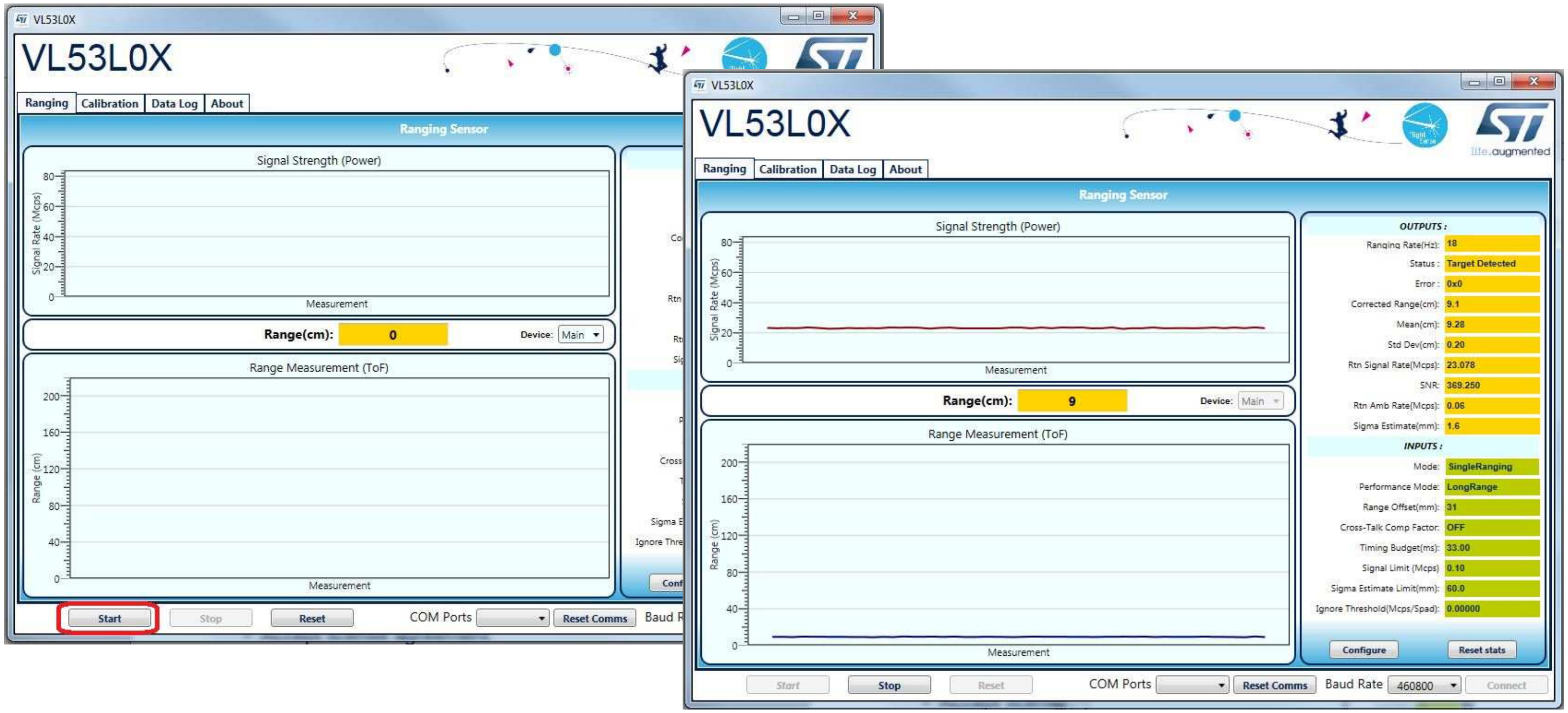
GET SOFTWARE

Part Number ▲	Software Version	Marketing Status	Supplier	Order from ST
STSW-IMG006	1.0.0	Active	ST	Get Software

  - Accept license agreement.
  - Then “Save” and “Run” VL53L0X_setup.exe
  - Icon “VL53L0X” is installed on the user desktop space.

VL53L0X GUI : Installation(2/2)

- Download step
  - Connect STM32 Nucleo pack to an USB PC port.
  - Start PC graphic user interface by clicking “VL53L0X” icon.
  - Click on the “Start” button to get the ranging data.



Range(cm): 0
Device: Main

Range(cm): 9
Device: Main

OUTPUTS :

Ranging Rate(Hz):	18
Status :	Target Detected
Error :	0x0
Corrected Range(cm):	9.1
Mean(cm):	9.28
Std Dev(cm):	0.20
Rtn Signal Rate(Mcps):	23.078
SNR:	369.250
Rtn Amb Rate(Mcps):	0.06
Sigma Estimate(mm):	1.6

INPUTS :

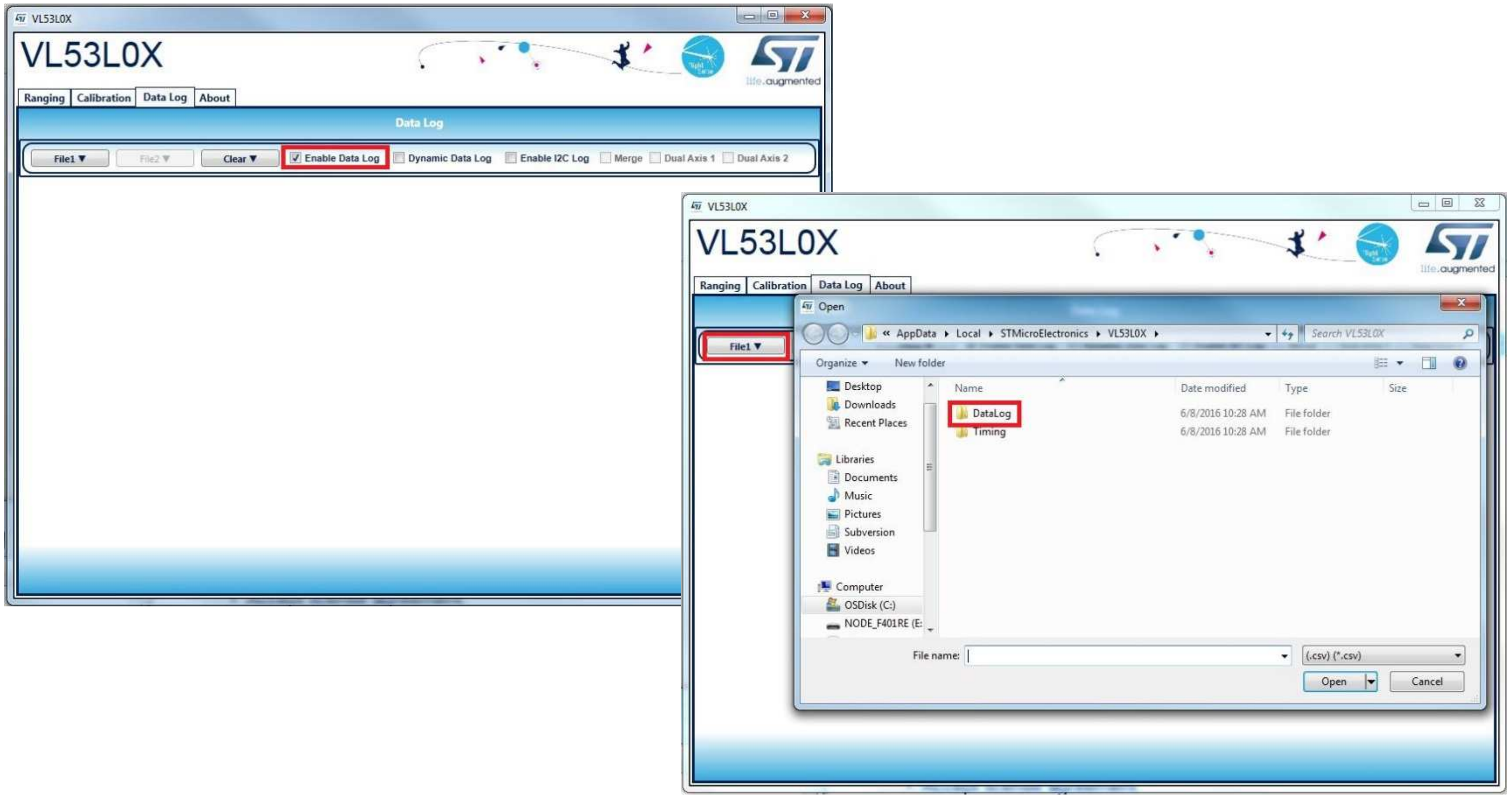
Mode:	SingleRanging
Performance Mode:	LongRange
Range Offset(mm):	31
Cross-Talk Comp Factor:	OFF
Timing Budget(ms):	33.00
Signal Limit (Mcps)	0.10
Sigma Estimate Limit(mm):	60.0
Ignore Threshold(Mcps/Spad):	0.00000

Baud Rate 460800

GUI Logging Data

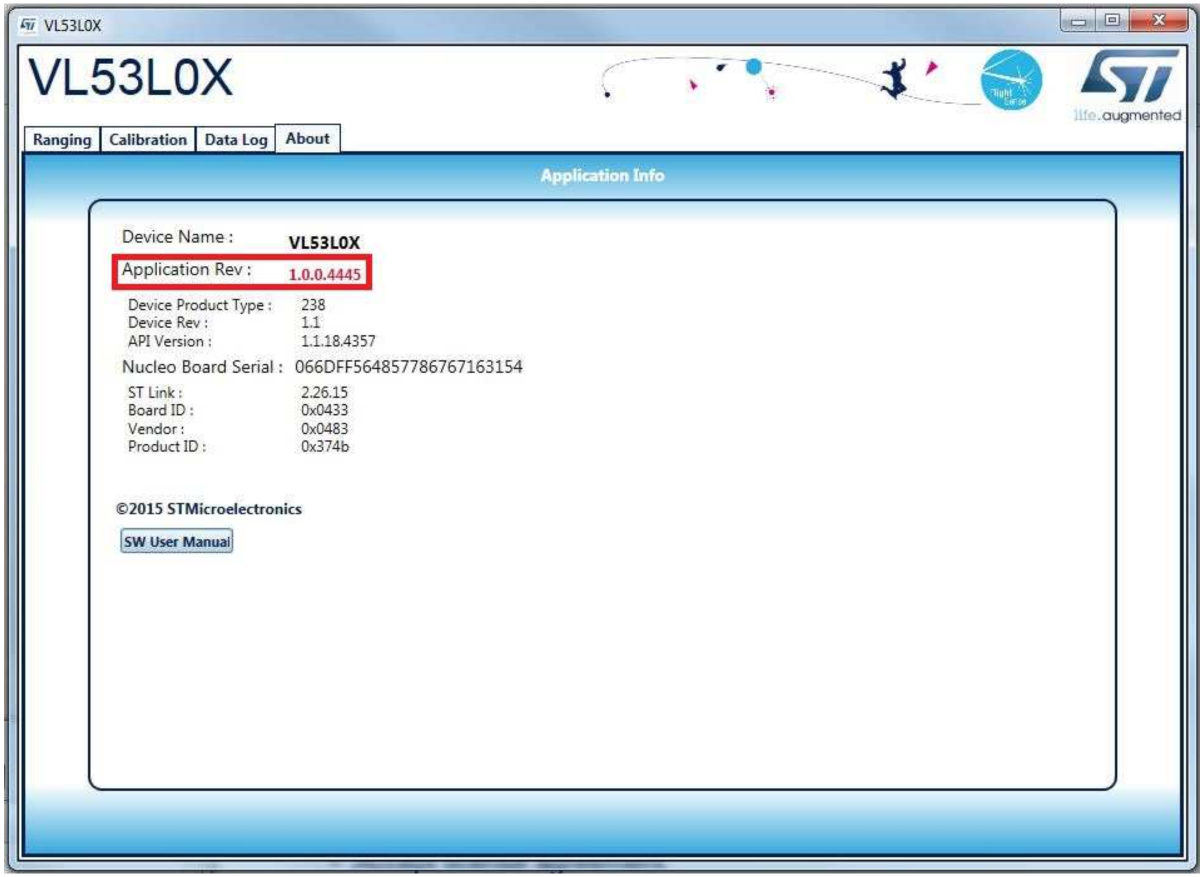
- Click the “Data Log” tab, and click the “Enable Data log”.

- To click the “File1” to get the path for saving log.



GUI Version

- Click the “About” tab, and check the “Application Rev”.



Device Name : VL53L0X

Application Rev : 1.0.0.4445

Device Product Type : 238
Device Rev : 1.1
API Version : 1.1.18.4357

Nucleo Board Serial : 066DFF564857786767163154

ST Link : 2.26.15
Board ID : 0x0433
Vendor : 0x0483
Product ID : 0x374b

©2015 STMicroelectronics

API Running with Windows

Quick Start Guide

Apr 2017

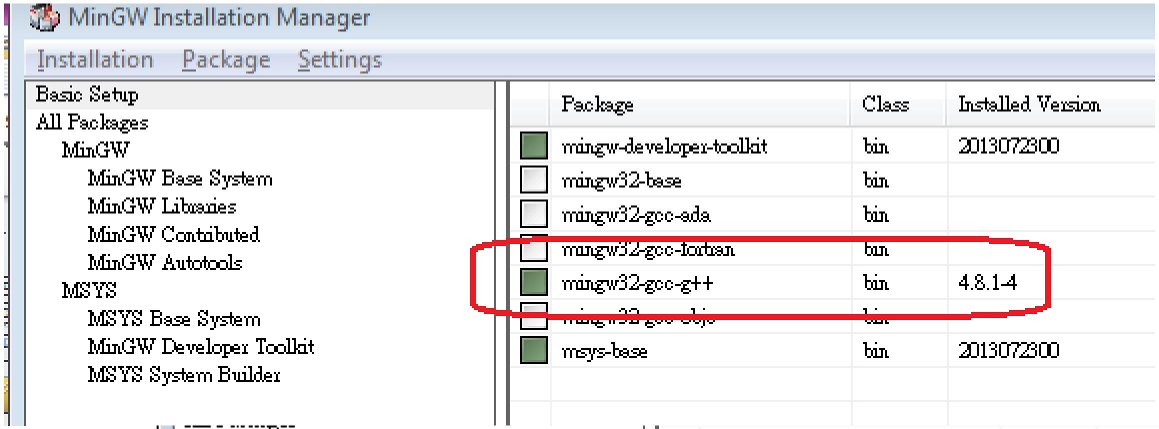
life.augmented

install GNU gcc/g++ in Windows

1. Download MinGW from http://www.mingw.org/ [Documentation]=>[Getting Start]=>[mingw-get-setup.exe]

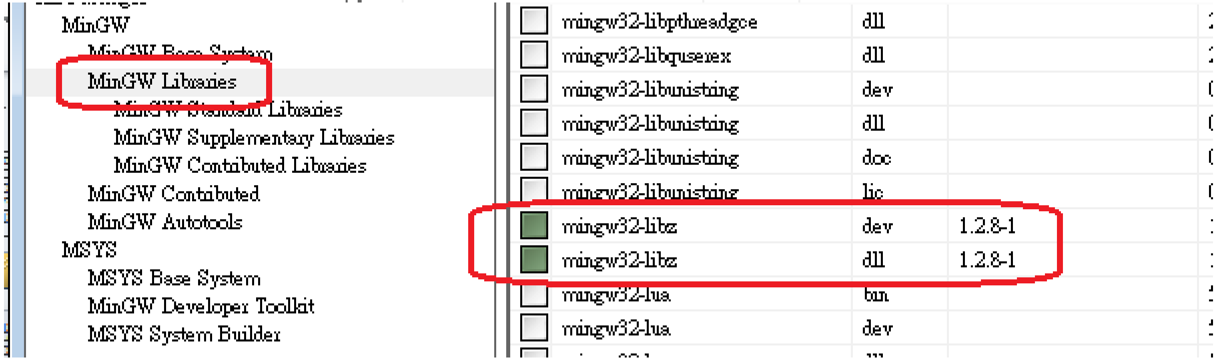
2 Lunch mingw-get-setup.exe

3. Make sure everything is installed.



Basic Setup
All Packages
- MinGW
  - MinGW Base System
  - MinGW Libraries
  - MinGW Contributed
  - MinGW Autotools
- MSYS
  - MSYS Base System
  - MinGW Developer Toolkit
  - MSYS System Builder

	Package	Class	Installed Version
☑	mingw-developer-toolkit	bin	2013072300
☐	mingw32-base	bin	
☐	mingw32-gcc-ada	bin	
☐	mingw32-gcc-fortran	bin	
☑	mingw32-gcc-g++	bin	4.8.1-4
☐	mingw32-gcc-objc	bin	
☑	msys-base	bin	2013072300

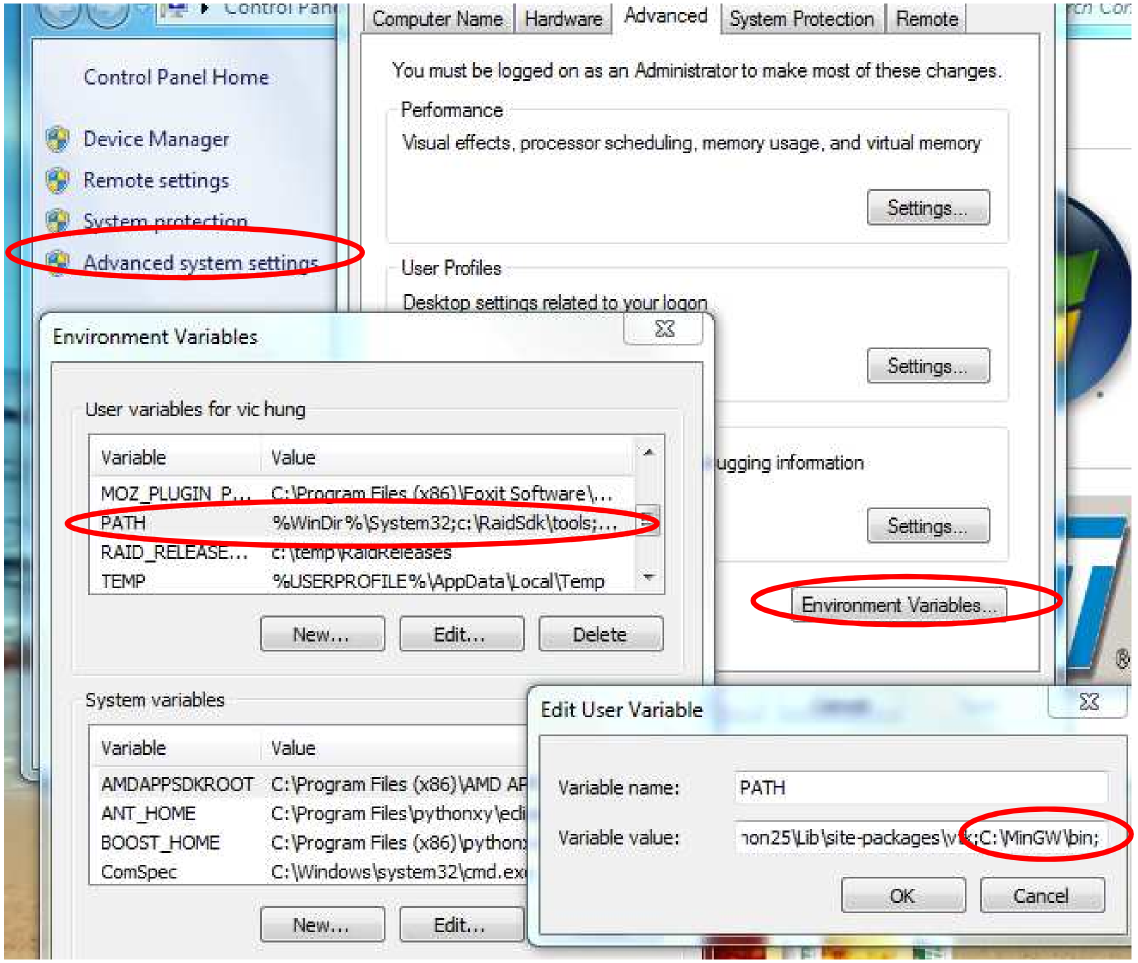


- MinGW
  - MinGW Base System
  - MinGW Libraries
    - MinGW Standard Libraries
    - MinGW Supplementary Libraries
    - MinGW Contributed Libraries
  - MinGW Contributed
  - MinGW Autotools
- MSYS
  - MSYS Base System
  - MinGW Developer Toolkit
  - MSYS System Builder

☐	mingw32-libpthreadgce	dll	
☐	mingw32-libquserex	dll	
☐	mingw32-libunistring	dev	
☐	mingw32-libunistring	dll	
☐	mingw32-libunistring	doc	
☐	mingw32-libunistring	lic	
☑	mingw32-libz	dev	1.2.8-1
☑	mingw32-libz	dll	1.2.8-1
☐	mingw32-lua	bin	
☐	mingw32-lua	dev	

install GNU gcc/g++ in Windows

4. Add c:\MinGW\bin to system path



You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage, and virtual memory

User Profiles

Desktop settings related to your logon

ugging information

Environment Variables

User variables for vic hung

Variable	Value
MOZ_PLUGIN_P...	C:\Program Files (x86)\Foxit Software\...
PATH	%WinDir%\System32;c:\RaidSdk\tools;...
RAID_RELEASE...	c:\temp\RaidReleases
TEMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
AMDAPPSDKROOT	C:\Program Files (x86)\AMD AP
ANT_HOME	C:\Program Files\pythonxy\ecli
BOOST_HOME	C:\Program Files (x86)\python
ComSpec	C:\Windows\system32\cmd.ex

Edit User Variable

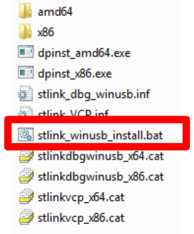
Variable name: PATH

Variable value: non25\Lib\site-packages\vtk;C:\MinGW\bin;

SW pre-requisites (to be done once)

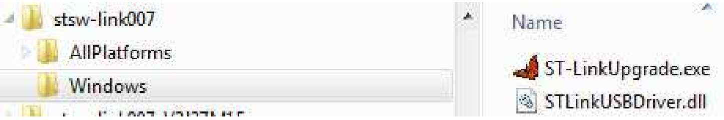
- Connect the Nucleo pack to the PC through USB
  - Wait for the board to be recognized as a mass storage device (some drivers will be installed automatically)

- Install ST-Link Virtual Com port drivers on the PC (**STSW-LINK009**)
  - Search for STSW-LINK009 on st.com, download, unzip
  - Launch stlink_winusb_install.bat



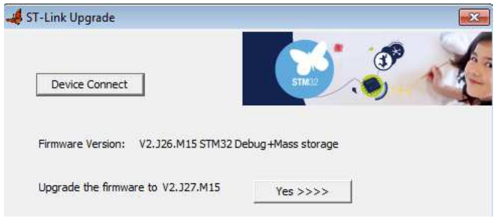
amd64
x86
dpinst_amd64.exe
dpinst_x86.exe
stlink_dbg_winusb.inf
stlink_VCP.inf
stlink_winusb_install.bat
stlinkdbgwinusb_x64.cat
stlinkdbgwinusb_x86.cat
stlinkvcp_x64.cat
stlinkvcp_x86.cat

- Upgrade ST-Link FW on the Nucleo board to get the latest version and benefit from best performances for UART over USB transfers (**STSW-LINK007**)
  - Search for STSW-LINK007 on st.com, download, unzip
  - Connect Nucleo board to the PC through USB
  - Launch ST-LinkUpgrade.exe, press Device Connect, then Yes



stsw-link007
AllPlatforms
Windows

Name
ST-LinkUpgrade.exe
STLinkUSBDriver.dll

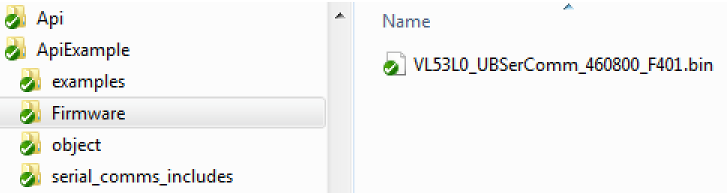


Firmware Version: V2.J26.M15 STM32 Debug+Mass storage

Upgrade the firmware to V2.J27.M15

Run SingleRanging example

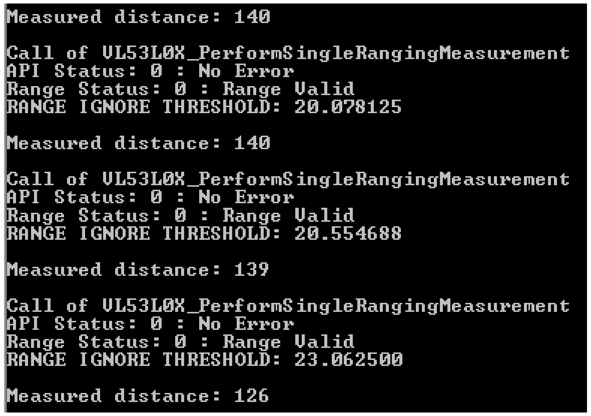
- Program Nucleo with FW doing Serial to I2C bridging
  - Connect Nucleo board to PC (through USB)
  - Drag & Drop .bin file from ***ApiExample\Firmware*** to Nucleo disk (Windows Explorer)
  - The board is ready to run examples



- Package contains several pre-compiled examples in ***ApiExample\examples*** directory
  - Connect Nucleo pack to the PC
  - Double-click on ***vl53l0x_SingleRanging_Example.exe***
  - This will automatically connect to Nucleo and open a windows showing ranging distances

- Source code of this simple example is provided in ***ApiExample\examples\src*** directory

- It is possible to (re)build the .exe programs by double-clicking on the .bat files
  - This requires gcc to be installed on the PC and available in the PATH



```
Measured distance: 140

Call of VL53L0X_PerformSingleRangingMeasurement
API Status: 0 : No Error
Range Status: 0 : Range Valid
RANGE IGNORE THRESHOLD: 20.078125

Measured distance: 140

Call of VL53L0X_PerformSingleRangingMeasurement
API Status: 0 : No Error
Range Status: 0 : Range Valid
RANGE IGNORE THRESHOLD: 20.554688

Measured distance: 139

Call of VL53L0X_PerformSingleRangingMeasurement
API Status: 0 : No Error
Range Status: 0 : Range Valid
RANGE IGNORE THRESHOLD: 23.062500

Measured distance: 126
```

compile and run sample(s)

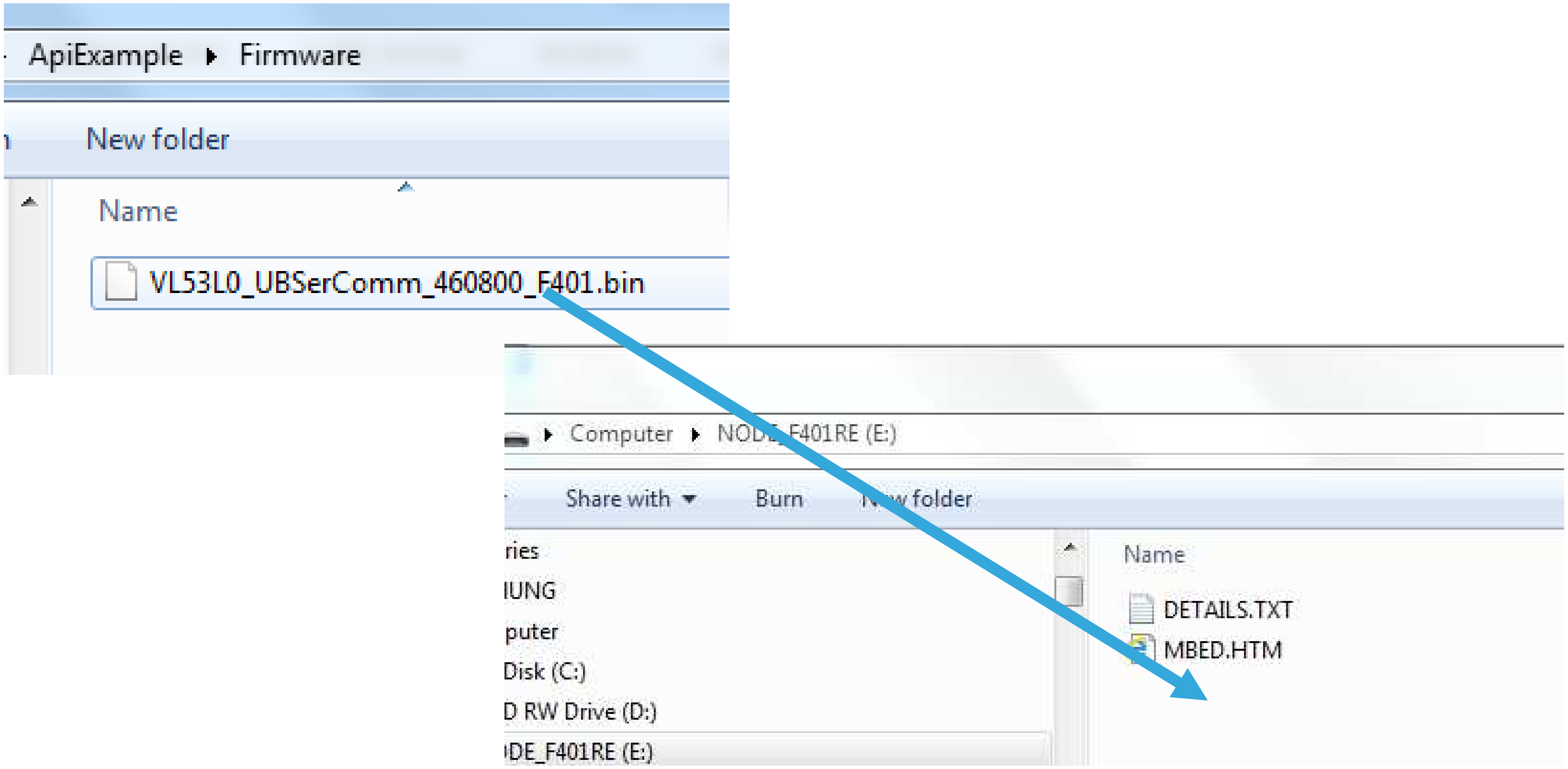
- Some sample codes can be found Inside ..\\VL53L0X_x.x.xx\ApiExample\examples\src

vl53l0x_SingleRanging_Example.c
vl53l0x_ContinuousRanging_Example.c
vl53l0x_SingleRanging_High_Accuracy_Example.c
vl53l0x_SingleRanging_High_Speed_Example.c
vl53l0x_SingleRanging_Long_Range_Example.c

It is convenient to use the samples to do the evaluation and debugging for VL53L0X under Windows OS.

compile and run sample(s)

- Copy VL53L0_UBSerComm_460800_F401.bin which in ..\\VL53L0X_x.x.xx\ApiExample\Firmware into ST Nucleo board.



compile and run sample(s)

- Run the relative .bat file to compile the modified C code in ..\\VL53L0X_x.x.xx\ApiExample\examples and to get the relative .exe file

BUILD_vl53l0x_SingleRanging_Example.bat
BUILD_vl53l0x_ContinuousRanging_Example.bat
BUILD_vl53l0x_SingleRanging_High_Accuracy_Example.bat
BUILD_vl53l0x_SingleRanging_High_Speed_Example.bat
BUILD_vl53l0x_SingleRanging_Long_Range_Example.bat

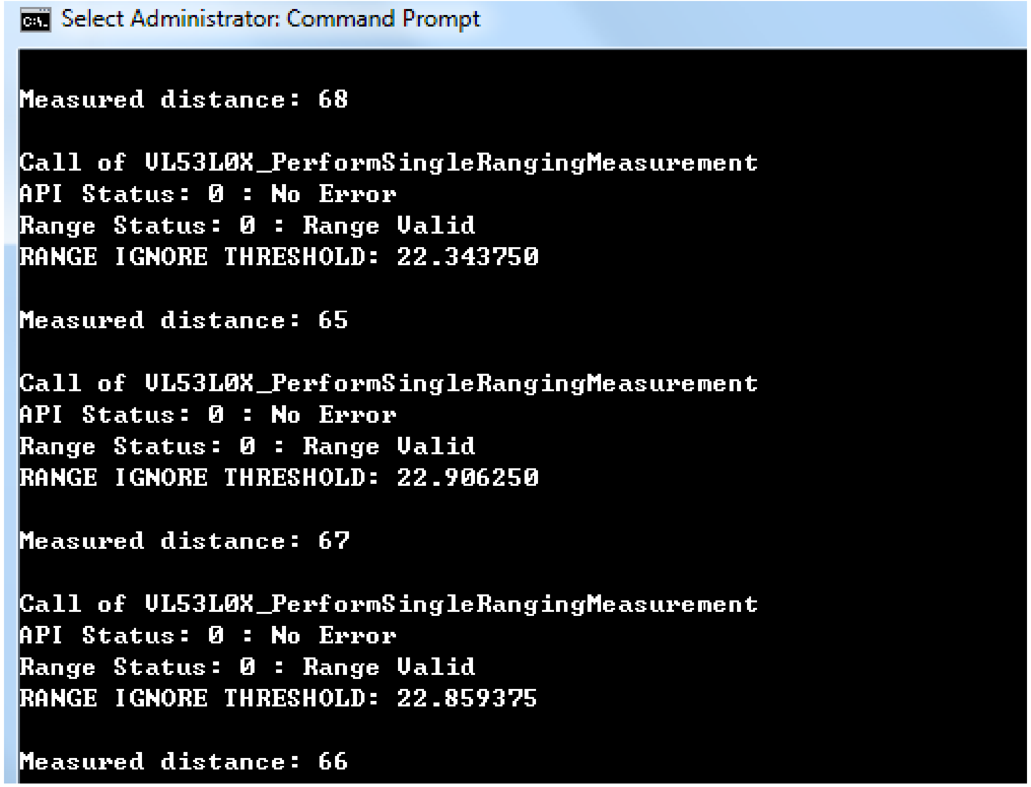
⬇

vl53l0x_SingleRanging_Example.exe
vl53l0x_ContinuousRanging_Example.exe
vl53l0x_SingleRanging_High_Accuracy_Example.exe
vl53l0x_SingleRanging_High_Speed_Example.exe
vl53l0x_SingleRanging_Long_Range_Example.exe

compile and run sample(s)

- Open a Command Prompt window to run the created program to get the values from VL53L0X.

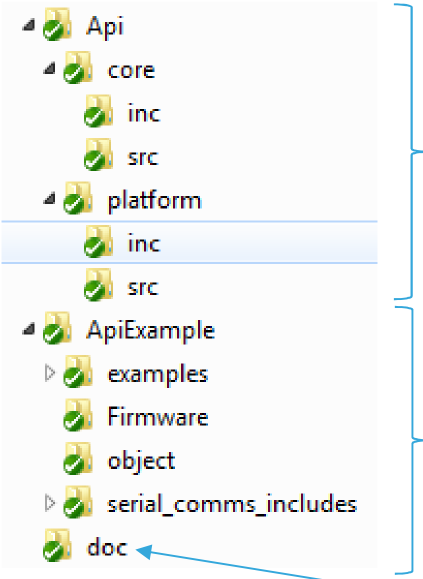
For example run vl53l0x_SingleRanging_Example.exe to get the ranging values.



```
Measured distance: 68

Call of VL53L0X_PerformSingleRangingMeasurement
API Status: 0 : No Error
Range Status: 0 : Range Valid
RANGE IGNORE THRESHOLD: 22.343750

Measured distance: 65

Call of VL53L0X_PerformSingleRangingMeasurement
API Status: 0 : No Error
Range Status: 0 : Range Valid
RANGE IGNORE THRESHOLD: 22.906250

Measured distance: 67

Call of VL53L0X_PerformSingleRangingMeasurement
API Status: 0 : No Error
Range Status: 0 : Range Valid
RANGE IGNORE THRESHOLD: 22.859375

Measured distance: 66
```

VL53L0X API : Content

- Unzip the package on your PC



```
Api
  core
    inc
    src
  platform
    inc
    src
ApiExample
  examples
  Firmware
  object
  serial_comms_includes
doc
```

VL53L0X API source code (core & platform)

Examples running on the PC (with Nucleo pack connected)

API Doxygen documentation

X-CUBE-53L0A1

Apr 2017

life.augmented

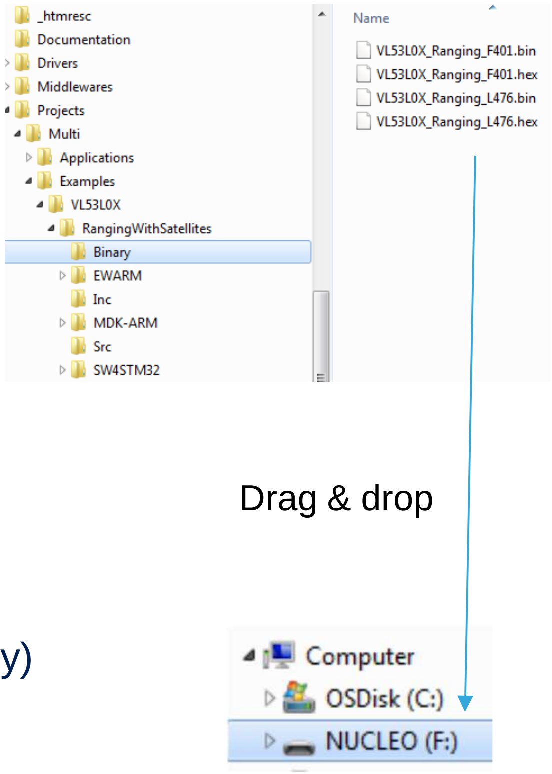
X-CUBE-53L0A1 : Purpose

- Give a full example of how VL53L0X device is integrated into a MCU sub-system taking benefit from the STM32 Open Development Environment

- Starting from this software package, user can
  - Run Ranging and Gesture detection demos with a simple drag & drop
  - Get basic data logging on PC through Virtual Com Port (Teraterm, Putty, etc…) to collect data or build simple PC GUIs
  - Import a project in his favorite IDE (Keil, IAR or STM32 Workbench) to browse the code, (re) compile, (re)flash Nucleo and debug (breakpoints, step into the code, etc…)
  - Understand how VL53L0X API has been ported on Nucleo
  - Get a working and real-time example of interrupt-based ranging mode
  - Modify the project code to change VL53L0X settings for the targeted application

X-CUBE-53L0A1 : Run a demo

- Hardware
  - Nucleo F401 (or L476) + VL53L0X Expansion board + VL53L0X satellites (optional)
- PC connection
  - Plug hardware to PC through USB
  - Wait for drivers to be installed and Nucleo to be seen in Windows explorer
- Flash and run the demo
  - Drag & drop the correct binary (.bin) from Examples or Applications directories onto the Nucleo mass storage
- Refer to X-CUBE-53L0X User Manual (in Documentations directory) to get more details on the demo and the way to change modes…



_htmresc
Documentation
Drivers
Middlewares
Projects
Multi
Applications
Examples
VL53L0X
RangingWithSatellites
Binary
EWARM
Inc
MDK-ARM
Src
SW4STM32

Name
VL53L0X_Ranging_F401.bin
VL53L0X_Ranging_F401.hex
VL53L0X_Ranging_L476.bin
VL53L0X_Ranging_L476.hex

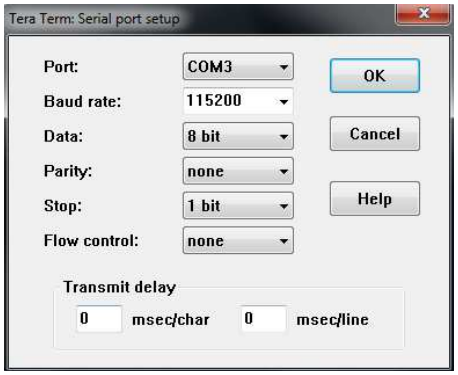
Drag & drop

Computer
OSDisk (C:)
NUCLEO (F:)

X-CUBE-53L0A1 : Getting further

- Please read X-CUBE-53L0X User Manual (in Documentations directory) to know how to
  - Get Data logging on the PC from Nucleo through serial port (over USB)
  - Install STM32 Workbench (SW4STM32) – Eclipse
  - Import a project in STM32 Workbench
  - Compile, Flash, Debug
  - Browse the code and get key functions implemented in each demo

Tera Term



Tera Term: Serial port setup

Port: COM3

Baud rate: 115200

Data: 8 bit

Parity: none

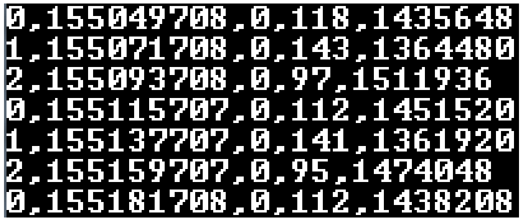
Stop: 1 bit

Flow control: none

Transmit delay

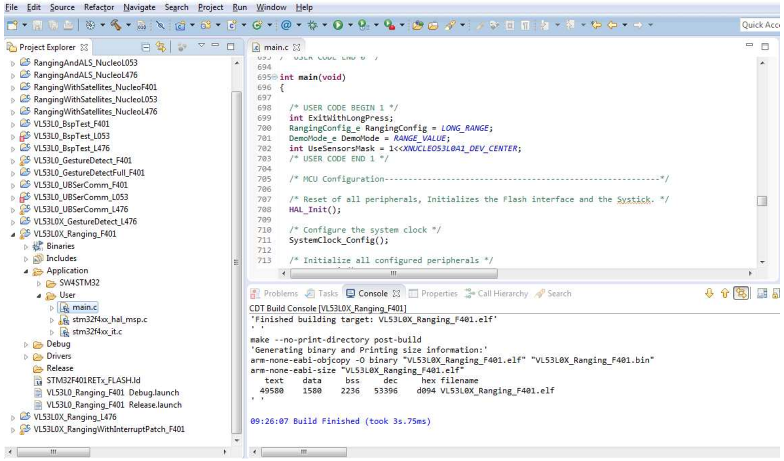
0 msec/char 0 msec/line

Data logging



```
0,155049708,0,118,1435648
1,155071708,0,143,1364480
2,155093708,0,97,1511936
0,155115707,0,112,1451520
1,155137707,0,141,1361920
2,155159707,0,95,1474048
0,155181708,0,112,1438208
```

Project in STM32 Workbench



File Edit Source Refactor Navigate Search Project Run Window Help
Quick Acc
Project Explorer
main.c
Problems Tasks Console Properties Call Hierarchy Search

```
RangingAndALS_NucleoL053
RangingAndALS_NucleoL476
RangingWithSatellites_NucleoF401
RangingWithSatellites_NucleoL053
RangingWithSatellites_NucleoL476
VL53L0_BspTest_F401
VL53L0_BspTest_L053
VL53L0_BspTest_L476
VL53L0_GestureDetect_F401
VL53L0_GestureDetectFull_F401
VL53L0_UBSerComm_F401
VL53L0_UBSerComm_L053
VL53L0_UBSerComm_L476
VL53L0X_GestureDetect_L476
VL53L0X_Ranging_F401
    Binaries
    Includes
    Application
        SW4STM32
        User
            main.c
            stm32f4xx_hal_msp.c
            stm32f4xx_it.c
    Debug
    Drivers
    Release
    STM32F401RETx_FLASH.ld
    VL53L0_Ranging_F401 Debug.launch
    VL53L0_Ranging_F401 Release.launch
VL53L0X_Ranging_L476
VL53L0X_RangingWithInterruptPatch_F401
```

```c
int main(void)
{

  /* USER CODE BEGIN 1 */
  int ExitWithLongPress;
  RangingConfig_e RangingConfig = LONG_RANGE;
  DemoMode_e DemoMode = RANGE_VALUE;
  int UseSensorsMask = 1<<XNUCLEO53L0A1_DEV_CENTER;
  /* USER CODE END 1 */

  /* MCU Configuration----------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* Configure the system clock */
  SystemClock_Config();

  /* Initialize all configured peripherals */
```

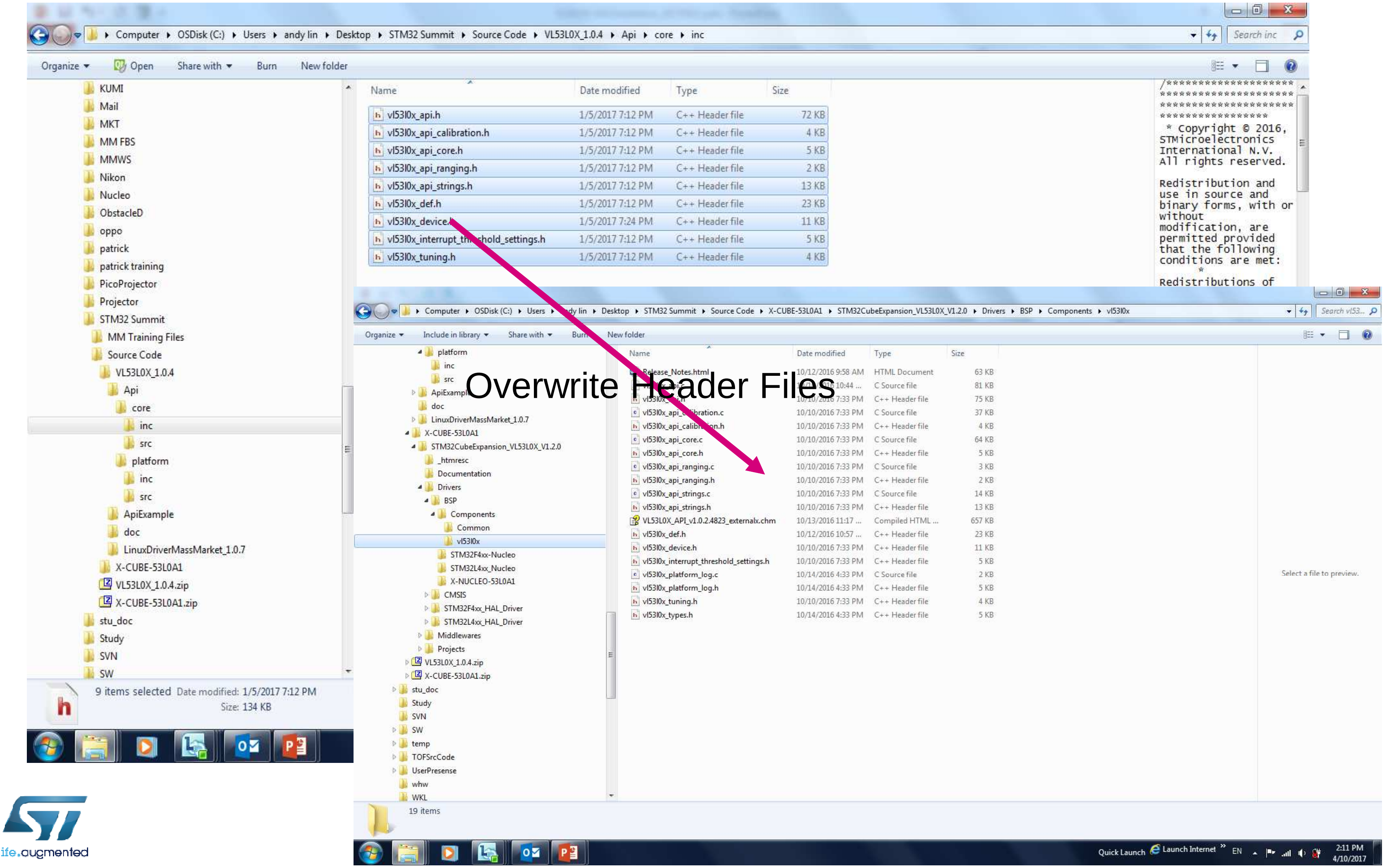
CDT Build Console [VL53L0X_Ranging_F401]

```
'Finished building target: VL53L0X_Ranging_F401.elf'
' '
make --no-print-directory post-build
'Generating binary and Printing size information:'
arm-none-eabi-objcopy -O binary "VL53L0X_Ranging_F401.elf" "VL53L0X_Ranging_F401.bin"
arm-none-eabi-size "VL53L0X_Ranging_F401.elf"
   text    data     bss     dec     hex filename
  49580    1580    2236   53396    d094 VL53L0X_Ranging_F401.elf
' '

09:26:07 Build Finished (took 3s.75ms)
```

Update Latest API to X-CUBE-53L0A1 (1)

- Propose: Make your SW updated with latest performance improvement and bug fixes.
- How to update API?
  - Replace header file and source file in X-CUBE-53L0A1 package by those in STSW-005 package
    - X-CUBE-53L0A1\STM32CubeExpansion_VL53L0X_V1.2.0\Drivers\BSP\Components\vl53l0x
    - VL53L0X_1.0.4\Api\core\inc
    - VL53L0X_1.0.4\Api\core\src
  - Recompile this project
  - Drag bin file to F401.

Update Latest API to X-CUBE-53L0A1 (2)



Computer ▸ OSDisk (C:) ▸ Users ▸ andy lin ▸ Desktop ▸ STM32 Summit ▸ Source Code ▸ VL53L0X_1.0.4 ▸ Api ▸ core ▸ inc
Search inc
Organize ▾
Open
Share with ▾
Burn
New folder

Name	Date modified	Type	Size
vl53l0x_api.h	1/5/2017 7:12 PM	C++ Header file	72 KB
vl53l0x_api_calibration.h	1/5/2017 7:12 PM	C++ Header file	4 KB
vl53l0x_api_core.h	1/5/2017 7:12 PM	C++ Header file	5 KB
vl53l0x_api_ranging.h	1/5/2017 7:12 PM	C++ Header file	2 KB
vl53l0x_api_strings.h	1/5/2017 7:12 PM	C++ Header file	13 KB
vl53l0x_def.h	1/5/2017 7:12 PM	C++ Header file	23 KB
vl53l0x_device.h	1/5/2017 7:24 PM	C++ Header file	11 KB
vl53l0x_interrupt_threshold_settings.h	1/5/2017 7:12 PM	C++ Header file	5 KB
vl53l0x_tuning.h	1/5/2017 7:12 PM	C++ Header file	4 KB

```
/********************
********************
********************
******************
 * Copyright © 2016,
STMicroelectronics
International N.V.
All rights reserved.

Redistribution and
use in source and
binary forms, with or
without
modification, are
permitted provided
that the following
conditions are met:
  *
Redistributions of
```

9 items selected Date modified: 1/5/2017 7:12 PM
Size: 134 KB

Computer ▸ OSDisk (C:) ▸ Users ▸ andy lin ▸ Desktop ▸ STM32 Summit ▸ Source Code ▸ X-CUBE-53L0A1 ▸ STM32CubeExpansion_VL53L0X_V1.2.0 ▸ Drivers ▸ BSP ▸ Components ▸ vl53l0x
Search vl53...
Organize ▾
Include in library ▾
Share with ▾
Burn
New folder

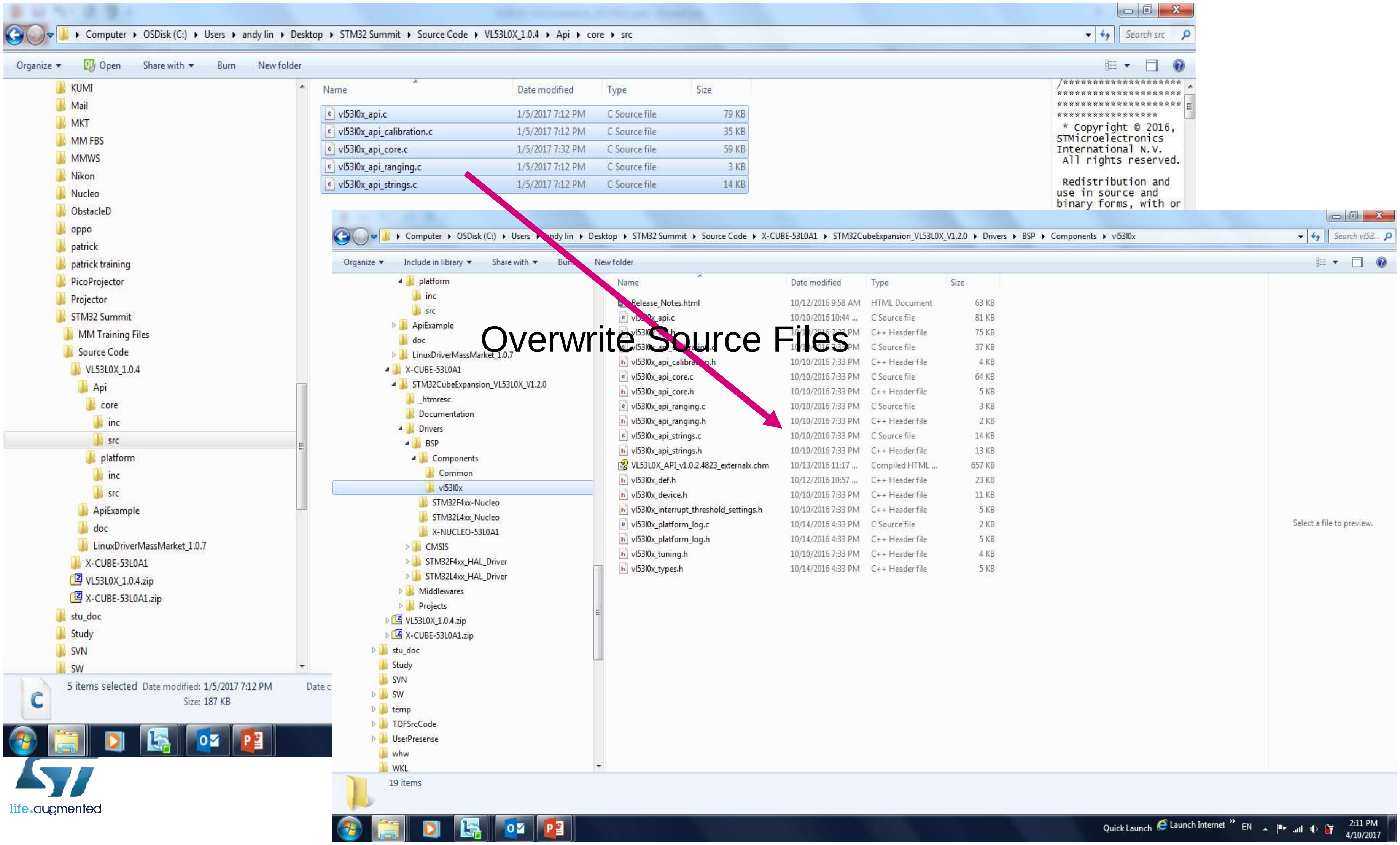
Overwrite Header Files

Name	Date modified	Type	Size
Release_Notes.html	10/12/2016 9:58 AM	HTML Document	63 KB
vl53l0x_api.c	10/1[illegible] 10:44 ...	C Source file	81 KB
vl53l0x_api.h	10/10/2016 7:33 PM	C++ Header file	75 KB
vl53l0x_api_calibration.c	10/10/2016 7:33 PM	C Source file	37 KB
vl53l0x_api_calibration.h	10/10/2016 7:33 PM	C++ Header file	4 KB
vl53l0x_api_core.c	10/10/2016 7:33 PM	C Source file	64 KB
vl53l0x_api_core.h	10/10/2016 7:33 PM	C++ Header file	5 KB
vl53l0x_api_ranging.c	10/10/2016 7:33 PM	C Source file	3 KB
vl53l0x_api_ranging.h	10/10/2016 7:33 PM	C++ Header file	2 KB
vl53l0x_api_strings.c	10/10/2016 7:33 PM	C Source file	14 KB
vl53l0x_api_strings.h	10/10/2016 7:33 PM	C++ Header file	13 KB
VL53L0X_API_v1.0.2.4823_externalx.chm	10/13/2016 11:17 ...	Compiled HTML ...	657 KB
vl53l0x_def.h	10/12/2016 10:57 ...	C++ Header file	23 KB
vl53l0x_device.h	10/10/2016 7:33 PM	C++ Header file	11 KB
vl53l0x_interrupt_threshold_settings.h	10/10/2016 7:33 PM	C++ Header file	5 KB
vl53l0x_platform_log.c	10/14/2016 4:33 PM	C Source file	2 KB
vl53l0x_platform_log.h	10/14/2016 4:33 PM	C++ Header file	5 KB
vl53l0x_tuning.h	10/10/2016 7:33 PM	C++ Header file	4 KB
vl53l0x_types.h	10/14/2016 4:33 PM	C++ Header file	5 KB

Select a file to preview.

19 items

Update Latest API to X-CUBE-53L0A1 (3)



Computer ▸ OSDisk (C:) ▸ Users ▸ andy lin ▸ Desktop ▸ STM32 Summit ▸ Source Code ▸ VL53L0X_1.0.4 ▸ Api ▸ core ▸ src

Name	Date modified	Type	Size
vl53l0x_api.c	1/5/2017 7:12 PM	C Source file	79 KB
vl53l0x_api_calibration.c	1/5/2017 7:12 PM	C Source file	35 KB
vl53l0x_api_core.c	1/5/2017 7:32 PM	C Source file	59 KB
vl53l0x_api_ranging.c	1/5/2017 7:12 PM	C Source file	3 KB
vl53l0x_api_strings.c	1/5/2017 7:12 PM	C Source file	14 KB

```
/*******************
*******************
*******************
*****************
 * Copyright © 2016,
STMicroelectronics
International N.V.
 All rights reserved.

 Redistribution and
use in source and
binary forms, with or
```

5 items selected Date modified: 1/5/2017 7:12 PM
Size: 187 KB

Computer ▸ OSDisk (C:) ▸ Users ▸ andy lin ▸ Desktop ▸ STM32 Summit ▸ Source Code ▸ X-CUBE-53L0A1 ▸ STM32CubeExpansion_VL53L0X_V1.2.0 ▸ Drivers ▸ BSP ▸ Components ▸ vl53l0x

Name	Date modified	Type	Size
Release_Notes.html	10/12/2016 9:58 AM	HTML Document	63 KB
vl53l0x_api.c	10/10/2016 10:44 ...	C Source file	81 KB
vl53l0x_api.h	10/10/2016 7:33 PM	C++ Header file	75 KB
vl53l0x_api_calibration.c	10/10/2016 7:33 PM	C Source file	37 KB
vl53l0x_api_calibration.h	10/10/2016 7:33 PM	C++ Header file	4 KB
vl53l0x_api_core.c	10/10/2016 7:33 PM	C Source file	64 KB
vl53l0x_api_core.h	10/10/2016 7:33 PM	C++ Header file	5 KB
vl53l0x_api_ranging.c	10/10/2016 7:33 PM	C Source file	3 KB
vl53l0x_api_ranging.h	10/10/2016 7:33 PM	C++ Header file	2 KB
vl53l0x_api_strings.c	10/10/2016 7:33 PM	C Source file	14 KB
vl53l0x_api_strings.h	10/10/2016 7:33 PM	C++ Header file	13 KB
VL53L0X_API_v1.0.2.4823_externalx.chm	10/13/2016 11:17 ...	Compiled HTML ...	657 KB
vl53l0x_def.h	10/12/2016 10:57 ...	C++ Header file	23 KB
vl53l0x_device.h	10/10/2016 7:33 PM	C++ Header file	11 KB
vl53l0x_interrupt_threshold_settings.h	10/10/2016 7:33 PM	C++ Header file	5 KB
vl53l0x_platform_log.c	10/14/2016 4:33 PM	C Source file	2 KB
vl53l0x_platform_log.h	10/14/2016 4:33 PM	C++ Header file	5 KB
vl53l0x_tuning.h	10/10/2016 7:33 PM	C++ Header file	4 KB
vl53l0x_types.h	10/14/2016 4:33 PM	C++ Header file	5 KB

19 items

Select a file to preview.

Overwrite Source Files

life.augmented

Install STM32 IDE of your choice (1/2)

- Pre-configured projects are available for
  - Keil : http://www.keil.com/
  - IAR : https://www.iar.com/
  - STM32 Workbench (Eclipse-based) : http://www.openstm32.org/HomePage

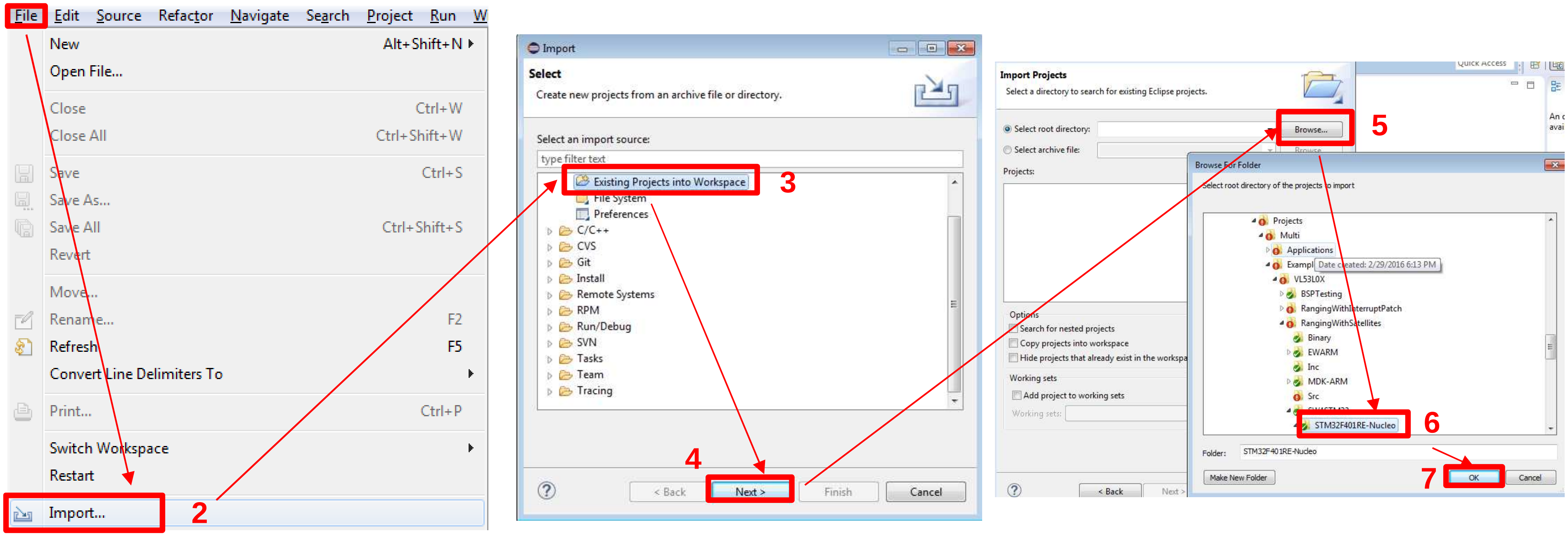
- Rest of the document will focus on STM32 Workbench (SW4STM32) as it is free and full featured (no code limit). Be aware that compiling the projects with all features (including data logging) will exceed the 32 KB limit of the free editions of Keil and IAR

Install STM32 IDE of your choice (2/2)

- Go here : http://www.openstm32.org/System+Workbench+for+STM32
- Log in or register
- Follow the install procedure given here : http://www.openstm32.org/Installing+System+Workbench+for+STM32+with+installer
  - Workbench for STM32 installer (Windows 7 64 bits) : install_sw4stm32_win_64bits-v1.3.exe
  - JAVARE needed : you'll get redirected to the Oracle JAVA website.
    - Warning, The architecture version for System Workbench for STM32 MUST be identical to your Java architecture version. (example : SW4 STM32 64bits only works with JavaRE 7(and upper) 64 bits)

Import a project in SW4STM32

- Start STM32 System Workbench
- Import the project as follows…



Create new projects from an archive file or directory.

Select an import source:

type filter text

Select a directory to search for existing Eclipse projects.

● Select root directory:

○ Select archive file:

Projects:

Options

□ Search for nested projects

□ Copy projects into workspace

□ Hide projects that already exist in the workspa

Working sets

□ Add project to working sets

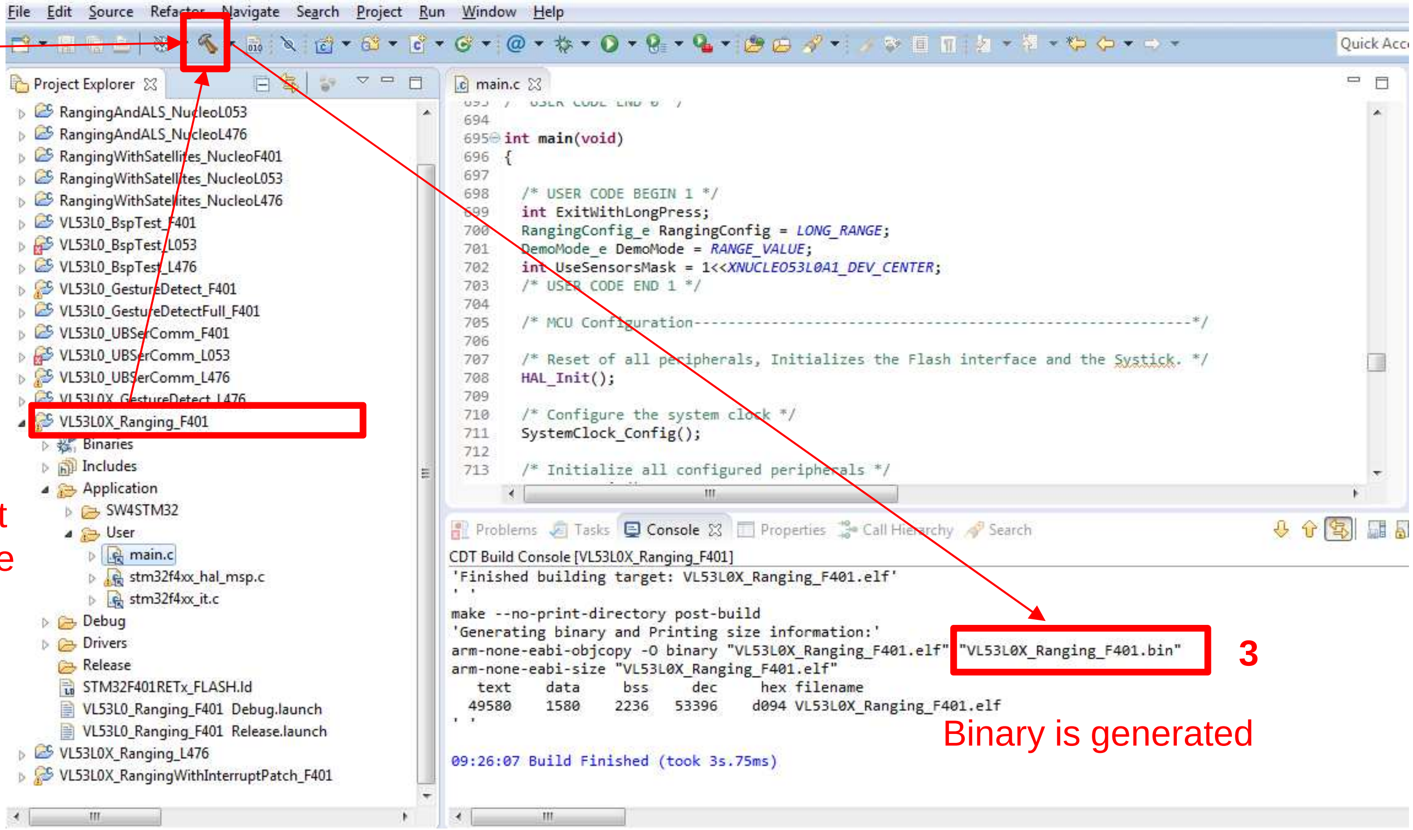
Working sets:

Select root directory of the projects to import

Date created: 2/29/2016 6:13 PM

Folder: STM32F401RE-Nucleo

Compile the project



```
int main(void)
{

  /* USER CODE BEGIN 1 */
  int ExitWithLongPress;
  RangingConfig_e RangingConfig = LONG_RANGE;
  DemoMode_e DemoMode = RANGE_VALUE;
  int UseSensorsMask = 1<<XNUCLEO53L0A1_DEV_CENTER;
  /* USER CODE END 1 */

  /* MCU Configuration----------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* Configure the system clock */
  SystemClock_Config();

  /* Initialize all configured peripherals */
```

CDT Build Console [VL53L0X_Ranging_F401]

```
'Finished building target: VL53L0X_Ranging_F401.elf'
' '
make --no-print-directory post-build
'Generating binary and Printing size information:'
arm-none-eabi-objcopy -O binary "VL53L0X_Ranging_F401.elf" "VL53L0X_Ranging_F401.bin"
arm-none-eabi-size "VL53L0X_Ranging_F401.elf"
   text    data     bss     dec     hex filename
  49580    1580    2236   53396    d094 VL53L0X_Ranging_F401.elf
' '

09:26:07 Build Finished (took 3s.75ms)
```

2

Build the project

1

Select a loaded project (several projects can be loaded at the same time in Eclipse)

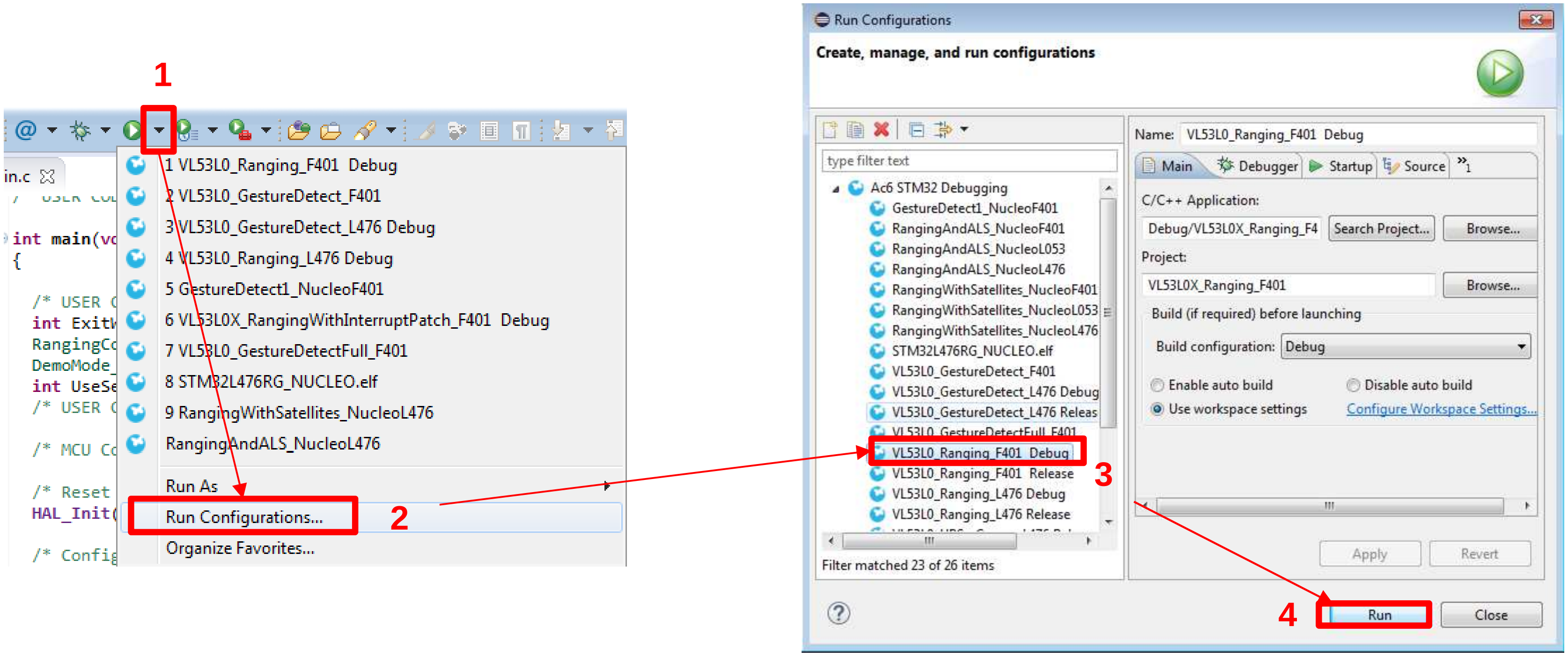
3

Binary is generated

Flash the board

- Connect Nucleo board to the PC

- The first time, select and launch the “run” configuration provided with the project



1 VL53L0_Ranging_F401 Debug
2 VL53L0_GestureDetect_F401
3 VL53L0_GestureDetect_L476 Debug
4 VL53L0_Ranging_L476 Debug
5 GestureDetect1_NucleoF401
6 VL53L0X_RangingWithInterruptPatch_F401 Debug
7 VL53L0_GestureDetectFull_F401
8 STM32L476RG_NUCLEO.elf
9 RangingWithSatellites_NucleoL476
RangingAndALS_NucleoL476
Run As
Run Configurations...
Organize Favorites...

```
int main(vo
{
  /* USER C
  int ExitW
  RangingCo
  DemoMode_
  int UseSe
  /* USER C

  /* MCU Co

  /* Reset
  HAL_Init(

  /* Config
```

Run Configurations

Create, manage, and run configurations

type filter text

- Ac6 STM32 Debugging
  - GestureDetect1_NucleoF401
  - RangingAndALS_NucleoF401
  - RangingAndALS_NucleoL053
  - RangingAndALS_NucleoL476
  - RangingWithSatellites_NucleoF401
  - RangingWithSatellites_NucleoL053
  - RangingWithSatellites_NucleoL476
  - STM32L476RG_NUCLEO.elf
  - VL53L0_GestureDetect_F401
  - VL53L0_GestureDetect_L476 Debug
  - VL53L0_GestureDetect_L476 Releas
  - VL53L0_GestureDetectFull_F401
  - VL53L0_Ranging_F401 Debug
  - VL53L0_Ranging_F401 Release
  - VL53L0_Ranging_L476 Debug
  - VL53L0_Ranging_L476 Release

Filter matched 23 of 26 items

Name: VL53L0_Ranging_F401 Debug

Main | Debugger | Startup | Source

C/C++ Application: Debug/VL53L0X_Ranging_F4 | Search Project... | Browse...

Project: VL53L0X_Ranging_F401 | Browse...

Build (if required) before launching

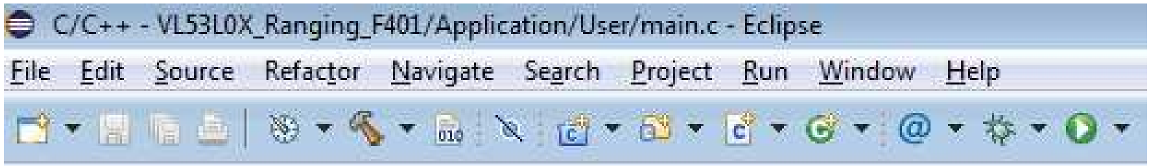
Build configuration: Debug

○ Enable auto build ○ Disable auto build
● Use workspace settings Configure Workspace Settings...

Apply Revert

Run Close

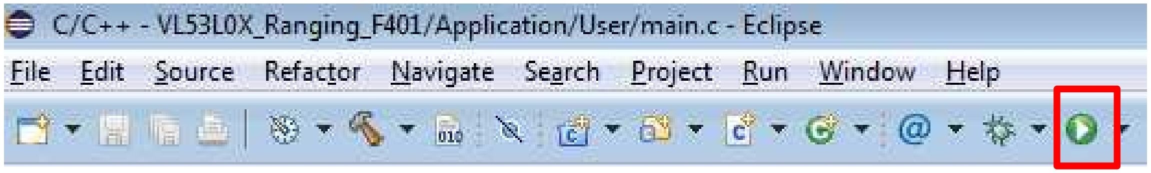
Use the buttons to go next, step into, add breakpoints, etc…



C/C++ - VL53L0X_Ranging_F401/Application/User/main.c - Eclipse

File Edit Source Refactor Navigate Search Project Run Window Help

Flash the board (1/2)

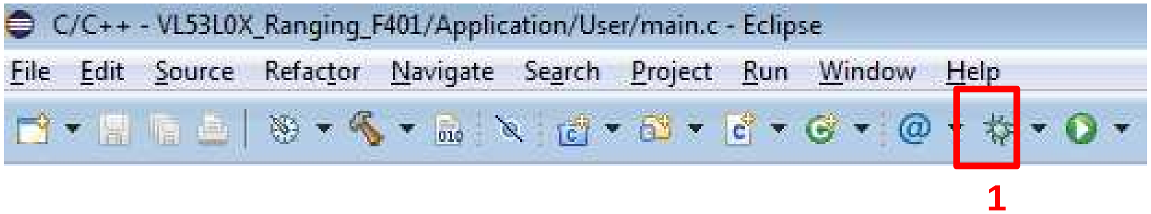

C/C++ - VL53L0X_Ranging_F401/Application/User/main.c - Eclipse
File
Edit
Source
Refactor
Navigate
Search
Project
Run
Window
Help
1

Next times, simply click on the Run greer le the project and flash the new code

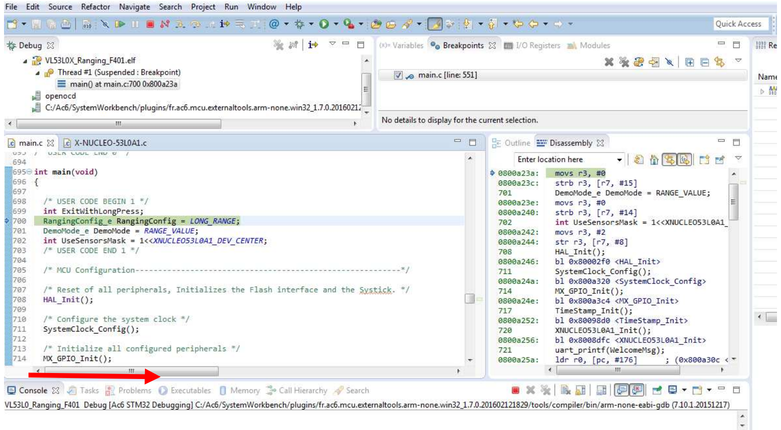
- Demo will start on Nucleo…

Debug

- Press the Debug green button : this will launch debugger of the previous run configuration


C/C++ - VL53L0X_Ranging_F401/Application/User/main.c - Eclipse
File
Edit
Source
Refactor
Navigate
Search
Project
Run
Window
Help
1

- This will open the Debug perspective in Eclipse, program will start and break at main()



```
VL53L0X_Ranging_F401.elf
  Thread #1 (Suspended : Breakpoint)
    main() at main.c:700 0x800a23a
  openocd
  C:/Ac6/SystemWorkbench/plugins/fr.ac6.mcu.externaltools.arm-none.win32_1.7.0.2016021
```

```
☑ main.c [line: 551]
```

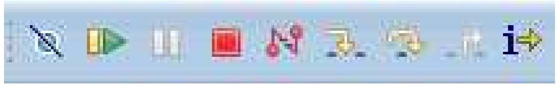
No details to display for the current selection.

```c
int main(void)
{

  /* USER CODE BEGIN 1 */
  int ExitWithLongPress;
  RangingConfig_e RangingConfig = LONG_RANGE;
  DemoMode_e DemoMode = RANGE_VALUE;
  int UseSensorsMask = 1<<XNUCLEO53L0A1_DEV_CENTER;
  /* USER CODE END 1 */

  /* MCU Configuration----------------------------------------------------------*/

  /* Reset of all peripherals, Initializes the Flash interface and the Systick. */
  HAL_Init();

  /* Configure the system clock */
  SystemClock_Config();

  /* Initialize all configured peripherals */
  MX_GPIO_Init();
```

```
0800a23a:   movs r3, #0
0800a23c:   strb r3, [r7, #15]
701         DemoMode_e DemoMode = RANGE_VALUE;
0800a23e:   movs r3, #0
0800a240:   strb r3, [r7, #14]
702         int UseSensorsMask = 1<<XNUCLEO53L0A1_
0800a242:   movs r3, #2
0800a244:   str r3, [r7, #8]
708         HAL_Init();
0800a246:   bl 0x80002f0 <HAL_Init>
711         SystemClock_Config();
0800a24a:   bl 0x800a320 <SystemClock_Config>
714         MX_GPIO_Init();
0800a24e:   bl 0x800a3c4 <MX_GPIO_Init>
717         TimeStamp_Init();
0800a252:   bl 0x80098d0 <TimeStamp_Init>
720         XNUCLEO53L0A1_Init();
0800a256:   bl 0x8008dfc <XNUCLEO53L0A1_Init>
721         uart_printf(WelcomeMsg);
0800a25a:   ldr r0, [pc, #176]      ; (0x800a30c <
```

```
VL53L0_Ranging_F401  Debug [Ac6 STM32 Debugging] C:/Ac6/SystemWorkbench/plugins/fr.ac6.mcu.externaltools.arm-none.win32_1.7.0.201602121829/tools/compiler/bin/arm-none-eabi-gdb (7.10.1.20151217)
```

Use the buttons to go next, step into, add breakpoints, etc…



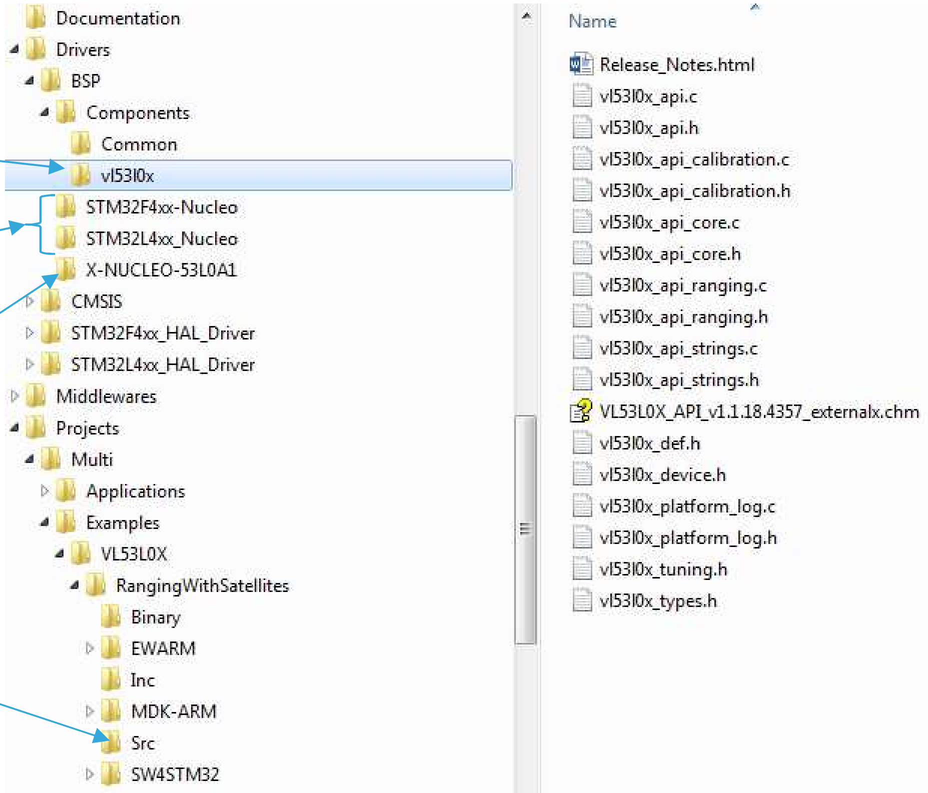
RangingWithSatellites code review (1/5)

VL53L0X API

Nucleo/STM32 Hardware Abstraction Layer

X-NUCLEO-53L0A1 expansion board API

main.c file

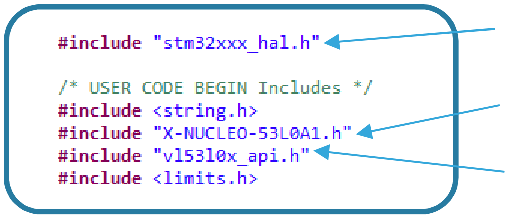


```
Documentation
Drivers
  BSP
    Components
      Common
      vl53l0x
    STM32F4xx-Nucleo
    STM32L4xx_Nucleo
    X-NUCLEO-53L0A1
  CMSIS
  STM32F4xx_HAL_Driver
  STM32L4xx_HAL_Driver
Middlewares
Projects
  Multi
    Applications
    Examples
      VL53L0X
        RangingWithSatellites
          Binary
          EWARM
          Inc
          MDK-ARM
          Src
          SW4STM32
```

```
Name
Release_Notes.html
vl53l0x_api.c
vl53l0x_api.h
vl53l0x_api_calibration.c
vl53l0x_api_calibration.h
vl53l0x_api_core.c
vl53l0x_api_core.h
vl53l0x_api_ranging.c
vl53l0x_api_ranging.h
vl53l0x_api_strings.c
vl53l0x_api_strings.h
VL53L0X_API_v1.1.18.4357_externalx.chm
vl53l0x_def.h
vl53l0x_device.h
vl53l0x_platform_log.c
vl53l0x_platform_log.h
vl53l0x_tuning.h
vl53l0x_types.h
```

life.augmented

RangingWithSatellites code review (2/5)

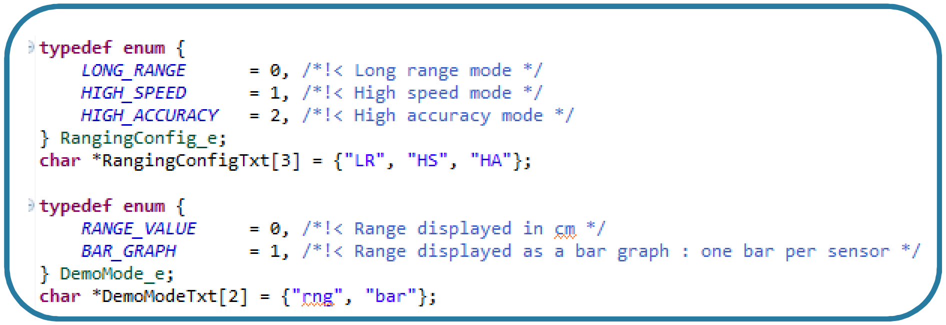


```
#include "stm32xxx_hal.h"

/* USER CODE BEGIN Includes */
#include <string.h>
#include "X-NUCLEO-53L0A1.h"
#include "vl53l0x_api.h"
#include <limits.h>
```

This file is located in each project “inc” directory and does the link towards the targeted CPU : F401 or L476. This allows to share the same main.c file with all boards…

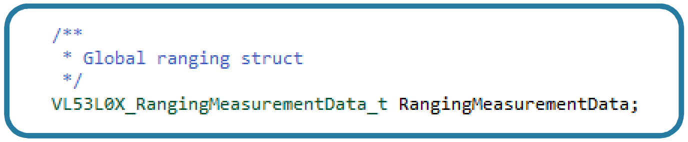
The X-NUCLEO-53L0A1 Expansion Board API located in Drivers/BSP/X-NUCLEO-53L0A1

The VL53L0X API located in Drivers/BSP/Components/vl53l0x



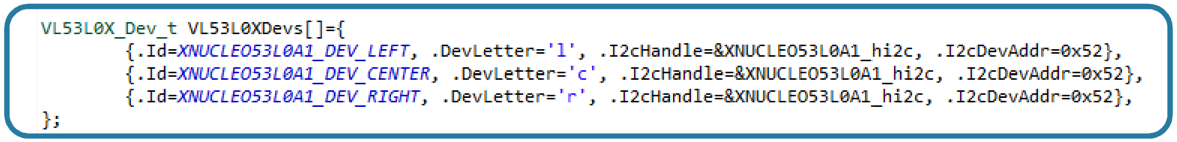
```
typedef enum {
    LONG_RANGE      = 0, /*!< Long range mode */
    HIGH_SPEED      = 1, /*!< High speed mode */
    HIGH_ACCURACY   = 2, /*!< High accuracy mode */
} RangingConfig_e;
char *RangingConfigTxt[3] = {"LR", "HS", "HA"};

typedef enum {
    RANGE_VALUE     = 0, /*!< Range displayed in cm */
    BAR_GRAPH       = 1, /*!< Range displayed as a bar graph : one bar per sensor */
} DemoMode_e;
char *DemoModeTxt[2] = {"rng", "bar"};
```

Key global variables to manage the various modes of the demo



```
/**
 * Global ranging struct
 */
VL53L0X_RangingMeasurementData_t RangingMeasurementData;
```

The VL53L0X Ranging Measurement data structure



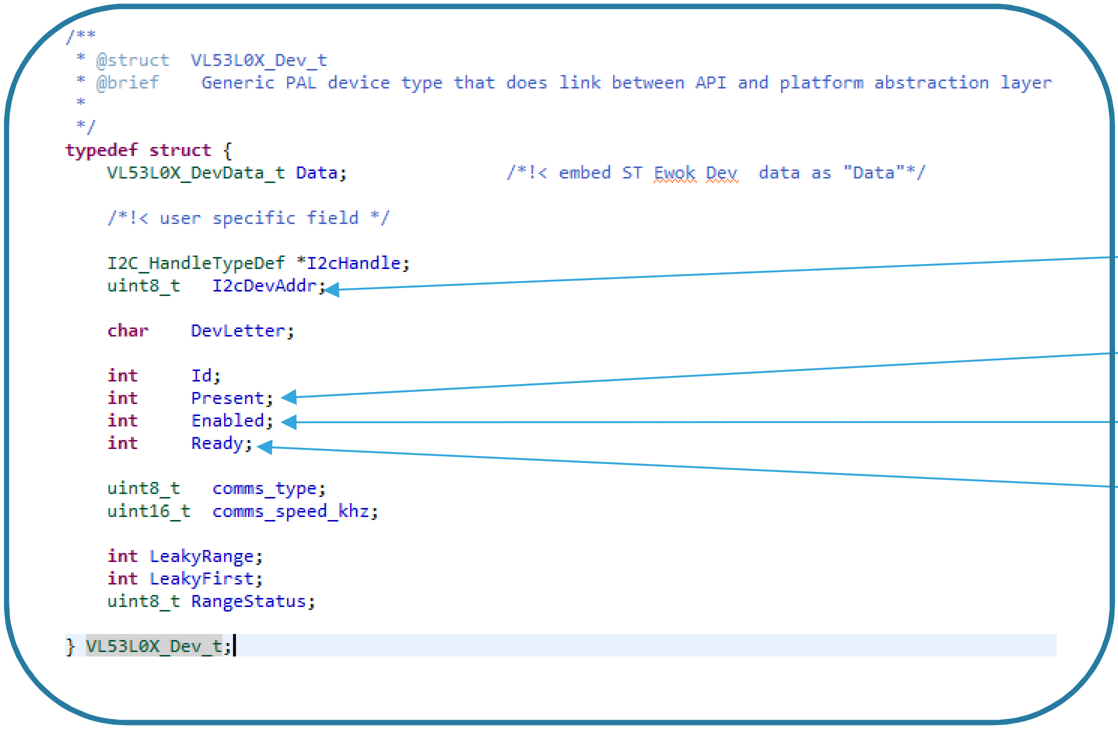
```
VL53L0X_Dev_t VL53L0XDevs[]={
        {.Id=XNUCLEO53L0A1_DEV_LEFT, .DevLetter='l', .I2cHandle=&XNUCLEO53L0A1_hi2c, .I2cDevAddr=0x52},
        {.Id=XNUCLEO53L0A1_DEV_CENTER, .DevLetter='c', .I2cHandle=&XNUCLEO53L0A1_hi2c, .I2cDevAddr=0x52},
        {.Id=XNUCLEO53L0A1_DEV_RIGHT, .DevLetter='r', .I2cHandle=&XNUCLEO53L0A1_hi2c, .I2cDevAddr=0x52},
};
```

3 instances of the VL53L0X Device structures : one per device (see next slide)

life.augmented

RangingWithSatellites code review (3/5)

The VL53L0X Device structure definition : vl53l0x_platform.h in Drivers/BSP/X-NUCLEO-53L0A1
(contains all what is needed for one sensor to range)



```
/**
 * @struct  VL53L0X_Dev_t
 * @brief    Generic PAL device type that does link between API and platform abstraction layer
 *
 */
typedef struct {
    VL53L0X_DevData_t Data;               /*!< embed ST Ewok Dev  data as "Data"*/

    /*!< user specific field */

    I2C_HandleTypeDef *I2cHandle;
    uint8_t   I2cDevAddr;

    char    DevLetter;

    int     Id;
    int     Present;
    int     Enabled;
    int     Ready;

    uint8_t   comms_type;
    uint16_t  comms_speed_khz;

    int LeakyRange;
    int LeakyFirst;
    uint8_t RangeStatus;

} VL53L0X_Dev_t;
```

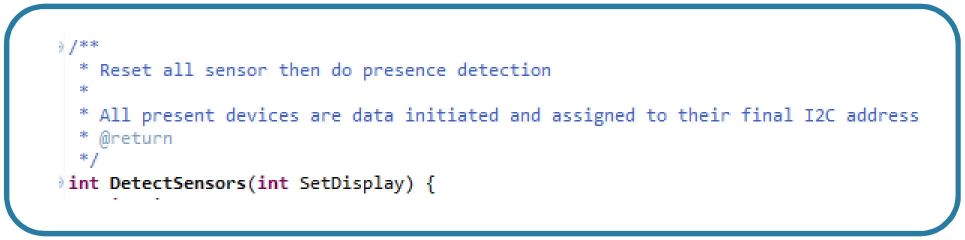
I2C address : will be different for each sensor

Sensor is detected on the board

Sensor ranging is enabled

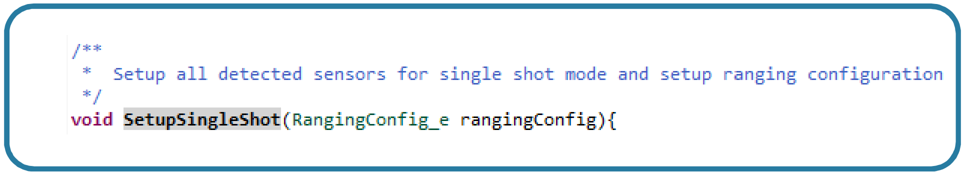
New ranging sample is ready

RangingWithSatellites code review (4/5)



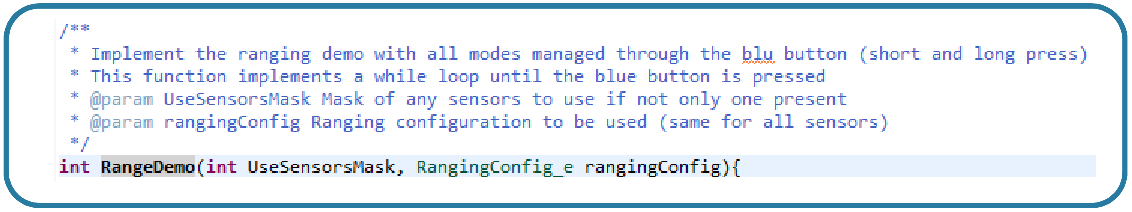
```
/**
 * Reset all sensor then do presence detection
 *
 * All present devices are data initiated and assigned to their final I2C address
 * @return
 */
int DetectSensors(int SetDisplay) {
```

This function can be called at any time from main() : Typically, to check the number of sensors connected on the board and initialize them to their final I2C addresses : “Present” fields of each Device structure is updated.



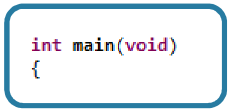
```
/**
 *  Setup all detected sensors for single shot mode and setup ranging configuration
 */
void SetupSingleShot(RangingConfig_e rangingConfig){
```

Each present sensor (see previous function) is initialized for ranging in single short mode and with the given ranging configuration.



```
/**
 * Implement the ranging demo with all modes managed through the blu button (short and long press)
 * This function implements a while loop until the blue button is pressed
 * @param UseSensorsMask Mask of any sensors to use if not only one present
 * @param rangingConfig Ranging configuration to be used (same for all sensors)
 */
int RangeDemo(int UseSensorsMask, RangingConfig_e rangingConfig){
```

This implements the ranging demo state machine



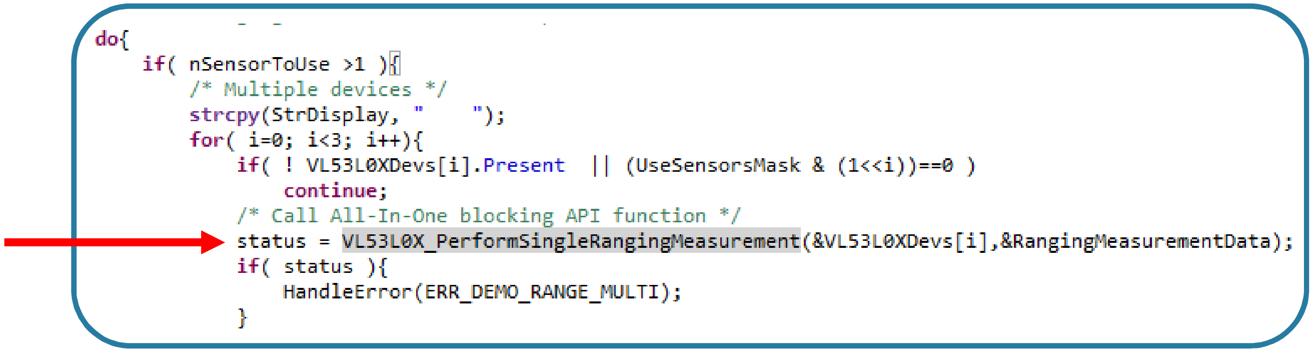
```
int main(void)
{
```

Init hardware and calls RangeDemo

RangingWithSatellites code review (5/5)

- Ranging operation for each enabled device is performed by the VL53L0X_PerformSingleRangingMeasurement API function (called in the RangeDemo() function)

- The above function being blocking, multi-devices ranging is sequential (not simultaneous)



```
do{
    if( nSensorToUse >1 ){
        /* Multiple devices */
        strcpy(StrDisplay, "    ");
        for( i=0; i<3; i++){
            if( ! VL53L0XDevs[i].Present  || (UseSensorsMask & (1<<i))==0 )
                continue;
            /* Call All-In-One blocking API function */
            status = VL53L0X_PerformSingleRangingMeasurement(&VL53L0XDevs[i],&RangingMeasurementData);
            if( status ){
                HandleError(ERR_DEMO_RANGE_MULTI);
            }
```

RangeDemo() extract (main.c)

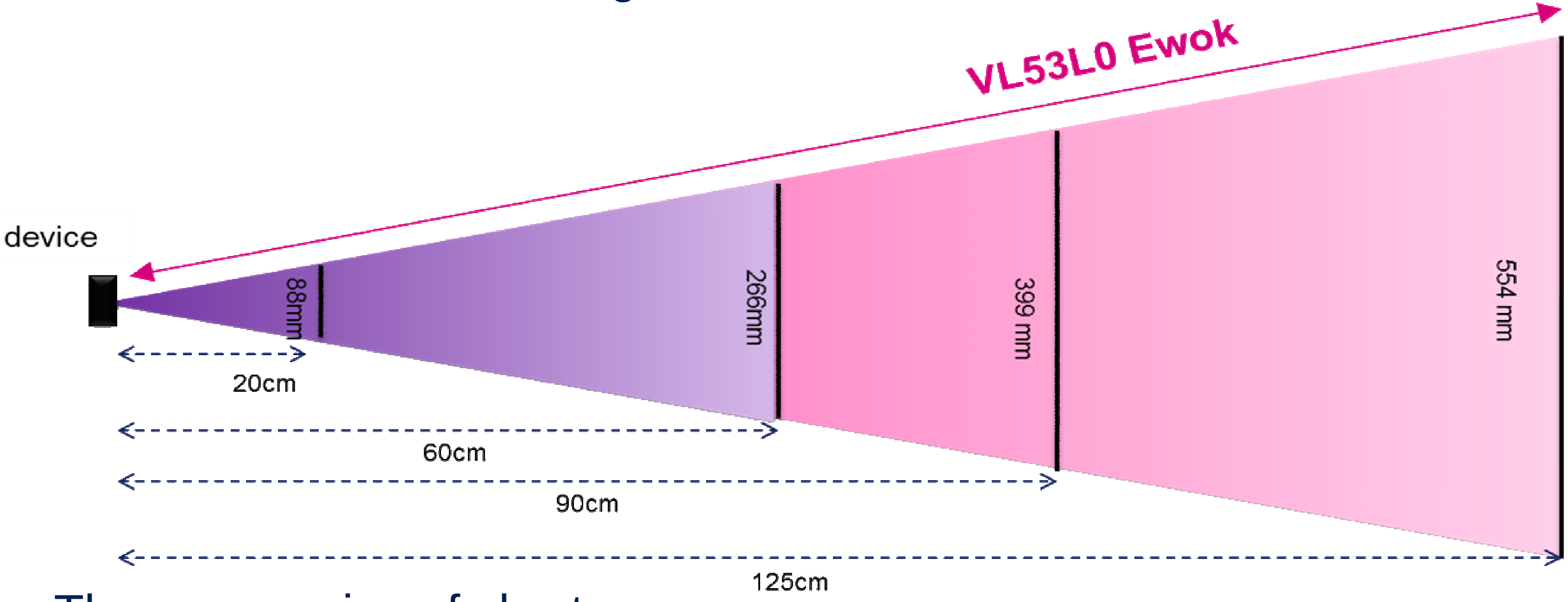
Dark Box Design For Calibration VL53L0

Apr 2017

life.augmented

Chart Square Area

- VL53L0 device act as single central zone with FOV of ~25º



- The square size of chart

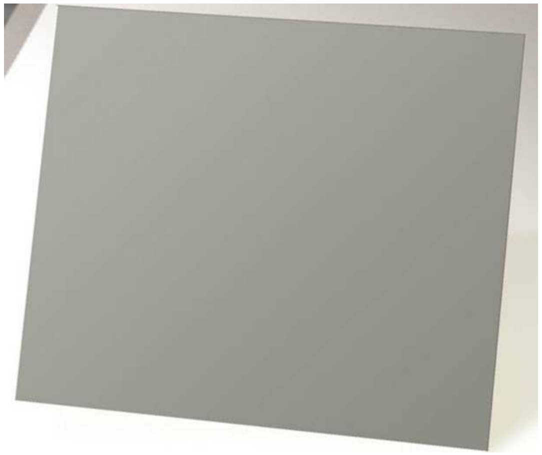
Chart/Distance	Minimum Square Area
White (88%)/ 100mm	44mm X 44mm
Gray (17%)/ 700mm	310mm X 310mm

Chart Spec

- ST’ suggestion
  - Munsell Chart

Chart	xRite ST P/N
Munsell_N4.75 (17%)	STMN475/MA4 (17%)
Munsell_N9.5 (88%)	STMN95/MA4 (88%)

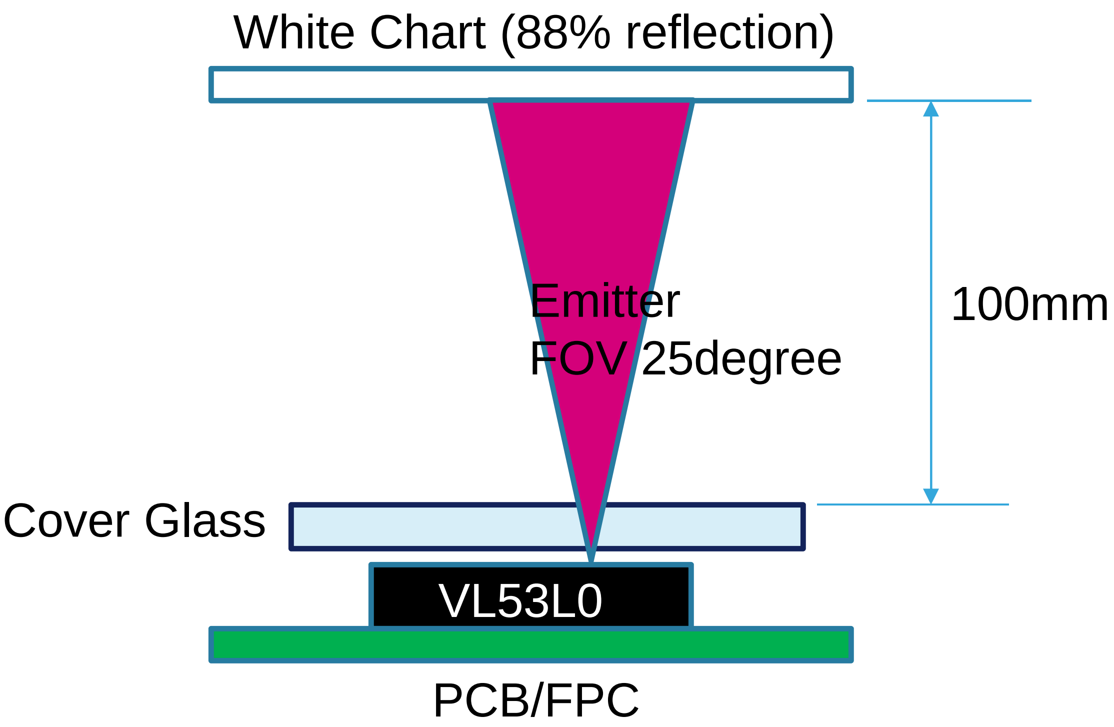
- Distributor info
  - Agency “Deep Blue”, 深藍科技股份有限公司,
    - gavin@deepblue.com.tw
    - 邱仁俊(0936-272-850)
  - Gain Associates Inc.
    - Sales: Alex LEE
    - Mobile: 0930 290 510
    - Office: +886 2 8661 0010 ext.200
    - Website: www.gain.com.tw



Offset Calibration

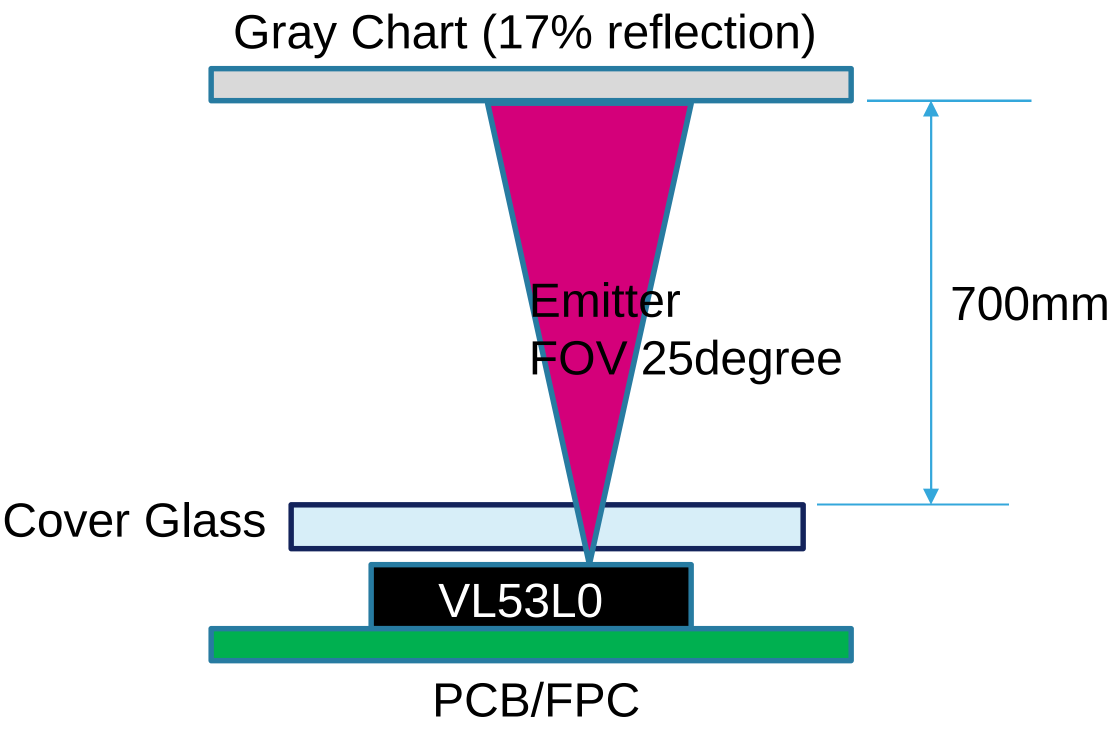
- Please use the offset calibration code to get and store into the memory

- Use 88% white card on 100mm distance from top of cover glass.



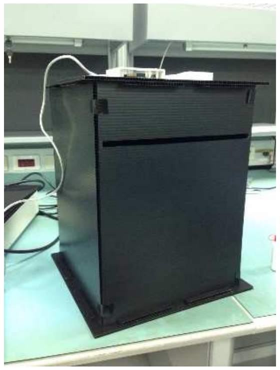
Cross Talk Calibration

- Please use the Cross talk calibration code to get and store into the memory
- Use 17% gray card on 700mm distance from top of cover glass.



Dark Box Example

- For one VL53L0 device
  - Box size should be 315mm X 315mm X 700mm
  - There are two chart for different distance.
  - One white chart is on 100mm above VL53L0 for offset calibration.
  - The other gray chart is on 700mm above VL53L0 for xtalk calibration.

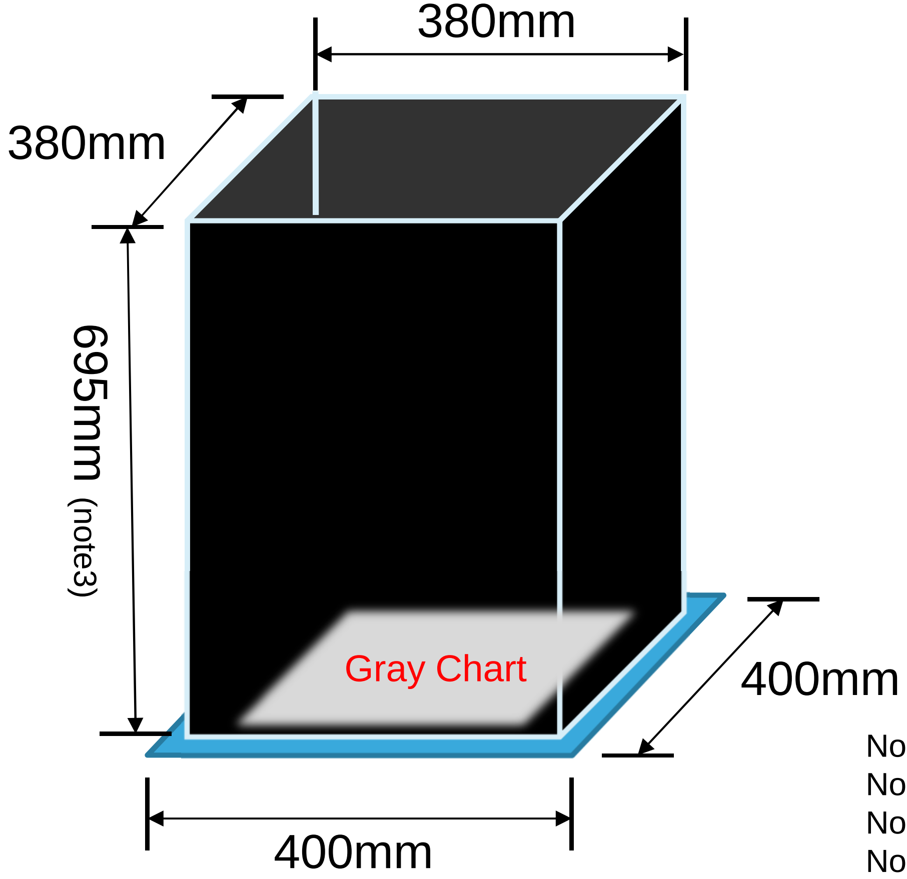


Example for EVK

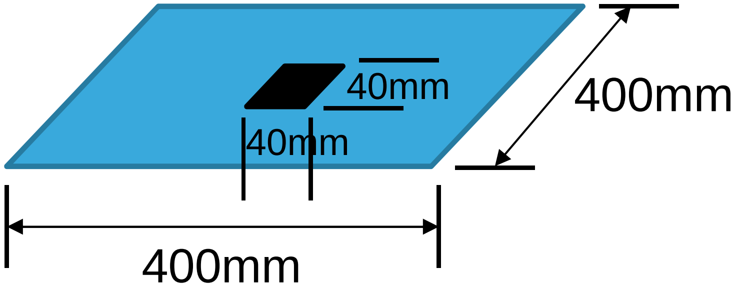
Chart/Distance	Minimum Square Area one VL53L0
White (88%)/ 100mm	44mm X 44mm
Gray (17%)/ 700mm	310mm X 310mm

Example of Box

Body of box



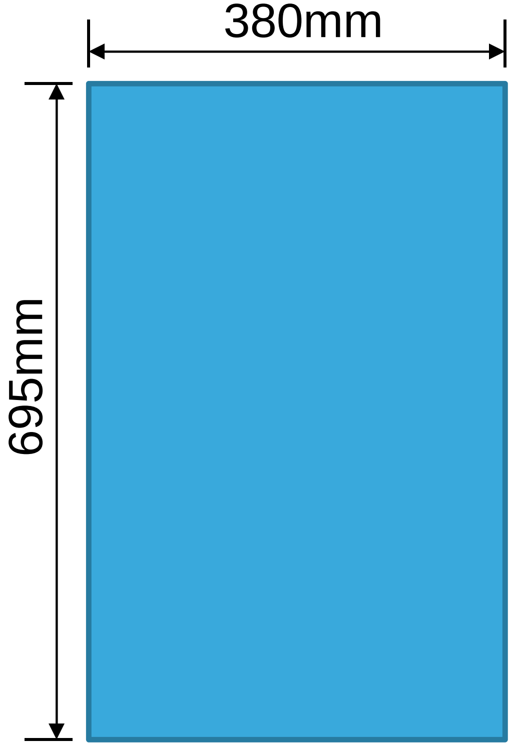
Cover of box



Note1: Board thickness is 5mm
Note2: Need to tape to fix the board
Note3: The length should be changed by cover glass quality
Note4: Please refer “EwokCut1 1 Xtalk calibration distance_v2” to choice the length of note3

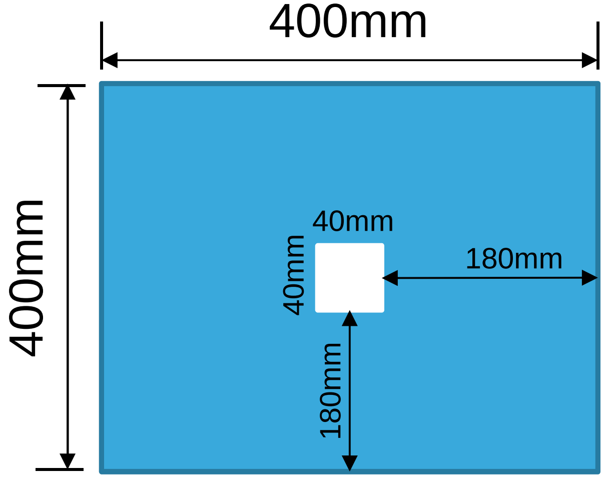
Disassembling Part of Box

Side Board

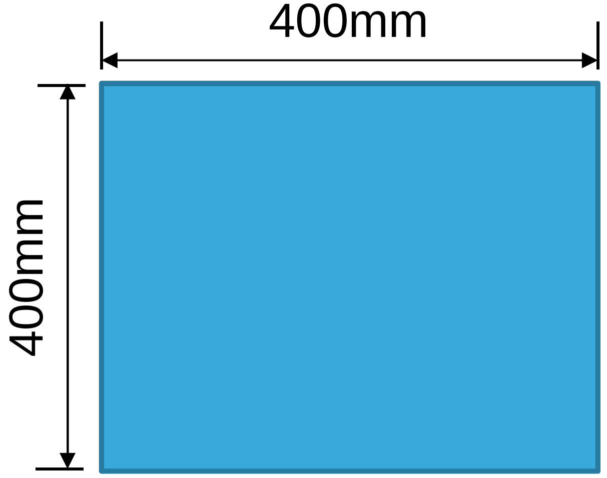


X 4 pcs

Top Board



Bottom Board



ID design and cover window guide for VL53L0X

Apr 2017

Scope

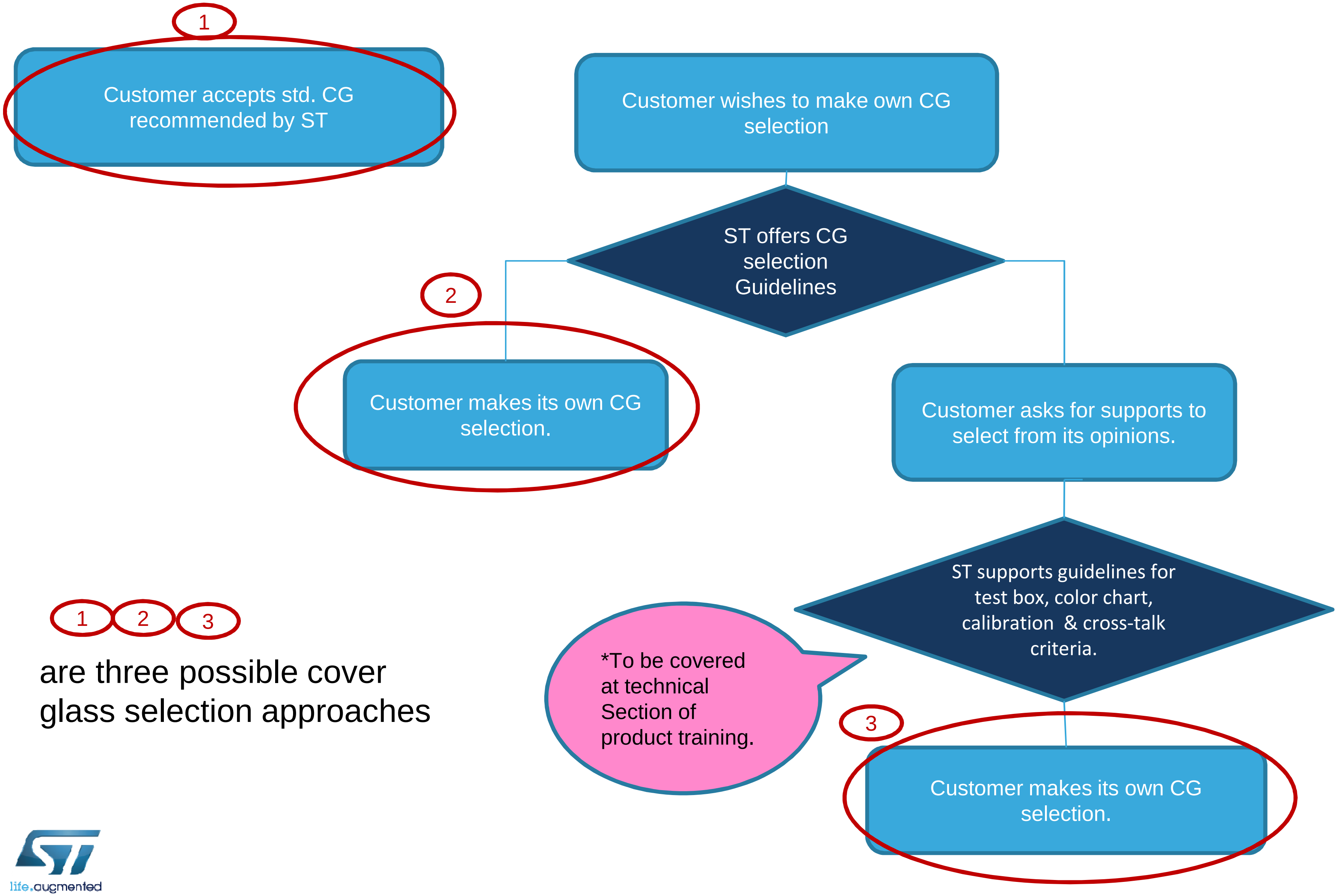
- Provide guidelines for ID design and how to assess cover-window quality
- Expected performance level of an optimised cover-window with 0.5mm air-gap
  - PMMA embedded filtering can reach 0.1~0.3kcps ST's crosstalk.
  - Gorilla Glass can achieve at best 0.3~0.7kcps ST's crosstalk.
  - PC material can achieve at <0.7kcps ST's crosstalk.
- In this document, ST shares recommendations on cover-window selection and Design Requirements for optimising the Systems.
- ST's recommends parametric values based on experience for "Best in Class" Applications.
  - If the Industrial Design and cover-window quality deviates from ST's recommended ''Best In Class'' recommendation, the system performance can be negatively impacted. It is the integrator's responsibility to perform the system study and define their own system specification.

Why a cover window is needed above ST's sensor ?

- The cover window serves two main purposes:
  - To provide physical protection of the device, including dust ingress prevention.
  - To provide optical filtering for the sensor.

- The cover window will normally be opaque with either two circular apertures or one oval aperture to allow the emission and receiving of light.

Cover Glass Management



1

Customer accepts std. CG recommended by ST

Customer wishes to make own CG selection

ST offers CG selection Guidelines

2

Customer makes its own CG selection.

Customer asks for supports to select from its opinions.

ST supports guidelines for test box, color chart, calibration & cross-talk criteria.

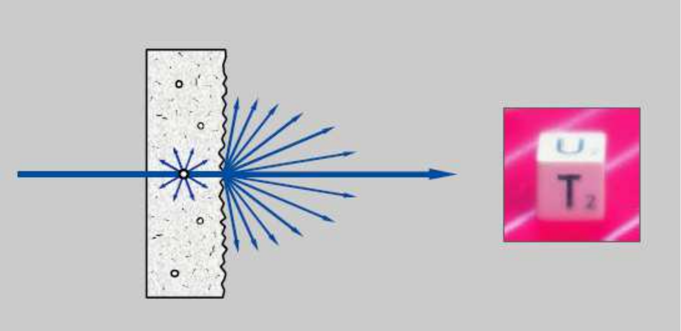
*To be covered at technical Section of product training.

3

Customer makes its own CG selection.

1 2 3

are three possible cover glass selection approaches



Embedded particles/holes or rough surface are major contributor to light scattering in cover window

Ideal ID & window

- An **<u>Ideal cover window</u>** has:
  - No structural defects in the plastic or glass material
  - No surface defects that can induce light scatter or smudge sensitivity with fingerprint
  - Transmission >90% in near-IR (940nm +/-100nm) and Haze <6%
  - Outer coatings that do not degrade immunity to fingerprint (Anti-fingerprint or Anti-Reflective coatings).
  - Single material. Use of Dual material may alter performance.

- An **<u>ideal ID design</u>** has:
  - Small airgap (<0.5mm)
  - Thin window (0.6mm is optimum for hardness and performance for PMMA)
  - Low window tilt <2degrees
  - Tight tolerances

Quality control

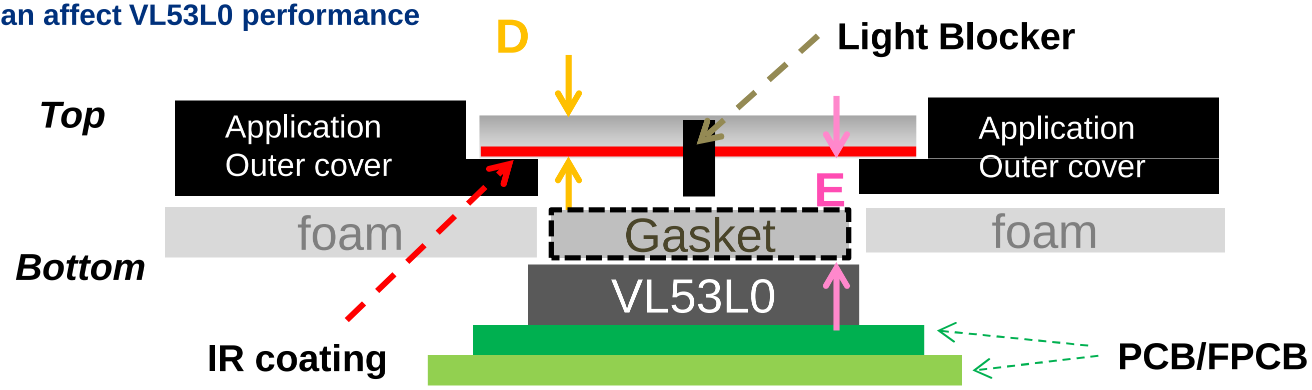
- ST’s ToF sensors measure permanently 2 key parameters to monitor window quality:
  - Return signal from the object (Transmission)
  - Crosstalk in Mcps or kcps (cps = Photon Counts per second) to measure cover window light scattering. But this also encompasses other parameters from the phone design (air-gap, light reflections in the phone housing etc…)

- Cover-window vendors usually control cover-window quality to ensure it is free of surface and structural defects.
  - Window vendor are all be able to measure the transmission of final product and need to control/monitor quality in production
  - Window vendors might not be able to control the level of scatter (clarity or haze) which is different from Transmission. Part of transmitted light will be lost in scattering and can impact the overall system.

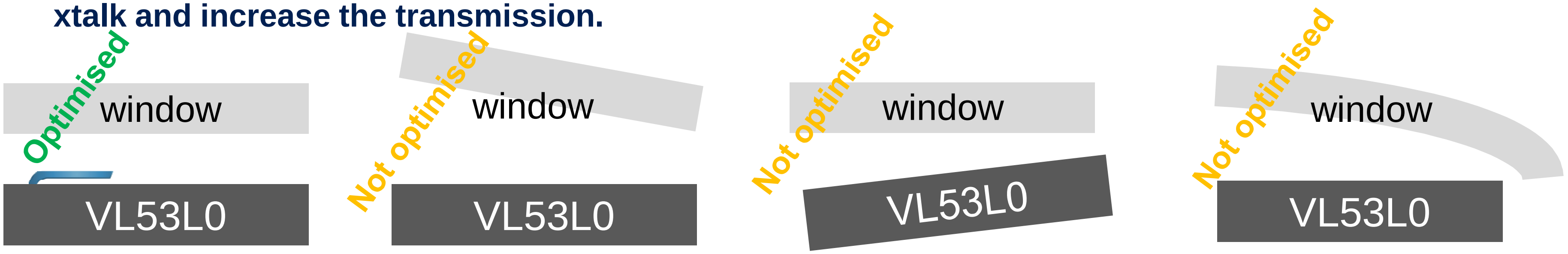
ID design recommendations

- It is Integrator responsibility to comply with Industrial Design (ID) and Coverwindow recommendations from ST to ensure optimised performances. A small airgap (E) and thin coverglass (D) with high transmittance is best. **If not possible to reduce to airgap & CG thickness then a gasket or window embedded Light Blocker is essential.**

- **Distance of VL53L0X to:**
  - **IR sensor: Avoid optical interference with other IR sensors emitting in same wavelength or do not activate at the same time. I using different wavelength, no risk of interference, any distance is ok.**
  - **Antennas: 5mm < distance**
  - **Flash: Depends on the use of Flash with AF-assist or Depth Map. If Flash temperature rises to >60degrees and VL53L0 is very close, it can affect VL53L0 performance**



- **For optimal performance, the cover-window needs to be parallel to VL53L0 to help reduce xtalk and increase the transmission.**



Cover Window soiling

- **Smudge** is also referred to as soiling from fingerprint, grease, dust, water or anything that can be on top of the cover-window and interfere the light from the sensor.

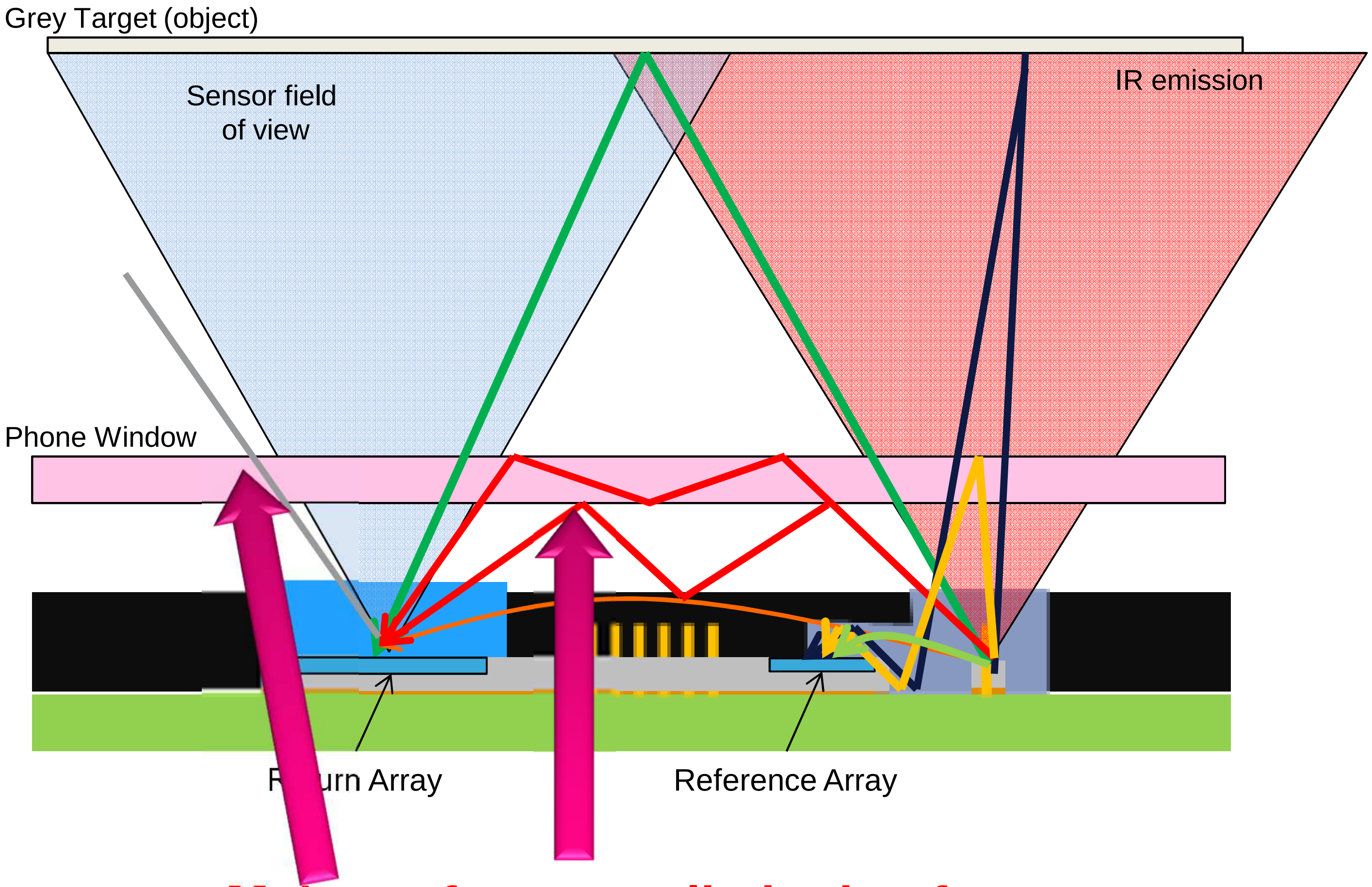
- Any protective film/coating with high surface tensile strength on top of coverwindow maybe considered sensitive for Time-Of-Flight technology. These materials are more sensitive to affect optical scattering with soling/smudge.

- Not all windows will be sensitive to smudge but impact needs to be assessed.

- What can result in high Xtalk with smudge on window ?
  - ID design, if system have high xtalk then it is likely to have more xtalk with smudge
  - Use of some coating on the cover-window (like some AFC or some Anti-Reflective coating)
  - Type of window surface finishing (roughness and Haze parameters allow verifying)
  - The window compound itself
  - *Or it could be the combination of above things*

Source: Schott materials

ST’s ToF Module & Optical Paths



- **Main performance limitation factors**

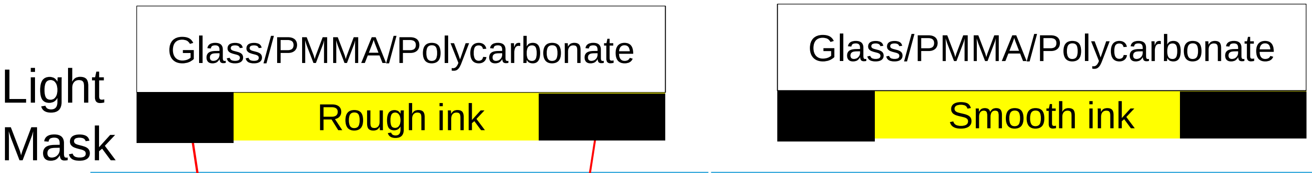
Signal

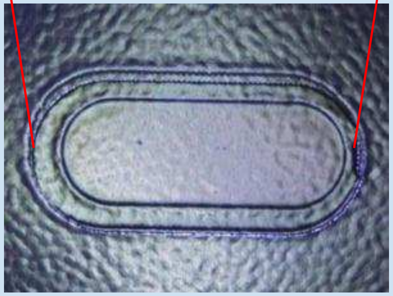
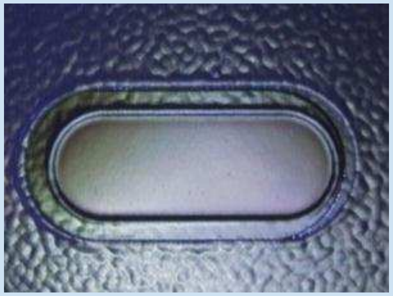
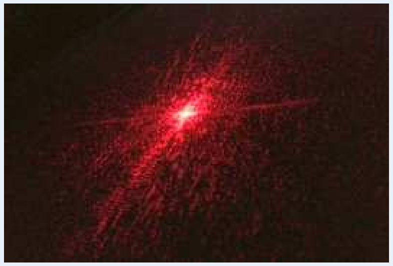
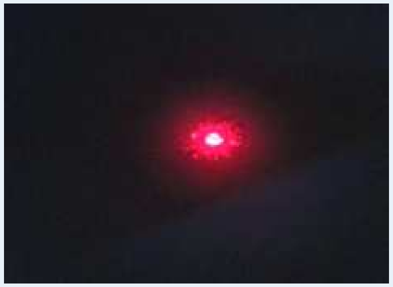
- Coupling
- Returns

Noise

- X-talk on reference
- Returns on reference
- Leakage
- X-talk on return
- Ambient on Reference
- Ambient on Return

Cover window coatings

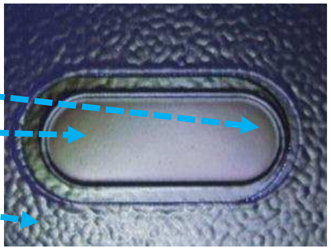


Rough ink	Smooth ink
	
	
High Cross talk	**Low Cross talk**

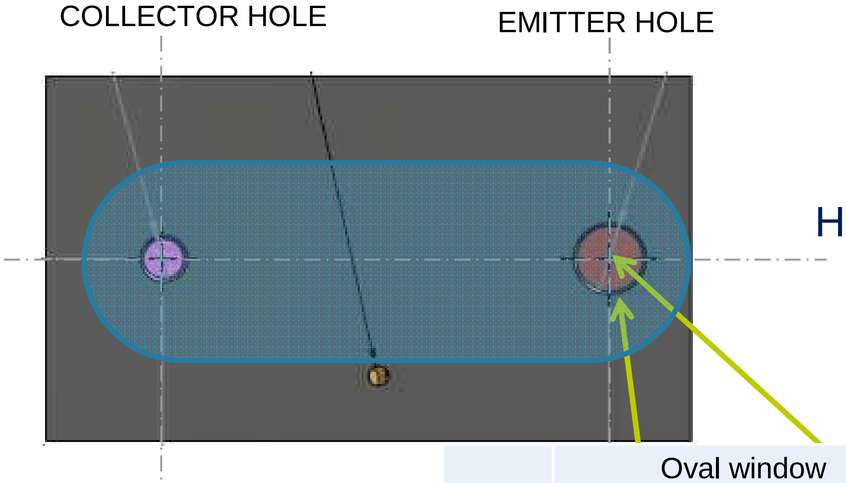
- Keep ‘glass’ surface finish smooth

Glass artwork

- **1-hole glass artwork is good *though 2-hole design is best***
  - Emitter hole: transparent (ideal 100% IR transmission)
  - Collector hole: high transparent Ink
  - Everywhere else: Coloured Light opaque paint is accepted



- **Production control of Applications assembly tolerances (x,y,z & tilt) at integrator is very important**. Documentation is available to calculate the minimum aperture size for a specific air gap between the VL53L0 and cover glass. *Please ask ST.*



Hole diameter

Window size Example
With ±0µm assembly
tolerance used

for other assembly tolerances refer to ST's hole size calculation document

	Oval window	
Air Gap	X(mm)	Y(mm)
0	3.00	0.40
0.05	3.00	0.43
0.1	3.00	0.46
0.15	3.00	0.49
0.2	3.00	0.53
0.3	3.00	0.59
0.4	3.00	0.65
0.5	3.00	0.72

Hornix (3rd Party) Cover Glass Introduction

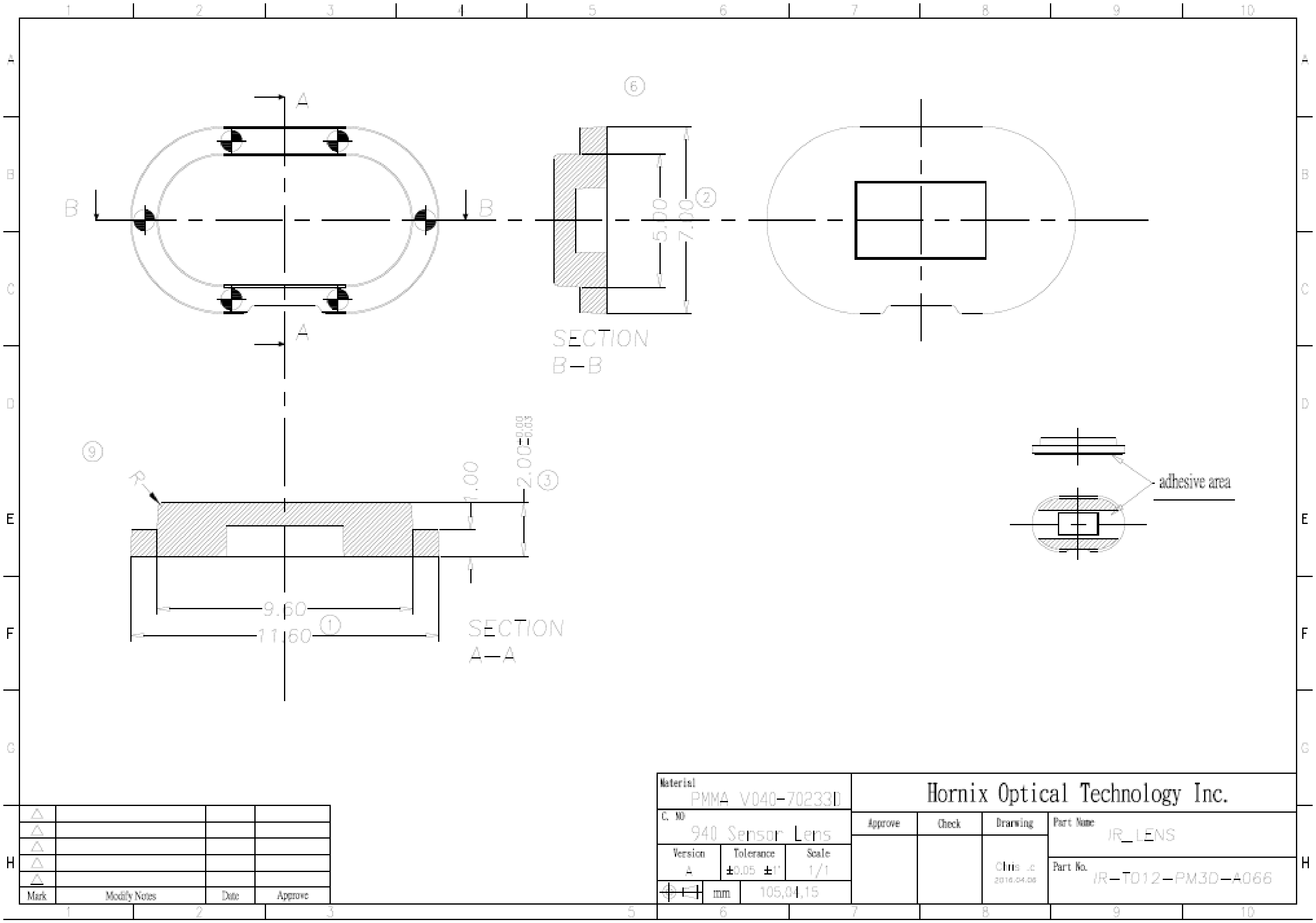
Apr 2017

HORNIX PMMA reference windows (1)

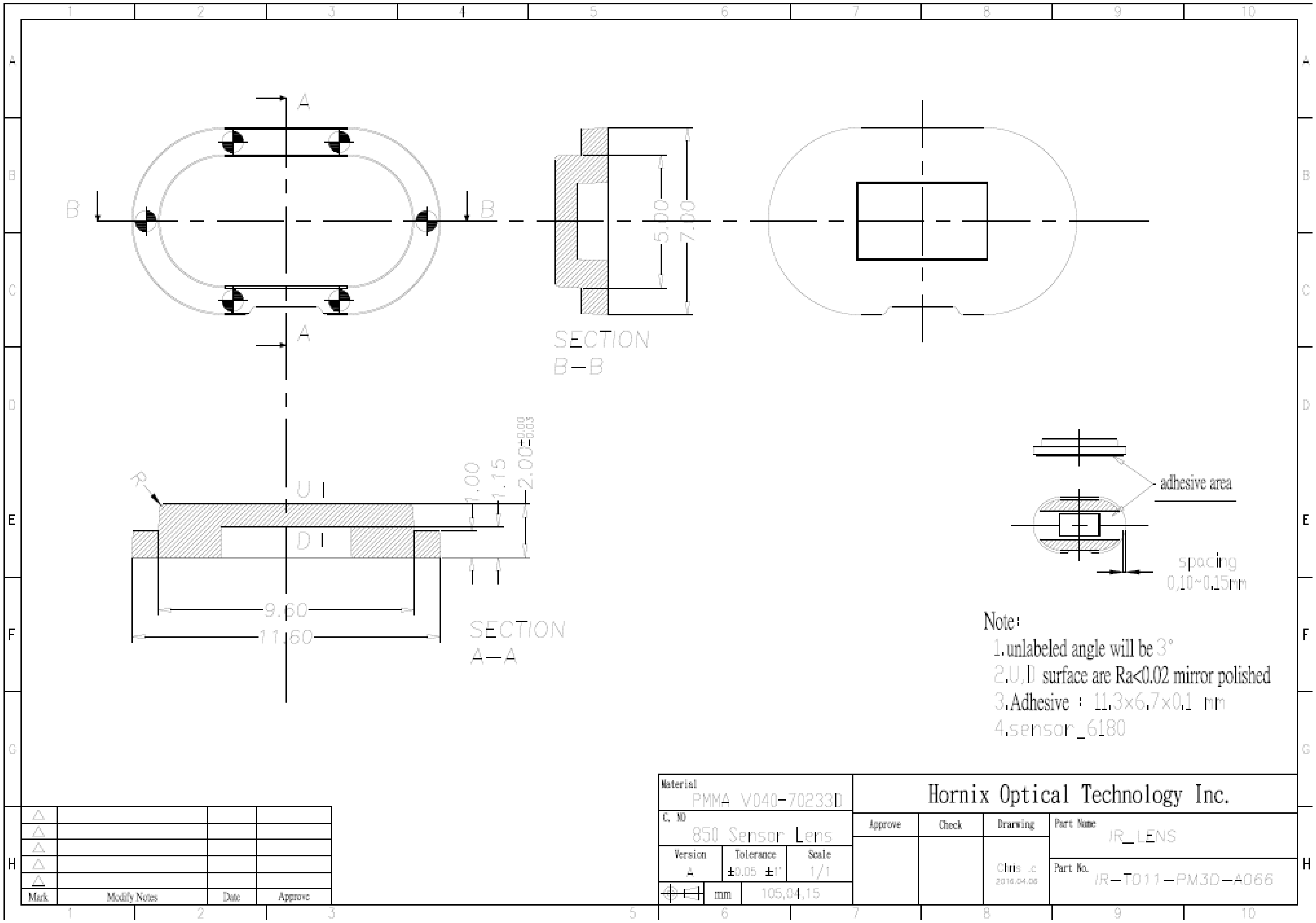
	VL53L0X	VL6180X
Reference	**IR-T012-PM3D-A066**	**IR-T011-PM3D-A066**
Material	PMMA	PMMA
Window Invisibility method	Embedded	Embedded
Hardness	1H~3H	1H~3H
AFC	N/A	N/A
ARC	Not required for Hornix solution. Transmission rate>90%	Not required for Hornix solution. Transmission rate>90%
Roughness Rq	TBC	TBC
Thickness	0.85mm	0.85mm
Airgap	0.15mm	0.15mm
Haze	<6%	<6%
xtalk	0.1~0.3kcps	<0.2mcps
Temperature use	0-80degree C	0-80degree C
Ready for order	1-Jul	1-Jul
Drawing		
Hornix Contact	Sales contact: pmcontact@hornix.com.tw / ray.chen@hornix.com.tw (+886-976-235-265) Technical contact: pinpin@hornix.com.tw (+886-930-842-622) 閎喬光學股份有限公司 Hornix Optical Technology Inc. 桃園市新屋區清華二街79號 TEL: +886-3-4972665 FAX: +886-3-4972695	

life.augmented

HORNIX PMMA window (2) VL53L0 CG Drawing



HORNIX PMMA window (3) VL6180 CG Drawing



Note:
1.unlabeled angle will be 3°
2.U,D surface are Ra<0.02 mirror polished
3.Adhesive : 11.3x6.7x0.1 mm
4.sensor_6180

Material PMMA V040-70233D	Hornix Optical Technology Inc.			
C. NO 850 Sensor Lens	Approve	Check	Drawing	Part Name IR_LENS
Version A | Tolerance ±0.05 ±1° | Scale 1/1			Chris .c 2016.04.08	Part No. IR—T011—PM3D—A066
mm | 105.04.15				