OPTIGA™ Trust X
Datasheet

Key Features
- High-end security controller
- Turnkey solution
- Mutual authentication using ECDSA
- DTLS client IETF standard RFC 6347
- Secure communication using DTLS
- Compliant with the USB Type-C™ Authentication standard
- I2C interface
- Up to 10 kB user memory
- Cryptographic support: ECC NIST P256 and P384, AES-128 (via DTLS client), SHA-256, TRNG, DRNG
- PG-USON-10-2 package (3 x 3 mm)
- Standard & extended temperature ranges
- Full system integration support with Host Software Library
- Common Criteria Certified EAL6+ (high) hardware
- Crypto ToolBox with ECC NIST P256, P384, SHA-256 (sign, verify, key generation, ECDH, key derivation)
- Device Security Monitor
- Lifetime for Industrial Automation and Infrastructure is 20 years and 15 years for other Application Profiles

Benefits
- Protection of IP and data
- Protection of business case
- Protection of corporate image
- Safeguarding of quality and safety

Applications
- Industrial control and automation
- Consumer electronics and Smart home
- Medical devices

About this document
Scope and purpose
This Datasheet provides information to enable integration of a security device, and includes package, connectivity and technical data.

Intended audience
This Datasheet is intended for device integrators and board manufacturers.
OPTIGA™ Trust X
Datasheet

Introduction

About this document ........................................................................................................1
1 Introduction ...................................................................................................................3
  1.1 Broad range of benefits .........................................................................................3
  1.2 Enhanced security .................................................................................................3
  1.3 Fast and easy integration .......................................................................................3
  1.4 Applications ..........................................................................................................3
  1.5 Device Features ......................................................................................................3
2 System Block Diagram ...............................................................................................6
3 Interface and Schematics ............................................................................................8
  3.1 System Integration Schematics ..............................................................................8
4 Description of packages .............................................................................................9
  4.1 PG-USON-10-2 ....................................................................................................9
  4.2 Production sample marking pattern .....................................................................10
5 Technical Data ............................................................................................................12
  5.1 I2C Interface Characteristics ..............................................................................12
    5.1.1 I2C Standard/Fast Mode Interface Characteristics ........................................12
    5.1.2 I2C Fast Mode Plus Interface Characteristics ............................................13
    5.1.3 Electrical Characteristics ..............................................................................14
    5.1.4 DC Electrical Characteristics ......................................................................14
    5.1.5 AC Electrical Characteristics ......................................................................14
    5.1.6 Start-Up of I2C Interface ..............................................................................15
    5.1.6.1 Startup after Power-On ............................................................................15
    5.1.6.2 Startup for Warm Resets ........................................................................16
6 Connecting to Host .....................................................................................................18
  6.1 OPTIGA™ Trust X Host Software Architecture ....................................................18
  6.2 Release Package Folder Structure .....................................................................18
  6.3 Host Software Folder Structure .........................................................................19
  6.4 Porting Notes ......................................................................................................21
  6.5 Communication with OPTIGA™ Trust X ............................................................21
  6.6 Reference code on XMC4500 for communicating with OPTIGA™ Trust X ........23
7 OPTIGA™ Trust X Commands ..............................................................................26
8 Security Monitor ......................................................................................................27
  8.1 Security Events ...................................................................................................27
  8.2 Security Policy ....................................................................................................27
9 RoHS Compliance ....................................................................................................28
10 Appendix A – Infineon I2C Protocol Registry Map ...............................................29
  10.1 IFX I2C Protocol Variations ..............................................................................31
11 Appendix B – Power Management .........................................................................33
  11.1 Low Power Sleep Mode .....................................................................................33

Revision history ............................................................................................................34
1 Introduction

As embedded systems (e.g. IoT devices) are increasingly gaining the attention of attackers, Infineon offers the OPTIGA™ Trust X as a turnkey security solution for industrial automation systems, smart homes, consumer devices and medical devices. This high-end security controller comes with full system integration support for easy and cost-effective deployment of high-end security for your assets.

1.1 Broad range of benefits

Integrated into your device, the OPTIGA™ Trust X supports protection of your brand and business case, differentiates your product from your competitors, and adds value to your product, making it stronger against cyberattacks.

1.2 Enhanced security

The OPTIGA™ Trust X is based on advanced security controller with built-in tamper proof NVM for secure storage and Symmetric/Asymmetric crypto engine to support ECC 256, AES-128 and SHA-256. This new security technology greatly enhances your overall system security.

1.3 Fast and easy integration

The turnkey setup – with full system integration and all key/certificate material preprogrammed – reduces your efforts for design, integration and deployment to a minimum. As a turnkey solution, the OPTIGA™ Trust X comes with preprogrammed OS/Application code locked and with host-side modules to integrate with host micro controller software. The extended temperature range of −40°C to +105°C combined with a standardized I2C interface and the small PG-USON-10-2 footprint will facilitate onboarding in your existing ecosystem. Almost 30 years in a market-leading position with nearly 20 billion security controllers shipped worldwide are the result of Infineon's strong expertise and its commitment to make security a success factor for you.

1.4 Applications

The OPTIGA™ Trust X covers a broad range of use cases necessary for many types of applications that include the following:

a) Network node protection such as TLS or DTLS
b) Protect the Authenticity, Integrity and Confidentiality of your product, data and IP
c) Mutual Authentication
d) Secure Communication
e) Datastore Protection
f) Lifecycle Management
g) Platform Integrity Protection
h) Secure Updates

1.5 Device Features

The OPTIGA™ Trust X comes with upto 10kB user memory that can be used to store X.509 certificates. OPTIGA™ Trust X is based on Common Criteria Certified EAL6+ (high) hardware enabling it to prevent physical attacks on the device itself and providing high assurance that the keys or arbitrary data stored cannot be accessed by an unauthorized entity. OPTIGA™ Trust X supports a highspeed I2C communication interface of up to 1MHz (FM+).
Introduction

Table 1

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Temperature range</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td>OPTIGA™ Trust X</td>
<td>Embedded security solution for connected devices</td>
<td>−25°C to +85°C Standard Temperature Range (STR)</td>
<td>PG-USON-10-2</td>
</tr>
<tr>
<td>SLS 32AIA020X4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OPTIGA™ Trust X</td>
<td>Embedded security solution for connected devices</td>
<td>−40°C to +105°C Extended Temperature Range (ETR)</td>
<td>PG-USON-10-2</td>
</tr>
<tr>
<td>SLS 32AIA020X2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Evaluation Kit</td>
<td>Includes host micro controller connected to OPTIGA™ Trust X with USB/Ethernet adapters to connect to external world which enables you to evaluate OPTIGA™ Trust X features and start the Design-In activity</td>
<td></td>
<td>Board</td>
</tr>
</tbody>
</table>

Infineon and its distribution partners offer a wide range of customization options (e.g. X.509 certificate generation and key provisioning) for the security chip.

Table 2

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>AES</td>
<td>Advanced Encryption Standard</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>AUTH</td>
<td>Authentication</td>
</tr>
<tr>
<td>CA</td>
<td>Certification Authority</td>
</tr>
<tr>
<td>DTLS</td>
<td>Datagram Transport Layer Security</td>
</tr>
<tr>
<td>DRNG</td>
<td>Deterministic Random Number Generator</td>
</tr>
<tr>
<td>EAL</td>
<td>Evaluation Assurance Level</td>
</tr>
<tr>
<td>ECC</td>
<td>Elliptic Curve Cryptography</td>
</tr>
<tr>
<td>ECDH</td>
<td>Elliptic Curve Diffie Hellman</td>
</tr>
<tr>
<td>ECDSA</td>
<td>Elliptic Curve Digital Signature Algorithm</td>
</tr>
<tr>
<td>ETR</td>
<td>Extended Temperature Range</td>
</tr>
<tr>
<td>IETF</td>
<td>Internet Engineering Task Force</td>
</tr>
<tr>
<td>IOT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>IP</td>
<td>Intellectual Property</td>
</tr>
<tr>
<td>I2C</td>
<td>Inter-Integrated Circuit</td>
</tr>
<tr>
<td>NIST</td>
<td>National Institute of Standards and Technology</td>
</tr>
<tr>
<td>OCP</td>
<td>OPTIGA™ Crypto and Protected Communication</td>
</tr>
<tr>
<td>OS</td>
<td>Operating System</td>
</tr>
<tr>
<td>PAL</td>
<td>Platform Abstraction Layer</td>
</tr>
<tr>
<td>PKI</td>
<td>Public Key Infrastructure</td>
</tr>
<tr>
<td>RFC</td>
<td>Request For Comments</td>
</tr>
<tr>
<td>TLS</td>
<td>Transport Layer Security</td>
</tr>
</tbody>
</table>
## Introduction

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRNG</td>
<td>True Random Number Generator</td>
</tr>
<tr>
<td>SHA</td>
<td>Secure Hash Algorithm</td>
</tr>
<tr>
<td>SKU</td>
<td>Stock Keeping Unit</td>
</tr>
<tr>
<td>STR</td>
<td>Standard Temperature Range</td>
</tr>
<tr>
<td>USB</td>
<td>Universal Synchronous Bus</td>
</tr>
</tbody>
</table>
2 System Block Diagram

The following figure depicts the system block diagram for OPTIGA™ Trust X.

![System Block Diagram](image)

The System Block Diagram is explained below for each layer.

1. Local Host
   - Application – This is the target application which utilizes OPTIGA™ Trust X for its security needs
   - DTLS – DTLS client aka. OCP Library provides APIs for performing Mutual Authentication and Encrypted Communication using OPTIGA™ Trust X
   - AUTH – Authentication aka. Integration Library provides APIs for performing One Way Authentication for Brand Protection and IP Protection using OPTIGA™ Trust X
   - Command Library – Provides APIs to send and receive commands to and from OPTIGA™ Trust X. Any TLS stack can be integrated to offload crypto operations to OPTIGA™ Trust X via this Command Library.
   - Crypto Lib Wrapper – Provides wrapper APIs for Third Party crypto library, mainly used in One Way Authentication
   - Crypto Library – External cryptographic software which is used for One Way Authentication
   - OPTIGA Comms – Provides wrapper APIs for communication with OPTIGA™ Trust X
   - Infineon I2C Protocol – Infineon protocol over I2C (IFX I2C) to communicate with OPTIGA™ Trust X
   - PAL – A layer that abstracts platform specific drivers (e.g. i2c, timer, gpio, sockets etc.)

2. OPTIGA™ Trust X
   - Arbitrary Data Objects – The target application can store upto 4.5kB (~4600 bytes) of data into OPTIGA™ Trust X
   - X.509 – Upto 4, X.509 based Certificates can be stored into OPTIGA™ Trust X
   - Keys – Upto 4, ECC based keys can be stored into OPTIGA™ Trust X
   - Mutual Authentication Trust Anchor – Customer PKI domain Trust Anchor for Mutual Authentication (TLS/DTLS) can be stored into OPTIGA™ Trust X
OPTIGA™ Trust X

Datasheet

System Block Diagram

- Firmware Update Trust Anchor – Customer PKI domain Trust Anchor for Firmware Updates can be stored into OPTIGA™ Trust X
- Crypto Functions - OPTIGA™ Trust X provides cryptographic functions and protocols that can be invoked via local host

**Note:** *Unique ECC private keys and X.509 Certificates – During production at Infineon fab, unique asymmetric keys (private and public) are generated. The public key is signed by customer specific CA and resulting X.509 certificate issued is securely stored on OPTIGA™ Trust X. Special measures are taken to prevent leakage and modification of private key at the Common Criteria Certified production site*
3 Interface and Schematics

This section explains the schematics of the product and gives some recommendations as to how the controller should be externally connected.

3.1 System Integration Schematics

Figure 1 illustrates how to integrate OPTIGA™ Trust X to your local host.

![System Integration Schematic Diagram](image)

**Figure 2** System Integration Schematic Diagram

**Note:** Value of the pullup resistors depends on the target application circuit and the targeted I2C frequency.
4 Description of packages

This chapter provides information on the package types and how the interfaces of each product are assigned to the package pins. For further information on compliance of the packages with European Parliament Directives, see "RoHS Compliance" on Page 28.

For details and recommendations regarding the assembly of packages on PCBs, please see the following: http://www.infineon.com/cms/en/product/technology/packages/

4.1 PG-USON-10-2

The package dimensions (in mm) of the controller in PG-USON-10-2 packages are given below.

![PG-USON-10-2 Package Outline](image)

Figure 3 PG-USON-10-2 Package Outline

The following figure shows the footprint of the PG-USON-10-2 package:

![PG-USON-10-2 Package Footprint](image)

Figure 4 PG-USON-10-2 Package Footprint

The figure below shows the PG-USON-10-2 in top view:
Figure 5  PG-USON-10-2 top view

4.2 Production sample marking pattern

The following figure describes the productive sample marking pattern on PG-USON-10-2.

Figure 6  PG-USON-10-2 sample marking pattern

The black dot indicates pin 01 for the chip. The following table describes the sample marking pattern:

<table>
<thead>
<tr>
<th>Table 3</th>
<th>Marking table for PG-USON-10-2 Packages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indicator</td>
<td>Description</td>
</tr>
<tr>
<td>LOT CODE</td>
<td>Defined and inserted during fabrication</td>
</tr>
<tr>
<td>ZZ</td>
<td>Indicates the Certifying Authority Serial Number / SKU#, e.g. &quot;oo&quot; would mean &quot;SKU#0&quot;</td>
</tr>
</tbody>
</table>
| H/E     | H = "Halogen-free", E = "Engineering samples"  
This indicator is followed by "YYWW", where YY is the "Year" and WW is the "Work Week" of the production. This is inserted during fabrication. Engineering samples have "E YYWW" and productive samples have "H YYWW" |
### Description of packages

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Description</th>
</tr>
</thead>
</table>
| 12345     | Convention: T&#$@ where:  
- The letter "T" indicates the OPTIGA Trust family  
- & indicates whether the product is a Trust X or Trust E controller  
- # indicates whether the controller is an ETR (E) or STR (S) variant  
- $ specifies the OPTIGA™ Trust X/E release version number  
- @ specifies the software version  
Example: "TXE10" means 'OPTIGA™ Trust X', 'ETR variant', 'release version 1', 'software version 0' |

The contacts and their functionality are given in the table below.

### Table 4  Contact Definitions and Functions of PG-USON-10-2 Packages

<table>
<thead>
<tr>
<th>Pin</th>
<th>Type</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>GND</td>
<td>Supply voltage (Ground)</td>
</tr>
<tr>
<td>02</td>
<td>NC</td>
<td>Not connected</td>
</tr>
<tr>
<td>03</td>
<td>I/O</td>
<td>Serial Data Line (SDA)</td>
</tr>
<tr>
<td>04</td>
<td>NC</td>
<td>Not connected</td>
</tr>
<tr>
<td>05</td>
<td>NC</td>
<td>Not connected</td>
</tr>
<tr>
<td>06</td>
<td>NC</td>
<td>Not connected</td>
</tr>
<tr>
<td>07</td>
<td>NC</td>
<td>Not connected</td>
</tr>
<tr>
<td>08</td>
<td>I/O</td>
<td>Serial Clock Line (SCL)</td>
</tr>
<tr>
<td>09</td>
<td>IN</td>
<td>Active Low Reset (RST)</td>
</tr>
<tr>
<td>10</td>
<td>PWR</td>
<td>Supply voltage (VCC)</td>
</tr>
</tbody>
</table>
5 Technical Data
This section summarizes the technical data of the product. It provides the operational characteristics as well as the electrical DC and AC characteristics.

5.1 I2C Interface Characteristics

Table 5 I2C Operation Supply and Input Voltages

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note or Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply voltage</td>
<td>$V_{CC,I2C}$</td>
<td>1.62 – 5.5</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>SDA, SCL input voltage</td>
<td>$V_{IN,I2C}$</td>
<td>0.3 – $V_{CC,I2C}$ + 0.5</td>
<td>V</td>
<td>$V_{CC,I2C}$ is in the operational supply range</td>
</tr>
<tr>
<td></td>
<td></td>
<td>−0.3 – 5.5</td>
<td>V</td>
<td>$V_{CC,I2C}$ is switched off</td>
</tr>
</tbody>
</table>

1) Whichever is lower

5.1.1 I2C Standard/Fast Mode Interface Characteristics

For operation of the I2C interface, the electrical characteristics are compliant with the I2C bus specification Rev. 4 for “standard-mode” ($f_{SCL}$ up to 100 kHz) and “fast-mode” ($f_{SCL}$ up to 400 kHz), with certain deviations as stated in the table below.

**Note:** $T_A$ as given for the operating temperature range of the controller unless otherwise stated.

Table 6 I2C Standard Mode Interface Characteristics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note or Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCL clock frequency</td>
<td>$f_{SCL}$</td>
<td>0 – 100 kHz</td>
<td>kHz</td>
<td></td>
</tr>
<tr>
<td>Input low-level</td>
<td>$V_{IL}$</td>
<td>−0.3 – 0.3 * $V_{CC,I2C}$</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>Low-level output voltage</td>
<td>$V_{OL1}$</td>
<td>0 – 0.4</td>
<td>V</td>
<td>Sink current 3 mA; $V_{CC,I2C}$ ≥ 2.7 V; Sink current 2 mA; $V_{CC,I2C}$ &lt; 2.7 V</td>
</tr>
<tr>
<td>Low-level output current</td>
<td>$I_{OL}$</td>
<td>3 – –</td>
<td>mA</td>
<td>$V_{OL} = 0.4 V$; $V_{CC,I2C}$ ≥ 2.7 V; $V_{OL} = 0.4 V$; $V_{CC,I2C}$ &lt; 2.7 V</td>
</tr>
<tr>
<td>Output fall time from $V_{ILmin}$ to $V_{ILmax}$ (at device pin)</td>
<td>$t_{OF}$</td>
<td>– – 250 ns</td>
<td>ns</td>
<td>$C_b ≤ 400 pF$; $V_{CC,I2C}$ ≥ 2.7 V; $C_b ≤ 200 pF$; $V_{CC,I2C}$ &lt; 2.7 V</td>
</tr>
<tr>
<td>Capacitive load for each bus line</td>
<td>$C_b$</td>
<td>– – 400 200 pF</td>
<td>pF</td>
<td>$V_{CC,I2C}$ ≥ 2.7 V; $V_{CC,I2C}$ &lt; 2.7 V</td>
</tr>
</tbody>
</table>
# I2C Fast Mode Interface Characteristics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCL clock frequency</td>
<td>$f_{\text{SCL}}$</td>
<td>0</td>
<td>–</td>
<td>400</td>
<td>kHz</td>
</tr>
<tr>
<td>Input low-level</td>
<td>$V_L$</td>
<td>–0.3</td>
<td>–</td>
<td>0.3 * $V_{\text{CC,I2C}}$</td>
<td>V</td>
</tr>
<tr>
<td>Low-level output voltage</td>
<td>$V_{\text{OL1}}$</td>
<td>0</td>
<td>–</td>
<td>0.4</td>
<td>V</td>
</tr>
<tr>
<td>Low-level output current</td>
<td>$I_{\text{OL}}$</td>
<td>3</td>
<td>2</td>
<td>–</td>
<td>mA</td>
</tr>
<tr>
<td>Output fall time from $V_{\text{ILmin}}$ to $V_{\text{ILmax}}$ (at device pin)</td>
<td>$t_{\text{OF}}$</td>
<td>20 * $V_{\text{CC,I2C}}$ / 5.5 V</td>
<td>–</td>
<td>250</td>
<td>ns</td>
</tr>
<tr>
<td>Capacitive load for each bus line</td>
<td>$C_b$</td>
<td>15^2</td>
<td>–</td>
<td>400</td>
<td>pF</td>
</tr>
</tbody>
</table>

1) A min. capacitive load is necessary to reach $t_{\text{OF}}$
2) A min. capacitive load is necessary to reach $t_{\text{fmin}}$

## 5.1.2 I2C Fast Mode Plus Interface Characteristics

For operation of the I2C interface, the electrical characteristics are compliant with the I2C bus specification Rev. 4 for “fast mode plus” ($f_{\text{SCL}}$ up to 1 MHz), with certain deviations as stated in the table below.

Note: $T_A$ as given for the operating temperature range of the controller unless otherwise stated.

### Table 8 I2C Fast Mode Plus Interface Characteristics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Min.</th>
<th>Typ.</th>
<th>Max.</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCL clock frequency</td>
<td>$f_{\text{SCL}}$</td>
<td>0</td>
<td>–</td>
<td>1000</td>
<td>kHz</td>
</tr>
<tr>
<td>Input low-level</td>
<td>$V_L$</td>
<td>–0.3</td>
<td>–</td>
<td>0.3 * $V_{\text{CC,I2C}}$</td>
<td>V</td>
</tr>
<tr>
<td>Low-level output voltage</td>
<td>$V_{\text{OL1}}$</td>
<td>0</td>
<td>–</td>
<td>0.4</td>
<td>V</td>
</tr>
<tr>
<td>Low-level output current</td>
<td>$I_{\text{OL}}$</td>
<td>3</td>
<td>2</td>
<td>–</td>
<td>mA</td>
</tr>
<tr>
<td>Output fall time from $V_{\text{ILmin}}$ to $V_{\text{ILmax}}$ (at device pin)</td>
<td>$t_{\text{OF}}$</td>
<td>20 * $V_{\text{CC,I2C}}$ / 5.5 V</td>
<td>–</td>
<td>120</td>
<td>ns</td>
</tr>
</tbody>
</table>

Note: $C_b \leq 150$ pF
### 5.1.3 Electrical Characteristics

**Note:** $T_A$ as given for the operating temperature range of the controller unless otherwise stated. All currents flowing into the controller are considered positive.

### 5.1.4 DC Electrical Characteristics

$T_A$ as given for the controller’s operating ambient temperature range unless otherwise stated. All currents flowing into the controller are considered positive.

#### Table 9 Electrical Characteristics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note or Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supply voltage</td>
<td>$V_{CC}$</td>
<td>1.62 – 5.5</td>
<td>V</td>
<td>Overall functional range</td>
</tr>
<tr>
<td></td>
<td>$V_{CC,I2C}$</td>
<td>1.62 – 5.5</td>
<td>V</td>
<td>Supply voltage range for operation of I2C</td>
</tr>
<tr>
<td>Supply current</td>
<td>$I_{CCAVG}$</td>
<td>– 20.0</td>
<td>mA</td>
<td>While running a typical authentication profile $T_A = 25^\circ C$; $V_{CC} = 5.0$ V</td>
</tr>
<tr>
<td></td>
<td>Supply current, in sleep mode</td>
<td>$I_{CCS3}$</td>
<td>– 70 100</td>
<td>$\mu$A</td>
</tr>
<tr>
<td>RST input low voltage</td>
<td>$V_{IL}$</td>
<td>– 0.3</td>
<td>V</td>
<td>$I_{IL} = -50 \mu A$ to $+20 \mu A$</td>
</tr>
<tr>
<td>RST input high voltage</td>
<td>$V_{IH}$</td>
<td>0.7 * $V_{CC}$</td>
<td>$V_{CC} + 0.3$</td>
<td>$I_{IL} = -50 \mu A$ to $+20 \mu A$</td>
</tr>
</tbody>
</table>

1) Supply current can be limited from 6mA to 15mA by software commands.

### 5.1.5 AC Electrical Characteristics

$T_A$ as given for the controller’s operating ambient temperature range unless otherwise stated. All currents flowing into the controller are considered positive.

#### Table 10 AC Characteristics

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note or Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$V_{CC}$ rampup time</td>
<td>$t_{VCCR}$</td>
<td>1 – 1000</td>
<td>$\mu$s</td>
<td>400 mV to 90% of $V_{CC}$ target voltage ramp</td>
</tr>
</tbody>
</table>
The $V_{CC}$ ramp is depicted in Figure 7. 90% of the target supply voltage must be reached within $t_{VCCR}$ after it has exceeded 400 mV. Moreover, its variation must be kept within a ±10% range.

![Figure 7: Vcc Rampup](image)

### 5.1.6 Start-Up of I2C Interface

There are 2 variants possible for performing the startup procedure:

- Startup after power-on
- Startup for warm resets

#### 5.1.6.1 Startup after Power-On

The activation of the I2C interface after power-on needs the following reset procedure.

- VCC is powered up and the state of the SDA and SCL line are set to high level during power-up
- The first transmission may start at the earliest $t_{STARTUP}$ after power-up of the device

The following figure shows the startup timing of the I2C interface for this case.
5.1.6.2 Startup for Warm Resets

When using the reset signal for triggering a warm reset after power-on, the activation of the I2C interface needs the following reset procedure:

- VCC remains powered up.
- The terminal stops I2C communication. SDA and SCL lines are set to high level before RST is set to low level.
- After its falling edge, RST has to be kept at low level for at least $t_1$. At the latest $t_2$ after the falling edge of RST, the terminal must set RST to high level.
- The first transmission may start at the earliest $t_{\text{STARTUP}}$ after the rising edge of RST

The following figure shows the timing for this startup case.
### Figure 9  
Startup of I2C Interface for Warm Resets

*Note: If NVM programming was requested prior to the reset, $t_{\text{STARTUP}}$ will be extended from a typical value of 10 ms to a maximum of 12 ms.*

### Table 12  
Startup of I2C Interface for Warm Resets

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Values</th>
<th>Unit</th>
<th>Note or Test Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Startup time</td>
<td>$t_{\text{STARTUP}}$</td>
<td>10</td>
<td>Typ.</td>
<td>Max. ms</td>
</tr>
<tr>
<td>Rise time</td>
<td>$t_R$</td>
<td>1</td>
<td>μs</td>
<td>From 10% to 90% of signal amplitude</td>
</tr>
<tr>
<td>Fall time</td>
<td>$t_F$</td>
<td>1</td>
<td>μs</td>
<td>From 10% to 90% of signal amplitude</td>
</tr>
<tr>
<td>Reset detection</td>
<td>$t_1$</td>
<td>10</td>
<td>μs</td>
<td></td>
</tr>
<tr>
<td>Reset low</td>
<td></td>
<td>10</td>
<td>2500</td>
<td>μs</td>
</tr>
</tbody>
</table>

1) Reset triggered by software (without power off/on cycle)
6 Connecting to Host

6.1 OPTIGA™ Trust X Host Software Architecture

In Figure 1 the System Block Diagram was explained which covered the OPTIGA™ Trust X Host Library layers. In following sections, we will cover how to communicate with OPTIGA™ Trust X using I2C.

![OPTIGA™ Trust X Host Software Architecture](image10)

6.2 Release Package Folder Structure

The following figure shows the release package structure when OPTIGA™ Trust X is installed/extracted on PC.

![Release Package Folder Structure](image11)
1. `<INSTALLDIR>` is the root directory to which the release contents are installed or extracted. The content of each subdirectory under installed directory `<INSTALLDIR>` is explained below.

2. CA Certificates
   This directory contains OPTIGA™ Trust X Test and Productive Trust-Anchor/CA certificates.

3. DemoUI
   This directory contains binaries and Demo UI Application for OPTIGA™ Trust X.

4. Documentation
   This directory contains all common OPTIGA™ Trust X documentation.

5. Host
   This directory contains source files, header files, binaries, documents, API as compiled help (CHM) and sample application for OPTIGA™ Trust X Host Software.

6. PC
   This directory contains source files, header files, binaries and sample application for OPTIGA™ Trust X PC Software.

7. TestServer
   This directory contains Sample Test Server Application and Test certificates required for DTLS client feature demonstration.

### 6.3 Host Software Folder Structure

The following figure shows the Host Software folder structure when OPTIGA™ Trust X is installed on PC.

![Host Software Folder Structure](image)

**Figure 12** Host Software Folder Structure

1. Bin
   This directory contains prebuilt binaries for Eval Kit based on XMC4500 Relax Kit v1 that communicates with OPTIGA™ Trust X.

2. Documentation
Connecting to Host

This directory contains documentation outlining software for Eval Kit based on XMC4500 Relax Kit v1.

3. Projects

This directory contains project files for Eval Kit based on XMC4500 Relax Kit v1.

4. Source

This directory contains all source files for OPTIGA™ Trust X Host Software Library.

Further the following figure elaborates the Host Software source folder structure.

![Host Source Folder Structure](image)

Figure 13  Host Source Folder Structure

1. auth – This folder contains sources for One Way Authentication which are platform independent. The layer is also known as Integration Library.
2. cmd – This folder contains sources for all OPTIGA™ Trust X commands which are platform independent.
3. common – This folder contains sources that are common for all functionality (e.g. utilities).
4. cryptolib – This folder contains binaries for crypto library wrapper which is platform independent.
5. dtls – This folder contains sources for Mutual Authentication and Encrypted Communication using DTLS client, which are platform independent. The layer is also known as OCP Library.
6. ifx_i2c – This folder contains sources for Infineon protocol over I2C (aka IFX I2C).
7. include – This folder contains header files for all Host Software.
8. pal – This folder contains all the platform dependent code.
9. transparent_channel – This folder contains transparent channel communication mainly used for Eval Kit.

6.4 Porting Notes

The Platform Abstraction Layer (PAL) APIs have to be updated to integrate the OPTIGA™ Trust X host libraries in the local host target platform.

The PAL reference code for the XMC4500 Relax kit is provided as part of package which can be used. The implementation can be referred in "<INSTALLDIR>/Host/Source/pal/xmc4500" and the header files are available in "<INSTALLDIR>/Host/Source/Include" with the required APIs used by upper layers. The header files are platform agnostic and would not require any change.

6.5 Communication with OPTIGA™ Trust X

The hardware/platform resource configuration with respect to I2C master and GPIOs (Vdd and Reset) are to be updated in pal_ifx_i2c_config.c. These configurations are used by the IFX I2C implementation to communicate with OPTIGA™ Trust X.

1. Update I2C master platform specific context[e.g. (void*)&i2c_master_0]

2. Update platform specific context for GPIOs (Vdd and Reset) [e.g. (void*)&pin_3_4]

3. Update PAL I2C APIs [pal_i2c.c] to communicate with OPTIGA™ Trust X
Connecting to Host

The pal_i2c is expected to provide the APIs for I2C driver initialization, de-initialization, read, write and set bitrate kind of operations

a) **pal_i2c_init**
b) **pal_i2c_deinit**
c) **pal_i2c_read**
d) **pal_i2c_write**
e) **pal_i2c_set_bitrate**

In few target platforms, the I2C master driver initialization (**pal_i2c_init**) is done during the platform start up. In such an environment, there is no need to implement **pal_i2c_init** and **pal_i2c_deinit** functions. Otherwise, these (**pal_i2c_init** & **pal_i2c_deinit**) functions must be implemented as per the upper layer expectations based on the need. The details of these expectations are available in the Host library API documentation (chm).

The reference implementation of PAL I2C based on XMC4500 Relax kit does not need to have the platform I2C driver initialization explicitly done as part of **pal_i2c_init** as it is taken care by the DAVE library initialization. Hence **pal_i2c_init** & **pal_i2c_deinit** are not implemented.

In addition to the above specified APIs, the PAL I2C must handle the events from the low level I2C driver and invoke the upper layer handlers registered with PAL I2C context for the respective transaction as shown in the below example.

```c
//I2C driver callback function when the transmit is completed successfully
void i2c_master_end_of_transmit_callback(void)
{
    invoke_upper_layer_callback(gp_pal_i2c_current_ctx,
        (uint8_t)PAL_I2C_EVENT_TX_SUCCESS);
}
```

In above example the I2C driver callback, when transmit is successful invokes the handler to inform the result.

4. Update PAL GPIO [**pal_gpio.c**] to power on and reset the OPTIGA™ Trust X
   a) **pal_gpio_set_high**
   b) **pal_gpio_set_low**

5. Update PAL Timer [**pal_os_timer.c**] to enable timer
   a) **pal_os_timer_get_time_inMilliseconds**
   b) **pal_os_timer_delay_inMilliseconds**

6. Update Event management for the asynchronous interactions for IFX I2C [**pal_os_event.c**]
   a) **pal_os_event_register_callback_oneshot**
   b) **scheduler_timer_isr**

The **pal_os_event_register_callback_oneshot** function is expected to register the handler and context provided as part of input parameters and triggers the timer for the requested time.

```c
void pal_os_event_register_callback_oneshot(
    register_callback callback,
    void* callback_args,
    uint32_t time_us)
{
    callback_registered = callback;
    callback_ctx = callback_args;

    //lint --e{534} suppress "Return value is not required to be checked"
    TIMER_SetTimeInterval(&scheduler_timer, (time_us*100));
}
```
And the handler registered must be invoked once the timer is elapsed as shown in `scheduler_timer_isr`

```c
void scheduler_timer_isr(void)
{
    TIMER_ClearEvent(&scheduler_timer);
    //lint --e{534} suppress "Return value is not required to be checked"
    TIMER_Stop(&scheduler_timer);
    TIMER_Clear(&scheduler_timer);

    if (callback_registered)
    {
        callback_registered((void*)callback_ctx);
    }
}
```

### 6.6 Reference code on XMC4500 for communicating with OPTIGA™ Trust X

```c
static volatile uint32_t optiga_pal_event_status;
extern void ifx_i2c_pl_pal_event_handler(void *p_ctx,uint8_t event);
void optiga_pal_i2c_event_handler(void* upper_layer_ctx, uint8_t event);

pal_i2c_t optiga_pal_i2c_context_0 =
{
    /// Pointer to I2C master context
    (void*)&i2c_master_0,
    /// Slave address
    0x30,
    /// Upper layer context
    NULL,
    /// Callback event handler
    pal_i2c_slave_1_event_handler
};

// Pal optiga slave 1 event handler
void optiga_pal_i2c_event_handler(void* upper_layer_ctx, uint8_t event)
{
    optiga_pal_event_status = event;
}

/* Function to verify I2C communication*/
pal_status_t test_optiga_communication(void)
{
    pal_status_t pal_return_status;
    uint8_t data_buffer[10] = {0x82};
    uint16_t data_length =1;

    // Set callback handler for i2c
    optiga_pal_i2c_context_0.upper_layer_event_handler
```
// Send 0x82 command to slave to check the state
optiga_pal_event_status = PAL_I2C_EVENT_BUSY;

do
{
    pal_return_status =
        pal_i2c_write(&optiga_pal_i2c_context_0,
                      data_buffer, data_length);
    if (pal_return_status == PAL_STATUS_FAILURE)
    {
        break;
    }
}

// Wait until slave completes write operation
} while (optiga_pal_event_status !=
          PAL_I2C_EVENT_TX_SUCCESS);

optiga_pal_event_status = PAL_I2C_EVENT_BUSY;
data_length = 4;

// Read the response for 0x82 command

do
{
    pal_return_status =
        pal_i2c_read(&optiga_pal_i2c_context_0,
                     data_buffer,
                     data_length);
    if (pal_return_status == PAL_STATUS_FAILURE)
    {
        break;
    }
}

// Wait until slave completes read operation
} while (optiga_pal_event_status !=
          PAL_I2C_EVENT_RX_SUCCESS);

return pal_return_status;

/***************************************************************
* Main Function
***********************************
***************************/

/**
* This function is the entry point of sample.
*
* \retval
* 0 on success
* 1 on failure
*/

int32_t main(Void)
{
    DAVE_STATUS_t status;
pal_status_t pal_return_status;

// Initialize your host code here (e.g. timers etc)
// Initialisation of DAVE Apps for XMC4500
status = DAVE_Init();

// Stop if DAVE init fails
if (status == DAVE_STATUS_FAILURE)
{
    while (1U)
    {
    
    }
}

pal_return_status = test_optiga_communication();

return pal_return_status;
7  OPTIGA™ Trust X Commands

This section provides short description of OPTIGA™ Trust X commands and mapping of these commands w.r.t Use Cases.

Table 13  OPTIGA™ Trust X command table

<table>
<thead>
<tr>
<th>Command Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetDataObject</td>
<td>Command to get (read) a data object</td>
</tr>
<tr>
<td>SetDataObject</td>
<td>Command to set (write) a data object</td>
</tr>
<tr>
<td>GetRandom</td>
<td>Command to generate a random stream</td>
</tr>
<tr>
<td>SetAuthScheme</td>
<td>Command to set the authentication scheme which gets used subsequently</td>
</tr>
<tr>
<td>GetAuthMsg</td>
<td>Command to get (receive from OPTIGA™ Trust X) an authentication message</td>
</tr>
<tr>
<td>SetAuthMsg</td>
<td>Command to set (send to OPTIGA™ Trust X) an authentication message</td>
</tr>
<tr>
<td>ProcUpLinkMsg</td>
<td>Command to process an up-link message for DTLS(receive from OPTIGA™ Trust X)</td>
</tr>
<tr>
<td>ProcDownLinkMsg</td>
<td>Command to process a down-link message for DTLS (send to OPTIGA™ Trust X)</td>
</tr>
<tr>
<td>CalcHash</td>
<td>Command to calculate a Hash</td>
</tr>
<tr>
<td>CalcSign</td>
<td>Command to calculate a signature</td>
</tr>
<tr>
<td>VerifySign</td>
<td>Command to verify a signature</td>
</tr>
<tr>
<td>CalcSSec</td>
<td>Command to execute a Diffie-Hellmann key agreement</td>
</tr>
<tr>
<td>DeriveKey</td>
<td>Command to derive keys</td>
</tr>
<tr>
<td>GenKeyPair</td>
<td>Command to generate public/private key pairs</td>
</tr>
<tr>
<td>OpenApplication</td>
<td>Command to launch an application</td>
</tr>
</tbody>
</table>

Table 14  Mapping of OPTIGA™ Trust X command with Use cases

<table>
<thead>
<tr>
<th>Use Case</th>
<th>OPTIGA™ Trust X commands used</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mutual Authentication using DTLS</td>
<td>SetAuthScheme, ProcUpLinkMsg &amp; ProcDownLinkMsg</td>
</tr>
<tr>
<td>One Way Authentication</td>
<td>GetRandom, GetDataObject, SetAuthScheme, SetAuthMsg &amp; GetAuthMsg</td>
</tr>
<tr>
<td>Crypto Toolbox</td>
<td>GetRandom, SetAuthScheme, SetAuthMsg, GetAuthMsg, CalcHash, CalcSign, VerifySign, CalcSSec, DeriveKey, GenKeyPair</td>
</tr>
<tr>
<td>Read General Purpose Data</td>
<td>GetDataObject</td>
</tr>
<tr>
<td>Write General Purpose Data</td>
<td>SetDataObject</td>
</tr>
</tbody>
</table>
8 Security Monitor

The Security Monitor is a central component which enforces the security policy of the OPTIGA™ Trust X. It consumes security events sent by security aware parts of the OPTIGA™ Trust X embedded SW and takes actions accordingly.

### 8.1 Security Events

The following table provides the definition of not permitted security events considered by the OPTIGA™ Trust X implementation.

<table>
<thead>
<tr>
<th>Event</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Decryption Failure</td>
<td>This event occurs in case a decryption and/or integrity check of provided data lead to an integrity failure.</td>
</tr>
<tr>
<td>Private Key Use</td>
<td>This event occurs in case the internal services are going to use an OPTIGA™ Trust X hosted private key.</td>
</tr>
<tr>
<td>Suspect System Behavior</td>
<td>This event occurs in case the embedded software detects inconsistencies with the expected behavior of the system. Those inconsistencies might be redundant information which doesn’t fit to their counterpart.</td>
</tr>
</tbody>
</table>

### 8.2 Security Policy

Security Monitor judges the notified security events regarding the number of occurrence over time and in case those violate the permitted usage profile of the system takes actions to throttle down the performance and thus the possible frequency of attacks.

The permitted usage profile is defined as:

1. One protected operation (refer to Table 15) events per $t_{\text{max}}$ period.
2. A Suspect System Behavior event is never permitted and will cause setting the SEC to its maximum.
3. $t_{\text{max}}$ is set to 5 seconds ($\pm$ 5%).

With other words it must not allow more than one out of the protected operations per $t_{\text{max}}$ period (worst case, ref to bullet 1. above). This condition must be stable, at least after 500 uninterrupted executions of protected operations.

For more information, please refer to Solution Reference Manual document available as part of the package.
RoHS Compliance

On January 27, 2003 the European Parliament and the council adopted the directives:

- 2002/95/EC on the Restriction of the use of certain Hazardous Substances in electrical and electronic equipment ("RoHS")
- 2002/96/EC on Waste Electrical and Electrical and Electronic Equipment ("WEEE")

Some of these restricted (lead) or recycling-relevant (brominated flame retardants) substances are currently found in the terminations (e.g. lead finish, bumps, balls) and substrate materials or mold compounds.

The European Union has finalized the Directives. It is the member states' task to convert these Directives into national laws. Most national laws are available, some member states have extended timelines for implementation. The laws arising from these Directives have come into force in 2006 or 2007.

The electro and electronic industry has to eliminate lead and other hazardous materials from their products. In addition, discussions are on-going with regard to the separate recycling of ceratin materials, e.g. plastic containing brominated flame retardants.

Infineon Technologies is fully committed to giving its customers maximum support in their efforts to convert to lead-free and halogen-free products. For this reason, Infineon Technologies' "Green Products" are ROHS-compliant.

Since all hazardous substances have been removed, Infineon Technologies calls its lead-free and halogen-free semiconductor packages "green." Details on Infineon Technologies' definition and upper limits for the restricted materials can be found here.

The assembly process of our high-technology semiconductor chips is an integral part of our quality strategy. Accordingly, we will accurately evaluate and test alternative materials in order to replace lead and halogen so that we end up with the same or higher quality standards for our products.

The use of lead-free solders for board assembly results in higher process temperatures and increased requirements for the heat resistivity of semiconductor packages. This issue is addressed by Infineon Technologies by a new classification of the Moisture Sensitivity Level (MSL). In a first step the existing products have been classified according to the new requirements.

\[\text{Lead-free} \quad \begin{array}{l} 
\text{Balls, FC bumps, Lead finish} \\
\text{Halogen-free} \\
\text{Substrate, Mold compound}
\end{array}\]

Any material used by Infineon Technologies is PBB and PBDE-free. Plastic containing brominated flame retardants, as mentioned in the WEEE directive, will be replaced if technically/economically beneficial.
10 Appendix A – Infineon I2C Protocol Registry Map

OPTIGA™ Trust X supports IFX I2C v1.65 and is implemented as I2C slave, which uses different address locations for status, control and data communication registers. These registers with description are outlined below in the following table.

<table>
<thead>
<tr>
<th>Register Address</th>
<th>Name</th>
<th>Size in Bytes</th>
<th>Description</th>
<th>Master Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x80</td>
<td>DATA</td>
<td>DATA_REG_LEN</td>
<td>This is the location where data shall be read from or written to the I2C slave</td>
<td>Read / Write</td>
</tr>
<tr>
<td>0x81</td>
<td>DATA_REG_LEN</td>
<td>2</td>
<td>This register holds the maximum data register (Addr 0x80) length. The allowed values are 0x0010 up to 0xFFFF. After writing the new data register length it becomes effective with the next I2C master access. However, in case the slave could not accept the new length it indicates its maximum possible length within this register. Therefore it is recommended to read the value back after writing it to be sure the I2C slave did accept the new value. Note: the value of MAX_PACKET_SIZE is derived from this value or vice versa (MAX_PACKET_SIZE=DATA_REG_LEN-5)</td>
<td>Read / Write</td>
</tr>
<tr>
<td>0x82</td>
<td>I2C_STATE</td>
<td>4</td>
<td>Bits 31:24 of this register provides the I2C state in regards to the supported features (e.g. clock stretching ...) and whether the device is busy executing a command and/or ready to return a response etc. Bits 15:0 defining the length of the response data block at the physical layer.</td>
<td>Read only</td>
</tr>
<tr>
<td>0x83</td>
<td>BASE_ADDR</td>
<td>2</td>
<td>This register holds the I2C base address as specified by Table 17. If not differently defined by a particular project the default value at reset is 0x20. After writing a different address the new address become effective with the next I2C master access. In case the bit 15 is set in addition to the new address (bit 6:0) it becomes the new default address at reset (persistent storage).</td>
<td>Write only</td>
</tr>
<tr>
<td>0x84</td>
<td>MAX_SCL_FREQ</td>
<td>4</td>
<td>This register holds the maximum clock frequency in KHz supported by the I2C slave. The value gets adjusted to the register I2C_Mode setting. Fast Mode (Fm): The allowed values are 50 up to 400. Fast Mode (Fm+): The allowed values are 50 up to 1000.</td>
<td>Read</td>
</tr>
<tr>
<td>0x85</td>
<td>GUARD_TIME†</td>
<td>4</td>
<td>For details refer to Table 20</td>
<td>Read only</td>
</tr>
<tr>
<td>0x86</td>
<td>TRANS_TIMEOUT†</td>
<td>4</td>
<td>For details refer to Table 20</td>
<td>Read only</td>
</tr>
</tbody>
</table>

† In case the register returns 0xFFFFFFFF the register is not supported and the default values specified in Table ‘List of protocol variations’ shall be applied.
# Appendix A – Infineon I2C Protocol Registry Map

<table>
<thead>
<tr>
<th>Register Address</th>
<th>Name</th>
<th>Size in Bytes</th>
<th>Description</th>
<th>Master Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>0x88</td>
<td>SOFT_RESET</td>
<td>2</td>
<td>Writing to this register will cause a device reset. This feature is optional</td>
<td>Write only</td>
</tr>
<tr>
<td>0x89</td>
<td>I2C_MODE</td>
<td>2</td>
<td>This register holds the current I2C Mode as defined by Table 18. The default mode is SM &amp; FM (011B).</td>
<td>Read / Write</td>
</tr>
</tbody>
</table>

## Table 17 Definition of BASE_ADDR

<table>
<thead>
<tr>
<th>Fields</th>
<th>Bits</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEF_ADDR</td>
<td>15</td>
<td>0</td>
<td>Volatile address setting by bit 6:0, lost after reset.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Persistent address setting by bit 6:0, becoming default after reset.</td>
</tr>
<tr>
<td>BASE_ADDR</td>
<td>6:0</td>
<td>0x00-0x7F</td>
<td>I²C base address specified by Table 16</td>
</tr>
</tbody>
</table>

## Table 18 Definition of I2C_MODE

<table>
<thead>
<tr>
<th>Fields</th>
<th>Bits</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEF_MODE</td>
<td>15</td>
<td>0</td>
<td>Volatile mode setting by bit 2:0, lost after reset.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Persistent mode setting by bit 2:0, becoming default after reset.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>This bit is always read as 0.</td>
</tr>
<tr>
<td>MODE¹</td>
<td>2:0</td>
<td>001</td>
<td>Sm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>010</td>
<td>Fm</td>
</tr>
<tr>
<td></td>
<td></td>
<td>011</td>
<td>SM &amp; Fm (fab out default)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>100</td>
<td>Fm+</td>
</tr>
<tr>
<td></td>
<td></td>
<td>other values</td>
<td>not valid; writing will be ignored</td>
</tr>
</tbody>
</table>

¹ In case the register returns 0xFFFFFFFF the register and its functionality is not supported

² This mode defines the adherence of the bus signals to the electrical characteristics according standard I2C bus specification
Appendix A – Infineon I2C Protocol Registry Map

<table>
<thead>
<tr>
<th>31</th>
<th>30</th>
<th>29</th>
<th>28</th>
<th>27</th>
<th>26</th>
<th>25</th>
<th>24</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSY</td>
<td>RESP_RDY</td>
<td>RFU</td>
<td>SOFT_RESET</td>
<td>CONT_READ</td>
<td>REP_START</td>
<td>CLK_STRETCHING</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>22</td>
<td>21</td>
<td>20</td>
<td>19</td>
<td>18</td>
<td>17</td>
<td>16</td>
</tr>
</tbody>
</table>

RFU

15-0

Length of data block to be read

### Table 19 - Definition of I2C_STATE

<table>
<thead>
<tr>
<th>Field</th>
<th>Bit(s)</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BUSY</td>
<td>31</td>
<td>0</td>
<td>Device is not busy</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Device is busy executing a command</td>
</tr>
<tr>
<td>RESP_RDY</td>
<td>30</td>
<td>0</td>
<td>Device is not ready to return a response</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Device is ready to return a response</td>
</tr>
<tr>
<td>SOFT_RESET</td>
<td>27</td>
<td>0</td>
<td>SOFT_RESET not supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>SOFT_RESET supported</td>
</tr>
<tr>
<td>CONT_READ</td>
<td>26</td>
<td>0</td>
<td>Continue Read not supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Continue Read supported</td>
</tr>
<tr>
<td>REP_START</td>
<td>25</td>
<td>0</td>
<td>Repeated start not supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Repeated start supported</td>
</tr>
<tr>
<td>CLK_STRETCHING</td>
<td>24</td>
<td>0</td>
<td>Clock stretching not supported</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Clock stretching supported</td>
</tr>
</tbody>
</table>

### 10.1 IFX I2C Protocol Variations

To fit best to application specific requirements the protocol might be tailored by specifying a couple of parameters which is described in the following table.

### Table 20 - List of Protocol Variations

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAX_PACKET_SIZE</td>
<td>0x110</td>
<td>Maximum packet size accepted by the receiver. The protocol limits this value to 0xFFFF, but there might be project specific requirements to reduce the transport buffers size for the sake of less RAM footprint in the communication stack. If shortened, it could be statically defined or negotiated at the physical layer.</td>
</tr>
<tr>
<td>WIN_SIZE</td>
<td>1</td>
<td>Window size of the sliding windows algorithm. The value could be 1 up to 2.</td>
</tr>
<tr>
<td>MAX_NET_CHAN</td>
<td>1</td>
<td>Maximum number of network channels. The value could be 1 up to 16. One indicates the OSI Layer 3 is not used and the CHAN field of the PCTR must be set to 0000.</td>
</tr>
<tr>
<td>CHAINING</td>
<td>TRUE</td>
<td>Chaining on the transport layer is supported (TRUE) or not (FALSE)</td>
</tr>
<tr>
<td>TRANS_TIMEOUT</td>
<td>10 ms</td>
<td>(Re) transmission timeout specifies the number of milliseconds to be elapsed until the transmitter considers a frame transmission is lost and retransmits the non-acknowledged frame. The Timer gets started as soon as the complete frame is</td>
</tr>
</tbody>
</table>
### Infineon I2C Protocol Registry Map

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| transmitted. The value could be 1 up to 1000. However, as higher the number as longer does it take to recover from a frame transmission error.  
**Note:** The acknowledge timeout on the receiver side must be shorter than the retransmission timeout to avoid unnecessary frame repetitions. |
| **TRANS_REPEAT** | 3             | Number of transmissions to be repeated until the transmitter considers the connection is lost and starts a re-synchronization with the receiver. The value could be 1 up to 4. |
| **BASE_ADDR**   | 0x30          | I2C (base) address. This address could be statically defined or dynamically negotiated by the physical layer. If not different specified the default value is 0x30. |
| **MAX_SCL_FREQ** | 1000 kHz      | Maximum SCL clock frequency in kHz. |
| **GUARD_TIME**  | 50 µs         | Minimum time to be elapsed at the I2C master measured from read data (STOP condition) until the next write data (Start condition) is allowed to happen.  
**Note 1:** For two consecutive accesses on the same device GUARD TIME re-specifies the value of tBUF as specified by [I2Cbus].  
**Note 2:** Even if another I2C address is accessed in between GUARD TIME has to be respected for two consecutive accesses on the same device. |
| **SOFT_RESET**  | Any write attempt to the SOFT_RESET register will trigger a warm reset (reset w/o power cycle). This register is optional and its presence is indicated by the I2C_STATE register’s "SOFT_RESET" flag. |
Appendix B – Power Management

When operating, the power consumption of OPTIGA™ Trust X is limited to meet the requirements regarding the power limitation set by the Host. The power limitation is implemented by utilizing the current limitation feature of the underlying hardware device in steps of 1mA from 6mA to 15 mA with a precision of ±5%.

11.1 Low Power Sleep Mode

The OPTIGA™ Trust X automatically enters a low-power mode after a configurable delay. Once it has entered Sleep mode, the OPTIGA™ Trust X resumes normal operation as soon as its address is detected on the I2C bus. In case no command is sent to the OPTIGA™ Trust X it behaves as shown in Figure 14.

1. As soon as the OPTIGA™ Trust X is idle it starts to count down the “delay to sleep” time (tSDY).
2. In case this time elapses the device enters the “go to sleep” procedure.
3. The “go to sleep” procedure waits until all idle tasks are finished (e.g. counting down the SEC). In case all idle tasks are finished and no command is pending, the OPTIGA™ Trust X enters sleep mode.

Figure 14 Go-to-Sleep Diagram
## Revision history

<table>
<thead>
<tr>
<th>Document version</th>
<th>Date of release</th>
<th>Description of changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.6</td>
<td>08.02.2019</td>
<td>Updated PG-USON10-2 footprint</td>
</tr>
<tr>
<td>2.5</td>
<td>31.01.2018</td>
<td>Feedback incorporation from all internal regions</td>
</tr>
<tr>
<td>2.4</td>
<td>11.01.2018</td>
<td>Feedback incorporation from all internal regions</td>
</tr>
<tr>
<td>2.3</td>
<td>01.01.2018</td>
<td>Feedback incorporation from all internal regions</td>
</tr>
<tr>
<td>2.2</td>
<td>12.12.2017</td>
<td>Feedback from all internal regions</td>
</tr>
<tr>
<td>2.1</td>
<td>23.06.2017</td>
<td>Updated Key features and Enhanced Security</td>
</tr>
<tr>
<td>2.0</td>
<td>08.06.2017</td>
<td>Updated Key features and Enhanced Security</td>
</tr>
<tr>
<td>1.4</td>
<td>22.02.2017</td>
<td>First version release</td>
</tr>
<tr>
<td>1.3</td>
<td></td>
<td>Internal review</td>
</tr>
<tr>
<td>1.2</td>
<td></td>
<td>Internal review</td>
</tr>
<tr>
<td>1.1</td>
<td></td>
<td>Internal review</td>
</tr>
<tr>
<td>1.0</td>
<td></td>
<td>Internal review</td>
</tr>
</tbody>
</table>
IMPORTANT NOTICE

The information given in this document shall in no event be regarded as a guarantee of conditions or characteristics (“Beschaffenheitsgarantie”).

With respect to any examples, hints or any typical values stated herein and/or any information regarding the application of the product, Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind, including without limitation warranties of non-infringement of intellectual property rights of any third party.

In addition, any information given in this document is subject to customer’s compliance with its obligations stated in this document and any applicable legal requirements, norms and standards concerning customer’s products and any use of the product of Infineon Technologies in customer’s applications.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer’s technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (www.infineon.com).

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies’ products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.