VL53L0X API Specification

Version 1.0.4.4960
1/2/2017 4:42:00 PM

Table of Contents

Documentation

Introduction

The Photonics Abstraction Layer (PAL) is intended to provide an API functions to aid the development of applications.

Overview

This document is intended to aid in the development of applications around PAL sensor family and describes the various API functions provided by the API delivered by ST as open source C code.

Some of the API files are hardware and platform dependent (specially I2C access) so need to be adapted to the platform used by the customer.

Device Info from API

The API provide a function that can be used to obtain information of the device used like the cut version. This function is *VL53L0X_GetDeviceInfo()* .

Coding Standards

The implementation of this API will follow Linux Kernel rules as defined in https://www.kernel.org/doc/Documentation/CodingStyle

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*

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

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 Lenght 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(str, ##__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) < VL53L0X_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 greather or equal to the max lenght described before.

```
void print_range_status(VL53L0X_RangingMeasurementData_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);
}
```

Disclaimer

Module Index

Modules

Here is a list of all modules:

Data Structure Index

Data Structures

Here are the data structures with brief descriptions:

File Index

File List

Here is a list of all files with brief descriptions:

Module Documentation

VL53L0X Platform Functions

VL53L0X Platform Functions.

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. Macros

- #define PALDevDataGet(Dev, field) (Dev->Data.field)
  Get ST private structure VL53L0X_DevData_t data access.
- #define PALDevDataSet(Dev, field, data) (Dev->Data.field)=(data)
  Set ST private structure VL53L0X_DevData_t data field.

Typedefs

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

Functions

- VL53L0X_Error VL53L0X_PollingDelay (VL53L0X_DEV Dev)
  execute delay in all polling API call

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 need 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

PAL Register Access Functions

PAL Register Access Functions.

Functions

- VL53L0X_Error VL53L0X_LockSequenceAccess (VL53L0X_DEV Dev)
  Lock comms interface to serialize all commands to a shared I2C interface for a specific device.
- VL53L0X_Error VL53L0X_UnlockSequenceAccess (VL53L0X_DEV Dev)
  Unlock comms interface to serialize all commands to a shared I2C interface for a specific device.
- VL53L0X_Error VL53L0X_WriteMulti (VL53L0X_DEV Dev, uint8_t index, uint8_t *pdata, uint32_t count)
  Writes the supplied byte buffer to the device.
- 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.
- VL53L0X_Error VL53L0X_WrByte (VL53L0X_DEV Dev, uint8_t index, uint8_t data)
  Write single byte register.
- VL53L0X_Error VL53L0X_WrWord (VL53L0X_DEV Dev, uint8_t index, uint16_t data)
  Write word register.
- VL53L0X_Error VL53L0X_WrDWord (VL53L0X_DEV Dev, uint8_t index, uint32_t data)
  Write double word (4 byte) register.
- VL53L0X_Error VL53L0X_RdByte (VL53L0X_DEV Dev, uint8_t index, uint8_t *data)
  Read single byte register.
- VL53L0X_Error VL53L0X_RdWord (VL53L0X_DEV Dev, uint8_t index, uint16_t *data)
  Read word (2byte) register.
- VL53L0X_Error VL53L0X_RdDWord (VL53L0X_DEV Dev, uint8_t index, uint32_t *data)
  Read dword (4byte) register.
- VL53L0X_Error VL53L0X_UpdateByte (VL53L0X_DEV Dev, uint8_t index, uint8_t AndData, uint8_t OrData)
  Threat safe Update (read/modify/write) single byte register.

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*)

Threat 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

Basic type definition

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

VL53L0X cut1.1 Function Definition

VL53L0X cut1.1 Function Definition.

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.

VL53L0X General Functions

General functions and definitions.

Functions

- VL53L0X_API VL53L0X_Error VL53L0X_GetVersion (VL53L0X_Version_t *pVersion)
  Return the VL53L0X PAL Implementation Version.
- VL53L0X_API VL53L0X_Error VL53L0X_GetPalSpecVersion (VL53L0X_Version_t *pPalSpecVersion)
  Return the PAL Specification Version used for the current implementation.
- VL53L0X_API VL53L0X_Error VL53L0X_GetProductRevision (VL53L0X_DEV Dev, uint8_t *pProductRevisionMajor, uint8_t *pProductRevisionMinor)
  Reads the Product Revision for a for given Device This function can be used to distinguish cut1.0 from cut1.1.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceInfo (VL53L0X_DEV Dev, VL53L0X_DeviceInfo_t *pVL53L0X_DeviceInfo)
  Reads the Device information for given Device.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceErrorStatus (VL53L0X_DEV Dev, VL53L0X_DeviceError *pDeviceErrorStatus)
  Read current status of the error register for the selected device.
- VL53L0X_API VL53L0X_Error VL53L0X_GetRangeStatusString (uint8_t RangeStatus, char *pRangeStatusString)
  Human readable Range Status string for a given RangeStatus.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceErrorString (VL53L0X_DeviceError ErrorCode, char *pDeviceErrorString)
  Human readable error string for a given Error Code.
- VL53L0X_API VL53L0X_Error VL53L0X_GetPalErrorString (VL53L0X_Error PalErrorCode, char *pPalErrorString)
  Human readable error string for current PAL error status.
- VL53L0X_API VL53L0X_Error VL53L0X_GetPalStateString (VL53L0X_State PalStateCode, char *pPalStateString)
  Human readable PAL State string.
- VL53L0X_API VL53L0X_Error VL53L0X_GetPalState (VL53L0X_DEV Dev, VL53L0X_State *pPalState)
  Reads the internal state of the PAL for a given Device.
- 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.

- VL53L0X_API VL53L0X_Error VL53L0X_GetPowerMode (VL53L0X_DEV Dev, VL53L0X_PowerModes *pPowerMode)
  Get the power mode for a given Device.
- VL53L0X_API VL53L0X_Error VL53L0X_SetOffsetCalibrationDataMicroMeter (VL53L0X_DEV Dev, int32_t OffsetCalibrationDataMicroMeter)
  Set or over-hide part to part calibration offset.
- VL53L0X_API VL53L0X_Error VL53L0X_GetOffsetCalibrationDataMicroMeter (VL53L0X_DEV Dev, int32_t *pOffsetCalibrationDataMicroMeter)
  Get part to part calibration offset.
- VL53L0X_API VL53L0X_Error VL53L0X_SetLinearityCorrectiveGain (VL53L0X_DEV Dev, int16_t LinearityCorrectiveGain)
  Set the linearity corrective gain.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLinearityCorrectiveGain (VL53L0X_DEV Dev, uint16_t *pLinearityCorrectiveGain)
  Get the linearity corrective gain.
- VL53L0X_API VL53L0X_Error VL53L0X_SetGroupParamHold (VL53L0X_DEV Dev, uint8_t GroupParamHold)
  Set Group parameter Hold state.
- VL53L0X_API VL53L0X_Error VL53L0X_GetUpperLimitMilliMeter (VL53L0X_DEV Dev, uint16_t *pUpperLimitMilliMeter)
  Get the maximal distance for actual setup.
- VL53L0X_Error VL53L0X_GetTotalSignalRate (VL53L0X_DEV Dev, FixPoint1616_t *pTotalSignalRate)
  Get the Total Signal Rate.

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 * *pPalSpecVersion*)

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 for 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
pProductRevision Major	Pointer to Product Revision Major for a given Device
pProductRevision Minor	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_Device Info	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 * *pDeviceErrorStatus*)

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

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 *OffsetCalibrationDataMicroMeter*)

Set or over-hide part to part calibration offset.

See also:

VL53L0X_DataInit() VL53L0X_GetOffsetCalibrationDataMicroMeter()

Note:

This function Access to the device

Parameters:

Dev	Device Handle
OffsetCalibration DataMicroMeter	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 backup device NVM value

Note:

This function Access to the device

Parameters:

Dev	Device Handle
pOffsetCalibration DataMicroMeter	Return part to part calibration offset from device (microns)

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 * *pLinearityCorrectiveGain*)

Get the linearity corrective gain.

Function Description

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

Note:

This function Access to the device

Parameters:

Dev	Device Handle
pLinearityCorrectiveGain	Pointer to the 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_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 * *pUpperLimitMilliMeter*)

Get the maximal distance for actual setup.

Function Description

Device must be initialized through *VL53L0X_SetParameters()* prior calling 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 Init Functions

VL53L0X Init Functions.

Functions

- VL53L0X_API VL53L0X_Error VL53L0X_SetDeviceAddress (VL53L0X_DEV Dev, uint8_t DeviceAddress)
  Set new device address.
- VL53L0X_API VL53L0X_Error VL53L0X_DataInit (VL53L0X_DEV Dev)
  One time device initialization.

- VL53L0X_API VL53L0X_Error VL53L0X_SetTuningSettingBuffer (VL53L0X_DEV Dev, uint8_t *pTuningSettingBuffer, uint8_t UseInternalTuningSettings)
  Set the tuning settings pointer.
- 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.
- 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.
- 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.
- 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.

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 * *pTuningSettingBuffer*, uint8_t *UseInternalTuningSettings*)

Set the tuning settings pointer.

This function is used to specify the Tuning settings buffer to be used for a given device. The buffer contains all the necessary data to permit the API to write tuning settings. This function permit to force the usage of either external or internal tuning 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 value. of the 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 Parameters Functions

Functions used to prepare and setup the device.

Functions

- VL53L0X_API VL53L0X_Error VL53L0X_SetDeviceParameters (VL53L0X_DEV Dev, const VL53L0X_DeviceParameters_t *pDeviceParameters)
  Prepare device for operation.

- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceParameters (VL53L0X_DEV Dev, VL53L0X_DeviceParameters_t *pDeviceParameters)
  Retrieve current device parameters.
- VL53L0X_API VL53L0X_Error VL53L0X_SetDeviceMode (VL53L0X_DEV Dev, VL53L0X_DeviceModes DeviceMode)
  Set a new device mode.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceMode (VL53L0X_DEV Dev, VL53L0X_DeviceModes *pDeviceMode)
  Get current new device mode.
- VL53L0X_API VL53L0X_Error VL53L0X_SetRangeFractionEnable (VL53L0X_DEV Dev, uint8_t Enable)
  Sets the resolution of range measurements.
- VL53L0X_API VL53L0X_Error VL53L0X_GetFractionEnable (VL53L0X_DEV Dev, uint8_t *pEnable)
  Gets the fraction enable parameter indicating the resolution of range measurements.
- VL53L0X_API VL53L0X_Error VL53L0X_SetHistogramMode (VL53L0X_DEV Dev, VL53L0X_HistogramModes HistogramMode)
  Set a new Histogram mode.
- VL53L0X_API VL53L0X_Error VL53L0X_GetHistogramMode (VL53L0X_DEV Dev, VL53L0X_HistogramModes *pHistogramMode)
  Get current new device mode.
- VL53L0X_API VL53L0X_Error VL53L0X_SetMeasurementTimingBudgetMicroSeconds (VL53L0X_DEV Dev, uint32_t MeasurementTimingBudgetMicroSeconds)
  Set Ranging Timing Budget in microseconds.
- VL53L0X_API VL53L0X_Error VL53L0X_GetMeasurementTimingBudgetMicroSeconds (VL53L0X_DEV Dev, uint32_t *pMeasurementTimingBudgetMicroSeconds)
  Get Ranging Timing Budget in microseconds.
- VL53L0X_API VL53L0X_Error VL53L0X_GetVcselPulsePeriod (VL53L0X_DEV Dev, VL53L0X_VcselPeriod VcselPeriodType, uint8_t *pVCSELPulsePeriod)
  Gets the VCSEL pulse period.
- VL53L0X_API VL53L0X_Error VL53L0X_SetVcselPulsePeriod (VL53L0X_DEV Dev, VL53L0X_VcselPeriod VcselPeriodType, uint8_t VCSELPulsePeriod)
  Sets the VCSEL pulse period.
- VL53L0X_API VL53L0X_Error VL53L0X_SetSequenceStepEnable (VL53L0X_DEV Dev, VL53L0X_SequenceStepId SequenceStepId, uint8_t SequenceStepEnabled)
  Sets the (on/off) state of a requested sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepEnable (VL53L0X_DEV Dev, VL53L0X_SequenceStepId SequenceStepId, uint8_t *pSequenceStepEnabled)
  Gets the (on/off) state of a requested sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepEnables (VL53L0X_DEV Dev, VL53L0X_SchedulerSequenceSteps_t *pSchedulerSequenceSteps)
  Gets the (on/off) state of all sequence steps.
- VL53L0X_API VL53L0X_Error VL53L0X_SetSequenceStepTimeout (VL53L0X_DEV Dev, VL53L0X_SequenceStepId SequenceStepId, FixPoint1616_t TimeOutMilliSecs)
  Sets the timeout of a requested sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepTimeout (VL53L0X_DEV Dev, VL53L0X_SequenceStepId SequenceStepId, FixPoint1616_t *pTimeOutMilliSecs)
  Gets the timeout of a requested sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_GetNumberOfSequenceSteps (VL53L0X_DEV Dev, uint8_t *pNumberOfSequenceSteps)
  Gets number of sequence steps managed by the API.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepsInfo (VL53L0X_SequenceStepId SequenceStepId, char *pSequenceStepsString)

Gets the name of a given sequence step.

- VL53L0X_API VL53L0X_Error VL53L0X_SetInterMeasurementPeriodMilliSeconds (VL53L0X_DEV Dev, uint32_t InterMeasurementPeriodMilliSeconds)
  Program continuous mode Inter-Measurement period in milliseconds.
- VL53L0X_API VL53L0X_Error VL53L0X_GetInterMeasurementPeriodMilliSeconds (VL53L0X_DEV Dev, uint32_t *pInterMeasurementPeriodMilliSeconds)
  Get continuous mode Inter-Measurement period in milliseconds.
- VL53L0X_API VL53L0X_Error VL53L0X_SetXTalkCompensationEnable (VL53L0X_DEV Dev, uint8_t XTalkCompensationEnable)
  Enable/Disable Cross talk compensation feature.
- VL53L0X_API VL53L0X_Error VL53L0X_GetXTalkCompensationEnable (VL53L0X_DEV Dev, uint8_t *pXTalkCompensationEnable)
  Get Cross talk compensation rate.
- VL53L0X_API VL53L0X_Error VL53L0X_SetXTalkCompensationRateMegaCps (VL53L0X_DEV Dev, FixPoint1616_t XTalkCompensationRateMegaCps)
  Set Cross talk compensation rate.
- VL53L0X_API VL53L0X_Error VL53L0X_GetXTalkCompensationRateMegaCps (VL53L0X_DEV Dev, FixPoint1616_t *pXTalkCompensationRateMegaCps)
  Get Cross talk compensation rate.
- VL53L0X_API VL53L0X_Error VL53L0X_SetRefCalibration (VL53L0X_DEV Dev, uint8_t 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 (uint16_t *pNumberOfLimitCheck)
  Get the number of the check limit managed by a given Device.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckInfo (VL53L0X_DEV Dev, uint16_t LimitCheckId, char *pLimitCheckString)
  Return a description string for a given limit check number.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckStatus (VL53L0X_DEV Dev, uint16_t LimitCheckId, uint8_t *pLimitCheckStatus)
  Return a the Status of the specified check limit.
- VL53L0X_API VL53L0X_Error VL53L0X_SetLimitCheckEnable (VL53L0X_DEV Dev, uint16_t LimitCheckId, uint8_t LimitCheckEnable)
  Enable/Disable a specific limit check.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckEnable (VL53L0X_DEV Dev, uint16_t LimitCheckId, uint8_t *pLimitCheckEnable)
  Get specific limit check enable state.
- VL53L0X_API VL53L0X_Error VL53L0X_SetLimitCheckValue (VL53L0X_DEV Dev, uint16_t LimitCheckId, FixPoint1616_t LimitCheckValue)
  Set a specific limit check value.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckValue (VL53L0X_DEV Dev, uint16_t LimitCheckId, FixPoint1616_t *pLimitCheckValue)
  Get a specific limit check value.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckCurrent (VL53L0X_DEV Dev, uint16_t LimitCheckId, FixPoint1616_t *pLimitCheckCurrent)
  Get the current value of the signal used for the limit check.
- VL53L0X_API VL53L0X_Error VL53L0X_SetWrapAroundCheckEnable (VL53L0X_DEV Dev, uint8_t WrapAroundCheckEnable)
  Enable (or disable) Wrap around Check.

- VL53L0X_API VL53L0X_Error VL53L0X_GetWrapAroundCheckEnable (VL53L0X_DEV Dev, uint8_t *pWrapAroundCheckEnable)
  Get setup of Wrap around Check.

Detailed Description

Functions used to prepare and setup the device.

Function Documentation

VL53L0X_API VL53L0X_Error VL53L0X_SetDeviceParameters (VL53L0X_DEV *Dev*, const VL53L0X_DeviceParameters_t * *pDeviceParameters*)

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 *Dev*, VL53L0X_DeviceParameters_t * *pDeviceParameters*)

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 *Dev*, VL53L0X_DeviceModes *DeviceMode*)

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_DEVICEMODE_SINGLE_RANGING VL53L0X_DEVICEMODE_CONTINUOUS_RANGING VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY VL53L0X_HISTOGRAMMODE_RETURN_ONLY VL53L0X_HISTOGRAMMODE_BOTH

Returns:

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_MODE_NOT_SUPPORTED This error occurs when DeviceMode is not in the supported list

VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceMode (VL53L0X_DEV *Dev*, VL53L0X_DeviceModes * *pDeviceMode*)

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_RANGING VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY VL53L0X_HISTOGRAMMODE_RETURN_ONLY VL53L0X_HISTOGRAMMODE_BOTH

Returns:

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_MODE_NOT_SUPPORTED This error occurs when 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_ONLY VL53L0X_HISTOGRAMMODE_RETURN_ONLY VL53L0X_HISTOGRAMMODE_BOTH

Returns:

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_MODE_NOT_SUPPORTED This error occurs when 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 are: VL53L0X_HISTOGRAMMODE_DISABLED VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY VL53L0X_HISTOGRAMMODE_RETURN_ONLY VL53L0X_HISTOGRAMMODE_BOTH

Returns:

VL53L0X_ERROR_NONE Success
"Other error code" See VL53L0X_Error

VL53L0X_API VL53L0X_Error VL53L0X_SetMeasurementTimingBudgetMicroSeconds (VL53L0X_DEV *Dev*, uint32_t *MeasurementTimingBudgetMicroSeconds*)

Set Ranging Timing Budget in microseconds.

Function Description

Defines the maximum time allowed by the user to the device to run a full ranging sequence for the current mode (ranging, histogram, ASL ...)

Note:

This function Access to the device

Parameters:

Dev	Device Handle
MeasurementTimingBudgetMicroSeconds	Max measurement time in microseconds. Valid values are: >= 17000 microsecs when wraparound enabled >= 12000 microsecs when wraparound disabled

Returns:

VL53L0X_ERROR_NONE Success
VL53L0X_ERROR_INVALID_PARAMS This error is returned if MeasurementTimingBudgetMicroSeconds out of range
"Other error code" See VL53L0X_Error

VL53L0X_API VL53L0X_Error VL53L0X_GetMeasurementTimingBudgetMicroSeconds (VL53L0X_DEV *Dev*, uint32_t * *pMeasurementTimingBudgetMicroSeconds*)

Get Ranging Timing Budget in microseconds.

Function Description

Returns the programmed the maximum time allowed by the user to the device to run a full ranging sequence for the current mode (ranging, histogram, ASL ...)

Note:

This function Access to the device

Parameters:

Dev	Device Handle
pMeasurementTimingBudgetMicroSeconds	Max measurement time in microseconds. Valid values are: >= 17000 microsecs when wraparound enabled >= 12000 microsecs 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 *VcselPeriodType*, 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 *VcselPeriodType*, 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 *SequenceStepEnabled*)

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 *Dev*, VL53L0X_SequenceStepId *SequenceStepId*, uint8_t * *pSequenceStepEnabled*)

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

Function Description

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

Note:

This function Accesses the device

Parameters:

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

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_GetSequenceStepEnables (VL53L0X_DEV *Dev*, VL53L0X_SchedulerSequenceSteps_t * *pSchedulerSequenceSteps*)

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 *Dev*, VL53L0X_SequenceStepId *SequenceStepId*, FixPoint1616_t *TimeOutMilliSecs*)**

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 *Dev*, VL53L0X_SequenceStepId *SequenceStepId*, FixPoint1616_t * *pTimeOutMilliSecs*)**

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 * *pNumberOfSequenceSteps*)**

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 of sequence steps.

Returns:

VL53L0X_ERROR_NONE Success
"Other error code" See VL53L0X_Error

VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepsInfo (VL53L0X_SequenceStepId *SequenceStepId*, char * *pSequenceStepsString*)

Gets the name of a given sequence step.

Function Description

This function retrieves the name of sequence steps corresponding to SequenceStepId.

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 *Dev*, uint32_t *InterMeasurementPeriodMilliSeconds*)

Program continuous mode Inter-Measurement period in milliseconds.

Function Description

When trying to set too short time return INVALID_PARAMS minimal value

Note:

This function Access to the device

Parameters:

Dev	Device Handle
InterMeasurementPeriodMilliSeconds	Inter-Measurement Period in ms.

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 minimal value

Note:

This function Access to the device

Parameters:

Dev	Device Handle
pInterMeasurementPeriodMilliSeconds	Pointer to programmed Inter-Measurement Period in milliseconds.

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 set to zero the Cross Talk value by using *VL53L0X_SetXTalkCompensationRateMegaCps()* .

Parameters:

Dev	Device Handle
XTalkCompensationEnable	Cross talk compensation to be set 0=disabled else = 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 set to zero the Cross Talk value by using *VL53L0X_SetXTalkCompensationRateMegaCps()* .

Parameters:

Dev	Device Handle
pXTalkCompensationEnable	Pointer to the Cross talk compensation state 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 counts per second (16.16 fix point) 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 * *pXTalkCompensationRateMegaCps*)

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 in Mega counts per second (16.16 fix point) see datasheet for details

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 * *pNumberOfLimitCheck*)

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<= LimitCheckId < 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 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_GetNumberOfLimitCheck()).
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<= LimitCheckId < 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 * *pLimitCheckCurrent*)

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

Function Description

This function get a the current value of the signal used for the limit check. To obtain the latest value you should run a ranging before. The 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<= LimitCheckId < 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, other = enabled

Returns:

VL53L0X_ERROR_NONE Success
"Other error code" See VL53L0X_Error

VL53L0X_API VL53L0X_Error VL53L0X_GetWrapAroundCheckEnable (VL53L0X_DEV *Dev*, uint8_t * *pWrapAroundCheckEnable*)

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 0=disabled or 1 = enabled

Returns:

VL53L0X_ERROR_NONE Success
"Other error code" See VL53L0X_Error

VL53L0X Measurement Functions

Functions used for the measurements.

Functions

- VL53L0X_API VL53L0X_Error VL53L0X_PerformSingleMeasurement (VL53L0X_DEV Dev)
  Single shot measurement.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformRefCalibration (VL53L0X_DEV Dev, uint8_t *pVhvSettings, uint8_t *pPhaseCal)
  Perform Reference Calibration.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformXTalkMeasurement (VL53L0X_DEV Dev, uint32_t TimeoutMs, FixPoint1616_t *pXtalkPerSpad, uint8_t *pAmbientTooHigh)
  Perform XTalk Measurement.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformXTalkCalibration (VL53L0X_DEV Dev, FixPoint1616_t XTalkCalDistance, FixPoint1616_t *pXTalkCompensationRateMegaCps)
  Perform XTalk Calibration.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformOffsetCalibration (VL53L0X_DEV Dev, FixPoint1616_t CalDistanceMilliMeter, int32_t *pOffsetMicroMeter)
  Perform Offset Calibration.
- VL53L0X_API VL53L0X_Error VL53L0X_StartMeasurement (VL53L0X_DEV Dev)
  Start device measurement.
- VL53L0X_API VL53L0X_Error VL53L0X_StopMeasurement (VL53L0X_DEV Dev)
  Stop device measurement.
- VL53L0X_API VL53L0X_Error VL53L0X_GetMeasurementDataReady (VL53L0X_DEV Dev, uint8_t *pMeasurementDataReady)
  Return Measurement Data Ready.
- VL53L0X_API VL53L0X_Error VL53L0X_WaitDeviceReadyForNewMeasurement (VL53L0X_DEV Dev, uint32_t MaxLoop)
  Wait for device ready for a new measurement command.
- VL53L0X_API VL53L0X_Error VL53L0X_GetMeasurementRefSignal (VL53L0X_DEV Dev, FixPoint1616_t *pMeasurementRefSignal)
  Retrieve the Reference Signal after a measurements.
- VL53L0X_API VL53L0X_Error VL53L0X_GetRangingMeasurementData (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData)
  Retrieve the measurements from device for a given setup.
- VL53L0X_API VL53L0X_Error VL53L0X_GetHistogramMeasurementData (VL53L0X_DEV Dev, VL53L0X_HistogramMeasurementData_t *pHistogramMeasurementData)

Retrieve the measurements from device for a given setup.

- VL53L0X_API VL53L0X_Error VL53L0X_PerformSingleRangingMeasurement (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData)
  Performs a single ranging measurement and retrieve the ranging measurement data.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformSingleHistogramMeasurement (VL53L0X_DEV Dev, VL53L0X_HistogramMeasurementData_t *pHistogramMeasurementData)
  Performs a single histogram measurement and retrieve the histogram measurement data Is equivalent to VL53L0X_PerformSingleMeasurement + VL53L0X_GetHistogramMeasurementData.
- 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.
- VL53L0X_API VL53L0X_Error VL53L0X_GetNumberOfROIZones (VL53L0X_DEV Dev, uint8_t *pNumberOfROIZones)
  Get the number of ROI Zones managed by the Device.
- VL53L0X_API VL53L0X_Error VL53L0X_GetMaxNumberOfROIZones (VL53L0X_DEV Dev, uint8_t *pMaxNumberOfROIZones)
  Get the Maximum number of ROI Zones managed by the Device.

Detailed Description

Functions used for the measurements.

Function Documentation

VL53L0X_API VL53L0X_Error VL53L0X_PerformSingleMeasurement (VL53L0X_DEV *Dev*)

Single shot measurement.

Function Description

Perform simple measurement sequence (Start measure, Wait measure to end, and returns when measurement is done). Once function returns, user can get valid data by calling VL53L0X_GetRangingMeasurement or VL53L0X_GetHistogramMeasurement depending on defined measurement mode User should Clear the interrupt in case this are enabled by using the function VL53L0X_ClearInterruptMask().

Warning:

This function is a blocking function

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

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 * *pXtalkPerSpad*, uint8_t * *pAmbientTooHigh*)

Perform XTalk Measurement.

Measures the current cross talk from glass in front of the sensor. This functions 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.

Warning:

This function is a blocking function
This function is not supported when the final range vcsel clock period is set 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 fixpoint 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 *XTalkCalDistance*, FixPoint1616_t * *pXTalkCompensationRateMegaCps*)

Perform XTalk Calibration.

Perform a XTalk calibration of the Device. This function will launch a ranging measurement, if interrupts are enabled an interrupt will be done. This function will clear the interrupt generated automatically. This function will program a new value for the XTalk compensation and it will enable the cross talk before exit. 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 change the device mode to VL53L0X_DEVICEMODE_SINGLE_RANGING

Parameters:

Dev	Device Handle
XTalkCalDistance	XTalkCalDistance value used for the XTalk computation.
pXTalkCompensationRateMegaCps	Pointer to new XTalkCompensation value.

Returns:

VL53L0X_ERROR_NONE Success
"Other error code" See VL53L0X_Error

VL53L0X_API VL53L0X_Error VL53L0X_PerformOffsetCalibration (VL53L0X_DEV *Dev*, FixPoint1616_t *CalDistanceMilliMeter*, int32_t * *pOffsetMicroMeter*)

Perform Offset Calibration.

Perform a Offset calibration of the Device. This function will launch a ranging measurement, if interrupts are enabled an interrupt will be done. This function will clear the interrupt generated automatically. This function will program a new 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 * *pMeasurementDataReady*)

Return Measurement Data Ready.

Function Description

This function indicate that a measurement data is ready. This function check if interrupt mode is used then check is done accordingly. If perform function clear the interrupt, this function will not work, like in case of *VL53L0X_PerformSingleRangingMeasurement()* . The previous function is blocking function, VL53L0X_GetMeasurementDataReady is used for non-blocking capture.

Note:

This function Access to the device

Parameters:

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

Returns:

VL53L0X_ERROR_NONE Success
"Other error code" See VL53L0X_Error

VL53L0X_API VL53L0X_Error VL53L0X_WaitDeviceReadyForNewMeasurement (VL53L0X_DEV *Dev*, uint32_t *MaxLoop*)

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_API VL53L0X_Error VL53L0X_GetMeasurementRefSignal (VL53L0X_DEV *Dev*, FixPoint1616_t * *pMeasurementRefSignal*)

Retrieve the Reference Signal after a measurements.

Function Description

Get Reference Signal from last successful Ranging measurement This function 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 *Dev*, VL53L0X_RangingMeasurementData_t * *pRangingMeasurementData*)

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()* before get data. PAL will fill a NumberOfROIZones times the corresponding data structure used in the measurement function.

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 *Dev*, VL53L0X_HistogramMeasurementData_t * *pHistogramMeasurementData*)

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()* before get data. PAL will fill a NumberOfROIZones 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 *Dev*, VL53L0X_RangingMeasurementData_t * *pRangingMeasurementData*)

Performs a single ranging measurement and retrieve the ranging measurement data.

Function Description

This function will change the device mode to VL53L0X_DEVICEMODE_SINGLE_RANGING with *VL53L0X_SetDeviceMode()* , It performs measurement with *VL53L0X_PerformSingleMeasurement()* It get data from last successful Ranging measurement with *VL53L0X_GetRangingMeasurementData* . Finally it clear the interrupt with *VL53L0X_ClearInterruptMask()* .

Note:

This function Access to the device
This function change the device mode to VL53L0X_DEVICEMODE_SINGLE_RANGING

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 *Dev*, VL53L0X_HistogramMeasurementData_t * *pHistogramMeasurementData*)

Performs a single histogram measurement and retrieve the histogram measurement data Is equivalent to VL53L0X_PerformSingleMeasurement + VL53L0X_GetHistogramMeasurementData.

Function Description

Get data from last successful Ranging measurement. This function will clear the interrupt in case of these are enabled.

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 API manage 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 after a 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 * *pMaxNumberOfROIZones*)

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 Interrupt Functions

Functions used for interrupt managements.

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.
- 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.
- 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.
- 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.
- VL53L0X_API VL53L0X_Error VL53L0X_GetStopCompletedStatus (VL53L0X_DEV Dev, uint32_t *pStopStatus)
  Return device stop completion status.
- VL53L0X_API VL53L0X_Error VL53L0X_ClearInterruptMask (VL53L0X_DEV Dev, uint32_t InterruptMask)
  Clear given system interrupt condition.
- VL53L0X_API VL53L0X_Error VL53L0X_GetInterruptMaskStatus (VL53L0X_DEV Dev, uint32_t *pInterruptMaskStatus)
  Return device interrupt status.
- VL53L0X_API VL53L0X_Error VL53L0X_EnableInterruptMask (VL53L0X_DEV Dev, uint32_t InterruptMask)
  Configure ranging interrupt reported to system.

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_OUT,
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 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_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, ...) for a given device.

Function Description

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

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

Function Description

Get high and low Interrupt thresholds for a given mode (ranging, ALS, ...) 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 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 completiob 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 * *pInterruptMaskStatus*)

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
pInterruptMaskStatus	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

VL53L0X SPAD Functions

Functions used for SPAD managements.

Functions

- VL53L0X_API VL53L0X_Error VL53L0X_SetSpadAmbientDamperThreshold (VL53L0X_DEV Dev, uint16_t SpadAmbientDamperThreshold)
  Set the SPAD Ambient Damper Threshold value.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSpadAmbientDamperThreshold (VL53L0X_DEV Dev, uint16_t *pSpadAmbientDamperThreshold)
  Get the current SPAD Ambient Damper Threshold value.
- VL53L0X_API VL53L0X_Error VL53L0X_SetSpadAmbientDamperFactor (VL53L0X_DEV Dev, uint16_t SpadAmbientDamperFactor)
  Set the SPAD Ambient Damper Factor value.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSpadAmbientDamperFactor (VL53L0X_DEV Dev, uint16_t *pSpadAmbientDamperFactor)
  Get the current SPAD Ambient Damper Factor value.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformRefSpadManagement (VL53L0X_DEV Dev, uint32_t *refSpadCount, uint8_t *isApertureSpads)
  Performs Reference Spad Management.
- VL53L0X_API VL53L0X_Error VL53L0X_SetReferenceSpads (VL53L0X_DEV Dev, uint32_t refSpadCount, uint8_t isApertureSpads)
  Applies Reference SPAD configuration.
- VL53L0X_API VL53L0X_Error VL53L0X_GetReferenceSpads (VL53L0X_DEV Dev, uint32_t *refSpadCount, uint8_t *isApertureSpads)
  Retrieves SPAD configuration.

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
SpadAmbientDam	SPAD Ambient Damper Threshold value

perThreshold	

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

Note:

This function Access to the device

Parameters:

Dev	Device Handle
pSpadAmbientDamperThreshold	Pointer to programmed SPAD Ambient Damper Threshold value

Returns:

VL53L0X_ERROR_NONE Success
"Other error code" See VL53L0X_Error

VL53L0X_API VL53L0X_Error VL53L0X_SetSpadAmbientDamperFactor (VL53L0X_DEV *Dev*, uint16_t *SpadAmbientDamperFactor*)

Set the SPAD Ambient Damper Factor value.

Function Description

This function set the SPAD Ambient Damper Factor value

Note:

This function Access to the device

Parameters:

Dev	Device Handle
SpadAmbientDamperFactor	SPAD Ambient Damper Factor value

Returns:

VL53L0X_ERROR_NONE Success
"Other error code" See VL53L0X_Error

VL53L0X_API VL53L0X_Error VL53L0X_GetSpadAmbientDamperFactor (VL53L0X_DEV *Dev*, uint16_t * *pSpadAmbientDamperFactor*)

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 Ambient Damper Factor 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 * *isApertureSpads*)

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 in the reference spad configuration.
"Other error code" See VL53L0X_Error

VL53L0X Defines

VL53L0X Defines.

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.* General Macro Defines

General Macro Defines. Data Structures

- struct VL53L0X_Version_t
- *Defines the parameters of the Get Version Functions.* struct VL53L0X_DeviceInfo_t
- *Defines the parameters of the Get Device Info Functions.* struct VL53L0X_DMaxLUT_t
- *Structure defining data pair that makes up the DMAX Lookup table.* struct VL53L0X_DeviceParameters_t
- *Defines all parameters for the device.* struct VL53L0X_RangingMeasurementData_t
- struct VL53L0X_HistogramMeasurementData_t
- struct VL53L0X_SpadData_t
- *Spad Configuration Data.* 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.* struct VL53L0X_RangeData_t
- *Range measurement data.* struct VL53L0X_HistogramData_t

Histogram measurement data. Macros

- #define VL53L0X10_SPECIFICATION_VER_MAJOR 1
  PAL SPECIFICATION major version.
- #define VL53L0X10_SPECIFICATION_VER_MINOR 2
  PAL SPECIFICATION minor version.
- #define VL53L0X10_SPECIFICATION_VER_SUB 7
  PAL SPECIFICATION sub version.
- #define VL53L0X10_SPECIFICATION_VER_REVISION 1440
  PAL SPECIFICATION sub version.
- #define VL53L0X10_IMPLEMENTATION_VER_MAJOR 1
  VL53L0X PAL IMPLEMENTATION major version.
- #define VL53L0X10_IMPLEMENTATION_VER_MINOR 0
  VL53L0X PAL IMPLEMENTATION minor version.
- #define VL53L0X10_IMPLEMENTATION_VER_SUB 9
  VL53L0X PAL IMPLEMENTATION sub version.
- #define VL53L0X10_IMPLEMENTATION_VER_REVISION 3673
  VL53L0X PAL IMPLEMENTATION sub version.
- #define VL53L0X_SPECIFICATION_VER_MAJOR 1
  PAL SPECIFICATION major version.
- #define VL53L0X_SPECIFICATION_VER_MINOR 2
  PAL SPECIFICATION minor version.
- #define VL53L0X_SPECIFICATION_VER_SUB 7
  PAL SPECIFICATION sub version.
- #define VL53L0X_SPECIFICATION_VER_REVISION 1440
  PAL SPECIFICATION sub version.
- #define VL53L0X_IMPLEMENTATION_VER_MAJOR 1
  VL53L0X PAL IMPLEMENTATION major version.
- #define VL53L0X_IMPLEMENTATION_VER_MINOR 0
  VL53L0X PAL IMPLEMENTATION minor version.
- #define VL53L0X_IMPLEMENTATION_VER_SUB 4
  VL53L0X PAL IMPLEMENTATION sub version.
- #define VL53L0X_IMPLEMENTATION_VER_REVISION 4960
  VL53L0X PAL IMPLEMENTATION sub version.
- #define VL53L0X_DEFAULT_MAX_LOOP 2000
- #define VL53L0X_MAX_STRING_LENGTH 32
- #define VL53L0X_DMAX_LUT_SIZE 7
- #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 4

VL53L0X PAL IMPLEMENTATION sub version.
Definition at line 83 of file vl53l0x_def.h.

#define VL53L0X_IMPLEMENTATION_VER_REVISION 4960

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_DMAX_LUT_SIZE 7

Defines the number of items in the DMAX lookup table
Definition at line 235 of file vl53l0x_def.h.

#define VL53L0X_HISTOGRAM_BUFFER_SIZE 24

Definition at line 369 of file vl53l0x_def.h.

#define VL53L0X_REF_SPAD_BUFFER_SIZE 6

Definition at line 393 of file vl53l0x_def.h.

Error and Warning code returned by API

The following DEFINE are used to identify the PAL ERROR.

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) - 20)
- #define VL53L0X_ERROR_INVALID_COMMAND ((VL53L0X_Error) - 30)
- #define VL53L0X_ERROR_DIVISION_BY_ZERO ((VL53L0X_Error) - 40)
- #define VL53L0X_ERROR_REF_SPAD_INIT ((VL53L0X_Error) - 50)
- #define VL53L0X_ERROR_NOT_IMPLEMENTED ((VL53L0X_Error) - 99)

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 140 of file vl53l0x_def.h.

#define VL53L0X_ERROR_UNDEFINED ((VL53L0X_Error) - 3)

Unqualified error

Definition at line 143 of file vl53l0x_def.h.

#define VL53L0X_ERROR_INVALID_PARAMS ((VL53L0X_Error) - 4)

Parameter passed is invalid or out of range

Definition at line 145 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 147 of file vl53l0x_def.h.

#define VL53L0X_ERROR_RANGE_ERROR ((VL53L0X_Error) - 6)

Device report a ranging error interrupt status

Definition at line 149 of file vl53l0x_def.h.

#define VL53L0X_ERROR_TIME_OUT ((VL53L0X_Error) - 7)

Aborted due to time out

Definition at line 151 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 153 of file vl53l0x_def.h.

#define VL53L0X_ERROR_BUFFER_TOO_SMALL ((VL53L0X_Error) - 9)

...

Definition at line 155 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 157 of file vl53l0x_def.h.

#define VL53L0X_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED ((VL53L0X_Error) - 11)

unsupported GPIO functionality

Definition at line 159 of file vl53l0x_def.h.

#define VL53L0X_ERROR_INTERRUPT_NOT_CLEARED ((VL53L0X_Error) - 12)

Error during interrupt clear

Definition at line 161 of file vl53l0x_def.h.

#define VL53L0X_ERROR_CONTROL_INTERFACE ((VL53L0X_Error) - 20)

error reported from IO functions

Definition at line 163 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 165 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 169 of file vl53l0x_def.h.

#define VL53L0X_ERROR_REF_SPAD_INIT ((VL53L0X_Error) - 50)

Error during reference SPAD initialization

Definition at line 171 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 173 of file vl53l0x_def.h.

Typedef Documentation

typedef int8_t VL53L0X_Error

Definition at line 131 of file vl53l0x_def.h.

Defines Device modes

Defines all possible modes for the device.

Macros

- #define VL53L0X_DEVICEMODE_SINGLE_RANGING ((VL53L0X_DeviceModes) 0)
- #define VL53L0X_DEVICEMODE_CONTINUOUS_RANGING ((VL53L0X_DeviceModes) 1)
- #define VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM ((VL53L0X_DeviceModes) 2)
- #define VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING ((VL53L0X_DeviceModes) 3)
- #define VL53L0X_DEVICEMODE_SINGLE_ALS ((VL53L0X_DeviceModes) 10)
- #define VL53L0X_DEVICEMODE_GPIO_DRIVE ((VL53L0X_DeviceModes) 20)
- #define VL53L0X_DEVICEMODE_GPIO_OSC ((VL53L0X_DeviceModes) 21)

Typedefs

- typedef uint8_t VL53L0X_DeviceModes

Detailed Description

Defines all possible modes for the device.

Macro Definition Documentation

#define VL53L0X_DEVICEMODE_SINGLE_RANGING ((VL53L0X_DeviceModes) 0)

Definition at line 186 of file vl53l0x_def.h.

#define VL53L0X_DEVICEMODE_CONTINUOUS_RANGING ((VL53L0X_DeviceModes) 1)

Definition at line 187 of file vl53l0x_def.h.

#define VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM ((VL53L0X_DeviceModes) 2)

Definition at line 188 of file vl53l0x_def.h.

#define VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING ((VL53L0X_DeviceModes) 3)

Definition at line 189 of file vl53l0x_def.h.

#define VL53L0X_DEVICEMODE_SINGLE_ALS ((VL53L0X_DeviceModes) 10)

Definition at line 190 of file vl53l0x_def.h.

#define VL53L0X_DEVICEMODE_GPIO_DRIVE ((VL53L0X_DeviceModes) 20)

Definition at line 191 of file vl53l0x_def.h.

#define VL53L0X_DEVICEMODE_GPIO_OSC ((VL53L0X_DeviceModes) 21)

Definition at line 192 of file vl53l0x_def.h.

Typedef Documentation

typedef uint8_t VL53L0X_DeviceModes

Definition at line 184 of file vl53l0x_def.h.

Defines Histogram modes

Defines all possible Histogram modes for the device.

Macros

- #define VL53L0X_HISTOGRAMMODE_DISABLED ((VL53L0X_HistogramModes) 0)
- #define VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY ((VL53L0X_HistogramModes) 1)
- #define VL53L0X_HISTOGRAMMODE_RETURN_ONLY ((VL53L0X_HistogramModes) 2)

- #define VL53L0X_HISTOGRAMMODE_BOTH ((VL53L0X_HistogramModes) 3)

Typedefs

- typedef uint8_t VL53L0X_HistogramModes

Detailed Description

Defines all possible Histogram modes for the device.

Macro Definition Documentation

#define VL53L0X_HISTOGRAMMODE_DISABLED ((VL53L0X_HistogramModes) 0)

Histogram Disabled

Definition at line 204 of file vl53l0x_def.h.

#define VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY ((VL53L0X_HistogramModes) 1)

Histogram Reference array only

Definition at line 206 of file vl53l0x_def.h.

#define VL53L0X_HISTOGRAMMODE_RETURN_ONLY ((VL53L0X_HistogramModes) 2)

Histogram Return array only

Definition at line 208 of file vl53l0x_def.h.

#define VL53L0X_HISTOGRAMMODE_BOTH ((VL53L0X_HistogramModes) 3)

Histogram both Reference and Return Arrays

Definition at line 210 of file vl53l0x_def.h.

Typedef Documentation

typedef uint8_t VL53L0X_HistogramModes

Definition at line 202 of file vl53l0x_def.h.

List of available Power Modes

List of available Power Modes.

Macros

- #define VL53L0X_POWERMODE_STANDBY_LEVEL1 ((VL53L0X_PowerModes) 0)
- #define VL53L0X_POWERMODE_STANDBY_LEVEL2 ((VL53L0X_PowerModes) 1)
- #define VL53L0X_POWERMODE_IDLE_LEVEL1 ((VL53L0X_PowerModes) 2)
- #define VL53L0X_POWERMODE_IDLE_LEVEL2 ((VL53L0X_PowerModes) 3)

Typedefs

- typedef uint8_t VL53L0X_PowerModes

Detailed Description

List of available Power Modes.

Macro Definition Documentation

#define VL53L0X_POWERMODE_STANDBY_LEVEL1 ((VL53L0X_PowerModes) 0)

Standby level 1

Definition at line 223 of file vl53l0x_def.h.

#define VL53L0X_POWERMODE_STANDBY_LEVEL2 ((VL53L0X_PowerModes) 1)

Standby level 2

Definition at line 225 of file vl53l0x_def.h.

#define VL53L0X_POWERMODE_IDLE_LEVEL1 ((VL53L0X_PowerModes) 2)

Idle level 1

Definition at line 227 of file vl53l0x_def.h.

#define VL53L0X_POWERMODE_IDLE_LEVEL2 ((VL53L0X_PowerModes) 3)

Idle level 2

Definition at line 229 of file vl53l0x_def.h.

Typedef Documentation

typedef uint8_t VL53L0X_PowerModes

Definition at line 221 of file vl53l0x_def.h.

Defines the current status of the device

Defines the current status of the device.

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 299 of file vl53l0x_def.h.

#define VL53L0X_STATE_WAIT_STATICINIT ((VL53L0X_State) 1)

Device is initialized and wait for static initialization

Definition at line 301 of file vl53l0x_def.h.

#define VL53L0X_STATE_STANDBY ((VL53L0X_State) 2)

Device is in Low power Standby mode

Definition at line 303 of file vl53l0x_def.h.

#define VL53L0X_STATE_IDLE ((VL53L0X_State) 3)

Device has been initialized and ready to do measurements

Definition at line 305 of file vl53l0x_def.h.

#define VL53L0X_STATE_RUNNING ((VL53L0X_State) 4)

Device is performing measurement

Definition at line 307 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 309 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 311 of file vl53l0x_def.h.

Typedef Documentation

typedef uint8_t VL53L0X_State

Definition at line 297 of file vl53l0x_def.h.

Defines the Polarity

of the Interrupt Defines the Polarity of the Interrupt

Macros

- #define VL53L0X_INTERRUPTPOLARITY_LOW ((VL53L0X_InterruptPolarity)0)
- #define VL53L0X_INTERRUPTPOLARITY_HIGH ((VL53L0X_InterruptPolarity) 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_InterruptPolarity) 0)

Set active low polarity best setup for falling edge.

Definition at line 521 of file vl53l0x_def.h.

#define VL53L0X_INTERRUPTPOLARITY_HIGH ((VL53L0X_InterruptPolarity) 1)

Set active high polarity best setup for rising edge.

Definition at line 523 of file vl53l0x_def.h.

Typedef Documentation

typedef uint8_t VL53L0X_InterruptPolarity

Definition at line 519 of file vl53l0x_def.h.

Vcsel Period Defines

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

Macros

- #define VL53L0X_VCSEL_PERIOD_PRE_RANGE ((VL53L0X_VcselPeriod) 0)
- #define VL53L0X_VCSEL_PERIOD_FINAL_RANGE ((VL53L0X_VcselPeriod) 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 535 of file vl53l0x_def.h.

#define VL53L0X_VCSEL_PERIOD_FINAL_RANGE ((VL53L0X_VcselPeriod) 1)

Identifies the final range vcsel period.

Definition at line 537 of file vl53l0x_def.h.

Typedef Documentation

typedef uint8_t VL53L0X_VcselPeriod

Definition at line 533 of file vl53l0x_def.h.

Defines the steps

carried out by the scheduler during a range measurement.

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.

Defines the Polarity

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

Macros

- #define VL53L0X_SEQUENCESTEP_TCC ((VL53L0X_VcselPeriod) 0)
- #define VL53L0X_SEQUENCESTEP_DSS ((VL53L0X_VcselPeriod) 1)
- #define VL53L0X_SEQUENCESTEP_MSRC ((VL53L0X_VcselPeriod) 2)
- #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 565 of file vl53l0x_def.h.

#define VL53L0X_SEQUENCESTEP_DSS ((VL53L0X_VcselPeriod) 1)

Dynamic Spad Selection function Identifier.

Definition at line 567 of file vl53l0x_def.h.

#define VL53L0X_SEQUENCESTEP_MSRC ((VL53L0X_VcselPeriod) 2)

Minimum Signal Rate Check function Identifier.

Definition at line 569 of file vl53l0x_def.h.

#define VL53L0X_SEQUENCESTEP_PRE_RANGE ((VL53L0X_VcselPeriod) 3)

Pre-Range check Identifier.

Definition at line 571 of file vl53l0x_def.h.

#define VL53L0X_SEQUENCESTEP_FINAL_RANGE ((VL53L0X_VcselPeriod) 4)

Final Range Check Identifier.

Definition at line 573 of file vl53l0x_def.h.

#define VL53L0X_SEQUENCESTEP_NUMBER_OF_CHECKS 5

Number of Sequence Step Managed by the API.

Definition at line 576 of file vl53l0x_def.h.

Typedef Documentation

typedef uint8_t VL53L0X_SequenceStepId

Definition at line 563 of file vl53l0x_def.h.

General Macro Defines

General Macro Defines.

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) (FixPoint1616_t)(Value<<9)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT88(Value) (uint16_t)((Value>>8)&0xFFFF)
- #define VL53L0X_FIXPOINT88TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<8)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT412(Value) (uint16_t)((Value>>4)&0xFFFF)
- #define VL53L0X_FIXPOINT412TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<4)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT313(Value) (uint16_t)((Value>>3)&0xFFFF)
- #define VL53L0X_FIXPOINT313TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<3)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT08(Value) (uint8_t)((Value>>8)&0x00FF)
- #define VL53L0X_FIXPOINT08TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<8)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT53(Value) (uint8_t)((Value>>13)&0x00FF)
- #define VL53L0X_FIXPOINT53TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<13)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT102(Value) (uint16_t)((Value>>14)&0x0FFF)
- #define VL53L0X_FIXPOINT102TOFIXPOINT1616(Value) (FixPoint1616_t)(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 589 of file vl53l0x_def.h.

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

Definition at line 592 of file vl53l0x_def.h.

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

Definition at line 596 of file vl53l0x_def.h.

#define VL53L0X_GETARRAYPARAMETERFIELD(Dev, field, index, variable) (variable = (PALDevDataGet(Dev, CurrentParameters)).field[index])

Definition at line 599 of file vl53l0x_def.h.

#define VL53L0X_SETDEVICESPECIFICPARAMETER(Dev, field, value) PALDevDataSet(Dev, DeviceSpecificParameters.field, value)

Definition at line 602 of file vl53l0x_def.h.

#define VL53L0X_GETDEVICESPECIFICPARAMETER(Dev, field) PALDevDataGet(Dev, DeviceSpecificParameters).field

Definition at line 605 of file vl53l0x_def.h.

#define VL53L0X_FIXPOINT1616TOFIXPOINT97(Value) (uint16_t)((Value>>9)&0xFFFF)

Definition at line 609 of file vl53l0x_def.h.

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

Definition at line 611 of file vl53l0x_def.h.

#define VL53L0X_FIXPOINT1616TOFIXPOINT88(Value) (uint16_t)((Value>>8)&0xFFFF)

Definition at line 614 of file vl53l0x_def.h.

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

Definition at line 616 of file vl53l0x_def.h.

#define VL53L0X_FIXPOINT1616TOFIXPOINT412(Value) (uint16_t)((Value>>4)&0xFFFF)

Definition at line 619 of file vl53l0x_def.h.

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

Definition at line 621 of file vl53l0x_def.h.

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

Definition at line 624 of file vl53l0x_def.h.

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

Definition at line 626 of file vl53l0x_def.h.

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

Definition at line 629 of file vl53l0x_def.h.

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

Definition at line 631 of file vl53l0x_def.h.

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

Definition at line 634 of file vl53l0x_def.h.

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

Definition at line 636 of file vl53l0x_def.h.

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

Definition at line 639 of file vl53l0x_def.h.

#define VL53L0X_FIXPOINT102TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<12)

Definition at line 641 of file vl53l0x_def.h.

#define VL53L0X_MAKEUINT16(lsb, msb)

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

Definition at line 644 of file vl53l0x_def.h.

VL53L0X cut1.1 Device Specific Defines

Device specific defines.

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

Device Error

Device Error code.

Macros

- #define VL53L0X_DEVICEERROR_NONE ((VL53L0X_DeviceError) 0)
- #define VL53L0X_DEVICEERROR_VCSELCONTINUITYTESTFAILURE ((VL53L0X_DeviceError) 1)
- #define VL53L0X_DEVICEERROR_VCSELWATCHDOGTESTFAILURE ((VL53L0X_DeviceError) 2)
- #define VL53L0X_DEVICEERROR_NOVHVVALUEFOUND ((VL53L0X_DeviceError) 3)
- #define VL53L0X_DEVICEERROR_MSRCNOTARGET ((VL53L0X_DeviceError) 4)
- #define VL53L0X_DEVICEERROR_SNRCHECK ((VL53L0X_DeviceError) 5)
- #define VL53L0X_DEVICEERROR_RANGEPHASECHECK ((VL53L0X_DeviceError) 6)
- #define VL53L0X_DEVICEERROR_SIGMATHRESHOLDCHECK ((VL53L0X_DeviceError) 7)
- #define VL53L0X_DEVICEERROR_TCC ((VL53L0X_DeviceError) 8)
- #define VL53L0X_DEVICEERROR_PHASECONSISTENCY ((VL53L0X_DeviceError) 9)
- #define VL53L0X_DEVICEERROR_MINCLIP ((VL53L0X_DeviceError) 10)
- #define VL53L0X_DEVICEERROR_RANGECOMPLETE ((VL53L0X_DeviceError) 11)
- #define VL53L0X_DEVICEERROR_ALGOUNDERFLOW ((VL53L0X_DeviceError) 12)
- #define VL53L0X_DEVICEERROR_ALGOOVERFLOW ((VL53L0X_DeviceError) 13)
- #define VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD ((VL53L0X_DeviceError) 14)

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 ((VL53L0X_DeviceError) 1)

Definition at line 58 of file vl53l0x_device.h.

#define VL53L0X_DEVICEERROR_VCSELWATCHDOGTESTFAILURE ((VL53L0X_DeviceError) 2)

Definition at line 59 of file vl53l0x_device.h.

#define VL53L0X_DEVICEERROR_NOVHVVALUEFOUND ((VL53L0X_DeviceError) 3)

Definition at line 60 of file vl53l0x_device.h.

#define VL53L0X_DEVICEERROR_MSRCNOTARGET ((VL53L0X_DeviceError) 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_DeviceError) 11)

Definition at line 68 of file vl53l0x_device.h.

#define VL53L0X_DEVICEERROR_ALGOUNDERFLOW ((VL53L0X_DeviceError) 12)

Definition at line 69 of file vl53l0x_device.h.

#define VL53L0X_DEVICEERROR_ALGOOVERFLOW ((VL53L0X_DeviceError) 13)

Definition at line 70 of file vl53l0x_device.h.

#define VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD ((VL53L0X_DeviceError) 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.

Check Enable list

Check Enable code.

Macros

- #define VL53L0X_CHECKENABLE_SIGMA_FINAL_RANGE 0
- #define VL53L0X_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE 1
- #define VL53L0X_CHECKENABLE_SIGNAL_REF_CLIP 2
- #define VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD 3
- #define VL53L0X_CHECKENABLE_SIGNAL_RATE_MSRC 4
- #define VL53L0X_CHECKENABLE_SIGNAL_RATE_PRE_RANGE 5
- #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.

Gpio Functionality

Defines the different functionalities for the device GPIO(s)

Macros

- #define VL53L0X_GPIOFUNCTIONALITY_OFF ((VL53L0X_GpioFunctionality) 0)
- #define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_LOW ((VL53L0X_GpioFunctionality) 1)
- #define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HIGH ((VL53L0X_GpioFunctionality) 2)
- #define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OUT ((VL53L0X_GpioFunctionality) 3)
- #define VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY ((VL53L0X_GpioFunctionality) 4)

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_GpioFunctionality) 0)

NO Interrupt

Definition at line 102 of file vl53l0x_device.h.

#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_LOW ((VL53L0X_GpioFunctionality) 1)

Level Low (value < thresh_low)

Definition at line 104 of file vl53l0x_device.h.

#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HIGH ((VL53L0X_GpioFunctionality) 2)

Level High (value > thresh_high)

Definition at line 106 of file vl53l0x_device.h.

#define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OUT ((VL53L0X_GpioFunctionality) 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.

Define Registers

List of all the defined registers.

Macros

- #define VL53L0X_REG_SYSRANGE_START 0x000
- #define VL53L0X_REG_SYSRANGE_MODE_MASK 0x0F
  mask existing bit in VL53L0X_REG_SYSRANGE_START
- #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
- #define VL53L0X_REG_SYSRANGE_MODE_SINGLESHOT 0x00
  bit 1 write 0 in VL53L0X_REG_SYSRANGE_START set single shot mode
- #define VL53L0X_REG_SYSRANGE_MODE_BACKTOBACK 0x02
  bit 1 write 1 in VL53L0X_REG_SYSRANGE_START set back-to-back operation mode
- #define VL53L0X_REG_SYSRANGE_MODE_TIMED 0x04
  bit 2 write 1 in VL53L0X_REG_SYSRANGE_START set timed operation mode
- #define VL53L0X_REG_SYSRANGE_MODE_HISTOGRAM 0x08
  bit 3 write 1 in VL53L0X_REG_SYSRANGE_START set histogram operation 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 0x0004

- #define VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO 0x000A
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED 0x00
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW 0x01
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH 0x02
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW 0x03
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_READY 0x04
- #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_RTN 0x00BC
- #define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_RTN 0x00C0
- #define VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_REF 0x00D0
- #define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_REF 0x00D4
- #define VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF 0x00B6
- #define VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET_MM 0x0028
- #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 0x0056
- #define VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HIGH 0x0057
- #define VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LIMIT 0x0064
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR 0X0067
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_LOW 0x0047
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_HIGH 0x0048
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RATE_RTN_LIMIT 0x0044
- #define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_HI 0X0061
- #define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_LO 0X0062
- #define VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD 0x0050
- #define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_HI 0x0051
- #define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_LO 0x0052
- #define VL53L0X_REG_SYSTEM_HISTOGRAM_BIN 0x0081
- #define VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SELECT 0x0033
- #define VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL 0x0055
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_VCSEL_PERIOD 0x0070
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_HI 0x0071
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_LO 0x0072
- #define VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RATE_MCPS 0x0020
- #define VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP 0x0046
- #define VL53L0X_REG_SOFT_RESET_GO2_SOFT_RESET_N 0x00bf
- #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_0 0x0B0
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1 0x0B1
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2 0x0B2
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3 0x0B3
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4 0x0B4
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5 0x0B5
- #define VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELECT 0xB6
- #define VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF_SPAD 0x4E /* 0x14E */
- #define VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET 0x4F /* 0x14F */
- #define VL53L0X_REG_POWER_MANAGEMENT_GO1_POWER_FORCE 0x80
- #define VL53L0X_SPEED_OF_LIGHT_IN_AIR 2997

- #define VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA__EXTSUP_HV 0x0089
- #define VL53L0X_REG_ALGO_PHASECAL_LIM 0x0030 /* 0x130 */
- #define VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT 0x0030

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 129 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSRANGE_MODE_SINGLESHOT 0x00

bit 1 write 0 in VL53L0X_REG_SYSRANGE_START set single shot mode

Definition at line 131 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 135 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 139 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 143 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_THRESH_HIGH 0x000C

Definition at line 146 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_THRESH_LOW 0x000E

Definition at line 147 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_SEQUENCE_CONFIG 0x0001

Definition at line 150 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_RANGE_CONFIG 0x0009

Definition at line 151 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_INTERMEASUREMENT_PERIOD 0x0004

Definition at line 152 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO 0x000A

Definition at line 155 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED 0x00

Definition at line 156 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW 0x01

Definition at line 157 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH 0x02

Definition at line 158 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW 0x03

Definition at line 159 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_READY 0x04

Definition at line 160 of file vl53l0x_device.h.

#define VL53L0X_REG_GPIO_HV_MUX_ACTIVE_HIGH 0x0084

Definition at line 162 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_INTERRUPT_CLEAR 0x000B

Definition at line 165 of file vl53l0x_device.h.

#define VL53L0X_REG_RESULT_INTERRUPT_STATUS 0x0013

Definition at line 168 of file vl53l0x_device.h.

#define VL53L0X_REG_RESULT_RANGE_STATUS 0x0014

Definition at line 169 of file vl53l0x_device.h.

#define VL53L0X_REG_RESULT_CORE_PAGE 1

Definition at line 171 of file vl53l0x_device.h.

#define VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_RTN 0x00BC

Definition at line 172 of file vl53l0x_device.h.

#define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_RTN 0x00C0

Definition at line 173 of file vl53l0x_device.h.

#define VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_REF 0x00D0

Definition at line 174 of file vl53l0x_device.h.

#define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_REF 0x00D4

Definition at line 175 of file vl53l0x_device.h.

#define VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF 0x00B6

Definition at line 176 of file vl53l0x_device.h.

#define VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET_MM 0x0028

Definition at line 180 of file vl53l0x_device.h.

#define VL53L0X_REG_I2C_SLAVE_DEVICE_ADDRESS 0x008a

Definition at line 182 of file vl53l0x_device.h.

#define VL53L0X_REG_MSRC_CONFIG_CONTROL 0x0060

Definition at line 185 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_CONFIG_MIN_SNR 0X0027

Definition at line 187 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_LOW 0x0056

Definition at line 188 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HIGH 0x0057

Definition at line 189 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LIMIT 0x0064

Definition at line 190 of file vl53l0x_device.h.

#define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR 0X0067

Definition at line 192 of file vl53l0x_device.h.

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_LOW 0x0047

Definition at line 193 of file vl53l0x_device.h.

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_HIGH 0x0048

Definition at line 194 of file vl53l0x_device.h.

#define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RATE_RTN_LIMIT 0x0044

Definition at line 195 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_HI 0X0061

Definition at line 198 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_LO 0X0062

Definition at line 199 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD 0x0050

Definition at line 202 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_HI 0x0051

Definition at line 203 of file vl53l0x_device.h.

#define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_LO 0x0052

Definition at line 204 of file vl53l0x_device.h.

#define VL53L0X_REG_SYSTEM_HISTOGRAM_BIN 0x0081

Definition at line 206 of file vl53l0x_device.h.

#define VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SELECT 0x0033

Definition at line 207 of file vl53l0x_device.h.

#define VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL 0x0055

Definition at line 208 of file vl53l0x_device.h.

#define VL53L0X_REG_FINAL_RANGE_CONFIG_VCSEL_PERIOD 0x0070

Definition at line 210 of file vl53l0x_device.h.

#define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_HI 0x0071

Definition at line 211 of file vl53l0x_device.h.

#define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_LO 0x0072

Definition at line 212 of file vl53l0x_device.h.

#define VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RATE_MCPS 0x0020

Definition at line 213 of file vl53l0x_device.h.

#define VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP 0x0046

Definition at line 215 of file vl53l0x_device.h.

#define VL53L0X_REG_SOFT_RESET_GO2_SOFT_RESET_N 0x00bf

Definition at line 218 of file vl53l0x_device.h.

#define VL53L0X_REG_IDENTIFICATION_MODEL_ID 0x00c0

Definition at line 219 of file vl53l0x_device.h.

#define VL53L0X_REG_IDENTIFICATION_REVISION_ID 0x00c2

Definition at line 220 of file vl53l0x_device.h.

#define VL53L0X_REG_OSC_CALIBRATE_VAL 0x00f8

Definition at line 222 of file vl53l0x_device.h.

#define VL53L0X_SIGMA_ESTIMATE_MAX_VALUE 65535

Definition at line 225 of file vl53l0x_device.h.

#define VL53L0X_REG_GLOBAL_CONFIG_VCSEL_WIDTH 0x032

Definition at line 228 of file vl53l0x_device.h.

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_0 0x0B0

Definition at line 229 of file vl53l0x_device.h.

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1 0x0B1

Definition at line 230 of file vl53l0x_device.h.

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2 0x0B2

Definition at line 231 of file vl53l0x_device.h.

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3 0x0B3

Definition at line 232 of file vl53l0x_device.h.

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4 0x0B4

Definition at line 233 of file vl53l0x_device.h.

#define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5 0x0B5

Definition at line 234 of file vl53l0x_device.h.

#define VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELECT 0xB6

Definition at line 236 of file vl53l0x_device.h.

#define VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF_SPAD 0x4E /* 0x14E */

Definition at line 237 of file vl53l0x_device.h.

#define VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET 0x4F /* 0x14F */

Definition at line 238 of file vl53l0x_device.h.

#define VL53L0X_REG_POWER_MANAGEMENT_GO1_POWER_FORCE 0x80

Definition at line 239 of file vl53l0x_device.h.

#define VL53L0X_SPEED_OF_LIGHT_IN_AIR 2997

Definition at line 245 of file vl53l0x_device.h.

#define VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA__EXTSUP_HV 0x0089

Definition at line 247 of file vl53l0x_device.h.

#define VL53L0X_REG_ALGO_PHASECAL_LIM 0x0030 /* 0x130 */

Definition at line 249 of file vl53l0x_device.h.

#define VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT 0x0030

Definition at line 250 of file vl53l0x_device.h.

Data Structure Documentation

VL53L0X_Dev_t Struct Reference

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

`#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

VL53L0X_DevData_t Struct Reference

VL53L0X PAL device ST private data structure

End user should never access any of these field directly.

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

Data Fields

- int32_t Part2PartOffsetNVMMicroMeter
- int32_t Part2PartOffsetAdjustmentNVMMicroMeter
- VL53L0X_DeviceParameters_t CurrentParameters
- VL53L0X_RangingMeasurementData_t LastRangeMeasure
- VL53L0X_HistogramMeasurementData_t LastHistogramMeasure
- VL53L0X_DeviceSpecificParameters_t DeviceSpecificParameters
- 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 UseInternalTuningSettings
- uint16_t LinearityCorrectiveGain

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 460 of file vl53l0x_def.h.

Field Documentation

int32_t VL53L0X_DevData_t::Part2PartOffsetNVMMicroMeter

backed up NVM value

Definition at line 461 of file vl53l0x_def.h.

int32_t VL53L0X_DevData_t::Part2PartOffsetAdjustmentNVMMicroMeter

backed up NVM value representing additional offset adjustment

Definition at line 463 of file vl53l0x_def.h.

VL53L0X_DeviceParameters_t VL53L0X_DevData_t::CurrentParameters

Current Device Parameter

Definition at line 465 of file vl53l0x_def.h.

VL53L0X_RangingMeasurementData_t VL53L0X_DevData_t::LastRangeMeasure

Ranging Data

Definition at line 467 of file vl53l0x_def.h.

VL53L0X_HistogramMeasurementData_t VL53L0X_DevData_t::LastHistogramMeasure

Histogram Data

Definition at line 469 of file vl53l0x_def.h.

VL53L0X_DeviceSpecificParameters_t VL53L0X_DevData_t::DeviceSpecificParameters

Parameters specific to the device

Definition at line 471 of file vl53l0x_def.h.

VL53L0X_SpadData_t VL53L0X_DevData_t::SpadData

Spad Data

Definition at line 473 of file vl53l0x_def.h.

uint8_t VL53L0X_DevData_t::SequenceConfig

Internal value for the sequence config

Definition at line 475 of file vl53l0x_def.h.

uint8_t VL53L0X_DevData_t::RangeFractionalEnable

Enable/Disable fractional part of ranging data

Definition at line 477 of file vl53l0x_def.h.

VL53L0X_State VL53L0X_DevData_t::PalState

Current state of the PAL for this device

Definition at line 479 of file vl53l0x_def.h.

VL53L0X_PowerModes VL53L0X_DevData_t::PowerMode

Current Power Mode

Definition at line 481 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 483 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 485 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 489 of file vl53l0x_def.h.

uint8_t VL53L0X_DevData_t::StopVariable

StopVariable used during the stop sequence

Definition at line 493 of file vl53l0x_def.h.

uint16_t VL53L0X_DevData_t::targetRefRate

Target Ambient Rate for Ref spad management

Definition at line 495 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 497 of file vl53l0x_def.h.

FixPoint1616_t VL53L0X_DevData_t::SignalEstimate

Signal Estimate - based on ambient & VCSEL rates and cross talk

Definition at line 501 of file vl53l0x_def.h.

FixPoint1616_t VL53L0X_DevData_t::LastSignalRefMcps

Latest Signal ref in Mcps

Definition at line 503 of file vl53l0x_def.h.

uint8_t* VL53L0X_DevData_t::pTuningSettingsPointer

Pointer for Tuning Settings table

Definition at line 505 of file vl53l0x_def.h.

uint8_t VL53L0X_DevData_t::UseInternalTuningSettings

Indicate if we use Tuning Settings table

Definition at line 507 of file vl53l0x_def.h.

uint16_t VL53L0X_DevData_t::LinearityCorrectiveGain

Linearity Corrective Gain value in x1000

Definition at line 509 of file vl53l0x_def.h.

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

- vl53l0x_def.h

VL53L0X_DeviceInfo_t Struct Reference

Defines the parameters of the Get Device Info Functions.

```
#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

VL53L0X_DeviceParameters_t Struct Reference

Defines all parameters for the device.

`#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_OF_CHECKS]
- uint8_t LimitChecksStatus [VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS]
- FixPoint1616_t LimitChecksValue [VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS]
- VL53L0X_DMaxLUT_t dmax_lut
- uint8_t WrapAroundCheckEnable

Detailed Description

Defines all parameters for the device.

Definition at line 249 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 250 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 252 of file vl53l0x_def.h.

uint32_t VL53L0X_DeviceParameters_t::MeasurementTimingBudgetMicroSeconds

Defines the allowed total time for a single measurement

Definition at line 256 of file vl53l0x_def.h.

uint32_t VL53L0X_DeviceParameters_t::InterMeasurementPeriodMilliSeconds

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

Definition at line 258 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceParameters_t::XTalkCompensationEnable

Tells if Crosstalk compensation shall be enable or not

Definition at line 262 of file vl53l0x_def.h.

uint16_t VL53L0X_DeviceParameters_t::XTalkCompensationRangeMilliMeter

CrossTalk compensation range in millimeter

Definition at line 264 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 266 of file vl53l0x_def.h.

int32_t VL53L0X_DeviceParameters_t::RangeOffsetMicroMeters

Range offset adjustment (mm).

Definition at line 270 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceParameters_t::LimitChecksEnable[VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS]

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

Definition at line 273 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceParameters_t::LimitChecksStatus[VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS]

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

Definition at line 275 of file vl53l0x_def.h.

FixPoint1616_t VL53L0X_DeviceParameters_t::LimitChecksValue[VL53L0X_CHECKENABLE_NUMBER_OF_CHECKS]

This Array store all the Limit Check value for this device

Definition at line 279 of file vl53l0x_def.h.

VL53L0X_DMaxLUT_t VL53L0X_DeviceParameters_t::dmax_lut

Lookup table defining ambient rates and associated dmax values.

Definition at line 282 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceParameters_t::WrapAroundCheckEnable

Tells if Wrap Around Check shall be enable or not

Definition at line 287 of file vl53l0x_def.h.

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

- vl53l0x_def.h

VL53L0X_DeviceSpecificParameters_t Struct Reference

`#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 406 of file vl53l0x_def.h.

Field Documentation

FixPoint1616_t VL53L0X_DeviceSpecificParameters_t::OscFrequencyMHz

Definition at line 407 of file vl53l0x_def.h.

uint16_t VL53L0X_DeviceSpecificParameters_t::LastEncodedTimeout

Definition at line 409 of file vl53l0x_def.h.

VL53L0X_GpioFunctionality VL53L0X_DeviceSpecificParameters_t::Pin0GpioFunctionality

Definition at line 412 of file vl53l0x_def.h.

uint32_t VL53L0X_DeviceSpecificParameters_t::FinalRangeTimeoutMicroSecs

Execution time of the final range

Definition at line 415 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceSpecificParameters_t::FinalRangeVcselPulsePeriod

Vcsel pulse period (pll clocks) for the final range measurement

Definition at line 417 of file vl53l0x_def.h.

uint32_t VL53L0X_DeviceSpecificParameters_t::PreRangeTimeoutMicroSecs

Execution time of the final range

Definition at line 419 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceSpecificParameters_t::PreRangeVcselPulsePeriod

Vcsel pulse period (pll clocks) for the pre-range measurement

Definition at line 421 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 424 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 426 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 430 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceSpecificParameters_t::ReadDataFromDeviceDone

Definition at line 437 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceSpecificParameters_t::ModuleId

Definition at line 438 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceSpecificParameters_t::Revision

Definition at line 439 of file vl53l0x_def.h.

char VL53L0X_DeviceSpecificParameters_t::ProductId[VL53L0X_MAX_STRING_LENGTH]

Definition at line 440 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceSpecificParameters_t::ReferenceSpadCount

Definition at line 442 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceSpecificParameters_t::ReferenceSpadType

Definition at line 443 of file vl53l0x_def.h.

uint8_t VL53L0X_DeviceSpecificParameters_t::RefSpadsInitialised

Definition at line 444 of file vl53l0x_def.h.

uint32_t VL53L0X_DeviceSpecificParameters_t::PartUIDUpper

Unique Part ID Upper

Definition at line 445 of file vl53l0x_def.h.

uint32_t VL53L0X_DeviceSpecificParameters_t::PartUIDLower

Unique Part ID Lower Peek Signal rate at 400 mm

Definition at line 446 of file vl53l0x_def.h.

FixPoint1616_t VL53L0X_DeviceSpecificParameters_t::SignalRateMeasFixed400mm

Definition at line 448 of file vl53l0x_def.h.

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

- vl53l0x_def.h

VL53L0X_DMaxLUT_t Struct Reference

Structure defining data pair that makes up the DMAX Lookup table.

`#include <vl53l0x_def.h>`

Data Fields

- FixPoint1616_t ambRate_mcps [VL53L0X_DMAX_LUT_SIZE]
- FixPoint1616_t dmax_mm [VL53L0X_DMAX_LUT_SIZE]

Detailed Description

Structure defining data pair that makes up the DMAX Lookup table.

Definition at line 240 of file vl53l0x_def.h.

Field Documentation

FixPoint1616_t VL53L0X_DMaxLUT_t::ambRate_mcps[VL53L0X_DMAX_LUT_SIZE]

Ambient rate (mcps)

Definition at line 241 of file vl53l0x_def.h.

FixPoint1616_t VL53L0X_DMaxLUT_t::dmax_mm[VL53L0X_DMAX_LUT_SIZE]

DMAX Value (mm)

Definition at line 243 of file vl53l0x_def.h.

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

- vl53l0x_def.h

VL53L0X_HistogramData_t Struct Reference

Histogram measurement data.

`#include <vl53l0x_def.h>`

Detailed Description

Histogram measurement data.

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

- vl53l0x_def.h

VL53L0X_HistogramMeasurementData_t Struct Reference

`#include <vl53l0x_def.h>`

Data Fields

- uint32_t HistogramData [VL53L0X_HISTOGRAM_BUFFER_SIZE]
- uint8_t FirstBin
- uint8_t BufferSize
- uint8_t NumberOfBins
- VL53L0X_DeviceError ErrorStatus

Detailed Description

Definition at line 375 of file vl53l0x_def.h.

Field Documentation

uint32_t VL53L0X_HistogramMeasurementData_t::HistogramData[VL53L0X_HISTOGRAM_BUFFER_SIZE]

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

Definition at line 377 of file vl53l0x_def.h.

uint8_t VL53L0X_HistogramMeasurementData_t::FirstBin

First Bin value

Definition at line 382 of file vl53l0x_def.h.

uint8_t VL53L0X_HistogramMeasurementData_t::BufferSize

Buffer Size - Set by the user.

Definition at line 383 of file vl53l0x_def.h.

uint8_t VL53L0X_HistogramMeasurementData_t::NumberOfBins

Number of bins filled by the histogram measurement

Definition at line 384 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 387 of file vl53l0x_def.h.

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

- vl53l0x_def.h

VL53L0X_RangeData_t Struct Reference

Range measurement data.

`#include <vl53l0x_def.h>`

Detailed Description

Range measurement data.

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

- vl53l0x_def.h

VL53L0X_RangingMeasurementData_t Struct Reference

`#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 321 of file vl53l0x_def.h.

Field Documentation

uint32_t VL53L0X_RangingMeasurementData_t::TimeStamp

32-bit time stamp.

Definition at line 322 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 323 of file vl53l0x_def.h.

uint16_t VL53L0X_RangingMeasurementData_t::RangeMilliMeter

range distance in millimeter.

Definition at line 329 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 331 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 337 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 ambien t light.

Definition at line 342 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 348 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 353 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 357 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 361 of file vl53l0x_def.h.

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

- vl53l0x_def.h

VL53L0X_SchedulerSequenceSteps_t Struct Reference

```
#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 548 of file vl53l0x_def.h.

Field Documentation

uint8_t VL53L0X_SchedulerSequenceSteps_t::TccOn

Reports if Target Centre Check On

Definition at line 549 of file vl53l0x_def.h.

uint8_t VL53L0X_SchedulerSequenceSteps_t::MsrcOn

Reports if MSRC On

Definition at line 550 of file vl53l0x_def.h.

uint8_t VL53L0X_SchedulerSequenceSteps_t::DssOn

Reports if DSS On

Definition at line 551 of file vl53l0x_def.h.

uint8_t VL53L0X_SchedulerSequenceSteps_t::PreRangeOn

Reports if Pre-Range On

Definition at line 552 of file vl53l0x_def.h.

uint8_t VL53L0X_SchedulerSequenceSteps_t::FinalRangeOn

Reports if Final-Range On

Definition at line 553 of file vl53l0x_def.h.

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

- vl53l0x_def.h

VL53L0X_SpadData_t Struct Reference

Spad Configuration Data.

```
#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 399 of file vl53l0x_def.h.

Field Documentation

uint8_t VL53L0X_SpadData_t::RefSpadEnables[VL53L0X_REF_SPAD_BUFFER_SIZE]

Reference Spad Enables

Definition at line 400 of file vl53l0x_def.h.

uint8_t VL53L0X_SpadData_t::RefGoodSpadMap[VL53L0X_REF_SPAD_BUFFER_SIZE]

Reference Spad Good Spad Map

Definition at line 402 of file vl53l0x_def.h.

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

- vl53l0x_def.h

VL53L0X_Version_t Struct Reference

Defines the parameters of the Get Version Functions.

`#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

File Documentation

PAL_disclaimer.c File Reference

no code doxygen doc only

Detailed Description

no code doxygen doc only

Definition in file PAL_disclaimer.c.

---

vl53l0x_api.h File Reference

```
#include "vl53l0x_api_strings.h"
#include "vl53l0x_def.h"
#include "vl53l0x_platform.h"
```

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_Version_t *pPalSpecVersion)
  Return the PAL Specification Version used for the current implementation.
- VL53L0X_API VL53L0X_Error VL53L0X_GetProductRevision (VL53L0X_DEV Dev, uint8_t *pProductRevisionMajor, uint8_t *pProductRevisionMinor)
  Reads the Product Revision for a for given Device This function can be used to distinguish cut1.0 from cut1.1.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceInfo (VL53L0X_DEV Dev, VL53L0X_DeviceInfo_t *pVL53L0X_DeviceInfo)
  Reads the Device information for given Device.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceErrorStatus (VL53L0X_DEV Dev, VL53L0X_DeviceError *pDeviceErrorStatus)
  Read current status of the error register for the selected device.
- VL53L0X_API VL53L0X_Error VL53L0X_GetRangeStatusString (uint8_t RangeStatus, char *pRangeStatusString)
  Human readable Range Status string for a given RangeStatus.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceErrorString (VL53L0X_DeviceError ErrorCode, char *pDeviceErrorString)
  Human readable error string for a given Error Code.
- VL53L0X_API VL53L0X_Error VL53L0X_GetPalErrorString (VL53L0X_Error PalErrorCode, char *pPalErrorString)
  Human readable error string for current PAL error status.
- VL53L0X_API VL53L0X_Error VL53L0X_GetPalStateString (VL53L0X_State PalStateCode, char *pPalStateString)
  Human readable PAL State string.

- VL53L0X_API VL53L0X_Error VL53L0X_GetPalState (VL53L0X_DEV Dev, VL53L0X_State *pPalState)
  Reads the internal state of the PAL for a given Device.
- 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.
- VL53L0X_API VL53L0X_Error VL53L0X_GetPowerMode (VL53L0X_DEV Dev, VL53L0X_PowerModes *pPowerMode)
  Get the power mode for a given Device.
- VL53L0X_API VL53L0X_Error VL53L0X_SetOffsetCalibrationDataMicroMeter (VL53L0X_DEV Dev, int32_t OffsetCalibrationDataMicroMeter)
  Set or over-hide part to part calibration offset.
- VL53L0X_API VL53L0X_Error VL53L0X_GetOffsetCalibrationDataMicroMeter (VL53L0X_DEV Dev, int32_t *pOffsetCalibrationDataMicroMeter)
  Get part to part calibration offset.
- VL53L0X_API VL53L0X_Error VL53L0X_SetLinearityCorrectiveGain (VL53L0X_DEV Dev, int16_t LinearityCorrectiveGain)
  Set the linearity corrective gain.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLinearityCorrectiveGain (VL53L0X_DEV Dev, uint16_t *pLinearityCorrectiveGain)
  Get the linearity corrective gain.
- VL53L0X_API VL53L0X_Error VL53L0X_SetGroupParamHold (VL53L0X_DEV Dev, uint8_t GroupParamHold)
  Set Group parameter Hold state.
- VL53L0X_API VL53L0X_Error VL53L0X_GetUpperLimitMilliMeter (VL53L0X_DEV Dev, uint16_t *pUpperLimitMilliMeter)
  Get the maximal distance for actual setup.
- VL53L0X_Error VL53L0X_GetTotalSignalRate (VL53L0X_DEV Dev, FixPoint1616_t *pTotalSignalRate)
  Get the Total Signal Rate.
- VL53L0X_API VL53L0X_Error VL53L0X_SetDeviceAddress (VL53L0X_DEV Dev, uint8_t DeviceAddress)
  Set new device address.
- VL53L0X_API VL53L0X_Error VL53L0X_DataInit (VL53L0X_DEV Dev)
  One time device initialization.
- VL53L0X_API VL53L0X_Error VL53L0X_SetTuningSettingBuffer (VL53L0X_DEV Dev, uint8_t *pTuningSettingBuffer, uint8_t UseInternalTuningSettings)
  Set the tuning settings pointer.
- 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.
- 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.
- 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.
- 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.

- VL53L0X_API VL53L0X_Error VL53L0X_SetDeviceParameters (VL53L0X_DEV Dev, const VL53L0X_DeviceParameters_t *pDeviceParameters)
  Prepare device for operation.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceParameters (VL53L0X_DEV Dev, VL53L0X_DeviceParameters_t *pDeviceParameters)
  Retrieve current device parameters.
- VL53L0X_API VL53L0X_Error VL53L0X_SetDeviceMode (VL53L0X_DEV Dev, VL53L0X_DeviceModes DeviceMode)
  Set a new device mode.
- VL53L0X_API VL53L0X_Error VL53L0X_GetDeviceMode (VL53L0X_DEV Dev, VL53L0X_DeviceModes *pDeviceMode)
  Get current new device mode.
- VL53L0X_API VL53L0X_Error VL53L0X_SetRangeFractionEnable (VL53L0X_DEV Dev, uint8_t Enable)
  Sets the resolution of range measurements.
- VL53L0X_API VL53L0X_Error VL53L0X_GetFractionEnable (VL53L0X_DEV Dev, uint8_t *pEnable)
  Gets the fraction enable parameter indicating the resolution of range measurements.
- VL53L0X_API VL53L0X_Error VL53L0X_SetHistogramMode (VL53L0X_DEV Dev, VL53L0X_HistogramModes HistogramMode)
  Set a new Histogram mode.
- VL53L0X_API VL53L0X_Error VL53L0X_GetHistogramMode (VL53L0X_DEV Dev, VL53L0X_HistogramModes *pHistogramMode)
  Get current new device mode.
- VL53L0X_API VL53L0X_Error VL53L0X_SetMeasurementTimingBudgetMicroSeconds (VL53L0X_DEV Dev, uint32_t MeasurementTimingBudgetMicroSeconds)
  Set Ranging Timing Budget in microseconds.
- VL53L0X_API VL53L0X_Error VL53L0X_GetMeasurementTimingBudgetMicroSeconds (VL53L0X_DEV Dev, uint32_t *pMeasurementTimingBudgetMicroSeconds)
  Get Ranging Timing Budget in microseconds.
- VL53L0X_API VL53L0X_Error VL53L0X_GetVcselPulsePeriod (VL53L0X_DEV Dev, VL53L0X_VcselPeriod VcselPeriodType, uint8_t *pVCSELPulsePeriod)
  Gets the VCSEL pulse period.
- VL53L0X_API VL53L0X_Error VL53L0X_SetVcselPulsePeriod (VL53L0X_DEV Dev, VL53L0X_VcselPeriod VcselPeriodType, uint8_t VCSELPulsePeriod)
  Sets the VCSEL pulse period.
- VL53L0X_API VL53L0X_Error VL53L0X_SetSequenceStepEnable (VL53L0X_DEV Dev, VL53L0X_SequenceStepId SequenceStepId, uint8_t SequenceStepEnabled)
  Sets the (on/off) state of a requested sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepEnable (VL53L0X_DEV Dev, VL53L0X_SequenceStepId SequenceStepId, uint8_t *pSequenceStepEnabled)
  Gets the (on/off) state of a requested sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepEnables (VL53L0X_DEV Dev, VL53L0X_SchedulerSequenceSteps_t *pSchedulerSequenceSteps)
  Gets the (on/off) state of all sequence steps.
- VL53L0X_API VL53L0X_Error VL53L0X_SetSequenceStepTimeout (VL53L0X_DEV Dev, VL53L0X_SequenceStepId SequenceStepId, FixPoint1616_t TimeOutMilliSecs)
  Sets the timeout of a requested sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepTimeout (VL53L0X_DEV Dev, VL53L0X_SequenceStepId SequenceStepId, FixPoint1616_t *pTimeOutMilliSecs)
  Gets the timeout of a requested sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_GetNumberOfSequenceSteps (VL53L0X_DEV Dev, uint8_t *pNumberOfSequenceSteps)

Gets number of sequence steps managed by the API.

- VL53L0X_API VL53L0X_Error VL53L0X_GetSequenceStepsInfo (VL53L0X_SequenceStepId SequenceStepId, char *pSequenceStepsString)
  Gets the name of a given sequence step.
- VL53L0X_API VL53L0X_Error VL53L0X_SetInterMeasurementPeriodMilliSeconds (VL53L0X_DEV Dev, uint32_t InterMeasurementPeriodMilliSeconds)
  Program continuous mode Inter-Measurement period in milliseconds.
- VL53L0X_API VL53L0X_Error VL53L0X_GetInterMeasurementPeriodMilliSeconds (VL53L0X_DEV Dev, uint32_t *pInterMeasurementPeriodMilliSeconds)
  Get continuous mode Inter-Measurement period in milliseconds.
- VL53L0X_API VL53L0X_Error VL53L0X_SetXTalkCompensationEnable (VL53L0X_DEV Dev, uint8_t XTalkCompensationEnable)
  Enable/Disable Cross talk compensation feature.
- VL53L0X_API VL53L0X_Error VL53L0X_GetXTalkCompensationEnable (VL53L0X_DEV Dev, uint8_t *pXTalkCompensationEnable)
  Get Cross talk compensation rate.
- VL53L0X_API VL53L0X_Error VL53L0X_SetXTalkCompensationRateMegaCps (VL53L0X_DEV Dev, FixPoint1616_t XTalkCompensationRateMegaCps)
  Set Cross talk compensation rate.
- VL53L0X_API VL53L0X_Error VL53L0X_GetXTalkCompensationRateMegaCps (VL53L0X_DEV Dev, FixPoint1616_t *pXTalkCompensationRateMegaCps)
  Get Cross talk compensation rate.
- VL53L0X_API VL53L0X_Error VL53L0X_SetRefCalibration (VL53L0X_DEV Dev, uint8_t 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 (uint16_t *pNumberOfLimitCheck)
  Get the number of the check limit managed by a given Device.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckInfo (VL53L0X_DEV Dev, uint16_t LimitCheckId, char *pLimitCheckString)
  Return a description string for a given limit check number.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckStatus (VL53L0X_DEV Dev, uint16_t LimitCheckId, uint8_t *pLimitCheckStatus)
  Return a the Status of the specified check limit.
- VL53L0X_API VL53L0X_Error VL53L0X_SetLimitCheckEnable (VL53L0X_DEV Dev, uint16_t LimitCheckId, uint8_t LimitCheckEnable)
  Enable/Disable a specific limit check.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckEnable (VL53L0X_DEV Dev, uint16_t LimitCheckId, uint8_t *pLimitCheckEnable)
  Get specific limit check enable state.
- VL53L0X_API VL53L0X_Error VL53L0X_SetLimitCheckValue (VL53L0X_DEV Dev, uint16_t LimitCheckId, FixPoint1616_t LimitCheckValue)
  Set a specific limit check value.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckValue (VL53L0X_DEV Dev, uint16_t LimitCheckId, FixPoint1616_t *pLimitCheckValue)
  Get a specific limit check value.
- VL53L0X_API VL53L0X_Error VL53L0X_GetLimitCheckCurrent (VL53L0X_DEV Dev, uint16_t LimitCheckId, FixPoint1616_t *pLimitCheckCurrent)
  Get the current value of the signal used for the limit check.

- VL53L0X_API VL53L0X_Error VL53L0X_SetWrapAroundCheckEnable (VL53L0X_DEV Dev, uint8_t WrapAroundCheckEnable)
  Enable (or disable) Wrap around Check.
- VL53L0X_API VL53L0X_Error VL53L0X_GetWrapAroundCheckEnable (VL53L0X_DEV Dev, uint8_t *pWrapAroundCheckEnable)
  Get setup of Wrap around Check.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformSingleMeasurement (VL53L0X_DEV Dev)
  Single shot measurement.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformRefCalibration (VL53L0X_DEV Dev, uint8_t *pVhvSettings, uint8_t *pPhaseCal)
  Perform Reference Calibration.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformXTalkMeasurement (VL53L0X_DEV Dev, uint32_t TimeoutMs, FixPoint1616_t *pXtalkPerSpad, uint8_t *pAmbientTooHigh)
  Perform XTalk Measurement.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformXTalkCalibration (VL53L0X_DEV Dev, FixPoint1616_t XTalkCalDistance, FixPoint1616_t *pXTalkCompensationRateMegaCps)
  Perform XTalk Calibration.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformOffsetCalibration (VL53L0X_DEV Dev, FixPoint1616_t CalDistanceMilliMeter, int32_t *pOffsetMicroMeter)
  Perform Offset Calibration.
- VL53L0X_API VL53L0X_Error VL53L0X_StartMeasurement (VL53L0X_DEV Dev)
  Start device measurement.
- VL53L0X_API VL53L0X_Error VL53L0X_StopMeasurement (VL53L0X_DEV Dev)
  Stop device measurement.
- VL53L0X_API VL53L0X_Error VL53L0X_GetMeasurementDataReady (VL53L0X_DEV Dev, uint8_t *pMeasurementDataReady)
  Return Measurement Data Ready.
- VL53L0X_API VL53L0X_Error VL53L0X_WaitDeviceReadyForNewMeasurement (VL53L0X_DEV Dev, uint32_t MaxLoop)
  Wait for device ready for a new measurement command.
- VL53L0X_API VL53L0X_Error VL53L0X_GetMeasurementRefSignal (VL53L0X_DEV Dev, FixPoint1616_t *pMeasurementRefSignal)
  Retrieve the Reference Signal after a measurements.
- VL53L0X_API VL53L0X_Error VL53L0X_GetRangingMeasurementData (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData)
  Retrieve the measurements from device for a given setup.
- VL53L0X_API VL53L0X_Error VL53L0X_GetHistogramMeasurementData (VL53L0X_DEV Dev, VL53L0X_HistogramMeasurementData_t *pHistogramMeasurementData)
  Retrieve the measurements from device for a given setup.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformSingleRangingMeasurement (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData)
  Performs a single ranging measurement and retrieve the ranging measurement data.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformSingleHistogramMeasurement (VL53L0X_DEV Dev, VL53L0X_HistogramMeasurementData_t *pHistogramMeasurementData)
  Performs a single histogram measurement and retrieve the histogram measurement data Is equivalent to VL53L0X_PerformSingleMeasurement + VL53L0X_GetHistogramMeasurementData.
- 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.
- VL53L0X_API VL53L0X_Error VL53L0X_GetNumberOfROIZones (VL53L0X_DEV Dev, uint8_t *pNumberOfROIZones)
  Get the number of ROI Zones managed by the Device.

- VL53L0X_API VL53L0X_Error VL53L0X_GetMaxNumberOfROIZones (VL53L0X_DEV Dev, uint8_t *pMaxNumberOfROIZones)
  Get the Maximum number of ROI Zones managed by the Device.
- 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.
- 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.
- 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.
- 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.
- VL53L0X_API VL53L0X_Error VL53L0X_GetStopCompletedStatus (VL53L0X_DEV Dev, uint32_t *pStopStatus)
  Return device stop completion status.
- VL53L0X_API VL53L0X_Error VL53L0X_ClearInterruptMask (VL53L0X_DEV Dev, uint32_t InterruptMask)
  Clear given system interrupt condition.
- VL53L0X_API VL53L0X_Error VL53L0X_GetInterruptMaskStatus (VL53L0X_DEV Dev, uint32_t *pInterruptMaskStatus)
  Return device interrupt status.
- VL53L0X_API VL53L0X_Error VL53L0X_EnableInterruptMask (VL53L0X_DEV Dev, uint32_t InterruptMask)
  Configure ranging interrupt reported to system.
- VL53L0X_API VL53L0X_Error VL53L0X_SetSpadAmbientDamperThreshold (VL53L0X_DEV Dev, uint16_t SpadAmbientDamperThreshold)
  Set the SPAD Ambient Damper Threshold value.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSpadAmbientDamperThreshold (VL53L0X_DEV Dev, uint16_t *pSpadAmbientDamperThreshold)
  Get the current SPAD Ambient Damper Threshold value.
- VL53L0X_API VL53L0X_Error VL53L0X_SetSpadAmbientDamperFactor (VL53L0X_DEV Dev, uint16_t SpadAmbientDamperFactor)
  Set the SPAD Ambient Damper Factor value.
- VL53L0X_API VL53L0X_Error VL53L0X_GetSpadAmbientDamperFactor (VL53L0X_DEV Dev, uint16_t *pSpadAmbientDamperFactor)
  Get the current SPAD Ambient Damper Factor value.
- VL53L0X_API VL53L0X_Error VL53L0X_PerformRefSpadManagement (VL53L0X_DEV Dev, uint32_t *refSpadCount, uint8_t *isApertureSpads)
  Performs Reference Spad Management.
- VL53L0X_API VL53L0X_Error VL53L0X_SetReferenceSpads (VL53L0X_DEV Dev, uint32_t refSpadCount, uint8_t isApertureSpads)
  Applies Reference SPAD configuration.
- VL53L0X_API VL53L0X_Error VL53L0X_GetReferenceSpads (VL53L0X_DEV Dev, uint32_t *refSpadCount, uint8_t *isApertureSpads)
  Retrieves SPAD configuration.

Macro Definition Documentation

#define VL53L0X_API

Definition at line 48 of file vl53l0x_api.h.

---

vl53l0x_api_calibration.h File Reference

```
#include "vl53l0x_def.h"
#include "vl53l0x_platform.h"
```

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 *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*)

---

vl53l0x_api_core.h File Reference

```
#include "vl53l0x_def.h"
#include "vl53l0x_platform.h"
```

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_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 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_timeout)
- 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_Error VL53L0X_set_measurement_timing_budget_micro_seconds (VL53L0X_DEV Dev, uint32_t MeasurementTimingBudgetMicroSeconds)
- VL53L0X_Error VL53L0X_get_measurement_timing_budget_micro_seconds (VL53L0X_DEV Dev, uint32_t *pMeasurementTimingBudgetMicroSeconds)
- VL53L0X_Error VL53L0X_load_tuning_settings (VL53L0X_DEV Dev, uint8_t *pTuningSettingBuffer)
- VL53L0X_Error VL53L0X_calc_sigma_estimate (VL53L0X_DEV Dev, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData, FixPoint1616_t *pSigmaEstimate)
- VL53L0X_Error VL53L0X_calc_dmax (VL53L0X_DEV Dev, FixPoint1616_t ambRateMeas, 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, uint8_t DeviceRangeStatus, FixPoint1616_t SignalRate, uint16_t EffectiveSpadRtnCount, VL53L0X_RangingMeasurementData_t *pRangingMeasurementData, uint8_t *pPalRangeStatus)
- 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)

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 *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_timeout*)

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_Error VL53L0X_set_measurement_timing_budget_micro_seconds (VL53L0X_DEV *Dev*, uint32_t *MeasurementTimingBudgetMicroSeconds*)

VL53L0X_Error VL53L0X_get_measurement_timing_budget_micro_seconds (VL53L0X_DEV *Dev*, uint32_t * *pMeasurementTimingBudgetMicroSeconds*)

VL53L0X_Error VL53L0X_load_tuning_settings (VL53L0X_DEV *Dev*, uint8_t * *pTuningSettingBuffer*)

VL53L0X_Error VL53L0X_calc_sigma_estimate (VL53L0X_DEV *Dev*, VL53L0X_RangingMeasurementData_t * *pRangingMeasurementData*, FixPoint1616_t * *pSigmaEstimate*)

VL53L0X_Error VL53L0X_calc_dmax (VL53L0X_DEV *Dev*, FixPoint1616_t *ambRateMeas*, 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*, uint8_t *DeviceRangeStatus*, FixPoint1616_t *SignalRate*, uint16_t *EffectiveSpadRtnCount*, VL53L0X_RangingMeasurementData_t * *pRangingMeasurementData*, uint8_t * *pPalRangeStatus*)

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*)

---

vl53l0x_api_ranging.h File Reference

```
#include "vl53l0x_def.h"
#include "vl53l0x_platform.h"
```

---

vl53l0x_api_strings.h File Reference

```
#include "vl53l0x_def.h"
#include "vl53l0x_platform.h"
```

Macros

- #define VL53L0X_STRING_DEVICE_INFO_NAME "VL53L0X cut1.0"
- #define VL53L0X_STRING_DEVICE_INFO_NAME_TS0 "VL53L0X TS0"
- #define VL53L0X_STRING_DEVICE_INFO_NAME_TS1 "VL53L0X TS1"
- #define VL53L0X_STRING_DEVICE_INFO_NAME_TS2 "VL53L0X TS2"
- #define VL53L0X_STRING_DEVICE_INFO_NAME_ES1 "VL53L0X ES1 or later"
- #define VL53L0X_STRING_DEVICE_INFO_TYPE "VL53L0X"
- #define VL53L0X_STRING_ERROR_NONE "No Error"
- #define VL53L0X_STRING_ERROR_CALIBRATION_WARNING "Calibration Warning 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 error"
- #define VL53L0X_STRING_ERROR_NOT_SUPPORTED "Not supported error"
- #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 error"
- #define VL53L0X_STRING_ERROR_BUFFER_TOO_SMALL "Buffer too small"
- #define VL53L0X_STRING_ERROR_GPIO_NOT_EXISTING "GPIO not existing"
- #define VL53L0X_STRING_ERROR_GPIO_FUNCTIONALITY_NOT_SUPPORTED "GPIO funct not supported"
- #define VL53L0X_STRING_ERROR_INTERRUPT_NOT_CLEARED "Interrupt not Cleared"
- #define VL53L0X_STRING_ERROR_CONTROL_INTERFACE "Control Interface Error"
- #define VL53L0X_STRING_ERROR_INVALID_COMMAND "Invalid Command Error"
- #define VL53L0X_STRING_ERROR_DIVISION_BY_ZERO "Division by zero Error"
- #define VL53L0X_STRING_ERROR_REF_SPAD_INIT "Reference Spad Init Error"
- #define VL53L0X_STRING_ERROR_NOT_IMPLEMENTED "Not implemented error"
- #define VL53L0X_STRING_UNKNOW_ERROR_CODE "Unknown Error Code"
- #define VL53L0X_STRING_RANGESTATUS_NONE "No Update"
- #define VL53L0X_STRING_RANGESTATUS_RANGEVALID "Range Valid"
- #define VL53L0X_STRING_RANGESTATUS_SIGMA "Sigma Fail"
- #define VL53L0X_STRING_RANGESTATUS_SIGNAL "Signal Fail"

- #define VL53L0X_STRING_RANGESTATUS_MINRANGE "Min Range Fail"
- #define VL53L0X_STRING_RANGESTATUS_PHASE "Phase Fail"
- #define VL53L0X_STRING_RANGESTATUS_HW "Hardware Fail"
- #define VL53L0X_STRING_STATE_POWERDOWN "POWERDOWN State"
- #define VL53L0X_STRING_STATE_WAIT_STATICINIT "Wait for staticinit State"
- #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_VCSELCONTINUITYTESTFAILURE "VCSEL Continuity Test Failure"
- #define VL53L0X_STRING_DEVICEERROR_VCSELWATCHDOGTESTFAILURE "VCSEL Watchdog Test Failure"
- #define VL53L0X_STRING_DEVICEERROR_NOVHVVALUEFOUND "No VHV Value found"
- #define VL53L0X_STRING_DEVICEERROR_MSRCNOTARGET "MSRC No Target Error"
- #define VL53L0X_STRING_DEVICEERROR_SNRCHECK "SNR Check Exit"
- #define VL53L0X_STRING_DEVICEERROR_RANGEPHASECHECK "Range Phase Check Error"
- #define VL53L0X_STRING_DEVICEERROR_SIGMATHRESHOLDCHECK "Sigma Threshold Check Error"
- #define VL53L0X_STRING_DEVICEERROR_TCC "TCC Error"
- #define VL53L0X_STRING_DEVICEERROR_PHASECONSISTENCY "Phase Consistency Error"
- #define VL53L0X_STRING_DEVICEERROR_MINCLIP "Min Clip Error"
- #define VL53L0X_STRING_DEVICEERROR_RANGECOMPLETE "Range Complete"
- #define VL53L0X_STRING_DEVICEERROR_ALGOUNDERFLOW "Range Algo Underflow Error"
- #define VL53L0X_STRING_DEVICEERROR_ALGOOVERFLOW "Range Algo Overlow Error"
- #define VL53L0X_STRING_DEVICEERROR_RANGEIGNORETHRESHOLD "Range Ignore Threshold Error"
- #define VL53L0X_STRING_DEVICEERROR_UNKNOWN "Unknown error code"
- #define VL53L0X_STRING_CHECKENABLE_SIGMA_FINAL_RANGE "SIGMA FINAL RANGE"
- #define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE "SIGNAL RATE FINAL RANGE"
- #define VL53L0X_STRING_CHECKENABLE_SIGNAL_REF_CLIP "SIGNAL REF CLIP"
- #define VL53L0X_STRING_CHECKENABLE_RANGE_IGNORE_THRESHOLD "RANGE IGNORE THRESHOLD"
- #define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_MSRC "SIGNAL RATE MSRC"
- #define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_PRE_RANGE "SIGNAL RATE PRE RANGE"
- #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 RANGE"
- #define VL53L0X_STRING_SEQUENCESTEP_FINAL_RANGE "FINAL RANGE"

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 "Calibration 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_SUPPORTED "GPIO 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_VCSELCONTINUITYTESTFAILURE "VCSEL Continuity Test Failure"

Definition at line 218 of file vl53l0x_api_strings.h.

#define VL53L0X_STRING_DEVICEERROR_VCSELWATCHDOGTESTFAILURE "VCSEL 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 "Range Phase Check Error"

Definition at line 228 of file vl53l0x_api_strings.h.

#define VL53L0X_STRING_DEVICEERROR_SIGMATHRESHOLDCHECK "Sigma 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 Overlow Error"

Definition at line 242 of file vl53l0x_api_strings.h.

#define VL53L0X_STRING_DEVICEERROR_RANGEIGNORETHRESHOLD "Range 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 "RANGE IGNORE THRESHOLD"

Definition at line 256 of file vl53l0x_api_strings.h.

#define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_MSRC "SIGNAL RATE MSRC"

Definition at line 258 of file vl53l0x_api_strings.h.

#define VL53L0X_STRING_CHECKENABLE_SIGNAL_RATE_PRE_RANGE "SIGNAL 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_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*)

---

vl53l0x_def.h File Reference

Type definitions for VL53L0X API.

```
#include "vl53l0x_device.h"
#include "vl53l0x_types.h"
```

Data Structures

- struct VL53L0X_Version_t
- *Defines the parameters of the Get Version Functions.* struct VL53L0X_DeviceInfo_t
- *Defines the parameters of the Get Device Info Functions.* struct VL53L0X_DMaxLUT_t
- *Structure defining data pair that makes up the DMAX Lookup table.* struct VL53L0X_DeviceParameters_t
- *Defines all parameters for the device.* struct VL53L0X_RangingMeasurementData_t
- struct VL53L0X_HistogramMeasurementData_t
- struct VL53L0X_SpadData_t
- *Spad Configuration Data.* 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.* struct VL53L0X_SchedulerSequenceSteps_t

Macros

- #define VL53L0X10_SPECIFICATION_VER_MAJOR 1
  PAL SPECIFICATION major version.
- #define VL53L0X10_SPECIFICATION_VER_MINOR 2
  PAL SPECIFICATION minor version.
- #define VL53L0X10_SPECIFICATION_VER_SUB 7
  PAL SPECIFICATION sub version.

- #define VL53L0X10_SPECIFICATION_VER_REVISION 1440
  PAL SPECIFICATION sub version.
- #define VL53L0X10_IMPLEMENTATION_VER_MAJOR 1
  VL53L0X PAL IMPLEMENTATION major version.
- #define VL53L0X10_IMPLEMENTATION_VER_MINOR 0
  VL53L0X PAL IMPLEMENTATION minor version.
- #define VL53L0X10_IMPLEMENTATION_VER_SUB 9
  VL53L0X PAL IMPLEMENTATION sub version.
- #define VL53L0X10_IMPLEMENTATION_VER_REVISION 3673
  VL53L0X PAL IMPLEMENTATION sub version.
- #define VL53L0X_SPECIFICATION_VER_MAJOR 1
  PAL SPECIFICATION major version.
- #define VL53L0X_SPECIFICATION_VER_MINOR 2
  PAL SPECIFICATION minor version.
- #define VL53L0X_SPECIFICATION_VER_SUB 7
  PAL SPECIFICATION sub version.
- #define VL53L0X_SPECIFICATION_VER_REVISION 1440
  PAL SPECIFICATION sub version.
- #define VL53L0X_IMPLEMENTATION_VER_MAJOR 1
  VL53L0X PAL IMPLEMENTATION major version.
- #define VL53L0X_IMPLEMENTATION_VER_MINOR 0
  VL53L0X PAL IMPLEMENTATION minor version.
- #define VL53L0X_IMPLEMENTATION_VER_SUB 4
  VL53L0X PAL IMPLEMENTATION sub version.
- #define VL53L0X_IMPLEMENTATION_VER_REVISION 4960
  VL53L0X PAL IMPLEMENTATION sub version.
- #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) - 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) - 20)
- #define VL53L0X_ERROR_INVALID_COMMAND ((VL53L0X_Error) - 30)
- #define VL53L0X_ERROR_DIVISION_BY_ZERO ((VL53L0X_Error) - 40)
- #define VL53L0X_ERROR_REF_SPAD_INIT ((VL53L0X_Error) - 50)
- #define VL53L0X_ERROR_NOT_IMPLEMENTED ((VL53L0X_Error) - 99)
- #define VL53L0X_DEVICEMODE_SINGLE_RANGING ((VL53L0X_DeviceModes) 0)
- #define VL53L0X_DEVICEMODE_CONTINUOUS_RANGING ((VL53L0X_DeviceModes) 1)
- #define VL53L0X_DEVICEMODE_SINGLE_HISTOGRAM ((VL53L0X_DeviceModes) 2)
- #define VL53L0X_DEVICEMODE_CONTINUOUS_TIMED_RANGING ((VL53L0X_DeviceModes) 3)
- #define VL53L0X_DEVICEMODE_SINGLE_ALS ((VL53L0X_DeviceModes) 10)
- #define VL53L0X_DEVICEMODE_GPIO_DRIVE ((VL53L0X_DeviceModes) 20)

- #define VL53L0X_DEVICEMODE_GPIO_OSC ((VL53L0X_DeviceModes) 21)
- #define VL53L0X_HISTOGRAMMODE_DISABLED ((VL53L0X_HistogramModes) 0)
- #define VL53L0X_HISTOGRAMMODE_REFERENCE_ONLY ((VL53L0X_HistogramModes) 1)
- #define VL53L0X_HISTOGRAMMODE_RETURN_ONLY ((VL53L0X_HistogramModes) 2)
- #define VL53L0X_HISTOGRAMMODE_BOTH ((VL53L0X_HistogramModes) 3)
- #define VL53L0X_POWERMODE_STANDBY_LEVEL1 ((VL53L0X_PowerModes) 0)
- #define VL53L0X_POWERMODE_STANDBY_LEVEL2 ((VL53L0X_PowerModes) 1)
- #define VL53L0X_POWERMODE_IDLE_LEVEL1 ((VL53L0X_PowerModes) 2)
- #define VL53L0X_POWERMODE_IDLE_LEVEL2 ((VL53L0X_PowerModes) 3)
- #define VL53L0X_DMAX_LUT_SIZE 7
- #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)0)
- #define VL53L0X_INTERRUPTPOLARITY_HIGH ((VL53L0X_InterruptPolarity) 1)
- #define VL53L0X_VCSEL_PERIOD_PRE_RANGE ((VL53L0X_VcselPeriod) 0)
- #define VL53L0X_VCSEL_PERIOD_FINAL_RANGE ((VL53L0X_VcselPeriod) 1)
- #define VL53L0X_SEQUENCESTEP_TCC ((VL53L0X_VcselPeriod) 0)
- #define VL53L0X_SEQUENCESTEP_DSS ((VL53L0X_VcselPeriod) 1)
- #define VL53L0X_SEQUENCESTEP_MSRC ((VL53L0X_VcselPeriod) 2)
- #define VL53L0X_SEQUENCESTEP_PRE_RANGE ((VL53L0X_VcselPeriod) 3)
- #define VL53L0X_SEQUENCESTEP_FINAL_RANGE ((VL53L0X_VcselPeriod) 4)
- #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) 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) (FixPoint1616_t)(Value<<9)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT88(Value) (uint16_t)((Value>>8)&0xFFFF)
- #define VL53L0X_FIXPOINT88TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<8)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT412(Value) (uint16_t)((Value>>4)&0xFFFF)
- #define VL53L0X_FIXPOINT412TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<4)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT313(Value) (uint16_t)((Value>>3)&0xFFFF)
- #define VL53L0X_FIXPOINT313TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<3)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT08(Value) (uint8_t)((Value>>8)&0x00FF)
- #define VL53L0X_FIXPOINT08TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<8)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT53(Value) (uint8_t)((Value>>13)&0x00FF)
- #define VL53L0X_FIXPOINT53TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<13)
- #define VL53L0X_FIXPOINT1616TOFIXPOINT102(Value) (uint16_t)((Value>>14)&0x0FFF)
- #define VL53L0X_FIXPOINT102TOFIXPOINT1616(Value) (FixPoint1616_t)(Value<<12)
- #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.

---

vl53l0x_device.h File Reference

```
#include "vl53l0x_types.h"
```

Macros

- #define VL53L0X_DEVICEERROR_NONE ((VL53L0X_DeviceError) 0)
- #define VL53L0X_DEVICEERROR_VCSELCONTINUITYTESTFAILURE ((VL53L0X_DeviceError) 1)
- #define VL53L0X_DEVICEERROR_VCSELWATCHDOGTESTFAILURE ((VL53L0X_DeviceError) 2)
- #define VL53L0X_DEVICEERROR_NOVHVVALUEFOUND ((VL53L0X_DeviceError) 3)
- #define VL53L0X_DEVICEERROR_MSRCNOTARGET ((VL53L0X_DeviceError) 4)
- #define VL53L0X_DEVICEERROR_SNRCHECK ((VL53L0X_DeviceError) 5)
- #define VL53L0X_DEVICEERROR_RANGEPHASECHECK ((VL53L0X_DeviceError) 6)
- #define VL53L0X_DEVICEERROR_SIGMATHRESHOLDCHECK ((VL53L0X_DeviceError) 7)
- #define VL53L0X_DEVICEERROR_TCC ((VL53L0X_DeviceError) 8)
- #define VL53L0X_DEVICEERROR_PHASECONSISTENCY ((VL53L0X_DeviceError) 9)
- #define VL53L0X_DEVICEERROR_MINCLIP ((VL53L0X_DeviceError) 10)
- #define VL53L0X_DEVICEERROR_RANGECOMPLETE ((VL53L0X_DeviceError) 11)
- #define VL53L0X_DEVICEERROR_ALGOUNDERFLOW ((VL53L0X_DeviceError) 12)
- #define VL53L0X_DEVICEERROR_ALGOOVERFLOW ((VL53L0X_DeviceError) 13)
- #define VL53L0X_DEVICEERROR_RANGEIGNORETHRESHOLD ((VL53L0X_DeviceError) 14)
- #define VL53L0X_CHECKENABLE_SIGMA_FINAL_RANGE 0
- #define VL53L0X_CHECKENABLE_SIGNAL_RATE_FINAL_RANGE 1
- #define VL53L0X_CHECKENABLE_SIGNAL_REF_CLIP 2
- #define VL53L0X_CHECKENABLE_RANGE_IGNORE_THRESHOLD 3
- #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_LOW ((VL53L0X_GpioFunctionality) 1)
- #define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_HIGH ((VL53L0X_GpioFunctionality) 2)

- #define VL53L0X_GPIOFUNCTIONALITY_THRESHOLD_CROSSED_OUT ((VL53L0X_GpioFunctionality) 3)
- #define VL53L0X_GPIOFUNCTIONALITY_NEW_MEASURE_READY ((VL53L0X_GpioFunctionality) 4)
- #define VL53L0X_REG_SYSRANGE_START 0x000
- #define VL53L0X_REG_SYSRANGE_MODE_MASK 0x0F
  mask existing bit in VL53L0X_REG_SYSRANGE_START
- #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
- #define VL53L0X_REG_SYSRANGE_MODE_SINGLESHOT 0x00
  bit 1 write 0 in VL53L0X_REG_SYSRANGE_START set single shot mode
- #define VL53L0X_REG_SYSRANGE_MODE_BACKTOBACK 0x02
  bit 1 write 1 in VL53L0X_REG_SYSRANGE_START set back-to-back operation mode
- #define VL53L0X_REG_SYSRANGE_MODE_TIMED 0x04
  bit 2 write 1 in VL53L0X_REG_SYSRANGE_START set timed operation mode
- #define VL53L0X_REG_SYSRANGE_MODE_HISTOGRAM 0x08
  bit 3 write 1 in VL53L0X_REG_SYSRANGE_START set histogram operation 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 0x0004
- #define VL53L0X_REG_SYSTEM_INTERRUPT_CONFIG_GPIO 0x000A
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_DISABLED 0x00
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_LOW 0x01
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_LEVEL_HIGH 0x02
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_OUT_OF_WINDOW 0x03
- #define VL53L0X_REG_SYSTEM_INTERRUPT_GPIO_NEW_SAMPLE_READY 0x04
- #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_RTN 0x00BC
- #define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_RTN 0x00C0
- #define VL53L0X_REG_RESULT_CORE_AMBIENT_WINDOW_EVENTS_REF 0x00D0
- #define VL53L0X_REG_RESULT_CORE_RANGING_TOTAL_EVENTS_REF 0x00D4
- #define VL53L0X_REG_RESULT_PEAK_SIGNAL_RATE_REF 0x00B6
- #define VL53L0X_REG_ALGO_PART_TO_PART_RANGE_OFFSET_MM 0x0028
- #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 0x0056
- #define VL53L0X_REG_PRE_RANGE_CONFIG_VALID_PHASE_HIGH 0x0057
- #define VL53L0X_REG_PRE_RANGE_MIN_COUNT_RATE_RTN_LIMIT 0x0064
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_SNR 0X0067
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_LOW 0x0047
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_VALID_PHASE_HIGH 0x0048
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_MIN_COUNT_RATE_RTN_LIMIT 0x0044
- #define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_HI 0X0061
- #define VL53L0X_REG_PRE_RANGE_CONFIG_SIGMA_THRESH_LO 0X0062
- #define VL53L0X_REG_PRE_RANGE_CONFIG_VCSEL_PERIOD 0x0050
- #define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_HI 0x0051

- #define VL53L0X_REG_PRE_RANGE_CONFIG_TIMEOUT_MACROP_LO 0x0052
- #define VL53L0X_REG_SYSTEM_HISTOGRAM_BIN 0x0081
- #define VL53L0X_REG_HISTOGRAM_CONFIG_INITIAL_PHASE_SELECT 0x0033
- #define VL53L0X_REG_HISTOGRAM_CONFIG_READOUT_CTRL 0x0055
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_VCSEL_PERIOD 0x0070
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_HI 0x0071
- #define VL53L0X_REG_FINAL_RANGE_CONFIG_TIMEOUT_MACROP_LO 0x0072
- #define VL53L0X_REG_CROSSTALK_COMPENSATION_PEAK_RATE_MCPS 0x0020
- #define VL53L0X_REG_MSRC_CONFIG_TIMEOUT_MACROP 0x0046
- #define VL53L0X_REG_SOFT_RESET_GO2_SOFT_RESET_N 0x00bf
- #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_0 0x0B0
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_1 0x0B1
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_2 0x0B2
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_3 0x0B3
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_4 0x0B4
- #define VL53L0X_REG_GLOBAL_CONFIG_SPAD_ENABLES_REF_5 0x0B5
- #define VL53L0X_REG_GLOBAL_CONFIG_REF_EN_START_SELECT 0xB6
- #define VL53L0X_REG_DYNAMIC_SPAD_NUM_REQUESTED_REF_SPAD 0x4E /* 0x14E */
- #define VL53L0X_REG_DYNAMIC_SPAD_REF_EN_START_OFFSET 0x4F /* 0x14F */
- #define VL53L0X_REG_POWER_MANAGEMENT_GO1_POWER_FORCE 0x80
- #define VL53L0X_SPEED_OF_LIGHT_IN_AIR 2997
- #define VL53L0X_REG_VHV_CONFIG_PAD_SCL_SDA__EXTSUP_HV 0x0089
- #define VL53L0X_REG_ALGO_PHASECAL_LIM 0x0030 /* 0x130 */
- #define VL53L0X_REG_ALGO_PHASECAL_CONFIG_TIMEOUT 0x0030

Typedefs

- typedef uint8_t VL53L0X_DeviceError
- typedef uint8_t VL53L0X_GpioFunctionality

---

vl53l0x_doxydoc.c File Reference

---

vl53l0x_i2c_platform.h File Reference

```
#include "vl53l0x_def.h"
#include <stdint.h>
#include <stdarg.h>
```

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 .

Functions

- int32_t VL53L0X_comms_initialise (uint8_t comms_type, uint16_t comms_speed_khz)
  Initialise platform comms.
- int32_t VL53L0X_comms_close (void)
  Close platform comms.
- int32_t VL53L0X_cycle_power (void)
  Cycle Power to Device.
- 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.
- 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.
- int32_t VL53L0X_write_byte (uint8_t address, uint8_t index, uint8_t data)
  Writes a single byte to the device.
- int32_t VL53L0X_write_word (uint8_t address, uint8_t index, uint16_t data)
  Writes a single word (16-bit unsigned) to the device.
- int32_t VL53L0X_write_dword (uint8_t address, uint8_t index, uint32_t data)
  Writes a single dword (32-bit unsigned) to the device.
- int32_t VL53L0X_read_byte (uint8_t address, uint8_t index, uint8_t *pdata)
  Reads a single byte from the device.
- int32_t VL53L0X_read_word (uint8_t address, uint8_t index, uint16_t *pdata)
  Reads a single word (16-bit unsigned) from the device.
- int32_t VL53L0X_read_dword (uint8_t address, uint8_t index, uint32_t *pdata)
  Reads a single dword (32-bit unsigned) from the device.
- int32_t VL53L0X_platform_wait_us (int32_t wait_us)
  Implements a programmable wait in us.
- int32_t VL53L0X_wait_ms (int32_t wait_ms)
  Implements a programmable wait in ms.
- int32_t VL53L0X_set_gpio (uint8_t level)
  Set GPIO value.
- int32_t VL53L0X_get_gpio (uint8_t *plevel)
  Get GPIO value.
- int32_t VL53L0X_release_gpio (void)
  Release force on GPIO.
- int32_t VL53L0X_get_timer_frequency (int32_t *ptimer_freq_hz)
  Get the frequency of the timer used for ranging results time stamps.
- 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())

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	- uin16_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_interrupt_threshold_settings.h File Reference

Variables

- uint8_t InterruptThresholdSettings []

Variable Documentation

uint8_t InterruptThresholdSettings[]

Definition at line 39 of file vl53l0x_interrupt_threshold_settings.h.

---

vl53l0x_platform.h File Reference

Function prototype definitions for Ewok Platform layer.

```
#include "vl53l0x_def.h"
#include "vl53l0x_platform_log.h"
#include "vl53l0x_i2c_platform.h"
```

Data Structures

- struct VL53L0X_Dev_t

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

- #define PALDevDataGet(Dev, field) (Dev->Data.field)
  Get ST private structure VL53L0X_DevData_t data access.
- #define PALDevDataSet(Dev, field, data) (Dev->Data.field)=(data)
  Set ST private structure VL53L0X_DevData_t data field.

Typedefs

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

Functions

- VL53L0X_Error VL53L0X_LockSequenceAccess (VL53L0X_DEV Dev)
  Lock comms interface to serialize all commands to a shared I2C interface for a specific device.
- VL53L0X_Error VL53L0X_UnlockSequenceAccess (VL53L0X_DEV Dev)
  Unlock comms interface to serialize all commands to a shared I2C interface for a specific device.
- VL53L0X_Error VL53L0X_WriteMulti (VL53L0X_DEV Dev, uint8_t index, uint8_t *pdata, uint32_t count)
  Writes the supplied byte buffer to the device.
- 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.
- VL53L0X_Error VL53L0X_WrByte (VL53L0X_DEV Dev, uint8_t index, uint8_t data)
  Write single byte register.
- VL53L0X_Error VL53L0X_WrWord (VL53L0X_DEV Dev, uint8_t index, uint16_t data)
  Write word register.
- VL53L0X_Error VL53L0X_WrDWord (VL53L0X_DEV Dev, uint8_t index, uint32_t data)
  Write double word (4 byte) register.
- VL53L0X_Error VL53L0X_RdByte (VL53L0X_DEV Dev, uint8_t index, uint8_t *data)
  Read single byte register.
- VL53L0X_Error VL53L0X_RdWord (VL53L0X_DEV Dev, uint8_t index, uint16_t *data)
  Read word (2byte) register.
- VL53L0X_Error VL53L0X_RdDWord (VL53L0X_DEV Dev, uint8_t index, uint32_t *data)
  Read dword (4byte) register.
- VL53L0X_Error VL53L0X_UpdateByte (VL53L0X_DEV Dev, uint8_t index, uint8_t AndData, uint8_t OrData)
  Threat safe Update (read/modify/write) single byte register.
- VL53L0X_Error VL53L0X_PollingDelay (VL53L0X_DEV Dev)
  execute delay in all polling API call

Detailed Description

Function prototype definitions for Ewok Platform layer.

All end user OS/platform/application porting.

Definition in file vl53l0x_platform.h.

---

vl53l0x_platform_log.h File Reference

platform log function definition

```
#include <stdio.h>
#include <string.h>
```

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.

vl53l0x_tuning.h File Reference

```
#include "vl53l0x_def.h"
```

Variables

- uint8_t DefaultTuningSettings []

Variable Documentation

uint8_t DefaultTuningSettings[]

Definition at line 41 of file vl53l0x_tuning.h.

vl53l0x_types.h File Reference

VL53L0X types definition.

```
#include <stdint.h>
#include <stddef.h>
```

Typedefs

- typedef uint32_t FixPoint1616_t
  use where fractional values are expected

- typedef unsigned long long uint64_t
- typedef unsigned int uint32_t
  Typedef defining 32 bit unsigned int type.
- typedef int int32_t
  Typedef defining 32 bit int type.
- typedef unsigned short uint16_t
  Typedef defining 16 bit unsigned short type.
- typedef short int16_t
  Typedef defining 16 bit short type.
- typedef unsigned char uint8_t
  Typedef defining 8 bit unsigned char type.
- typedef signed char int8_t
  Typedef defining 8 bit char type.

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.

Index