



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Platform

All API settings that are platform-dependent must be adapted to the platform on which API is compiled/running.

This is done in **VL53L0X_platform.h** file. Platform settings are described in the **VL53L0X Platform Functions** module.

1. PAL device type definition

User must provide **VL53L0X_Dev_t** type (in **VL53L0X_platform.h** file) as all API functions and macros rely on **VL53L0X_Dev_t dev** (given as first argument). This **dev** object does the link between API and platform abstraction layer and is passed from function to function down to final platform abstraction layer that handles final access to the device :

```
int VL53L0X_xxxx(VL53L0X_Dev_t dev, ... )
```

In single device case, **dev** can be as simple as an integer being the i2c device address

For more elaborated platform, **dev** can be a pointer to a structure containing all necessary items for the platform.

2. Read & Write access

API low-level functions rely on a few set of read & write functions which perform the access to the device. These functions must be implemented with respect to the platform on which API is compiled and running. Internal PAL register access functions should be used :

- `VL53L0X_WriteMulti()`
- `VL53L0X_ReadMulti()`
- `VL53L0X_WrByte()`
- `VL53L0X_WrWord()`
- `VL53L0X_WrDWord()`
- `VL53L0X_UpdateByte()`
- `VL53L0X_RdByte()`
- `VL53L0X_RdWord()`
- `VL53L0X_RdDWord()`

3. Data Types declaration

API functions rely on data types which are defined in **VL53L0X_types.h** file (under **platform/template** directory). This file may require user attention and porting in case of warning messages.

4. Delay for polling operations

API polling high level functions do call the function `VL53L0X_PollingDelay()` inside their while loop. A default implementation of the `VL53L0X_PollingDelay()` function is provided. You may decide to change and implement your own `VL53L0X_PollingDelay()` function.

5. API logging

All API functions entry and leave can be logged to help debugging issues. By default logging is disabled please define VL53L0X_LOG_ENABLE at compilation level. If logging is enabled, a small set of macros must be implemented to adapt logging operation to the platform : `_LOG_FUNCTION_START`, `_LOG_FUNCTION_END` and `_LOG_FUNCTION_END_FMT`

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

RangeStatus

The Range Status is contained in the [**VL53L0X_RangingMeasurementData_t**](#) and give the quality of the latest ranging.

This is a 8 bit data which contains the following fields:

Value 0 = Range Valid

This value indicate that the ranging is valid.

Value 1 = Sigma Fail

This value indicate that the sigma limit check has failed. Use the function [**VL53L0X_SetLimitCheckEnable\(\)**](#) and [**VL53L0X_SetLimitCheckValue\(\)**](#) to manage the limit.

Value 2 = Signal Fail

This value indicate that the signal check has failed. This can happens when there is no target or when the Range Ignore threshold check has failed. Use the function [**VL53L0X_SetLimitCheckEnable\(\)**](#) and [**VL53L0X_SetLimitCheckValue\(\)**](#) to manage the limit.

Value 3 = Min Range Fail

This value indicate that the min range check has failed. Use the function [**VL53L0X_SetLimitCheckEnable\(\)**](#) and [**VL53L0X_SetLimitCheckValue\(\)**](#) to manage the limit.

Value 4 = Phase Fail

This value indicate that the Phase check has failed.

Value 5 = HardWare Fail

This value indicate that the Hardware check has failed.

Value 255 = None

No Update

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

[Main Page](#)

[Related Pages](#)

[Modules](#)

[Data Structures](#)

[Files](#)

Strings

The API uses character strings to inform the user about the state of the API, the meaning of the error or about the name of a particular mode.

1. String can be removed

At compilation stage a DEFINE can be used to remove all the strings to save some space on device. Strings will be replaced with empty string. The Define to be used is USE_EMPTY_STRING:

- if USE_EMPTY_STRING is defined: all the strings are replaced with empty string.
- if USE_EMPTY_STRING is NOT defined: all the strings are well defined and not empty.

2. Max Length String

The API uses the macro VL53L0X_COPYSTRING to copy strings. For example the following code from get device info

```
VL53L0X_COPYSTRING(pVL53L0X_DeviceInfo->Type,  
VL53L0X_STRING_DEVICE_INFO_TYPE);
```

This MACRO is defined inside platform code. This means that is the responsibility of the customer to use the right function to copy the string. In the Platform gives as example this is:

```
#define VL53L0X_COPYSTRING(str, ...) strcpy(s  
tr, ##__VA_ARGS__)
```

In previous example we copy the string defined in VL53L0X_STRING_DEVICE_INFO_TYPE in a field in a structure pVL53L0X_DeviceInfo->Type. This is defined with a max lenght:

```
char Type[VL53L0X_MAX_STRING_LENGTH];
```

In that case by construction the Define:

```
len(VL53L0X_STRING_DEVICE_INFO_TYPE) < VL53L0  
X_MAX_STRING_LENGTH.
```

In the API the max lenght is defined in the VL53L0X_api_def.h as follow:

```
#define VL53L0X_MAX_STRING_LENGTH 32
```

In the API there are some functions which output directly the string like the following:

```
VL53L0X_Error VL53L0X_GetRangeStatusString(uint8_t  
RangeStatus,
```

```
    char *pRangeStatusString)
```

Even in that case a copy string is done. To avoid overflow problem when the copy is done, the string which will contains the one is copied, should be greater or equal to the max lenght described before.

```
void
    print_range_status(VL53L0X_RangingMeasurementD-
        ata_t* pRangingMeasurementData){
    char buf[VL53L0X_MAX_STRING_LENGTH];
    uint8_t RangeStatus;

        RangeStatus = pRangingMeasurementData-
    >RangeStatus;

    VL53L0X_GetRangeStatusString(RangeStatus, buf);
        printf("Range Status: %i : %s\n",
    RangeStatus, buf);
}
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

[Main Page](#)

[Related Pages](#)

[Modules](#)

[Data Structures](#)

[Files](#)

Disclaimer

Copyright (C) 2015 STMicroelectronics Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

A copy of the license is included in the section entitled "GNU Free Documentation License".

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

Modules

Here is a list of all modules:

[detail level [1](#) [2](#)]

<p>▼ VL53L0X Platform Functions</p> <p>PAL Register Access Functions</p> <p>Basic type definition</p>	<p>VL53L0X Platform Functions</p> <p>PAL Register Access Functions</p> <p>File vl53l0x_types.h files hold basic type definition that may requires porting</p>
<p>▼ VL53L0X cut1.1 Function Definition</p> <p>VL53L0X General Functions</p>	<p>VL53L0X cut1.1 Function Definition</p> <p>General functions and definitions</p>
<p>VL53L0X Init Functions</p>	<p>VL53L0X Init Functions</p>
<p>VL53L0X Parameters Functions</p>	<p>Functions used to prepare and setup the device</p>
<p>VL53L0X Measurement Functions</p>	<p>Functions used for the measurements</p>
<p>VL53L0X Interrupt Functions</p>	<p>Functions used for interrupt managements</p>

VL53L0X SPAD Functions	Functions used for SPAD managements
▼ VL53L0X Defines	VL53L0X Defines
Error and Warning code returned by API	The following DEFINE are used to identify the PAL ERROR
Defines Device modes	Defines all possible modes for the device
Defines Histogram modes	Defines all possible Histogram modes for the device
List of available Power Modes	List of available Power Modes
Defines the current status of the device	Defines the current status of the device
Defines the Polarity	Of the Interrupt Defines the Polarity of the Interrupt
Vcsel Period Defines	Defines the range measurement for which to access the vcsel period
Defines the steps	Carried out by the scheduler during a range measurement
Defines the Polarity	Of the Interrupt Defines the the sequence steps performed during ranging
General Macro Defines	General Macro Defines
▼ VL53L0X cut1.1 Device Specific Defines	Device specific defines
Device Error	Device Error code

Check Enable list
Gpio Functionality

Define Registers

Check Enable code
Defines the different functionalities for the device GPIO(s)
List of all the defined registers

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[Modules](#) | [Data Structures](#) | [Macros](#) | [Typedefs](#) |
[Functions](#)

VL53L0X Platform Functions

VL53L0X Platform Functions. [More...](#)

Modules

PAL Register Access Functions

PAL Register Access Functions.

Basic type definition

file `vl53l0x_types.h` files hold basic type definition that may requires porting

Data Structures

struct **VL53L0X_Dev_t**

Generic PAL device type that does link between API and platform abstraction layer. [More...](#)

Macros

```
#define PALDevDataGet(Dev, field)  (Dev->Data.field)  
Get ST private structure VL53L0X_DevData_t data access.  
More...
```

```
#define PALDevDataSet(Dev, field, data)  (Dev->Data.field)=(data)  
Set ST private structure VL53L0X_DevData_t data field.  
More...
```

Typedefs

`typedef VL53L0X_Dev_t * VL53L0X_DEV`
Declare the device Handle as a pointer
of the structure `VL53L0X_Dev_t`. More...

Functions

VL53L0X_Error VL53L0X_PollingDelay (VL53L0X_DEV Dev)

execute delay in all polling API call More...

Detailed Description

VL53L0X Platform Functions.

Macro Definition Documentation

```
#define PALDevDataGet( Dev,  
                      field  
                    )  (Dev->Data.field)
```

Get ST private structure **VL53L0X_DevData_t** data access.

Parameters

Dev Device Handle

field ST structure field name It maybe used and as real data "ref" not just as "get" for sub-structure item like
PALDevDataGet(FilterData.field)[i] or
PALDevDataGet(FilterData.MeasurementIndex)++

Definition at line **84** of file **vl53l0x_platform.h**.

```
#define PALDevDataSet( Dev,  
                      field,  
                      data  
                    )  (Dev->Data.field)=(data)
```

Set ST private structure **VL53L0X_DevData_t** data field.

Parameters

Dev Device Handle

field ST structure field name

data Data to be set

Definition at line **93** of file **vl53l0x_platform.h**.

Typedef Documentation

```
typedef VL53L0X_Dev_t* VL53L0X_DEV
```

Declare the device Handle as a pointer of the structure
VL53L0X_Dev_t.

Definition at line **73** of file **vl53l0x_platform.h**.

Function Documentation

VL53L0X_Error VL53L0X_PollingDelay (VL53L0X_DEV Dev)

execute delay in all polling API call

A typical multi-thread or RTOs implementation is to sleep the task for some 5ms (with 100Hz max rate faster polling is not needed) if nothing specific is needed you can define it as an empty/void macro

```
1 | #define VL53L0X_PollingDelay( . . . ) (void)0
```

Parameters

Dev Device Handle

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Functions

PAL Register Access Functions

[VL53L0X Platform Functions](#)

PAL Register Access Functions. [More...](#)

Functions

VL53L0X_Error	VL53L0X_LockSequenceAccess (VL53L0X_DEV Dev) Lock comms interface to serialize all commands to a shared I2C interface for a specific device. More...
VL53L0X_Error	VL53L0X_UnlockSequenceAccess (VL53L0X_DEV Dev) Unlock comms interface to serialize all commands to a shared I2C interface for a specific device. More...
VL53L0X_Error	VL53L0X_WriteMulti (VL53L0X_DEV Dev, uint8_t index, uint8_t *pdata, uint32_t count) Writes the supplied byte buffer to the device. More...
VL53L0X_Error	VL53L0X_ReadMulti (VL53L0X_DEV Dev, uint8_t index, uint8_t *pdata, uint32_t count) Reads the requested number of bytes from the device. More...
VL53L0X_Error	VL53L0X_WrByte (VL53L0X_DEV Dev, uint8_t index, uint8_t data) Write single byte register. More...
VL53L0X_Error	VL53L0X_WrWord (VL53L0X_DEV Dev, uint8_t index, uint16_t data) Write word register. More...
VL53L0X_Error	VL53L0X_WrDWord (VL53L0X_DEV Dev, uint8_t index, uint32_t data) Write double word (4 byte) register. More...
VL53L0X_Error	VL53L0X_RdByte (VL53L0X_DEV Dev, uint8_t

index, `uint8_t` *data)
Read single byte register. [More...](#)

`VL53L0X_Error` [`VL53L0X_RdWord` \(`VL53L0X_DEV` Dev, `uint8_t` index, `uint16_t` *data\)](#)
Read word (2byte) register. [More...](#)

`VL53L0X_Error` [`VL53L0X_RdDWord` \(`VL53L0X_DEV` Dev, `uint8_t` index, `uint32_t` *data\)](#)
Read dword (4byte) register. [More...](#)

`VL53L0X_Error` [`VL53L0X_UpdateByte` \(`VL53L0X_DEV` Dev, `uint8_t` index, `uint8_t` AndData, `uint8_t` OrData\)](#)
Thread safe Update (read/modify/write) single byte register. [More...](#)

Detailed Description

PAL Register Access Functions.

Function Documentation

VL53L0X_Error

VL53L0X_LockSequenceAccess

(**VL53L0X_DEV Dev**)

Lock comms interface to serialize all commands to a shared I2C interface for a specific device.

Parameters

Dev Device Handle

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

VL53L0X_Error

VL53L0X_UnlockSequenceAccess

(**VL53L0X_DEV Dev**)

Unlock comms interface to serialize all commands to a shared I2C interface for a specific device.

Parameters

Dev Device Handle

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

VL53L0X_Error **VL53L0X_WriteMulti** (**VL53L0X_DEV Dev,**

uint8_t index,

uint8_t * pdata,

uint32_t count

```
)
```

Writes the supplied byte buffer to the device.

Parameters

Dev Device Handle
index The register index
pdata Pointer to `uint8_t` buffer containing the data to be written
count Number of bytes in the supplied byte buffer

Returns

`VL53L0X_ERROR_NONE` Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_Error VL53L0X_ReadMulti( VL53L0X_DEV Dev,  
                                  uint8_t      index,  
                                  uint8_t *    pdata,  
                                  uint32_t     count  
)
```

Reads the requested number of bytes from the device.

Parameters

Dev Device Handle
index The register index
pdata Pointer to the `uint8_t` buffer to store read data
count Number of `uint8_t`'s to read

Returns

`VL53L0X_ERROR_NONE` Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_Error VL53L0X_WrByte( VL53L0X_DEV Dev,  
                               uint8_t      index,
```

```
    uint8_t data  
)  
)
```

Write single byte register.

Parameters

Dev Device Handle
index The register index
data 8 bit register data

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X Error](#)

```
VL53L0X_Error VL53L0X_WrWord( VL53L0X_DEV Dev,  
                                uint8_t index,  
                                uint16_t data  
                            )
```

Write word register.

Parameters

Dev Device Handle
index The register index
data 16 bit register data

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X Error](#)

```
VL53L0X_Error VL53L0X_WrDWord( VL53L0X_DEV Dev,  
                                uint8_t index,  
                                uint32_t data  
                                )
```

Write double word (4 byte) register.

Parameters

Dev Device Handle
index The register index
data 32 bit register data

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_Error VL53L0X_RdByte ( VL53L0X_DEV Dev,
                                  uint8_t      index,
                                  uint8_t *    data
                                )
```

Read single byte register.

Parameters

Dev Device Handle
index The register index
data pointer to 8 bit data

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_Error VL53L0X_RdWord ( VL53L0X_DEV Dev,
                                 uint8_t      index,
                                 uint16_t *   data
                               )
```

Read word (2byte) register.

Parameters

Dev Device Handle
index The register index
data pointer to 16 bit data

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_Error VL53L0X_RdDWord ( VL53L0X_DEV Dev,
                                    uint8_t      index,
                                    uint32_t *   data
                                )
```

Read dword (4byte) register.

Parameters

Dev Device Handle
index The register index
data pointer to 32 bit data

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_Error VL53L0X_UpdateByte ( VL53L0X_DEV Dev,
                                       uint8_t      index,
                                       uint8_t      AndData,
                                       uint8_t      OrData
                                   )
```

Thread safe Update (read/modify/write) single byte register.

Final_reg = (Initial_reg & and_data) |or_data

Parameters

Dev Device Handle
index The register index
AndData 8 bit and data
OrData 8 bit or data

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

Basic type definition

VL53L0X Platform Functions

file **vl53l0x_types.h** files hold basic type definition that may requires porting More...

file **vl53l0x_types.h** files hold basic type definition that may requires porting

contains type that must be defined for the platform
when target platform and compiler provide stdint.h and stddef.h it is enough to include it.

If stdint.h is not available review and adapt all signed and unsigned 8/16/32 bits basic types.

If stddef.h is not available review and adapt NULL definition .

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_Dev_t Struct Reference

[VL53L0X Platform Functions](#)

Generic PAL device type that does link between API and platform abstraction layer. [More...](#)

```
#include <vl53l0x_platform.h>
```

Data Fields

`VL53L0X_DevData_t Data`

`uint8_t I2cDevAddr`

`uint8_t comms_type`

`uint16_t comms_speed_khz`

Detailed Description

Generic PAL device type that does link between API and platform abstraction layer.

Definition at line **58** of file [vl53l0x_platform.h](#).

Field Documentation

VL53L0X_DevData_t VL53L0X_Dev_t::Data

embed ST Ewok Dev data as "Data" user specific field

Definition at line **59** of file [vl53l0x_platform.h](#).

uint8_t VL53L0X_Dev_t::I2cDevAddr

i2c device address user specific field

Definition at line **62** of file [vl53l0x_platform.h](#).

uint8_t VL53L0X_Dev_t::comms_type

Type of comms : VL53L0X_COMMS_I2C or VL53L0X_COMMS_SPI

Definition at line **63** of file [vl53l0x_platform.h](#).

uint16_t VL53L0X_Dev_t::comms_speed_khz

Comms speed [kHz] : typically 400kHz for I2C

Definition at line **64** of file [vl53l0x_platform.h](#).

The documentation for this struct was generated from the following file:

- [vl53l0x_platform.h](#)
-

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Modules

VL53L0X cut1.1 Function Definition

VL53L0X cut1.1 Function Definition. [More...](#)

Modules

VL53L0X General Functions

General functions and definitions.

VL53L0X Init Functions

VL53L0X Init Functions.

VL53L0X Parameters Functions

Functions used to prepare and setup the device.

VL53L0X Measurement Functions

Functions used for the measurements.

VL53L0X Interrupt Functions

Functions used for interrupt managements.

VL53L0X SPAD Functions

Functions used for SPAD managements.

Detailed Description

VL53L0X cut1.1 Function Definition.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Functions

VL53L0X General Functions

[VL53L0X cut1.1 Function Definition](#)

General functions and definitions. More...

Functions

VL53L0X_API VL53L0X_Error	VL53L0X_GetVersion (VL53L0X_Version_t *pVersion) Return the VL53L0X PAL Implementation Version. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetPalSpecVersion (VL53L0X_Version_t *pPalSpecVersion) Return the PAL Specification Version used by the current implementation. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetProductRevision (VL53L0X_Device_t *pProductRevisionMajor, *pProductRevisionMinor) Reads the Product Revision for a given Device. This function can be used to distinguish cut1.0 from cut1.1. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetDeviceInfo (VL53L0X_Device_t *pVL53L0X_Device) Reads the Device information for given device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetDeviceErrorStatus (VL53L0X_DEV Dev, VL53L0X_Device_t *pDeviceErrorStatus) Read current status of the error register for selected device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetRangeStatusString (uint8_t RangeStatus, char *pRangeStatusString) Human readable Range Status string for RangeStatus. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetDeviceErrorString (VL53L0X_DeviceError ErrorCode, char *pDeviceErrorString)

*pDeviceErrorString)

Human readable error string for a given Device Code. [More...](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetPalErrorString (VL53L0X_Error)

PalErrorCode, char *pPalErrorString)

Human readable error string for current error status. [More...](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetPalStateString (VL53L0X_Error)

PalStateCode, char *pPalStateString)

Human readable PAL State string. [More...](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetPalState (VL53L0X_Error)

VL53L0X_State *pPalState)

Reads the internal state of the PAL for a Device. [More...](#)

VL53L0X_API VL53L0X_Error

VL53L0X_SetPowerMode (VL53L0X_Error)

VL53L0X_PowerModes PowerMode)

Set the power mode for a given Device. Device power mode can be Standby or Idle. [More...](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetPowerMode (VL53L0X_Error)

VL53L0X_PowerModes *pPowerMode)

Get the power mode for a given Device. [More...](#)

VL53L0X_API VL53L0X_Error

VL53L0X_SetOffsetCalibrationData (VL53L0X_Error)

(VL53L0X_DEV Dev, int32_t

OffsetCalibrationDataMicroMeter)

Set or over-hide part to part calibration offset. [More...](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetOffsetCalibrationData (VL53L0X_Error)

(VL53L0X_DEV Dev, int32_t

*pOffsetCalibrationDataMicroMeter)

Get part to part calibration offset. [More...](#)

VL53L0X_API	VL53L0X_Error	VL53L0X_SetLinearityCorrectiveGain (VL53L0X_DEV Dev, int16_t LinearityCorrectiveGain) Set the linearity corrective gain. More.
VL53L0X_API	VL53L0X_Error	VL53L0X_GetLinearityCorrectiveGain (VL53L0X_DEV Dev, uint16_t *pLinearityCorrectiveGain) Get the linearity corrective gain. More.
VL53L0X_API	VL53L0X_Error	VL53L0X_SetGroupParamHold (VL53L0X_DEV Dev, uint8_t GroupParamHold) Set Group parameter Hold state. More
VL53L0X_API	VL53L0X_Error	VL53L0X_GetUpperLimitMilliMeter (VL53L0X_DEV Dev, uint16_t *pUpperLimitMilliMeter) Get the maximal distance for actual sensor. More...
	VL53L0X_Error	VL53L0X_GetTotalSignalRate (VL53L0X_DEV Dev, FixPoint1616_t *pTotalSignalRate) Get the Total Signal Rate. More...

Detailed Description

General functions and definitions.

Function Documentation

**VL53L0X_API VL53L0X_Error
VL53L0X_GetVersion (VL53L0X_Version_t * pVersion)**

Return the VL53L0X PAL Implementation Version.

Note

This function doesn't access to the device

Parameters

pVersion Pointer to current PAL Implementation Version

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

**VL53L0X_API VL53L0X_Error
VL53L0X_GetPalSpecVersion (VL53L0X_Version_t * pPalSpecVers**

Return the PAL Specification Version used for the current implementation

Note

This function doesn't access to the device

Parameters

pPalSpecVersion Pointer to current PAL Specification Version

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetProductRevision( VL53L0X_DEV Dev,
                            uint8_t * pProductRevisionMajor,
                            uint8_t * pProductRevisionMinor
                          )
```

Reads the Product Revision for a given Device This function can be used to distinguish cut1.0 from cut1.1.

Note

This function Access to the device

Parameters

Dev	Device Handle
pProductRevisionMajor	Pointer to Product Revision Major for a given Device
pProductRevisionMinor	Pointer to Product Revision Minor for a given Device

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API
VL53L0X_Error
VL53L0X_GetDeviceInfo( VL53L0X_DEV Dev,
                       VL53L0X_DeviceInfo_t * pVL53L0X_DeviceInfo
                     )
```

Reads the Device information for given Device.

Note

This function Access to the device

Parameters

Dev	Device Handle
pVL53L0X_DeviceInfo	Pointer to current device info for a given Device

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_GetDeviceErrorStatus ( VL53L0X_DEV           Dev,
                               VL53L0X_DeviceError * pDevice
                           )
```

Read current status of the error register for the selected device.

Note

This function Access to the device

Parameters

Dev Device Handle
pDeviceErrorStatus Pointer to current error code of the device

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_GetRangeStatusString ( uint8_t RangeStatus,
                               char * pRangeStatusString
                           )
```

Human readable Range Status string for a given RangeStatus.

Note

This function doesn't access to the device

Parameters

RangeStatus The RangeStatus code as stored on [VL53L0X_RangingMeasurementData_t](#)
pRangeStatusString The returned RangeStatus string.

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetDeviceErrorString ( VL53L0X_DeviceError ErrorCode,
                                char * pDeviceErrorString
                            )
```

Human readable error string for a given Error Code.

Note

This function doesn't access to the device

Parameters

ErrorCode The error code as stored on [VL53L0X_DeviceError](#)
pDeviceErrorString The error string corresponding to the ErrorCode

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetPalErrorString ( VL53L0X_Error PalErrorCode,
                                char * pPalErrorString
                            )
```

Human readable error string for current PAL error status.

Note

This function doesn't access to the device

Parameters

PalErrorCode The error code as stored on [VL53L0X_Error](#)
pPalErrorString The error string corresponding to the PalErrorCode

PalErrorCode

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetPalStateString ( VL53L0X_State PalStateCode,
                            char *          pPalStateString
                          )
```

Human readable PAL State string.

Note

This function doesn't access to the device

Parameters

PalStateCode The State code as stored on *VL53L0X_State*
pPalStateString The State string corresponding to the
PalStateCode

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetPalState ( VL53L0X_DEV   Dev,
                       VL53L0X_State * pPalState
                     )
```

Reads the internal state of the PAL for a given Device.

Note

This function doesn't access to the device

Parameters

Dev Device Handle

pPalState Pointer to current state of the PAL for a given Device

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API](#)

[VL53L0X_Error](#)

```
VL53L0X_SetPowerMode ( VL53L0X_DEV Dev,  
                         VL53L0X_PowerModes PowerMode  
                     )
```

Set the power mode for a given Device The power mode can be Standby or Idle.

Different level of both Standby and Idle can exists. This function should not be used when device is in Ranging state.

Note

This function Access to the device

Parameters

Dev Device Handle

PowerMode The value of the power mode to set. see

[VL53L0X_PowerModes](#) Valid values are:

VL53L0X_POWERMODE_STANDBY_LEVEL1,

VL53L0X_POWERMODE_IDLE_LEVEL1

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_MODE_NOT_SUPPORTED This error occurs when PowerMode is not in the supported list

"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API](#)

```
VL53L0X_Error VL53L0X_GetPowerMode( VL53L0X_DEV Dev,  
                                         VL53L0X_PowerModes * pPowerMode  
                                       )
```

Get the power mode for a given Device.

Note

This function Access to the device

Parameters

Dev Device Handle

pPowerMode Pointer to the current value of the power mode.
see [VL53L0X_PowerModes](#) Valid values are:
VL53L0X_POWERMODE_STANDBY_LEVEL1,
VL53L0X_POWERMODE_IDLE_LEVEL1

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error  
VL53L0X_SetOffsetCalibrationDataMicroMeter( VL53L0X_DEV Dev,  
                                              int32_t Off  
                                            )
```

Set or over-hide part to part calibration offset.

See also

[VL53L0X_Datalinit\(\)](#) [VL53L0X_GetOffsetCalibrationDataMicroM](#)

Note

This function Access to the device

Parameters

Dev Device Handle

OffsetCalibrationDataMicroMeter Offset (microns)

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetOffsetCalibrationDataMicroMeter ( VL53L0X_DEV Dev,
                                              int32_t * pOffsetCalibrationDataMicroMeter
                                            )
```

Get part to part calibration offset.

Function Description

Should only be used after a successful call to *VL53L0X_DataInit* to

Note

This function Access to the device

Parameters

Dev	Device Handle
pOffsetCalibrationDataMicroMeter	Return part to part calibratior

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_SetLinearityCorrectiveGain ( VL53L0X_DEV Dev,
                                       int16_t LinearityCorrectiveGain
                                     )
```

Set the linearity corrective gain.

Note

This function Access to the device

Parameters

Dev Device Handle
LinearityCorrectiveGain Linearity corrective gain in x1000 if value is 1000 then no modification is applied.

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_GetLinearityCorrectiveGain ( VL53L0X_DEV Dev,
                                      uint16_t * pLinearityC
                                    )
```

Get the linearity corrective gain.

Function Description

Should only be used after a successful call to *VL53L0X_DataInit* to set device NVM value

Note

This function Access to the device

Parameters

Dev Device Handle
pLinearityCorrectiveGain Pointer to the linearity corrective gain if value is 1000 then no modification is applied

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_SetGroupParamHold ( VL53L0X_DEV Dev,
                            uint8_t GroupParamHold
                          )
```

Set Group parameter Hold state.

Function Description

Set or remove device internal group parameter hold

Note

This function is not Implemented

Parameters

Dev Device Handle

GroupParamHold Group parameter Hold state to be set (on/off)

Returns

VL53L0X_ERROR_NOT_IMPLEMENTED Not implemented

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetUpperLimitMilliMeter ( VL53L0X_DEV Dev,
                                    uint16_t *      pUpperLimitMi
                                  )
```

Get the maximal distance for actual setup.

Function Description

Device must be initialized through *VL53L0X_SetParameters()* prior this function.

Any range value more than the value returned is to be considered as "no target detected" or "no target in detectable range"

Warning

The maximal distance depends on the setup

Note

This function is not Implemented

Parameters

Dev Device Handle

pUpperLimitMilliMeter The maximal range limit for actual setup (in millimeter)

Returns

VL53L0X_ERROR_NOT_IMPLEMENTED Not implemented

VL53L0X_Error

```
VL53L0X_GetTotalSignalRate ( VL53L0X_DEV Dev,
                             FixPoint1616_t * pTotalSignalRate
                           )
```

Get the Total Signal Rate.

Function Description

This function will return the Total Signal Rate after a good ranging is done.

Note

This function access to Device

Parameters

Dev Device Handle

pTotalSignalRate Total Signal Rate value in Mega count per second

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Functions

VL53L0X Init Functions

[VL53L0X cut1.1 Function Definition](#)

VL53L0X Init Functions. [More...](#)

Functions

VL53L0X_API VL53L0X_Error	VL53L0X_SetDeviceAddress (VL53L0X_DEV Dev, uint8_t DeviceAddress) Set new device address. More...
VL53L0X_API VL53L0X_Error	VL53L0X_DataInit (VL53L0X_DEV Dev) One time device initialization. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetTuningSettingBuffer (VL53L0X_DEV Dev, uint8_t *pTuningSettingBuffer, uint8_t UseInternalTuningSettings) Set the tuning settings pointer. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetTuningSettingBuffer (VL53L0X_DEV Dev, uint8_t **ppTuningSettingBuffer, uint8_t *pUseInternalTuningSettings) Get the tuning settings pointer and the internal external switch value. More...
VL53L0X_API VL53L0X_Error	VL53L0X_StaticInit (VL53L0X_DEV Dev) Do basic device init (and eventually patch loading) This function will change the VL53L0X_State from VL53L0X_STATE_WAIT_STATICINIT to VL53L0X_STATE_IDLE. More...
VL53L0X_API VL53L0X_Error	VL53L0X_WaitDeviceBooted (VL53L0X_DEV Dev) Wait for device booted after chip

enable (hardware standby) This function can be run only when VL53L0X_State is VL53L0X_STATE_POWERDOWN.
[More...](#)

VL53L0X_API VL53L0X_Error

VL53L0X_ResetDevice (VL53L0X_DEV Dev)

Do an hard reset or soft reset (depending on implementation) of the device call of this function, device must be in same state as right after a power-up sequence.This function will change the VL53L0X_State to VL53L0X_STATE_POWERDOWN.
[More...](#)

Detailed Description

VL53L0X Init Functions.

Function Documentation

```
VL53L0X_API VL53L0X_Error  
VL53L0X_SetDeviceAddress ( VL53L0X_DEV Dev,  
                           uint8_t          DeviceAddress  
                         )
```

Set new device address.

After completion the device will answer to the new address programmed. This function should be called when several devices are used in parallel before start programming the sensor. When a single device us used, there is no need to call this function.

Note

This function Access to the device

Parameters

Dev Device Handle

DeviceAddress The new Device address

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error  
VL53L0X_DataInit ( VL53L0X_DEV Dev )
```

One time device initialization.

To be called once and only once after device is brought out of reset (Chip enable) and booted see [VL53L0X_WaitDeviceBooted\(\)](#)

Function Description

When not used after a fresh device "power up" or reset, it may return **VL53L0X_ERROR_CALIBRATION_WARNING** meaning wrong calibration data may have been fetched from device that can result in ranging offset error

If application cannot execute device reset or need to run VL53L0X_DataInit multiple time then it must ensure proper offset calibration saving and restore on its own by using **VL53L0X_GetOffsetCalibrationData()** on first power up and then **VL53L0X_SetOffsetCalibrationData()** in all subsequent init This function will change the VL53L0X_State from **VL53L0X_STATE_POWERDOWN** to **VL53L0X_STATE_WAIT_STATICINIT**.

Note

This function Access to the device

Parameters

Dev Device Handle

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

VL53L0X_API VL53L0X_Error

```
VL53L0X_SetTuningSettingBuffer( VL53L0X_DEV Dev,
                                uint8_t *      pTuningSetting,
                                uint8_t          UseInternalTuni
                                )
```

Set the tuning settings pointer.

This function is used to specify the Tuning settings buffer to be used for device. The buffer contains all the necessary data to permit the API to use settings. This function permit to force the usage of either external or internal settings.

Note

This function Access to the device

Parameters

Dev	Device Handle
pTuningSettingBuffer	Pointer to tuning settings buffer.
UseInternalTuningSettings	Use internal tuning settings value.

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_GetTuningSettingBuffer ( VL53L0X_DEV Dev,
                                  uint8_t ** ppTuningSettingBuffer,
                                  uint8_t * pUseInternalTuningSettings
                                )
```

Get the tuning settings pointer and the internal external switch value.

This function is used to get the Tuning settings buffer pointer and the var switch to select either external or internal tuning settings.

Note

This function Access to the device

Parameters

Dev	Device Handle
ppTuningSettingBuffer	Pointer to tuning settings buffer.
pUseInternalTuningSettings	Pointer to store Use internal tuning settings value.

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_StaticInit ( VL53L0X_DEV Dev )
```

Do basic device init (and eventually patch loading) This function will change the VL53L0X_State from VL53L0X_STATE_WAIT_STATICINIT to VL53L0X_STATE_IDLE.

In this stage all default setting will be applied.

Note

This function Access to the device

Parameters

Dev Device Handle

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_WaitDeviceBooted

([VL53L0X_DEV Dev](#))

Wait for device booted after chip enable (hardware standby) This function can be run only when VL53L0X_State is VL53L0X_STATE_POWERDOWN.

Note

This function is not Implemented

Parameters

Dev Device Handle

Returns

VL53L0X_ERROR_NOT_IMPLEMENTED Not implemented

VL53L0X_API VL53L0X_Error

VL53L0X_ResetDevice

([VL53L0X_DEV Dev](#))

Do an hard reset or soft reset (depending on implementation) of the device call of this function, device must be in same state as right

after a power-up sequence. This function will change the VL53L0X_State to VL53L0X_STATE_POWERDOWN.

Note

This function Access to the device

Parameters

Dev Device Handle

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Functions

VL53L0X Parameters Functions

[VL53L0X cut1.1 Function Definition](#)

Functions used to prepare and setup the device. [More...](#)

Functions

VL53L0X_API VL53L0X_Error	VL53L0X_SetDeviceParameters (VL53L0X_DeviceParameters_t *pDeviceParameters) Prepare device for operation. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetDeviceParameters (VL53L0X_DeviceParameters_t *pDeviceParameters) Retrieve current device parameters. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetDeviceMode (VL53L0X_DeviceModes DeviceMode) Set a new device mode. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetDeviceMode (VL53L0X_DeviceModes *pDeviceMode) Get current new device mode. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetRangeFractionEnable (uint8_t Enable) Sets the resolution of range measurement. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetFractionEnable (VL53L0X_FractionEnable_t *pEnable) Gets the fraction enable parameter including resolution of range measurements. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetHistogramMode (VL53L0X_HistogramModes HistogramMode) Set a new Histogram mode. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetHistogramMode (VL53L0X_HistogramModes *pHistogramMode) Get current new device mode. More...

VL53L0X_API VL53L0X_Error **VL53L0X_SetMeasurementTimingBudget**(**VL53L0X_DEV** Dev, **uint32_t** MeasurementTimingBudgetMicroSeconds) Set Ranging Timing Budget in microseconds

VL53L0X_API VL53L0X_Error **VL53L0X_GetMeasurementTimingBudget**(**VL53L0X_DEV** Dev, **uint32_t** *pMeasurementTimingBudgetMicroSeconds) Get Ranging Timing Budget in microseconds

VL53L0X_API VL53L0X_Error **VL53L0X_GetVcselPulsePeriod** (**VL53L0X_VcselPeriod** VcselPeriodType, **uint32_t** *pVCSELPeriod) Gets the VCSEL pulse period. [More...](#)

VL53L0X_API VL53L0X_Error **VL53L0X_SetVcselPulsePeriod** (**VL53L0X_VcselPeriod** VcselPeriodType, **VL53L0X_VcselPeriod** VcselPeriodType, **uint32_t** VCSELPulsePeriod) Sets the VCSEL pulse period. [More...](#)

VL53L0X_API VL53L0X_Error **VL53L0X_SetSequenceStepEnable**(**VL53L0X_SequenceStepId** SequenceStepId, **uint8_t** SequenceStepEnabled) Sets the (on/off) state of a requested sequence step. [More...](#)

VL53L0X_API VL53L0X_Error **VL53L0X_GetSequenceStepEnable**(**VL53L0X_SequenceStepId** SequenceStepId, **uint8_t** *pSequenceStepEnabled) Gets the (on/off) state of a requested sequence step. [More...](#)

VL53L0X_API VL53L0X_Error **VL53L0X_GetSequenceStepEnables**(**VL53L0X_DEV** Dev, **VL53L0X_SchedulerSequence** SchedulerSequence, **uint8_t** *pSchedulerSequenceSteps) Gets the (on/off) state of all sequence steps. [More...](#)

VL53L0X_SetSequenceStepTimeout

VL53L0X_API VL53L0X_Error Dev, **VL53L0X_SequenceStepId** Seq
FixPoint1616_t TimeOutMilliSecs)
Sets the timeout of a requested sequence.

VL53L0X_API VL53L0X_Error Dev, **VL53L0X_SequenceStepId** Seq
FixPoint1616_t *pTimeOutMilliSecs)
Gets the timeout of a requested sequence.

VL53L0X_API VL53L0X_Error **VL53L0X_GetNumberOfSequenceSteps**
Dev, **uint8_t** *pNumberOfSequenceSteps
Gets number of sequence steps managed.
More...

VL53L0X_API VL53L0X_Error **VL53L0X_GetSequenceStepsInfo**
(**VL53L0X_SequenceStepId** SequenceStepId, **char** *pSequenceStepsString)
Gets the name of a given sequence step.

VL53L0X_API VL53L0X_Error **VL53L0X_SetInterMeasurementPeriod**
(**VL53L0X_DEV** Dev, **uint32_t** InterMeasurementPeriodMilliseconds)
Program continuous mode Inter-Measurement period in milliseconds. More...

VL53L0X_API VL53L0X_Error **VL53L0X_GetInterMeasurementPeriod**
(**VL53L0X_DEV** Dev, **uint32_t** *pInterMeasurementPeriodMilliseconds)
Get continuous mode Inter-Measurement period in milliseconds. More...

VL53L0X_API VL53L0X_Error **VL53L0X_SetXTalkCompensationEnable**
(**VL53L0X_DEV** Dev, **uint8_t** XTalkCompensationEnable)
Enable/Disable Cross talk compensation.

VL53L0X_API VL53L0X_Error **VL53L0X_GetXTalkCompensationEnable**
(**VL53L0X_DEV** Dev, **uint8_t** *pXTalkCompensationEnable)

	Get Cross talk compensation rate. More... VL53L0X_SetXTalkCompensationRate (VL53L0X_DEV Dev, FixPoint1616_t *pXTalkCompensationRateMegaCps) Set Cross talk compensation rate. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetXTalkCompensationRate (VL53L0X_DEV Dev, FixPoint1616_t *pXTalkCompensationRateMegaCps) Get Cross talk compensation rate. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetRefCalibration (VL53L0X_VhvSettings , uint8_t PhaseCal) Set Reference Calibration Parameters
VL53L0X_API VL53L0X_Error	VL53L0X_GetRefCalibration (VL53L0X_VhvSettings , uint8_t *pPhaseCal) Get Reference Calibration Parameters
VL53L0X_API VL53L0X_Error	VL53L0X_GetNumberOfLimitCheck (VL53L0X_Device , uint8_t *pNumberOfLimitCheck) Get the number of the check limit manager for the specified Device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetLimitCheckInfo (VL53L0X_Device , uint16_t LimitCheckId, char *pLimitCheckInfo) Return a description string for a given limit check. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetLimitCheckStatus (VL53L0X_Device , uint16_t LimitCheckId, uint8_t *pLimitCheckStatus) Return a the Status of the specified check.
VL53L0X_API VL53L0X_Error	VL53L0X_SetLimitCheckEnable (VL53L0X_Device , uint16_t LimitCheckId, uint8_t LimitCheckEnable) Enable/Disable a specific limit check. More...

VL53L0X_API VL53L0X_Error	VL53L0X_GetLimitCheckEnable (VL53L0X.h) <code>uint16_t LimitCheckId, uint8_t *pLimitCheckEnable</code> Get specific limit check enable state. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetLimitCheckValue (VL53L0X.h) <code>uint16_t LimitCheckId, FixPoint1616_t pLimitCheckValue</code> Set a specific limit check value. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetLimitCheckValue (VL53L0X.h) <code>uint16_t LimitCheckId, FixPoint1616_t *pLimitCheckValue</code> Get a specific limit check value. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetLimitCheckCurrent (VL53L0X.h) <code>uint16_t LimitCheckId, FixPoint1616_t *pLimitCheckCurrent</code> Get the current value of the signal use. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetWrapAroundCheckEnable (VL53L0X.h) <code>Dev, uint8_t WrapAroundCheckEnable</code> Enable (or disable) Wrap around Check.
VL53L0X_API VL53L0X_Error	VL53L0X_GetWrapAroundCheckEnable (VL53L0X.h) <code>Dev, uint8_t *pWrapAroundCheckEnable</code> Get setup of Wrap around Check. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetDmaxCalParameters (VL53L0X.h) <code>uint16_t RangeMilliMeter, FixPoint1616_t SignalRateRtnMegaCps</code> Set Dmax Calibration Parameters for a range. If one of the parameter is zero, this function will read the parameter from NVM. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetDmaxCalParameters (VL53L0X.h) <code>uint16_t *pRangeMilliMeter, FixPoint1616_t *pSignalRateRtnMegaCps</code>

Get Dmax Calibration Parameters for :
More...

Detailed Description

Functions used to prepare and setup the device.

Function Documentation

```
VL53L0X_API VL53L0X_Error  
VL53L0X_SetDeviceParameters ( VL53L0X_DEV  
                           const VL53L0X_DeviceParameters_t *  
                           )
```

Prepare device for operation.

Function Description

Update device with provided parameters

- Then start ranging operation.

Note

This function Access to the device

Parameters

Dev Device Handle

pDeviceParameters Pointer to store current device parameters.

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

```
VL53L0X_API VL53L0X_Error  
VL53L0X_GetDeviceParameters ( VL53L0X_DEV  
                           VL53L0X_DeviceParameters_t *  
                           )
```

Retrieve current device parameters.

Function Description

Get actual parameters of the device

- Then start ranging operation.

Note

This function Access to the device

Parameters

Dev Device Handle

pDeviceParameters Pointer to store current device parameters.

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_SetDeviceMode

```
(VL53L0X_DEV De  
VL53L0X_DeviceModes De  
)
```

Set a new device mode.

Function Description

Set device to a new mode (ranging, histogram ...)

Note

This function doesn't Access to the device

Parameters

Dev Device Handle

DeviceMode New device mode to apply Valid values are:

```
VL53L0X_DEVICE_MODE_SINGLE_RANGING  
VL53L0X_DEVICE_MODE_CONTINUOUS_RANGING  
VL53L0X_DEVICE_MODE_CONTINUOUS_TIMED_F  
VL53L0X_DEVICE_MODE_SINGLE_HISTOGRAM  
VL53L0X_HISTOGRAM_MODE_REFERENCE_ONL  
VL53L0X_HISTOGRAM_MODE_RETURN_ONLY  
VL53L0X_HISTOGRAM_MODE_BOTH
```

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_MODE_NOT_SUPPORTED This error occurs if DeviceMode is not in the supported list

VL53L0X_API VL53L0X_Error

VL53L0X_GetDeviceMode

```
( VL53L0X_DEV Dev,
  VL53L0X_DeviceModes * pD
)
```

Get current new device mode.

Function Description

Get actual mode of the device(ranging, histogram ...)

Note

This function doesn't Access to the device

Parameters

Dev Device Handle

pDeviceMode Pointer to current apply mode value Valid values are
VL53L0X_DEVICEMODE_SINGLE_RANGING
VL53L0X_DEVICEMODE_CONTINUOUS_RANGING
VL53L0X_DEVICEMODE_CONTINUOUS_TIMED
VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM
VL53L0X_HISTOGRAMMODE_REFERENCE_ON
VL53L0X_HISTOGRAMMODE_RETURN_ONLY
VL53L0X_HISTOGRAMMODE_BOTH

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_MODE_NOT_SUPPORTED This error occurs if DeviceMode is not in the supported list

VL53L0X_API VL53L0X_Error

VL53L0X_SetRangeFractionEnable

```
( VL53L0X_DEV Dev,
  uint8_t Enable
)
```

Sets the resolution of range measurements.

Function Description

Set resolution of range measurements to either 0.25mm if fraction enabled or 1mm if not enabled.

Note

This function Accesses the device

Parameters

Dev Device Handle

Enable Enable high resolution

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetFractionEnable

```
( VL53L0X_DEV Dev,  
  uint8_t * pEnable  
 )
```

Gets the fraction enable parameter indicating the resolution of range measurements.

Function Description

Gets the fraction enable state, which translates to the resolution of range measurements as follows :Enabled:=0.25mm resolution, Not Enabled:=1mm resolution.

Note

This function Accesses the device

Parameters

Dev Device Handle

pEnable Output Parameter reporting the fraction enable state.

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API](#)

[VL53L0X_Error](#)

VL53L0X_SetHistogramMode (VL53L0X_DEV Dev,
VL53L0X_HistogramModes HistogramMode)

Set a new Histogram mode.

Function Description

Set device to a new Histogram mode

Note

This function doesn't Access to the device

Parameters

Dev Device Handle

HistogramMode New device mode to apply Valid values are:

VL53L0X_HISTOGRAMMODE_DISABLED
VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM
VL53L0X_HISTOGRAMMODE_REFERENCE_C
VL53L0X_HISTOGRAMMODE_RETURN_ONLY
VL53L0X_HISTOGRAMMODE_BOTH

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_MODE_NOT_SUPPORTED This error occurs if the HistogramMode is not in the supported list

"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API](#) [VL53L0X_Error](#)

VL53L0X_GetHistogramMode (VL53L0X_DEV Dev,
VL53L0X_HistogramModes * pHistogramMode)

)

Get current new device mode.

Function Description

Get current Histogram mode of a Device

Note

This function doesn't Access to the device

Parameters

Dev Device Handle

pHistogramMode Pointer to current Histogram Mode value Valid values:
VL53L0X_HISTOGRAMMODE_DISABLED
VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM
VL53L0X_HISTOGRAMMODE_REFERENCE
VL53L0X_HISTOGRAMMODE_RETURN_ONL
VL53L0X_HISTOGRAMMODE_BOTH

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API VL53L0X_Error](#)

VL53L0X_SetMeasurementTimingBudgetMicroSeconds ([VL53L0X_Error](#),
[uint32_t](#))

Set Ranging Timing Budget in microseconds.

Function Description

Defines the maximum time allowed by the user to the device to run (ranging, histogram, ASL ...)

Note

This function Access to the device

Parameters

Dev

Device Handle

MeasurementTimingBudgetMicroSeconds Max measurement time in microseconds when wraparound disabled

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS This error is returned if Me
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetMeasurementTimingBudgetMicroSeconds ([VL53L0X_DEV](#),
[uint32_t](#))

Get Ranging Timing Budget in microseconds.

Function Description

Returns the programmed the maximum time allowed by the user to current mode (ranging, histogram, ASL ...)

Note

This function Access to the device

Parameters

Dev

Device Handle

pMeasurementTimingBudgetMicroSeconds Max measurement time in microseconds when wraparound disabled

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetVcselPulsePeriod ([VL53L0X_DEV](#),

[Dev](#),

```
VL53L0X_VcselPeriod VcselPeriod  
uint8_t * pVCSELPulsePeriod  
)
```

Gets the VCSEL pulse period.

Function Description

This function retrieves the VCSEL pulse period for the given period type.

Note

This function Accesses the device

Parameters

Dev	Device Handle
VcselPeriodType	VCSEL period identifier (pre-range final).
pVCSELPulsePeriod	Pointer to VCSEL period value.

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS Error VcselPeriodType parameter not supported.

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_SetVcselPulsePeriod ( VL53L0X_DEV Dev,  
                                VL53L0X_VcselPeriod VcselPeriod  
                                uint8_t VCSELPulsePeriod  
)
```

Sets the VCSEL pulse period.

Function Description

This function retrieves the VCSEL pulse period for the given period type.

Note

This function Accesses the device

Parameters

Dev Device Handle
VcselPeriodType VCSEL period identifier (pre-range|final).
VCSELPulsePeriod VCSEL period value

Returns

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_INVALID_PARAMS Error VcselPeriodType parameter not supported.
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_SetSequenceStepEnable( VL53L0X_DEV Dev
                               VL53L0X_SequenceStepId SequenceStepId
                               uint8_t StepId
                               )
```

Sets the (on/off) state of a requested sequence step.

Function Description

This function enables/disables a requested sequence step.

Note

This function Accesses the device

Parameters

Dev Device Handle
SequenceStepId Sequence step identifier.
SequenceStepEnabled Demanded state {0=Off,1=On} is enabled

Returns

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_INVALID_PARAMS Error SequenceStepId parameter not supported.
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error  
VL53L0X_GetSequenceStepEnable ( VL53L0X_DEV  
                                VL53L0X_SequenceStepId Se  
                                uint8_t *  
                                pS  
                                )
```

Gets the (on/off) state of a requested sequence step.

Function Description

This function retrieves the state of a requested sequence step, i.e.

Note

This function Accesses the device

Parameters

Dev	Device Handle
SequenceStepId	Sequence step identifier.
pSequenceStepEnabled	Out parameter reporting if the sequence {0=Off,1=On}.

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS Error SequenceStepId para

"Other error code" See **VL53L0X_Error**

```
VL53L0X_API VL53L0X_Error  
VL53L0X_GetSequenceStepEnables ( VL53L0X_DEV  
                                VL53L0X_SchedulerSequence  
                                )
```

Gets the (on/off) state of all sequence steps.

Function Description

This function retrieves the state of all sequence step in the scheduler

Note

This function Accesses the device

Parameters

Dev Device Handle
pSchedulerSequenceSteps Pointer to struct containing result.

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_SetSequenceStepTimeout (VL53L0X_DEV

D
VL53L0X_SequenceStepId S

T
FixPoint1616_t

)

Sets the timeout of a requested sequence step.

Function Description

This function sets the timeout of a requested sequence step.

Note

This function Accesses the device

Parameters

Dev Device Handle
SequenceStepId Sequence step identifier.
TimeOutMilliSecs Demanded timeout

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS Error SequenceStepId parameter not supported.

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetSequenceStepTimeout (VL53L0X_DEV

D

```
VL53L0X_SequenceStepId S  
FixPoint1616_t * p  
)
```

Gets the timeout of a requested sequence step.

Function Description

This function retrieves the timeout of a requested sequence step.

Note

This function Accesses the device

Parameters

Dev Device Handle
SequenceStepId Sequence step identifier.
pTimeOutMilliSecs Timeout value.

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS Error SequenceStepId parameter not supported.

"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
```

```
VL53L0X_GetNumberOfSequenceSteps ( VL53L0X_DEV Dev,  
                                    uint8_t * pNumber  
)
```

Gets number of sequence steps managed by the API.

Function Description

This function retrieves the number of sequence steps currently managed by the API.

Note

This function Accesses the device

Parameters

Dev Device Handle
pNumberOfSequenceSteps Out parameter reporting the number steps.

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_GetSequenceStepsInfo ( VL53L0X_SequenceStepId Seq
                                char *                pSe
                                )

```

Gets the name of a given sequence step.

Function Description

This function retrieves the name of sequence steps corresponding

Note

This function doesn't Accesses the device

Parameters

SequenceStepId Sequence step identifier.
pSequenceStepsString Pointer to Info string

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_SetInterMeasurementPeriodMilliSeconds ( VL53L0X_DEV
                                                uint32_t
                                                )

```

Program continuous mode Inter-Measurement period in milliseconds.

Function Description

When trying to set too short time return INVALID_PARAMS minimum value.

Note

This function Access to the device

Parameters

Dev

Device Handle

InterMeasurementPeriodMilliSeconds Inter-Measurement Period in milliseconds.

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetInterMeasurementPeriodMilliSeconds ( VL53L0X_DEV Dev,
                                                uint32_t * pInterMeasurementPeriodMilliSeconds )
```

Get continuous mode Inter-Measurement period in milliseconds.

Function Description

When trying to set too short time return INVALID_PARAMS minimum value.

Note

This function Access to the device

Parameters

Dev

Device Handle

pInterMeasurementPeriodMilliSeconds Pointer to programmed value.

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_SetXTalkCompensationEnable ( VL53L0X_DEV Dev,
```

```
uint8_t XTalkCompensationEnable(
```

Enable/Disable Cross talk compensation feature.

Note

This function is not Implemented. Enable/Disable Cross Talk by setting XTalk value by using [VL53L0X_SetXTalkCompensationRateMega](#)

Parameters

Dev Device Handle

XTalkCompensationEnable Cross talk compensation to be set 0=disabled or 1 = enabled

Returns

VL53L0X_ERROR_NOT_IMPLEMENTED Not implemented

```
VL53L0X_API VL53L0X_Error
```

```
VL53L0X_GetXTalkCompensationEnable( VL53L0X_DEV Dev,
```



```
                                  uint8_t * pXTalkCompensationEnable,
```



```
                                  )
```

Get Cross talk compensation rate.

Note

This function is not Implemented. Enable/Disable Cross Talk by setting XTalk value by using [VL53L0X_SetXTalkCompensationRateMega](#)

Parameters

Dev Device Handle

pXTalkCompensationEnable Pointer to the Cross talk compensation rate 0=disabled or 1 = enabled

Returns

VL53L0X_ERROR_NOT_IMPLEMENTED Not implemented

```
VL53L0X_API VL53L0X_Error VL53L0X_SetXTalkCompensationRateMegaCps ( VL53L0X_DEV Dev, FixPoint1616_t XTalkCompensationRateMegaCps )
```

Set Cross talk compensation rate.

Function Description

Set Cross talk compensation rate.

Note

This function Access to the device

Parameters

Dev

Device Handle

XTalkCompensationRateMegaCps Compensation rate in Mega cps
see datasheet for details

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error VL53L0X_GetXTalkCompensationRateMegaCps ( VL53L0X_DEV Dev, FixPoint1616_t * XTalkCompensationRateMegaCps )
```

Get Cross talk compensation rate.

Function Description

Get Cross talk compensation rate.

Note

This function Access to the device

Parameters

Dev

Device Handle

pXTalkCompensationRateMegaCps Pointer to Compensation rate (fixed point) see datasheet for conversion

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_SetRefCalibration

```
( VL53L0X_DEV Dev,  
  uint8_t        VhvSettings,  
  uint8_t        PhaseCal  
)
```

Set Reference Calibration Parameters.

Function Description

Set Reference Calibration Parameters.

Note

This function Access to the device

Parameters

Dev Device Handle

VhvSettings Parameter for VHV

PhaseCal Parameter for PhaseCal

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetRefCalibration

```
( VL53L0X_DEV Dev,  
  uint8_t *      pVhvSettings,  
  uint8_t *      pPhaseCal  
)
```

Get Reference Calibration Parameters.

Function Description

Get Reference Calibration Parameters.

Note

This function Access to the device

Parameters

Dev Device Handle

pVhvSettings Pointer to VHV parameter

pPhaseCal Pointer to PhaseCal Parameter

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

VL53L0X_GetNumberOfLimitCheck (uint16_t * pNumberOfLimitC

Get the number of the check limit managed by a given Device.

Function Description

This function give the number of the check limit managed by the Device.

Note

This function doesn't Access to the device

Parameters

pNumberOfLimitCheck Pointer to the number of check limit.

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API

VL53L0X_Error

```
VL53L0X_GetLimitCheckInfo ( VL53L0X_DEV Dev,
                            uint16_t      LimitCheckId,
                            char *        pLimitCheckString
                          )
```

Return a description string for a given limit check number.

Function Description

This function returns a description string for a given limit check number. The limit check is identified with the LimitCheckId.

Note

This function doesn't Access to the device

Parameters

Dev

Device Handle

LimitCheckId

Limit Check ID (0<= LimitCheckId <
[VL53L0X_GetNumberOfLimitCheck\(\)](#)).

pLimitCheckString

Pointer to the description string of the
given check limit.

Returns

[VL53L0X_ERROR_NONE](#) Success

[VL53L0X_ERROR_INVALID_PARAMS](#) This error is returned
when LimitCheckId value is out of range.

"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
```

```
VL53L0X_GetLimitCheckStatus ( VL53L0X_DEV Dev,
                               uint16_t      LimitCheckId,
                               uint8_t *     pLimitCheckStatus
                             )
```

Return a the Status of the specified check limit.

Function Description

This function returns the Status of the specified check limit. The value indicate if the check is fail or not. The limit check is identified with the LimitCheckId.

Note

This function doesn't Access to the device

Parameters

Dev	Device Handle
LimitCheckId	Limit Check ID ($0 \leq \text{LimitCheckId} < \text{VL53L0X_GetNumberOfLimitCheck()}$).
pLimitCheckStatus	Pointer to the Limit Check Status of the given check limit. LimitCheckStatus : 0 the check is not fail 1 the check if fail or not enabled

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS This error is returned when LimitCheckId value is out of range.

"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API VL53L0X_Error](#)

```
VL53L0X_SetLimitCheckEnable( VL53L0X_DEV Dev,
                               uint16_t      LimitCheckId,
                               uint8_t       LimitCheckEnable
                             )
```

Enable/Disable a specific limit check.

Function Description

This function Enable/Disable a specific limit check. The limit check is identified with the LimitCheckId.

Note

This function doesn't Access to the device

Parameters

Dev	Device Handle
------------	---------------

LimitCheckId Limit Check ID (0<= LimitCheckId < **VL53L0X_GetNumberOfLimitCheck()**).

LimitCheckEnable if 1 the check limit corresponding to LimitCheckId is Enabled if 0 the check limit corresponding to LimitCheckId is disabled

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS This error is returned when LimitCheckId value is out of range.

"Other error code" See **VL53L0X_Error**

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetLimitCheckEnable ( VL53L0X_DEV Dev,
                               uint16_t      LimitCheckId,
                               uint8_t *     pLimitCheckEnable
                             )
```

Get specific limit check enable state.

Function Description

This function get the enable state of a specific limit check. The limit check is identified with the LimitCheckId.

Note

This function Access to the device

Parameters

Dev Device Handle

LimitCheckId Limit Check ID (0<= LimitCheckId < **VL53L0X_GetNumberOfLimitCheck()**).

pLimitCheckEnable Pointer to the check limit enable value. if 1 the check limit corresponding to LimitCheckId is Enabled if 0 the check limit corresponding to LimitCheckId is disabled

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS This error is returned when the LimitCheckId value is out of range.

"Other error code" See [VL53L0X_Error](#)

VL53L0X API VL53L0X Error

```
VL53L0X_SetLimitCheckValue ( VL53L0X_DEV Dev,  
                           uint16_t     LimitCheckId,  
                           FixPoint1616_t LimitCheckValue  
                         )
```

Set a specific limit check value.

Function Description

This function set a specific limit check value. The limit check is identified with the LimitCheckId.

Note

This function Access to the device

Parameters

Dev Device Handle

LimitCheckId Limit Check ID (0<= LimitCheckId < **VL53L0X_GetNumberOfLimitChecks**)

LimitCheckValue Limit check Value for a given LimitCheckId

Returns

VL53L0X ERROR NONE Success

VL53L0X_ERROR_INVALID_PARAMS This error is returned when either LimitCheckId or LimitCheckValue value is out of range.

"Other error code" See [VL53L0X Error](#)

VL53L0X API VL53L0X Error

VL53L0X_GetLimitCheckValue (**VL53L0X_DEV** Dev,
 uint16_t LimitCheckId,

```
FixPoint1616_t * pLimitCheckValue  
 )
```

Get a specific limit check value.

Function Description

This function get a specific limit check value from device then it updates internal values and check enables. The limit check is identified with the LimitCheckId.

Note

This function Access to the device

Parameters

Dev

Device Handle

LimitCheckId

Limit Check ID ($0 \leq \text{LimitCheckId} < \text{VL53L0X_GetNumberOfLimitCheck()}$).

pLimitCheckValue

Pointer to Limit check Value for a given LimitCheckId.

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS This error is returned when LimitCheckId value is out of range.

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetLimitCheckCurrent ( VL53L0X_DEV Dev,  
                                uint16_t LimitCheckId,  
                                FixPoint1616_t * pLimitCheckCu  
 )
```

Get the current value of the signal used for the limit check.

Function Description

This function get the current value of the signal used for the limit check. To obtain the latest value you should run a ranging before.

value reported is linked to the limit check identified with the LimitCheckId.

Note

This function Access to the device

Parameters

Dev	Device Handle
LimitCheckId	Limit Check ID ($0 \leq \text{LimitCheckId} < \text{VL53L0X_GetNumberOfLimitCheck}()$).
pLimitCheckCurrent	Pointer to current Value for a given LimitCheckId

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS This error is returned when LimitCheckId value is out of range.

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_SetWrapAroundCheckEnable( VL53L0X_DEV Dev,
                                    uint8_t          WrapAroundCheckEnable )
```

Enable (or disable) Wrap around Check.

Note

This function Access to the device

Parameters

Dev	Device Handle
WrapAroundCheckEnable	Wrap around Check to be set 0=disabled 1=enabled

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_GetWrapAroundCheckEnable ( VL53L0X_DEV Dev,
                                     uint8_t * pWrapArc
                                     )
```

Get setup of Wrap around Check.

Function Description

This function get the wrapAround check enable parameters

Note

This function Access to the device

Parameters

Dev	Device Handle
pWrapAroundCheckEnable	Pointer to the Wrap around Check state 1 = enabled

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

```
VL53L0X_API VL53L0X_Error
VL53L0X_SetDmaxCalParameters ( VL53L0X_DEV Dev,
                                 uint16_t RangeMilliMeter
                                 FixPoint1616_t SignalRateRtnl
                                 )
```

Set Dmax Calibration Parameters for a given device When one of the parameters is zero, this function will get parameter from NVM.

Note

This function doesn't Access to the device

Parameters

Dev	Device Handle
RangeMilliMeter	Calibration Distance

SignalRateRtnMegaCps Signal rate return read at CalDistance

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetDmaxCalParameters ( VL53L0X_DEV Dev,
                                uint16_t * pRangeMilliMeter,
                                FixPoint1616_t * pSignalRateRtnMegaCps
                            )
```

Get Dmax Calibration Parameters for a given device.

Note

This function Access to the device

Parameters

Dev Device Handle

pRangeMilliMeter Pointer to Calibration Distance

pSignalRateRtnMegaCps Pointer to Signal rate return

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Functions

VL53L0X Measurement Functions

[VL53L0X cut1.1 Function Definition](#)

Functions used for the measurements. [More...](#)

Functions

VL53L0X_API VL53L0X_Error [**VL53L0X_PerformSingleMeasurement \(VL53L0X_DEV Dev\)**](#)
Single shot measurement. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_PerformRefCalibration \(VL53L0X_DEV Dev, uint8_t *pVhvSettings, uint8_t *pXtalkPerSpad, uint8_t *pAmbientTooHigh\)**](#)
Perform Reference Calibration. More..

VL53L0X_API VL53L0X_Error [**VL53L0X_PerformXTalkMeasurement \(VL53L0X_DEV Dev, uint32_t Timeout, FixPoint1616_t *pXTalkPerSpad, uint8_t *pAmbientTooHigh\)**](#)
Perform XTalk Measurement. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_PerformXTalkCalibration \(VL53L0X_DEV Dev, FixPoint1616_t XTalkCalDistance, FixPoint1616_t *pXTalkCompensationRateMegaCps\)**](#)
Perform XTalk Calibration. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_PerformOffsetCalibration \(VL53L0X_DEV Dev, FixPoint1616_t CalDistanceMilliMeter, int32_t *pOffset\)**](#)
Perform Offset Calibration. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_StartMeasurement \(VL53L0X_DEV Dev\)**](#)
Start device measurement. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_StopMeasurement \(VL53L0X_DEV Dev\)**](#)
Stop device measurement. More...

VL53L0X_GetMeasurementDataRea

VL53L0X_API VL53L0X_Error	VL53L0X_DEV Dev, uint8_t *pMeasurementDataReady) Return Measurement Data Ready. More...
VL53L0X_API VL53L0X_Error	VL53L0X_WaitDeviceReadyForNewCommand (VL53L0X_DEV Dev, uint32_t MaxLoopCount) Wait for device ready for a new measurement command. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetMeasurementRefSignal (VL53L0X_DEV Dev, FixPoint1616_t *pMeasurementRefSignal) Retrieve the Reference Signal after a measurement. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetRangingMeasurement (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData *pRangingMeasurementData) Retrieve the measurements from device setup. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetHistogramMeasurement (VL53L0X_DEV Dev, VL53L0X_HistogramMeasurementData *pHistogramMeasurementData) Retrieve the measurements from device setup. More...
VL53L0X_API VL53L0X_Error	VL53L0X_PerformSingleRangingMeasurement (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData *pRangingMeasurementData) Performs a single ranging measurement and retrieve the ranging measurement data.
VL53L0X_API VL53L0X_Error	VL53L0X_PerformSingleHistogramMeasurement (VL53L0X_DEV Dev,

[VL53L0X_HistogramMeasurementD](#)

(*pHistogramMeasurementData)

Performs a single histogram measurer
retrieve the histogram measurement d
equivalent to

[VL53L0X_PerformSingleMeasuremen](#)

[VL53L0X_GetHistogramMeasurement](#)

[VL53L0X_API VL53L0X_Error](#)**[VL53L0X_SetNumberOfROIZones](#)**

([VL53L0X_DEV](#) Dev, [uint8_t](#) Number)

Set the number of ROI Zones to be us
specific Device. [More...](#)

[VL53L0X_API VL53L0X_Error](#)**[VL53L0X_GetNumberOfROIZones](#)**

([VL53L0X_DEV](#) Dev, [uint8_t](#)

*pNumberOfROIZones)

Get the number of ROI Zones manage
Device. [More...](#)

[VL53L0X_API VL53L0X_Error](#)**[VL53L0X_GetMaxNumberOfROIZon](#)**

([VL53L0X_DEV](#) Dev, [uint8_t](#)

*pMaxNumberOfROIZones)

Get the Maximum number of ROI Zone
by the Device. [More...](#)

Detailed Description

Functions used for the measurements.

)

Perform Reference Calibration.

Perform a reference calibration of the Device. This function should be run from time to time before doing a ranging measurement. This function will launch a special ranging measurement, so if interrupt are enable an interrupt will be done. This function will clear the interrupt generated automatically.

Warning

This function is a blocking function

Note

This function Access to the device

Parameters

Dev Device Handle

pVhvSettings Pointer to vhv settings parameter.

pPhaseCal Pointer to PhaseCal parameter.

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_PerformXTalkMeasurement( VL53L0X_DEV      Dev,
                                    uint32_t       TimeoutMs,
                                    FixPoint1616_t * pXtalkPers,
                                    uint8_t *       pAmbientP )
```

Perform XTalk Measurement.

Measures the current cross talk from glass in front of the sensor. This function performs a histogram measurement and uses the results to measure the crosstalk. For the function to be successful, there must be no target in front of the sensor.

the sensor.

Warning

This function is a blocking function

This function is not supported when the final range vcsel clock period is below 10 PCLKS.

Note

This function Access to the device

Parameters

Dev	Device Handle
TimeoutMs	Histogram measurement duration.
pXtalkPerSpad	Output parameter containing the crosstalk measurement result, in MCPS/Spad. Format fixed 16:16.
pAmbientTooHigh	Output parameter which indicate that pXtalkPerSpad is not good if the Ambient is too high.

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_INVALID_PARAMS vcsel clock period not supported for this operation. Must not be less than 10PCLKS.

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_PerformXTalkCalibration( VL53L0X_DEV Dev,
                                    FixPoint1616_t XTalkCalDist
                                    FixPoint1616_t * pXTalkComp
                                  )
```

Perform XTalk Calibration.

Perform a XTalk calibration of the Device. This function will launch a range interrupt. If the range interrupts are enabled an interrupt will be done. This function will clear the range interrupt automatically. This function will program a new value for the XTalk compensation value. It will enable the cross talk before exit. This function will disable the range interrupt.

`VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD.`

Warning

This function is a blocking function

Note

This function Access to the device

This function change the device mode to `VL53L0X_DEVICEMODE`

Parameters

Dev

Device Handle

XTalkCalDistance

XTalkCalDistance value use computation.

pXTalkCompensationRateMegaCps Pointer to new XTalkCompe

Returns

`VL53L0X_ERROR_NONE` Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

**VL53L0X_PerformOffsetCalibration (VL53L0X_DEV Dev,
FixPoint1616_t CalDistanceM
int32_t * pOffsetMicroL
)**

Perform Offset Calibration.

Perform a Offset calibration of the Device. This function will launch a range measurement, if interrupts are enabled an interrupt will be done. This function will clear the interrupt generated automatically. This function will program the value for the Offset calibration value. This function will disable the `VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD`.

Warning

This function is a blocking function

Note

This function Access to the device

This function does not change the device mode.

Parameters

Dev	Device Handle
CalDistanceMilliMeter	Calibration distance value used for the offset compensation.
pOffsetMicroMeter	Pointer to new Offset value computed by the function.

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API VL53L0X_Error](#)

VL53L0X_StartMeasurement

([VL53L0X_DEV Dev](#))

Start device measurement.

Started measurement will depend on device parameters set through [VL53L0X_SetParameters\(\)](#) This is a non-blocking function. This function will change the VL53L0X_State from VL53L0X_STATE_IDLE to VL53L0X_STATE_RUNNING.

Note

This function Access to the device

Parameters

Dev	Device Handle
------------	---------------

Returns

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_MODE_NOT_SUPPORTED This error occurs when DeviceMode programmed with [VL53L0X_SetDeviceMode](#) is not in the supported list:
Supported mode are:
VL53L0X_DEVICEMODE_SINGLE_RANGING,
VL53L0X_DEVICEMODE_CONTINUOUS_RANGING,
VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING

VL53L0X_ERROR_TIME_OUT Time out on start measurement
"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API VL53L0X_Error](#)
[VL53L0X_StopMeasurement](#)

([VL53L0X_DEV Dev](#))

Stop device measurement.

Will set the device in standby mode at end of current measurement
Not necessary in single mode as device shall return automatically in
standby mode at end of measurement. This function will change the
VL53L0X_State from VL53L0X_STATE_RUNNING to
VL53L0X_STATE_IDLE.

Note

This function Access to the device

Parameters

Dev Device Handle

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API VL53L0X_Error](#)

[VL53L0X_GetMeasurementDataReady](#) ([VL53L0X_DEV Dev](#),
 [uint8_t * pMeasuremen](#)
)

Return Measurement Data Ready.

Function Description

This function indicate that a measurement data is ready. This function
interrupt mode is used then check is done accordingly. If perform in
the interrupt, this function will not work, like in case of
[VL53L0X_PerformSingleRangingMeasurement\(\)](#). The previous
blocking function, VL53L0X_GetMeasurementDataReady is used f

capture.

Note

This function Access to the device

Parameters

Dev Device Handle

pMeasurementDataReady Pointer to Measurement Data Ready. 0 = not ready, 1 = data ready

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X Error](#)

```
VL53L0X_Error
VL53L0X_WaitDeviceReadyForNewMeasurement (VL53L0X_DEV
                                            uint32_t
                                            )
```

Wait for device ready for a new measurement command.

Blocking function.

Note

This function is not Implemented

Parameters

Dev Device Handle

MaxLoop Max Number of polling loop (timeout).

Returns

VL53L0X ERROR NOT IMPLEMENTED Not implemented

```
VL53L0X_Error VL53L0X_GetMeasurementRefSignal ( VL53L0X_DEV Dev,  
                                                FixPoint1616_t * pMeasurer  
                                              )
```

Retrieve the Reference Signal after a measurements.

Function Description

Get Reference Signal from last successful Ranging measurement
return a valid value after that you call the
VL53L0X_GetRangingMeasurementData().

Note

This function Access to the device

Parameters

Dev Device Handle

pMeasurementRefSignal Pointer to the Ref Signal to fill up.

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

VL53L0X_API VL53L0X_Error

**VL53L0X_GetRangingMeasurementData (VL53L0X_DEV
 VL53L0X_RangingMeasu
)**

Retrieve the measurements from device for a given setup.

Function Description

Get data from last successful Ranging measurement

Warning

USER should take care about **VL53L0X_GetNumberOfROIZones**
NumberOfROIZones times the corresponding data structure used i

Note

This function Access to the device

Parameters

Dev Device Handle

pRangingMeasurementData Pointer to the data structure to fill up

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetHistogramMeasurementData( VL53L0X_DEV
                                      VL53L0X_HistogramMeas
                                      )
```

Retrieve the measurements from device for a given setup.

Function Description

Get data from last successful Histogram measurement

Warning

USER should take care about [VL53L0X_GetNumberOfROIZones](#) times the corresponding data structure used in the measurement function

Note

This function is not Implemented

Parameters

Dev Device Handle

pHistogramMeasurementData Pointer to the histogram data structure

Returns

VL53L0X_ERROR_NOT_IMPLEMENTED Not implemented

VL53L0X_API VL53L0X_Error

```
VL53L0X_PerformSingleRangingMeasurement( VL53L0X_DEV
                                            VL53L0X_Ranging
                                            )
```

Performs a single ranging measurement and retrieve the ranging meas

Function Description

This function will change the device mode to VL53L0X_DEVICEMODE. It performs measurement with VL53L0X_SetDeviceMode(), It performs measurement with VL53L0X_PerformSingleMeasurement() and last successful Ranging measurement with VL53L0X_GetRangingResult() and finally clear interrupt mask with VL53L0X_ClearInterruptMask().

Note

This function Access to the device

This function change the device mode to VL53L0X_DEVICEMODE

Parameters

Dev Device Handle

pRangingMeasurementData Pointer to the data structure to fill up

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

VL53L0X_API VL53L0X_Error

**VL53L0X_PerformSingleHistogramMeasurement (VL53L0X_DEV
VL53L0X_HistogramMeasurementData)**

Performs a single histogram measurement and retrieve the histogram result. This function is equivalent to VL53L0X_PerformSingleMeasurement + VL53L0X_GetHistogramMeasurement.

Function Description

Get data from last successful Ranging measurement. This function is equivalent to VL53L0X_PerformSingleMeasurement + VL53L0X_GetRangingResult.

Note

This function is not Implemented

Parameters

Dev Device Handle

pHistogramMeasurementData Pointer to the data structure to fill up

Returns

VL53L0X_ERROR_NOT_IMPLEMENTED Not implemented

```
VL53L0X_API VL53L0X_Error  
VL53L0X_SetNumberOfROIZones ( VL53L0X_DEV Dev,  
                                uint8_t          NumberOfROIZones  
                            )
```

Set the number of ROI Zones to be used for a specific Device.

Function Description

Set the number of ROI Zones to be used for a specific Device. The programmed value should be less than the max number of ROI Zones given with [VL53L0X_GetMaxNumberOfROIZones\(\)](#). This version of the API manages only one zone.

Parameters

Dev Device Handle
NumberOfROIZones Number of ROI Zones to be used for a specific Device.

Returns

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_INVALID_PARAMS This error is returned if NumberOfROIZones != 1

```
VL53L0X_API VL53L0X_Error  
VL53L0X_GetNumberOfROIZones ( VL53L0X_DEV Dev,  
                                uint8_t *      pNumberOfROIZones  
                            )
```

Get the number of ROI Zones managed by the Device.

Function Description

Get number of ROI Zones managed by the Device. USER should take care about [VL53L0X_GetNumberOfROIZones\(\)](#) before get data and perform measurement. PAL will fill a NumberOfROIZones times the corresponding data structure used in the measurement function.

Note

This function doesn't Access to the device

Parameters

Dev Device Handle
pNumberOfROIZones Pointer to the Number of ROI Zones value.

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_API VL53L0X_Error

VL53L0X_GetMaxNumberOfROIZones (**VL53L0X_DEV Dev,**
 uint8_t * pMaxNumb
)

Get the Maximum number of ROI Zones managed by the Device.

Function Description

Get Maximum number of ROI Zones managed by the Device.

Note

This function doesn't Access to the device

Parameters

Dev Device Handle
pMaxNumberOfROIZones Pointer to the Maximum Number of ROI Zones value.

Returns

VL53L0X_ERROR_NONE Success



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Functions

VL53L0X Interrupt Functions

[VL53L0X cut1.1 Function Definition](#)

Functions used for interrupt managements. [More...](#)

Functions

VL53L0X_API VL53L0X_Error	VL53L0X_SetGpioConfig (VL53L0X_DEV Dev, uint8_t Pin, VL53L0X_DeviceModes DeviceMode, VL53L0X_GpioFunctionality Functionality, VL53L0X_InterruptPolarity Polarity) Set the configuration of GPIO pin for a given device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetGpioConfig (VL53L0X_DEV Dev, uint8_t Pin, VL53L0X_DeviceModes *pDeviceMode, VL53L0X_GpioFunctionality *pFunctionality, VL53L0X_InterruptPolarity *pPolarity) Get current configuration for GPIO pin for a given device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetInterruptThresholds (VL53L0X_DEV Dev, VL53L0X_DeviceModes DeviceMode, FixPoint1616_t ThresholdLow, FixPoint1616_t ThresholdHigh) Set low and high Interrupt thresholds for a given mode (ranging, ALS, ...) for a given device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetInterruptThresholds (VL53L0X_DEV Dev, VL53L0X_DeviceModes DeviceMode, FixPoint1616_t *pThresholdLow, FixPoint1616_t

`*pThresholdHigh)`
Get high and low Interrupt thresholds
for a given mode (ranging, ALS, ...)
for a given device. [More...](#)

VL53L0X_API VL53L0X_Error **VL53L0X_GetStopCompletedStatus**
`(VL53L0X_DEV Dev, uint32_t *pStopStatus)`
Return device stop completion
status. [More...](#)

VL53L0X_API VL53L0X_Error **VL53L0X_ClearInterruptMask**
`(VL53L0X_DEV Dev, uint32_t InterruptMask)`
Clear given system interrupt
condition. [More...](#)

VL53L0X_API VL53L0X_Error **VL53L0X_GetInterruptMaskStatus**
`(VL53L0X_DEV Dev, uint32_t *pInterruptMaskStatus)`
Return device interrupt status.
[More...](#)

VL53L0X_API VL53L0X_Error **VL53L0X_EnableInterruptMask**
`(VL53L0X_DEV Dev, uint32_t InterruptMask)`
Configure ranging interrupt reported
to system. [More...](#)

Detailed Description

Functions used for interrupt managements.

Function Documentation

[VL53L0X_API](#)

[VL53L0X_Error](#)

```
VL53L0X_SetGpioConfig ( VL53L0X_DEV Dev,  
                          uint8_t Pin,  
                          VL53L0X_DeviceModes DeviceMode,  
                          VL53L0X_GpioFunctionality Functionality,  
                          VL53L0X_InterruptPolarity Polarity  
                        )
```

Set the configuration of GPIO pin for a given device.

Note

This function Access to the device

Parameters

Dev Device Handle

Pin ID of the GPIO Pin

Functionality Select Pin functionality. Refer to
[VL53L0X_GpioFunctionality](#)

DeviceMode Device Mode associated to the Gpio.

Polarity Set interrupt polarity. Active high or active low see
[VL53L0X_InterruptPolarity](#)

Returns

[VL53L0X_ERROR_NONE](#) Success

[VL53L0X_ERROR_GPIO_NOT_EXISTING](#) Only Pin=0 is accepted

[VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED](#)
This error occurs when Functionality programmed is not in the
supported list: Supported value are:

[VL53L0X_GPIOFUNCTIONALITY_OFF](#),

[VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_LOW](#)

[VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HIGH](#)

VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OU
VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY
"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API](#)
[VL53L0X_Error](#)

```
VL53L0X_GetGpioConfig ( VL53L0X_DEV Dev,
                           uint8_t Pin,
                           VL53L0X_DeviceModes * pDeviceMode,
                           VL53L0X_GpioFunctionality * pFunctionality,
                           VL53L0X_InterruptPolarity * pPolarity
                         )
```

Get current configuration for GPIO pin for a given device.

Note

This function Access to the device

Parameters

Dev	Device Handle
Pin	ID of the GPIO Pin
pDeviceMode	Pointer to Device Mode associated to the Gpio.
pFunctionality	Pointer to Pin functionality. Refer to VL53L0X_GpioFunctionality
pPolarity	Pointer to interrupt polarity. Active high or active low see VL53L0X_InterruptPolarity

Returns

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_GPIO_NOT_EXISTING Only Pin=0 is accepted
VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED error occurs when Functionality programmed is not in the supported range. Supported value are: VL53L0X_GPIOFUNCTIONALITY_OFF, VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_LOW, VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HIGH, VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OUT

VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY
"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_SetInterruptThresholds( VL53L0X_DEV           Dev,
                               VL53L0X_DeviceModes DeviceMode,
                               FixPoint1616_t      ThresholdLow,
                               FixPoint1616_t      ThresholdHigh
                           )
```

Set low and high Interrupt thresholds for a given mode (ranging, ALS, ..) given device.

Function Description

Set low and high Interrupt thresholds for a given mode (ranging, ALS, AI) for a given device

Note

This function Access to the device

DeviceMode is ignored for the current device

Parameters

Dev Device Handle

DeviceMode Device Mode for which change thresholds

ThresholdLow Low threshold (mm, lux ... , depending on the mode)

ThresholdHigh High threshold (mm, lux ... , depending on the mode)

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_GetInterruptThresholds( VL53L0X_DEV           Dev,
                                 VL53L0X_DeviceModes DeviceMode,
                                 FixPoint1616_t *      pThresholdLow,
                                 FixPoint1616_t *      pThresholdHigh
                             )
```

```
FixPoint1616_t * pThres  
)  
)
```

Get high and low Interrupt thresholds for a given mode (ranging, ALS, .
given device.

Function Description

Get high and low Interrupt thresholds for a given mode (ranging, ALS, .
a given device

Note

This function Access to the device
DeviceMode is ignored for the current device

Parameters

Dev Device Handle

DeviceMode Device Mode from which read thresholds

pThresholdLow Low threshold (mm, lux ...), depending on the mode

pThresholdHigh High threshold (mm, lux ...), depending on the mode

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
```

```
VL53L0X_GetStopCompletedStatus ( VL53L0X_DEV Dev,  
                                  uint32_t * pStopStatus  
)  
)
```

Return device stop completion status.

Function Description

Returns stop completion status. User shall call this function after
a stop command

Note

This function Access to the device

Parameters

Dev Device Handle
pStopStatus Pointer to status variable to update

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_ClearInterruptMask ( VL53L0X_DEV Dev,
                            uint32_t      InterruptMask
                          )
```

Clear given system interrupt condition.

Function Description

Clear given interrupt(s).

Note

This function Access to the device

Parameters

Dev Device Handle
InterruptMask Mask of interrupts to clear

Returns

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_INTERRUPT_NOT_CLEARED Cannot clear interrupts
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_GetInterruptMaskStatus ( VL53L0X_DEV Dev,
                                 uint32_t *      pInterruptMask
                               )
```

Return device interrupt status.

Function Description

Returns currently raised interrupts by the device. User shall be able to activate/deactivate interrupts through [VL53L0X_SetGpioConfig\(\)](#)

Note

This function Access to the device

Parameters

Dev Device Handle
plInterruptMaskStatus Pointer to status variable to update

Returns

VL53L0X_ERROR_NONE Success
"Other error code" See [VL53L0X_Error](#)

```
VL53L0X_API VL53L0X_Error
VL53L0X_EnableInterruptMask ( VL53L0X_DEV Dev,
                             uint32_t      InterruptMask
                           )
```

Configure ranging interrupt reported to system.

Note

This function is not Implemented

Parameters

Dev Device Handle
InterruptMask Mask of interrupt to Enable/disable (0:interrupt disabled or 1: interrupt enabled)

Returns

VL53L0X_ERROR_NOT_IMPLEMENTED Not implemented

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Functions

VL53L0X SPAD Functions

[VL53L0X cut1.1 Function Definition](#)

Functions used for SPAD managements. [More...](#)

Functions

VL53L0X_API VL53L0X_Error	VL53L0X_SetSpadAmbientDamperT (VL53L0X_DEV Dev, uint16_t SpadAmbientDamperThreshold) Set the SPAD Ambient Damper Thresl value. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetSpadAmbientDamperT (VL53L0X_DEV Dev, uint16_t *pSpadAmbientDamperThreshold) Get the current SPAD Ambient Dampe Threshold value. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetSpadAmbientDamperF (VL53L0X_DEV Dev, uint16_t SpadAmbientDamperFactor) Set the SPAD Ambient Damper Factor More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetSpadAmbientDamperF (VL53L0X_DEV Dev, uint16_t *pSpadAmbientDamperFactor) Get the current SPAD Ambient Dampe value. More...
VL53L0X_API VL53L0X_Error	VL53L0X_PerformRefSpadManagen (VL53L0X_DEV Dev, uint32_t *refSpa uint8_t *isApertureSpads) Performs Reference Spad Manageme More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetReferenceSpads (VL53L0X_DEV Dev, uint32_t refSpa uint8_t isApertureSpads) Applies Reference SPAD configurator

VL53L0X_API VL53L0X_Error **VL53L0X_GetReferenceSpads**
(**VL53L0X_DEV** Dev, **uint32_t** *refSpads,
uint8_t *isApertureSpads)
Retrieves SPAD configuration. [More...](#)

Detailed Description

Functions used for SPAD managements.

Function Documentation

VL53L0X_API VL53L0X_Error

**VL53L0X_SetSpadAmbientDamperThreshold (VL53L0X_DEV Dev,
 uint16_t SpadAmbientDamperThreshold)**

Set the SPAD Ambient Damper Threshold value.

Function Description

This function set the SPAD Ambient Damper Threshold value

Note

This function Access to the device

Parameters

Dev Device Handle
SpadAmbientDamperThreshold SPAD Ambient Damper Threshold

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See **VL53L0X_Error**

VL53L0X_API VL53L0X_Error

**VL53L0X_GetSpadAmbientDamperThreshold (VL53L0X_DEV Dev,
 uint16_t * pSpadAmbientDamperThreshold)**

Get the current SPAD Ambient Damper Threshold value.

Function Description

This function get the SPAD Ambient Damper Threshold value

)

Get the current SPAD Ambient Damper Factor value.

Function Description

This function get the SPAD Ambient Damper Factor value

Note

This function Access to the device

Parameters

Dev	Device Handle
pSpadAmbientDamperFactor	Pointer to programmed SPAD Aml value

Returns

VL53L0X_ERROR_NONE Success

"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API VL53L0X_Error](#)

```
VL53L0X_PerformRefSpadManagement ( VL53L0X_DEV Dev,
                                         uint32_t *      refSpadCount,
                                         uint8_t *       isAperture );
                                         )
```

Performs Reference Spad Management.

Function Description

The reference SPAD initialization procedure determines the minimum amount of reference spads to be enables to achieve a target reference signal rate and should be performed once during initialization.

Note

This function Access to the device

This function change the device mode to

VL53L0X_DEVICEMODE_SINGLE_RANGING

Parameters

Dev	Device Handle
refSpadCount	Reports ref Spad Count
isApertureSpads	Reports if spads are of type aperture or non-aperture. 1:=aperture, 0:=Non-Aperture

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_REF_SPAD_INIT Error in the Ref Spad procedure

"Other error code" See [VL53L0X_Error](#)

VL53L0X_API VL53L0X_Error

```
VL53L0X_SetReferenceSpads ( VL53L0X_DEV Dev,
                           uint32_t      refSpadCount,
                           uint8_t       isApertureSpads
                         )
```

Applies Reference SPAD configuration.

Function Description

This function applies a given number of reference spads, identified as either Aperture or Non-Aperture. The requested spad count and type are stored within the device specific parameters data for access by the host.

Note

This function Access to the device

Parameters

Dev	Device Handle
refSpadCount	Number of ref spads.
isApertureSpads	Defines if spads are of type aperture or non-aperture. 1:=aperture, 0:=Non-Aperture

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_REF_SPAD_INIT Error in the in the

reference spad configuration.

"Other error code" See [VL53L0X_Error](#)

[VL53L0X_API VL53L0X_Error](#)

```
VL53L0X_GetReferenceSpads ( VL53L0X_DEV Dev,
                            uint32_t *      refSpadCount,
                            uint8_t *       isApertureSpads
                          )
```

Retrieves SPAD configuration.

Function Description

This function retrieves the current number of applied reference spads and also their type : Aperture or Non-Aperture.

Note

This function Access to the device

Parameters

Dev	Device Handle
refSpadCount	Number ref Spad Count
isApertureSpads	Reports if spads are of type aperture or non-aperture. 1:=aperture, 0:=Non-Aperture

Returns

VL53L0X_ERROR_NONE Success

VL53L0X_ERROR_REF_SPAD_INIT Error in the reference spad configuration.

"Other error code" See [VL53L0X_Error](#)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[Modules](#) | [Data Structures](#) | [Macros](#)

VL53L0X Defines

VL53L0X Defines. [More...](#)

Modules

Error and Warning code returned by API

The following DEFINE are used to identify the PAL ERROR.

Defines Device modes

Defines all possible modes for the device.

Defines Histogram modes

Defines all possible Histogram modes for the device.

List of available Power Modes

List of available Power Modes.

Defines the current status of the device

Defines the current status of the device.

Defines the Polarity

of the Interrupt Defines the Polarity of the Interrupt

Vcsel Period Defines

Defines the range measurement for which to access the vcsel period.

Defines the steps

carried out by the scheduler during a range measurement.

Defines the Polarity

of the Interrupt Defines the the sequence steps performed during ranging.

General Macro Defines

General Macro Defines.

Data Structures

struct **VL53L0X_Version_t**

Defines the parameters of the Get Version Functions. [More...](#)

struct **VL53L0X_DeviceInfo_t**

Defines the parameters of the Get Device Info Functions.

[More...](#)

struct **VL53L0X_DeviceParameters_t**

Defines all parameters for the device. [More...](#)

struct **VL53L0X_DMaxData_t**

Structure containing the Dmax computation parameters and data. [More...](#)

struct **VL53L0X_RangingMeasurementData_t**

struct **VL53L0X_HistogramMeasurementData_t**

struct **VL53L0X_SpadData_t**

Spad Configuration Data. [More...](#)

struct **VL53L0X_DeviceSpecificParameters_t**

struct **VL53L0X_DevData_t**

VL53L0X PAL device ST private data structure

End user should never access any of these field directly.

[More...](#)

struct **VL53L0X_RangeData_t**

Range measurement data. [More...](#)

struct **VL53L0X_HistogramData_t**

Histogram measurement data. [More...](#)

Macros

```
#define VL53L0X10_SPECIFICATION_VER_MAJOR 1  
PAL SPECIFICATION major version. More...
```

```
#define VL53L0X10_SPECIFICATION_VER_MINOR 2  
PAL SPECIFICATION minor version. More...
```

```
#define VL53L0X10_SPECIFICATION_VER_SUB 7  
PAL SPECIFICATION sub version. More...
```

```
#define VL53L0X10_SPECIFICATION_VER_REVISION 1440  
PAL SPECIFICATION sub version. More...
```

```
#define VL53L0X10_IMPLEMENTATION_VER_MAJOR 1  
VL53L0X PAL IMPLEMENTATION major version. More...
```

```
#define VL53L0X10_IMPLEMENTATION_VER_MINOR 0  
VL53L0X PAL IMPLEMENTATION minor version. More...
```

```
#define VL53L0X10_IMPLEMENTATION_VER_SUB 9  
VL53L0X PAL IMPLEMENTATION sub version. More...
```

```
#define VL53L0X10_IMPLEMENTATION_VER_REVISION 3673  
VL53L0X PAL IMPLEMENTATION sub version. More...
```

```
#define VL53L0X_SPECIFICATION_VER_MAJOR 1  
PAL SPECIFICATION major version. More...
```

```
#define VL53L0X_SPECIFICATION_VER_MINOR 2  
PAL SPECIFICATION minor version. More...
```

```
#define VL53L0X_SPECIFICATION_VER_SUB 7  
PAL SPECIFICATION sub version. More...
```

```
#define VL53L0X_SPECIFICATION_VER_REVISION 1440  
PAL SPECIFICATION sub version. More...
```

```
#define VL53L0X_IMPLEMENTATION_VER_MAJOR 1  
VL53L0X PAL IMPLEMENTATION major version. More...
```

```
#define VL53L0X_IMPLEMENTATION_VER_MINOR 0  
VL53L0X PAL IMPLEMENTATION minor version. More...
```

```
#define VL53L0X_IMPLEMENTATION_VER_SUB 2  
VL53L0X PAL IMPLEMENTATION sub version. More...
```

```
#define VL53L0X_IMPLEMENTATION_VER_REVISION 4823  
VL53L0X PAL IMPLEMENTATION sub version. More...
```

```
#define VL53L0X_DEFAULT_MAX_LOOP 2000
```

```
#define VL53L0X_MAX_STRING_LENGTH 32
```

```
#define VL53L0X_HISTOGRAM_BUFFER_SIZE 24
```

```
#define VL53L0X_REF_SPAD_BUFFER_SIZE 6
```

Detailed Description

VL53L0X Defines.

Macro Definition Documentation

```
#define VL53L0X10_SPECIFICATION_VER_MAJOR 1
```

PAL SPECIFICATION major version.

Definition at line [52](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X10_SPECIFICATION_VER_MINOR 2
```

PAL SPECIFICATION minor version.

Definition at line [54](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X10_SPECIFICATION_VER_SUB 7
```

PAL SPECIFICATION sub version.

Definition at line [56](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X10_SPECIFICATION_VER_REVISION 1440
```

PAL SPECIFICATION sub version.

Definition at line [58](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X10_IMPLEMENTATION_VER_MAJOR 1
```

VL53L0X PAL IMPLEMENTATION major version.

Definition at line [61](#) of file `vl53l0x_def.h`.

```
#define VL53L0X10_IMPLEMENTATION_VER_MINOR 0
```

VL53L0X PAL IMPLEMENTATION minor version.

Definition at line [63](#) of file `vl53l0x_def.h`.

```
#define VL53L0X10_IMPLEMENTATION_VER_SUB 9
```

VL53L0X PAL IMPLEMENTATION sub version.

Definition at line [65](#) of file `vl53l0x_def.h`.

```
#define VL53L0X10_IMPLEMENTATION_VER_REVISION 3673
```

VL53L0X PAL IMPLEMENTATION sub version.

Definition at line [67](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_SPECIFICATION_VER_MAJOR 1
```

PAL SPECIFICATION major version.

Definition at line [70](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_SPECIFICATION_VER_MINOR 2
```

PAL SPECIFICATION minor version.

Definition at line [72](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_SPECIFICATION_VER_SUB 7
```

PAL SPECIFICATION sub version.

Definition at line [74](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_SPECIFICATION_VER_REVISION 1440
```

PAL SPECIFICATION sub version.

Definition at line [76](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_IMPLEMENTATION_VER_MAJOR 1
```

VL53L0X PAL IMPLEMENTATION major version.

Definition at line [79](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_IMPLEMENTATION_VER_MINOR 0
```

VL53L0X PAL IMPLEMENTATION minor version.

Definition at line [81](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_IMPLEMENTATION_VER_SUB 2
```

VL53L0X PAL IMPLEMENTATION sub version.

Definition at line [83](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_IMPLEMENTATION_VER_REVISION 4823
```

VL53L0X PAL IMPLEMENTATION sub version.

Definition at line [85](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_DEFAULT_MAX_LOOP 2000
```

Definition at line [86](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_MAX_STRING_LENGTH 32
```

Definition at line [87](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_HISTOGRAM_BUFFER_SIZE 24
```

Definition at line [346](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_REF_SPAD_BUFFER_SIZE 6
```

Definition at line [368](#) of file `vl53l0x_def.h`.



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

[Macros](#) | [Typedefs](#)

Error and Warning code returned by API

[VL53L0X Defines](#)

The following DEFINE are used to identify the PAL ERROR. More...

Macros

```
#define VL53L0X_ERROR_NONE ((VL53L0X_Error) 0)

#define VL53L0X_ERROR_CALIBRATION_WARNING ((VL53L0X_Error) -1)

#define VL53L0X_ERROR_MIN_CLIPPED ((VL53L0X_Error) -2)

#define VL53L0X_ERROR_UNDEFINED ((VL53L0X_Error) -3)

#define VL53L0X_ERROR_INVALID_PARAMS ((VL53L0X_Error) -4)

#define VL53L0X_ERROR_NOT_SUPPORTED ((VL53L0X_Error) -5)

#define VL53L0X_ERROR_RANGE_ERROR ((VL53L0X_Error) -6)

#define VL53L0X_ERROR_TIME_OUT ((VL53L0X_Error) -7)

#define VL53L0X_ERROR_MODE_NOT_SUPPORTED ((VL53L0X_Error) -8)

#define VL53L0X_ERROR_BUFFER_TOO_SMALL ((VL53L0X_Error) -9)

#define VL53L0X_ERROR_GPIO_NOT_EXISTING ((VL53L0X_Error) -10)

#define VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED ((VL53L0X_Error) -11)

#define VL53L0X_ERROR_INTERRUPT_NOT_CLEARED ((VL53L0X_Error) -12)

#define VL53L0X_ERROR_CONTROL_INTERFACE ((VL53L0X_Error) -13)

#define VL53L0X_ERROR_INVALID_COMMAND ((VL53L0X_Error) -14)

#define VL53L0X_ERROR_DIVISION_BY_ZERO ((VL53L0X_Error) -15)

#define VL53L0X_ERROR_REF_SPAD_INIT ((VL53L0X_Error) -50)
```

```
#define VL53L0X_ERROR_NOT_IMPLEMENTED ((VL53L0X_Error
```

TypeDefs

```
typedef int8_t VL53L0X_Error
```

Detailed Description

The following DEFINE are used to identify the PAL ERROR.

Macro Definition Documentation

```
#define VL53L0X_ERROR_NONE ((VL53L0X_Error) 0)
```

Definition at line [133](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_ERROR_CALIBRATION_WARNING ((VL53L0X_Error)  
-1)
```

Warning invalid calibration data may be in used *VL53L0X_InitData()*
VL53L0X_GetOffsetCalibrationData
VL53L0X_SetOffsetCalibrationData

Definition at line [134](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_ERROR_MIN_CLIPPED ((VL53L0X_Error) -2)
```

Warning parameter passed was clipped to min before to be applied

Definition at line [139](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_ERROR_UNDEFINED ((VL53L0X_Error) -3)
```

Unqualified error

Definition at line [142](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_ERROR_INVALID_PARAMS ((VL53L0X_Error)  
-4)
```

Parameter passed is invalid or out of range

Definition at line [144](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_ERROR_NOT_SUPPORTED ((VL53L0X_Error)-5)
```

Function is not supported in current mode or configuration

Definition at line [146](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_ERROR_RANGE_ERROR ((VL53L0X_Error)-6)
```

Device report a ranging error interrupt status

Definition at line [148](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_ERROR_TIME_OUT ((VL53L0X_Error)-7)
```

Aborted due to time out

Definition at line [150](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_ERROR_MODE_NOT_SUPPORTED ((VL53L0X_Error)-8)
```

Asked mode is not supported by the device

Definition at line [152](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_ERROR_BUFFER_TOO_SMALL ((VL53L0X_Error)-9)
```

...

Definition at line [154](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_ERROR_GPIO_NOT_EXISTING ((VL53L0X_Error) -10)
```

User tried to setup a non-existing GPIO pin

Definition at line [156](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED ((VL53L0X_Error)  
-11)
```

unsupported GPIO functionality

Definition at line [158](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_ERROR_INTERRUPT_NOT_CLEARED ((VL53L0X_Error)  
-12)
```

Error during interrupt clear

Definition at line [160](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_ERROR_CONTROL_INTERFACE ((VL53L0X_Error)  
-20)
```

error reported from IO functions

Definition at line [162](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_ERROR_INVALID_COMMAND ((VL53L0X_Error) -30)
```

The command is not allowed in the current device state (power down)

Definition at line [164](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_ERROR_DIVISION_BY_ZERO ((VL53L0X_Error) -40)
```

In the function a division by zero occurs

Definition at line [167](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_ERROR_REF_SPAD_INIT ((VL53L0X_Error)  
-50)
```

Error during reference SPAD initialization

Definition at line [169](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_ERROR_NOT_IMPLEMENTED ((VL53L0X_Error) -99)
```

Tells requested functionality has not been implemented yet or not compatible with the device

Definition at line [171](#) of file `vl53l0x_def.h`.

Typedef Documentation

```
typedef int8_t VL53L0X_Error
```

Definition at line [131](#) of file [vl53l0x_def.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[Macros](#) | [Typedefs](#)

Defines Device modes

[VL53L0X Defines](#)

Defines all possible modes for the device. [More...](#)

Macros

```
#define VL53L0X_DEVICEMODE_SINGLE_RANGING ((VL53L0X_DeviceMode)0x00000000)

#define VL53L0X_DEVICEMODE_CONTINUOUS_RANGING ((VL53L0X_DeviceMode)0x00000001)

#define VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM ((VL53L0X_DeviceMode)0x00000002)

#define VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING ((VL53L0X_DeviceMode)0x00000003)

#define VL53L0X_DEVICEMODE_SINGLE_ALS ((VL53L0X_DeviceMode)0x00000004)

#define VL53L0X_DEVICEMODE_GPIO_DRIVE ((VL53L0X_DeviceMode)0x00000005)

#define VL53L0X_DEVICEMODE_GPIO_OSC ((VL53L0X_DeviceMode)0x00000006)
```

Typedefs

```
typedef uint8_t VL53L0X_DeviceModes
```

Detailed Description

Defines all possible modes for the device.

Macro Definition Documentation

```
#define  
VL53L0X_DEVICEMODE_SINGLE_RANGING ((VL53L0X_DeviceM  
0)
```

Definition at line [183](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_DEVICEMODE_CONTINUOUS_RANGING ((VL53L0X_De  
1)
```

Definition at line [184](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM ((VL53L0X_Device  
2)
```

Definition at line [185](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING ((VL53  
3)
```

Definition at line [186](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_DEVICEMODE_SINGLE_ALS ((VL53L0X_DeviceModes)  
10)
```

Definition at line [187](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_DEVICEMODE_GPIO_DRIVE ((VL53L0X_DeviceModes)  
20)
```

Definition at line [188](#) of file `vl53l0x_def.h`.

```
#define  
VL53L0X_DEVICEMODE_GPIO_OSC ((VL53L0X_DeviceModes)  
21)
```

Definition at line [189](#) of file `vl53l0x_def.h`.

Typedef Documentation

```
typedef uint8_t VL53L0X_DeviceModes
```

Definition at line [181](#) of file [vl53l0x_def.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

[Macros](#) | [Typedefs](#)

Defines Histogram modes

[VL53L0X Defines](#)

Defines all possible Histogram modes for the device. [More...](#)

Macros

```
#define VL53L0X_HISTOGRAMMODE_DISABLED ((VL53L0X_Hist  
#define VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY ((VL53L0X_Hist  
#define VL53L0X_HISTOGRAMMODE_RETURN_ONLY ((VL53L0X_Hist  
#define VL53L0X_HISTOGRAMMODE_BOTH ((VL53L0X_Histogra
```

Typedefs

```
typedef uint8_t VL53L0X_HistogramModes
```

Detailed Description

Defines all possible Histogram modes for the device.

Macro Definition Documentation

```
#define  
VL53L0X_HISTOGRAMMODE_DISABLED ((VL53L0X_HistogramMode)  
0)
```

Histogram Disabled

Definition at line [201](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY ((VL53L0X_HistogramMode)  
1)
```

Histogram Reference array only

Definition at line [203](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_HISTOGRAMMODE_RETURN_ONLY ((VL53L0X_HistogramMode)  
2)
```

Histogram Return array only

Definition at line [205](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_HISTOGRAMMODE_BOTH ((VL53L0X_HistogramModes)  
3)
```

Histogram both Reference and Return Arrays

Definition at line [207](#) of file `vl53l0x_def.h`.

Typedef Documentation

```
typedef uint8_t VL53L0X_HistogramModes
```

Definition at line [199](#) of file [vl53l0x_def.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

[Macros](#) | [Typedefs](#)

List of available Power Modes

[VL53L0X Defines](#)

List of available Power Modes. [More...](#)

Macros

```
#define VL53L0X_POWERMODE_STANDBY_LEVEL1 ((VL53L0X_  
0)  
  
#define VL53L0X_POWERMODE_STANDBY_LEVEL2 ((VL53L0X_  
1)  
  
#define VL53L0X_POWERMODE_IDLE_LEVEL1 ((VL53L0X_Powe  
  
#define VL53L0X_POWERMODE_IDLE_LEVEL2 ((VL53L0X_Powe
```

Typedefs

```
typedef uint8_t VL53L0X_PowerModes
```

Detailed Description

List of available Power Modes.

Macro Definition Documentation

```
#define  
VL53L0X_POWERMODE_STANDBY_LEVEL1 ((VL53L0X_PowerMode)  
0)
```

Standby level 1

Definition at line [220](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_POWERMODE_STANDBY_LEVEL2 ((VL53L0X_PowerMode)  
1)
```

Standby level 2

Definition at line [222](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_POWERMODE_IDLE_LEVEL1 ((VL53L0X_PowerModes)  
2)
```

Idle level 1

Definition at line [224](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_POWERMODE_IDLE_LEVEL2 ((VL53L0X_PowerModes)  
3)
```

Idle level 2

Definition at line [226](#) of file `vl53l0x_def.h`.

Typedef Documentation

```
typedef uint8_t VL53L0X_PowerModes
```

Definition at line [218](#) of file [vl53l0x_def.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

[Macros](#) | [Typedefs](#)

Defines the current status of the device

[VL53L0X Defines](#)

Defines the current status of the device. More...

Macros

```
#define VL53L0X_STATE_POWERDOWN ((VL53L0X_State) 0)

#define VL53L0X_STATE_WAIT_STATICINIT ((VL53L0X_State) 1)

#define VL53L0X_STATE_STANDBY ((VL53L0X_State) 2)

#define VL53L0X_STATE_IDLE ((VL53L0X_State) 3)

#define VL53L0X_STATE_RUNNING ((VL53L0X_State) 4)

#define VL53L0X_STATE_UNKNOWN ((VL53L0X_State) 98)

#define VL53L0X_STATE_ERROR ((VL53L0X_State) 99)
```

Typedefs

```
typedef uint8_t VL53L0X_State
```

Detailed Description

Defines the current status of the device.

Macro Definition Documentation

```
#define VL53L0X_STATE_POWERDOWN ((VL53L0X_State) 0)
```

Device is in HW reset

Definition at line [275](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_STATE_WAIT_STATICINIT ((VL53L0X_State)  
1)
```

Device is initialized and wait for static initialization

Definition at line [277](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_STATE_STANDBY ((VL53L0X_State) 2)
```

Device is in Low power Standby mode

Definition at line [279](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_STATE_IDLE ((VL53L0X_State) 3)
```

Device has been initialized and ready to do measurements

Definition at line [281](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_STATE_RUNNING ((VL53L0X_State) 4)
```

Device is performing measurement

Definition at line [283](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_STATE_UNKNOWN ((VL53L0X_State) 98)
```

Device is in unknown state and need to be rebooted

Definition at line [285](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_STATE_ERROR ((VL53L0X_State) 99)
```

Device is in error state and need to be rebooted

Definition at line [287](#) of file [vl53l0x_def.h](#).

Typedef Documentation

```
typedef uint8_t VL53L0X_State
```

Definition at line [273](#) of file [vl53l0x_def.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

[Macros](#) | [Typedefs](#)

Defines the Polarity

[VL53L0X Defines](#)

of the Interrupt Defines the Polarity of the Interrupt [More...](#)

Macros

```
#define VL53L0X_INTERRUPTPOLARITY_LOW ((VL53L0X_InterruptPolarity & 0x00000001) == 0)
#define VL53L0X_INTERRUPTPOLARITY_HIGH ((VL53L0X_InterruptPolarity & 0x00000001) == 1)
```

Typedefs

```
typedef uint8_t VL53L0X_InterruptPolarity
```

Detailed Description

of the Interrupt Defines the Polarity of the Interrupt

Macro Definition Documentation

```
#define  
VL53L0X_INTERRUPTPOLARITY_LOW ((VL53L0X InterruptPolar  
0)
```

Set active low polarity best setup for falling edge.

Definition at line [498](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_INTERRUPTPOLARITY_HIGH ((VL53L0X InterruptPolar  
1)
```

Set active high polarity best setup for rising edge.

Definition at line [500](#) of file [vl53l0x_def.h](#).

Typedef Documentation

```
typedef uint8_t VL53L0X_InterruptPolarity
```

Definition at line [496](#) of file [vl53l0x_def.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

[Macros](#) | [Typedefs](#)

Vcsel Period Defines

[VL53L0X Defines](#)

Defines the range measurement for which to access the vcsel period.
[More...](#)

Macros

```
#define VL53L0X_VCSEL_PERIOD_PRE_RANGE ((VL53L0X_VcselPeriodPreRange) 0)  
  
#define VL53L0X_VCSEL_PERIOD_FINAL_RANGE ((VL53L0X_VcselPeriodFinalRange) 1)
```

Typedefs

```
typedef uint8_t VL53L0X_VcselPeriod
```

Detailed Description

Defines the range measurement for which to access the vcsel period.

Macro Definition Documentation

```
#define  
VL53L0X_VCSEL_PERIOD_PRE_RANGE ((VL53L0X_VcselPeriod)  
0)
```

Identifies the pre-range vcsel period.

Definition at line [512](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_VCSEL_PERIOD_FINAL_RANGE ((VL53L0X_VcselPeriod)  
1)
```

Identifies the final range vcsel period.

Definition at line [514](#) of file [vl53l0x_def.h](#).

Typedef Documentation

```
typedef uint8_t VL53L0X_VcselPeriod
```

Definition at line [510](#) of file [vl53l0x_def.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

Data Structures

Defines the steps

VL53L0X Defines

carried out by the scheduler during a range measurement. [More...](#)

Data Structures

```
struct VL53L0X_SchedulerSequenceSteps_t
```

Detailed Description

carried out by the scheduler during a range measurement.

Defines the states of all the steps in the scheduler i.e.
enabled/disabled.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_SchedulerSequenceSteps_t Struct Reference

[VL53L0X Defines](#) » [Defines the steps](#)

#include <[vl53l0x_def.h](#)>

Data Fields

`uint8_t TccOn`

`uint8_t MsrcOn`

`uint8_t DssOn`

`uint8_t PreRangeOn`

`uint8_t FinalRangeOn`

Detailed Description

Definition at line **525** of file **vl53l0x_def.h**.

Field Documentation

uint8_t VL53L0X_SchedulerSequenceSteps_t::TccOn

Reports if Target Centre Check On

Definition at line [526](#) of file [vl53l0x_def.h](#).

uint8_t VL53L0X_SchedulerSequenceSteps_t::MsrcOn

Reports if MSRC On

Definition at line [527](#) of file [vl53l0x_def.h](#).

uint8_t VL53L0X_SchedulerSequenceSteps_t::DssOn

Reports if DSS On

Definition at line [528](#) of file [vl53l0x_def.h](#).

uint8_t VL53L0X_SchedulerSequenceSteps_t::PreRangeOn

Reports if Pre-Range On

Definition at line [529](#) of file [vl53l0x_def.h](#).

uint8_t VL53L0X_SchedulerSequenceSteps_t::FinalRangeOn

Reports if Final-Range On

Definition at line **530** of file [**vl53l0x_def.h**](#).

The documentation for this struct was generated from the following file:

- [**vl53l0x_def.h**](#)
-

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

[Macros](#) | [Typedefs](#)

Defines the Polarity

[VL53L0X Defines](#)

of the Interrupt Defines the the sequence steps performed during ranging. [More...](#)

Macros

```
#define VL53L0X_SEQUENCESTEP_TCC ((VL53L0X_VcselPeriod  
#define VL53L0X_SEQUENCESTEP_DSS ((VL53L0X_VcselPeriod  
#define VL53L0X_SEQUENCESTEP_MSRC ((VL53L0X_VcselPeriod  
#define VL53L0X_SEQUENCESTEP_PRE_RANGE ((VL53L0X_VcselPeriod  
    3)  
#define VL53L0X_SEQUENCESTEP_FINAL_RANGE ((VL53L0X_VcselPeriod  
    4)  
#define VL53L0X_SEQUENCESTEP_NUMBER_OF_CHECKS 5
```

Typedefs

```
typedef uint8_t VL53L0X_SequenceStepId
```

Detailed Description

of the Interrupt Defines the the sequence steps performed during ranging.

Macro Definition Documentation

```
#define  
VL53L0X_SEQUENCESTEP_TCC ((VL53L0X_VcselPeriod) 0)
```

Target CentreCheck identifier.

Definition at line [542](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_SEQUENCESTEP_DSS ((VL53L0X_VcselPeriod) 1)
```

Dynamic Spad Selection function Identifier.

Definition at line [544](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_SEQUENCESTEP_MSRC ((VL53L0X_VcselPeriod) 2)
```

Minimum Signal Rate Check function Identifier.

Definition at line [546](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_SEQUENCESTEP_PRE_RANGE ((VL53L0X_VcselPeriod)  
3)
```

Pre-Range check Identifier.

Definition at line [548](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_SEQUENCESTEP_FINAL_RANGE ((VL53L0X_VcselPeri  
4)
```

Final Range Check Identifier.

Definition at line [550](#) of file [vl53l0x_def.h](#).

```
#define VL53L0X_SEQUENCESTEP_NUMBER_OF_CHECKS 5
```

Number of Sequence Step Managed by the API.

Definition at line [553](#) of file [vl53l0x_def.h](#).

Typedef Documentation

```
typedef uint8_t VL53L0X_SequenceStepId
```

Definition at line **540** of file **vl53l0x_def.h**.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Macros

General Macro Defines

[VL53L0X Defines](#)

General Macro Defines. [More...](#)

Macros

#define	VL53L0X_SETPARAMETERFIELD (Dev, field, value) PALDevDataSet (Dev, CurrentParameters.field, value)
#define	VL53L0X_GETPARAMETERFIELD (Dev, field, variable) variable = PALDevDataGet (Dev, CurrentParameters).field
#define	VL53L0X_SETARRAYPARAMETERFIELD (Dev, field, index, value) PALDevDataSet (Dev, CurrentParameters.field[index], value)
#define	VL53L0X_GETARRAYPARAMETERFIELD (Dev, field, index, variable) variable = PALDevDataGet (Dev, CurrentParameters).field[index]
#define	VL53L0X_SETDEVICESPECIFICPARAMETER (Dev, field, value) PALDevDataSet (Dev, DeviceSpecificParameters.field, value)
#define	VL53L0X_GETDEVICESPECIFICPARAMETER (Dev, field) PALDevDataGet (Dev, DeviceSpecificParameters).field
#define	VL53L0X_FIXPOINT1616TOFIXPOINT97 (Value) (uint16_t)((Value>>9)&0xFFFF)
#define	VL53L0X_FIXPOINT97TOFIXPOINT1616 (Value) (FixPoint16)(Value<<9)
#define	VL53L0X_FIXPOINT1616TOFIXPOINT88 (Value) (uint16_t)((Value>>8)&0xFFFF)
#define	VL53L0X_FIXPOINT88TOFIXPOINT1616 (Value) (FixPoint16)(Value<<8)
#define	VL53L0X_FIXPOINT1616TOFIXPOINT412 (Value) (uint16_t)((Value>>4)&0xFFFF)
#define	VL53L0X_FIXPOINT412TOFIXPOINT1616 (Value) (FixPoint16)(Value<<4)

```
(Value<<4)

#define VL53L0X_FIXPOINT1616TOFIXPOINT313(Value) (uint16_t)
((Value>>3)&0xFFFF)

#define VL53L0X_FIXPOINT313TOFIXPOINT1616(Value) (FixPoint)
(Value<<3)

#define VL53L0X_FIXPOINT1616TOFIXPOINT08(Value) (uint8_t)
((Value>>8)&0x00FF)

#define VL53L0X_FIXPOINT08TOFIXPOINT1616(Value) (FixPoint1)
(Value<<8)

#define VL53L0X_FIXPOINT1616TOFIXPOINT53(Value) (uint8_t)
((Value>>13)&0x00FF)

#define VL53L0X_FIXPOINT53TOFIXPOINT1616(Value) (FixPoint1)
(Value<<13)

#define VL53L0X_FIXPOINT1616TOFIXPOINT102(Value) (uint16_t)
((Value>>14)&0x0FFF)

#define VL53L0X_FIXPOINT102TOFIXPOINT1616(Value) (FixPoint)
(Value<<12)

#define VL53L0X_MAKEUINT16(lsb, msb)
```

Detailed Description

General Macro Defines.

Macro Definition Documentation

```
#define  
VL53L0X_SETPARAMETERFIELD( Dev,  
                             field,  
                             value  
                           PALDevDataSet(Dev,  
                                         CurrentParameters.field,  
                                         ) value)
```

Definition at line [566](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_GETPARAMETERFIELD( Dev,  
                            field,  
                            variable  
                            variable =  
                            PALDevDataGet(Dev,  
                                         CurrentParameters).field
```

Definition at line [569](#) of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_SETARRAYPARAMETERFIELD( Dev,  
                                 field,  
                                 index,  
                                 value  
                               PALDevDataSet(Dev,  
                                             CurrentParameters.field,  
                                             ) value)
```

Definition at line 573 of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_GETARRAYPARAMETERFIELD( Dev,  
field,  
index,  
variable  
    variable =  
PALDevDataGet(Dev,  
) CurrentParameters).fi
```

Definition at line 576 of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_SETDEVICESPECIFICPARAMETER( Dev,  
field,  
value  
    PALDevDataSet(Dev,  
DeviceSpecificPara  
) value)
```

Definition at line 580 of file [vl53l0x_def.h](#).

```
#define  
VL53L0X_GETDEVICESPECIFICPARAMETER( Dev,  
field  
    PALDevDataGet(  
) DeviceSpecificPar
```

Definition at line 583 of file [vl53l0x_def.h](#).

```
#define (uint16_t)
```

```
VL53L0X_FIXPOINT1616TOFIXPOINT97 ( Value ) ((Value>>9)&0xFF)
```

Definition at line 587 of file [vl53l0x_def.h](#).

```
#define VL53L0X_FIXPOINT97TOFIXPOINT1616 ( Value ) (Value<<9) (FixPoint1616_t)
```

Definition at line 589 of file [vl53l0x_def.h](#).

```
#define VL53L0X_FIXPOINT1616TOFIXPOINT88 ( Value ) ((Value>>8)&0xFF) (uint16_t)
```

Definition at line 592 of file [vl53l0x_def.h](#).

```
#define VL53L0X_FIXPOINT88TOFIXPOINT1616 ( Value ) (Value<<8) (FixPoint1616_t)
```

Definition at line 594 of file [vl53l0x_def.h](#).

```
#define VL53L0X_FIXPOINT1616TOFIXPOINT412 ( Value ) ((Value>>4)&0xFF) (uint16_t)
```

Definition at line 597 of file [vl53l0x_def.h](#).

```
#define VL53L0X_FIXPOINT412TOFIXPOINT1616 ( Value ) (Value<<4) (FixPoint1616_t)
```

Definition at line 599 of file [vl53l0x_def.h](#).

```
#define VL53L0X_FIXPOINT1616TOFIXPOINT313 ( Value ) ((Value>>3)&0xFF) (uint16_t)
```

Definition at line [602](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_FIXPOINT313TOFIXPOINT1616 ( Value ) (Value<<3) (FixPoint1616_t)
```

Definition at line [604](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_FIXPOINT1616TOFIXPOINT08 ( Value ) ((Value>>8)&0x00FF) (uint8_t)
```

Definition at line [607](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_FIXPOINT08TOFIXPOINT1616 ( Value ) (Value<<8) (FixPoint1616_t)
```

Definition at line [609](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_FIXPOINT1616TOFIXPOINT53 ( Value ) ((Value>>13)&0x0000FFFF) (uint8_t)
```

Definition at line [612](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_FIXPOINT53TOFIXPOINT1616 ( Value ) (Value<<13) (FixPoint1616_t)
```

Definition at line [614](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_FIXPOINT1616TOFIXPOINT102 ( Value ) ((Value>>14)&0x000000FF) (uint16_t)
```

Definition at line [617](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_FIXPOINT102TOFIXPOINT1616( Value ) (Value<<12) (FixPoint1616)
```

Definition at line [619](#) of file `vl53l0x_def.h`.

```
#define VL53L0X_MAKEUINT16( lsb,  
                           msb  
                         )
```

Value:

```
(uint16_t)((((uint16_t)msb)<<8) + \  
           (uint16_t)lsb)
```

Definition at line [622](#) of file `vl53l0x_def.h`.



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_Version_t Struct Reference

[VL53L0X Defines](#)

Defines the parameters of the Get Version Functions. More...

```
#include <vl53l0x_def.h>
```

Data Fields

`uint32_t revision`

`uint8_t major`

`uint8_t minor`

`uint8_t build`

Detailed Description

Defines the parameters of the Get Version Functions.

Definition at line **100** of file **vl53l0x_def.h**.

Field Documentation

uint32_t VL53L0X_Version_t::revision

revision number

Definition at line **101** of file [vl53l0x_def.h](#).

uint8_t VL53L0X_Version_t::major

major number

Definition at line **102** of file [vl53l0x_def.h](#).

uint8_t VL53L0X_Version_t::minor

minor number

Definition at line **103** of file [vl53l0x_def.h](#).

uint8_t VL53L0X_Version_t::build

build number

Definition at line **104** of file [vl53l0x_def.h](#).

The documentation for this struct was generated from the following file:

- [vl53l0x_def.h](#)
-

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_DeviceInfo_t Struct Reference

[VL53L0X Defines](#)

Defines the parameters of the Get Device Info Functions. [More...](#)

```
#include <vl53l0x_def.h>
```

Data Fields

char **Name** [VL53L0X_MAX_STRING_LENGTH]

char **Type** [VL53L0X_MAX_STRING_LENGTH]

char **ProductId** [VL53L0X_MAX_STRING_LENGTH]

uint8_t ProductType

uint8_t ProductRevisionMajor

uint8_t ProductRevisionMinor

Detailed Description

Defines the parameters of the Get Device Info Functions.

Definition at line [110](#) of file **vl53l0x_def.h**.

Field Documentation

`char`

`VL53L0X_DeviceInfo_t::Name[VL53L0X_MAX_STRING_LENGTH]`

Name of the Device e.g. Left_Distance

Definition at line [111](#) of file `vl53l0x_def.h`.

`char`

`VL53L0X_DeviceInfo_t::Type[VL53L0X_MAX_STRING_LENGTH]`

Type of the Device e.g VL53L0X

Definition at line [113](#) of file `vl53l0x_def.h`.

`char`

`VL53L0X_DeviceInfo_t::ProductId[VL53L0X_MAX_STRING_LENGTH]`

Product Identifier String

Definition at line [115](#) of file `vl53l0x_def.h`.

`uint8_t VL53L0X_DeviceInfo_t::ProductType`

Product Type, VL53L0X = 1, VL53L1 = 2

Definition at line [117](#) of file `vl53l0x_def.h`.

`uint8_t VL53L0X_DeviceInfo_t::ProductRevisionMajor`

Product revision major

Definition at line [119](#) of file [vl53l0x_def.h](#).

```
uint8_t VL53L0X_DeviceInfo_t::ProductRevisionMinor
```

Product revision minor

Definition at line [121](#) of file [vl53l0x_def.h](#).

The documentation for this struct was generated from the following file:

- [vl53l0x_def.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

Data Fields

VL53L0X_DeviceParameters_t Struct Reference

[VL53L0X Defines](#)

Defines all parameters for the device. [More...](#)

```
#include <vl53l0x_def.h>
```

Data Fields

VL53L0X_DeviceModes DeviceMode

VL53L0X_HistogramModes HistogramMode

uint32_t MeasurementTimingBudgetMicroSeconds

uint32_t InterMeasurementPeriodMilliSeconds

uint8_t XTalkCompensationEnable

uint16_t XTalkCompensationRangeMilliMeter

FixPoint1616_t XTalkCompensationRateMegaCps

int32_t RangeOffsetMicroMeters

**uint8_t LimitChecksEnable
[VL53L0X_CHECKENABLE_NUMBER]**

**uint8_t LimitChecksStatus
[VL53L0X_CHECKENABLE_NUMBER]**

**FixPoint1616_t LimitChecksValue
[VL53L0X_CHECKENABLE_NUMBER]**

uint8_t WrapAroundCheckEnable

Detailed Description

Defines all parameters for the device.

Definition at line **234** of file **vl53l0x_def.h**.

Field Documentation

VL53L0X_DeviceModes

VL53L0X_DeviceParameters_t::DeviceMode

Defines type of measurement to be done for the next measure

Definition at line [235](#) of file [vl53l0x_def.h](#).

VL53L0X_HistogramModes

VL53L0X_DeviceParameters_t::HistogramMode

Defines type of histogram measurement to be done for the next measure

Definition at line [237](#) of file [vl53l0x_def.h](#).

uint32_t

VL53L0X_DeviceParameters_t::MeasurementTimingBudgetMicroS

Defines the allowed total time for a single measurement

Definition at line [240](#) of file [vl53l0x_def.h](#).

uint32_t

VL53L0X_DeviceParameters_t::InterMeasurementPeriodMilliSecor

Defines time between two consecutive measurements (between two measurement starts). If set to 0 means back-to-back mode

Definition at line [242](#) of file [vl53l0x_def.h](#).

`uint8_t`

`VL53L0X_DeviceParameters_t::XTalkCompensationEnable`

Tells if Crosstalk compensation shall be enable or not

Definition at line [245](#) of file `vl53l0x_def.h`.

`uint16_t`

`VL53L0X_DeviceParameters_t::XTalkCompensationRangeMilliMet`

CrossTalk compensation range in millimeter

Definition at line [247](#) of file `vl53l0x_def.h`.

`FixPoint1616_t`

`VL53L0X_DeviceParameters_t::XTalkCompensationRateMegaCps`

CrossTalk compensation rate in Mega counts per seconds.
Expressed in 16.16 fixed point format.

Definition at line [249](#) of file `vl53l0x_def.h`.

`int32_t VL53L0X_DeviceParameters_t::RangeOffsetMicroMeters`

Range offset adjustment (mm).

Definition at line [252](#) of file `vl53l0x_def.h`.

`uint8_t`

`VL53L0X_DeviceParameters_t::LimitChecksEnable[VL53L0X_CHE`

This Array store all the Limit Check enable for this device.

Definition at line [255](#) of file `vl53l0x_def.h`.

`uint8_t`

`VL53L0X_DeviceParameters_t::LimitChecksStatus[VL53L0X_CHECKS_STATUS]`

This Array store all the Status of the check linked to last measurement.

Definition at line [257](#) of file `vl53l0x_def.h`.

`FixPoint1616_t`

`VL53L0X_DeviceParameters_t::LimitChecksValue[VL53L0X_CHECKS_VALUE]`

This Array store all the Limit Check value for this device

Definition at line [260](#) of file `vl53l0x_def.h`.

`uint8_t`

`VL53L0X_DeviceParameters_t::WrapAroundCheckEnable`

Tells if Wrap Around Check shall be enable or not

Definition at line [263](#) of file `vl53l0x_def.h`.

The documentation for this struct was generated from the following file:

- `vl53l0x_def.h`

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_DMaxData_t Struct Reference

[VL53L0X Defines](#)

Structure containing the Dmax computation parameters and data.

[More...](#)

```
#include <vl53l0x_def.h>
```

Data Fields

int32_t AmbTuningWindowFactor_K

int32_t RetSignalAt0mm

Detailed Description

Structure containing the Dmax computation parameters and data.

Definition at line **295** of file **vl53l0x_def.h**.

Field Documentation

int32_t VL53L0X_DMaxData_t::AmbTuningWindowFactor_K

internal algo tuning (*1000)

Definition at line [296](#) of file [vl53l0x_def.h](#).

int32_t VL53L0X_DMaxData_t::RetSignalAt0mm

intermediate dmax computation value caching

Definition at line [298](#) of file [vl53l0x_def.h](#).

The documentation for this struct was generated from the following file:

- [vl53l0x_def.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_RangingMeasurementData_t Struct Reference

[VL53L0X Defines](#)

```
#include <vl53l0x_def.h>
```

Data Fields

`uint32_t TimeStamp`

`uint32_t MeasurementTimeUsec`

`uint16_t RangeMilliMeter`

`uint16_t RangeDMaxMilliMeter`

`FixPoint1616_t SignalRateRtnMegaCps`

`FixPoint1616_t AmbientRateRtnMegaCps`

`uint16_t EffectiveSpadRtnCount`

`uint8_t ZoneId`

`uint8_t RangeFractionalPart`

`uint8_t RangeStatus`

Detailed Description

Definition at line **306** of file **vl53l0x_def.h**.

Field Documentation

uint32_t VL53L0X_RangingMeasurementData_t::TimeStamp

32-bit time stamp.

Definition at line [307](#) of file [vl53l0x_def.h](#).

**uint32_t
VL53L0X_RangingMeasurementData_t::MeasurementTimeUsec**

Give the Measurement time needed by the device to do the measurement.

Definition at line [308](#) of file [vl53l0x_def.h](#).

uint16_t VL53L0X_RangingMeasurementData_t::RangeMilliMeter

range distance in millimeter.

Definition at line [313](#) of file [vl53l0x_def.h](#).

**uint16_t
VL53L0X_RangingMeasurementData_t::RangeDMaxMilliMeter**

Tells what is the maximum detection distance of the device in current setup and environment conditions (Filled when applicable)

Definition at line [315](#) of file [vl53l0x_def.h](#).

FixPoint1616_t

VL53L0X_RangingMeasurementData_t::SignalRateRtnMegaCps

Return signal rate (MCPS)
these is a 16.16 fix point value, which is effectively a measure of target reflectance.

Definition at line [320](#) of file [vl53l0x_def.h](#).

FixPoint1616_t

VL53L0X_RangingMeasurementData_t::AmbientRateRtnMegaCps

Return ambient rate (MCPS)
these is a 16.16 fix point value, which is effectively a measure of the ambient light.

Definition at line [324](#) of file [vl53l0x_def.h](#).

uint16_t

VL53L0X_RangingMeasurementData_t::EffectiveSpadRtnCount

Return the effective SPAD count for the return signal. To obtain Real value it should be divided by 256

Definition at line [329](#) of file [vl53l0x_def.h](#).

uint8_t VL53L0X_RangingMeasurementData_t::ZoneId

Denotes which zone and range scheduler stage the range data relates to.

Definition at line [333](#) of file [vl53l0x_def.h](#).

uint8_t

VL53L0X_RangingMeasurementData_t::RangeFractionalPart

Fractional part of range distance. Final value is a FixPoint168 value.

Definition at line [336](#) of file [vl53l0x_def.h](#).

`uint8_t VL53L0X_RangingMeasurementData_t::RangeStatus`

Range Status for the current measurement. This is device dependent. Value = 0 means value is valid. See [RangeStatus](#)

Definition at line [339](#) of file [vl53l0x_def.h](#).

The documentation for this struct was generated from the following file:

- [vl53l0x_def.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_HistogramMeasurementData_t Struct Reference

[VL53L0X Defines](#)

```
#include <vl53l0x_def.h>
```

Data Fields

`uint32_t HistogramData
[VL53L0X_HISTOGRAM_BUFFER_SIZE]`

`uint8_t HistogramType`

`uint8_t FirstBin`

`uint8_t BufferSize`

`uint8_t NumberOfBins`

`VL53L0X_DeviceError ErrorStatus`

Detailed Description

Definition at line **352** of file **vl53l0x_def.h**.

Field Documentation

`uint32_t`

`VL53L0X_HistogramMeasurementData_t::HistogramData[VL53L0X]`

Histogram data

Definition at line [354](#) of file `vl53l0x_def.h`.

`uint8_t VL53L0X_HistogramMeasurementData_t::HistogramType`

Indicate the types of histogram data : Return only, Reference only, both Return and Reference

Definition at line [356](#) of file `vl53l0x_def.h`.

`uint8_t VL53L0X_HistogramMeasurementData_t::FirstBin`

First Bin value

Definition at line [358](#) of file `vl53l0x_def.h`.

`uint8_t VL53L0X_HistogramMeasurementData_t::BufferSize`

Buffer Size - Set by the user.

Definition at line [359](#) of file `vl53l0x_def.h`.

`uint8_t VL53L0X_HistogramMeasurementData_t::NumberOfBins`

Number of bins filled by the histogram measurement

Definition at line [360](#) of file [vl53l0x_def.h](#).

[**VL53L0X_DeviceError**](#)

[**VL53L0X_HistogramMeasurementData_t::ErrorStatus**](#)

Error status of the current measurement.

see [**VL53L0X_DeviceError**](#) [**VL53L0X_GetStatusErrorString\(\)**](#)

Definition at line [363](#) of file [vl53l0x_def.h](#).

The documentation for this struct was generated from the following file:

- [vl53l0x_def.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_SpadData_t Struct Reference

[VL53L0X Defines](#)

Spad Configuration Data. [More...](#)

```
#include <vl53l0x_def.h>
```

Data Fields

uint8_t RefSpadEnables [VL53L0X_REF_SPAD_BUFFER_SIZE]

uint8_t RefGoodSpadMap [VL53L0X_REF_SPAD_BUFFER_SIZE]

Detailed Description

Spad Configuration Data.

Definition at line **374** of file **vl53l0x_def.h**.

Field Documentation

`uint8_t`

`VL53L0X_SpadData_t::RefSpadEnables[VL53L0X_REF_SPAD_BU`

Reference Spad Enables

Definition at line [375](#) of file `vl53l0x_def.h`.

`uint8_t`

`VL53L0X_SpadData_t::RefGoodSpadMap[VL53L0X_REF_SPAD_BU`

Reference Spad Good Spad Map

Definition at line [377](#) of file `vl53l0x_def.h`.

The documentation for this struct was generated from the following file:

- `vl53l0x_def.h`

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_DeviceSpecificParameters_t Struct Reference

[VL53L0X Defines](#)

#include <[vl53l0x_def.h](#)>

Data Fields

FixPoint1616_t OscFrequencyMHz

uint16_t LastEncodedTimeout

VL53L0X_GpioFunctionality Pin0GpioFunctionality

uint32_t FinalRangeTimeoutMicroSecs

uint8_t FinalRangeVcselPulsePeriod

uint32_t PreRangeTimeoutMicroSecs

uint8_t PreRangeVcselPulsePeriod

uint16_t SigmaEstRefArray

uint16_t SigmaEstEffPulseWidth

uint16_t SigmaEstEffAmbWidth

uint8_t ReadDataFromDeviceDone

uint8_t ModuleId

uint8_t Revision

char ProductId
[VL53L0X_MAX_STRING_LENGTH]

uint8_t ReferenceSpadCount

uint8_t ReferenceSpadType

uint8_t RefSpadsInitialised

uint32_t PartUIDUpper

uint32_t PartUIDLower

FixPoint1616_t SignalRateMeasFixed400mm

Detailed Description

Definition at line **381** of file **vl53l0x_def.h**.

Field Documentation

FixPoint1616_t

VL53L0X_DeviceSpecificParameters_t::OscFrequencyMHz

Definition at line **382** of file **vl53l0x_def.h**.

uint16_t

VL53L0X_DeviceSpecificParameters_t::LastEncodedTimeout

Definition at line **384** of file **vl53l0x_def.h**.

VL53L0X_GpioFunctionality

VL53L0X_DeviceSpecificParameters_t::Pin0GpioFunctionality

Definition at line **387** of file **vl53l0x_def.h**.

uint32_t

VL53L0X_DeviceSpecificParameters_t::FinalRangeTimeoutMicroS

Execution time of the final range

Definition at line **390** of file **vl53l0x_def.h**.

uint8_t

VL53L0X_DeviceSpecificParameters_t::FinalRangeVcselPulsePeri

Vcsel pulse period (pll clocks) for the final range measurement

Definition at line **392** of file **vl53l0x_def.h**.

`uint32_t`

`VL53L0X_DeviceSpecificParameters_t::PreRangeTimeoutMicroSec`

Execution time of the final range

Definition at line [394](#) of file `vl53l0x_def.h`.

`uint8_t`

`VL53L0X_DeviceSpecificParameters_t::PreRangeVcselPulsePeriod`

Vcsel pulse period (pll clocks) for the pre-range measurement

Definition at line [396](#) of file `vl53l0x_def.h`.

`uint16_t`

`VL53L0X_DeviceSpecificParameters_t::SigmaEstRefArray`

Reference array sigma value in 1/100th of [mm] e.g. 100 = 1mm

Definition at line [399](#) of file `vl53l0x_def.h`.

`uint16_t`

`VL53L0X_DeviceSpecificParameters_t::SigmaEstEffPulseWidth`

Effective Pulse width for sigma estimate in 1/100th of ns e.g. 900 = 9.0ns

Definition at line [401](#) of file `vl53l0x_def.h`.

`uint16_t`

`VL53L0X_DeviceSpecificParameters_t::SigmaEstEffAmbWidth`

Effective Ambient width for sigma estimate in 1/100th of ns e.g. 500 = 5.0ns

Definition at line [404](#) of file [vl53l0x_def.h](#).

```
uint8_t  
VL53L0X_DeviceSpecificParameters_t::ReadDataFromDeviceDone
```

Definition at line [409](#) of file [vl53l0x_def.h](#).

```
uint8_t VL53L0X_DeviceSpecificParameters_t::ModuleId
```

Definition at line [411](#) of file [vl53l0x_def.h](#).

```
uint8_t VL53L0X_DeviceSpecificParameters_t::Revision
```

Definition at line [412](#) of file [vl53l0x_def.h](#).

```
char  
VL53L0X_DeviceSpecificParameters_t::ProductId[VL53L0X_MAX_
```

Definition at line [413](#) of file [vl53l0x_def.h](#).

```
uint8_t  
VL53L0X_DeviceSpecificParameters_t::ReferenceSpadCount
```

Definition at line [415](#) of file [vl53l0x_def.h](#).

```
uint8_t  
VL53L0X_DeviceSpecificParameters_t::ReferenceSpadType
```

Definition at line [416](#) of file [vl53l0x_def.h](#).

`uint8_t`

`VL53L0X_DeviceSpecificParameters_t::RefSpadsInitialised`

Definition at line [417](#) of file [vl53l0x_def.h](#).

`uint32_t VL53L0X_DeviceSpecificParameters_t::PartUIDUpper`

Unique Part ID Upper

Definition at line [418](#) of file [vl53l0x_def.h](#).

`uint32_t VL53L0X_DeviceSpecificParameters_t::PartUIDLower`

Unique Part ID Lower

Definition at line [419](#) of file [vl53l0x_def.h](#).

`FixPoint1616_t`

`VL53L0X_DeviceSpecificParameters_t::SignalRateMeasFixed400m`

Peek Signal rate at 400 mm

Definition at line [420](#) of file [vl53l0x_def.h](#).

The documentation for this struct was generated from the following file:

- [vl53l0x_def.h](#)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	Data Fields

VL53L0X_DevData_t Struct Reference

[VL53L0X Defines](#)

VL53L0X PAL device ST private data structure
End user should never access any of these field directly. [More...](#)

```
#include <vl53l0x_def.h>
```

Data Fields

VL53L0X_DMaxData_t DMaxData

int32_t Part2PartOffsetNVMMMicr

int32_t Part2PartOffsetAdjustme

VL53L0X_DeviceParameters_t CurrentParameters

VL53L0X_RangingMeasurementData_t LastRangeMeasure

VL53L0X_HistogramMeasurementData_t LastHistogramMeasure

VL53L0X_DeviceSpecificParameters_t DeviceSpecificParamete

VL53L0X_SpadData_t SpadData

uint8_t SequenceConfig

uint8_t RangeFractionalEnable

VL53L0X_State PalState

VL53L0X_PowerModes PowerMode

uint16_t SigmaEstRefArray

uint16_t SigmaEstEffPulseWidth

uint16_t SigmaEstEffAmbWidth

uint8_t StopVariable

uint16_t targetRefRate

FixPoint1616_t SigmaEstimate

FixPoint1616_t SignalEstimate

FixPoint1616_t LastSignalRefMcps

uint8_t * pTuningSettingsPointer

uint8_t UseInternalTuningSetting

uint16_t LinearityCorrectiveGain

uint16_t DmaxCalRangeMilliMete

FixPoint1616_t DmaxCalSignalRateRtnM

Detailed Description

VL53L0X PAL device ST private data structure
End user should never access any of these field directly.

These must never access directly but only via macro

Definition at line [433](#) of file [vl53l0x_def.h](#).

Field Documentation

VL53L0X_DMaxData_t VL53L0X_DevData_t::DMaxData

Dmax Data

Definition at line [434](#) of file [vl53l0x_def.h](#).

int32_t VL53L0X_DevData_t::Part2PartOffsetNVMMicroMeter

backed up NVM value

Definition at line [436](#) of file [vl53l0x_def.h](#).

**int32_t
VL53L0X_DevData_t::Part2PartOffsetAdjustmentNVMMicroMeter**

backed up NVM value representing additional offset adjustment

Definition at line [438](#) of file [vl53l0x_def.h](#).

**VL53L0X_DeviceParameters_t
VL53L0X_DevData_t::CurrentParameters**

Current Device Parameter

Definition at line [440](#) of file [vl53l0x_def.h](#).

**VL53L0X_RangingMeasurementData_t
VL53L0X_DevData_t::LastRangeMeasure**

Ranging Data

Definition at line [442](#) of file [vl53l0x_def.h](#).

VL53L0X_HistogramMeasurementData_t
VL53L0X_DevData_t::LastHistogramMeasure

Histogram Data

Definition at line [444](#) of file [vl53l0x_def.h](#).

VL53L0X_DeviceSpecificParameters_t
VL53L0X_DevData_t::DeviceSpecificParameters

Parameters specific to the device

Definition at line [446](#) of file [vl53l0x_def.h](#).

VL53L0X_SpadData_t VL53L0X_DevData_t::SpadData

Spad Data

Definition at line [448](#) of file [vl53l0x_def.h](#).

uint8_t VL53L0X_DevData_t::SequenceConfig

Internal value for the sequence config

Definition at line [450](#) of file [vl53l0x_def.h](#).

uint8_t VL53L0X_DevData_t::RangeFractionalEnable

Enable/Disable fractional part of ranging data

Definition at line [452](#) of file [vl53l0x_def.h](#).

VL53L0X_State VL53L0X_DevData_t::PalState

Current state of the PAL for this device

Definition at line [454](#) of file [vl53l0x_def.h](#).

VL53L0X_PowerModes VL53L0X_DevData_t::PowerMode

Current Power Mode

Definition at line [456](#) of file [vl53l0x_def.h](#).

uint16_t VL53L0X_DevData_t::SigmaEstRefArray

Reference array sigma value in 1/100th of [mm] e.g. 100 = 1mm

Definition at line [458](#) of file [vl53l0x_def.h](#).

uint16_t VL53L0X_DevData_t::SigmaEstEffPulseWidth

Effective Pulse width for sigma estimate in 1/100th of ns e.g. 900 = 9.0ns

Definition at line [460](#) of file [vl53l0x_def.h](#).

uint16_t VL53L0X_DevData_t::SigmaEstEffAmbWidth

Effective Ambient width for sigma estimate in 1/100th of ns e.g. 500 = 5.0ns

Definition at line [463](#) of file [vl53l0x_def.h](#).

uint8_t VL53L0X_DevData_t::StopVariable

StopVariable used during the stop sequence

Definition at line [466](#) of file [vl53l0x_def.h](#).

uint16_t VL53L0X_DevData_t::targetRefRate

Target Ambient Rate for Ref spad management

Definition at line [468](#) of file [vl53l0x_def.h](#).

FixPoint1616_t VL53L0X_DevData_t::SigmaEstimate

Sigma Estimate - based on ambient & VCSEL rates and signal_total_events

Definition at line [470](#) of file [vl53l0x_def.h](#).

FixPoint1616_t VL53L0X_DevData_t::SignalEstimate

Signal Estimate - based on ambient & VCSEL rates and cross talk

Definition at line [473](#) of file [vl53l0x_def.h](#).

FixPoint1616_t VL53L0X_DevData_t::LastSignalRefMcps

Latest Signal ref in Mcps

Definition at line [475](#) of file [vl53l0x_def.h](#).

uint8_t* VL53L0X_DevData_t::pTuningSettingsPointer

Pointer for Tuning Settings table

Definition at line [477](#) of file [vl53l0x_def.h](#).

```
uint8_t VL53L0X_DevData_t::UseInternalTuningSettings
```

Indicate if we use Tuning Settings table

Definition at line [479](#) of file [vl53l0x_def.h](#).

```
uint16_t VL53L0X_DevData_t::LinearityCorrectiveGain
```

Linearity Corrective Gain value in x1000

Definition at line [481](#) of file [vl53l0x_def.h](#).

```
uint16_t VL53L0X_DevData_t::DmaxCalRangeMilliMeter
```

Dmax Calibration Range millimeter

Definition at line [483](#) of file [vl53l0x_def.h](#).

```
FixPoint1616_t
```

```
VL53L0X_DevData_t::DmaxCalSignalRateRtnMegaCps
```

Dmax Calibration Signal Rate Return MegaCps

Definition at line [485](#) of file [vl53l0x_def.h](#).

The documentation for this struct was generated from the following file:

- [vl53l0x_def.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	

VL53L0X_RangeData_t Struct Reference

[VL53L0X Defines](#)

Range measurement data. More...

```
#include <vl53l0x_def.h>
```

Detailed Description

Range measurement data.

The documentation for this struct was generated from the following file:

- **vl53l0x_def.h**
-

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	

VL53L0X_HistogramData_t Struct Reference

[VL53L0X Defines](#)

Histogram measurement data. [More...](#)

```
#include <vl53l0x_def.h>
```

Detailed Description

Histogram measurement data.

The documentation for this struct was generated from the following file:

- **vl53l0x_def.h**
-

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Modules

VL53L0X cut1.1 Device Specific Defines

Device specific defines. [More...](#)

Modules

Device Error

Device Error code.

Check Enable list

Check Enable code.

Gpio Functionality

Defines the different functionalities for the device GPIO(s)

Define Registers

List of all the defined registers.

Detailed Description

Device specific defines.

To be adapted by implementer for the targeted device. VL53L0X cut1.1
Device Specific Defines

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[Macros](#) | [Typedefs](#)

Device Error

[VL53L0X cut1.1 Device Specific Defines](#)

Device Error code. [More...](#)

Macros

```
#define VL53L0X_DEVICEERROR_NONE ((VL53L0X_DeviceError)

#define VL53L0X_DEVICEERROR_VCSELCONTINUITYTESTFAILU
1)

#define VL53L0X_DEVICEERROR_VCSELWATCHDOGTESTFAILU
2)

#define VL53L0X_DEVICEERROR_NOVHVVALUEFOUND ((VL53L

#define VL53L0X_DEVICEERROR_MSRCNOTARGET ((VL53L0X_

#define VL53L0X_DEVICEERROR_SNRCHECK ((VL53L0X_Device

#define VL53L0X_DEVICEERROR_RANGEPHASECHECK ((VL53L0X

#define VL53L0X_DEVICEERROR_SIGMATHRESHOLDCHECK ((\

#define VL53L0X_DEVICEERROR_TCC ((VL53L0X_DeviceError) {

#define VL53L0X_DEVICEERROR_PHASECONSISTENCY ((VL53L0X

#define VL53L0X_DEVICEERROR_MINCLIP ((VL53L0X_DeviceEri

#define VL53L0X_DEVICEERROR_RANGECOMPLETE ((VL53L0X

#define VL53L0X_DEVICEERROR_ALGOUNDERFLOW ((VL53L0X

#define VL53L0X_DEVICEERROR_ALGOOVERFLOW ((VL53L0X

#define VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD (
```

Typedefs

```
typedef uint8_t VL53L0X_DeviceError
```

Detailed Description

Device Error code.

This enum is Device specific it should be updated in the implementation Use *VL53L0X_GetStatusErrorString()* to get the string. It is related to Status Register of the Device.

Macro Definition Documentation

```
#define  
VL53L0X_DEVICEERROR_NONE ((VL53L0X_DeviceError) 0)
```

0 NoError

Definition at line [56](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_VCSELCONTINUITYTESTFAILURE ((VL  
1)
```

Definition at line [58](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_VCSELWATCHDOGTESTFAILURE ((VL  
2)
```

Definition at line [59](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_NOVHVVALUEFOUND ((VL53L0X_Device  
3)
```

Definition at line [60](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_MSRCNOTARGET ((VL53L0X_DeviceE  
4)
```

Definition at line 61 of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_SNRCHECK ((VL53L0X_DeviceError)  
5)
```

Definition at line 62 of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_RANGEPHASECHECK ((VL53L0X_DeviceError)  
6)
```

Definition at line 63 of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_SIGMATHRESHOLDCHECK ((VL53L0X_DeviceError)  
7)
```

Definition at line 64 of file `vl53l0x_device.h`.

```
#define VL53L0X_DEVICEERROR_TCC ((VL53L0X_DeviceError)  
8)
```

Definition at line 65 of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_PHASECONSISTENCY ((VL53L0X_DeviceError)  
9)
```

Definition at line 66 of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_MINCLIP ((VL53L0X_DeviceError) 10)
```

Definition at line [67](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_RANGECOMPLETE ((VL53L0X_DeviceE  
11)
```

Definition at line [68](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_ALGOUNDERFLOW ((VL53L0X_DeviceE  
12)
```

Definition at line [69](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_ALGOOVERFLOW ((VL53L0X_DeviceE  
13)
```

Definition at line [70](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD ((VL53L0  
14)
```

Definition at line [71](#) of file `vl53l0x_device.h`.

Typedef Documentation

```
typedef uint8_t VL53L0X_DeviceError
```

Definition at line **54** of file **vl53l0x_device.h**.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Macros

Check Enable list

VL53L0X cut1.1 Device Specific Defines

Check Enable code. More...

Macros

```
#define VL53L0X_CHECKENABLE_SIGMA_FINAL_RANGE 0  
  
#define VL53L0X_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE  
  
#define VL53L0X_CHECKENABLE_SIGNAL_REF_CLIP 2  
  
#define VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD  
  
#define VL53L0X_CHECKENABLE_SIGNAL_RATE_MSRC 4  
  
#define VL53L0X_CHECKENABLE_SIGNAL_RATE_PRE_RANGE  
  
#define VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS 6
```

Detailed Description

Check Enable code.

Define used to specify the LimitCheckId. Use
VL53L0X_GetLimitCheckInfo() to get the string.

Macro Definition Documentation

```
#define VL53L0X_CHECKENABLE_SIGMA_FINAL_RANGE 0
```

Definition at line [84](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE 1
```

Definition at line [85](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_CHECKENABLE_SIGNAL_REF_CLIP 2
```

Definition at line [86](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD 3
```

Definition at line [87](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_CHECKENABLE_SIGNAL_RATE_MSRC 4
```

Definition at line [88](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_CHECKENABLE_SIGNAL_RATE_PRE_RANGE 5
```

Definition at line [89](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS 6
```

Definition at line [91](#) of file [vl53l0x_device.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

[Macros](#) | [Typedefs](#)

Gpio Functionality

[VL53L0X cut1.1 Device Specific Defines](#)

Defines the different functionalities for the device GPIO(s) [More...](#)

Macros

```
#define VL53L0X_GPIOFUNCTIONALITY_OFF ((VL53L0X_GpioFu  
  
#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED  
1)  
  
#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED  
2)  
  
#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED  
3)  
  
#define VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY
```

Typedefs

```
typedef uint8_t VL53L0X_GpioFunctionality
```

Detailed Description

Defines the different functionalities for the device GPIO(s)

Macro Definition Documentation

```
#define  
VL53L0X_GPIOFUNCTIONALITY_OFF ((VL53L0X_GpioFunctional  
0)
```

NO Interrupt

Definition at line [102](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_LOW  
1)
```

Level Low (value < thresh_low)

Definition at line [104](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HIGH  
2)
```

Level High (value > thresh_high)

Definition at line [106](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OUT (3)
```

Out Of Window (value < thresh_low OR value > thresh_high)

Definition at line [108](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY ((VL53L0X_GPIOFUNCTIONALITY  
4))
```

New Sample Ready

Definition at line [111](#) of file `vl53l0x_device.h`.

Typedef Documentation

```
typedef uint8_t VL53L0X_GpioFunctionality
```

Definition at line [100](#) of file [vl53l0x_device.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page
Files

Related Pages

Modules

Data Structures

Macros

Define Registers

VL53L0X cut1.1 Device Specific Defines

List of all the defined registers. [More...](#)

Macros

```
#define VL53L0X_REG_SYSRANGE_START 0x000
```

```
#define VL53L0X_REG_SYSRANGE_MODE_MASK 0x0F  
mask existing bit in VL53L0X_REG_SYSRANGE_START More...
```

```
#define VL53L0X_REG_SYSRANGE_MODE_START_STOP 0x01  
bit 0 in VL53L0X_REG_SYSRANGE_START write 1 toggle start  
and arm next shot in single shot mode More...
```

```
#define VL53L0X_REG_SYSRANGE_MODE_SINGLESHTOT 0x00  
bit 1 write 0 in VL53L0X_REG_SYSRANGE_START set single shot mode More...
```

```
#define VL53L0X_REG_SYSRANGE_MODE_BACKTOBACK 0x02  
bit 1 write 1 in VL53L0X_REG_SYSRANGE_START set back-to-back mode More...
```

```
#define VL53L0X_REG_SYSRANGE_MODE_TIMED 0x04  
bit 2 write 1 in VL53L0X_REG_SYSRANGE_START set timed mode More...
```

```
#define VL53L0X_REG_SYSRANGE_MODE_HISTOGRAM 0x08  
bit 3 write 1 in VL53L0X_REG_SYSRANGE_START set histogram mode More...
```

```
#define VL53L0X_REG_SYSTEM_THRESH_HIGH 0x000C
```

```
#define VL53L0X_REG_SYSTEM_THRESH_LOW 0x000E
```

```
#define VL53L0X_REG_SYSTEM_SEQUENCE_CONFIG 0x0001
```

```
#define VL53L0X_REG_SYSTEM_RANGE_CONFIG 0x0009
```

```
#define VL53L0X_REG_SYSTEM_INTERMEASUREMENT_PERIOD
```

```
#define VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO 0x00000000
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED 0x00000000
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW 0x00000000
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH 0x00000001
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW 0x00000000
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_READY 0x00000000
#define VL53L0X_REG_GPIO_HV_MUX_ACTIVE_HIGH 0x0084
#define VL53L0X_REG_SYSTEM_INTERRUPT_CLEAR 0x000B
#define VL53L0X_REG_RESULT_INTERRUPT_STATUS 0x0013
#define VL53L0X_REG_RESULT_RANGE_STATUS 0x0014
#define VL53L0X_REG_RESULT_CORE_PAGE 1
#define VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS 0x00000000
#define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS 0x00000000
#define VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS 0x00000000
#define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS 0x00000000
#define VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF 0x00000000
#define VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET 0x00000000
#define VL53L0X_REG_I2C_SLAVE_DEVICE_ADDRESS 0x008a
#define VL53L0X_REG_MSRC_CONFIG_CONTROL 0x0060
```

```
#define VL53L0X_REG_PRE_RANGE_CONFIG_MIN_SNR 0x0027

#define VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_LO

#define VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HI

#define VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LI

#define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR 0x0060

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_L

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_H

#define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RA

#define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_I

#define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_L

#define VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD 0x0000

#define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACRO

#define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACRO

#define VL53L0X_REG_SYSTEM_HISTOGRAM_BIN 0x0081

#define VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SI

#define VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VCSEL_PERIOD

#define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACR
```

```
#define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACR

#define VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RA

#define VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP 0x00c0

#define VL53L0X_REG_SOFT_RESET_G02_SOFT_RESET_N 0x00c1

#define VL53L0X_REG_IDENTIFICATION_MODEL_ID 0x00c0

#define VL53L0X_REG_IDENTIFICATION_REVISION_ID 0x00c2

#define VL53L0X_REG_OSC_CALIBRATE_VAL 0x00f8

#define VL53L0X_SIGMA_ESTIMATE_MAX_VALUE 65535

#define VL53L0X_REG_GLOBAL_CONFIG_VSEL_WIDTH 0x032

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_0

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_6

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_7

#define VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELE

#define VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF */

#define VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET
```

```
#define VL53L0X_REG_POWER_MANAGEMENT_GO1_POWER_F0 0x00000000  
#define VL53L0X_SPEED_OF_LIGHT_IN_AIR 2997  
#define VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA__EXTSUP__VHV 0x00000000  
#define VL53L0X_REG_ALGO_PHASECAL_LIM 0x0030 /* 0x130 */  
#define VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT 0x00000000
```

Detailed Description

List of all the defined registers.

Macro Definition Documentation

```
#define VL53L0X_REG_SYSRANGE_START 0x000
```

Definition at line [123](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSRANGE_MODE_MASK 0x0F
```

mask existing bit in `VL53L0X_REG_SYSRANGE_START`

Definition at line [125](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSRANGE_MODE_START_STOP 0x01
```

bit 0 in `VL53L0X_REG_SYSRANGE_START` write 1 toggle state in continuous mode and arm next shot in single shot mode

Definition at line [128](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSRANGE_MODE_SINGLESHT 0x00
```

bit 1 write 0 in `VL53L0X_REG_SYSRANGE_START` set single shot mode

Definition at line [130](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_SYSRANGE_MODE_BACKTOBACK 0x02
```

bit 1 write 1 in `VL53L0X_REG_SYSRANGE_START` set back-to-

back operation mode

Definition at line [133](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSRANGE_MODE_TIMED 0x04
```

bit 2 write 1 in `VL53L0X_REG_SYSRANGE_START` set timed operation mode

Definition at line [136](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSRANGE_MODE_HISTOGRAM 0x08
```

bit 3 write 1 in `VL53L0X_REG_SYSRANGE_START` set histogram operation mode

Definition at line [139](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSTEM_THRESH_HIGH 0x000C
```

Definition at line [142](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSTEM_THRESH_LOW 0x000E
```

Definition at line [143](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSTEM_SEQUENCE_CONFIG 0x0001
```

Definition at line [146](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSTEM_RANGE_CONFIG 0x0009
```

Definition at line [147](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_SYSTEM_INTERMEASUREMENT_PERIOD 0x0004
```

Definition at line [148](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO 0x000A
```

Definition at line [151](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED 0x00
```

Definition at line [152](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW 0x01
```

Definition at line [153](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH 0x02
```

Definition at line [154](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW 0
```

Definition at line [155](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_READY 0x0080
```

Definition at line [156](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_GPIO_HV_MUX_ACTIVE_HIGH 0x0084
```

Definition at line [158](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSTEM_INTERRUPT_CLEAR 0x000B
```

Definition at line [161](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_RESULT_INTERRUPT_STATUS 0x0013
```

Definition at line [164](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_RESULT_RANGE_STATUS 0x0014
```

Definition at line [165](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_RESULT_CORE_PAGE 1
```

Definition at line [167](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_RATE 0x0015
```

Definition at line [168](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_RTN
```

Definition at line [169](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_RE
```

Definition at line [170](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_REF
```

Definition at line [171](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF 0x00B6
```

Definition at line [172](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET_MM 0x
```

Definition at line [176](#) of file [vl53l0x_device.h](#).

```
#define VL53L0X_REG_I2C_SLAVE_DEVICE_ADDRESS 0x008a
```

Definition at line [178](#) of file [vl53l0x_device.h](#).

```
#define VL53L0X_REG_MSRC_CONFIG_CONTROL 0x0060
```

Definition at line [181](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_CONFIG_MIN_SNR 0X0027
```

Definition at line [183](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_LOW 0x00
```

Definition at line [184](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HIGH 0x00
```

Definition at line [185](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LIMIT 0x00
```

Definition at line [186](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR 0X0067
```

Definition at line [188](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_LOW 0x00
```

Definition at line [189](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_HIGH 0x
```

Definition at line [190](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RATE_RTNL
```

Definition at line [191](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_HI 0X00
```

Definition at line [194](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_LO 0X00
```

Definition at line [195](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD 0x0050
```

Definition at line [198](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_HI 0x
```

Definition at line [199](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_LO 0
```

Definition at line [200](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_REG_SYSTEM_HISTOGRAM_BIN 0x0081
```

Definition at line [202](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SELECT
```

Definition at line [203](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL 0x0055
```

Definition at line [204](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_FINAL_RANGE_CONFIG_VCSEL_PERIOD 0x0070
```

Definition at line [206](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_HI
```

Definition at line [207](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_LO
```

Definition at line [208](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RATE_MCP
```

Definition at line [209](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP 0x0046
```

Definition at line [211](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_SOFT_RESET_GO2_SOFT_RESET_N 0x00bf
```

Definition at line [214](#) of file [vl53l0x_device.h](#).

```
#define VL53L0X_REG_IDENTIFICATION_MODEL_ID 0x00c0
```

Definition at line [215](#) of file [vl53l0x_device.h](#).

```
#define VL53L0X_REG_IDENTIFICATION_REVISION_ID 0x00c2
```

Definition at line [216](#) of file [vl53l0x_device.h](#).

```
#define VL53L0X_REG_OSC_CALIBRATE_VAL 0x00f8
```

Definition at line [218](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_SIGMA_ESTIMATE_MAX_VALUE 65535
```

Definition at line [221](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_GLOBAL_CONFIG_VCSEL_WIDTH 0x032
```

Definition at line [224](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_0 0x0E
```

Definition at line [225](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1 0x0E
```

Definition at line [226](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2 0x0E
```

Definition at line [227](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3 0x0E
```

Definition at line [228](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4 0x0E
```

Definition at line [229](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5 0x0E
```

Definition at line [230](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELECT 0xE
```

Definition at line [232](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF_SPAD  
/* 0x14E */
```

Definition at line [233](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET 0x4F  
/* 0x14F */
```

Definition at line [234](#) of file `vl53l0x_device.h`.

```
#define  
VL53L0X_REG_POWER_MANAGEMENT_GO1_POWER_FORCE 0
```

Definition at line [235](#) of file `vl53l0x_device.h`.

```
#define VL53L0X_SPEED_OF_LIGHT_IN_AIR 2997
```

Definition at line [241](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA__EXTSUP_HV 0x0
```

Definition at line [243](#) of file [vl53l0x_device.h](#).

```
#define VL53L0X_REG_ALGO_PHASECAL_LIM 0x0030 /* 0x130  
*/
```

Definition at line [245](#) of file [vl53l0x_device.h](#).

```
#define  
VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT 0x0030
```

Definition at line [246](#) of file [vl53l0x_device.h](#).



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	

Data Structures

Here are the data structures with brief descriptions:

 VL53L0X_Dev_t	Generic PAL device type that does link between API and platform abstraction layer
 VL53L0X_DevData_t	VL53L0X PAL device ST private data structure End user should never access any of these field directly
 VL53L0X_DeviceInfo_t	Defines the parameters of the Get Device Info Functions
 VL53L0X_DeviceParameters_t	Defines all parameters for the device
 VL53L0X_DeviceSpecificParameters_t	
 VL53L0X_DMaxData_t	Structure containing the Dmax

		computation parameters and data
VL53L0X_HistogramData_t		Histogram measurement data
VL53L0X_HistogramMeasurementData_t		
VL53L0X_RangeData_t		Range measurement data
VL53L0X_RangingMeasurementData_t		
VL53L0X_SchedulerSequenceSteps_t		
VL53L0X_SpadData_t		Spad Configuration Data
VL53L0X_Version_t		Defines the parameters of the Get Version Functions

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	

Data Structure Index

V

V VL53L0X_DevData_t VL53L0X_Device VL53L0X_DeviceInfo_t VL53L0X_DeviceParameters_t VL53L0X_Dev_t VL53L0X_Device

V

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures	Data Structure Index	Data Fields	
All	Variables		
a b	c d e f h i l m n o p r s t u w	x	z

Here is a list of all struct and union fields with links to the structures/unions they belong to:

- a -

- AmbientRateRtnMegaCps : [VL53L0X_RangingMeasurementData_t](#)
- AmbTuningWindowFactor_K : [VL53L0X_DMaxData_t](#)

- b -

- BufferSize : [VL53L0X_HistogramMeasurementData_t](#)
- build : [VL53L0X_Version_t](#)

- c -

- comms_speed_khz : [VL53L0X_Dev_t](#)
- comms_type : [VL53L0X_Dev_t](#)
- CurrentParameters : [VL53L0X_DevData_t](#)

- d -

- Data : [VL53L0X_Dev_t](#)
- DeviceMode : [VL53L0X_DeviceParameters_t](#)

- DeviceSpecificParameters : **VL53L0X_DevData_t**
- DmaxCalRangeMilliMeter : **VL53L0X_DevData_t**
- DmaxCalSignalRateRtnMegaCps : **VL53L0X_DevData_t**
- DMaxData : **VL53L0X_DevData_t**
- DssOn : **VL53L0X_SchedulerSequenceSteps_t**

- e -

- EffectiveSpadRtnCount : **VL53L0X_RangingMeasurementData_t**
- ErrorStatus : **VL53L0X_HistogramMeasurementData_t**

- f -

- FinalRangeOn : **VL53L0X_SchedulerSequenceSteps_t**
- FinalRangeTimeoutMicroSecs :
VL53L0X_DeviceSpecificParameters_t
- FinalRangeVcselPulsePeriod :
VL53L0X_DeviceSpecificParameters_t
- FirstBin : **VL53L0X_HistogramMeasurementData_t**

- h -

- HistogramData : **VL53L0X_HistogramMeasurementData_t**
- HistogramMode : **VL53L0X_DeviceParameters_t**
- HistogramType : **VL53L0X_HistogramMeasurementData_t**

- i -

- I2cDevAddr : **VL53L0X_Dev_t**
- InterMeasurementPeriodMilliseconds :
VL53L0X_DeviceParameters_t

- l -

- LastEncodedTimeout : **VL53L0X_DeviceSpecificParameters_t**
- LastHistogramMeasure : **VL53L0X_DevData_t**
- LastRangeMeasure : **VL53L0X_DevData_t**
- LastSignalRefMcps : **VL53L0X_DevData_t**
- LimitChecksEnable : **VL53L0X_DeviceParameters_t**

- LimitChecksStatus : **VL53L0X_DeviceParameters_t**
- LimitChecksValue : **VL53L0X_DeviceParameters_t**
- LinearityCorrectiveGain : **VL53L0X_DevData_t**

- m -

- major : **VL53L0X_Version_t**
- MeasurementTimeUsec :
VL53L0X_RangingMeasurementData_t
- MeasurementTimingBudgetMicroSeconds :
VL53L0X_DeviceParameters_t
- minor : **VL53L0X_Version_t**
- ModuleId : **VL53L0X_DeviceSpecificParameters_t**
- MsrcOn : **VL53L0X_SchedulerSequenceSteps_t**

- n -

- Name : **VL53L0X_DeviceInfo_t**
- NumberOfBins : **VL53L0X_HistogramMeasurementData_t**

- o -

- OscFrequencyMHz : **VL53L0X_DeviceSpecificParameters_t**

- p -

- PalState : **VL53L0X_DevData_t**
- Part2PartOffsetAdjustmentNVMMicroMeter :
VL53L0X_DevData_t
- Part2PartOffsetNVMMicroMeter : **VL53L0X_DevData_t**
- PartUIDLower : **VL53L0X_DeviceSpecificParameters_t**
- PartUIDUpper : **VL53L0X_DeviceSpecificParameters_t**
- Pin0GpioFunctionality : **VL53L0X_DeviceSpecificParameters_t**
- PowerMode : **VL53L0X_DevData_t**
- PreRangeOn : **VL53L0X_SchedulerSequenceSteps_t**
- PreRangeTimeoutMicroSecs :
VL53L0X_DeviceSpecificParameters_t
- PreRangeVcselPulsePeriod :
VL53L0X_DeviceSpecificParameters_t

- ProductId : **VL53L0X_DeviceInfo_t** ,
VL53L0X_DeviceSpecificParameters_t
- ProductRevisionMajor : **VL53L0X_DeviceInfo_t**
- ProductRevisionMinor : **VL53L0X_DeviceInfo_t**
- ProductType : **VL53L0X_DeviceInfo_t**
- pTuningSettingsPointer : **VL53L0X_DevData_t**

- r -

- RangeDMaxMilliMeter : **VL53L0X_RangingMeasurementData_t**
- RangeFractionalEnable : **VL53L0X_DevData_t**
- RangeFractionalPart : **VL53L0X_RangingMeasurementData_t**
- RangeMilliMeter : **VL53L0X_RangingMeasurementData_t**
- RangeOffsetMicroMeters : **VL53L0X_DeviceParameters_t**
- RangeStatus : **VL53L0X_RangingMeasurementData_t**
- ReadDataFromDeviceDone :
VL53L0X_DeviceSpecificParameters_t
- ReferenceSpadCount : **VL53L0X_DeviceSpecificParameters_t**
- ReferenceSpadType : **VL53L0X_DeviceSpecificParameters_t**
- RefGoodSpadMap : **VL53L0X_SpadData_t**
- RefSpadEnables : **VL53L0X_SpadData_t**
- RefSpadsInitialised : **VL53L0X_DeviceSpecificParameters_t**
- RetSignalAt0mm : **VL53L0X_DMaxData_t**
- Revision : **VL53L0X_DeviceSpecificParameters_t**
- revision : **VL53L0X_Version_t**

- S -

- SequenceConfig : **VL53L0X_DevData_t**
- SigmaEstEffAmbWidth : **VL53L0X_DevData_t** ,
VL53L0X_DeviceSpecificParameters_t
- SigmaEstEffPulseWidth : **VL53L0X_DevData_t** ,
VL53L0X_DeviceSpecificParameters_t
- SigmaEstimate : **VL53L0X_DevData_t**
- SigmaEstRefArray : **VL53L0X_DevData_t** ,
VL53L0X_DeviceSpecificParameters_t
- SignalEstimate : **VL53L0X_DevData_t**
- SignalRateMeasFixed400mm :
VL53L0X_DeviceSpecificParameters_t
- SignalRateRtnMegaCps :

VL53L0X_RangingMeasurementData_t

- SpadData : **VL53L0X_DevData_t**
- StopVariable : **VL53L0X_DevData_t**

- t -

- targetRefRate : **VL53L0X_DevData_t**
- TccOn : **VL53L0X_SchedulerSequenceSteps_t**
- TimeStamp : **VL53L0X_RangingMeasurementData_t**
- Type : **VL53L0X_DeviceInfo_t**

- u -

- UseInternalTuningSettings : **VL53L0X_DevData_t**

- w -

- WrapAroundCheckEnable : **VL53L0X_DeviceParameters_t**

- x -

- XTalkCompensationEnable : **VL53L0X_DeviceParameters_t**
- XTalkCompensationRangeMilliMeter :
VL53L0X_DeviceParameters_t
- XTalkCompensationRateMegaCps :
VL53L0X_DeviceParameters_t

- z -

- ZonelId : **VL53L0X_RangingMeasurementData_t**



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
Data Structures		Data Structure Index	Data Fields
All	Variables		
a b	c d e f h i l m n o p r s t u w	x	z

- a -

- AmbientRateRtnMegaCps : [VL53L0X_RangingMeasurementData_t](#)
- AmbTuningWindowFactor_K : [VL53L0X_DMaxData_t](#)

- b -

- BufferSize : [VL53L0X_HistogramMeasurementData_t](#)
- build : [VL53L0X_Version_t](#)

- c -

- comms_speed_khz : [VL53L0X_Dev_t](#)
- comms_type : [VL53L0X_Dev_t](#)
- CurrentParameters : [VL53L0X_DevData_t](#)

- d -

- Data : [VL53L0X_Dev_t](#)
- DeviceMode : [VL53L0X_DeviceParameters_t](#)
- DeviceSpecificParameters : [VL53L0X_DevData_t](#)

- DmaxCalRangeMilliMeter : **VL53L0X_DevData_t**
- DmaxCalSignalRateRtnMegaCps : **VL53L0X_DevData_t**
- DMaxData : **VL53L0X_DevData_t**
- DssOn : **VL53L0X_SchedulerSequenceSteps_t**

- e -

- EffectiveSpadRtnCount : **VL53L0X_RangingMeasurementData_t**
- ErrorStatus : **VL53L0X_HistogramMeasurementData_t**

- f -

- FinalRangeOn : **VL53L0X_SchedulerSequenceSteps_t**
- FinalRangeTimeoutMicroSecs :
VL53L0X_DeviceSpecificParameters_t
- FinalRangeVcselPulsePeriod :
VL53L0X_DeviceSpecificParameters_t
- FirstBin : **VL53L0X_HistogramMeasurementData_t**

- h -

- HistogramData : **VL53L0X_HistogramMeasurementData_t**
- HistogramMode : **VL53L0X_DeviceParameters_t**
- HistogramType : **VL53L0X_HistogramMeasurementData_t**

- i -

- I2cDevAddr : **VL53L0X_Dev_t**
- InterMeasurementPeriodMilliSeconds :
VL53L0X_DeviceParameters_t

- l -

- LastEncodedTimeout : **VL53L0X_DeviceSpecificParameters_t**
- LastHistogramMeasure : **VL53L0X_DevData_t**
- LastRangeMeasure : **VL53L0X_DevData_t**
- LastSignalRefMcps : **VL53L0X_DevData_t**
- LimitChecksEnable : **VL53L0X_DeviceParameters_t**
- LimitChecksStatus : **VL53L0X_DeviceParameters_t**

- LimitChecksValue : **VL53L0X_DeviceParameters_t**
- LinearityCorrectiveGain : **VL53L0X_DevData_t**

- m -

- major : **VL53L0X_Version_t**
- MeasurementTimeUsec :
VL53L0X_RangingMeasurementData_t
- MeasurementTimingBudgetMicroSeconds :
VL53L0X_DeviceParameters_t
- minor : **VL53L0X_Version_t**
- ModuleId : **VL53L0X_DeviceSpecificParameters_t**
- MsrcOn : **VL53L0X_SchedulerSequenceSteps_t**

- n -

- Name : **VL53L0X_DeviceInfo_t**
- NumberOfBins : **VL53L0X_HistogramMeasurementData_t**

- o -

- OscFrequencyMHz : **VL53L0X_DeviceSpecificParameters_t**

- p -

- PalState : **VL53L0X_DevData_t**
- Part2PartOffsetAdjustmentNVMMicroMeter :
VL53L0X_DevData_t
- Part2PartOffsetNVMMicroMeter : **VL53L0X_DevData_t**
- PartUIDLower : **VL53L0X_DeviceSpecificParameters_t**
- PartUIDUpper : **VL53L0X_DeviceSpecificParameters_t**
- Pin0GpioFunctionality : **VL53L0X_DeviceSpecificParameters_t**
- PowerMode : **VL53L0X_DevData_t**
- PreRangeOn : **VL53L0X_SchedulerSequenceSteps_t**
- PreRangeTimeoutMicroSecs :
VL53L0X_DeviceSpecificParameters_t
- PreRangeVcselPulsePeriod :
VL53L0X_DeviceSpecificParameters_t
- ProductId : **VL53L0X_DeviceInfo_t**,

VL53L0X_DeviceSpecificParameters_t

- ProductRevisionMajor : **VL53L0X_DeviceInfo_t**
- ProductRevisionMinor : **VL53L0X_DeviceInfo_t**
- ProductType : **VL53L0X_DeviceInfo_t**
- pTuningSettingsPointer : **VL53L0X_DevData_t**

- r -

- RangeDMaxMilliMeter : **VL53L0X_RangingMeasurementData_t**
- RangeFractionalEnable : **VL53L0X_DevData_t**
- RangeFractionalPart : **VL53L0X_RangingMeasurementData_t**
- RangeMilliMeter : **VL53L0X_RangingMeasurementData_t**
- RangeOffsetMicroMeters : **VL53L0X_DeviceParameters_t**
- RangeStatus : **VL53L0X_RangingMeasurementData_t**
- ReadDataFromDeviceDone :
VL53L0X_DeviceSpecificParameters_t
- ReferenceSpadCount : **VL53L0X_DeviceSpecificParameters_t**
- ReferenceSpadType : **VL53L0X_DeviceSpecificParameters_t**
- RefGoodSpadMap : **VL53L0X_SpadData_t**
- RefSpadEnables : **VL53L0X_SpadData_t**
- RefSpadsInitialised : **VL53L0X_DeviceSpecificParameters_t**
- RetSignalAt0mm : **VL53L0X_DMaxData_t**
- Revision : **VL53L0X_DeviceSpecificParameters_t**
- revision : **VL53L0X_Version_t**

- s -

- SequenceConfig : **VL53L0X_DevData_t**
- SigmaEstEffAmbWidth : **VL53L0X_DevData_t** ,
VL53L0X_DeviceSpecificParameters_t
- SigmaEstEffPulseWidth : **VL53L0X_DevData_t** ,
VL53L0X_DeviceSpecificParameters_t
- SigmaEstimate : **VL53L0X_DevData_t**
- SigmaEstRefArray : **VL53L0X_DevData_t** ,
VL53L0X_DeviceSpecificParameters_t
- SignalEstimate : **VL53L0X_DevData_t**
- SignalRateMeasFixed400mm :
VL53L0X_DeviceSpecificParameters_t
- SignalRateRtnMegaCps :
VL53L0X_RangingMeasurementData_t

- SpadData : [VL53L0X_DevData_t](#)
- StopVariable : [VL53L0X_DevData_t](#)

- t -

- targetRefRate : [VL53L0X_DevData_t](#)
- TccOn : [VL53L0X_SchedulerSequenceSteps_t](#)
- TimeStamp : [VL53L0X_RangingMeasurementData_t](#)
- Type : [VL53L0X_DeviceInfo_t](#)

- u -

- UseInternalTuningSettings : [VL53L0X_DevData_t](#)

- w -

- WrapAroundCheckEnable : [VL53L0X_DeviceParameters_t](#)

- x -

- XTalkCompensationEnable : [VL53L0X_DeviceParameters_t](#)
- XTalkCompensationRangeMilliMeter :
[VL53L0X_DeviceParameters_t](#)
- XTalkCompensationRateMegaCps :
[VL53L0X_DeviceParameters_t](#)

- z -

- ZonId : [VL53L0X_RangingMeasurementData_t](#)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		

File List

Here is a list of all files with brief descriptions:

 PAL_disclaimer.c	No code doxygen doc only
 vl53l0x_api.h	
 vl53l0x_api_calibration.h	
 vl53l0x_api_core.h	
 vl53l0x_api_ranging.h	
 vl53l0x_api_strings.h	
 vl53l0x_def.h	Type definitions for VL53L0X API
 vl53l0x_device.h	
 vl53l0x_doxydoc.c	
 vl53l0x_i2c_platform.h	
 vl53l0x_interrupt_threshold_settings.h	
 vl53l0x_platform.h	Function prototype definitions for Ewok Platform layer
 vl53l0x_platform_log.h	Platform log function definition
 vl53l0x_tuning.h	
 vl53l0x_types.h	VL53L0X types

definition

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
doc >			

PAL_disclaimer.c File Reference

no code doxygen doc only [More...](#)

[Go to the source code of this file.](#)

Detailed Description

no code doxygen doc only

Definition in file **PAL_disclaimer.c**.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	core inc

Macros | Functions

vl53l0x_api.h File Reference

```
#include "vl53l0x_api_strings.h" #include "vl53l0x_def.h"  
#include "vl53l0x_platform.h"
```

Go to the source code of this file.

Macros

```
#define VL53L0X_API
```

Functions

VL53L0X_API VL53L0X_Error VL53L0X_GetVersion (VL53L0X_Version_t *pVersion)
Return the VL53L0X PAL Implementation Version.

VL53L0X_API VL53L0X_Error VL53L0X_GetPalSpecVersion (VL53L0X_PalSpecVersion_t *pPalSpecVersion)
Return the PAL Specification Version used by the implementation. [More...](#)

VL53L0X_API VL53L0X_Error VL53L0X_GetProductRevision (VL53L0X_ProductRevision_t *pProductRevisionMajor, VL53L0X_ProductRevision_t *pProductRevisionMinor)
Reads the Product Revision for a given function. This function can be used to distinguish cut versions. [More...](#)

VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceInfo (VL53L0X_DeviceInfo_t *pVL53L0X_DeviceInfo)
Reads the Device information for given device.

VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceErrorStatus (VL53L0X_DeviceError_t *pDeviceError)
Read current status of the error register for a given device. [More...](#)

VL53L0X_API VL53L0X_Error VL53L0X_GetRangeStatusString (uint8_t RangeStatus, char *pRangeStatusString)
Human readable Range Status string for a given RangeStatus. [More...](#)

VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceErrorString (VL53L0X_ErrorCode ErrorCode, char *pDeviceErrorString)
Human readable error string for a given ErrorCode. [More...](#)

VL53L0X_API VL53L0X_Error [VL53L0X_GetPalErrorString \(VL53L0X_PalErrorCode, char *pPalErrorString\)](#)
Human readable error string for current error code.
[More...](#)

VL53L0X_API VL53L0X_Error [VL53L0X_GetPalStateString \(VL53L0X_PalStateCode, char *pPalStateString\)](#)
Human readable PAL State string. [More...](#)

VL53L0X_API VL53L0X_Error [VL53L0X_GetPalState \(VL53L0X_Device, VL53L0X_State *pPalState\)](#)
Reads the internal state of the PAL for the specified device.
[More...](#)

VL53L0X_API VL53L0X_Error [VL53L0X_SetPowerMode \(VL53L0X_Device, VL53L0X_PowerModes PowerMode\)](#)
Set the power mode for a given Device. The power mode can be Standby or Idle. [More...](#)

VL53L0X_API VL53L0X_Error [VL53L0X_GetPowerMode \(VL53L0X_Device, VL53L0X_PowerModes *pPowerMode\)](#)
Get the power mode for a given Device.

VL53L0X_API VL53L0X_Error [VL53L0X_SetOffsetCalibrationData \(VL53L0X_DEV Dev, int32_t OffsetCalibrationDataMicroMeter\)](#)
Set or over-hide part to part calibration offset.

VL53L0X_API VL53L0X_Error [VL53L0X_GetOffsetCalibrationData \(VL53L0X_DEV Dev, int32_t *pOffsetCalibrationDataMicroMeter\)](#)
Get part to part calibration offset. [More...](#)

VL53L0X_API VL53L0X_Error [VL53L0X_SetLinearityCorrectiveGain \(VL53L0X_Device, int16_t LinearityCorrectiveGain\)](#)
Set the linearity corrective gain. [More...](#)

VL53L0X_GetLinearityCorrectiveGain

VL53L0X_API VL53L0X_Error Dev, `uint16_t` *pLinearityCorrectiveGain
Get the linearity corrective gain. More...

VL53L0X_API VL53L0X_Error **VL53L0X_SetGroupParamHold** (`VL53L0X_GroupParam`,
`uint8_t` GroupParamHold)
Set Group parameter Hold state. More...

VL53L0X_API VL53L0X_Error **VL53L0X_GetUpperLimitMilliMeter** (`VL53L0X_UpperLimitMilliMeter`,
`uint16_t` *pUpperLimitMilliMeter)
Get the maximal distance for actual sensor. More...

VL53L0X_Error **VL53L0X_GetTotalSignalRate** (`VL53L0X_TotalsignalRate`,
`FixPoint1616_t` *pTotalSignalRate)
Get the Total Signal Rate. More...

VL53L0X_API VL53L0X_Error **VL53L0X_SetDeviceAddress** (`VL53L0X_DeviceAddress`,
`uint8_t` DeviceAddress)
Set new device address. More...

VL53L0X_API VL53L0X_Error **VL53L0X_DatalInit** (`VL53L0X_DEV`,
DataInitParam *pDataInitParam)
One time device initialization. More...

VL53L0X_API VL53L0X_Error **VL53L0X_SetTuningSettingBuffer** (`VL53L0X_TuningSetting`,
`uint8_t` *pTuningSettingBuffer, `uint8_t`*pUseInternalTuningSettings)
Set the tuning settings pointer. More...

VL53L0X_API VL53L0X_Error **VL53L0X_GetTuningSettingBuffer** (`VL53L0X_TuningSetting`,
`uint8_t` **ppTuningSettingBuffer, `uint8_t`*pUseInternalTuningSettings)
Get the tuning settings pointer and the switch value. More...

VL53L0X_API VL53L0X_Error **VL53L0X_StaticInit** (`VL53L0X_DEV`,
StaticInitParam *pStaticInitParam)
Do basic device init (and eventually pause).
This function will change the VL53L0X_Status to
VL53L0X_STATE_WAIT_STATICINIT

VL53L0X_STATE_IDLE. More...

VL53L0X_API VL53L0X_Error	VL53L0X_WaitDeviceBooted (VL53L0X_Error) Wait for device booted after chip enable (from standby) This function can be run only when VL53L0X_State is VL53L0X_STATE_IDLE. More...
----------------------------------	--

VL53L0X_API VL53L0X_Error	VL53L0X_ResetDevice (VL53L0X_Error) Do an hard reset or soft reset (depends on implementation) of the device call of this function must be in same state as right after a powerdown sequence. This function will change the state to VL53L0X_STATE_POWERDOWN. More...
----------------------------------	---

VL53L0X_API VL53L0X_Error	VL53L0X_SetDeviceParameters (VL53L0X_Error) const VL53L0X_DeviceParameters_1_t *pDeviceParameters) Prepare device for operation. More...
----------------------------------	--

VL53L0X_API VL53L0X_Error	VL53L0X_GetDeviceParameters (VL53L0X_Error) VL53L0X_DeviceParameters_t *pDeviceParameters) Retrieve current device parameters. More...
----------------------------------	--

VL53L0X_API VL53L0X_Error	VL53L0X_SetDeviceMode (VL53L0X_Error) VL53L0X_DeviceModes DeviceMode Set a new device mode. More...
----------------------------------	---

VL53L0X_API VL53L0X_Error	VL53L0X_GetDeviceMode (VL53L0X_Error) VL53L0X_DeviceModes *pDeviceMode Get current new device mode. More...
----------------------------------	---

VL53L0X_API VL53L0X_Error	VL53L0X_SetRangeFractionEnable (VL53L0X_Error) uint8_t Enable) Sets the resolution of range measurement.
----------------------------------	--

VL53L0X_API VL53L0X_Error	VL53L0X_GetFractionEnable (VL53L0X_Error) uint8_t *pEnable)
----------------------------------	--

	Gets the fraction enable parameter inc resolution of range measurements. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetHistogramMode (VL53L0X_HistogramModes) Histogram mode. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetHistogramMode (VL53L0X_HistogramModes) *pHistogramMode Get current new device mode. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetMeasurementTimingBudget (VL53L0X_DEV) Dev, uint32_t MeasurementTimingBudgetMicroSeconds Set Ranging Timing Budget in microseconds
VL53L0X_API VL53L0X_Error	VL53L0X_GetMeasurementTimingBudget (VL53L0X_DEV) Dev, uint32_t *pMeasurementTimingBudgetMicroSeconds Get Ranging Timing Budget in microseconds
VL53L0X_API VL53L0X_Error	VL53L0X_GetVcselPulsePeriod (VL53L0X_VcselPeriod) VcselPeriodType *pVCSEL_PulsePeriod) Gets the VCSEL pulse period. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetVcselPulsePeriod (VL53L0X_VcselPeriod) VcselPeriodType VCSEL_PulsePeriod) Sets the VCSEL pulse period. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetSequenceStepEnable (VL53L0X_SequenceStepId) SequenceStepId SequenceStepEnabled) Sets the (on/off) state of a requested sequence step. More...
	VL53L0X_GetSequenceStepEnable

VL53L0X_API VL53L0X_Error **VL53L0X_SequenceStepId** Sequence
*pSequenceStepEnabled)
Gets the (on/off) state of a requested sequence step.
More...

VL53L0X_API VL53L0X_Error **VL53L0X_GetSequenceStepEnabled**
Dev, **VL53L0X_SchedulerSequenceSteps**
*pSchedulerSequenceSteps)
Gets the (on/off) state of all sequence steps.

VL53L0X_API VL53L0X_Error **VL53L0X_SetSequenceStepTimeout**
Dev, **VL53L0X_SequenceStepId** Seq
FixPoint1616_t TimeOutMilliSecs)
Sets the timeout of a requested sequence step.

VL53L0X_API VL53L0X_Error **VL53L0X_GetSequenceStepTimeout**
Dev, **VL53L0X_SequenceStepId** Seq
FixPoint1616_t *pTimeOutMilliSecs)
Gets the timeout of a requested sequence step.

VL53L0X_API VL53L0X_Error **VL53L0X_GetNumberOfSequenceSteps**
Dev, **uint8_t** *pNumberOfSequenceSteps)
Gets number of sequence steps managed by the device.
More...

VL53L0X_API VL53L0X_Error **VL53L0X_GetSequenceStepsInfo**
(**VL53L0X_SequenceStepId** SequenceStepId,
*pSequenceStepsString)
Gets the name of a given sequence step.

VL53L0X_API VL53L0X_Error **VL53L0X_SetInterMeasurementPeriod**
(**VL53L0X_DEV** Dev, **uint32_t**
InterMeasurementPeriodMilliseconds)
Program continuous mode Inter-Measurement period in milliseconds.
More...

VL53L0X_API VL53L0X_Error **VL53L0X_GetInterMeasurementPeriod**
(**VL53L0X_DEV** Dev, **uint32_t**

	*pInterMeasurementPeriodMillisecond Get continuous mode Inter-Measurement period in milliseconds. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetXTalkCompensationEnable (VL53L0X_DEV Dev, uint8_t XTalkCompensationEnable) Enable/Disable Cross talk compensation.
VL53L0X_API VL53L0X_Error	VL53L0X_GetXTalkCompensationRate (VL53L0X_DEV Dev, uint8_t *pXTalkCompensationEnable) Get Cross talk compensation rate. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetXTalkCompensationRate (VL53L0X_DEV Dev, FixPoint1616_t XTalkCompensationRateMegaCps) Set Cross talk compensation rate. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetXTalkCompensationRate (VL53L0X_DEV Dev, FixPoint1616_t *pXTalkCompensationRateMegaCps) Get Cross talk compensation rate. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetRefCalibration (VL53L0X_DEV Dev, VhvSettings , uint8_t PhaseCal) Set Reference Calibration Parameters
VL53L0X_API VL53L0X_Error	VL53L0X_GetRefCalibration (VL53L0X_DEV Dev, uint8_t *pVhvSettings, uint8_t *pPhaseCal) Get Reference Calibration Parameters
VL53L0X_API VL53L0X_Error	VL53L0X_GetNumberOfLimitCheck (VL53L0X_DEV Dev, uint8_t *pNumberOfLimitCheck) Get the number of the check limit manager for the Device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetLimitCheckInfo (VL53L0X_DEV Dev, uint16_t LimitCheckId, char *pLimitCheckName)

Return a description string for a given
More...

VL53L0X_API VL53L0X_Error

VL53L0X_GetLimitCheckStatus (VL53L0X API)
`uint16_t` LimitCheckId, `uint8_t` *pLimitStatus)

Return a the Status of the specified check.

VL53L0X_API VL53L0X_Error

VL53L0X_SetLimitCheckEnable (VL53L0X API)
`uint16_t` LimitCheckId, `uint8_t` LimitCheckEnable)

Enable/Disable a specific limit check. More...

VL53L0X_API VL53L0X_Error

VL53L0X_GetLimitCheckEnable (VL53L0X API)
`uint16_t` LimitCheckId, `uint8_t` *pLimitCheckEnable)

Get specific limit check enable state. More...

VL53L0X_API VL53L0X_Error

VL53L0X_SetLimitCheckValue (VL53L0X API)
`uint16_t` LimitCheckId, `FixPoint1616_t` pLimitCheckValue)

Set a specific limit check value. More...

VL53L0X_API VL53L0X_Error

VL53L0X_GetLimitCheckValue (VL53L0X API)
`uint16_t` LimitCheckId, `FixPoint1616_t` *pLimitCheckValue)

Get a specific limit check value. More...

VL53L0X_API VL53L0X_Error

VL53L0X_GetLimitCheckCurrent (VL53L0X API)
`uint16_t` LimitCheckId, `FixPoint1616_t` *pLimitCheckCurrent)

Get the current value of the signal used for the check.

More...

VL53L0X_API VL53L0X_Error

VL53L0X_SetWrapAroundCheckEnable (VL53L0X API)
`VL53L0X_t` Dev, `uint8_t` WrapAroundCheckEnable)

Enable (or disable) Wrap around Check.

VL53L0X_API VL53L0X_Error

VL53L0X_GetWrapAroundCheckEnable (VL53L0X API)
`VL53L0X_t` Dev, `uint8_t` *pWrapAroundCheckEnable)

Get setup of Wrap around Check. More...

VL53L0X_API VL53L0X_Error	VL53L0X_SetDmaxCalParameters (VL53L0X_Error) uint16_t RangeMilliMeter, FixPoint16_SignalRateRtnMegaCps) Set Dmax Calibration Parameters for a one of the parameter is zero, this function reads parameter from NVM. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetDmaxCalParameters (VL53L0X_Error) uint16_t *pRangeMilliMeter, FixPoint16 *pSignalRateRtnMegaCps) Get Dmax Calibration Parameters for a More...
VL53L0X_API VL53L0X_Error	VL53L0X_PerformSingleMeasurement (VL53L0X_Error) Dev) Single shot measurement. More...
VL53L0X_API VL53L0X_Error	VL53L0X_PerformRefCalibration (VL53L0X_Error) uint8_t *pVhvSettings, uint8_t *pPhase) Perform Reference Calibration. More..
VL53L0X_API VL53L0X_Error	VL53L0X_PerformXTalkMeasurement (VL53L0X_Error) Dev, uint32_t TimeoutMs, FixPoint16 *pXtalkPerSpad, uint8_t *pAmbientTo) Perform XTalk Measurement. More...
VL53L0X_API VL53L0X_Error	VL53L0X_PerformXTalkCalibration (VL53L0X_Error) FixPoint1616_t XTalkCalDistance, FixPoint16 *pXTalkCompensationRateMegaCps) Perform XTalk Calibration. More...
VL53L0X_API VL53L0X_Error	VL53L0X_PerformOffsetCalibration (VL53L0X_Error) FixPoint1616_t CalDistanceMilliMeter, *pOffsetMicroMeter) Perform Offset Calibration. More...
VL53L0X_API VL53L0X_Error	VL53L0X_StartMeasurement (VL53L0X_Error)

Start device measurement. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_StopMeasurement \(VL53L0X_StopMeasurement\)**](#)
Stop device measurement. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_GetMeasurementDataReady \(VL53L0X_GetMeasurementDataReady\)**](#)
Dev, **uint8_t** *pMeasurementDataReady
Return Measurement Data Ready. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_WaitDeviceReadyForNewMeasurement \(VL53L0X_WAITDEVICEREADYFORNEWMEASUREMENT\)**](#)
VL53L0X_DEV Dev, **uint32_t** MaxLoops
Wait for device ready for a new measurement. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_GetMeasurementRefSignal \(VL53L0X_GetMeasurementRefSignal\)**](#)
Dev, **FixPoint1616_t** *pMeasurementRefSignal
Retrieve the Reference Signal after a measurement. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_GetRangingMeasurement \(VL53L0X_GetRangingMeasurement\)**](#)
VL53L0X_DEV Dev,
[**VL53L0X_RangingMeasurementData \(VL53L0X_RangingMeasurementData\)**](#)
*pRangingMeasurementData)
Retrieve the measurements from device. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_GetHistogramMeasurement \(VL53L0X_GetHistogramMeasurement\)**](#)
VL53L0X_DEV Dev,
[**VL53L0X_HistogramMeasurementData \(VL53L0X_HistogramMeasurementData\)**](#)
*pHistogramMeasurementData)
Retrieve the measurements from device. More...

VL53L0X_API VL53L0X_Error [**VL53L0X_PerformSingleRangingMeasurement \(VL53L0X_PerformSingleRangingMeasurement\)**](#)
VL53L0X_DEV Dev,
[**VL53L0X_RangingMeasurementData \(VL53L0X_RangingMeasurementData\)**](#)
*pRangingMeasurementData)
Performs a single ranging measurement.

ranging measurement data. More...

VL53L0X_API VL53L0X_Error	VL53L0X_PerformSingleHistogram (VL53L0X_DEV Dev, VL53L0X_HistogramMeasurementD *pHistogramMeasurementData) Performs a single histogram measure- ment. Returns histogram measurement data. Is equivalent to VL53L0X_PerformSingleMeasuremen- t and VL53L0X_GetHistogramMeasuremen-
VL53L0X_API VL53L0X_Error	VL53L0X_SetNumberOfROIZones (uint8_t NumberOfROIZones) Set the number of ROI Zones to be used by the Device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetNumberOfROIZones (uint8_t *pNumberOfROIZones) Get the number of ROI Zones managed by the Device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetMaxNumberOfROIZones (VL53L0X_DEV Dev, uint8_t *pMaxNumberOfROIZones) Get the Maximum number of ROI Zones supported by the Device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_SetGpioConfig (VL53L0X_Pin, VL53L0X_DeviceModes DeviceMode, VL53L0X_GpioFunctionality Functionality, VL53L0X_InterruptPolarity Polarity) Set the configuration of GPIO pin for a device. More...
VL53L0X_API VL53L0X_Error	VL53L0X_GetGpioConfig (VL53L0X_Pin, VL53L0X_DeviceModes *pDeviceMode, VL53L0X_GpioFunctionality *pFunctionality, VL53L0X_InterruptPolarity *pPolarity) Get current configuration for GPIO pin

More...

VL53L0X_API VL53L0X_Error	VL53L0X_SetInterruptThresholds (VL53L0X_DeviceModes DeviceMode, uint32_t ThresholdLow, FixPoint1616_t ThresholdHigh) Set low and high Interrupt thresholds for ranging, ALS, ...) for a given device. More...
---	---

VL53L0X_API VL53L0X_Error	VL53L0X_GetInterruptThresholds (VL53L0X_DeviceModes DeviceMode, uint32_t *pThresholdLow, FixPoint1616_t *pThresholdHigh) Get high and low Interrupt thresholds for ranging, ALS, ...) for a given device. More...
---	---

VL53L0X_API VL53L0X_Error	VL53L0X_GetStopCompletedStatus (VL53L0X_DeviceModes Dev, uint32_t *pStopStatus) Return device stop completion status.
---	--

VL53L0X_API VL53L0X_Error	VL53L0X_ClearInterruptMask (VL53L0X_DeviceModes Dev, uint32_t InterruptMask) Clear given system interrupt condition.
---	---

VL53L0X_API VL53L0X_Error	VL53L0X_GetInterruptMaskStatus (VL53L0X_DeviceModes Dev, uint32_t *pInterruptMaskStatus) Return device interrupt status. More...
---	---

VL53L0X_API VL53L0X_Error	VL53L0X_EnableInterruptMask (VL53L0X_DeviceModes Dev, uint32_t InterruptMask) Configure ranging interrupt reported to
---	--

VL53L0X_API VL53L0X_Error	VL53L0X_SetSpadAmbientDamperThreshold (VL53L0X_DEV Dev, uint16_t SpadAmbientDamperThreshold) Set the SPAD Ambient Damper Threshold
---	---

VL53L0X_API VL53L0X_Error	VL53L0X_GetSpadAmbientDamperThreshold (VL53L0X_DEV Dev, uint16_t *pSpadAmbientDamperThreshold) Get the SPAD Ambient Damper Threshold
---	---

`*pSpadAmbientDamperThreshold)`
Get the current SPAD Ambient Damper Factor
[More...](#)

VL53L0X_API VL53L0X_Error [**VL53L0X_SetSpadAmbientDamperFactor**](#) (**VL53L0X_DEV** Dev, **uint16_t** *pSpadAmbientDamperFactor)
Set the SPAD Ambient Damper Factor

VL53L0X_API VL53L0X_Error [**VL53L0X_GetSpadAmbientDamperFactor**](#) (**VL53L0X_DEV** Dev, **uint16_t** *pSpadAmbientDamperFactor)
Get the current SPAD Ambient Damper Factor
[More...](#)

VL53L0X_API VL53L0X_Error [**VL53L0X_PerformRefSpadManagement**](#) (Dev, **uint32_t** *refSpadCount, **uint8_t** *isApe)
Performs Reference Spad Management

VL53L0X_API VL53L0X_Error [**VL53L0X_SetReferenceSpads**](#) (**VL53L0X_DEV** Dev, **uint32_t** refSpadCount, **uint8_t** isApe)
Applies Reference SPAD configuration

VL53L0X_API VL53L0X_Error [**VL53L0X_GetReferenceSpads**](#) (**VL53L0X_DEV** Dev, **uint32_t** *refSpadCount, **uint8_t** *isApe)
Retrieves SPAD configuration. [More...](#)

Macro Definition Documentation

```
#define VL53L0X_API
```

Definition at line [48](#) of file [vl53l0x_api.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	core
			inc

Functions

vl53l0x_api_calibration.h File Reference

```
#include "vl53l0x_def.h" #include "vl53l0x_platform.h"
```

Go to the source code of this file.

Functions

VL53L0X_Error	VL53L0X_perform_xtalk_calibration (VL53L0X_DEV Dev, FixPoint1616_t XTalkCalDistance, FixPoint1616_t *pXTalkCompensationRateMegaCps)
VL53L0X_Error	VL53L0X_perform_offset_calibration (VL53L0X_DEV Dev, FixPoint1616_t CalDistanceMilliMeter, int32_t *pOffsetMicroMeter)
VL53L0X_Error	VL53L0X_set_offset_calibration_data_micro_meter (VL53L0X_DEV Dev, int32_t OffsetCalibrationDataMicroMeter)
VL53L0X_Error	VL53L0X_get_offset_calibration_data_micro_meter (VL53L0X_DEV Dev, int32_t *pOffsetCalibrationDataMicroMeter)
VL53L0X_Error	VL53L0X_apply_offset_adjustment (VL53L0X_DEV Dev)
VL53L0X_Error	VL53L0X_perform_ref_spad_management (VL53L0X_DEV Dev, uint32_t *refSpadCount, uint8_t *isApertureSpads)
VL53L0X_Error	VL53L0X_set_reference_spads (VL53L0X_DEV Dev, uint32_t count, uint8_t isApertureSpads)
VL53L0X_Error	VL53L0X_get_reference_spads (VL53L0X_DEV Dev, uint32_t *pSpadCount, uint8_t *pIsApertureSpads)
VL53L0X_Error	VL53L0X_perform_phase_calibration (VL53L0X_DEV Dev, uint8_t *pPhaseCal, const uint8_t get_data_enable, const uint8_t restore_config)

VL53L0X_Error	VL53L0X_perform_ref_calibration (VL53L0X_DEV Dev, uint8_t *pVhvSettings, uint8_t *pPhaseCal, uint8_t get_data_enable)
VL53L0X_Error	VL53L0X_set_ref_calibration (VL53L0X_DEV Dev, uint8_t VhvSettings, uint8_t PhaseCal)
VL53L0X_Error	VL53L0X_get_ref_calibration (VL53L0X_DEV Dev, uint8_t *pVhvSettings, uint8_t *pPhaseCal)

Function Documentation

VL53L0X_Error

```
VL53L0X_perform_xtalk_calibration( VL53L0X_DEV Dev,  
                                     FixPoint1616_t XTalkCalDis  
                                     FixPoint1616_t * pXTalkCom  
                                     )
```

VL53L0X_Error

```
VL53L0X_perform_offset_calibration( VL53L0X_DEV Dev,  
                                      FixPoint1616_t CalDistance  
                                      int32_t *          pOffsetMicro  
                                      )
```

VL53L0X_Error

```
VL53L0X_set_offset_calibration_data_micro_meter( VL53L0X_DEV Dev,  
                                                int32_t *          pOffsetMicro  
                                                )
```

VL53L0X_Error

```
VL53L0X_get_offset_calibration_data_micro_meter( VL53L0X_DEV Dev,  
                                                int32_t *          pOffsetMicro  
                                                )
```

VL53L0X_Error

```
VL53L0X_apply_offset_adjustment( VL53L0X_DEV Dev )
```

VL53L0X_Error

```
VL53L0X_perform_ref_spad_management( VL53L0X_DEV Dev,  
                                       uint32_t *          refSpad  
                                       uint8_t *           isAperture  
                                       )
```

)

VL53L0X_Error

```
VL53L0X_set_reference_spads (VL53L0X_DEV Dev,  
                           uint32_t      count,  
                           uint8_t       isApertureSpads  
                         )
```

VL53L0X Error

```
VL53L0X_get_reference_spads( VL53L0X_DEV Dev,  
                            uint32_t *      pSpadCount,  
                            uint8_t *       plsApertureSpads  
                          )
```

VL53L0X Error

```
VL53L0X_perform_phase_calibration( VL53L0X_DEV Dev,  
                                    uint8_t * pPhaseCal,  
                                    const uint8_t get_data_en,  
                                    const uint8_t restore_con  
                                )
```

VI 53I 0X Error

```
VL53L0X_perform_ref_calibration( VL53L0X_DEV Dev,  
                                  uint8_t * pVhvSettings,  
                                  uint8_t * pPhaseCal,  
                                  uint8_t get_data_enabl  
                                )
```

VL53L0X Error

VL53L0X set ref calibration

```
( VL53L0X_DEV Dev,  
    uint8_t      VhvSettings,  
    uint8_t      PhaseCal  
)
```

VL53L0X_Error

VL53L0X_get_ref_calibration (**VL53L0X_DEV** **Dev**,
 uint8_t * **pVhvSettings**,
 uint8_t * **pPhaseCal**
)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	core
			inc

Functions

vl53l0x_api_core.h File Reference

```
#include "vl53l0x_def.h" #include "vl53l0x_platform.h"
```

Go to the source code of this file.

Functions

VL53L0X_Error **VL53L0X_reverse_bytes** (**uint8_t** *data, **uint32_t** size)

VL53L0X_Error **VL53L0X_measurement_poll_for_completion** (**VL53L0X_DEV** Dev)

uint8_t **VL53L0X_encode_vcSEL_period** (**uint8_t** vcSEL_peri

uint8_t **VL53L0X_decode_vcSEL_period** (**uint8_t** vcSEL_peri

uint32_t **VL53L0X_isqrt** (**uint32_t** num)

uint32_t **VL53L0X_quadrature_sum** (**uint32_t** a, **uint32_t** b)

VL53L0X_Error **VL53L0X_get_info_from_device** (**VL53L0X_DEV** Dev, **VL53L0X_Info** option)

VL53L0X_Error **VL53L0X_set_vcSEL_pulse_period** (**VL53L0X_DEV** Dev, **VL53L0X_VcSELPeriod** VcSELPeriodType, **uint8_t** VCSELPulsePeriodPCLK)

VL53L0X_Error **VL53L0X_get_vcSEL_pulse_period** (**VL53L0X_DEV** Dev, **VL53L0X_VcSELPeriod** VcSELPeriodType, **uint8_t** *pVCSELPulsePeriodPCLK)

uint32_t **VL53L0X_decode_timeout** (**uint16_t** encoded_timeo

VL53L0X_Error **get_sequence_step_timeout** (**VL53L0X_DEV** Dev, **VL53L0X_SequenceStepId** SequenceStepId, **uint32_t** *pTimeOutMicroSecs)

VL53L0X_Error **set_sequence_step_timeout** (**VL53L0X_DEV** Dev, **VL53L0X_SequenceStepId** SequenceStepId, **uint32_t** TimeOutMicroSecs)

VL53L0X_set_measurement_timing_budget_microSecs (**VL53L0X_DEV** Dev, **VL53L0X_MeasurementTimingBudget** timingBudget, **uint32_t** microSecs)

VL53L0X_Error	VL53L0X_Error (VL53L0X_DEV Dev, uint32_t MeasurementTimingBudgetMicroSeconds)
VL53L0X_Error	VL53L0X_get_measurement_timing_budget_microseconds (VL53L0X_DEV Dev, uint32_t *pMeasurementTimingBudgetMicroSeconds)
VL53L0X_Error	VL53L0X_load_tuning_settings (VL53L0X_DEV Dev, *pTuningSettingBuffer)
VL53L0X_Error	VL53L0X_calc_sigma_estimate (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData, FixPoint1616_t *pSigmaEstimate, uint32_t *pDmax_mm)
VL53L0X_Error	VL53L0X_get_total_xtalk_rate (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData, FixPoint1616_t *ptotal_xtalk_rate_mcps)
VL53L0X_Error	VL53L0X_get_total_signal_rate (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData, FixPoint1616_t *ptotal_signal_rate_mcps)
VL53L0X_Error	VL53L0X_get_pal_range_status (VL53L0X_DEV Dev, DeviceRangeStatus, FixPoint1616_t SignalRate, uint8_t EffectiveSpadRtnCount, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData, uint8_t *pPalRangeStatus)
uint32_t	VL53L0X_calc_timeout_mclks (VL53L0X_DEV Dev, timeout_period_us, uint8_t vcsel_period_pclks)
uint16_t	VL53L0X_encode_timeout (uint32_t timeout_macrc)

Function Documentation

```
VL53L0X_Error VL53L0X_reverse_bytes( uint8_t * data,  
                                      uint32_t size  
                                    )
```

```
VL53L0X_Error  
VL53L0X_measurement_poll_for_completion( VL53L0X_DEV Dev
```

```
uint8_t  
VL53L0X_encode_vcSEL_period( uint8_t vcSEL_period_pcLks )
```

```
uint8_t  
VL53L0X_decode_vcSEL_period( uint8_t vcSEL_period_Reg )
```

```
uint32_t VL53L0X_isqrt( uint32_t num )
```

```
uint32_t VL53L0X_quadrature_sum( uint32_t a,  
                                  uint32_t b  
                                )
```

```
VL53L0X_Error  
VL53L0X_get_info_from_device( VL53L0X_DEV Dev,  
                               uint8_t option  
                             )
```

```
VL53L0X_Error  
VL53L0X_set_vcSEL_pulse_period( VL53L0X_DEV Dev,  
                                 VL53L0X_VcSELPeriod VcSELPe  
                               uint8_t VCSELPE )
```

)

**VL53L0X_Error VL53L0X_get_vcsel_pulse_period(VL53L0X_DEV Dev,
VL53L0X_VcselPeriod VcselPe
uint8_t * pVCSEL
)**

uint32_t VL53L0X_decode_timeout(uint16_t encoded_timeout)

**VL53L0X_Error get_sequence_step_timeout(VL53L0X_DEV Dev,
VL53L0X_SequenceStepId Sequence
uint32_t * pTimeOut
)**

**VL53L0X_Error set_sequence_step_timeout(VL53L0X_DEV Dev,
VL53L0X_SequenceStepId Sequence
uint32_t TimeOut
)**

**VL53L0X_Error VL53L0X_set_measurement_timing_budget_micro_seconds(VL53L0X_DEV Dev,
uint32_t TimeBudget
)**

**VL53L0X_Error VL53L0X_get_measurement_timing_budget_micro_seconds(VL53L0X_DEV Dev,
uint32_t * pTimeBudget
)**

VL53L0X_Error

```
VL53L0X_load_tuning_settings ( VL53L0X_DEV Dev,  
                                uint8_t *          pTuningSettingBu  
                                )
```

```
VL53L0X_Error  
VL53L0X_calc_sigma_estimate ( VL53L0X_DEV  
                               VL53L0X_RangingMeasurementD  
                               FixPoint1616_t *  
                               uint32_t *  
                               )
```

```
VL53L0X_Error  
VL53L0X_get_total_xtalk_rate ( VL53L0X_DEV  
                               VL53L0X_RangingMeasurementDa  
                               FixPoint1616_t *  
                               )
```

```
VL53L0X_Error  
VL53L0X_get_total_signal_rate ( VL53L0X_DEV  
                               VL53L0X_RangingMeasurementD  
                               FixPoint1616_t *  
                               )
```

```
VL53L0X_Error  
VL53L0X_get_pal_range_status ( VL53L0X_DEV  
                                uint8_t  
                                FixPoint1616_t  
                                uint16_t  
                                VL53L0X_RangingMeasurementD  
                                uint8_t *  
                                )
```

```
uint32_t  
VL53L0X_calc_timeout_mclks ( VL53L0X_DEV Dev,
```

```
        uint32_t      timeout_period_us,
        uint8_t       vcsel_period_pclks
    )
```

```
uint16_t VL53L0X_encode_timeout( uint32_t timeout_macro_clks )
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	core

vl53l0x_api_ranging.h File Reference

```
#include "vl53l0x_def.h" #include "vl53l0x_platform.h"
```

[Go to the source code of this file.](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	core
			inc

Macros | Functions

vl53l0x_api_strings.h

File Reference

```
#include "vl53l0x_def.h" #include "vl53l0x_platform.h"
```

Go to the source code of this file.

Macros

```
#define VL53L0X_STRING_DEVICE_INFO_NAME "VL53L0X cut1.0"

#define VL53L0X_STRING_DEVICE_INFO_NAME_TS0 "VL53L0X 1"

#define VL53L0X_STRING_DEVICE_INFO_NAME_TS1 "VL53L0X 1"

#define VL53L0X_STRING_DEVICE_INFO_NAME_TS2 "VL53L0X 1"

#define VL53L0X_STRING_DEVICE_INFO_NAME_ES1 "VL53L0X ES1"

#define VL53L0X_STRING_DEVICE_INFO_TYPE "VL53L0X"

#define VL53L0X_STRING_ERROR_NONE "No Error"

#define VL53L0X_STRING_ERROR_CALIBRATION_WARNING "Calibration error"

#define VL53L0X_STRING_ERROR_MIN_CLIPPED "Min clipped error"

#define VL53L0X_STRING_ERROR_UNDEFINED "Undefined error"

#define VL53L0X_STRING_ERROR_INVALID_PARAMS "Invalid parameters"

#define VL53L0X_STRING_ERROR_NOT_SUPPORTED "Not supported"

#define VL53L0X_STRING_ERROR_RANGE_ERROR "Range error"

#define VL53L0X_STRING_ERROR_TIME_OUT "Time out error"

#define VL53L0X_STRING_ERROR_MODE_NOT_SUPPORTED "Mode not supported"

#define VL53L0X_STRING_ERROR_BUFFER_TOO_SMALL "Buffer too small"
```

```
#define VL53L0X_STRING_ERROR_GPIO_NOT_EXISTING "GPIO  
#define VL53L0X_STRING_ERROR_GPIO_FUNCTIONALITY_NOT_  
funct not supported"  
  
#define VL53L0X_STRING_ERROR_INTERRUPT_NOT_CLEARED  
Cleared"  
  
#define VL53L0X_STRING_ERROR_CONTROL_INTERFACE "Con  
  
#define VL53L0X_STRING_ERROR_INVALID_COMMAND "Invalid  
#define VL53L0X_STRING_ERROR_DIVISION_BY_ZERO "Division  
  
#define VL53L0X_STRING_ERROR_REF_SPAD_INIT "Reference S  
  
#define VL53L0X_STRING_ERROR_NOT_IMPLEMENTED "Not imple  
  
#define VL53L0X_STRING_UNKNOW_ERROR_CODE "Unknown E  
  
#define VL53L0X_STRING_RANGESTATUS_NONE "No Update"  
  
#define VL53L0X_STRING_RANGESTATUS_RANGEVALID "Range  
  
#define VL53L0X_STRING_RANGESTATUS_SIGMA "Sigma Fail"  
  
#define VL53L0X_STRING_RANGESTATUS_SIGNAL "Signal Fail"  
  
#define VL53L0X_STRING_RANGESTATUS_MINRANGE "Min Ran  
  
#define VL53L0X_STRING_RANGESTATUS_PHASE "Phase Fail"  
  
#define VL53L0X_STRING_RANGESTATUS_HW "Hardware Fail"  
  
#define VL53L0X_STRING_STATE_POWERDOWN "POWERDOWN  
  
#define VL53L0X_STRING_STATE_WAIT_STATICINIT "Wait for stat
```

```
#define VL53L0X_STRING_STATE_STANDBY "STANDBY State"

#define VL53L0X_STRING_STATE_IDLE "IDLE State"

#define VL53L0X_STRING_STATE_RUNNING "RUNNING State"

#define VL53L0X_STRING_STATE_UNKNOWN "UNKNOWN State"

#define VL53L0X_STRING_STATE_ERROR "ERROR State"

#define VL53L0X_STRING_DEVICEERROR_NONE "No Update"

#define VL53L0X_STRING_DEVICEERROR_VSELCONTINUITYTE Continuity Test Failure"

#define VL53L0X_STRING_DEVICEERROR_VSELWATCHDOGT Watchdog Test Failure"

#define VL53L0X_STRING_DEVICEERROR_NOVHVVALUEFOUND found"

#define VL53L0X_STRING_DEVICEERROR_MSRCNOTARGET "M

#define VL53L0X_STRING_DEVICEERROR_SNRCHECK "SNR Ch

#define VL53L0X_STRING_DEVICEERROR_RANGEPHASECHECK Check Error"

#define VL53L0X_STRING_DEVICEERROR_SIGMATHRESHOLDC Threshold Check Error"

#define VL53L0X_STRING_DEVICEERROR_TCC "TCC Error"

#define VL53L0X_STRING_DEVICEERROR_PHASECONSISTENCY Error"

#define VL53L0X_STRING_DEVICEERROR_MINCLIP "Min Clip Err
```

```
#define VL53L0X_STRING_DEVICEERROR_RANGECOMPLETE "I  
#define VL53L0X_STRING_DEVICEERROR_ALGOUNDERFLOW "I  
#define VL53L0X_STRING_DEVICEERROR_ALGOOVERFLOW "R  
#define VL53L0X_STRING_DEVICEERROR_RANGEIGNORETHRES  
#define VL53L0X_STRING_DEVICEERROR_UNKNOWN "Unknown  
#define VL53L0X_STRING_CHECKENABLE_SIGMA_FINAL_RANG  
#define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_FINAL  
#define VL53L0X_STRING_CHECKENABLE_SIGNAL_REF_CLIP '  
#define VL53L0X_STRING_CHECKENABLE_RANGE_IGNORE_THI  
#define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_MSRC  
#define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_PRE_  
#define VL53L0X_STRING_SEQUENCESTEP_TCC "TCC"  
#define VL53L0X_STRING_SEQUENCESTEP_DSS "DSS"  
#define VL53L0X_STRING_SEQUENCESTEP_MSRC "MSRC"  
#define VL53L0X_STRING_SEQUENCESTEP_PRE_RANGE "PRE
```

```
#define VL53L0X_STRING_SEQUENCESTEP_FINAL_RANGE "FIN
```

Functions

VL53L0X_Error **VL53L0X_get_device_info** (**VL53L0X_DEV** Dev,
VL53L0X_DeviceInfo_t *pVL53L0X_DeviceInfo)

VL53L0X_Error **VL53L0X_get_device_error_string**
(**VL53L0X_DeviceError** ErrorCode, char
*pDeviceErrorString)

VL53L0X_Error **VL53L0X_get_range_status_string** (**uint8_t**
RangeStatus, char *pRangeStatusString)

VL53L0X_Error **VL53L0X_get_pal_error_string** (**VL53L0X_Error**
PalErrorCode, char *pPalErrorString)

VL53L0X_Error **VL53L0X_get_pal_state_string** (**VL53L0X_State**
PalStateCode, char *pPalStateString)

VL53L0X_Error **VL53L0X_get_sequence_steps_info**
(**VL53L0X_SequenceStepId** SequenceStepId, char
*pSequenceStepsString)

VL53L0X_Error **VL53L0X_get_limit_check_info** (**VL53L0X_DEV**
Dev, **uint16_t** LimitCheckId, char
*pLimitCheckString)

Macro Definition Documentation

```
#define VL53L0X_STRING_DEVICE_INFO_NAME "VL53L0X  
cut1.0"
```

Definition at line [145](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICE_INFO_NAME_TS0 "VL53L0X TS0"
```

Definition at line [146](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICE_INFO_NAME_TS1 "VL53L0X TS1"
```

Definition at line [147](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICE_INFO_NAME_TS2 "VL53L0X TS2"
```

Definition at line [148](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICE_INFO_NAME_ES1 "VL53L0X ES1 or  
later"
```

Definition at line [149](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_DEVICE_INFO_TYPE "VL53L0X"
```

Definition at line [150](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_ERROR_NONE "No Error"
```

Definition at line [153](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_CALIBRATION_WARNING "Calibra  
tion Warning Error"
```

Definition at line [155](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_ERROR_MIN_CLIPPED "Min clipped  
error"
```

Definition at line [157](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_ERROR_UNDEFINED "Undefined  
error"
```

Definition at line [159](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_ERROR_INVALID_PARAMS "Invalid  
parameters error"
```

Definition at line [161](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_ERROR_NOT_SUPPORTED "Not  
supported error"
```

Definition at line [163](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_ERROR_RANGE_ERROR "Range  
error"
```

Definition at line [165](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_ERROR_TIME_OUT "Time out error"
```

Definition at line [167](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_MODE_NOT_SUPPORTED "Mode  
not supported error"
```

Definition at line [169](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_BUFFER_TOO_SMALL "Buffer too  
small"
```

Definition at line [171](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_GPIO_NOT_EXISTING "GPIO not  
existing"
```

Definition at line [173](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORT
```

funct not supported"

Definition at line [175](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_INTERRUPT_NOT_CLEARED "Interrupt  
not Cleared"
```

Definition at line [177](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_CONTROL_INTERFACE "Control  
Interface Error"
```

Definition at line [179](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_INVALID_COMMAND "Invalid  
Command Error"
```

Definition at line [181](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_DIVISION_BY_ZERO "Division by  
zero Error"
```

Definition at line [183](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_ERROR_REF_SPAD_INIT "Reference Spad  
Init Error"
```

Definition at line [185](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_ERROR_NOT_IMPLEMENTED "Not implemented error"
```

Definition at line [187](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_UNKNOW_ERROR_CODE "Unknown Error  
Code"
```

Definition at line [190](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_RANGESTATUS_NONE "No Update"
```

Definition at line [196](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_RANGESTATUS_RANGEVALID "Range  
Valid"
```

Definition at line [197](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_RANGESTATUS_SIGMA "Sigma  
Fail"
```

Definition at line [198](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_RANGESTATUS_SIGNAL "Signal  
Fail"
```

Definition at line [199](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_RANGESTATUS_MINRANGE "Min  
Range Fail"
```

Definition at line [200](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_RANGESTATUS_PHASE "Phase  
Fail"
```

Definition at line [201](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_RANGESTATUS_HW "Hardware  
Fail"
```

Definition at line [202](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_STATE_POWERDOWN "POWERDOWN  
State"
```

Definition at line [206](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_STATE_WAIT_STATICINIT "Wait for  
staticinit State"
```

Definition at line [207](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_STATE_STANDBY "STANDBY State"
```

Definition at line [209](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_STATE_IDLE "IDLE State"
```

Definition at line [210](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_STATE_RUNNING "RUNNING State"
```

Definition at line [211](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_STATE_UNKNOWN "UNKNOWN State"
```

Definition at line [212](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_STATE_ERROR "ERROR State"
```

Definition at line [213](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_DEVICEERROR_NONE "No Update"
```

Definition at line [217](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICEERROR_VSELCONTINUITYTESTFAILU  
Continuity Test Failure"
```

Definition at line [218](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICEERROR_VSELWATCHDOGTESTFAILU  
Watchdog Test Failure"
```

Definition at line [220](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICEERROR_NOVHVVALUEFOUND "No  
VHV Value found"
```

Definition at line [222](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICEERROR_MSRCNOTARGET "MSRC  
No Target Error"
```

Definition at line [224](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_DEVICEERROR_SNRCHECK "SNR  
Check Exit"
```

Definition at line [226](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICEERROR_RANGEPHASECHECK "Rang  
Phase Check Error"
```

Definition at line [228](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICEERROR_SIGMATHRESHOLDCHECK "S  
Threshold Check Error"
```

Definition at line [230](#) of file `vl53l0x_api_strings.h`.

```
#define VL53L0X_STRING_DEVICEERROR_TCC "TCC Error"
```

Definition at line 232 of file [vl53l0x_api_strings.h](#).

```
#define  
VL53L0X_STRING_DEVICEERROR_PHASECONSISTENCY "Phase  
Consistency Error"
```

Definition at line 234 of file [vl53l0x_api_strings.h](#).

```
#define VL53L0X_STRING_DEVICEERROR_MINCLIP "Min Clip  
Error"
```

Definition at line 236 of file [vl53l0x_api_strings.h](#).

```
#define  
VL53L0X_STRING_DEVICEERROR_RANGECOMPLETE "Range  
Complete"
```

Definition at line 238 of file [vl53l0x_api_strings.h](#).

```
#define  
VL53L0X_STRING_DEVICEERROR_ALGOUNDERFLOW "Range  
Algo Underflow Error"
```

Definition at line 240 of file [vl53l0x_api_strings.h](#).

```
#define  
VL53L0X_STRING_DEVICEERROR_ALGOOVERFLOW "Range  
Algo Overflow Error"
```

Definition at line 242 of file [vl53l0x_api_strings.h](#).

```
#define  
VL53L0X_STRING_DEVICEERROR_RANGEIGNORETHRESHOLD  
Ignore Threshold Error"
```

Definition at line [244](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_DEVICEERROR_UNKNOWN "Unknown error  
code"
```

Definition at line [246](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_CHECKENABLE_SIGMA_FINAL_RANGE "SIGMA  
FINAL RANGE"
```

Definition at line [250](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE  
"SIGNAL RATE FINAL RANGE"
```

Definition at line [252](#) of file `vl53l0x_api_strings.h`.

```
#define  
VL53L0X_STRING_CHECKENABLE_SIGNAL_REF_CLIP "SIGNAL  
REF CLIP"
```

Definition at line [254](#) of file `vl53l0x_api_strings.h`.

```
#define
```

```
VL53L0X_STRING_CHECKENABLE_RANGE_IGNORE_THRESHOLD  
IGNORE THRESHOLD"
```

Definition at line [256](#) of file [vl53l0x_api_strings.h](#).

```
#define  
VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_MSRC "SIGN  
RATE MSRC"
```

Definition at line [258](#) of file [vl53l0x_api_strings.h](#).

```
#define  
VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_PRE_RANGE  
RATE PRE RANGE"
```

Definition at line [260](#) of file [vl53l0x_api_strings.h](#).

```
#define VL53L0X_STRING_SEQUENCESTEP_TCC "TCC"
```

Definition at line [264](#) of file [vl53l0x_api_strings.h](#).

```
#define VL53L0X_STRING_SEQUENCESTEP_DSS "DSS"
```

Definition at line [265](#) of file [vl53l0x_api_strings.h](#).

```
#define VL53L0X_STRING_SEQUENCESTEP_MSRC "MSRC"
```

Definition at line [266](#) of file [vl53l0x_api_strings.h](#).

```
#define  
VL53L0X_STRING_SEQUENCESTEP_PRE_RANGE "PRE
```

RANGE"

Definition at line **267** of file **vl53l0x_api_strings.h**.

```
#define  
VL53L0X_STRING_SEQUENCESTEP_FINAL_RANGE "FINAL  
RANGE"
```

Definition at line **268** of file **vl53l0x_api_strings.h**.

Function Documentation

VL53L0X_Error

```
VL53L0X_get_device_info(VL53L0X_DEV Dev,  
                         VL53L0X_DeviceInfo_t * pVL53L0X_DeviceInfo)  
{
```

VL53L0X Error

VL53L0X Error

```
VL53L0X_get_range_status_string ( uint8_t RangeStatus,  
                                  char * pRangeStatusString  
                                )
```

VL53L0X Error

```
VL53L0X_get_pal_error_string ( VL53L0X_Error PalErrorCode,  
                                char *          pPalErrorString  
                            )
```

VL53L0X Error

```
VL53L0X_get_pal_state_string ( VL53L0X_State PalStateCode,  
                                char *          pPalStateString  
                            )
```

VL53L0X Error

```
VL53L0X_get_sequence_steps_info( VL53L0X_SequenceStepId S  
                                char *  
                                )  
{
```

```
VL53L0X_Error
```

```
VL53L0X_get_limit_check_info ( VL53L0X_DEV Dev,
                                uint16_t      LimitCheckId,
                                char *        pLimitCheckString
                            )
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	core inc

[Data Structures](#) | [Macros](#) | [Typedefs](#)

vl53l0x_def.h File Reference

Type definitions for VL53L0X API. More...

```
#include "vl53l0x_device.h" #include "vl53l0x_types.h"
```

[Go to the source code of this file.](#)

Data Structures

struct **VL53L0X_Version_t**

Defines the parameters of the Get Version Functions. [More...](#)

struct **VL53L0X_DeviceInfo_t**

Defines the parameters of the Get Device Info Functions.

[More...](#)

struct **VL53L0X_DeviceParameters_t**

Defines all parameters for the device. [More...](#)

struct **VL53L0X_DMaxData_t**

Structure containing the Dmax computation parameters and data. [More...](#)

struct **VL53L0X_RangingMeasurementData_t**

struct **VL53L0X_HistogramMeasurementData_t**

struct **VL53L0X_SpadData_t**

Spad Configuration Data. [More...](#)

struct **VL53L0X_DeviceSpecificParameters_t**

struct **VL53L0X_DevData_t**

VL53L0X PAL device ST private data structure

End user should never access any of these field directly.

[More...](#)

struct **VL53L0X_SchedulerSequenceSteps_t**

Macros

```
#define VL53L0X10_SPECIFICATION_VER_MAJOR 1  
PAL SPECIFICATION major version. More...
```

```
#define VL53L0X10_SPECIFICATION_VER_MINOR 2  
PAL SPECIFICATION minor version. More...
```

```
#define VL53L0X10_SPECIFICATION_VER_SUB 7  
PAL SPECIFICATION sub version. More...
```

```
#define VL53L0X10_SPECIFICATION_VER_REVISION 1440  
PAL SPECIFICATION sub version. More...
```

```
#define VL53L0X10_IMPLEMENTATION_VER_MAJOR 1  
VL53L0X PAL IMPLEMENTATION major version. More...
```

```
#define VL53L0X10_IMPLEMENTATION_VER_MINOR 0  
VL53L0X PAL IMPLEMENTATION minor version. More...
```

```
#define VL53L0X10_IMPLEMENTATION_VER_SUB 9  
VL53L0X PAL IMPLEMENTATION sub version. More...
```

```
#define VL53L0X10_IMPLEMENTATION_VER_REVISION 3673  
VL53L0X PAL IMPLEMENTATION sub version. More...
```

```
#define VL53L0X_SPECIFICATION_VER_MAJOR 1  
PAL SPECIFICATION major version. More...
```

```
#define VL53L0X_SPECIFICATION_VER_MINOR 2  
PAL SPECIFICATION minor version. More...
```

```
#define VL53L0X_SPECIFICATION_VER_SUB 7  
PAL SPECIFICATION sub version. More...
```

```
#define VL53L0X_SPECIFICATION_VER_REVISION 1440
    PAL SPECIFICATION sub version. More...

#define VL53L0X_IMPLEMENTATION_VER_MAJOR 1
    VL53L0X PAL IMPLEMENTATION major version. More...

#define VL53L0X_IMPLEMENTATION_VER_MINOR 0
    VL53L0X PAL IMPLEMENTATION minor version. More...

#define VL53L0X_IMPLEMENTATION_VER_SUB 2
    VL53L0X PAL IMPLEMENTATION sub version. More...

#define VL53L0X_IMPLEMENTATION_VER_REVISION 4823
    VL53L0X PAL IMPLEMENTATION sub version. More...

#define VL53L0X_DEFAULT_MAX_LOOP 2000

#define VL53L0X_MAX_STRING_LENGTH 32

#define VL53L0X_ERROR_NONE ((VL53L0X_Error) 0)

#define VL53L0X_ERROR_CALIBRATION_WARNING ((VL53L0X_Error) -1)

#define VL53L0X_ERROR_MIN_CLIPPED ((VL53L0X_Error) -2)

#define VL53L0X_ERROR_UNDEFINED ((VL53L0X_Error) -3)

#define VL53L0X_ERROR_INVALID_PARAMS ((VL53L0X_Error) -4)

#define VL53L0X_ERROR_NOT_SUPPORTED ((VL53L0X_Error) -5)

#define VL53L0X_ERROR_RANGE_ERROR ((VL53L0X_Error) -6)

#define VL53L0X_ERROR_TIME_OUT ((VL53L0X_Error) -7)

#define VL53L0X_ERROR_MODE_NOT_SUPPORTED ((VL53L0X_Error) -8)
```

```
#define VL53L0X_ERROR_BUFFER_TOO_SMALL ((VL53L0X_Error) -1)

#define VL53L0X_ERROR_GPIO_NOT_EXISTING ((VL53L0X_Error) -2)

#define VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED ((VL53L0X_Error) -3)

#define VL53L0X_ERROR_INTERRUPT_NOT_CLEARED ((VL53L0X_Error) -4)

#define VL53L0X_ERROR_CONTROL_INTERFACE ((VL53L0X_Error) -5)

#define VL53L0X_ERROR_INVALID_COMMAND ((VL53L0X_Error) -6)

#define VL53L0X_ERROR_DIVISION_BY_ZERO ((VL53L0X_Error) -7)

#define VL53L0X_ERROR_REF_SPAD_INIT ((VL53L0X_Error) -50)

#define VL53L0X_ERROR_NOT_IMPLEMENTED ((VL53L0X_Error) -51)

#define VL53L0X_DEVICEMODE_SINGLE_RANGING ((VL53L0X_DeviceMode) 0)

#define VL53L0X_DEVICEMODE_CONTINUOUS_RANGING ((VL53L0X_DeviceMode) 1)

#define VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM ((VL53L0X_DeviceMode) 2)

#define VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING ((VL53L0X_DeviceMode) 3)

#define VL53L0X_DEVICEMODE_SINGLE_ALS ((VL53L0X_DeviceMode) 4)

#define VL53L0X_DEVICEMODE_GPIO_DRIVE ((VL53L0X_DeviceMode) 5)

#define VL53L0X_DEVICEMODE_GPIO_OSC ((VL53L0X_DeviceMode) 6)

#define VL53L0X_HISTOGRAMMODE_DISABLED ((VL53L0X_HistogramMode) 0)

#define VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY ((VL53L0X_HistogramMode) 1)
```

```
#define VL53L0X_HISTOGRAMMODE_RETURN_ONLY ((VL53L0X_HISTOGRAMMODE_RETURN_ONLY))

#define VL53L0X_HISTOGRAMMODE_BOTH ((VL53L0X_HistogramMode))

#define VL53L0X_POWERMODE_STANDBY_LEVEL1 ((VL53L0X_PowerMode))

#define VL53L0X_POWERMODE_STANDBY_LEVEL2 ((VL53L0X_PowerMode))

#define VL53L0X_POWERMODE_IDLE_LEVEL1 ((VL53L0X_PowerMode))

#define VL53L0X_POWERMODE_IDLE_LEVEL2 ((VL53L0X_PowerMode))

#define VL53L0X_STATE_POWERDOWN ((VL53L0X_State) 0)

#define VL53L0X_STATE_WAIT_STATICINIT ((VL53L0X_State) 1)

#define VL53L0X_STATE_STANDBY ((VL53L0X_State) 2)

#define VL53L0X_STATE_IDLE ((VL53L0X_State) 3)

#define VL53L0X_STATE_RUNNING ((VL53L0X_State) 4)

#define VL53L0X_STATE_UNKNOWN ((VL53L0X_State) 98)

#define VL53L0X_STATE_ERROR ((VL53L0X_State) 99)

#define VL53L0X_HISTOGRAM_BUFFER_SIZE 24

#define VL53L0X_REF_SPAD_BUFFER_SIZE 6

#define VL53L0X_INTERRUPTPOLARITY_LOW ((VL53L0X_InterruptPolarity))

#define VL53L0X_INTERRUPTPOLARITY_HIGH ((VL53L0X_InterruptPolarity))

#define VL53L0X_VCSEL_PERIOD_PRE_RANGE ((VL53L0X_Vcse))

#define VL53L0X_VCSEL_PERIOD_FINAL_RANGE ((VL53L0X_Vcse))
```

```
#define VL53L0X_SEQUENCESTEP_TCC ((VL53L0X_VcselPeriod  
#define VL53L0X_SEQUENCESTEP_DSS ((VL53L0X_VcselPeriod  
#define VL53L0X_SEQUENCESTEP_MSRC ((VL53L0X_VcselPeriod  
#define VL53L0X_SEQUENCESTEP_PRE_RANGE ((VL53L0X_VcselPeriod  
#define VL53L0X_SEQUENCESTEP_FINAL_RANGE ((VL53L0X_VcselPeriod  
#define VL53L0X_SEQUENCESTEP_NUMBER_OF_CHECKS 5  
  
#define VL53L0X_SetParameterField(Dev, field, value) PALDevDataSet(Dev, CurrentParameters.field, value)  
#define VL53L0X_GetParameterField(Dev, field, variable) variable = PALDevDataGet(Dev, CurrentParameters.field)  
#define VL53L0X_SetArrayParameterField(Dev, field, index, value) CurrentParameters.field[index] = value  
#define VL53L0X_GetArrayParameterField(Dev, field, index) PALDevDataGet(Dev, CurrentParameters.field[index])  
#define VL53L0X_SetDeviceSpecificParameter(Dev, field, value) DeviceSpecificParameters.field = value  
#define VL53L0X_GetDeviceSpecificParameter(Dev, field) DeviceSpecificParameters.field  
  
#define VL53L0X_FIXPOINT1616TOFIXPOINT97(Value) (uint16_t)(Value >> 7)  
#define VL53L0X_FIXPOINT97TOFIXPOINT1616(Value) (FixPoint16)((Value << 7) & 0xFFFF)  
#define VL53L0X_FIXPOINT1616TOFIXPOINT88(Value) (uint16_t)((Value >> 8) & 0xFFFF)  
#define VL53L0X_FIXPOINT88TOFIXPOINT1616(Value) (FixPoint16)((Value << 8) & 0xFFFF)
```

```
#define VL53L0X_FIXPOINT1616TOFIXPOINT412(Value) (uint16_t)(Value)

#define VL53L0X_FIXPOINT412TOFIXPOINT1616(Value) (FixPoint16)(Value)

#define VL53L0X_FIXPOINT1616TOFIXPOINT313(Value) (uint16_t)(Value)

#define VL53L0X_FIXPOINT313TOFIXPOINT1616(Value) (FixPoint16)(Value)

#define VL53L0X_FIXPOINT1616TOFIXPOINT08(Value) (uint8_t)((Value) >> 8)

#define VL53L0X_FIXPOINT08TOFIXPOINT1616(Value) (FixPoint16)(Value)

#define VL53L0X_FIXPOINT1616TOFIXPOINT53(Value) (uint8_t)((Value) << 8)

#define VL53L0X_FIXPOINT53TOFIXPOINT1616(Value) (FixPoint16)(Value)

#define VL53L0X_FIXPOINT1616TOFIXPOINT102(Value) (uint16_t)((Value) << 2)

#define VL53L0X_FIXPOINT102TOFIXPOINT1616(Value) (FixPoint16)(Value)

#define VL53L0X_MAKEUINT16(lsb, msb)
```

Typedefs

```
typedef int8_t VL53L0X_Error
```

```
typedef uint8_t VL53L0X_DeviceModes
```

```
typedef uint8_t VL53L0X_HistogramModes
```

```
typedef uint8_t VL53L0X_PowerModes
```

```
typedef uint8_t VL53L0X_State
```

```
typedef uint8_t VL53L0X_InterruptPolarity
```

```
typedef uint8_t VL53L0X_VcselPeriod
```

```
typedef uint8_t VL53L0X_SequenceStepId
```

Detailed Description

Type definitions for VL53L0X API.

Definition in file **vl53l0x_def.h**.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			Macros Typedefs

vl53l0x_device.h File Reference

```
#include "vl53l0x_types.h"
```

Go to the source code of this file.

Macros

```
#define VL53L0X_DEVICEERROR_NONE ((VL53L0X_DeviceError)

#define VL53L0X_DEVICEERROR_VCSELCONTINUITYTESTFAILU

#define VL53L0X_DEVICEERROR_VCSELWATCHDOGTESTFAILU

#define VL53L0X_DEVICEERROR_NOVHVVALUEFOUND ((VL53L

#define VL53L0X_DEVICEERROR_MSRCNOTARGET ((VL53L0X_

#define VL53L0X_DEVICEERROR_SNRCHECK ((VL53L0X_Device

#define VL53L0X_DEVICEERROR_RANGEPHASECHECK ((VL53L

#define VL53L0X_DEVICEERROR_SIGMATHRESHOLDCHECK ((\

#define VL53L0X_DEVICEERROR_TCC ((VL53L0X_DeviceError) {

#define VL53L0X_DEVICEERROR_PHASECONSISTENCY ((VL53L

#define VL53L0X_DEVICEERROR_MINCLIP ((VL53L0X_DeviceEri

#define VL53L0X_DEVICEERROR_RANGECOMPLETE ((VL53L0X

#define VL53L0X_DEVICEERROR_ALGOUNDERFLOW ((VL53L0X

#define VL53L0X_DEVICEERROR_ALGOOVERFLOW ((VL53L0X

#define VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD (0

#define VL53L0X_CHECKENABLE_SIGMA_FINAL_RANGE 0

#define VL53L0X_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE
```

```
#define VL53L0X_CHECKENABLE_SIGNAL_REF_CLIP 2

#define VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD 1

#define VL53L0X_CHECKENABLE_SIGNAL_RATE_MSRC 4

#define VL53L0X_CHECKENABLE_SIGNAL_RATE_PRE_RANGE 5

#define VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS 6

#define VL53L0X_GPIOFUNCTIONALITY_OFF ((VL53L0X_GpioFunctionality)<=0)

#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_1) 1

#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_2) 2

#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_3) 3

#define VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY 0x00000001

#define VL53L0X_REG_SYSRANGE_START 0x000

#define VL53L0X_REG_SYSRANGE_MODE_MASK 0x0F
mask existing bit in VL53L0X_REG_SYSRANGE_START More...

#define VL53L0X_REG_SYSRANGE_MODE_START_STOP 0x01
bit 0 in VL53L0X_REG_SYSRANGE_START write 1 toggle start
shot in single shot mode More...

#define VL53L0X_REG_SYSRANGE_MODE_SINGLESHT 0x00
bit 1 write 0 in VL53L0X_REG_SYSRANGE_START set single shot mode More...

#define VL53L0X_REG_SYSRANGE_MODE_BACKTOBACK 0x02
bit 1 write 1 in VL53L0X_REG_SYSRANGE_START set back-to-back shot mode More...
```

```
#define VL53L0X_REG_SYSRANGE_MODE_TIMED 0x04
bit 2 write 1 in VL53L0X_REG_SYSRANGE_START set timed mode

#define VL53L0X_REG_SYSRANGE_MODE_HISTOGRAM 0x08
bit 3 write 1 in VL53L0X_REG_SYSRANGE_START set histogram mode

#define VL53L0X_REG_SYSTEM_THRESH_HIGH 0x000C
#define VL53L0X_REG_SYSTEM_THRESH_LOW 0x000E

#define VL53L0X_REG_SYSTEM_SEQUENCE_CONFIG 0x0001

#define VL53L0X_REG_SYSTEM_RANGE_CONFIG 0x0009

#define VL53L0X_REG_SYSTEM_INTERMEASUREMENT_PERIOD 0x000A

#define VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO 0x000B
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED 0x000B
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW 0x000B
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH 0x000B
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW 0x000B
#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_READY 0x000B

#define VL53L0X_REG_GPIO_HV_MUX_ACTIVE_HIGH 0x0084

#define VL53L0X_REG_SYSTEM_INTERRUPT_CLEAR 0x000B

#define VL53L0X_REG_RESULT_INTERRUPT_STATUS 0x0013

#define VL53L0X_REG_RESULT_RANGE_STATUS 0x0014
```

```
#define VL53L0X_REG_RESULT_CORE_PAGE 1

#define VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVE

#define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENT1

#define VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVE

#define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENT1

#define VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF 0x00000000

#define VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET 0x00000000

#define VL53L0X_REG_I2C_SLAVE_DEVICE_ADDRESS 0x008a

#define VL53L0X_REG_MSRC_CONFIG_CONTROL 0x0060

#define VL53L0X_REG_PRE_RANGE_CONFIG_MIN_SNR 0X0027

#define VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_LOW

#define VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HIGH

#define VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LI

#define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR 0X0060

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_LOW

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_HIGH

#define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RA

#define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_HI

#define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_L
```

```
#define VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD 0x0001

#define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACRO

#define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACRO

#define VL53L0X_REG_SYSTEM_HISTOGRAM_BIN 0x0081

#define VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SI

#define VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VCSEL_PERIOD

#define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACRO

#define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACRO

#define VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RA

#define VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP 0x0001

#define VL53L0X_REG_SOFT_RESET_G02_SOFT_RESET_N 0x0001

#define VL53L0X_REG_IDENTIFICATION_MODEL_ID 0x00c0

#define VL53L0X_REG_IDENTIFICATION_REVISION_ID 0x00c2

#define VL53L0X_REG_OSC_CALIBRATE_VAL 0x00f8

#define VL53L0X_SIGMA_ESTIMATE_MAX_VALUE 65535

#define VL53L0X_REG_GLOBAL_CONFIG_VCSEL_WIDTH 0x032

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF
```

```
#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_0 0x0000 /* 0x100 *  
#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1 0x0001 /* 0x101 *  
#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2 0x0002 /* 0x102 *  
#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3 0x0003 /* 0x103 *  
#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4 0x0004 /* 0x104 *  
#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5 0x0005 /* 0x105 *  
#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_6 0x0006 /* 0x106 *  
#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_7 0x0007 /* 0x107 *  
#define VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELE 0x0008 /* 0x108 *  
#define VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF 0x0009 /* 0x109 *  
#define VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET 0x000A /* 0x10A *  
#define VL53L0X_REG_POWER_MANAGEMENT_GO1_POWER_FON 0x000B /* 0x10B *  
#define VL53L0X_SPEED_OF_LIGHT_IN_AIR 2997  
#define VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA__EXTSUP__VHV 0x000C /* 0x10C *  
#define VL53L0X_REG_ALGO_PHASECAL_LIM 0x0030 /* 0x130 */  
#define VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT 0x0031 /* 0x131 */
```

Typedefs

```
typedef uint8_t VL53L0X_DeviceError
```

```
typedef uint8_t VL53L0X_GpioFunctionality
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
doc >			

vl53l0x_doxydoc.c File Reference

[Go to the source code of this file.](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	platform inc

[Macros](#) | [Typedefs](#) | [Functions](#)

vl53l0x_i2c_platform.h

File Reference

```
#include "vl53l0x_def.h" #include <stdint.h>
#include <stdarg.h>
```

[Go to the source code of this file.](#)

Macros

```
#define I2C 0x01
```

```
#define SPI 0x00
```

```
#define COMMS_BUFFER_SIZE 64
```

```
#define BYTES_PER_WORD 2
```

```
#define BYTES_PER_DWORD 4
```

```
#define VL53L0X_MAX_STRING_LENGTH_PLT 256
```

Typedefs

typedef unsigned char **bool_t**

TypeDef defining . [More...](#)

Functions

int32_t **VL53L0X_comms_initialise** (**uint8_t** comms_type, **uint16_t** comms_speed_khz)
Initialise platform comms. [More...](#)

int32_t **VL53L0X_comms_close** (**void**)
Close platform comms. [More...](#)

int32_t **VL53L0X_cycle_power** (**void**)
Cycle Power to Device. [More...](#)

int32_t **VL53L0X_write_multi** (**uint8_t** address, **uint8_t** index, **uint8_t** *pdata, **int32_t** count)
Writes the supplied byte buffer to the device. [More...](#)

int32_t **VL53L0X_read_multi** (**uint8_t** address, **uint8_t** index, **uint8_t** *pdata, **int32_t** count)
Reads the requested number of bytes from the device.
[More...](#)

int32_t **VL53L0X_write_byte** (**uint8_t** address, **uint8_t** index, **uint8_t** data)
Writes a single byte to the device. [More...](#)

int32_t **VL53L0X_write_word** (**uint8_t** address, **uint8_t** index, **uint16_t** data)
Writes a single word (16-bit unsigned) to the device. [More...](#)

int32_t **VL53L0X_write_dword** (**uint8_t** address, **uint8_t** index, **uint32_t** data)
Writes a single dword (32-bit unsigned) to the device.
[More...](#)

int32_t **VL53L0X_read_byte** (**uint8_t** address, **uint8_t** index, **uint8_t** *pdata)

Reads a single byte from the device. More...

int32_t **VL53L0X_read_word** (**uint8_t** address, **uint8_t** index, **uint16_t** *pdata)

Reads a single word (16-bit unsigned) from the device.
More...

int32_t **VL53L0X_read_dword** (**uint8_t** address, **uint8_t** index, **uint32_t** *pdata)

Reads a single dword (32-bit unsigned) from the device.
More...

int32_t **VL53L0X_platform_wait_us** (**int32_t** wait_us)

Implements a programmable wait in us. More...

int32_t **VL53L0X_wait_ms** (**int32_t** wait_ms)

Implements a programmable wait in ms. More...

int32_t **VL53L0X_set_gpio** (**uint8_t** level)

Set GPIO value. More...

int32_t **VL53L0X_get_gpio** (**uint8_t** *plevel)

Get GPIO value. More...

int32_t **VL53L0X_release_gpio** (**void**)

Release force on GPIO. More...

int32_t **VL53L0X_get_timer_frequency** (**int32_t** *ptimer_freq_hz)

Get the frequency of the timer used for ranging results time stamps. More...

int32_t **VL53L0X_get_timer_value** (**int32_t** *ptimer_count)

Get the timer value in units of timer_freq_hz (see
VL53L0X_get_timestamp_frequency()) More...

Macro Definition Documentation

```
#define I2C 0x01
```

Definition at line [55](#) of file `vl53l0x_i2c_platform.h`.

```
#define SPI 0x00
```

Definition at line [56](#) of file `vl53l0x_i2c_platform.h`.

```
#define COMMS_BUFFER_SIZE 64
```

Definition at line [58](#) of file `vl53l0x_i2c_platform.h`.

```
#define BYTES_PER_WORD 2
```

Definition at line [60](#) of file `vl53l0x_i2c_platform.h`.

```
#define BYTES_PER_DWORD 4
```

Definition at line [61](#) of file `vl53l0x_i2c_platform.h`.

```
#define VL53L0X_MAX_STRING_LENGTH_PLT 256
```

Definition at line [63](#) of file `vl53l0x_i2c_platform.h`.

Typedef Documentation

```
typedef unsigned char bool_t
```

Typedef defining .

The developer shoud modify this to suit the platform being deployed.

Typedef defining 8 bit unsigned char type.

The developer shoud modify this to suit the platform being deployed.

Definition at line **51** of file **vl53l0x_i2c_platform.h**.

Function Documentation

`int32_t`

`VL53L0X_comms_initialise (uint8_t comms_type,
 uint16_t comms_speed_khz
)`

Initialise platform comms.

Parameters

`comms_type` - selects between I2C and SPI

`comms_speed_khz` - unsigned short containing the I2C speed
in kHz

Returns

status - status 0 = ok, 1 = error

`int32_t VL53L0X_comms_close (void)`

Close platform comms.

Returns

status - status 0 = ok, 1 = error

`int32_t VL53L0X_cycle_power (void)`

Cycle Power to Device.

Returns

status - status 0 = ok, 1 = error

```
int32_t VL53L0X_write_multi( uint8_t  address,
                             uint8_t  index,
                             uint8_t * pdata,
                             int32_t   count
                           )
```

Writes the supplied byte buffer to the device.

Wrapper for SystemVerilog Write Multi task

```
1 Example:
2
3 uint8_t *spad_enables;
4
5 int status =
VL53L0X_write_multi(RET_SPAD_EN_0,
                     spad_enables, 36);
```

Parameters

- address** - uint8_t device address value
- index** - uint8_t register index value
- pdata** - pointer to uint8_t buffer containing the data to be written
- count** - number of bytes in the supplied byte buffer

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_read_multi( uint8_t  address,
                             uint8_t  index,
                             uint8_t * pdata,
                             int32_t   count
                           )
```

Reads the requested number of bytes from the device.

Wrapper for SystemVerilog Read Multi task

```
1 Example:  
2  
3 uint8_t buffer[COMMS_BUFFER_SIZE];  
4  
5 int status = status =  
VL53L0X_read_multi(DEVICE_ID, buffer, 2)
```

Parameters

address - uint8_t device address value
index - uint8_t register index value
pdata - pointer to the uint8_t buffer to store read data
count - number of uint8_t's to read

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_write_byte ( uint8_t address,  
                           uint8_t index,  
                           uint8_t data  
                         )
```

Writes a single byte to the device.

Wrapper for SystemVerilog Write Byte task

```
1 Example:  
2  
3 uint8_t page_number = MAIN_SELECT_PAGE;  
4  
5 int status =  
VL53L0X_write_byte(PAGE_SELECT,  
page_number);
```

Parameters

address - uint8_t device address value
index - uint8_t register index value
data - uint8_t data value to write

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_write_word( uint8_t address,  
                           uint8_t index,  
                           uint16_t data  
                         )
```

Writes a single word (16-bit unsigned) to the device.

Manages the big-endian nature of the device (first byte written is the MS byte). Uses SystemVerilog Write Multi task.

```
1 Example:  
2  
3 uint16_t nvm_ctrl_pulse_width = 0x0004;  
4  
5 int status =  
  VL53L0X_write_word(NVM_CTRL__PULSE_WIDTH_MSB,  
  nvm_ctrl_pulse_width);
```

Parameters

address - uint8_t device address value
index - uint8_t register index value
data - uint16_t data value write

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_write_dword( uint8_t address,  
                           uint8_t index,
```

```
        uint32_t data  
    )
```

Writes a single dword (32-bit unsigned) to the device.

Manages the big-endian nature of the device (first byte written is the MS byte). Uses SystemVerilog Write Multi task.

```
1 Example:  
2  
3 uint32_t nvm_data = 0x0004;  
4  
5 int status =  
    VL53L0X_write_dword(NVM_CTRL__DATAIN_MMM,  
    nvm_data);
```

Parameters

address - uint8_t device address value
index - uint8_t register index value
data - uint32_t data value to write

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_read_byte ( uint8_t  address,  
                            uint8_t  index,  
                            uint8_t * pdata  
    )
```

Reads a single byte from the device.

Uses SystemVerilog Read Byte task.

```
1 Example:  
2  
3 uint8_t device_status = 0;
```

```
4  
5 int status = VL53L0X_read_byte(STATUS,  
&device_status);
```

Parameters

- address** - uint8_t device address value
- index** - uint8_t register index value
- pdata** - pointer to uint8_t data value

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_read_word ( uint8_t    address,  
                            uint8_t    index,  
                            uint16_t * pdata  
                          )
```

Reads a single word (16-bit unsigned) from the device.

Manages the big-endian nature of the device (first byte read is the MS byte). Uses SystemVerilog Read Multi task.

```
1 Example:  
2  
3 uint16_t timeout = 0;  
4  
5 int status =  
VL53L0X_read_word(TIMEOUT_OVERALL_PERIODS_MSB,  
&timeout);
```

Parameters

- address** - uint8_t device address value
- index** - uint8_t register index value
- pdata** - pointer to uint16_t data value

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_read_dword( uint8_t    address,
                            uint8_t    index,
                            uint32_t * pdata
                        )
```

Reads a single dword (32-bit unsigned) from the device.

Manages the big-endian nature of the device (first byte read is the MS byte). Uses SystemVerilog Read Multi task.

```
1 Example:  
2  
3 uint32_t range_1 = 0;  
4  
5 int status =  
VL53L0X_read_dword(RANGE_1_MMM, &range_1);
```

Parameters

address - uint8_t device address value
index - uint8_t register index value
pdata - pointer to uint32_t data value

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_platform_wait_us( int32_t wait_us )
```

Implements a programmable wait in us.

Wrapper for SystemVerilog Wait in micro seconds task

Parameters

wait_us - integer wait in micro seconds

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_wait_ms ( int32_t wait_ms )
```

Implements a programmable wait in ms.

Wrapper for SystemVerilog Wait in milli seconds task

Parameters

wait_ms - integer wait in milli seconds

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_set_gpio ( uint8_t level )
```

Set GPIO value.

Parameters

level - input level - either 0 or 1

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_get_gpio ( uint8_t * plevel )
```

Get GPIO value.

Parameters

plevel - uint8_t pointer to store GPIO level (0 or 1)

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t VL53L0X_release_gpio ( void )
```

Release force on GPIO.

Returns

status - SystemVerilog status 0 = ok, 1 = error

```
int32_t  
VL53L0X_get_timer_frequency ( int32_t * ptimer_freq_hz )
```

Get the frequency of the timer used for ranging results time stamps.

Parameters

[out] **ptimer_freq_hz** : pointer for timer frequency

Returns

status : 0 = ok, 1 = error

```
int32_t VL53L0X_get_timer_value ( int32_t * ptimer_count )
```

Get the timer value in units of timer_freq_hz (see
VL53L0X_get_timestamp_frequency())

Parameters

[out] **ptimer_count** : pointer for timer count value

Returns

status : 0 = ok, 1 = error



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	core
			inc

Variables

vl53l0x_interrupt_threshold_settings.h File Reference

Go to the source code of this file.

Variables

`uint8_t InterruptThresholdSettings []`

Variable Documentation

uint8_t InterruptThresholdSettings[]

Definition at line [39](#) of file [vl53l0x_interrupt_threshold_settings.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	platform inc

[Data Structures](#) | [Macros](#) | [Typedefs](#) | [Functions](#)

vl53l0x_platform.h File Reference

Function prototype definitions for Ewok Platform layer. [More...](#)

```
#include "vl53l0x_def.h" #include "vl53l0x_platform_log.h"  
#include "vl53l0x_i2c_platform.h"
```

[Go to the source code of this file.](#)

Data Structures

struct **VL53L0X_Dev_t**

Generic PAL device type that does link between API and platform abstraction layer. [More...](#)

Macros

```
#define PALDevDataGet(Dev, field)  (Dev->Data.field)  
Get ST private structure VL53L0X_DevData_t data access.  
More...
```

```
#define PALDevDataSet(Dev, field, data)  (Dev->Data.field)=(data)  
Set ST private structure VL53L0X_DevData_t data field.  
More...
```

TypeDefs

`typedef VL53L0X_Dev_t * VL53L0X_DEV`

Declare the device Handle as a pointer
of the structure **`VL53L0X_Dev_t`**. More...

Functions

VL53L0X_Error	VL53L0X_LockSequenceAccess (VL53L0X_DEV Dev) Lock comms interface to serialize all commands to a shared I2C interface for a specific device. More...
VL53L0X_Error	VL53L0X_UnlockSequenceAccess (VL53L0X_DEV Dev) Unlock comms interface to serialize all commands to a shared I2C interface for a specific device. More...
VL53L0X_Error	VL53L0X_WriteMulti (VL53L0X_DEV Dev, uint8_t index, uint8_t *pdata, uint32_t count) Writes the supplied byte buffer to the device. More...
VL53L0X_Error	VL53L0X_ReadMulti (VL53L0X_DEV Dev, uint8_t index, uint8_t *pdata, uint32_t count) Reads the requested number of bytes from the device. More...
VL53L0X_Error	VL53L0X_WrByte (VL53L0X_DEV Dev, uint8_t index, uint8_t data) Write single byte register. More...
VL53L0X_Error	VL53L0X_WrWord (VL53L0X_DEV Dev, uint8_t index, uint16_t data) Write word register. More...
VL53L0X_Error	VL53L0X_WrDWord (VL53L0X_DEV Dev, uint8_t index, uint32_t data) Write double word (4 byte) register. More...
VL53L0X_Error	VL53L0X_RdByte (VL53L0X_DEV Dev, uint8_t

index, `uint8_t` *data)
Read single byte register. More...

VL53L0X_Error **VL53L0X_RdWord** (**VL53L0X_DEV** Dev, `uint8_t` index, `uint16_t` *data)
Read word (2byte) register. More...

VL53L0X_Error **VL53L0X_RdDWord** (**VL53L0X_DEV** Dev, `uint8_t` index, `uint32_t` *data)
Read dword (4byte) register. More...

VL53L0X_Error **VL53L0X_UpdateByte** (**VL53L0X_DEV** Dev, `uint8_t` index, `uint8_t` AndData, `uint8_t` OrData)
Thread safe Update (read/modify/write) single byte register. More...

VL53L0X_Error **VL53L0X_PollingDelay** (**VL53L0X_DEV** Dev)
execute delay in all polling API call More...

Detailed Description

Function prototype definitions for Ewok Platform layer.

All end user OS/platform/application porting.

Definition in file **vl53l0x_platform.h**.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	platform inc

[Macros](#) | [Enumerations](#)

vl53l0x_platform_log.h File Reference

platform log function definition [More...](#)

#include <stdio.h> #include <string.h>

[Go to the source code of this file.](#)

Macros

```
#define VL53L0X_ErrLog(...) (void)0
```

```
#define _LOG_FUNCTION_START(module, fmt, ...) (void)0
```

```
#define _LOG_FUNCTION_END(module, status, ...) (void)0
```

```
#define _LOG_FUNCTION_END_FMT(module, status, fmt,  
...) (void)0
```

```
#define VL53L0X_COPYSTRING(str, ...) strcpy(str,  
##__VA_ARGS__)
```

Enumerations

```
enum {  
    TRACE_LEVEL_NONE, TRACE_LEVEL_ERRORS,  
    TRACE_LEVEL_WARNING, TRACE_LEVEL_INFO,  
    TRACE_LEVEL_DEBUG, TRACE_LEVEL_ALL,  
    TRACE_LEVEL_IGNORE  
}  
  
enum { TRACE_FUNCTION_NONE = 0, TRACE_FUNCTION_I2C =  
1, TRACE_FUNCTION_ALL = 0x7fffffff }  
  
enum { TRACE_MODULE_NONE = 0x0, TRACE_MODULE_API =  
0x1, TRACE_MODULE_PLATFORM = 0x2,  
TRACE_MODULE_ALL = 0x7fffffff }
```

Detailed Description

platform log function definition

Definition in file **vl53l0x_platform_log.h**.

Macro Definition Documentation

```
#define VL53L0X_ErrLog( ... ) (void)0
```

Definition at line [103](#) of file `vl53l0x_platform_log.h`.

```
#define _LOG_FUNCTION_START( module,  
                           fmt,  
                           ...  
                           )   (void)0
```

Definition at line [104](#) of file `vl53l0x_platform_log.h`.

```
#define _LOG_FUNCTION_END( module,  
                         status,  
                         ...  
                         )   (void)0
```

Definition at line [105](#) of file `vl53l0x_platform_log.h`.

```
#define _LOG_FUNCTION_END_FMT( module,  
                            status,  
                            fmt,  
                            ...  
                            )   (void)0
```

Definition at line [106](#) of file `vl53l0x_platform_log.h`.

```
#define
```

```
VL53L0X_COPYSTRING      ( str,  
    ...  
)  strcpy(str, ##__VA_ARGS__)
```

Definition at line [109](#) of file [vl53l0x_platform_log.h](#).

Enumeration Type Documentation

anonymous enum

Enumerator
TRACE_LEVEL_NONE
TRACE_LEVEL_ERRORS
TRACE_LEVEL_WARNING
TRACE_LEVEL_INFO
TRACE_LEVEL_DEBUG
TRACE_LEVEL_ALL
TRACE_LEVEL_IGNORE

Definition at line [49](#) of file [vl53l0x_platform_log.h](#).

anonymous enum

Enumerator
TRACE_FUNCTION_NONE
TRACE_FUNCTION_I2C
TRACE_FUNCTION_ALL

Definition at line [59](#) of file [vl53l0x_platform_log.h](#).

anonymous enum

Enumerator
TRACE_MODULE_NONE
TRACE_MODULE_API
TRACE_MODULE_PLATFORM

`TRACE_MODULE_ALL`

Definition at line [65](#) of file `vl53l0x_platform_log.h`.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	core
			inc

Variables

vl53l0x_tuning.h File Reference

```
#include "vl53l0x_def.h"
```

Go to the source code of this file.

Variables

`uint8_t DefaultTuningSettings []`

Variable Documentation

`uint8_t DefaultTuningSettings[]`

Definition at line [41](#) of file `vl53l0x_tuning.h`.

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	platform inc

TypeDefs

vl53l0x_types.h File Reference

VL53L0X types definition. [More...](#)

```
#include <stdint.h> #include <stddef.h>
```

[Go to the source code of this file.](#)

Typedefs

typedef **uint32_t** **FixPoint1616_t**

use where fractional values are expected [More...](#)

typedef unsigned long long **uint64_t**

typedef unsigned int **uint32_t**

Typedef defining 32 bit unsigned int type. [More...](#)

typedef int **int32_t**

Typedef defining 32 bit int type. [More...](#)

typedef unsigned short **uint16_t**

Typedef defining 16 bit unsigned short type. [More...](#)

typedef short **int16_t**

Typedef defining 16 bit short type. [More...](#)

typedef unsigned char **uint8_t**

Typedef defining 8 bit unsigned char type. [More...](#)

typedef signed char **int8_t**

Typedef defining 8 bit char type. [More...](#)

Detailed Description

VL53L0X types definition.

Definition in file **vl53l0x_types.h**.

Typedef Documentation

typedef unsigned long long uint64_t

Definition at line [69](#) of file [vl53l0x_types.h](#).

typedef unsigned int uint32_t

Typedef defining 32 bit unsigned int type.

The developer should modify this to suit the platform being deployed.

Definition at line [75](#) of file [vl53l0x_types.h](#).

typedef int int32_t

Typedef defining 32 bit int type.

The developer should modify this to suit the platform being deployed.

Definition at line [80](#) of file [vl53l0x_types.h](#).

typedef unsigned short uint16_t

Typedef defining 16 bit unsigned short type.

The developer should modify this to suit the platform being

deployed.

Definition at line [85](#) of file `vl53l0x_types.h`.

typedef short int16_t

Typedef defining 16 bit short type.

The developer should modify this to suit the platform being deployed.

Definition at line [90](#) of file `vl53l0x_types.h`.

typedef unsigned char uint8_t

Typedef defining 8 bit unsigned char type.

The developer should modify this to suit the platform being deployed.

Definition at line [95](#) of file `vl53l0x_types.h`.

typedef signed char int8_t

Typedef defining 8 bit char type.

The developer should modify this to suit the platform being deployed.

Definition at line [100](#) of file `vl53l0x_types.h`.

typedef uint32_t FixPoint1616_t

use where fractional values are expected

Given a floating point value f it's .16 bit point is $(int)(f*(1<<16))$

Definition at line [109](#) of file [vl53l0x_types.h](#).

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c d f g i p s t u	v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- - -

- _LOG_FUNCTION_END : [vl53l0x_platform_log.h](#)
- _LOG_FUNCTION_END_FMT : [vl53l0x_platform_log.h](#)
- _LOG_FUNCTION_START : [vl53l0x_platform_log.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c d f g i p s t u	v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- b -

- bool_t : [vl53l0x_i2c_platform.h](#)
- BYTES_PER_DWORD : [vl53l0x_i2c_platform.h](#)
- BYTES_PER_WORD : [vl53l0x_i2c_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
_	b	c	d f g i p s t u v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- C -

- COMMS_BUFFER_SIZE : [vl53l0x_i2c_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c	d
	f	g	i
	p	s	t
	u	v	

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- d -

- DefaultTuningSettings : [vl53l0x_tuning.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c d f g i p s t u	v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- **f** -

- FixPoint1616_t : [vl53l0x_types.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
_	b	c d f g	i p s t u v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- g -

- get_sequence_step_timeout() : [vl53l0x_api_core.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c d f g i p s t u	v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- i -

- I2C : [vl53l0x_i2c_platform.h](#)
- int16_t : [vl53l0x_types.h](#)
- int32_t : [vl53l0x_types.h](#)
- int8_t : [vl53l0x_types.h](#)
- InterruptThresholdSettings :
[vl53l0x_interrupt_threshold_settings.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c d f g i p s t u	v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- p -

- PALDevDataGet : [vl53l0x_platform.h](#)
- PALDevDataSet : [vl53l0x_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
_	b	c d f g i p s t u	v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- S -

- set_sequence_step_timeout() : [vl53l0x_api_core.h](#)
- SPI : [vl53l0x_i2c_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b c d f g i p s	t	u v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- t -

- TRACE_FUNCTION_ALL : [vl53l0x_platform_log.h](#)
- TRACE_FUNCTION_I2C : [vl53l0x_platform_log.h](#)
- TRACE_FUNCTION_NONE : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_ALL : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_DEBUG : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_ERRORS : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_IGNORE : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_INFO : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_NONE : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_WARNING : [vl53l0x_platform_log.h](#)
- TRACE_MODULE_ALL : [vl53l0x_platform_log.h](#)
- TRACE_MODULE_API : [vl53l0x_platform_log.h](#)
- TRACE_MODULE_NONE : [vl53l0x_platform_log.h](#)
- TRACE_MODULE_PLATFORM : [vl53l0x_platform_log.h](#)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c d f g i p s t u	v

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- **u** -

- uint16_t : [vl53l0x_types.h](#)
- uint32_t : [vl53l0x_types.h](#)
- uint64_t : [vl53l0x_types.h](#)
- uint8_t : [vl53l0x_types.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b c d f g i p s t u	v	

Here is a list of all functions, variables, defines, enums, and typedefs with links to the files they belong to:

- v -

- VL53L0X10_IMPLEMENTATION_VER_MAJOR : [vl53l0x_def.h](#)
- VL53L0X10_IMPLEMENTATION_VER_MINOR : [vl53l0x_def.h](#)
- VL53L0X10_IMPLEMENTATION_VER_REVISION : [vl53l0x_def.h](#)
- VL53L0X10_IMPLEMENTATION_VER_SUB : [vl53l0x_def.h](#)
- VL53L0X10_SPECIFICATION_VER_MAJOR : [vl53l0x_def.h](#)
- VL53L0X10_SPECIFICATION_VER_MINOR : [vl53l0x_def.h](#)
- VL53L0X10_SPECIFICATION_VER_REVISION : [vl53l0x_def.h](#)
- VL53L0X10_SPECIFICATION_VER_SUB : [vl53l0x_def.h](#)
- VL53L0X_API : [vl53l0x_api.h](#)
- VL53L0X_apply_offset_adjustment() : [vl53l0x_api_calibration.h](#)
- VL53L0X_calc_sigma_estimate() : [vl53l0x_api_core.h](#)
- VL53L0X_calc_timeout_mclks() : [vl53l0x_api_core.h](#)
- VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS : [vl53l0x_device.h](#)
- VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD : [vl53l0x_device.h](#)
- VL53L0X_CHECKENABLE_SIGMA_FINAL_RANGE : [vl53l0x_device.h](#)
- VL53L0X_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE :

vl53l0x_device.h

- VL53L0X_CHECKENABLE_SIGNAL_RATE_MSRC :
vl53l0x_device.h
- VL53L0X_CHECKENABLE_SIGNAL_RATE_PRE_RANGE :
vl53l0x_device.h
- VL53L0X_CHECKENABLE_SIGNAL_REF_CLIP :
vl53l0x_device.h
- VL53L0X_ClearInterruptMask() : **vl53l0x_api.h**
- VL53L0X_comms_close() : **vl53l0x_i2c_platform.h**
- VL53L0X_comms_initialise() : **vl53l0x_i2c_platform.h**
- VL53L0X_COPYSTRING : **vl53l0x_platform_log.h**
- VL53L0X_cycle_power() : **vl53l0x_i2c_platform.h**
- VL53L0X_DataInit() : **vl53l0x_api.h**
- VL53L0X_decode_timeout() : **vl53l0x_api_core.h**
- VL53L0X_decode_vcsel_period() : **vl53l0x_api_core.h**
- VL53L0X_DEFAULT_MAX_LOOP : **vl53l0x_def.h**
- VL53L0X_DEV : **vl53l0x_platform.h**
- VL53L0X_DeviceError : **vl53l0x_device.h**
- VL53L0X_DEVICEERROR_ALGOOVERFLOW :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_ALGOUNDERFLOW :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_MINCLIP : **vl53l0x_device.h**
- VL53L0X_DEVICEERROR_MSRCNOTARGET :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_NONE : **vl53l0x_device.h**
- VL53L0X_DEVICEERROR_NOVHVVALUEFOUND :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_PHASECONSISTENCY :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_RANGECOMPLETE :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_RANGEPHASECHECK :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_SIGMATHRESHOLDCHECK :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_SNRCHECK : **vl53l0x_device.h**
- VL53L0X_DEVICEERROR_TCC : **vl53l0x_device.h**

- VL53L0X_DEVICEERROR_VSELCONTINUITYTESTFAILURE : [vl53l0x_device.h](#)
- VL53L0X_DEVICEERROR_VSELWATCHDOGTESTFAILURE : [vl53l0x_device.h](#)
- VL53L0X_DEVICEMODE_CONTINUOUS_RANGING : [vl53l0x_def.h](#)
- VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING : [vl53l0x_def.h](#)
- VL53L0X_DEVICEMODE_GPIO_DRIVE : [vl53l0x_def.h](#)
- VL53L0X_DEVICEMODE_GPIO_OSC : [vl53l0x_def.h](#)
- VL53L0X_DEVICEMODE_SINGLE_ALS : [vl53l0x_def.h](#)
- VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM : [vl53l0x_def.h](#)
- VL53L0X_DEVICEMODE_SINGLE_RANGING : [vl53l0x_def.h](#)
- VL53L0X_DeviceModes : [vl53l0x_def.h](#)
- VL53L0X_EnableInterruptMask() : [vl53l0x_api.h](#)
- VL53L0X_encode_timeout() : [vl53l0x_api_core.h](#)
- VL53L0X_encode_vsel_period() : [vl53l0x_api_core.h](#)
- VL53L0X_ErrLog : [vl53l0x_platform_log.h](#)
- VL53L0X_Error : [vl53l0x_def.h](#)
- VL53L0X_ERROR_BUFFER_TOO_SMALL : [vl53l0x_def.h](#)
- VL53L0X_ERROR_CALIBRATION_WARNING : [vl53l0x_def.h](#)
- VL53L0X_ERROR_CONTROL_INTERFACE : [vl53l0x_def.h](#)
- VL53L0X_ERROR_DIVISION_BY_ZERO : [vl53l0x_def.h](#)
- VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_GPIO_NOT_EXISTING : [vl53l0x_def.h](#)
- VL53L0X_ERROR_INTERRUPT_NOT_CLEARED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_INVALID_COMMAND : [vl53l0x_def.h](#)
- VL53L0X_ERROR_INVALID_PARAMS : [vl53l0x_def.h](#)
- VL53L0X_ERROR_MIN_CLIPPED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_MODE_NOT_SUPPORTED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_NONE : [vl53l0x_def.h](#)
- VL53L0X_ERROR_NOT_IMPLEMENTED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_NOT_SUPPORTED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_RANGE_ERROR : [vl53l0x_def.h](#)
- VL53L0X_ERROR_REF_SPAD_INIT : [vl53l0x_def.h](#)
- VL53L0X_ERROR_TIME_OUT : [vl53l0x_def.h](#)
- VL53L0X_ERROR_UNDEFINED : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT08TOFIXPOINT1616 : [vl53l0x_def.h](#)

- VL53L0X_FIXPOINT102TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT08 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT102 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT313 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT412 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT53 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT88 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT97 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT313TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT412TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT53TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT88TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT97TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_get_device_error_string() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_device_info() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_gpio() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_get_info_from_device() : [vl53l0x_api_core.h](#)
- VL53L0X_get_limit_check_info() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_measurement_timing_budget_micro_seconds() : [vl53l0x_api_core.h](#)
- VL53L0X_get_offset_calibration_data_micro_meter() : [vl53l0x_api_calibration.h](#)
- VL53L0X_get_pal_error_string() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_pal_range_status() : [vl53l0x_api_core.h](#)
- VL53L0X_get_pal_state_string() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_range_status_string() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_ref_calibration() : [vl53l0x_api_calibration.h](#)
- VL53L0X_get_reference_spads() : [vl53l0x_api_calibration.h](#)
- VL53L0X_get_sequence_steps_info() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_timer_frequency() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_get_timer_value() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_get_total_signal_rate() : [vl53l0x_api_core.h](#)
- VL53L0X_get_total_xtalk_rate() : [vl53l0x_api_core.h](#)
- VL53L0X_get_vcsel_pulse_period() : [vl53l0x_api_core.h](#)
- VL53L0X_GETARRAYPARAMETERFIELD : [vl53l0x_def.h](#)
- VL53L0X_GetDeviceErrorStatus() : [vl53l0x_api.h](#)
- VL53L0X_GetDeviceErrorString() : [vl53l0x_api.h](#)
- VL53L0X_GetDeviceInfo() : [vl53l0x_api.h](#)
- VL53L0X_GetDeviceMode() : [vl53l0x_api.h](#)
- VL53L0X_GetDeviceParameters() : [vl53l0x_api.h](#)

- VL53L0X_GETDEVICESPECIFICPARAMETER : [vl53l0x_def.h](#)
- VL53L0X_GetDmaxCalParameters() : [vl53l0x_api.h](#)
- VL53L0X_GetFractionEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetGpioConfig() : [vl53l0x_api.h](#)
- VL53L0X_GetHistogramMeasurementData() : [vl53l0x_api.h](#)
- VL53L0X_GetHistogramMode() : [vl53l0x_api.h](#)
- VL53L0X_GetInterMeasurementPeriodMilliSeconds() :
[vl53l0x_api.h](#)
- VL53L0X_GetInterruptMaskStatus() : [vl53l0x_api.h](#)
- VL53L0X_GetInterruptThresholds() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckCurrent() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckInfo() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckStatus() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckValue() : [vl53l0x_api.h](#)
- VL53L0X_GetLinearityCorrectiveGain() : [vl53l0x_api.h](#)
- VL53L0X_GetMaxNumberOfROIZones() : [vl53l0x_api.h](#)
- VL53L0X_GetMeasurementDataReady() : [vl53l0x_api.h](#)
- VL53L0X_GetMeasurementRefSignal() : [vl53l0x_api.h](#)
- VL53L0X_GetMeasurementTimingBudgetMicroSeconds() :
[vl53l0x_api.h](#)
- VL53L0X_GetNumberOfLimitCheck() : [vl53l0x_api.h](#)
- VL53L0X_GetNumberOfROIZones() : [vl53l0x_api.h](#)
- VL53L0X_GetNumberOfSequenceSteps() : [vl53l0x_api.h](#)
- VL53L0X_GetOffsetCalibrationDataMicroMeter() : [vl53l0x_api.h](#)
- VL53L0X_GetPalErrorString() : [vl53l0x_api.h](#)
- VL53L0X_GetPalSpecVersion() : [vl53l0x_api.h](#)
- VL53L0X_GetPalState() : [vl53l0x_api.h](#)
- VL53L0X_GetPalStateString() : [vl53l0x_api.h](#)
- VL53L0X_GETPARAMETERFIELD : [vl53l0x_def.h](#)
- VL53L0X_GetPowerMode() : [vl53l0x_api.h](#)
- VL53L0X_GetProductRevision() : [vl53l0x_api.h](#)
- VL53L0X_GetRangeStatusString() : [vl53l0x_api.h](#)
- VL53L0X_GetRangingMeasurementData() : [vl53l0x_api.h](#)
- VL53L0X_GetRefCalibration() : [vl53l0x_api.h](#)
- VL53L0X_GetReferenceSpads() : [vl53l0x_api.h](#)
- VL53L0X_GetSequenceStepEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetSequenceStepEnables() : [vl53l0x_api.h](#)
- VL53L0X_GetSequenceStepsInfo() : [vl53l0x_api.h](#)
- VL53L0X_GetSequenceStepTimeout() : [vl53l0x_api.h](#)

- VL53L0X_GetSpadAmbientDamperFactor() : [vl53l0x_api.h](#)
- VL53L0X_GetSpadAmbientDamperThreshold() : [vl53l0x_api.h](#)
- VL53L0X_GetStopCompletedStatus() : [vl53l0x_api.h](#)
- VL53L0X_GetTotalSignalRate() : [vl53l0x_api.h](#)
- VL53L0X_GetTuningSettingBuffer() : [vl53l0x_api.h](#)
- VL53L0X_GetUpperLimitMilliMeter() : [vl53l0x_api.h](#)
- VL53L0X_GetVcselPulsePeriod() : [vl53l0x_api.h](#)
- VL53L0X_GetVersion() : [vl53l0x_api.h](#)
- VL53L0X_GetWrapAroundCheckEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetXTalkCompensationEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetXTalkCompensationRateMegaCps() : [vl53l0x_api.h](#)
- VL53L0X_GpioFunctionality : [vl53l0x_device.h](#)
- VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY : [vl53l0x_device.h](#)
- VL53L0X_GPIOFUNCTIONALITY_OFF : [vl53l0x_device.h](#)
- VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HIGH : [vl53l0x_device.h](#)
- VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_LOW : [vl53l0x_device.h](#)
- VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OUT : [vl53l0x_device.h](#)
- VL53L0X_HISTOGRAM_BUFFER_SIZE : [vl53l0x_def.h](#)
- VL53L0X_HISTOGRAMMODE_BOTH : [vl53l0x_def.h](#)
- VL53L0X_HISTOGRAMMODE_DISABLED : [vl53l0x_def.h](#)
- VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY : [vl53l0x_def.h](#)
- VL53L0X_HISTOGRAMMODE_RETURN_ONLY : [vl53l0x_def.h](#)
- VL53L0X_HistogramModes : [vl53l0x_def.h](#)
- VL53L0X_IMPLEMENTATION_VER_MAJOR : [vl53l0x_def.h](#)
- VL53L0X_IMPLEMENTATION_VER_MINOR : [vl53l0x_def.h](#)
- VL53L0X_IMPLEMENTATION_VER_REVISION : [vl53l0x_def.h](#)
- VL53L0X_IMPLEMENTATION_VER_SUB : [vl53l0x_def.h](#)
- VL53L0X_InterruptPolarity : [vl53l0x_def.h](#)
- VL53L0X_INTERRUPTPOLARITY_HIGH : [vl53l0x_def.h](#)
- VL53L0X_INTERRUPTPOLARITY_LOW : [vl53l0x_def.h](#)
- VL53L0X_isqrt() : [vl53l0x_api_core.h](#)
- VL53L0X_load_tuning_settings() : [vl53l0x_api_core.h](#)
- VL53L0X_LockSequenceAccess() : [vl53l0x_platform.h](#)
- VL53L0X_MAKEUINT16 : [vl53l0x_def.h](#)
- VL53L0X_MAX_STRING_LENGTH : [vl53l0x_def.h](#)

- VL53L0X_MAX_STRING_LENGTH_PLT : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_measurement_poll_for_completion() : [**vl53l0x_api_core.h**](#)
- VL53L0X_perform_offset_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_perform_phase_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_perform_ref_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_perform_ref_spad_management() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_perform_xtalk_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_PerformOffsetCalibration() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformRefCalibration() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformRefSpadManagement() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformSingleHistogramMeasurement() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformSingleMeasurement() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformSingleRangingMeasurement() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformXTalkCalibration() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformXTalkMeasurement() : [**vl53l0x_api.h**](#)
- VL53L0X_platform_wait_us() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_PollingDelay() : [**vl53l0x_platform.h**](#)
- VL53L0X_POWERMODE_IDLE_LEVEL1 : [**vl53l0x_def.h**](#)
- VL53L0X_POWERMODE_IDLE_LEVEL2 : [**vl53l0x_def.h**](#)
- VL53L0X_POWERMODE_STANDBY_LEVEL1 : [**vl53l0x_def.h**](#)
- VL53L0X_POWERMODE_STANDBY_LEVEL2 : [**vl53l0x_def.h**](#)
- VL53L0X_PowerModes : [**vl53l0x_def.h**](#)
- VL53L0X_quadrature_sum() : [**vl53l0x_api_core.h**](#)
- VL53L0X_RdByte() : [**vl53l0x_platform.h**](#)
- VL53L0X_RdDWord() : [**vl53l0x_platform.h**](#)
- VL53L0X_RdWord() : [**vl53l0x_platform.h**](#)
- VL53L0X_read_byte() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_read_dword() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_read_multi() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_read_word() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_ReadMulti() : [**vl53l0x_platform.h**](#)
- VL53L0X_REF_SPAD_BUFFER_SIZE : [**vl53l0x_def.h**](#)
- VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET_MM : [**vl53l0x_device.h**](#)
- VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT :

vl53l0x_device.h

- VL53L0X_REG_ALGO_PHASECAL_LIM : **vl53l0x_device.h**
- VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RATE_MG : **vl53l0x_device.h**
- VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF_SPARE : **vl53l0x_device.h**
- VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET : **vl53l0x_device.h**
- VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RATE_RAMP : **vl53l0x_device.h**
- VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR : **vl53l0x_device.h**
- VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_H : **vl53l0x_device.h**
- VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_L : **vl53l0x_device.h**
- VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_HIGH : **vl53l0x_device.h**
- VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_LOW : **vl53l0x_device.h**
- VL53L0X_REG_FINAL_RANGE_CONFIG_VCSEL_PERIOD : **vl53l0x_device.h**
- VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELECT : **vl53l0x_device.h**
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_0 : **vl53l0x_device.h**
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1 : **vl53l0x_device.h**
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2 : **vl53l0x_device.h**
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3 : **vl53l0x_device.h**
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4 : **vl53l0x_device.h**
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5 : **vl53l0x_device.h**
- VL53L0X_REG_GLOBAL_CONFIG_VCSEL_WIDTH : **vl53l0x_device.h**
- VL53L0X_REG_GPIO_HV_MUX_ACTIVE_HIGH : **vl53l0x_device.h**

- VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SELECT : [vl53l0x_device.h](#)
- VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL : [vl53l0x_device.h](#)
- VL53L0X_REG_I2C_SLAVE_DEVICE_ADDRESS : [vl53l0x_device.h](#)
- VL53L0X_REG_IDENTIFICATION_MODEL_ID : [vl53l0x_device.h](#)
- VL53L0X_REG_IDENTIFICATION_REVISION_ID : [vl53l0x_device.h](#)
- VL53L0X_REG_MSRC_CONFIG_CONTROL : [vl53l0x_device.h](#)
- VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP : [vl53l0x_device.h](#)
- VL53L0X_REG_OSC_CALIBRATE_VAL : [vl53l0x_device.h](#)
- VL53L0X_REG_POWER_MANAGEMENT_GO1_POWER_FORCE : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_CONFIG_MIN_SNR : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_HI : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_LO : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_HI : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_LO : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HIGH : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_LOW : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD : [vl53l0x_device.h](#)
- VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LIMIT : [vl53l0x_device.h](#)
- VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_I : [vl53l0x_device.h](#)
- VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_I : [vl53l0x_device.h](#)
- VL53L0X_REG_RESULT_CORE_PAGE : [vl53l0x_device.h](#)
- VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_RE

: [vl53l0x_device.h](#)

- VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_RT : [vl53l0x_device.h](#)
- VL53L0X_REG_RESULT_INTERRUPT_STATUS : [vl53l0x_device.h](#)
- VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF : [vl53l0x_device.h](#)
- VL53L0X_REG_RESULT_RANGE_STATUS : [vl53l0x_device.h](#)
- VL53L0X_REG_SOFT_RESET_G02_SOFT_RESET_N : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSRANGE_MODE_BACKTOBACK : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSRANGE_MODE_HISTOGRAM : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSRANGE_MODE_MASK : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSRANGE_MODE_SINGLESHOT : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSRANGE_MODE_START_STOP : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSRANGE_MODE_TIMED : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSRANGE_START : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_HISTOGRAM_BIN : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_INTERMEASUREMENT_PERIOD : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_INTERRUPT_CLEAR : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_REF : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_RANGE_CONFIG : [vl53l0x_device.h](#)
- VL53L0X_REG_SYSTEM_SEQUENCE_CONFIG :

vl53l0x_device.h

- VL53L0X_REG_SYSTEM_THRESH_HIGH : **vl53l0x_device.h**
- VL53L0X_REG_SYSTEM_THRESH_LOW : **vl53l0x_device.h**
- VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA_EXTSUP_HV :
vl53l0x_device.h
- VL53L0X_release_gpio() : **vl53l0x_i2c_platform.h**
- VL53L0X_ResetDevice() : **vl53l0x_api.h**
- VL53L0X_reverse_bytes() : **vl53l0x_api_core.h**
- VL53L0X_SEQUENCESTEP_DSS : **vl53l0x_def.h**
- VL53L0X_SEQUENCESTEP_FINAL_RANGE : **vl53l0x_def.h**
- VL53L0X_SEQUENCESTEP_MSRC : **vl53l0x_def.h**
- VL53L0X_SEQUENCESTEP_NUMBER_OF_CHECKS :
vl53l0x_def.h
- VL53L0X_SEQUENCESTEP_PRE_RANGE : **vl53l0x_def.h**
- VL53L0X_SEQUENCESTEP_TCC : **vl53l0x_def.h**
- VL53L0X_SequenceStepId : **vl53l0x_def.h**
- VL53L0X_set_gpio() : **vl53l0x_i2c_platform.h**
- VL53L0X_set_measurement_timing_budget_micro_seconds() :
vl53l0x_api_core.h
- VL53L0X_set_offset_calibration_data_micro_meter() :
vl53l0x_api_calibration.h
- VL53L0X_set_ref_calibration() : **vl53l0x_api_calibration.h**
- VL53L0X_set_reference_spads() : **vl53l0x_api_calibration.h**
- VL53L0X_set_vcsel_pulse_period() : **vl53l0x_api_core.h**
- VL53L0X_SETARRAYPARAMETERFIELD : **vl53l0x_def.h**
- VL53L0X_SetDeviceAddress() : **vl53l0x_api.h**
- VL53L0X_SetDeviceMode() : **vl53l0x_api.h**
- VL53L0X_SetDeviceParameters() : **vl53l0x_api.h**
- VL53L0X_SETDEVICESPECIFICPARAMETER : **vl53l0x_def.h**
- VL53L0X_SetDmaxCalParameters() : **vl53l0x_api.h**
- VL53L0X_SetGpioConfig() : **vl53l0x_api.h**
- VL53L0X_SetGroupParamHold() : **vl53l0x_api.h**
- VL53L0X_SetHistogramMode() : **vl53l0x_api.h**
- VL53L0X_SetInterMeasurementPeriodMilliSeconds() :
vl53l0x_api.h
- VL53L0X_SetInterruptThresholds() : **vl53l0x_api.h**
- VL53L0X_SetLimitCheckEnable() : **vl53l0x_api.h**
- VL53L0X_SetLimitCheckValue() : **vl53l0x_api.h**
- VL53L0X_SetLinearityCorrectiveGain() : **vl53l0x_api.h**
- VL53L0X_SetMeasurementTimingBudgetMicroSeconds() :

vl53l0x_api.h

- VL53L0X_SetNumberOfROIZones() : **vl53l0x_api.h**
- VL53L0X_SetOffsetCalibrationDataMicroMeter() : **vl53l0x_api.h**
- VL53L0X_SETPARAMETERFIELD : **vl53l0x_def.h**
- VL53L0X_SetPowerMode() : **vl53l0x_api.h**
- VL53L0X_SetRangeFractionEnable() : **vl53l0x_api.h**
- VL53L0X_SetRefCalibration() : **vl53l0x_api.h**
- VL53L0X_SetReferenceSpads() : **vl53l0x_api.h**
- VL53L0X_SetSequenceStepEnable() : **vl53l0x_api.h**
- VL53L0X_SetSequenceStepTimeout() : **vl53l0x_api.h**
- VL53L0X_SetSpadAmbientDamperFactor() : **vl53l0x_api.h**
- VL53L0X_SetSpadAmbientDamperThreshold() : **vl53l0x_api.h**
- VL53L0X_SetTuningSettingBuffer() : **vl53l0x_api.h**
- VL53L0X_SetVcselPulsePeriod() : **vl53l0x_api.h**
- VL53L0X_SetWrapAroundCheckEnable() : **vl53l0x_api.h**
- VL53L0X_SetXTalkCompensationEnable() : **vl53l0x_api.h**
- VL53L0X_SetXTalkCompensationRateMegaCps() : **vl53l0x_api.h**
- VL53L0X_SIGMA_ESTIMATE_MAX_VALUE : **vl53l0x_device.h**
- VL53L0X_SPECIFICATION_VER_MAJOR : **vl53l0x_def.h**
- VL53L0X_SPECIFICATION_VER_MINOR : **vl53l0x_def.h**
- VL53L0X_SPECIFICATION_VER_REVISION : **vl53l0x_def.h**
- VL53L0X_SPECIFICATION_VER_SUB : **vl53l0x_def.h**
- VL53L0X_SPEED_OF_LIGHT_IN_AIR : **vl53l0x_device.h**
- VL53L0X_StartMeasurement() : **vl53l0x_api.h**
- VL53L0X_State : **vl53l0x_def.h**
- VL53L0X_STATE_ERROR : **vl53l0x_def.h**
- VL53L0X_STATE_IDLE : **vl53l0x_def.h**
- VL53L0X_STATE_POWERDOWN : **vl53l0x_def.h**
- VL53L0X_STATE_RUNNING : **vl53l0x_def.h**
- VL53L0X_STATE_STANDBY : **vl53l0x_def.h**
- VL53L0X_STATE_UNKNOWN : **vl53l0x_def.h**
- VL53L0X_STATE_WAIT_STATICINIT : **vl53l0x_def.h**
- VL53L0X_StaticInit() : **vl53l0x_api.h**
- VL53L0X_StopMeasurement() : **vl53l0x_api.h**
- VL53L0X_STRING_CHECKENABLE_RANGE_IGNORE_THRESHOLD : **vl53l0x_api_strings.h**
- VL53L0X_STRING_CHECKENABLE_SIGMA_FINAL_RANGE : **vl53l0x_api_strings.h**
- VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE : **vl53l0x_api_strings.h**

- VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_MSRC :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_PRE_RANG :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_CHECKENABLE_SIGNAL_REF_CLIP :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICE_INFO_NAME :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICE_INFO_NAME_ES1 :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICE_INFO_NAME_TS0 :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICE_INFO_NAME_TS1 :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICE_INFO_NAME_TS2 :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICE_INFO_TYPE :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_ALGOOVERFLOW :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_ALGOUNDERFLOW :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_MINCLIP :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_MSRCNOTARGET :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_NONE :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_NOVHVVALUEFOUND :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_PHASECONSISTENCY :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_RANGECOMPLETE :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_RANGEIGNORETHRESHOL :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_RANGEPHASECHECK :
[vl53l0x_api_strings.h](#)
- VL53L0X_STRING_DEVICEERROR_SIGMATHRESHOLDCHECK :
[vl53l0x_api_strings.h](#)

- VL53L0X_STRING_DEVICEERROR_SNRCHECK :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_DEVICEERROR_TCC :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_DEVICEERROR_UNKNOWN :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_DEVICEERROR_VCSELCONTINUITYTESTFA :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_DEVICEERROR_VCSELWATCHDOGTESTFAI :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_BUFFER_TOO_SMALL :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_CALIBRATION_WARNING :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_CONTROL_INTERFACE :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_DIVISION_BY_ZERO :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_GPIO_FUNCTIONALITY_NOT_SUPP :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_GPIO_NOT_EXISTING :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_INTERRUPT_NOT_CLEARED :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_INVALID_COMMAND :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_INVALID_PARAMS :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_MIN_CLIPPED :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_MODE_NOT_SUPPORTED :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_NONE :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_NOT_IMPLEMENTED :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_NOT_SUPPORTED :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_RANGE_ERROR :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_ERROR_REF_SPAD_INIT :

vl53l0x_api_strings.h

- VL53L0X_STRING_ERROR_TIME_OUT : **vl53l0x_api_strings.h**
- VL53L0X_STRING_ERROR_UNDEFINED :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_HW : **vl53l0x_api_strings.h**
- VL53L0X_STRING_RANGESTATUS_MINRANGE :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_NONE :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_PHASE :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_RANGEVALID :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_SIGMA :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_SIGNAL :
vl53l0x_api_strings.h
- VL53L0X_STRING_SEQUENCESTEP_DSS :
vl53l0x_api_strings.h
- VL53L0X_STRING_SEQUENCESTEP_FINAL_RANGE :
vl53l0x_api_strings.h
- VL53L0X_STRING_SEQUENCESTEP_MSRC :
vl53l0x_api_strings.h
- VL53L0X_STRING_SEQUENCESTEP_PRE_RANGE :
vl53l0x_api_strings.h
- VL53L0X_STRING_SEQUENCESTEP_TCC :
vl53l0x_api_strings.h
- VL53L0X_STRING_STATE_ERROR : **vl53l0x_api_strings.h**
- VL53L0X_STRING_STATE_IDLE : **vl53l0x_api_strings.h**
- VL53L0X_STRING_STATE_POWERDOWN :
vl53l0x_api_strings.h
- VL53L0X_STRING_STATE_RUNNING : **vl53l0x_api_strings.h**
- VL53L0X_STRING_STATE_STANDBY : **vl53l0x_api_strings.h**
- VL53L0X_STRING_STATE_UNKNOWN : **vl53l0x_api_strings.h**
- VL53L0X_STRING_STATE_WAIT_STATICINIT :
vl53l0x_api_strings.h
- VL53L0X_STRING_UNKNOW_ERROR_CODE :
vl53l0x_api_strings.h
- VL53L0X_UnlockSequenceAccess() : **vl53l0x_platform.h**
- VL53L0X_UpdateByte() : **vl53l0x_platform.h**

- VL53L0X_VCSEL_PERIOD_FINAL_RANGE : [vl53l0x_def.h](#)
- VL53L0X_VCSEL_PERIOD_PRE_RANGE : [vl53l0x_def.h](#)
- VL53L0X_VcselPeriod : [vl53l0x_def.h](#)
- VL53L0X_wait_ms() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_WaitDeviceBooted() : [vl53l0x_api.h](#)
- VL53L0X_WaitDeviceReadyForNewMeasurement() :
[vl53l0x_api.h](#)
- VL53L0X_WrByte() : [vl53l0x_platform.h](#)
- VL53L0X_WrDWord() : [vl53l0x_platform.h](#)
- VL53L0X_write_byte() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_write_dword() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_write_multi() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_write_word() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_WriteMulti() : [vl53l0x_platform.h](#)
- VL53L0X_WrWord() : [vl53l0x_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
g	s	v	

- g -

- get_sequence_step_timeout() : [vl53l0x_api_core.h](#)

- s -

- set_sequence_step_timeout() : [vl53l0x_api_core.h](#)

- v -

- VL53L0X_apply_offset_adjustment() : [vl53l0x_api_calibration.h](#)
- VL53L0X_calc_sigma_estimate() : [vl53l0x_api_core.h](#)
- VL53L0X_calc_timeout_mciks() : [vl53l0x_api_core.h](#)
- VL53L0X_ClearInterruptMask() : [vl53l0x_api.h](#)
- VL53L0X_comms_close() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_comms_initialise() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_cycle_power() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_DataInit() : [vl53l0x_api.h](#)
- VL53L0X_decode_timeout() : [vl53l0x_api_core.h](#)
- VL53L0X_decode_vcsel_period() : [vl53l0x_api_core.h](#)
- VL53L0X_EnableInterruptMask() : [vl53l0x_api.h](#)
- VL53L0X_encode_timeout() : [vl53l0x_api_core.h](#)
- VL53L0X_encode_vcsel_period() : [vl53l0x_api_core.h](#)

- VL53L0X_get_device_error_string() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_device_info() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_gpio() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_get_info_from_device() : [vl53l0x_api_core.h](#)
- VL53L0X_get_limit_check_info() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_measurement_timing_budget_micro_seconds() : [vl53l0x_api_core.h](#)
- VL53L0X_get_offset_calibration_data_micro_meter() : [vl53l0x_api_calibration.h](#)
- VL53L0X_get_pal_error_string() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_pal_range_status() : [vl53l0x_api_core.h](#)
- VL53L0X_get_pal_state_string() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_range_status_string() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_ref_calibration() : [vl53l0x_api_calibration.h](#)
- VL53L0X_get_reference_spads() : [vl53l0x_api_calibration.h](#)
- VL53L0X_get_sequence_steps_info() : [vl53l0x_api_strings.h](#)
- VL53L0X_get_timer_frequency() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_get_timer_value() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_get_total_signal_rate() : [vl53l0x_api_core.h](#)
- VL53L0X_get_total_xtalk_rate() : [vl53l0x_api_core.h](#)
- VL53L0X_get_vcsel_pulse_period() : [vl53l0x_api_core.h](#)
- VL53L0X_GetDeviceErrorStatus() : [vl53l0x_api.h](#)
- VL53L0X_GetDeviceErrorString() : [vl53l0x_api.h](#)
- VL53L0X_GetDeviceInfo() : [vl53l0x_api.h](#)
- VL53L0X_GetDeviceMode() : [vl53l0x_api.h](#)
- VL53L0X_GetDeviceParameters() : [vl53l0x_api.h](#)
- VL53L0X_GetDmaxCalParameters() : [vl53l0x_api.h](#)
- VL53L0X_GetFractionEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetGpioConfig() : [vl53l0x_api.h](#)
- VL53L0X_GetHistogramMeasurementData() : [vl53l0x_api.h](#)
- VL53L0X_GetHistogramMode() : [vl53l0x_api.h](#)
- VL53L0X_GetInterMeasurementPeriodMilliSeconds() : [vl53l0x_api.h](#)
- VL53L0X_GetInterruptMaskStatus() : [vl53l0x_api.h](#)
- VL53L0X_GetInterruptThresholds() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckCurrent() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckInfo() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckStatus() : [vl53l0x_api.h](#)
- VL53L0X_GetLimitCheckValue() : [vl53l0x_api.h](#)

- VL53L0X_GetLinearityCorrectiveGain() : [vl53l0x_api.h](#)
- VL53L0X_GetMaxNumberOfROIZones() : [vl53l0x_api.h](#)
- VL53L0X_GetMeasurementDataReady() : [vl53l0x_api.h](#)
- VL53L0X_GetMeasurementRefSignal() : [vl53l0x_api.h](#)
- VL53L0X_GetMeasurementTimingBudgetMicroSeconds() :
[vl53l0x_api.h](#)
- VL53L0X_GetNumberOfLimitCheck() : [vl53l0x_api.h](#)
- VL53L0X_GetNumberOfROIZones() : [vl53l0x_api.h](#)
- VL53L0X_GetNumberOfSequenceSteps() : [vl53l0x_api.h](#)
- VL53L0X_GetOffsetCalibrationDataMicroMeter() : [vl53l0x_api.h](#)
- VL53L0X_GetPalErrorString() : [vl53l0x_api.h](#)
- VL53L0X_GetPalSpecVersion() : [vl53l0x_api.h](#)
- VL53L0X_GetPalState() : [vl53l0x_api.h](#)
- VL53L0X_GetPalStateString() : [vl53l0x_api.h](#)
- VL53L0X_GetPowerMode() : [vl53l0x_api.h](#)
- VL53L0X_GetProductRevision() : [vl53l0x_api.h](#)
- VL53L0X_GetRangeStatusString() : [vl53l0x_api.h](#)
- VL53L0X_GetRangingMeasurementData() : [vl53l0x_api.h](#)
- VL53L0X_GetRefCalibration() : [vl53l0x_api.h](#)
- VL53L0X_GetReferenceSpads() : [vl53l0x_api.h](#)
- VL53L0X_GetSequenceStepEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetSequenceStepEnables() : [vl53l0x_api.h](#)
- VL53L0X_GetSequenceStepsInfo() : [vl53l0x_api.h](#)
- VL53L0X_GetSequenceStepTimeout() : [vl53l0x_api.h](#)
- VL53L0X_GetSpadAmbientDamperFactor() : [vl53l0x_api.h](#)
- VL53L0X_GetSpadAmbientDamperThreshold() : [vl53l0x_api.h](#)
- VL53L0X_GetStopCompletedStatus() : [vl53l0x_api.h](#)
- VL53L0X_GetTotalSignalRate() : [vl53l0x_api.h](#)
- VL53L0X_GetTuningSettingBuffer() : [vl53l0x_api.h](#)
- VL53L0X_GetUpperLimitMilliMeter() : [vl53l0x_api.h](#)
- VL53L0X_GetVcselPulsePeriod() : [vl53l0x_api.h](#)
- VL53L0X_GetVersion() : [vl53l0x_api.h](#)
- VL53L0X_GetWrapAroundCheckEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetXTalkCompensationEnable() : [vl53l0x_api.h](#)
- VL53L0X_GetXTalkCompensationRateMegaCps() : [vl53l0x_api.h](#)
- VL53L0X_isqrt() : [vl53l0x_api_core.h](#)
- VL53L0X_load_tuning_settings() : [vl53l0x_api_core.h](#)
- VL53L0X_LockSequenceAccess() : [vl53l0x_platform.h](#)
- VL53L0X_measurement_poll_for_completion() :
[vl53l0x_api_core.h](#)

- VL53L0X_perform_offset_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_perform_phase_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_perform_ref_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_perform_ref_spad_management() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_perform_xtalk_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_PerformOffsetCalibration() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformRefCalibration() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformRefSpadManagement() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformSingleHistogramMeasurement() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformSingleMeasurement() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformSingleRangingMeasurement() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformXTalkCalibration() : [**vl53l0x_api.h**](#)
- VL53L0X_PerformXTalkMeasurement() : [**vl53l0x_api.h**](#)
- VL53L0X_platform_wait_us() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_PollingDelay() : [**vl53l0x_platform.h**](#)
- VL53L0X_quadrature_sum() : [**vl53l0x_api_core.h**](#)
- VL53L0X_RdByte() : [**vl53l0x_platform.h**](#)
- VL53L0X_RdDWord() : [**vl53l0x_platform.h**](#)
- VL53L0X_RdWord() : [**vl53l0x_platform.h**](#)
- VL53L0X_read_byte() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_read_dword() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_read_multi() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_read_word() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_ReadMulti() : [**vl53l0x_platform.h**](#)
- VL53L0X_release_gpio() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_ResetDevice() : [**vl53l0x_api.h**](#)
- VL53L0X_reverse_bytes() : [**vl53l0x_api_core.h**](#)
- VL53L0X_set_gpio() : [**vl53l0x_i2c_platform.h**](#)
- VL53L0X_set_measurement_timing_budget_micro_seconds() : [**vl53l0x_api_core.h**](#)
- VL53L0X_set_offset_calibration_data_micro_meter() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_set_ref_calibration() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_set_reference_spads() : [**vl53l0x_api_calibration.h**](#)
- VL53L0X_set_vcsel_pulse_period() : [**vl53l0x_api_core.h**](#)
- VL53L0X_SetDeviceAddress() : [**vl53l0x_api.h**](#)
- VL53L0X_SetDeviceMode() : [**vl53l0x_api.h**](#)

- VL53L0X_SetDeviceParameters() : [vl53l0x_api.h](#)
- VL53L0X_SetDmaxCalParameters() : [vl53l0x_api.h](#)
- VL53L0X_SetGpioConfig() : [vl53l0x_api.h](#)
- VL53L0X_SetGroupParamHold() : [vl53l0x_api.h](#)
- VL53L0X_SetHistogramMode() : [vl53l0x_api.h](#)
- VL53L0X_SetInterMeasurementPeriodMilliSeconds() :
[vl53l0x_api.h](#)
- VL53L0X_SetInterruptThresholds() : [vl53l0x_api.h](#)
- VL53L0X_SetLimitCheckEnable() : [vl53l0x_api.h](#)
- VL53L0X_SetLimitCheckValue() : [vl53l0x_api.h](#)
- VL53L0X_SetLinearityCorrectiveGain() : [vl53l0x_api.h](#)
- VL53L0X_SetMeasurementTimingBudgetMicroSeconds() :
[vl53l0x_api.h](#)
- VL53L0X_SetNumberOfROIZones() : [vl53l0x_api.h](#)
- VL53L0X_SetOffsetCalibrationDataMicroMeter() : [vl53l0x_api.h](#)
- VL53L0X_SetPowerMode() : [vl53l0x_api.h](#)
- VL53L0X_SetRangeFractionEnable() : [vl53l0x_api.h](#)
- VL53L0X_SetRefCalibration() : [vl53l0x_api.h](#)
- VL53L0X_SetReferenceSpads() : [vl53l0x_api.h](#)
- VL53L0X_SetSequenceStepEnable() : [vl53l0x_api.h](#)
- VL53L0X_SetSequenceStepTimeout() : [vl53l0x_api.h](#)
- VL53L0X_SetSpadAmbientDamperFactor() : [vl53l0x_api.h](#)
- VL53L0X_SetSpadAmbientDamperThreshold() : [vl53l0x_api.h](#)
- VL53L0X_SetTuningSettingBuffer() : [vl53l0x_api.h](#)
- VL53L0X_SetVcselPulsePeriod() : [vl53l0x_api.h](#)
- VL53L0X_SetWrapAroundCheckEnable() : [vl53l0x_api.h](#)
- VL53L0X_SetXTalkCompensationEnable() : [vl53l0x_api.h](#)
- VL53L0X_SetXTalkCompensationRateMegaCps() : [vl53l0x_api.h](#)
- VL53L0X_StartMeasurement() : [vl53l0x_api.h](#)
- VL53L0X_StaticInit() : [vl53l0x_api.h](#)
- VL53L0X_StopMeasurement() : [vl53l0x_api.h](#)
- VL53L0X_UnlockSequenceAccess() : [vl53l0x_platform.h](#)
- VL53L0X_UpdateByte() : [vl53l0x_platform.h](#)
- VL53L0X_wait_ms() : [vl53l0x_i2c_platform.h](#)
- VL53L0X_WaitDeviceBooted() : [vl53l0x_api.h](#)
- VL53L0X_WaitDeviceReadyForNewMeasurement() :
[vl53l0x_api.h](#)
- VL53L0X_WrByte() : [vl53l0x_platform.h](#)
- VL53L0X_WrDWord() : [vl53l0x_platform.h](#)
- VL53L0X_write_byte() : [vl53l0x_i2c_platform.h](#)

- VL53L0X_write_dword() : [vl53l0x_i2c_platform.h](#)
 - VL53L0X_write_multi() : [vl53l0x_i2c_platform.h](#)
 - VL53L0X_write_word() : [vl53l0x_i2c_platform.h](#)
 - VL53L0X_WriteMulti() : [vl53l0x_platform.h](#)
 - VL53L0X_WrWord() : [vl53l0x_platform.h](#)
-

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
			Enumerator
			Macros

- DefaultTuningSettings : [vl53l0x_tuning.h](#)
- InterruptThresholdSettings :
[vl53l0x_interrupt_threshold_settings.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
			Enumerator
			Macros

- bool_t : [vl53l0x_i2c_platform.h](#)
- FixPoint1616_t : [vl53l0x_types.h](#)
- int16_t : [vl53l0x_types.h](#)
- int32_t : [vl53l0x_types.h](#)
- int8_t : [vl53l0x_types.h](#)
- uint16_t : [vl53l0x_types.h](#)
- uint32_t : [vl53l0x_types.h](#)
- uint64_t : [vl53l0x_types.h](#)
- uint8_t : [vl53l0x_types.h](#)
- VL53L0X_DEV : [vl53l0x_platform.h](#)
- VL53L0X_DeviceError : [vl53l0x_device.h](#)
- VL53L0X_DeviceModes : [vl53l0x_def.h](#)
- VL53L0X_Error : [vl53l0x_def.h](#)
- VL53L0X_GpioFunctionality : [vl53l0x_device.h](#)
- VL53L0X_HistogramModes : [vl53l0x_def.h](#)
- VL53L0X_InterruptPolarity : [vl53l0x_def.h](#)
- VL53L0X_PowerModes : [vl53l0x_def.h](#)
- VL53L0X_SequenceStepId : [vl53l0x_def.h](#)
- VL53L0X_State : [vl53l0x_def.h](#)
- VL53L0X_VcselPeriod : [vl53l0x_def.h](#)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
			Enumerator
			Macros

- TRACE_FUNCTION_ALL : [vl53l0x_platform_log.h](#)
- TRACE_FUNCTION_I2C : [vl53l0x_platform_log.h](#)
- TRACE_FUNCTION_NONE : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_ALL : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_DEBUG : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_ERRORS : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_IGNORE : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_INFO : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_NONE : [vl53l0x_platform_log.h](#)
- TRACE_LEVEL_WARNING : [vl53l0x_platform_log.h](#)
- TRACE_MODULE_ALL : [vl53l0x_platform_log.h](#)
- TRACE_MODULE_API : [vl53l0x_platform_log.h](#)
- TRACE_MODULE_NONE : [vl53l0x_platform_log.h](#)
- TRACE_MODULE_PLATFORM : [vl53l0x_platform_log.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c i p s v	Enumerator Macros

- - -

- `_LOG_FUNCTION_END` : `vl53l0x_platform_log.h`
- `_LOG_FUNCTION_END_FMT` : `vl53l0x_platform_log.h`
- `_LOG_FUNCTION_START` : `vl53l0x_platform_log.h`

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
			Enumerator
-	b	c i p s v	Macros

- b -

- BYTES_PER_DWORD : [vl53l0x_i2c_platform.h](#)
- BYTES_PER_WORD : [vl53l0x_i2c_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures				
Files							
File List	Globals						
All	Functions	Variables	Typedefs				
-	b	c	i	p	s	v	Macros

- C -

- COMMS_BUFFER_SIZE : [vl53l0x_i2c_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures	
Files				
File List	Globals			
All	Functions	Variables	Typedefs	Enumerator
-	b	c	i	p s v

- i -

- I2C : [vl53l0x_i2c_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures			
Files						
File List	Globals					
All	Functions	Variables	Typedefs	Enumerator	Macros	
-	b	c	i	p	s	v

- p -

- PALDevDataGet : [vl53l0x_platform.h](#)
- PALDevDataSet : [vl53l0x_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
All	Functions	Variables	Typedefs
-	b	c	i
	p	s	v

- S -

- SPI : [vl53l0x_i2c_platform.h](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures			
Files						
File List	Globals					
All	Functions	Variables	Typedefs			
b	c	i	p	s	v	Macros

- v -

- VL53L0X10_IMPLEMENTATION_VER_MAJOR : [vl53l0x_def.h](#)
- VL53L0X10_IMPLEMENTATION_VER_MINOR : [vl53l0x_def.h](#)
- VL53L0X10_IMPLEMENTATION_VER_REVISION : [vl53l0x_def.h](#)
- VL53L0X10_IMPLEMENTATION_VER_SUB : [vl53l0x_def.h](#)
- VL53L0X10_SPECIFICATION_VER_MAJOR : [vl53l0x_def.h](#)
- VL53L0X10_SPECIFICATION_VER_MINOR : [vl53l0x_def.h](#)
- VL53L0X10_SPECIFICATION_VER_REVISION : [vl53l0x_def.h](#)
- VL53L0X10_SPECIFICATION_VER_SUB : [vl53l0x_def.h](#)
- VL53L0X_API : [vl53l0x_api.h](#)
- VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS : [vl53l0x_device.h](#)
- VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD : [vl53l0x_device.h](#)
- VL53L0X_CHECKENABLE_SIGMA_FINAL_RANGE : [vl53l0x_device.h](#)
- VL53L0X_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE : [vl53l0x_device.h](#)
- VL53L0X_CHECKENABLE_SIGNAL_RATE_MSRC : [vl53l0x_device.h](#)
- VL53L0X_CHECKENABLE_SIGNAL_RATE_PRE_RANGE :

vl53l0x_device.h

- VL53L0X_CHECKENABLE_SIGNAL_REF_CLIP :
vl53l0x_device.h
- VL53L0X_COPYSTRING : **vl53l0x_platform_log.h**
- VL53L0X_DEFAULT_MAX_LOOP : **vl53l0x_def.h**
- VL53L0X_DEVICEERROR_ALGOOVERFLOW :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_ALGOUNDERFLOW :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_MINCLIP : **vl53l0x_device.h**
- VL53L0X_DEVICEERROR_MSRCNOTARGET :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_NONE : **vl53l0x_device.h**
- VL53L0X_DEVICEERROR_NOVHVVALUEFOUND :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_PHASECONSISTENCY :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_RANGECOMPLETE :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_RANGEPHASECHECK :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_SIGMATHRESHOLDCHECK :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_SNRCHECK : **vl53l0x_device.h**
- VL53L0X_DEVICEERROR_TCC : **vl53l0x_device.h**
- VL53L0X_DEVICEERROR_VSELCONTINUITYTESTFAILURE :
vl53l0x_device.h
- VL53L0X_DEVICEERROR_VSELWATCHDOGTESTFAILURE :
vl53l0x_device.h
- VL53L0X_DEVICEMODE_CONTINUOUS_RANGING :
vl53l0x_def.h
- VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING :
vl53l0x_def.h
- VL53L0X_DEVICEMODE_GPIO_DRIVE : **vl53l0x_def.h**
- VL53L0X_DEVICEMODE_GPIO_OSC : **vl53l0x_def.h**
- VL53L0X_DEVICEMODE_SINGLE_ALS : **vl53l0x_def.h**
- VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM : **vl53l0x_def.h**
- VL53L0X_DEVICEMODE_SINGLE_RANGING : **vl53l0x_def.h**

- VL53L0X_ErrLog : [vl53l0x_platform_log.h](#)
- VL53L0X_ERROR_BUFFER_TOO_SMALL : [vl53l0x_def.h](#)
- VL53L0X_ERROR_CALIBRATION_WARNING : [vl53l0x_def.h](#)
- VL53L0X_ERROR_CONTROL_INTERFACE : [vl53l0x_def.h](#)
- VL53L0X_ERROR_DIVISION_BY_ZERO : [vl53l0x_def.h](#)
- VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_GPIO_NOT_EXISTING : [vl53l0x_def.h](#)
- VL53L0X_ERROR_INTERRUPT_NOT_CLEARED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_INVALID_COMMAND : [vl53l0x_def.h](#)
- VL53L0X_ERROR_INVALID_PARAMS : [vl53l0x_def.h](#)
- VL53L0X_ERROR_MIN_CLIPPED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_MODE_NOT_SUPPORTED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_NONE : [vl53l0x_def.h](#)
- VL53L0X_ERROR_NOT_IMPLEMENTED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_NOT_SUPPORTED : [vl53l0x_def.h](#)
- VL53L0X_ERROR_RANGE_ERROR : [vl53l0x_def.h](#)
- VL53L0X_ERROR_REF_SPAD_INIT : [vl53l0x_def.h](#)
- VL53L0X_ERROR_TIME_OUT : [vl53l0x_def.h](#)
- VL53L0X_ERROR_UNDEFINED : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT08TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT102TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT08 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT102 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT313 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT412 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT53 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT88 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT1616TOFIXPOINT97 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT313TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT412TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT53TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT88TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_FIXPOINT97TOFIXPOINT1616 : [vl53l0x_def.h](#)
- VL53L0X_GETARRAYPARAMETERFIELD : [vl53l0x_def.h](#)
- VL53L0X_GETDEVICESPECIFICPARAMETER : [vl53l0x_def.h](#)
- VL53L0X_GETPARAMETERFIELD : [vl53l0x_def.h](#)
- VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY : [vl53l0x_device.h](#)

- VL53L0X_GPIOFUNCTIONALITY_OFF : [vl53l0x_device.h](#)
- VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HIGH : [vl53l0x_device.h](#)
- VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_LOW : [vl53l0x_device.h](#)
- VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OUT : [vl53l0x_device.h](#)
- VL53L0X_HISTOGRAM_BUFFER_SIZE : [vl53l0x_def.h](#)
- VL53L0X_HISTOGRAMMODE_BOTH : [vl53l0x_def.h](#)
- VL53L0X_HISTOGRAMMODE_DISABLED : [vl53l0x_def.h](#)
- VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY : [vl53l0x_def.h](#)
- VL53L0X_HISTOGRAMMODE_RETURN_ONLY : [vl53l0x_def.h](#)
- VL53L0X_IMPLEMENTATION_VER_MAJOR : [vl53l0x_def.h](#)
- VL53L0X_IMPLEMENTATION_VER_MINOR : [vl53l0x_def.h](#)
- VL53L0X_IMPLEMENTATION_VER_REVISION : [vl53l0x_def.h](#)
- VL53L0X_IMPLEMENTATION_VER_SUB : [vl53l0x_def.h](#)
- VL53L0X_INTERRUPTPOLARITY_HIGH : [vl53l0x_def.h](#)
- VL53L0X_INTERRUPTPOLARITY_LOW : [vl53l0x_def.h](#)
- VL53L0X_MAKEUINT16 : [vl53l0x_def.h](#)
- VL53L0X_MAX_STRING_LENGTH : [vl53l0x_def.h](#)
- VL53L0X_MAX_STRING_LENGTH_PLT : [vl53l0x_i2c_platform.h](#)
- VL53L0X_POWERMODE_IDLE_LEVEL1 : [vl53l0x_def.h](#)
- VL53L0X_POWERMODE_IDLE_LEVEL2 : [vl53l0x_def.h](#)
- VL53L0X_POWERMODE_STANDBY_LEVEL1 : [vl53l0x_def.h](#)
- VL53L0X_POWERMODE_STANDBY_LEVEL2 : [vl53l0x_def.h](#)
- VL53L0X_REF_SPAD_BUFFER_SIZE : [vl53l0x_def.h](#)
- VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET_MM : [vl53l0x_device.h](#)
- VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT : [vl53l0x_device.h](#)
- VL53L0X_REG_ALGO_PHASECAL_LIM : [vl53l0x_device.h](#)
- VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RATE_MM : [vl53l0x_device.h](#)
- VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF_SPAD : [vl53l0x_device.h](#)
- VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET : [vl53l0x_device.h](#)
- VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RATE_RI

: [vl53l0x_device.h](#)

- VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR :
[vl53l0x_device.h](#)
- VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_H :
[vl53l0x_device.h](#)
- VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_L :
[vl53l0x_device.h](#)
- VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_HIGH :
[vl53l0x_device.h](#)
- VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_LOW :
[vl53l0x_device.h](#)
- VL53L0X_REG_FINAL_RANGE_CONFIG_VSEL_PERIOD :
[vl53l0x_device.h](#)
- VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELECT :
[vl53l0x_device.h](#)
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_0 :
[vl53l0x_device.h](#)
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1 :
[vl53l0x_device.h](#)
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2 :
[vl53l0x_device.h](#)
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3 :
[vl53l0x_device.h](#)
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4 :
[vl53l0x_device.h](#)
- VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5 :
[vl53l0x_device.h](#)
- VL53L0X_REG_GLOBAL_CONFIG_VSEL_WIDTH :
[vl53l0x_device.h](#)
- VL53L0X_REG_GPIO_HV_MUX_ACTIVE_HIGH :
[vl53l0x_device.h](#)
- VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SELECT :
[vl53l0x_device.h](#)
- VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL :
[vl53l0x_device.h](#)
- VL53L0X_REG_I2C_SLAVE_DEVICE_ADDRESS :
[vl53l0x_device.h](#)
- VL53L0X_REG_IDENTIFICATION_MODEL_ID :
[vl53l0x_device.h](#)
- VL53L0X_REG_IDENTIFICATION_REVISION_ID :

vl53l0x_device.h

- VL53L0X_REG_MSRC_CONFIG_CONTROL : **vl53l0x_device.h**
- VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP :
vl53l0x_device.h
- VL53L0X_REG_OSC_CALIBRATE_VAL : **vl53l0x_device.h**
- VL53L0X_REG_POWER_MANAGEMENT_G01_POWER_FORCE : **vl53l0x_device.h**
- VL53L0X_REG_PRE_RANGE_CONFIG_MIN_SNR :
vl53l0x_device.h
- VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_HI :
vl53l0x_device.h
- VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_LO :
vl53l0x_device.h
- VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_HI :
vl53l0x_device.h
- VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_LO :
vl53l0x_device.h
- VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HIGH :
vl53l0x_device.h
- VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_LOW :
vl53l0x_device.h
- VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD :
vl53l0x_device.h
- VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LIMIT :
vl53l0x_device.h
- VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_I : **vl53l0x_device.h**
- VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_I : **vl53l0x_device.h**
- VL53L0X_REG_RESULT_CORE_PAGE : **vl53l0x_device.h**
- VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_RE : **vl53l0x_device.h**
- VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_RT : **vl53l0x_device.h**
- VL53L0X_REG_RESULT_INTERRUPT_STATUS :
vl53l0x_device.h
- VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF :
vl53l0x_device.h
- VL53L0X_REG_RESULT_RANGE_STATUS : **vl53l0x_device.h**
- VL53L0X_REG_SOFT_RESET_G02_SOFT_RESET_N :

vl53l0x_device.h

- VL53L0X_REG_SYSRANGE_MODE_BACKTOBACK :
vl53l0x_device.h
- VL53L0X_REG_SYSRANGE_MODE_HISTOGRAM :
vl53l0x_device.h
- VL53L0X_REG_SYSRANGE_MODE_MASK : **vl53l0x_device.h**
- VL53L0X_REG_SYSRANGE_MODE_SINGLESHOT :
vl53l0x_device.h
- VL53L0X_REG_SYSRANGE_MODE_START_STOP :
vl53l0x_device.h
- VL53L0X_REG_SYSRANGE_MODE_TIMED : **vl53l0x_device.h**
- VL53L0X_REG_SYSTEM_HISTOGRAM_BIN : **vl53l0x_device.h**
- VL53L0X_REG_SYSTEM_INTERMEASUREMENT_PERIOD :
vl53l0x_device.h
- VL53L0X_REG_SYSTEM_INTERRUPT_CLEAR :
vl53l0x_device.h
- VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO :
vl53l0x_device.h
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED :
vl53l0x_device.h
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH :
vl53l0x_device.h
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW :
vl53l0x_device.h
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_READY : **vl53l0x_device.h**
- VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW : **vl53l0x_device.h**
- VL53L0X_REG_SYSTEM_RANGE_CONFIG : **vl53l0x_device.h**
- VL53L0X_REG_SYSTEM_SEQUENCE_CONFIG :
vl53l0x_device.h
- VL53L0X_REG_SYSTEM_THRESH_HIGH : **vl53l0x_device.h**
- VL53L0X_REG_SYSTEM_THRESH_LOW : **vl53l0x_device.h**
- VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA__EXTSUP_HV :
vl53l0x_device.h
- VL53L0X_SEQUENCESTEP_DSS : **vl53l0x_def.h**
- VL53L0X_SEQUENCESTEP_FINAL_RANGE : **vl53l0x_def.h**
- VL53L0X_SEQUENCESTEP_MSRC : **vl53l0x_def.h**
- VL53L0X_SEQUENCESTEP_NUMBER_OF_CHECKS :

vl53l0x_def.h

- VL53L0X_SEQUENCESTEP_PRE_RANGE : [**vl53l0x_def.h**](#)
- VL53L0X_SEQUENCESTEP_TCC : [**vl53l0x_def.h**](#)
- VL53L0X_SETARRAYPARAMETERFIELD : [**vl53l0x_def.h**](#)
- VL53L0X_SETDEVICESPECIFICPARAMETER : [**vl53l0x_def.h**](#)
- VL53L0X_SETPARAMETERFIELD : [**vl53l0x_def.h**](#)
- VL53L0X_SIGMA_ESTIMATE_MAX_VALUE : [**vl53l0x_device.h**](#)
- VL53L0X_SPECIFICATION_VER_MAJOR : [**vl53l0x_def.h**](#)
- VL53L0X_SPECIFICATION_VER_MINOR : [**vl53l0x_def.h**](#)
- VL53L0X_SPECIFICATION_VER_REVISION : [**vl53l0x_def.h**](#)
- VL53L0X_SPECIFICATION_VER_SUB : [**vl53l0x_def.h**](#)
- VL53L0X_SPEED_OF_LIGHT_IN_AIR : [**vl53l0x_device.h**](#)
- VL53L0X_STATE_ERROR : [**vl53l0x_def.h**](#)
- VL53L0X_STATE_IDLE : [**vl53l0x_def.h**](#)
- VL53L0X_STATE_POWERDOWN : [**vl53l0x_def.h**](#)
- VL53L0X_STATE_RUNNING : [**vl53l0x_def.h**](#)
- VL53L0X_STATE_STANDBY : [**vl53l0x_def.h**](#)
- VL53L0X_STATE_UNKNOWN : [**vl53l0x_def.h**](#)
- VL53L0X_WAIT_STATICINIT : [**vl53l0x_def.h**](#)
- VL53L0X_STRING_CHECKENABLE_RANGE_IGNORE_THRESH : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_CHECKENABLE_SIGMA_FINAL_RANGE : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_MSRC : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_PRE_RANGE : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_CHECKENABLE_SIGNAL_REF_CLIP : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_DEVICE_INFO_NAME : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_DEVICE_INFO_NAME_ES1 : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_DEVICE_INFO_NAME_TS0 : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_DEVICE_INFO_NAME_TS1 : [**vl53l0x_api_strings.h**](#)
- VL53L0X_STRING_DEVICE_INFO_NAME_TS2 :

vl53l0x_api_strings.h

- VL53L0X_STRING_DEVICE_INFO_TYPE :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_ALGOOVERFLOW :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_ALGOUNDERFLOW :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_MINCLIP :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_MSRCNOTARGET :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_NONE :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_NOVHVVALUEFOUND :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_PHASECONSISTENCY :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_RANGECOMPLETE :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_RANGEIGNORETHRESHOLD :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_RANGEPHASECHECK :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_SIGMATHRESHOLDCHECK :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_SNRCHECK :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_TCC :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_UNKNOWN :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_VCSELCONTINUITYTESTFAIL :
vl53l0x_api_strings.h
- VL53L0X_STRING_DEVICEERROR_VCSELWATCHDOGTESTFAIL :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_BUFFER_TOO_SMALL :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_CALIBRATION_WARNING :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_CONTROL_INTERFACE :
vl53l0x_api_strings.h

vl53l0x_api_strings.h

- VL53L0X_STRING_ERROR_DIVISION_BY_ZERO :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_GPIO_FUNCTIONALITY_NOT_SUPP :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_GPIO_NOT_EXISTING :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_INTERRUPT_NOT_CLEARED :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_INVALID_COMMAND :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_INVALID_PARAMS :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_MIN_CLIPPED :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_MODE_NOT_SUPPORTED :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_NONE :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_NOT_IMPLEMENTED :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_NOT_SUPPORTED :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_RANGE_ERROR :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_REF_SPAD_INIT :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_TIME_OUT :
vl53l0x_api_strings.h
- VL53L0X_STRING_ERROR_UNDEFINED :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_HW :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_MINRANGE :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_NONE :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_PHASE :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_RANGEVALID :
vl53l0x_api_strings.h
- VL53L0X_STRING_RANGESTATUS_SIGMA :
vl53l0x_api_strings.h

- VL53L0X_STRING_RANGESTATUS_SIGNAL :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_SEQUENCESTEP_DSS :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_SEQUENCESTEP_FINAL_RANGE :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_SEQUENCESTEP_MSRC :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_SEQUENCESTEP_PRE_RANGE :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_SEQUENCESTEP_TCC :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_STATE_ERROR : [`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_STATE_IDLE : [`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_STATE_POWERDOWN :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_STATE_RUNNING : [`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_STATE_STANDBY : [`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_STATE_UNKNOWN : [`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_STATE_WAIT_STATICINIT :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_STRING_UNKNOW_ERROR_CODE :
[`vl53l0x_api_strings.h`](#)
- VL53L0X_VCSEL_PERIOD_FINAL_RANGE : [`vl53l0x_def.h`](#)
- VL53L0X_VCSEL_PERIOD_PRE_RANGE : [`vl53l0x_def.h`](#)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

Related Pages

Here is a list of all related documentation pages:

Platform	All API settings that are platform-dependent must be adapted to the platform on which API is compiled/running
RangeStatus	The Range Status is contained in the VL53L0X_RangingMeasurementData_t and give the quality of the latest ranging
Strings	The API uses character strings to inform the user about the state of the API, the meaning of the error or about the name of a particular mode
Disclaimer	Copyright (C) 2015 STMicroelectronics Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts

Generated by Doxygen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	platform

vl53l0x_platform.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright ? 2015, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |     notice, this list of conditions and  
12 |       the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |       notice, this list of conditions and  
16 |       the following disclaimer in the  
17 |       documentation and/or other materials  
18 |       provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be  
21 |       used to endorse or promote products derived  
22 |       from this software without specific written  
23 |       permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | HOLDERS AND CONTRIBUTORS "AS IS" AND ANY  
27 | EXPRESS OR IMPLIED WARRANTIES, INCLUDING,  
28 | BUT NOT LIMITED TO, THE IMPLIED WARRANTIES  
29 | OF MERCHANTABILITY AND FITNESS FOR A  
30 | PARTICULAR PURPOSE ARE DISCLAIMED. IN NO  
31 | EVENT SHALL THE COPYRIGHT HOLDER OR  
32 | CONTRIBUTORS BE LIABLE FOR ANY DIRECT,  
33 | INDIRECT, INCIDENTAL, SPECIAL,  
34 | EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
35 | (INCLUDING, BUT NOT LIMITED TO,  
36 | PROCUREMENT OF SUBSTITUTE GOODS  
37 | OR SERVICES; LOSS OF USE,  
38 | DATA, OR PROFITS; OR BUSINESS  
39 | INTERRUPTION) HOWEVER CAUSED AND  
40 | ON ANY THEORY OF LIABILITY, WHETHER  
41 | IN CONTRACT, STRICT LIABILITY,  
42 | OR TORT (INCLUDING NEGLIGENCE  
43 | OR OTHERWISE) ARISING IN  
44 | ANY WAY OUT OF THE USE  
45 | OF THIS SOFTWARE, EVEN IF ADVISED  
46 | OF THE POSSIBILITY OF SUCH DAMAGE.
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

29 |

30 | #ifndef _VL53L0X_PLATFORM_H_
31 | #define _VL53L0X_PLATFORM_H_

32 |

33 | #include "vl53l0x_def.h"
34 | #include "vl53l0x_platform_log.h"

```
35 #include "vl53l0x_i2c_platform.h"
36
37 #ifdef __cplusplus
38 extern "C" {
39 #endif
40
58 typedef struct {
59     VL53L0X_DevData_t Data;
62     uint8_t I2cDevAddr;
63     uint8_t comms_type;
64     uint16_t comms_speed_khz;
66 } VL53L0X_Dev_t;
67
68
73 typedef VL53L0X_Dev_t* VL53L0X_DEV;
74
84 #define PALDevDataGet(Dev, field) (Dev->Data.field)
85
93 #define PALDevDataSet(Dev, field, data)
94     (Dev->Data.field)=(data)
95
108 VL53L0X_Error
109     VL53L0X_LockSequenceAccess(VL53L0X_DEV Dev);
110
116 VL53L0X_Error
117     VL53L0X_UnlockSequenceAccess(VL53L0X_DEV Dev);
118
128 VL53L0X_Error VL53L0X_WriteMulti(VL53L0X_DEV
129     Dev, uint8_t index, uint8_t *pdata, uint32_t
130     count);
131
139 VL53L0X_Error VL53L0X_ReadMulti(VL53L0X_DEV
140     Dev, uint8_t index, uint8_t *pdata, uint32_t
141     count);
```

```
140
149 VL53L0X_Error VL53L0X_WrByte(VL53L0X_DEV
    Dev, uint8_t index, uint8_t data);
150
159 VL53L0X_Error VL53L0X_WrWord(VL53L0X_DEV
    Dev, uint8_t index, uint16_t data);
160
169 VL53L0X_Error VL53L0X_WrDWord(VL53L0X_DEV
    Dev, uint8_t index, uint32_t data);
170
179 VL53L0X_Error VL53L0X_RdByte(VL53L0X_DEV
    Dev, uint8_t index, uint8_t *data);
180
189 VL53L0X_Error VL53L0X_RdWord(VL53L0X_DEV
    Dev, uint8_t index, uint16_t *data);
190
199 VL53L0X_Error VL53L0X_RdDWord(VL53L0X_DEV
    Dev, uint8_t index, uint32_t *data);
200
213 VL53L0X_Error VL53L0X_UpdateByte(VL53L0X_DEV
    Dev, uint8_t index, uint8_t AndData, uint8_t
    OrData);
214
230 VL53L0X_Error
    VL53L0X_PollingDelay(VL53L0X_DEV Dev); /*  

        usually best implemented as a real function */
231
234 #ifdef __cplusplus
235 }
236 #endif
237
238 #endif /* _VL53L0X_PLATFORM_H_ */
239
240
241
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			

vl53l0x_def.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright © 2016, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |       notice, this list of conditions and  
12 |       the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |       notice, this list of conditions and  
16 |       the following disclaimer in the  
17 |       documentation and/or other materials  
18 |       provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be used  
21 |       to endorse or promote products derived from  
22 |       this software without specific prior written  
23 |       permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | HOLDERS AND CONTRIBUTORS "AS IS" AND ANY  
27 | EXPRESS OR IMPLIED WARRANTIES, INCLUDING,  
28 | BUT NOT LIMITED TO, THE IMPLIED WARRANTIES  
29 | OF MERCHANTABILITY AND FITNESS FOR A  
30 | PARTICULAR PURPOSE ARE DISCLAIMED. IN NO  
31 | EVENT SHALL THE COPYRIGHT HOLDER OR  
32 | CONTRIBUTORS BE LIABLE FOR ANY DIRECT,  
33 | INDIRECT, INCIDENTAL, SPECIAL,  
34 | EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
35 | (INCLUDING, BUT NOT LIMITED TO,  
36 | PROCUREMENT OF SUBSTITUTE GOODS  
37 | OR SERVICES; LOSS OF USE,  
38 | DATA, OR PROFITS; OR BUSINESS  
39 | INTERRUPTION) HOWEVER CAUSED  
40 | AND ON ANY THEORY OF LIABILITY,  
41 | WHETHER IN CONTRACT, STRICT  
42 | LIABILITY, OR TORT (INCLUDING  
43 | NEGLIGENCE OR OTHERWISE)  
44 | ARISING IN ANY WAY OUT OF  
45 | THE USE OF THIS SOFTWARE,  
46 | EVEN IF ADVISED OF THE POSSIBILITY  
47 | OF SUCH DAMAGE.
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

37 | #ifndef _VL53L0X_DEF_H_
38 | #define _VL53L0X_DEF_H_

39 |

40 |

41 | #ifdef __cplusplus
42 | extern "C" {

```
43 #endif
44
52 #define VL53L0X10_SPECIFICATION_VER_MAJOR
1
53
54 #define VL53L0X10_SPECIFICATION_VER_MINOR
2
55
56 #define VL53L0X10_SPECIFICATION_VER_SUB    7
57
58 #define VL53L0X10_SPECIFICATION_VER_REVISION
1440
59
61 #define VL53L0X10_IMPLEMENTATION_VER_MAJOR
1
62
63 #define VL53L0X10_IMPLEMENTATION_VER_MINOR
0
64
65 #define VL53L0X10_IMPLEMENTATION_VER_SUB
9
66
67 #define
    VL53L0X10_IMPLEMENTATION_VER_REVISION 3673
68
70 #define VL53L0X_SPECIFICATION_VER_MAJOR 1
71
72 #define VL53L0X_SPECIFICATION_VER_MINOR 2
73
74 #define VL53L0X_SPECIFICATION_VER_SUB    7
75
76 #define VL53L0X_SPECIFICATION_VER_REVISION
1440
77
79 #define VL53L0X_IMPLEMENTATION_VER_MAJOR
1
80
```

```
81 #define VL53L0X_IMPLEMENTATION_VER_MINOR
82 0
83 #define VL53L0X_IMPLEMENTATION_VER_SUB 2
84
85 #define VL53L0X_IMPLEMENTATION_VER_REVISION
86 4823
87 #define VL53L0X_DEFAULT_MAX_LOOP 2000
88 #define VL53L0X_MAX_STRING_LENGTH 32
89
90 #include "vl53l0x_device.h"
91 #include "vl53l0x_types.h"
92
93
94 /*****
95 * PRIVATE define do not edit
96 *****/
97
100 typedef struct {
101     uint32_t revision;
102     uint8_t major;
103     uint8_t minor;
104     uint8_t build;
105 } VL53L0X_Version_t;
106
107
110 typedef struct {
111     char Name[VL53L0X_MAX_STRING_LENGTH];
113     char Type[VL53L0X_MAX_STRING_LENGTH];
115     char
116         ProductId[VL53L0X_MAX_STRING_LENGTH];
117     uint8_t ProductType;
119     uint8_t ProductRevisionMajor;
121     uint8_t ProductRevisionMinor;
123 } VL53L0X_DeviceInfo_t;
124
```

```
125
131 typedef int8_t VL53L0X_Error;
132
133 #define VL53L0X_ERROR_NONE
134     ((VL53L0X_Error) 0)
135
139 #define VL53L0X_ERROR_MIN_CLIPPED
140     ((VL53L0X_Error) -2)
141
142 #define VL53L0X_ERROR_UNDEFINED
143     ((VL53L0X_Error) -3)
144
145 #define VL53L0X_ERROR_INVALID_PARAMS
146     ((VL53L0X_Error) -4)
147
148 #define VL53L0X_ERROR_NOT_SUPPORTED
149     ((VL53L0X_Error) -5)
150
151 #define VL53L0X_ERROR_RANGE_ERROR
152     ((VL53L0X_Error) -6)
153
154 #define VL53L0X_ERROR_TIME_OUT
155     ((VL53L0X_Error) -7)
156
157 #define VL53L0X_ERROR_MODE_NOT_SUPPORTED
158     ((VL53L0X_Error) -8)
159
160 #define VL53L0X_ERROR_BUFFER_TOO_SMALL
161     ((VL53L0X_Error) -9)
162
163 #define VL53L0X_ERROR_GPIO_NOT_EXISTING
164     ((VL53L0X_Error) -10)
165
166 #define
167     VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED
```

```
    ((VL53L0X_Error) -11)
159
160 #define VL53L0X_ERROR_INTERRUPT_NOT_CLEARED
    ((VL53L0X_Error) -12)
161
162 #define VL53L0X_ERROR_CONTROL_INTERFACE
    ((VL53L0X_Error) -20)
163
164 #define VL53L0X_ERROR_INVALID_COMMAND
    ((VL53L0X_Error) -30)
165
166 #define VL53L0X_ERROR_DIVISION_BY_ZERO
    ((VL53L0X_Error) -40)
167
168 #define VL53L0X_ERROR_REF_SPAD_INIT
    ((VL53L0X_Error) -50)
169
170
171 #define VL53L0X_ERROR_NOT_IMPLEMENTED
    ((VL53L0X_Error) -99)
172
181 typedef uint8_t VL53L0X_DeviceModes;
182
183 #define VL53L0X_DEVICEMODE_SINGLE_RANGING
    ((VL53L0X_DeviceModes) 0)
184 #define
    VL53L0X_DEVICEMODE_CONTINUOUS_RANGING
    ((VL53L0X_DeviceModes) 1)
185 #define VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM
    ((VL53L0X_DeviceModes) 2)
186 #define
    VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING
    ((VL53L0X_DeviceModes) 3)
187 #define VL53L0X_DEVICEMODE_SINGLE_ALS
    ((VL53L0X_DeviceModes) 10)
188 #define VL53L0X_DEVICEMODE_GPIO_DRIVE
    ((VL53L0X_DeviceModes) 20)
189 #define VL53L0X_DEVICEMODE_GPIO_OSC
```

```
((VL53L0X_DeviceModes) 21)
190     /* ... Modes to be added depending on
device */
199 typedef uint8_t VL53L0X_HistogramModes;
200
201 #define VL53L0X_HISTOGRAMMODE_DISABLED
((VL53L0X_HistogramModes) 0)
202
203 #define VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY
((VL53L0X_HistogramModes) 1)
204
205 #define VL53L0X_HISTOGRAMMODE_RETURN_ONLY
((VL53L0X_HistogramModes) 2)
206
207 #define VL53L0X_HISTOGRAMMODE_BOTH
((VL53L0X_HistogramModes) 3)
208
209     /* ... Modes to be added depending on
device */
218 typedef uint8_t VL53L0X_PowerModes;
219
220 #define VL53L0X_POWERMODE_STANDBY_LEVEL1
((VL53L0X_PowerModes) 0)
221
222 #define VL53L0X_POWERMODE_STANDBY_LEVEL2
((VL53L0X_PowerModes) 1)
223
224 #define VL53L0X_POWERMODE_IDLE_LEVEL1
((VL53L0X_PowerModes) 2)
225
226 #define VL53L0X_POWERMODE_IDLE_LEVEL2
((VL53L0X_PowerModes) 3)
227
234 typedef struct {
235     VL53L0X_DeviceModes DeviceMode;
237     VL53L0X_HistogramModes HistogramMode;
240     uint32_t
```

```
    MeasurementTimingBudgetMicroSeconds;
242    uint32_t
    InterMeasurementPeriodMilliSeconds;
245    uint8_t XTalkCompensationEnable;
247    uint16_t
    XTalkCompensationRangeMilliMeter;
249    FixPoint1616_t
    XTalkCompensationRateMegaCps;
252    int32_t RangeOffsetMicroMeters;
255    uint8_t
    LimitChecksEnable[VL53L0X_CHECKENABLE_NUMBER_OF
    F_CHECKS];
257    uint8_t
    LimitChecksStatus[VL53L0X_CHECKENABLE_NUMBER_OF
    F_CHECKS];
260    FixPoint1616_t
    LimitChecksValue[VL53L0X_CHECKENABLE_NUMBER_OF
    _CHECKS];
263    uint8_t WrapAroundCheckEnable;
265 } VL53L0X_DeviceParameters_t;
266
267
273 typedef uint8_t VL53L0X_State;
274
275 #define VL53L0X_STATE_POWERDOWN
    ((VL53L0X_State) 0)
276
277 #define VL53L0X_STATE_WAIT_STATICINIT
    ((VL53L0X_State) 1)
278
279 #define VL53L0X_STATE_STANDBY
    ((VL53L0X_State) 2)
280
281 #define VL53L0X_STATE_IDLE
    ((VL53L0X_State) 3)
282
283 #define VL53L0X_STATE_RUNNING
```

```
    ((VL53L0X_State) 4)
284
285 #define VL53L0X_STATE_UNKNOWN
    ((VL53L0X_State) 98)
286
287 #define VL53L0X_STATE_ERROR
    ((VL53L0X_State) 99)
288
295 typedef struct {
296     int32_t AmbTuningWindowFactor_K;
298     int32_t RetSignalAt0mm;
300 } VL53L0X_DMaxData_t;
301
306 typedef struct {
307     uint32_t TimeStamp;
308     uint32_t MeasurementTimeUsec;
313     uint16_t RangeMilliMeter;
315     uint16_t RangeDMaxMilliMeter;
320     FixPoint1616_t SignalRateRtnMegaCps;
324     FixPoint1616_t AmbientRateRtnMegaCps;
329     uint16_t EffectiveSpadRtnCount;
333     uint8_t ZoneId;
336     uint8_t RangeFractionalPart;
339     uint8_t RangeStatus;
343 } VL53L0X_RangingMeasurementData_t;
344
345
346 #define VL53L0X_HISTOGRAM_BUFFER_SIZE 24
347
352 typedef struct {
353     /* Histogram Measurement data */
354     uint32_t
        HistogramData[VL53L0X_HISTOGRAM_BUFFER_SIZE];
356     uint8_t HistogramType;
358     uint8_t FirstBin;
359     uint8_t BufferSize;
360     uint8_t NumberOfBins;
```

```

363     VL53L0X_DeviceError ErrorStatus;
366 } VL53L0X_HistogramMeasurementData_t;
367
368 #define VL53L0X_REF_SPAD_BUFFER_SIZE 6
369
374 typedef struct {
375     uint8_t
376     RefSpadEnables[VL53L0X_REF_SPAD_BUFFER_SIZE];
377     uint8_t
378     RefGoodSpadMap[VL53L0X_REF_SPAD_BUFFER_SIZE];
379 } VL53L0X_SpadData_t;
380
381 typedef struct {
382     FixPoint1616_t OscFrequencyMHz; /* Frequency used */
383
384     uint16_t LastEncodedTimeout; /* last encoded Time out used for timing
385     budget*/
386
387     VL53L0X_GpioFunctionality
388     Pin0GpioFunctionality; /* store the functionality of the GPIO:
389     pin0 */
390     uint32_t FinalRangeTimeoutMicroSecs;
392     uint8_t FinalRangeVcselPulsePeriod;
394     uint32_t PreRangeTimeoutMicroSecs;
396     uint8_t PreRangeVcselPulsePeriod;
399     uint16_t SigmaEstRefArray;
401     uint16_t SigmaEstEffPulseWidth;
404     uint16_t SigmaEstEffAmbWidth;
409     uint8_t ReadDataFromDeviceDone; /* Indicate if read from device has
410     been done (==1) or not (==0) */
411     uint8_t ModuleId; /* Module ID */
412     uint8_t Revision; /* test Revision */

```

```

413     char
414         ProductId[VL53L0X_MAX_STRING_LENGTH];
415         /* Product Identifier String */
416         uint8_t ReferenceSpadCount; /* used for
417             ref spad management */
418         uint8_t ReferenceSpadType; /* used for
419             ref spad management */
420         uint8_t RefSpadsInitialised; /* reports
421             if ref spads are initialised. */
422         uint32_t PartUIDUpper;
423         uint32_t PartUIDLower;
424         FixPoint1616_t SignalRateMeasFixed400mm;
425     } VL53L0X_DeviceSpecificParameters_t;
426
427     typedef struct {
428         VL53L0X_DMaxData_t DMaxData;
429         int32_t Part2PartOffsetNVMMicroMeter;
430         int32_t
431             Part2PartOffsetAdjustmentNVMMicroMeter;
432         VL53L0X_DeviceParameters_t
433             CurrentParameters;
434         VL53L0X_RangingMeasurementData_t
435             LastRangeMeasure;
436         VL53L0X_HistogramMeasurementData_t
437             LastHistogramMeasure;
438         VL53L0X_DeviceSpecificParameters_t
439             DeviceSpecificParameters;
440         VL53L0X_SpadData_t SpadData;
441         uint8_t SequenceConfig;
442         uint8_t RangeFractionalEnable;
443         VL53L0X_State PalState;
444         VL53L0X_PowerModes PowerMode;
445         uint16_t SigmaEstRefArray;
446         uint16_t SigmaEstEffPulseWidth;
447         uint16_t SigmaEstEffAmbWidth;
448         uint8_t StopVariable;
449         uint16_t targetRefRate;

```

```
470     FixPoint1616_t SigmaEstimate;
473     FixPoint1616_t SignalEstimate;
475     FixPoint1616_t LastSignalRefMcps;
477     uint8_t *pTuningSettingsPointer;
479     uint8_t UseInternalTuningSettings;
481     uint16_t LinearityCorrectiveGain;
483     uint16_t DmaxCalRangeMilliMeter;
485     FixPoint1616_t
486         DmaxCalSignalRateRtnMegaCps;
488 } VL53L0X_DevData_t;
489
490
496 typedef uint8_t VL53L0X_InterruptPolarity;
497
498 #define VL53L0X_INTERRUPTPOLARITY_LOW
499     ((VL53L0X_InterruptPolarity) 0)
500 #define VL53L0X_INTERRUPTPOLARITY_HIGH
501     ((VL53L0X_InterruptPolarity) 1)
510 typedef uint8_t VL53L0X_VcselPeriod;
511
512 #define VL53L0X_VCSEL_PERIOD_PRE_RANGE
513     ((VL53L0X_VcselPeriod) 0)
514 #define VL53L0X_VCSEL_PERIOD_FINAL_RANGE
515     ((VL53L0X_VcselPeriod) 1)
525 typedef struct {
526     uint8_t TccOn;
527     uint8_t MsrcOn;
528     uint8_t DssOn;
529     uint8_t PreRangeOn;
530     uint8_t FinalRangeOn;
531 } VL53L0X_SchedulerSequenceSteps_t;
532
540 typedef uint8_t VL53L0X_SequenceStepId;
```

```
541
542 #define VL53L0X_SEQUENCESTEP_TCC
      ((VL53L0X_VcselPeriod) 0)
543
544 #define VL53L0X_SEQUENCESTEP_DSS
      ((VL53L0X_VcselPeriod) 1)
545
546 #define VL53L0X_SEQUENCESTEP_MSRC
      ((VL53L0X_VcselPeriod) 2)
547
548 #define VL53L0X_SEQUENCESTEP_PRE_RANGE
      ((VL53L0X_VcselPeriod) 3)
549
550 #define VL53L0X_SEQUENCESTEP_FINAL_RANGE
      ((VL53L0X_VcselPeriod) 4)
551
553 #define
      VL53L0X_SEQUENCESTEP_NUMBER_OF_CHECKS
      5
554
559 /* MACRO Definitions */
565 /* Defines */
566 #define VL53L0X_SETPARAMETERFIELD(Dev,
      field, value) \
567     PALDevDataSet(Dev,
      CurrentParameters.field, value)
568
569 #define VL53L0X_GETPARAMETERFIELD(Dev,
      field, variable) \
570     variable = PALDevDataGet(Dev,
      CurrentParameters).field
571
572
573 #define VL53L0X_SETARRAYPARAMETERFIELD(Dev,
      field, index, value) \
574     PALDevDataSet(Dev,
      CurrentParameters.field[index], value)
```

```
575  
576 #define VL53L0X_GETARRAYPARAMETERFIELD(Dev,  
      field, index, variable) \  
577     variable = PALDevDataGet(Dev,  
      CurrentParameters).field[index]  
578  
579  
580 #define  
      VL53L0X_SETDEVICESPECIFICPARAMETER(Dev, field,  
      value) \  
581         PALDevDataSet(Dev,  
      DeviceSpecificParameters.field, value)  
582  
583 #define  
      VL53L0X_GETDEVICESPECIFICPARAMETER(Dev, field)  
      \  
584         PALDevDataGet(Dev,  
      DeviceSpecificParameters).field  
585  
586  
587 #define  
      VL53L0X_FIXPOINT1616TOFIXPOINT97(Value) \  
588         (uint16_t)((Value>>9)&0xFFFF)  
589 #define  
      VL53L0X_FIXPOINT97TOFIXPOINT1616(Value) \  
590         (FixPoint1616_t)(Value<<9)  
591  
592 #define  
      VL53L0X_FIXPOINT1616TOFIXPOINT88(Value) \  
593         (uint16_t)((Value>>8)&0xFFFF)  
594 #define  
      VL53L0X_FIXPOINT88TOFIXPOINT1616(Value) \  
595         (FixPoint1616_t)(Value<<8)  
596  
597 #define  
      VL53L0X_FIXPOINT1616TOFIXPOINT412(Value) \  
598         (uint16_t)((Value>>4)&0xFFFF)
```

```
599 #define  
    VL53L0X_FIXPOINT412TOFIXPOINT1616(Value) \  
600     (FixPoint1616_t)(Value<<4)  
601  
602 #define  
    VL53L0X_FIXPOINT1616TOFIXPOINT313(Value) \  
603     (uint16_t)((Value>>3)&0xFFFF)  
604 #define  
    VL53L0X_FIXPOINT313TOFIXPOINT1616(Value) \  
605     (FixPoint1616_t)(Value<<3)  
606  
607 #define  
    VL53L0X_FIXPOINT1616TOFIXPOINT08(Value) \  
608     (uint8_t)((Value>>8)&0x00FF)  
609 #define  
    VL53L0X_FIXPOINT08TOFIXPOINT1616(Value) \  
610     (FixPoint1616_t)(Value<<8)  
611  
612 #define  
    VL53L0X_FIXPOINT1616TOFIXPOINT53(Value) \  
613     (uint8_t)((Value>>13)&0x00FF)  
614 #define  
    VL53L0X_FIXPOINT53TOFIXPOINT1616(Value) \  
615     (FixPoint1616_t)(Value<<13)  
616  
617 #define  
    VL53L0X_FIXPOINT1616TOFIXPOINT102(Value) \  
618     (uint16_t)((Value>>14)&0xFFFF)  
619 #define  
    VL53L0X_FIXPOINT102TOFIXPOINT1616(Value) \  
620     (FixPoint1616_t)(Value<<12)  
621  
622 #define VL53L0X_MAKEUINT16(msb, lsb)  
    (uint16_t)((((uint16_t)msb)<<8) + \  
623         (uint16_t)lsb)  
624  
635 #ifdef __cplusplus
```

```
636 }
637 #endif
638
639
640 #endif /* _VL53L0X_DEF_H_ */
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	platform

vl53l0x_types.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright ? 2015, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |     notice, this list of conditions and  
12 |       the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |       notice, this list of conditions and  
16 |       the following disclaimer in the  
17 |       documentation and/or other materials  
18 |       provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be  
21 |       used to endorse or promote products derived  
22 |       from this software without specific written  
23 |       permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | HOLDERS AND CONTRIBUTORS "AS IS" AND ANY  
27 | EXPRESS OR IMPLIED WARRANTIES, INCLUDING,  
28 | BUT NOT LIMITED TO, THE IMPLIED  
29 | WARRANTIES OF MERCHANTABILITY AND FITNESS  
30 | FOR A PARTICULAR PURPOSE ARE DISCLAIMED.  
31 | IN NO EVENT SHALL THE COPYRIGHT  
32 | OWNER OR CONTRIBUTORS BE LIABLE FOR ANY  
33 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,  
34 | EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
35 | (INCLUDING, BUT NOT LIMITED TO,  
36 | PROCUREMENT OF SUBSTITUTE GOODS  
37 | OR SERVICES; LOSS OF USE, DATA,  
38 | OR PROFITS; OR BUSINESS INTERRUPTION)  
39 | HOWEVER CAUSED AND ON ANY THEORY OF  
40 | LIABILITY, WHETHER IN CONTRACT,  
41 | STRICT LIABILITY, OR TORT  
42 | (INCLUDING NEGLIGENCE OR OTHERWISE)  
43 | ARISING IN ANY WAY OUT OF THE USE  
44 | OF THIS SOFTWARE, EVEN IF ADVISED  
45 | OF THE POSSIBILITY OF SUCH DAMAGE.
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

33 | #ifndef VL53L0X_TYPES_H_
34 | #define VL53L0X_TYPES_H_
35 |
46 | #include <stdint.h>
47 | #include <stddef.h>
48 |
49 | #ifndef NULL

```
50 #error "Error NULL definition should be  
      done. Please add required include "  
51 #endif  
52  
53  
54 #if ! defined(STDINT_H) &&  
    !defined(_GCC_STDINT_H)  
    &&!defined(__STDINT_DECLS) &&  
    !defined(_GCC_WRAP_STDINT_H)  
55  
56 #pragma message("Please review type  
      definition of STDINT define for your platform  
      and add to list above ")  
57  
58 /*  
59  * target platform do not provide stdint  
  or use a different #define than above  
60  * to avoid seeing the message below  
  adapt the #define list above or implement  
61  * all type and delete these pragma  
62 */  
63  
69 typedef unsigned long long uint64_t;  
70  
71  
75 typedef unsigned int uint32_t;  
76  
80 typedef int int32_t;  
81  
85 typedef unsigned short uint16_t;  
86  
90 typedef short int16_t;  
91  
95 typedef unsigned char uint8_t;  
96  
100 typedef signed char int8_t;  
101
```

```
103 #endif /* _STDINT_H */
104
105
109 typedef uint32_t FixPoint1616_t;
110
111#endif /* VL53L0X_TYPES_H_ */
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			

vl53l0x_device.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright © 2016, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |       notice, this list of conditions and  
12 |       the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |       notice, this list of conditions and  
16 |       the following disclaimer in the  
17 |       documentation and/or other materials  
18 |       provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be used  
21 |       to endorse or promote products derived from  
22 |       this software without specific written  
23 |       permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | OWNER "AS IS" AND ANY EXPRESS OR IMPLIED  
27 | WARRANTIES, INCLUDING, BUT NOT LIMITED TO,  
28 | THE IMPLIED WARRANTIES OF MERCHANTABILITY  
29 | AND FITNESS FOR A PARTICULAR PURPOSE ARE  
30 | DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT  
31 | OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT,  
32 | INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY,  
33 | OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT  
34 | LIMITED TO, PROCUREMENT OF SUBSTITUTE  
35 | GOODS; LOSS OF USE, DATA, OR PROFITS;  
36 | OR BUSINESS INTERRUPTION) HOWEVER CAUSED  
37 | AND ON ANY THEORY OF LIABILITY, WHETHER  
38 | IN CONTRACT, STRICT LIABILITY, OR  
39 | TORT (INCLUDING NEGLIGENCE OR OTHERWISE)  
40 | ARISING IN ANY WAY OUT OF THE USE OF  
41 | THIS SOFTWARE, EVEN IF ADVISED OF THE  
42 | POSSIBILITY OF SUCH DAMAGE.
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

34 | #ifndef _VL53L0X_DEVICE_H_
35 | #define _VL53L0X_DEVICE_H_
36 |
37 | #include "vl53l0x_types.h"
38 |
39 |

```
54 | typedef uint8_t VL53L0X_DeviceError;
55 |
56 | #define VL53L0X_DEVICEERROR_NONE
57 |     ((VL53L0X_DeviceError) 0)
58 | #define
59 |     VL53L0X_DEVICEERROR_VSELCONTINUITYTESTFAILURE
60 |         ((VL53L0X_DeviceError) 1)
61 | #define
62 |     VL53L0X_DEVICEERROR_VSELWATCHDOGTESTFAILURE
63 |         ((VL53L0X_DeviceError) 2)
64 | #define VL53L0X_DEVICEERROR_NOVHVVALUEFOUND
65 |     ((VL53L0X_DeviceError) 3)
66 | #define VL53L0X_DEVICEERROR_MSRCNOTARGET
67 |     ((VL53L0X_DeviceError) 4)
68 | #define VL53L0X_DEVICEERROR_SNRCHECK
69 |     ((VL53L0X_DeviceError) 5)
70 | #define VL53L0X_DEVICEERROR_RANGEPHASECHECK
71 |     ((VL53L0X_DeviceError) 6)
72 | #define
73 |     VL53L0X_DEVICEERROR_SIGMATHRESHOLDCHECK
74 |         ((VL53L0X_DeviceError) 7)
75 | #define VL53L0X_DEVICEERROR_TCC
76 |     ((VL53L0X_DeviceError) 8)
77 | #define VL53L0X_DEVICEERROR_PHASECONSISTENCY
78 |     ((VL53L0X_DeviceError) 9)
79 | #define VL53L0X_DEVICEERROR_MINCLIP
80 |     ((VL53L0X_DeviceError) 10)
81 | #define VL53L0X_DEVICEERROR_RANGECOMPLETE
82 |     ((VL53L0X_DeviceError) 11)
83 | #define VL53L0X_DEVICEERROR_ALGOUNDERFLOW
84 |     ((VL53L0X_DeviceError) 12)
85 | #define VL53L0X_DEVICEERROR_ALGOOVERFLOW
86 |     ((VL53L0X_DeviceError) 13)
87 | #define
88 |     VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD
89 |         ((VL53L0X_DeviceError) 14)
```

```
72
84 #define
VL53L0X_CHECKENABLE_SIGMA_FINAL_RANGE
0
85 #define
VL53L0X_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE
1
86 #define VL53L0X_CHECKENABLE_SIGNAL_REF_CLIP
2
87 #define
VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD
3
88 #define VL53L0X_CHECKENABLE_SIGNAL_RATE_MSRC
4
89 #define
VL53L0X_CHECKENABLE_SIGNAL_RATE_PRE_RANGE
5
90
91 #define VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS
6
92
100 typedef uint8_t VL53L0X_GpioFunctionality;
101
102 #define VL53L0X_GPIOFUNCTIONALITY_OFF
\
103     ((VL53L0X_GpioFunctionality) 0)
104 #define
VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_LO
W \
105     ((VL53L0X_GpioFunctionality) 1)
106 #define
VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HI
GH \
107     ((VL53L0X_GpioFunctionality) 2)
108 #define
VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OUT
T \

```

```
109     ((VL53L0X_GpioFunctionality) 3)
110
111 #define
112     VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY
113 \
114     ((VL53L0X_GpioFunctionality) 4)
115 /* Device register map */
116
117 #define VL53L0X_REG_SYSRANGE_START
118     0x000
119
120 #define VL53L0X_REG_SYSRANGE_MODE_MASK
121     0x0F
122
123 #define
124     VL53L0X_REG_SYSRANGE_MODE_START_STOP      0x01
125
126 #define
127     VL53L0X_REG_SYSRANGE_MODE_SINGLESHOT      0x00
128
129 #define
130     VL53L0X_REG_SYSRANGE_MODE_BACKTOBACK      0x02
131
132 #define
133     VL53L0X_REG_SYSRANGE_MODE_TIMED          0x04
134
135 #define
136     VL53L0X_REG_SYSRANGE_MODE_HISTOGRAM       0x08
137
138 #define
139     VL53L0X_REG_SYSTEM_THRESH_HIGH           0x000C
140
141 #define VL53L0X_REG_SYSTEM_THRESH_LOW           0x000E
142
143 #define VL53L0X_REG_SYSTEM_SEQUENCE_CONFIG
```

```
    0x0001
147 | #define VL53L0X_REG_SYSTEM_RANGE_CONFIG
    0x0009
148 | #define
      VL53L0X_REG_SYSTEM_INTERMEASUREMENT_PERIOD
    0x0004
149 |
150 |
151 | #define
      VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO
    0x000A
152 |     #define
      VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED
    0x00
153 |     #define
      VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW
    0x01
154 |     #define
      VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH
    0x02
155 |     #define
      VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW
    0x03
156 |     #define
      VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_READY
    0x04
157 |
158 | #define VL53L0X_REG_GPIO_HV_MUX_ACTIVE_HIGH
    0x0084
159 |
160 |
161 | #define VL53L0X_REG_SYSTEM_INTERRUPT_CLEAR
    0x000B
162 |
163 | /* Result registers */
164 | #define VL53L0X_REG_RESULT_INTERRUPT_STATUS
    0x0013
```

```
165 #define VL53L0X_REG_RESULT_RANGE_STATUS  
     0x0014  
166  
167 #define VL53L0X_REG_RESULT_CORE_PAGE 1  
168 #define  
     VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_  
     RTN    0x00BC  
169 #define  
     VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_R  
     TN     0x00C0  
170 #define  
     VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_  
     REF    0x00D0  
171 #define  
     VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_R  
     EF     0x00D4  
172 #define  
     VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF  
     0x00B6  
173  
174 /* Algo register */  
175  
176 #define  
     VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET_MM  
     0x0028  
177  
178 #define VL53L0X_REG_I2C_SLAVE_DEVICE_ADDRESS  
     0x008a  
179  
180 /* Check Limit registers */  
181 #define VL53L0X_REG_MSRC_CONFIG_CONTROL  
     0x0060  
182  
183 #define VL53L0X_REG_PRE_RANGE_CONFIG_MIN_SNR  
     0X0027  
184 #define  
     VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_LOW
```

```
    0x0056
185 | #define
      VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HIGH
      0x0057
186 | #define
      VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LIMIT
      0x0064
187 |
188 | #define
      VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR
      0X0067
189 | #define
      VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_LOW
      0x0047
190 | #define
      VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_HI
      H          0x0048
191 | #define
      VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RATE_
      RTN_LIMIT  0x0044
192 |
193 |
194 | #define
      VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_HI
      0X0061
195 | #define
      VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_LO
      0X0062
196 |
197 | /* PRE RANGE registers */
198 | #define
      VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD
      0x0050
199 | #define
      VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_HI
      0x0051
200 | #define
```

```
VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_L0
0x0052
201
202 #define VL53L0X_REG_SYSTEM_HISTOGRAM_BIN
0x0081
203 #define
VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SEL
ECT          0x0033
204 #define
VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL
0x0055
205
206 #define
VL53L0X_REG_FINAL_RANGE_CONFIG_VCSEL_PERIOD
0x0070
207 #define
VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_
HI          0x0071
208 #define
VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_
LO          0x0072
209 #define
VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RATE_M
CPS         0x0020
210
211 #define
VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP
0x0046
212
213
214 #define
VL53L0X_REG_SOFT_RESET_G02_SOFT_RESET_N
0x00bf
215 #define VL53L0X_REG_IDENTIFICATION_MODEL_ID
0x00c0
216 #define
VL53L0X_REG_IDENTIFICATION_REVISION_ID
```

```
    0x00c2
217 | #define VL53L0X_REG_OSC_CALIBRATE_VAL
0x00f8
219 |
220 |
221 | #define VL53L0X_SIGMA_ESTIMATE_MAX_VALUE
65535
222 | /* equivalent to a range sigma of 655.35mm
 */
223 |
224 | #define
    VL53L0X_REG_GLOBAL_CONFIG_VCSEL_WIDTH
    0x032
225 | #define
    VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_0
    0x0B0
226 | #define
    VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1
    0x0B1
227 | #define
    VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2
    0x0B2
228 | #define
    VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3
    0x0B3
229 | #define
    VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4
    0x0B4
230 | #define
    VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5
    0x0B5
231 |
232 | #define
    VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELECT
    0xB6
233 | #define
```

```
VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF_SPA  
D 0x4E /* 0x14E */  
234 #define  
VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET  
0x4F /* 0x14F */  
235 #define  
VL53L0X_REG_POWER_MANAGEMENT_G01_POWER_FORCE  
0x80  
236  
237 /*  
238 * Speed of light in um per 1E-10 Seconds  
239 */  
240  
241 #define VL53L0X_SPEED_OF_LIGHT_IN_AIR 2997  
242  
243 #define  
VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA__EXTSUP_HV  
0x0089  
244  
245 #define VL53L0X_REG_ALGO_PHASECAL_LIM  
0x0030 /* 0x130 */  
246 #define  
VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT  
0x0030  
247  
253 #endif  
254  
255 /* _VL53L0X_DEVICE_H_ */  
256  
257
```



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
doc			

PAL_disclaimer.c

Go to the documentation of this file.

1 |

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			

vl53l0x_api.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright © 2016, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 | * Redistributions of source code must  
10 | retain the above copyright  
11 | notice, this list of conditions and the  
12 | following disclaimer.  
13 | * Redistributions in binary form must  
14 | reproduce the above copyright  
15 | notice, this list of conditions and the  
16 | following disclaimer in the  
17 | documentation and/or other materials  
18 | provided with the distribution.  
19 | * Neither the name of STMicroelectronics
```

nor the

13 | names of its contributors may be used to
endorse or promote products

14 | derived from this software without specific
prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR
A PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE
POSSIBILITY OF SUCH DAMAGE.

27 |
******/

28 |

29 | #ifndef _VL53L0X_API_H_

30 | #define _VL53L0X_API_H_

31 |

32 | #include "vl53l0x_api_strings.h"

33 | #include "vl53l0x_def.h"

```
34 #include "vl53l0x_platform.h"
35
36 #ifdef __cplusplus
37 extern "C"
38 {
39 #endif
40
41 #ifdef _MSC_VER
42 #ifndef VL53L0X_API_EXPORTS
43 #define VL53L0X_API
44 _declspec(dllexport)
45 #else
46 #define VL53L0X_API
47 #endif
48 #define VL53L0X_Error
49#endif
50
70 VL53L0X_API VL53L0X_Error
    VL53L0X_GetVersion(VL53L0X_Version_t
        *pVersion);
71
82 VL53L0X_API VL53L0X_Error
    VL53L0X_GetPalSpecVersion(
        VL53L0X_Version_t *pPalSpecVersion);
84
99 VL53L0X_API VL53L0X_Error
    VL53L0X_GetProductRevision(VL53L0X_DEV Dev,
100     uint8_t *pProductRevisionMajor, uint8_t
        *pProductRevisionMinor);
101
113 VL53L0X_API VL53L0X_Error
    VL53L0X_GetDeviceInfo(VL53L0X_DEV Dev,
114     VL53L0X_DeviceInfo_t
        *pVL53L0X_DeviceInfo);
115
126 VL53L0X_API VL53L0X_Error
```

```
    VL53L0X_GetDeviceErrorStatus(VL53L0X_DEV Dev,
127        VL53L0X_DeviceError
128        *pDeviceErrorStatus);
129
140 VL53L0X_API VL53L0X_Error
141     VL53L0X_GetRangeStatusString(uint8_t
142         RangeStatus,
143         char *pRangeStatusString);
144
145 VL53L0X_API VL53L0X_Error
146     VL53L0X_GetDeviceErrorString(
147         VL53L0X_DeviceError ErrorCode, char
148         *pDeviceErrorString);
149
150 VL53L0X_API VL53L0X_Error
151     VL53L0X_GetPalErrorString(VL53L0X_Error
152         PalErrorCode,
153         char *pPalErrorString);
154
155 VL53L0X_API VL53L0X_Error
156     VL53L0X_GetPalStateString(VL53L0X_State
157         PalStateCode,
158         char *pPalStateString);
159
160 VL53L0X_API VL53L0X_Error
161     VL53L0X_GetPalState(VL53L0X_DEV Dev,
162         VL53L0X_State *pPalState);
163
164 VL53L0X_API VL53L0X_Error
165     VL53L0X_SetPowerMode(VL53L0X_DEV Dev,
166         VL53L0X_PowerModes PowerMode);
167
168 VL53L0X_API VL53L0X_Error
169     VL53L0X_GetPowerMode(VL53L0X_DEV Dev,
170         VL53L0X_PowerModes *pPowerMode);
171
172 VL53L0X_API VL53L0X_Error
```

```
    VL53L0X_SetOffsetCalibrationDataMicroMeter(  
249        VL53L0X_DEV Dev, int32_t  
      OffsetCalibrationDataMicroMeter);  
250  
266 VL53L0X_API VL53L0X_Error  
    VL53L0X_GetOffsetCalibrationDataMicroMeter(  
267        VL53L0X_DEV Dev, int32_t  
      *pOffsetCalibrationDataMicroMeter);  
268  
281 VL53L0X_API VL53L0X_Error  
    VL53L0X_SetLinearityCorrectiveGain(VL53L0X_DEV  
      Dev,  
282        int16_t LinearityCorrectiveGain);  
283  
300 VL53L0X_API VL53L0X_Error  
    VL53L0X_GetLinearityCorrectiveGain(VL53L0X_DEV  
      Dev,  
301        uint16_t *pLinearityCorrectiveGain);  
302  
315 VL53L0X_API VL53L0X_Error  
    VL53L0X_SetGroupParamHold(VL53L0X_DEV Dev,  
316        uint8_t GroupParamHold);  
317  
336 VL53L0X_API VL53L0X_Error  
    VL53L0X_GetUpperLimitMilliMeter(VL53L0X_DEV  
      Dev,  
337        uint16_t *pUpperLimitMilliMeter);  
338  
339  
352 VL53L0X_Error  
    VL53L0X_GetTotalSignalRate(VL53L0X_DEV Dev,  
353        FixPoint1616_t *pTotalSignalRate);  
354  
377 VL53L0X_API VL53L0X_Error  
    VL53L0X_SetDeviceAddress(VL53L0X_DEV Dev,  
378        uint8_t DeviceAddress);  
379
```

```
404 | VL53L0X_API VL53L0X_Error  
405 |     VL53L0X_DataInit(VL53L0X_DEV Dev);  
423 | VL53L0X_API VL53L0X_Error  
423 |     VL53L0X_SetTuningSettingBuffer(VL53L0X_DEV  
423 |         Dev,  
424 |             uint8_t *pTuningSettingBuffer, uint8_t  
424 |             UseInternalTuningSettings);  
425 |  
443 | VL53L0X_API VL53L0X_Error  
443 |     VL53L0X_GetTuningSettingBuffer(VL53L0X_DEV  
443 |         Dev,  
444 |             uint8_t **ppTuningSettingBuffer, uint8_t  
444 |             *pUseInternalTuningSettings);  
445 |  
458 | VL53L0X_API VL53L0X_Error  
458 |     VL53L0X_StaticInit(VL53L0X_DEV Dev);  
459 |  
470 | VL53L0X_API VL53L0X_Error  
470 |     VL53L0X_WaitDeviceBooted(VL53L0X_DEV Dev);  
471 |  
484 | VL53L0X_API VL53L0X_Error  
484 |     VL53L0X_ResetDevice(VL53L0X_DEV Dev);  
485 |  
506 | VL53L0X_API VL53L0X_Error  
506 |     VL53L0X_SetDeviceParameters(VL53L0X_DEV Dev,  
507 |         const VL53L0X_DeviceParameters_t  
507 |         *pDeviceParameters);  
508 |  
522 | VL53L0X_API VL53L0X_Error  
522 |     VL53L0X_GetDeviceParameters(VL53L0X_DEV Dev,  
523 |         VL53L0X_DeviceParameters_t  
523 |         *pDeviceParameters);  
524 |  
548 | VL53L0X_API VL53L0X_Error  
548 |     VL53L0X_SetDeviceMode(VL53L0X_DEV Dev,  
549 |         VL53L0X_DeviceModes DeviceMode);
```

```
550
573 VL53L0X_API VL53L0X_Error
      VL53L0X_GetDeviceMode(VL53L0X_DEV Dev,
574         VL53L0X_DeviceModes *pDeviceMode);
575
590 VL53L0X_API VL53L0X_Error
      VL53L0X_SetRangeFractionEnable(VL53L0X_DEV
      Dev,
591     uint8_t Enable);
592
610 VL53L0X_API VL53L0X_Error
      VL53L0X_GetFractionEnable(VL53L0X_DEV Dev,
611     uint8_t *pEnable);
612
634 VL53L0X_API VL53L0X_Error
      VL53L0X_SetHistogramMode(VL53L0X_DEV Dev,
635         VL53L0X_HistogramModes HistogramMode);
636
655 VL53L0X_API VL53L0X_Error
      VL53L0X_GetHistogramMode(VL53L0X_DEV Dev,
656         VL53L0X_HistogramModes *pHistogramMode);
657
678 VL53L0X_API VL53L0X_Error
      VL53L0X_SetMeasurementTimingBudgetMicroSeconds
      (
679     VL53L0X_DEV Dev, uint32_t
      MeasurementTimingBudgetMicroSeconds);
680
700 VL53L0X_API VL53L0X_Error
      VL53L0X_GetMeasurementTimingBudgetMicroSeconds
      (
701     VL53L0X_DEV Dev, uint32_t
      *pMeasurementTimingBudgetMicroSeconds);
702
719 VL53L0X_API VL53L0X_Error
      VL53L0X_GetVcselPulsePeriod(VL53L0X_DEV Dev,
720         VL53L0X_VcselPeriod VcselPeriodType,
```

```
    uint8_t *pVCSEL_PulsePeriod);  
721 |  
738 | VL53L0X_API VL53L0X_Error  
      VL53L0X_SetVcselPulsePeriod(VL53L0X_DEV Dev,  
739 |       VL53L0X_VcselPeriod VcselPeriodType,  
      uint8_t VCSEL_PulsePeriod);  
740 |  
758 | VL53L0X_API VL53L0X_Error  
      VL53L0X_SetSequenceStepEnable(VL53L0X_DEV Dev,  
759 |       VL53L0X_SequenceStepId SequenceStepId,  
      uint8_t SequenceStepEnabled);  
760 |  
778 | VL53L0X_API VL53L0X_Error  
      VL53L0X_GetSequenceStepEnable(VL53L0X_DEV Dev,  
779 |       VL53L0X_SequenceStepId SequenceStepId,  
      uint8_t *pSequenceStepEnabled);  
780 |  
794 | VL53L0X_API VL53L0X_Error  
      VL53L0X_GetSequenceStepEnables(VL53L0X_DEV  
      Dev,  
795 |       VL53L0X_SchedulerSequenceSteps_t  
      *pSchedulerSequenceSteps);  
796 |  
813 | VL53L0X_API VL53L0X_Error  
      VL53L0X_SetSequenceStepTimeout(VL53L0X_DEV  
      Dev,  
814 |       VL53L0X_SequenceStepId SequenceStepId,  
      FixPoint1616_t TimeOutMilliSecs);  
815 |  
832 | VL53L0X_API VL53L0X_Error  
      VL53L0X_GetSequenceStepTimeout(VL53L0X_DEV  
      Dev,  
833 |       VL53L0X_SequenceStepId SequenceStepId,  
834 |       FixPoint1616_t *pTimeOutMilliSecs);  
835 |  
851 | VL53L0X_API VL53L0X_Error  
      VL53L0X_GetNumberOfSequenceSteps(VL53L0X_DEV
```

```
    Dev,  
852     uint8_t *pNumberOfSequenceSteps);  
853  
869 VL53L0X_API VL53L0X_Error  
     VL53L0X_GetSequenceStepsInfo(  
870     VL53L0X_SequenceStepId SequenceStepId,  
     char *pSequenceStepsString);  
871  
885 VL53L0X_API VL53L0X_Error  
     VL53L0X_SetInterMeasurementPeriodMilliSeconds(  
886     VL53L0X_DEV Dev, uint32_t  
     InterMeasurementPeriodMilliSeconds);  
887  
902 VL53L0X_API VL53L0X_Error  
     VL53L0X_GetInterMeasurementPeriodMilliSeconds(  
903     VL53L0X_DEV Dev, uint32_t  
     *pInterMeasurementPeriodMilliSeconds);  
904  
917 VL53L0X_API VL53L0X_Error  
     VL53L0X_SetXTalkCompensationEnable(VL53L0X_DEV  
     Dev,  
918     uint8_t XTalkCompensationEnable);  
919  
932 VL53L0X_API VL53L0X_Error  
     VL53L0X_GetXTalkCompensationEnable(VL53L0X_DEV  
     Dev,  
933     uint8_t *pXTalkCompensationEnable);  
934  
949 VL53L0X_API VL53L0X_Error  
     VL53L0X_SetXTalkCompensationRateMegaCps(VL53L0  
         X_DEV Dev,  
950     FixPoint1616_t  
     XTalkCompensationRateMegaCps);  
951  
966 VL53L0X_API VL53L0X_Error  
     VL53L0X_GetXTalkCompensationRateMegaCps(VL53L0  
         X_DEV Dev,
```

```
967     FixPoint1616_t
968     *pXTalkCompensationRateMegaCps);
969
983 VL53L0X_API VL53L0X_Error
984     VL53L0X_SetRefCalibration(VL53L0X_DEV Dev,
985         uint8_t VhvSettings, uint8_t PhaseCal);
986
1000 VL53L0X_API VL53L0X_Error
1001     VL53L0X_GetRefCalibration(VL53L0X_DEV Dev,
1002         uint8_t *pVhvSettings, uint8_t
1003             *pPhaseCal);
1004
1015 VL53L0X_API VL53L0X_Error
1016     VL53L0X_GetNumberOfLimitCheck(
1017         uint16_t *pNumberOfLimitCheck);
1018
1037 VL53L0X_API VL53L0X_Error
1038     VL53L0X_GetLimitCheckInfo(VL53L0X_DEV Dev,
1039         uint16_t LimitCheckId, char
1040             *pLimitCheckString);
1041
1064 VL53L0X_API VL53L0X_Error
1065     VL53L0X_GetLimitCheckStatus(VL53L0X_DEV Dev,
1066         uint16_t LimitCheckId, uint8_t
1067             *pLimitCheckStatus);
1068
1088 VL53L0X_API VL53L0X_Error
1089     VL53L0X_SetLimitCheckEnable(VL53L0X_DEV Dev,
1090         uint16_t LimitCheckId, uint8_t
1091             LimitCheckEnable);
1092
1114 VL53L0X_API VL53L0X_Error
1115     VL53L0X_GetLimitCheckEnable(VL53L0X_DEV Dev,
1116         uint16_t LimitCheckId, uint8_t
1117             *pLimitCheckEnable);
1118
1136 VL53L0X_API VL53L0X_Error
```

```
    VL53L0X_SetLimitCheckValue(VL53L0X_DEV Dev,
1137        uint16_t LimitCheckId, FixPoint1616_t
        LimitCheckValue);
1138
1159 VL53L0X_API VL53L0X_Error
    VL53L0X_GetLimitCheckValue(VL53L0X_DEV Dev,
1160        uint16_t LimitCheckId, FixPoint1616_t
        *pLimitCheckValue);
1161
1183 VL53L0X_API VL53L0X_Error
    VL53L0X_GetLimitCheckCurrent(VL53L0X_DEV Dev,
1184        uint16_t LimitCheckId, FixPoint1616_t
        *pLimitCheckCurrent);
1185
1197 VL53L0X_API VL53L0X_Error
    VL53L0X_SetWrapAroundCheckEnable(VL53L0X_DEV
        Dev,
1198            uint8_t WrapAroundCheckEnable);
1199
1214 VL53L0X_API VL53L0X_Error
    VL53L0X_GetWrapAroundCheckEnable(VL53L0X_DEV
        Dev,
1215            uint8_t *pWrapAroundCheckEnable);
1216
1229 VL53L0X_API VL53L0X_Error
    VL53L0X_SetDmaxCalParameters(VL53L0X_DEV Dev,
1230        uint16_t RangeMilliMeter,
        FixPoint1616_t SignalRateRtnMegaCps);
1231
1244 VL53L0X_API VL53L0X_Error
    VL53L0X_GetDmaxCalParameters(VL53L0X_DEV Dev,
1245        uint16_t *pRangeMilliMeter,
        FixPoint1616_t *pSignalRateRtnMegaCps);
1246
1274 VL53L0X_API VL53L0X_Error
    VL53L0X_PerformSingleMeasurement(VL53L0X_DEV
        Dev);
```

```
1275
1296 VL53L0X_API VL53L0X_Error
    VL53L0X_PerformRefCalibration(VL53L0X_DEV Dev,
1297     uint8_t *pVhvSettings, uint8_t
    *pPhaseCal);
1298
1326 VL53L0X_API VL53L0X_Error
    VL53L0X_PerformXTalkMeasurement(VL53L0X_DEV
    Dev,
1327     uint32_t TimeoutMs, FixPoint1616_t
    *pXTalkPerSpad,
1328     uint8_t *pAmbientTooHigh);
1329
1356 VL53L0X_API VL53L0X_Error
    VL53L0X_PerformXTalkCalibration(VL53L0X_DEV
    Dev,
1357     FixPoint1616_t XTalkCalDistance,
1358     FixPoint1616_t
    *pXTalkCompensationRateMegaCps);
1359
1385 VL53L0X_API VL53L0X_Error
    VL53L0X_PerformOffsetCalibration(VL53L0X_DEV
    Dev,
1386     FixPoint1616_t CalDistanceMilliMeter,
    int32_t *pOffsetMicroMeter);
1387
1412 VL53L0X_API VL53L0X_Error
    VL53L0X_StartMeasurement(VL53L0X_DEV Dev);
1413
1429 VL53L0X_API VL53L0X_Error
    VL53L0X_StopMeasurement(VL53L0X_DEV Dev);
1430
1450 VL53L0X_API VL53L0X_Error
    VL53L0X_GetMeasurementDataReady(VL53L0X_DEV
    Dev,
1451     uint8_t *pMeasurementDataReady);
1452
```

```
1463 VL53L0X_API VL53L0X_Error  
    VL53L0X_WaitDeviceReadyForNewMeasurement(VL53L  
        0X_DEV Dev,  
1464     uint32_t MaxLoop);  
1465  
1481 VL53L0X_API VL53L0X_Error  
    VL53L0X_GetMeasurementRefSignal(VL53L0X_DEV  
        Dev,  
1482     FixPoint1616_t *pMeasurementRefSignal);  
1483  
1501 VL53L0X_API VL53L0X_Error  
    VL53L0X_GetRangingMeasurementData(VL53L0X_DEV  
        Dev,  
1502     VL53L0X_RangingMeasurementData_t  
        *pRangingMeasurementData);  
1503  
1520 VL53L0X_API VL53L0X_Error  
    VL53L0X_GetHistogramMeasurementData(VL53L0X_DE  
        V Dev,  
1521     VL53L0X_HistogramMeasurementData_t  
        *pHistogramMeasurementData);  
1522  
1545 VL53L0X_API VL53L0X_Error  
    VL53L0X_PerformSingleRangingMeasurement(VL53L0  
        X_DEV Dev,  
1546     VL53L0X_RangingMeasurementData_t  
        *pRangingMeasurementData);  
1547  
1564 VL53L0X_API VL53L0X_Error  
    VL53L0X_PerformSingleHistogramMeasurement(VL53  
        L0X_DEV Dev,  
1565     VL53L0X_HistogramMeasurementData_t  
        *pHistogramMeasurementData);  
1566  
1583 VL53L0X_API VL53L0X_Error  
    VL53L0X_SetNumberOfROIZones(VL53L0X_DEV Dev,  
1584     uint8_t NumberOfROIZones);
```

```
1585
1602 VL53L0X_API VL53L0X_Error
    VL53L0X_GetNumberOfROIZones(VL53L0X_DEV Dev,
1603     uint8_t *pNumberOfROIZones);
1604
1618 VL53L0X_API VL53L0X_Error
    VL53L0X_GetMaxNumberOfROIZones(VL53L0X_DEV
        Dev,
1619     uint8_t *pMaxNumberOfROIZones);
1620
1652 VL53L0X_API VL53L0X_Error
    VL53L0X_SetGpioConfig(VL53L0X_DEV Dev, uint8_t
        Pin,
1653     VL53L0X_DeviceModes DeviceMode,
        VL53L0X_GpioFunctionality Functionality,
1654     VL53L0X_InterruptPolarity Polarity);
1655
1680 VL53L0X_API VL53L0X_Error
    VL53L0X_GetGpioConfig(VL53L0X_DEV Dev, uint8_t
        Pin,
1681     VL53L0X_DeviceModes *pDeviceMode,
1682     VL53L0X_GpioFunctionality
        *pFunctionality,
1683     VL53L0X_InterruptPolarity *pPolarity);
1684
1704 VL53L0X_API VL53L0X_Error
    VL53L0X_SetInterruptThresholds(VL53L0X_DEV
        Dev,
1705     VL53L0X_DeviceModes DeviceMode,
        FixPoint1616_t ThresholdLow,
1706     FixPoint1616_t ThresholdHigh);
1707
1727 VL53L0X_API VL53L0X_Error
    VL53L0X_GetInterruptThresholds(VL53L0X_DEV
        Dev,
1728     VL53L0X_DeviceModes DeviceMode,
        FixPoint1616_t *pThresholdLow,
```

```
1729     FixPoint1616_t *pThresholdHigh);
1730
1745 VL53L0X_API VL53L0X_Error
    VL53L0X_GetStopCompletedStatus(VL53L0X_DEV
Dev,
1746     uint32_t *pStopStatus);
1747
1748
1764 VL53L0X_API VL53L0X_Error
    VL53L0X_ClearInterruptMask(VL53L0X_DEV Dev,
1765     uint32_t InterruptMask);
1766
1782 VL53L0X_API VL53L0X_Error
    VL53L0X_GetInterruptMaskStatus(VL53L0X_DEV
Dev,
1783     uint32_t *pInterruptMaskStatus);
1784
1795 VL53L0X_API VL53L0X_Error
    VL53L0X_EnableInterruptMask(VL53L0X_DEV Dev,
1796     uint32_t InterruptMask);
1797
1818 VL53L0X_API VL53L0X_Error
    VL53L0X_SetSpadAmbientDamperThreshold(VL53L0X_
DEV Dev,
1819     uint16_t SpadAmbientDamperThreshold);
1820
1835 VL53L0X_API VL53L0X_Error
    VL53L0X_GetSpadAmbientDamperThreshold(VL53L0X_
DEV Dev,
1836     uint16_t *pSpadAmbientDamperThreshold);
1837
1851 VL53L0X_API VL53L0X_Error
    VL53L0X_SetSpadAmbientDamperFactor(VL53L0X_DEV
Dev,
1852     uint16_t SpadAmbientDamperFactor);
1853
1868 VL53L0X_API VL53L0X_Error
```

```
    VL53L0X_GetSpadAmbientDamperFactor(VL53L0X_DEV
Dev,
1869        uint16_t *pSpadAmbientDamperFactor);
1870
1893 VL53L0X_API VL53L0X_Error
VL53L0X_PerformRefSpadManagement(VL53L0X_DEV
Dev,
1894        uint32_t *refSpadCount, uint8_t
*isApertureSpads);
1895
1917 VL53L0X_API VL53L0X_Error
VL53L0X_SetReferenceSpads(VL53L0X_DEV Dev,
1918        uint32_t refSpadCount, uint8_t
isApertureSpads);
1919
1939 VL53L0X_API VL53L0X_Error
VL53L0X_GetReferenceSpads(VL53L0X_DEV Dev,
1940        uint32_t *refSpadCount, uint8_t
*isApertureSpads);
1941
1946 #ifdef __cplusplus
1947 }
1948#endif
1949
1950#endif /* _VL53L0X_API_H_ */
```



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			

vl53l0x_api_calibration.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright © 2016, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |           notice, this list of conditions and  
12 |           the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |           notice, this list of conditions and  
16 |           the following disclaimer in the  
17 |           documentation and/or other materials  
18 |           provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be  
21 |       used to endorse or promote products derived  
22 |       from this software without specific written  
23 |       permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | HOLDERS AND CONTRIBUTORS "AS IS" AND ANY  
27 | EXPRESS OR IMPLIED WARRANTIES, INCLUDING,  
28 | BUT NOT LIMITED TO, THE IMPLIED WARRANTIES  
29 | OF MERCHANTABILITY AND FITNESS FOR A  
30 | PARTICULAR PURPOSE ARE DISCLAIMED. IN NO  
31 | EVENT SHALL THE COPYRIGHT HOLDER OR  
32 | CONTRIBUTORS BE LIABLE FOR ANY DIRECT,  
33 | INDIRECT, INCIDENTAL, SPECIAL,  
34 | EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
35 | (INCLUDING, BUT NOT LIMITED TO,  
36 | PROCUREMENT OF SUBSTITUTE GOODS  
37 | OR SERVICES; LOSS OF USE,  
38 | DATA, OR PROFITS; OR BUSINESS  
39 | INTERRUPTION) HOWEVER CAUSED AND  
40 | ON ANY THEORY OF LIABILITY, WHETHER  
41 | IN CONTRACT, STRICT LIABILITY,  
42 | OR TORT (INCLUDING NEGLIGENCE  
43 | OR OTHERWISE) ARISING IN  
44 | ANY WAY OUT OF THE USE  
45 | OF THIS SOFTWARE, EVEN IF ADVISED  
46 | OF THE POSSIBILITY OF SUCH DAMAGE.
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

29 | #ifndef _VL53L0X_API_CALIBRATION_H_
30 | #define _VL53L0X_API_CALIBRATION_H_

31 |

32 | #include "vl53l0x_def.h"
33 | #include "vl53l0x_platform.h"

34 |

```
35
36 #ifdef __cplusplus
37 extern "C" {
38 #endif
39
40 VL53L0X_Error
41     VL53L0X_perform_xtalk_calibration(VL53L0X_DEV
42                                         Dev,
43                                         FixPoint1616_t XTalkCalDistance,
44                                         FixPoint1616_t
45                                         *pXTalkCompensationRateMegaCps);
46
47
48 VL53L0X_Error
49     VL53L0X_perform_offset_calibration(VL53L0X_DEV
50                                         Dev,
51                                         FixPoint1616_t
52                                         CalDistanceMilliMeter,
53                                         int32_t *pOffsetMicroMeter);
54
55
56 VL53L0X_Error
57     VL53L0X_set_offset_calibration_data_micro_meter
58     (VL53L0X_DEV Dev,
59      int32_t
60      OffsetCalibrationDataMicroMeter);
61
62
63 VL53L0X_Error
64     VL53L0X_get_offset_calibration_data_micro_meter
65     (VL53L0X_DEV Dev,
66      int32_t
67      *pOffsetCalibrationDataMicroMeter);
68
69
70 VL53L0X_Error
71     VL53L0X_apply_offset_adjustment(VL53L0X_DEV
72                                         Dev);
73
74
75 VL53L0X_Error
76     VL53L0X_perform_ref_spad_management(VL53L0X_DE
```

```
    VL53L0X_Error
57     VL53L0X_set_reference_spads(VL53L0X_DEV Dev,
58         uint32_t count, uint8_t
59         *isApertureSpads);
60
61     VL53L0X_Error
62     VL53L0X_get_reference_spads(VL53L0X_DEV Dev,
63         uint32_t *pSpadCount, uint8_t
64         *pIsApertureSpads);
65     VL53L0X_Error
66     VL53L0X_perform_phase_calibration(VL53L0X_DEV
67         Dev,
68         uint8_t *pPhaseCal, const uint8_t
69         get_data_enable,
70         const uint8_t restore_config);
71
72     VL53L0X_Error
73     VL53L0X_perform_ref_calibration(VL53L0X_DEV Dev,
74         uint8_t *pVhvSettings, uint8_t
75         *pPhaseCal, uint8_t get_data_enable);
76
77     VL53L0X_Error
78     VL53L0X_set_ref_calibration(VL53L0X_DEV Dev,
79         uint8_t VhvSettings, uint8_t
80         PhaseCal);
81
82     VL53L0X_Error
83     VL53L0X_get_ref_calibration(VL53L0X_DEV Dev,
84         uint8_t *pVhvSettings, uint8_t
85         *pPhaseCal);
86
87
```

```
78  
79  
80  
81 #ifdef __cplusplus  
82 }  
83 #endif  
84  
85 #endif /* _VL53L0X_API_CALIBRATION_H_ */
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			

vl53l0x_api_core.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright ? 2016, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |           notice, this list of conditions and  
12 |           the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |           notice, this list of conditions and  
16 |           the following disclaimer in the  
17 |           documentation and/or other materials  
18 |           provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be  
21 |       used to endorse or promote products derived  
22 |       from this software without specific written  
23 |       permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | HOLDERS AND CONTRIBUTORS "AS IS" AND ANY  
27 | EXPRESS OR IMPLIED WARRANTIES, INCLUDING,  
28 | BUT NOT LIMITED TO, THE IMPLIED WARRANTIES  
29 | OF MERCHANTABILITY AND FITNESS FOR A  
30 | PARTICULAR PURPOSE ARE DISCLAIMED. IN NO  
31 | EVENT SHALL THE COPYRIGHT HOLDER OR  
32 | CONTRIBUTORS BE LIABLE FOR ANY DIRECT,  
33 | INDIRECT, INCIDENTAL, SPECIAL,  
34 | EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
35 | (INCLUDING, BUT NOT LIMITED TO,  
36 | PROCUREMENT OF SUBSTITUTE GOODS  
37 | OR SERVICES; LOSS OF USE,  
38 | DATA, OR PROFITS; OR BUSINESS  
39 | INTERRUPTION) HOWEVER CAUSED  
40 | AND ON ANY THEORY OF LIABILITY,  
41 | WHETHER IN CONTRACT, STRICT  
42 | LIABILITY, OR TORT (INCLUDING  
43 | NEGLIGENCE OR OTHERWISE)  
44 | ARISING IN ANY WAY OUT OF THE USE  
45 | OF THIS SOFTWARE, EVEN IF ADVISED  
46 | OF THE POSSIBILITY OF SUCH DAMAGE.
```

```
nor the  
13|       names of its contributors may be used  
to endorse or promote products  
14|       derived from this software without  
specific prior written permission.  
15|  
16| THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
HOLDERS AND CONTRIBUTORS "AS IS" AND  
17| ANY EXPRESS OR IMPLIED WARRANTIES,  
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED  
18| WARRANTIES OF MERCHANTABILITY, FITNESS FOR A  
PARTICULAR PURPOSE, AND  
19| NON-INFRINGEMENT OF INTELLECTUAL PROPERTY  
RIGHTS ARE DISCLAIMED.  
20| IN NO EVENT SHALL STMICROELECTRONICS  
INTERNATIONAL N.V. BE LIABLE FOR ANY  
21| DIRECT, INDIRECT, INCIDENTAL, SPECIAL,  
EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
22| (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT  
OF SUBSTITUTE GOODS OR SERVICES;  
23| LOSS OF USE, DATA, OR PROFITS; OR BUSINESS  
INTERRUPTION) HOWEVER CAUSED AND  
24| ON ANY THEORY OF LIABILITY, WHETHER IN  
CONTRACT, STRICT LIABILITY, OR TORT  
25| (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING  
IN ANY WAY OUT OF THE USE OF THIS  
26| SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY  
OF SUCH DAMAGE.  
27| *****/  
28|  
29| #ifndef _VL53L0X_API_CORE_H_  
30| #define _VL53L0X_API_CORE_H_  
31|  
32| #include "vl53l0x_def.h"  
33| #include "vl53l0x_platform.h"  
34|
```

```
35
36 #ifdef __cplusplus
37 extern "C" {
38 #endif
39
40
41 VL53L0X_Error VL53L0X_reverse_bytes(uint8_t
 *data, uint32_t size);
42
43 VL53L0X_Error
    VL53L0X_measurement_poll_for_completion(VL53L0
 X_DEV Dev);
44
45 uint8_t VL53L0X_encode_vcsel_period(uint8_t
 vcsel_period_pc1ks);
46
47 uint8_t VL53L0X_decode_vcsel_period(uint8_t
 vcsel_period_reg);
48
49 uint32_t VL53L0X_isqrt(uint32_t num);
50
51 uint32_t VL53L0X_quadrature_sum(uint32_t a,
 uint32_t b);
52
53 VL53L0X_Error
    VL53L0X_get_info_from_device(VL53L0X_DEV Dev,
 uint8_t option);
54
55 VL53L0X_Error
    VL53L0X_set_vcsel_pulse_period(VL53L0X_DEV
 Dev,
56         VL53L0X_VcSELPeriod VcSELPeriodType,
      uint8_t VCSELPulsePeriodPCLK);
57
58 VL53L0X_Error
    VL53L0X_get_vcsel_pulse_period(VL53L0X_DEV
 Dev,
```



```
    *pRangingMeasurementData,
82        FixPoint1616_t *pSigmaEstimate,
83        uint32_t *pDmax_mm);
84
85 VL53L0X_Error
86     VL53L0X_get_total_xtalk_rate(VL53L0X_DEV Dev,
87         VL53L0X_RangingMeasurementData_t
88         *pRangingMeasurementData,
89         FixPoint1616_t *ptotal_xtalk_rate_mcps);
90
91 VL53L0X_Error
92     VL53L0X_get_total_signal_rate(VL53L0X_DEV Dev,
93         VL53L0X_RangingMeasurementData_t
94         *pRangingMeasurementData,
95         FixPoint1616_t
96         *ptotal_signal_rate_mcps);
97
98 VL53L0X_Error
99     VL53L0X_get_pal_range_status(VL53L0X_DEV Dev,
100         uint8_t DeviceRangeStatus,
101         FixPoint1616_t SignalRate,
102         uint16_t EffectiveSpadRtnCount,
103         VL53L0X_RangingMeasurementData_t
104         *pRangingMeasurementData,
105         uint8_t *pPalRangeStatus);
106
107 uint32_t
108     VL53L0X_calc_timeout_mccls(VL53L0X_DEV Dev,
109         uint32_t timeout_period_us, uint8_t
110         vcsel_period_pccls);
111
112 uint16_t VL53L0X_encode_timeout(uint32_t
113     timeout_macro_clks);
114
115 #ifdef __cplusplus
116 }
117#endif
```

```
107
108 #endif /* _VL53L0X_API_CORE_H_ */
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			

vl53l0x_api_ranging.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright © 2016, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |           notice, this list of conditions and  
12 |           the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |           notice, this list of conditions and  
16 |           the following disclaimer in the  
17 |           documentation and/or other materials  
18 |           provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be  
21 |       used to endorse or promote products derived  
22 |       from this software without specific written  
23 |       permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | HOLDERS AND CONTRIBUTORS "AS IS" AND ANY  
27 | EXPRESS OR IMPLIED WARRANTIES, INCLUDING,  
28 | BUT NOT LIMITED TO, THE IMPLIED WARRANTIES  
29 | OF MERCHANTABILITY AND FITNESS FOR A  
30 | PARTICULAR PURPOSE ARE DISCLAIMED. IN NO  
31 | EVENT SHALL THE COPYRIGHT HOLDER OR  
32 | CONTRIBUTORS BE LIABLE FOR ANY DIRECT,  
33 | INDIRECT, INCIDENTAL, SPECIAL,  
34 | EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
35 | (INCLUDING, BUT NOT LIMITED TO,  
36 | PROCUREMENT OF SUBSTITUTE GOODS  
37 | OR SERVICES; LOSS OF USE,  
38 | DATA, OR PROFITS; OR BUSINESS  
39 | INTERRUPTION) HOWEVER CAUSED  
40 | AND ON ANY THEORY OF LIABILITY,  
41 | WHETHER IN CONTRACT, STRICT  
42 | LIABILITY, OR TORT (INCLUDING  
43 | NEGLIGENCE OR OTHERWISE)  
44 | ARISING IN ANY WAY OUT OF THE USE  
45 | OF THIS SOFTWARE, EVEN IF ADVISED  
46 | OF THE POSSIBILITY OF SUCH DAMAGE.
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

29 | #ifndef _VL53L0X_API_RANGING_H_
30 | #define _VL53L0X_API_RANGING_H_
31 |
32 | #include "vl53l0x_def.h"
33 | #include "vl53l0x_platform.h"
34 |

```
35
36 #ifdef __cplusplus
37 extern "C" {
38 #endif
39
40
41
42
43 #ifdef __cplusplus
44 }
45 #endif
46
47#endif /* _VL53L0X_API_RANGING_H_ */
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	/ VL53L0X_1.0.2 / Api / core / inc /		

vl53l0x_api_strings.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright ? 2016, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |           notice, this list of conditions and  
12 |           the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |           notice, this list of conditions and  
16 |           the following disclaimer in the  
17 |           documentation and/or other materials  
18 |           provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |           nor the names of its contributors  
21 |           may be used to endorse or promote products  
22 |           derived from this software without  
23 |           specific written permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | HOLDERS AND CONTRIBUTORS "AS IS" AND ANY  
27 | EXPRESS OR IMPLIED WARRANTIES, INCLUDING,  
28 | BUT NOT LIMITED TO, THE IMPLIED  
29 | WARRANTIES OF MERCHANTABILITY AND FITNESS  
30 | FOR A PARTICULAR PURPOSE ARE DISCLAIMED.  
31 | IN NO EVENT SHALL THE COPYRIGHT  
32 | OWNER OR CONTRIBUTORS BE LIABLE FOR ANY  
33 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,  
34 | EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
35 | (INCLUDING, BUT NOT LIMITED TO,  
36 | PROCUREMENT OF SUBSTITUTE GOODS  
37 | OR SERVICES; LOSS OF USE, DATA,  
38 | OR PROFITS; OR BUSINESS INTERRUPTION)  
39 | HOWEVER CAUSED AND ON ANY THEORY  
40 | OF LIABILITY, WHETHER IN CONTRACT,  
41 | STRICT LIABILITY, OR TORT  
42 | (INCLUDING NEGLIGENCE OR OTHERWISE)  
43 | ARISING IN ANY WAY OUT OF THE USE  
44 | OF THIS SOFTWARE, EVEN IF ADVISED  
45 | OF THE POSSIBILITY OF SUCH DAMAGE.  
46 |  
47 |  
48 |
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

29 | #ifndef VL53L0X_API_STRINGS_H_
30 | #define VL53L0X_API_STRINGS_H_
31 |
32 | #include "vl53l0x_def.h"
33 | #include "vl53l0x_platform.h"
34 |

```
35 #ifdef __cplusplus
36 extern "C" {
37 #endif
38
39
40 VL53L0X_Error
    VL53L0X_get_device_info(VL53L0X_DEV Dev,
41                         VL53L0X_DeviceInfo_t
42                         *pVL53L0X_DeviceInfo);
43 VL53L0X_Error
    VL53L0X_get_device_error_string(VL53L0X_Device
        Error ErrorCode,
44                         char *pDeviceErrorString);
45
46 VL53L0X_Error
    VL53L0X_get_range_status_string(uint8_t
        RangeStatus,
47                         char *pRangeStatusString);
48
49 VL53L0X_Error
    VL53L0X_get_pal_error_string(VL53L0X_Error
        PalErrorCode,
50                         char *pPalErrorString);
51
52 VL53L0X_Error
    VL53L0X_get_pal_state_string(VL53L0X_State
        PalStateCode,
53                         char *pPalStateString);
54
55 VL53L0X_Error
    VL53L0X_get_sequence_steps_info(
56                         VL53L0X_SequenceStepId
        SequenceStepId,
57                         char *pSequenceStepsString);
58
59 VL53L0X_Error
```

```
    VL53L0X_get_limit_check_info(VL53L0X_DEV Dev,
        uint16_t LimitCheckId,
        char *pLimitCheckString);
61
62
63 #ifdef USE_EMPTY_STRING
64     #define VL53L0X_STRING_DEVICE_INFO_NAME
65     """
66     #define
67         VL53L0X_STRING_DEVICE_INFO_NAME_TS0
68     """
69     #define
70         VL53L0X_STRING_DEVICE_INFO_NAME_TS1
71     """
72     #define VL53L0X_STRING_DEVICE_INFO_TYPE
73     """
74     /* PAL ERROR strings */
75     #define VL53L0X_STRING_ERROR_NONE
76     """
77     #define
78         VL53L0X_STRING_ERROR_CALIBRATION_WARNING
79     """
80     #define
81         VL53L0X_STRING_ERROR_MIN_CLIPPED
82     """
83     #define VL53L0X_STRING_ERROR_UNDEFINED
84     """
85     #define
86         VL53L0X_STRING_ERROR_INVALID_PARAMS
87     """
88 }
```

```
77 #define  
VL53L0X_STRING_ERROR_NOT_SUPPORTED  
""  
78 #define  
VL53L0X_STRING_ERROR_RANGE_ERROR  
""  
79 #define VL53L0X_STRING_ERROR_TIME_OUT  
""  
80 #define  
VL53L0X_STRING_ERROR_MODE_NOT_SUPPORTED  
""  
81 #define  
VL53L0X_STRING_ERROR_BUFFER_TOO_SMALL  
""  
82 #define  
VL53L0X_STRING_ERROR_GPIO_NOT_EXISTING  
""  
83 #define  
VL53L0X_STRING_ERROR_GPIO_FUNCTIONALITY_NOT_SU  
PPORTED  
""  
84 #define  
VL53L0X_STRING_ERROR_CONTROL_INTERFACE  
""  
85 #define  
VL53L0X_STRING_ERROR_INVALID_COMMAND  
""  
86 #define  
VL53L0X_STRING_ERROR_DIVISION_BY_ZERO  
""  
87 #define  
VL53L0X_STRING_ERROR_REF_SPAD_INIT  
""  
88 #define  
VL53L0X_STRING_ERROR_NOT_IMPLEMENTED  
""  
89  
90 #define
```

```
VL53L0X_STRING_UNKNOW_ERROR_CODE
"""

91
92
93
94    /* Range Status */
95    #define VL53L0X_STRING_RANGESTATUS_NONE
"""
96    #define
VL53L0X_STRING_RANGESTATUS_RANGEVALID
"""
97    #define
VL53L0X_STRING_RANGESTATUS_SIGMA
"""
98    #define
VL53L0X_STRING_RANGESTATUS_SIGNAL
"""
99    #define
VL53L0X_STRING_RANGESTATUS_MINRANGE
"""
100   #define
VL53L0X_STRING_RANGESTATUS_PHASE
"""
101   #define VL53L0X_STRING_RANGESTATUS_HW
"""

102
103
104    /* Range Status */
105    #define VL53L0X_STRING_STATE_POWERDOWN
"""
106    #define
VL53L0X_STRING_STATE_WAIT_STATICINIT
"""
107    #define VL53L0X_STRING_STATE_STANDBY
"""
108    #define VL53L0X_STRING_STATE_IDLE
"""
```

```
109     #define VL53L0X_STRING_STATE_RUNNING  
    ""  
110     #define VL53L0X_STRING_STATE_UNKNOWN  
    ""  
111     #define VL53L0X_STRING_STATE_ERROR  
    ""  
112  
113  
114     /* Device Specific */  
115     #define VL53L0X_STRING_DEVICEERROR_NONE  
    ""  
116     #define  
        VL53L0X_STRING_DEVICEERROR_VSELCONTINUITYTEST  
        FAILURE      ""  
117     #define  
        VL53L0X_STRING_DEVICEERROR_VSELWATCHDOGTESTFA  
        ILURE      ""  
118     #define  
        VL53L0X_STRING_DEVICEERROR_NOVHVVALUEFOUND  
    ""  
119     #define  
        VL53L0X_STRING_DEVICEERROR_MSRCNOTARGET  
    ""  
120     #define  
        VL53L0X_STRING_DEVICEERROR_SNRCHECK  
    ""  
121     #define  
        VL53L0X_STRING_DEVICEERROR_RANGEPHASECHECK  
    ""  
122     #define  
        VL53L0X_STRING_DEVICEERROR_SIGMATHRESHOLDCHECK  
    ""  
123     #define VL53L0X_STRING_DEVICEERROR_TCC  
    ""  
124     #define VL53L0X_STRING_DEVICEERROR_PHASECONSISTENCY  
    ""
```

```
125 #define  
VL53L0X_STRING_DEVICEERROR_MINCLIP  
""  
126 #define  
VL53L0X_STRING_DEVICEERROR_RANGECOMPLETE  
""  
127 #define  
VL53L0X_STRING_DEVICEERROR_ALGOUNDERFLOW  
""  
128 #define  
VL53L0X_STRING_DEVICEERROR_ALGOOVERFLOW  
""  
129 #define  
VL53L0X_STRING_DEVICEERROR_RANGEIGNORETHRESHOL  
D  
""  
130 #define  
VL53L0X_STRING_DEVICEERROR_UNKNOWN  
""  
131  
132 /* Check Enable */  
133 #define  
VL53L0X_STRING_CHECKENABLE_SIGMA_FINAL_RANGE  
""  
134 #define  
VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_FINAL_R  
ANGE  
""  
135 #define  
VL53L0X_STRING_CHECKENABLE_SIGNAL_REF_CLIP  
""  
136 #define  
VL53L0X_STRING_CHECKENABLE_RANGE_IGNORE_THRESH  
OLD  
""  
137  
138 /* Sequence Step */  
139 #define VL53L0X_STRING_SEQUENCESTEP_TCC  
""  
140 #define VL53L0X_STRING_SEQUENCESTEP_DSS
```

```
"""

141     #define
VL53L0X_STRING_SEQUENCESTEP_MSRC
"""

142     #define
VL53L0X_STRING_SEQUENCESTEP_PRE_RANGE
"""

143     #define
VL53L0X_STRING_SEQUENCESTEP_FINAL_RANGE
"""

144 #else
145     #define VL53L0X_STRING_DEVICE_INFO_NAME
"VL53L0X cut1.0"
146     #define
VL53L0X_STRING_DEVICE_INFO_NAME_TS0
"VL53L0X TS0"
147     #define
VL53L0X_STRING_DEVICE_INFO_NAME_TS1
"VL53L0X TS1"
148     #define
VL53L0X_STRING_DEVICE_INFO_NAME_TS2
"VL53L0X TS2"
149     #define
VL53L0X_STRING_DEVICE_INFO_NAME_ES1
"VL53L0X ES1 or later"
150     #define VL53L0X_STRING_DEVICE_INFO_TYPE
"VL53L0X"

151
152     /* PAL ERROR strings */
153     #define VL53L0X_STRING_ERROR_NONE \
154             "No Error"
155     #define
VL53L0X_STRING_ERROR_CALIBRATION_WARNING \
156             "Calibration Warning Error"
157     #define
VL53L0X_STRING_ERROR_MIN_CLIPPED \
158             "Min clipped error"
```

```
159 #define VL53L0X_STRING_ERROR_UNDEFINED \
160 "Undefined error"
161 #define VL53L0X_STRING_ERROR_INVALID_PARAMS \
162 "Invalid parameters error"
163 #define VL53L0X_STRING_ERROR_NOT_SUPPORTED \
164 "Not supported error"
165 #define VL53L0X_STRING_ERROR_RANGE_ERROR \
166 "Range error"
167 #define VL53L0X_STRING_ERROR_TIME_OUT \
168 "Time out error"
169 #define VL53L0X_STRING_ERROR_MODE_NOT_SUPPORTED \
170 "Mode not supported error"
171 #define VL53L0X_STRING_ERROR_BUFFER_TOO_SMALL \
172 "Buffer too small"
173 #define VL53L0X_STRING_ERROR_GPIO_NOT_EXISTING \
174 "GPIO not existing"
175 #define VL53L0X_STRING_ERROR_GPIO_FUNCTIONALITY_NOT_SU \
176 "PPORTED \
177 "GPIO funct not supported"
178 #define VL53L0X_STRING_ERROR_INTERRUPT_NOT_CLEARED \
179 "Interrupt not Cleared"
180 #define VL53L0X_STRING_ERROR_CONTROL_INTERFACE \
181 "Control Interface Error"
182 #define VL53L0X_STRING_ERROR_INVALID_COMMAND \
183 "Invalid Command Error"
```

```
    VL53L0X_STRING_ERROR_DIVISION_BY_ZERO \
184        "Division by zero Error"
185 #define
    VL53L0X_STRING_ERROR_REF_SPAD_INIT \
186        "Reference Spad Init Error"
187 #define
    VL53L0X_STRING_ERROR_NOT_IMPLEMENTED \
188        "Not implemented error"
189
190 #define
    VL53L0X_STRING_UNKNOW_ERROR_CODE \
191        "Unknown Error Code"
192
193
194
195 /* Range Status */
196 #define VL53L0X_STRING_RANGESTATUS_NONE
    "No Update"
197 #define
    VL53L0X_STRING_RANGESTATUS_RANGEVALID
    "Range Valid"
198 #define
    VL53L0X_STRING_RANGESTATUS_SIGMA
    "Sigma Fail"
199 #define
    VL53L0X_STRING_RANGESTATUS_SIGNAL
    "Signal Fail"
200 #define
    VL53L0X_STRING_RANGESTATUS_MINRANGE
    "Min Range Fail"
201 #define
    VL53L0X_STRING_RANGESTATUS_PHASE
    "Phase Fail"
202 #define VL53L0X_STRING_RANGESTATUS_HW
    "Hardware Fail"
203
204
```

```
205     /* Range Status */
206     #define VL53L0X_STRING_STATE_POWERDOWN
207         "POWERDOWN State"
208     #define
209         VL53L0X_STRING_STATE_WAIT_STATICINIT \
210             "Wait for staticinit State"
211     #define VL53L0X_STRING_STATE_STANDBY
212         "STANDBY State"
213     #define VL53L0X_STRING_STATE_IDLE
214         "IDLE State"
215
216     /* Device Specific */
217     #define VL53L0X_STRING_DEVICEERROR_NONE
218         "No Update"
219     #define
220         VL53L0X_STRING_DEVICEERROR_VCSELCONTINUITYTEST
221             FAILURE \
222                 "VCSEL Continuity Test Failure"
223     #define
224         VL53L0X_STRING_DEVICEERROR_VCSELWATCHDOGTESTFA
225             ILURE \
226                 "VCSEL Watchdog Test Failure"
227     #define
228         VL53L0X_STRING_DEVICEERROR_NOVHVVALUEFOUND \
229             "No VHV Value found"
230     #define
231         VL53L0X_STRING_DEVICEERROR_MSRCNOTARGET \
232             "MSRC No Target Error"
233     #define
234         VL53L0X_STRING_DEVICEERROR_SNRCHECK \
```

```
227         "SNR Check Exit"
228 #define VL53L0X_STRING_DEVICEERROR_RANGEPHASECHECK \
229         "Range Phase Check Error"
230 #define VL53L0X_STRING_DEVICEERROR_SIGMATHRESHOLDCHECK \
231         "Sigma Threshold Check Error"
232 #define VL53L0X_STRING_DEVICEERROR_TCC \
233         "TCC Error"
234 #define VL53L0X_STRING_DEVICEERROR_PHASECONSISTENCY \
235         "Phase Consistency Error"
236 #define VL53L0X_STRING_DEVICEERROR_MINCLIP \
237         "Min Clip Error"
238 #define VL53L0X_STRING_DEVICEERROR_RANGECOMPLETE \
239         "Range Complete"
240 #define VL53L0X_STRING_DEVICEERROR_ALGOUNDERFLOW \
241         "Range Algo Underflow Error"
242 #define VL53L0X_STRING_DEVICEERROR_ALGOOVERFLOW \
243         "Range Algo Overflow Error"
244 #define VL53L0X_STRING_DEVICEERROR_RANGEIGNORETHRESHOL \
245         "Range Ignore Threshold Error"
246 #define VL53L0X_STRING_DEVICEERROR_UNKNOWN \
247         "Unknown error code"
248
249 /* Check Enable */
250 #define VL53L0X_STRING_CHECKENABLE_SIGMA_FINAL_RANGE \
```

```
251         "SIGMA FINAL RANGE"
252 #define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_FINAL_R
ANGE \
253         "SIGNAL RATE FINAL RANGE"
254 #define VL53L0X_STRING_CHECKENABLE_SIGNAL_REF_CLIP \
255         "SIGNAL REF CLIP"
256 #define VL53L0X_STRING_CHECKENABLE_RANGE_IGNORE_THRESH
OLD \
257         "RANGE IGNORE THRESHOLD"
258 #define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_MSRC \
259         "SIGNAL RATE MSRC"
260 #define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_PRE_RAN
GE \
261         "SIGNAL RATE PRE RANGE"
262
263 /* Sequence Step */
264 #define VL53L0X_STRING_SEQUENCESTEP_TCC
"TCC"
265 #define VL53L0X_STRING_SEQUENCESTEP_DSS
"DSS"
266 #define VL53L0X_STRING_SEQUENCESTEP_MSRC
"MSRC"
267 #define VL53L0X_STRING_SEQUENCESTEP_PRE_RANGE
"PRE RANGE"
268 #define VL53L0X_STRING_SEQUENCESTEP_FINAL_RANGE
"FINAL RANGE"
269 #endif /* USE_EMPTY_STRING */
270
271
```

```
272 #ifdef __cplusplus  
273 }  
274 #endif  
275  
276 #endif  
277
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
doc >			

vl53l0x_doxydoc.c

Go to the documentation of this file.

```
1 | /*****  
2 | * @file PAL_doxydoc.c  no code doxygen doc  
3 | only *  
4 | */  
5 |
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries	VL53L0X_1.0.2	Api	platform
			inc

vl53l0x_i2c_platform.h

Go to the documentation of this file.

```
1  /*
2   * COPYRIGHT (C) STMicroelectronics 2014.
3   * All rights reserved.
4   *
5   * This software is the confidential and
6   * proprietary information of
7   * STMicroelectronics ("Confidential
8   * Information"). You shall not
9   * disclose such Confidential Information
10  and shall use it only in
11  *
12  * accordance with the terms of the license
13  * agreement you entered into
14  * with STMicroelectronics
15  *
16  * Programming Golden Rule: Keep it Simple!
17  *
18  */
19
20 #ifndef _VL53L0X_I2C_PLATFORM_H_
21 #define _VL53L0X_I2C_PLATFORM_H_
22
23
```

```
24 #include "vl53l0x_def.h"
25
26 #ifdef __cplusplus
27 extern "C" {
28 #endif
29
30 // Include uint8_t, unit16_t etc
31 // definitions
32
33 #include <stdint.h>
34 #include <stdarg.h>
35
36
37 // enum {TRUE = true, FALSE = false};
38
39
40 #ifndef bool_t
41 typedef unsigned char bool_t;
42 #endif
43
44
45
46 #define I2C 0x01
47 #define SPI 0x00
48
49
50 #define COMMS_BUFFER_SIZE 64 // MUST
51 be the same size as the SV task buffer
52
53
54
55 #define BYTES_PER_WORD 2
56 #define BYTES_PER_DWORD 4
57
58
59 #define VL53L0X_MAX_STRING_LENGTH_PLT
60 256
61
62
63
64
65 int32_t VL53L0X_comms_initialise(uint8_t
66 comms_type,
67
68     uint16_t comms_speed_khz);
69
70
71
72
73
74
75
76
77
```

```
85 int32_t VL53L0X_comms_close(void);
86
94 int32_t VL53L0X_cycle_power(void);
95
96
121 int32_t VL53L0X_write_multi(uint8_t address,
122     uint8_t index, uint8_t *pdata, int32_t
123     count);
122
123
148 int32_t VL53L0X_read_multi(uint8_t address,
149     uint8_t index, uint8_t *pdata, int32_t
150     count);
149
150
174 int32_t VL53L0X_write_byte(uint8_t address,
175     uint8_t index, uint8_t data);
175
176
201 int32_t VL53L0X_write_word(uint8_t address,
202     uint8_t index, uint16_t data);
202
203
228 int32_t VL53L0X_write_dword(uint8_t address,
229     uint8_t index, uint32_t data);
229
230
231
255 int32_t VL53L0X_read_byte(uint8_t address,
256     uint8_t index, uint8_t *pdata);
256
257
282 int32_t VL53L0X_read_word(uint8_t address,
283     uint8_t index, uint16_t *pdata);
283
284
309 int32_t VL53L0X_read_dword(uint8_t address,
```

```
    uint8_t index, uint32_t *pdata);
310
311
323 int32_t VL53L0X_platform_wait_us(int32_t
   wait_us);
324
325
337 int32_t VL53L0X_wait_ms(int32_t wait_ms);
338
339
349 int32_t VL53L0X_set_gpio(uint8_t level);
350
351
361 int32_t VL53L0X_get_gpio(uint8_t *plevel);
362
370 int32_t VL53L0X_release_gpio(void);
371
372
382 int32_t VL53L0X_get_timer_frequency(int32_t
   *ptimer_freq_hz);
383
393 int32_t VL53L0X_get_timer_value(int32_t
   *ptimer_count);
394
395
396
397
398
399 #ifdef __cplusplus
400 }
401#endif
402
403#endif // _VL53L0X_I2C_PLATFORM_H_
404
```



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			

vl53l0x_interrupt_threshold_settings.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright © 2016, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |       notice, this list of conditions and  
12 |       the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |       notice, this list of conditions and  
16 |       the following disclaimer in the  
17 |       documentation and/or other materials  
18 |       provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be used  
21 |       to endorse or promote products derived from  
22 |       this software without specific written  
23 |       permission.  
24 |  
25 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT  
26 | HOLDERS AND CONTRIBUTORS "AS IS" AND ANY  
27 | EXPRESS OR IMPLIED WARRANTIES, INCLUDING,  
28 | BUT NOT LIMITED TO, THE IMPLIED  
29 | WARRANTIES OF MERCHANTABILITY AND FITNESS  
30 | FOR A PARTICULAR PURPOSE ARE DISCLAIMED.  
31 | IN NO EVENT SHALL THE COPYRIGHT  
32 | OWNER OR CONTRIBUTORS BE LIABLE FOR ANY  
33 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,  
34 | EXEMPLARY, OR CONSEQUENTIAL DAMAGES  
35 | (INCLUDING, BUT NOT LIMITED TO,  
36 | PROCUREMENT OF SUBSTITUTE GOODS  
37 | OR SERVICES; LOSS OF USE, DATA,  
38 | OR PROFITS; OR BUSINESS INTERRUPTION)  
39 | HOWEVER CAUSED AND ON ANY THEORY  
40 | OF LIABILITY, WHETHER IN CONTRACT,  
41 | STRICT LIABILITY, OR TORT  
42 | (INCLUDING NEGLIGENCE OR OTHERWISE)  
43 | ARISING IN ANY WAY OUT OF THE USE  
44 | OF THIS SOFTWARE, EVEN IF ADVISED  
45 | OF THE POSSIBILITY OF SUCH DAMAGE.
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

29 |

30 | #ifndef
_VL53L0X_INTERRUPT_THRESHOLD_SETTINGS_H_
31 | #define
_VL53L0X_INTERRUPT_THRESHOLD_SETTINGS_H_
32 |

```
33
34 #ifdef __cplusplus
35 extern "C" {
36 #endif
37
38
39 uint8_t InterruptThresholdSettings[] = {
40
41     /* Start of Interrupt Threshold Settings
 */
42     0x1, 0xff, 0x00,
43     0x1, 0x80, 0x01,
44     0x1, 0xff, 0x01,
45     0x1, 0x00, 0x00,
46     0x1, 0xff, 0x01,
47     0x1, 0x4f, 0x02,
48     0x1, 0xFF, 0x0E,
49     0x1, 0x00, 0x03,
50     0x1, 0x01, 0x84,
51     0x1, 0x02, 0x0A,
52     0x1, 0x03, 0x03,
53     0x1, 0x04, 0x08,
54     0x1, 0x05, 0xC8,
55     0x1, 0x06, 0x03,
56     0x1, 0x07, 0x8D,
57     0x1, 0x08, 0x08,
58     0x1, 0x09, 0xC6,
59     0x1, 0x0A, 0x01,
60     0x1, 0x0B, 0x02,
61     0x1, 0x0C, 0x00,
62     0x1, 0x0D, 0xD5,
63     0x1, 0x0E, 0x18,
64     0x1, 0x0F, 0x12,
65     0x1, 0x10, 0x01,
66     0x1, 0x11, 0x82,
67     0x1, 0x12, 0x00,
68     0x1, 0x13, 0xD5,
```

69	0x1, 0x14, 0x18,
70	0x1, 0x15, 0x13,
71	0x1, 0x16, 0x03,
72	0x1, 0x17, 0x86,
73	0x1, 0x18, 0x0A,
74	0x1, 0x19, 0x09,
75	0x1, 0x1A, 0x08,
76	0x1, 0x1B, 0xC2,
77	0x1, 0x1C, 0x03,
78	0x1, 0x1D, 0x8F,
79	0x1, 0x1E, 0x0A,
80	0x1, 0x1F, 0x06,
81	0x1, 0x20, 0x01,
82	0x1, 0x21, 0x02,
83	0x1, 0x22, 0x00,
84	0x1, 0x23, 0xD5,
85	0x1, 0x24, 0x18,
86	0x1, 0x25, 0x22,
87	0x1, 0x26, 0x01,
88	0x1, 0x27, 0x82,
89	0x1, 0x28, 0x00,
90	0x1, 0x29, 0xD5,
91	0x1, 0x2A, 0x18,
92	0x1, 0x2B, 0x0B,
93	0x1, 0x2C, 0x28,
94	0x1, 0x2D, 0x78,
95	0x1, 0x2E, 0x28,
96	0x1, 0x2F, 0x91,
97	0x1, 0x30, 0x00,
98	0x1, 0x31, 0x0B,
99	0x1, 0x32, 0x00,
100	0x1, 0x33, 0x0B,
101	0x1, 0x34, 0x00,
102	0x1, 0x35, 0xA1,
103	0x1, 0x36, 0x00,
104	0x1, 0x37, 0xA0,
105	0x1, 0x38, 0x00,

106	0x1, 0x39, 0x04,
107	0x1, 0x3A, 0x28,
108	0x1, 0x3B, 0x30,
109	0x1, 0x3C, 0x0C,
110	0x1, 0x3D, 0x04,
111	0x1, 0x3E, 0x0F,
112	0x1, 0x3F, 0x79,
113	0x1, 0x40, 0x28,
114	0x1, 0x41, 0x1E,
115	0x1, 0x42, 0x2F,
116	0x1, 0x43, 0x87,
117	0x1, 0x44, 0x00,
118	0x1, 0x45, 0x0B,
119	0x1, 0x46, 0x00,
120	0x1, 0x47, 0x0B,
121	0x1, 0x48, 0x00,
122	0x1, 0x49, 0xA7,
123	0x1, 0x4A, 0x00,
124	0x1, 0x4B, 0xA6,
125	0x1, 0x4C, 0x00,
126	0x1, 0x4D, 0x04,
127	0x1, 0x4E, 0x01,
128	0x1, 0x4F, 0x00,
129	0x1, 0x50, 0x00,
130	0x1, 0x51, 0x80,
131	0x1, 0x52, 0x09,
132	0x1, 0x53, 0x08,
133	0x1, 0x54, 0x01,
134	0x1, 0x55, 0x00,
135	0x1, 0x56, 0x0F,
136	0x1, 0x57, 0x79,
137	0x1, 0x58, 0x09,
138	0x1, 0x59, 0x05,
139	0x1, 0x5A, 0x00,
140	0x1, 0x5B, 0x60,
141	0x1, 0x5C, 0x05,
142	0x1, 0x5D, 0xD1,

143	0x1, 0x5E, 0x0C,
144	0x1, 0x5F, 0x3C,
145	0x1, 0x60, 0x00,
146	0x1, 0x61, 0xD0,
147	0x1, 0x62, 0x0B,
148	0x1, 0x63, 0x03,
149	0x1, 0x64, 0x28,
150	0x1, 0x65, 0x10,
151	0x1, 0x66, 0x2A,
152	0x1, 0x67, 0x39,
153	0x1, 0x68, 0x0B,
154	0x1, 0x69, 0x02,
155	0x1, 0x6A, 0x28,
156	0x1, 0x6B, 0x10,
157	0x1, 0x6C, 0x2A,
158	0x1, 0x6D, 0x61,
159	0x1, 0x6E, 0x0C,
160	0x1, 0x6F, 0x00,
161	0x1, 0x70, 0x0F,
162	0x1, 0x71, 0x79,
163	0x1, 0x72, 0x00,
164	0x1, 0x73, 0x0B,
165	0x1, 0x74, 0x00,
166	0x1, 0x75, 0x0B,
167	0x1, 0x76, 0x00,
168	0x1, 0x77, 0xA1,
169	0x1, 0x78, 0x00,
170	0x1, 0x79, 0xA0,
171	0x1, 0x7A, 0x00,
172	0x1, 0x7B, 0x04,
173	0x1, 0xFF, 0x04,
174	0x1, 0x79, 0x1D,
175	0x1, 0x7B, 0x27,
176	0x1, 0x96, 0x0E,
177	0x1, 0x97, 0xFE,
178	0x1, 0x98, 0x03,
179	0x1, 0x99, 0xEF,

```
180     0x1, 0x9A, 0x02,
181     0x1, 0x9B, 0x44,
182     0x1, 0x73, 0x07,
183     0x1, 0x70, 0x01,
184     0x1, 0xff, 0x01,
185     0x1, 0x00, 0x01,
186     0x1, 0xff, 0x00,
187     0x00, 0x00, 0x00
188 };
189
190 #ifdef __cplusplus
191 }
192#endif
193
194#endif /*  

 _VL53L0X_INTERRUPT_THRESHOLD_SETTINGS_H_ */
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › platform › inc ›			

vl53l0x_platform_log.h

Go to the documentation of this file.

```
1 | /*****  
2 | Copyright © 2015, STMicroelectronics  
3 | International N.V.  
4 | All rights reserved.  
5 | Redistribution and use in source and binary  
6 | forms, with or without  
7 | modification, are permitted provided that  
8 | the following conditions are met:  
9 |     * Redistributions of source code must  
10 |       retain the above copyright  
11 |           notice, this list of conditions and  
12 |           the following disclaimer.  
13 |     * Redistributions in binary form must  
14 |       reproduce the above copyright  
15 |           notice, this list of conditions and  
16 |           the following disclaimer in the  
17 |           documentation and/or other materials  
18 |           provided with the distribution.  
19 |     * Neither the name of STMicroelectronics  
20 |       nor the names of its contributors may be used  
21 |       to endorse or promote products derived from  
22 |       this software without specific written  
23 |       permission.  
24 |  
25 |  
26 |
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

29 |

30 | #ifndef _VL53L0X_PLATFORM_LOG_H_
31 | #define _VL53L0X_PLATFORM_LOG_H_

32 |

33 | #include <stdio.h>
34 | #include <string.h>

```
35 /* LOG Functions */
36
37 #ifdef __cplusplus
38 extern "C" {
39 #endif
40
41 // #define VL53L0X_LOG_ENABLE 0
42
43 enum {
44     TRACE_LEVEL_NONE,
45     TRACE_LEVEL_ERRORS,
46     TRACE_LEVEL_WARNING,
47     TRACE_LEVEL_INFO,
48     TRACE_LEVEL_DEBUG,
49     TRACE_LEVEL_ALL,
50     TRACE_LEVEL_IGNORE
51 };
52
53 enum {
54     TRACE_FUNCTION_NONE = 0,
55     TRACE_FUNCTION_I2C = 1,
56     TRACE_FUNCTION_ALL = 0xffffffff //all
57     bits except sign
58 };
59
60 enum {
61     TRACE_MODULE_NONE = 0x0,
62     TRACE_MODULE_API = 0x1,
63     TRACE_MODULE_PLATFORM = 0x2,
64     TRACE_MODULE_ALL =
65     0xffffffff //all bits except sign
66 };
67
68
69 #ifdef VL53L0X_LOG_ENABLE
70
71 #include <sys/time.h>
```

```
76
77 extern uint32_t _trace_level;
78
79
80
81 int32_t VL53L0X_trace_config(char *filename,
82     uint32_t modules, uint32_t level, uint32_t
83     functions);
84
85
86 //extern FILE * log_file;
87
88 #define LOG_GET_TIME() (int)clock()
89
90 #define _LOG_FUNCTION_START(module, fmt, ...
91     ) \
92     trace_print_module_function(module,
93     _trace_level, TRACE_FUNCTION_ALL, "%ld <START>
94     %s "fmt"\n", LOG_GET_TIME(), __FUNCTION__,
95     ##__VA_ARGS__);
96 #define _LOG_FUNCTION_END(module, status,
97     ...) \
98     trace_print_module_function(module,
99     _trace_level, TRACE_FUNCTION_ALL, "%ld <END>
100    %s %d\n", LOG_GET_TIME(), __FUNCTION__,
101    (int)status, ##__VA_ARGS__)
102 #define _LOG_FUNCTION_END_FMT(module,
103     status, fmt, ... )\
104     trace_print_module_function(module,
105     _trace_level, TRACE_FUNCTION_ALL, "%ld <END>
106     %s %d "fmt"\n", LOG_GET_TIME(), __FUNCTION__,
```

```
(int)status,##__VA_ARGS__)
98 // __func__ is gcc only
100 //#define VL53L0X_ErrLog( fmt, ... )
101     fprintf(stderr, "VL53L0X_ErrLog %s" fmt "\n",
102             __func__, ##__VA_ARGS__)
103 #else /* VL53L0X_LOG_ENABLE no logging */
104     #define VL53L0X_ErrLog(...) (void)0
105     #define _LOG_FUNCTION_START(module, fmt,
106                                ...) (void)0
107     #define _LOG_FUNCTION_END(module,
108                               status, ...) (void)0
109     #define _LOG_FUNCTION_END_FMT(module,
110                                 status, fmt, ...) (void)0
111 #endif /* else */
112
113 #define VL53L0X_COPYSTRING(str, ...)
114     strcpy(str, ##__VA_ARGS__)
115 #ifdef __cplusplus
116 }
117#endif /* _VL53L0X_PLATFORM_LOG_H_ */
```



VL53L0X API Specification

1.0.2.4823

Main Page	Related Pages	Modules	Data Structures
Files			
File List	Globals		
deliveries › VL53L0X_1.0.2 › Api › core › inc ›			

vl53l0x_tuning.h

Go to the documentation of this file.

```
1 | /*****
2 | ****
3 | Copyright ? 2016, STMicroelectronics
4 | International N.V.
5 | All rights reserved.
6 |
7 | Redistribution and use in source and binary
8 | forms, with or without
9 | modification, are permitted provided that
10 | the following conditions are met:
11 |     * Redistributions of source code must
12 |       retain the above copyright
13 |     notice, this list of conditions and
14 |     the following disclaimer.
15 |     * Redistributions in binary form must
16 |       reproduce the above copyright
17 |     notice, this list of conditions and
18 |     the following disclaimer in the
19 |     documentation and/or other materials
20 |     provided with the distribution.
21 |     * Neither the name of STMicroelectronics
```

nor the

13 | names of its contributors may be used
to endorse or promote products

14 | derived from this software without
specific prior written permission.

15 |

16 | THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT
HOLDERS AND CONTRIBUTORS "AS IS" AND

17 | ANY EXPRESS OR IMPLIED WARRANTIES,
INCLUDING, BUT NOT LIMITED TO, THE IMPLIED

18 | WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
PARTICULAR PURPOSE, AND

19 | NON-INFRINGEMENT OF INTELLECTUAL PROPERTY
RIGHTS ARE DISCLAIMED.

20 | IN NO EVENT SHALL STMICROELECTRONICS
INTERNATIONAL N.V. BE LIABLE FOR ANY

21 | DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
EXEMPLARY, OR CONSEQUENTIAL DAMAGES

22 | (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT
OF SUBSTITUTE GOODS OR SERVICES;

23 | LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
INTERRUPTION) HOWEVER CAUSED AND

24 | ON ANY THEORY OF LIABILITY, WHETHER IN
CONTRACT, STRICT LIABILITY, OR TORT

25 | (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING
IN ANY WAY OUT OF THE USE OF THIS

26 | SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY
OF SUCH DAMAGE.

27 | *****/

28 |

29 |

30 | #ifndef _VL53L0X_TUNING_H_
31 | #define _VL53L0X_TUNING_H_

32 |

33 | #include "vl53l0x_def.h"

34 |

```
35
36 #ifdef __cplusplus
37 extern "C" {
38#endif
39
40
41 uint8_t DefaultTuningSettings[] = {
42
43     /* update 02/11/2015_v36 */
44     0x01, 0xFF, 0x01,
45     0x01, 0x00, 0x00,
46
47     0x01, 0xFF, 0x00,
48     0x01, 0x09, 0x00,
49     0x01, 0x10, 0x00,
50     0x01, 0x11, 0x00,
51
52     0x01, 0x24, 0x01,
53     0x01, 0x25, 0xff,
54     0x01, 0x75, 0x00,
55
56     0x01, 0xFF, 0x01,
57     0x01, 0x4e, 0x2c,
58     0x01, 0x48, 0x00,
59     0x01, 0x30, 0x20,
60
61     0x01, 0xFF, 0x00,
62     0x01, 0x30, 0x09, /* mja changed from
63     0x64. */
64     0x01, 0x54, 0x00,
65     0x01, 0x31, 0x04,
66     0x01, 0x32, 0x03,
67     0x01, 0x40, 0x83,
68     0x01, 0x46, 0x25,
69     0x01, 0x60, 0x00,
70     0x01, 0x27, 0x00,
```

71	0x01, 0x51, 0x00,
72	0x01, 0x52, 0x96,
73	0x01, 0x56, 0x08,
74	0x01, 0x57, 0x30,
75	0x01, 0x61, 0x00,
76	0x01, 0x62, 0x00,
77	0x01, 0x64, 0x00,
78	0x01, 0x65, 0x00,
79	0x01, 0x66, 0xa0,
80	
81	0x01, 0xFF, 0x01,
82	0x01, 0x22, 0x32,
83	0x01, 0x47, 0x14,
84	0x01, 0x49, 0xff,
85	0x01, 0x4a, 0x00,
86	
87	0x01, 0xFF, 0x00,
88	0x01, 0x7a, 0xa0,
89	0x01, 0x7b, 0x00,
90	0x01, 0x78, 0x21,
91	
92	0x01, 0xFF, 0x01,
93	0x01, 0x23, 0x34,
94	0x01, 0x42, 0x00,
95	0x01, 0x44, 0xff,
96	0x01, 0x45, 0x26,
97	0x01, 0x46, 0x05,
98	0x01, 0x40, 0x40,
99	0x01, 0x0E, 0x06,
100	0x01, 0x20, 0x1a,
101	0x01, 0x43, 0x40,
102	
103	0x01, 0xFF, 0x00,
104	0x01, 0x34, 0x03,
105	0x01, 0x35, 0x44,
106	
107	0x01, 0xFF, 0x01,

```
108      0x01, 0x31, 0x04,
109      0x01, 0x4b, 0x09,
110      0x01, 0x4c, 0x05,
111      0x01, 0x4d, 0x04,
112
113
114      0x01, 0xFF, 0x00,
115      0x01, 0x44, 0x00,
116      0x01, 0x45, 0x20,
117      0x01, 0x47, 0x08,
118      0x01, 0x48, 0x28,
119      0x01, 0x67, 0x00,
120      0x01, 0x70, 0x04,
121      0x01, 0x71, 0x01,
122      0x01, 0x72, 0xfe,
123      0x01, 0x76, 0x00,
124      0x01, 0x77, 0x00,
125
126      0x01, 0xFF, 0x01,
127      0x01, 0xd, 0x01,
128
129      0x01, 0xFF, 0x00,
130      0x01, 0x80, 0x01,
131      0x01, 0x01, 0xF8,
132
133      0x01, 0xFF, 0x01,
134      0x01, 0x8e, 0x01,
135      0x01, 0x00, 0x01,
136      0x01, 0xFF, 0x00,
137      0x01, 0x80, 0x00,
138
139      0x00, 0x00, 0x00
140  };
141
142 #ifdef __cplusplus
143 }
144#endif
```

```
145
146 #endif /* _VL53L0X_TUNING_H_ */
```

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

doc

doc Directory Reference

Files

file [**PAL_disclaimer.c**](#) [code]
no code doxygen doc only

file [**vl53l0x_doxydoc.c**](#) [code]

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

deliveries

deliveries Directory Reference

Directories

directory **VL53L0X_1.0.2**

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

deliveries > VL53L0X_1.0.2 >

VL53L0X_1.0.2 Directory Reference

Directories

[directory](#) [Api](#)

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[deliveries](#) › [VL53L0X_1.0.2](#) › [Api](#) ›

Api Directory Reference

Directories

directory **core**

directory **platform**

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[deliveries](#) › [VL53L0X_1.0.2](#) › [Api](#) › [core](#) ›

core Directory Reference

Directories

directory **inc**

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[deliveries](#) › [VL53L0X_1.0.2](#) › [Api](#) › [core](#) › [inc](#) ›

inc Directory Reference

Files

file [**vl53l0x_api.h**](#) [code]

file [**vl53l0x_api_calibration.h**](#) [code]

file [**vl53l0x_api_core.h**](#) [code]

file [**vl53l0x_api_ranging.h**](#) [code]

file [**vl53l0x_api_strings.h**](#) [code]

file [**vl53l0x_def.h**](#) [code]

Type definitions for VL53L0X API.

file [**vl53l0x_device.h**](#) [code]

file [**vl53l0x_interrupt_threshold_settings.h**](#) [code]

file [**vl53l0x_tuning.h**](#) [code]

Generated by Doxygen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[deliveries](#) › [VL53L0X_1.0.2](#) › [Api](#) › [platform](#) ›

platform Directory Reference

Directories

directory **inc**

Generated by DoxyGen (1.8.9.1)



VL53L0X API Specification

1.0.2.4823

Main Page

Related Pages

Modules

Data Structures

Files

[deliveries](#)

[VL53L0X_1.0.2](#)

[Api](#)

[platform](#)

[inc](#)

inc Directory Reference

Files

file [**vl53l0x_i2c_platform.h**](#) [code]

file [**vl53l0x_platform.h**](#) [code]

Function prototype definitions for Ewok Platform layer.

file [**vl53l0x_platform_log.h**](#) [code]

platform log function definition

file [**vl53l0x_types.h**](#) [code]

VL53L0X types definition.

Generated by DoxyGen (1.8.9.1)