
# MS1201 Security Sub-system

## FIPS 140-2 Non-Proprietary Security Policy

Document Revision: 1.6

Document Date: September 2021

Prepared by:

atsec information security Corp.

9130 Jollyville Road, Suite 260

Austin, TX 78759

www.atsec.com

eWBM:

14F, 9, Teheran-ro 20-gil,
Gangnam-gu Seoul, South Korea

phone:+82 2 556 7878

https://www.ewbm.co.kr/

For further information contact:

JS Jeong jsjeong@e-wbm.com
Stephen Oh oh@e-wbm.com

## Copyrights and Trademarks

# Table of Contents

# 1 Introduction

This document is the non-proprietary FIPS 140-2 Security Policy for the MS1201 Security Sub-system cryptographic module. It contains a specification of the rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-2 (Federal Information Processing Standards Publication 140-2) for a Security Level 2 module.

## 1.1 Purpose of the Security Policy

There are three major reasons that a security policy is needed:

- it is required for FIPS 140-2 validation,
- it allows individuals and organizations to determine whether a cryptographic module, as implemented, satisfies the stated security policy, and
- it describes the capabilities, protection and access rights provided by the cryptographic module, allowing individuals and organizations to determine whether it will meet their security requirements.

## 1.2 Target Audience

This document is part of the package of documents that are submitted for FIPS 140-2 conformance validation of the module. It is intended for the following audience:

- Developers.
- FIPS 140-2 testing lab.
- The Cryptographic Module Validation Program (CMVP).
- Customers using or considering integration of the MS1201 Security Sub-system or its single-chip embodiment.

# 2 Cryptographic Module Specification

## 2.1 Module Overview

The MS1201 Security Sub-system is a Silicon IP Security Module which includes a complete set of high-level and low-level cryptographic functions. It offers key management and crypto functions needed for platform and application security such as Content Protection and Mobile Payment, and can be used stand-alone or as a 'Root of Trust' to support a TEE-based platform.

The MS1201 Security Sub-system completely shields all key and security sensitive data from all CPUs, interfaces and memory. Security sensitive materials are stored as assets that never leave the MS1201 Security Sub-system in unencrypted and/or non-authenticated form.

Additionally, the MS1201 Security Sub-system offers hardware security features that are needed when operating in a Trusted Execution Environment (TEE). These features include One-Time-Programmable memory (OTP) access and management, Random Number Generation / entropy source, timers, (short) monotonic/non-volatile counters and import and export of keys and other assets.

## 2.2 Intended Usage

The primary application of the MS1201 Security Sub-system is in mobile communications and consumer electronics appliances, where authentication, encrypted content processing using standard protocols, and protection of keys and other sensitive assets are required. The MS1201 Security Sub-system is best suited for mobile phones, tablets, wireless handsets, PDA-like devices and set top boxes that have the resources and connectivity to download, store and play back digital media content. These small, battery-powered devices require a low power IP solution with these features available in the MS1201 Security Sub-system:

- Low-power and small footprint IP.
- Internal storage for protection and management of sensitive keys and assets.
- Root of Trust as true hardware interface to on chip One-Time Programmable (OTP) memory.
- Secure Timers (hardware counters).
- Encryption engines to offload computationally intensive symmetric algorithms: AES, Triple-DES.
- Hash engine to offload computationally intensive hash algorithms: SHA-1, SHA-2.
- Public Key Encryption, supporting RSA, ECDSA (sub-)functions.
- True random number generator (TRNG), also known as Non-deterministic random number generator (NDRNG).
- Embedded DMA controller for high speed symmetric crypto and hash data transfer.

## 2.3 FIPS 140-2 Module Information

For the purpose of this Cryptographic Module Validation, the MS1201 Security Sub-system is synthesized in silicon as a single-chip hardware module validated at overall security level 2.

Table 1 below shows the security level claimed for each of the eleven sections that comprise the FIPS 140-2:

| FIPS 140-2 Sections | | Security Level |
|---|---|---|
| 1 | Cryptographic Module Specification | 2 |
| 2 | Cryptographic Module Ports and Interfaces | 2 |
| 3 | Roles, Services and Authentication | 2 |
| 4 | Finite State Model | 2 |
| 5 | Physical Security | 3 |
| 6 | Operational Environment | N/A |
| 7 | Cryptographic Key Management | 2 |
| 8 | EMI/EMC | 2 |
| 9 | Self Tests | 2 |
| 10 | Design Assurance | 2 |
| 11 | Mitigation of Other Attacks | N/A |

**Table 1 - Security Levels**

The MS1201 Security Sub-system has been tested as a single chip embodiment.

## 2.4 Approved Modes of Operation

The MS1201 Security Sub-system has two modes of operation: FIPS mode and non-FIPS mode.

- In FIPS mode of operation, only FIPS-Approved or FIPS-Allowed cryptographic algorithms with specific modes and key lengths can be requested. Table 12, and Table 12a show all algorithms supported by the module in FIPS mode.
- In non-FIPS mode of operation, only the non-approved cryptographic algorithms listed in Table 13 are available.

The mode of operation is implicitly assumed depending on the service invoked. If the user is requesting *either* a FIPS-Approved service *or* a FIPS-Allowed service, then the module will implicitly be in FIPS mode. If the user is requesting a non-FIPS service, then the module is implicitly in non-FIPS mode.

## 2.5 System Block Diagram

The figure below shows a system diagram in which MS1201 Security Sub-system is integrated in a SoC (System on a Chip) with one or more CPUs, connected to a common bus system.

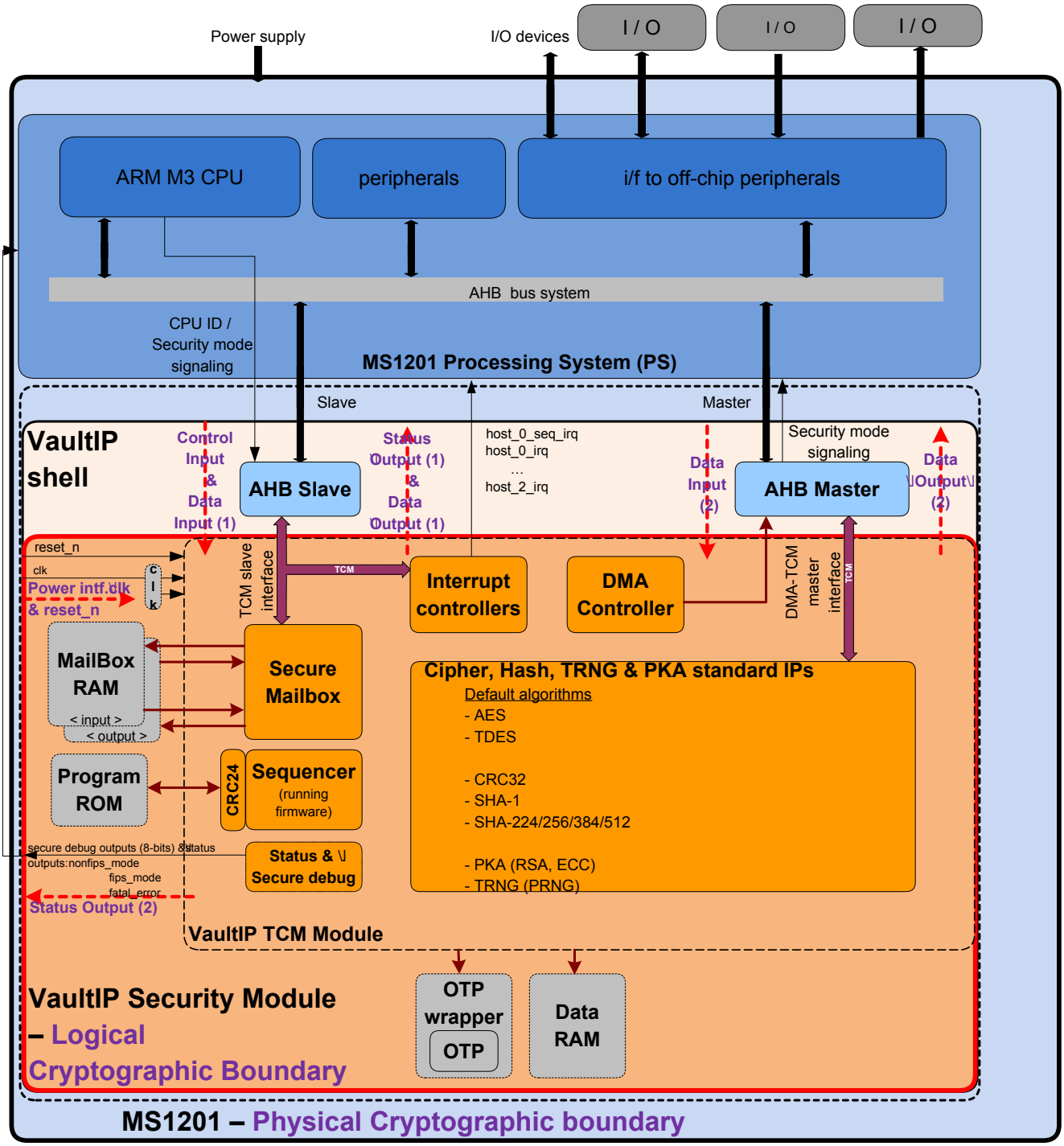


**Figure 1- System Block Diagram**

For the purpose of this validation, the physical cryptographic boundary is enclosed in the MS1201 single chip embodiment. The logical cryptographic module boundary is represented within the red line. The orange boxes represent the MS1201 Security Sub-system components that comprise the IP core (the MS1201 Security Sub- system firmware is stored in Program ROM). The grey boxes represent components that are provided in the IP core but must be replaced or adjusted during the synthesis process as they are technology dependent (OTP, RAM, ROM, etc.).

## 2.6 Hardware Block Diagram

MS1201 Security Sub-system consists of two major components, the Verilog RTL and the Firmware running from a ROM on the sequencer. The RTL implements the cryptographic algorithms and basic public key big number mathematics. The Firmware handles the higher level operations, manages the keys and takes care of the data transfers by setting up DMAs.

The breakdown of MS1201 Security Sub-system is shown in Figure 2; it shows the details of all interfaces that cross the security boundary and the first hierarchy levels of the MS1201 Security Sub-system RTL.

Firmware is located in the Program ROM. The firmware has many routines; typically, the sources for each routine are located in an individual assembly file. All firmware routines are located on the same hierarchical level.

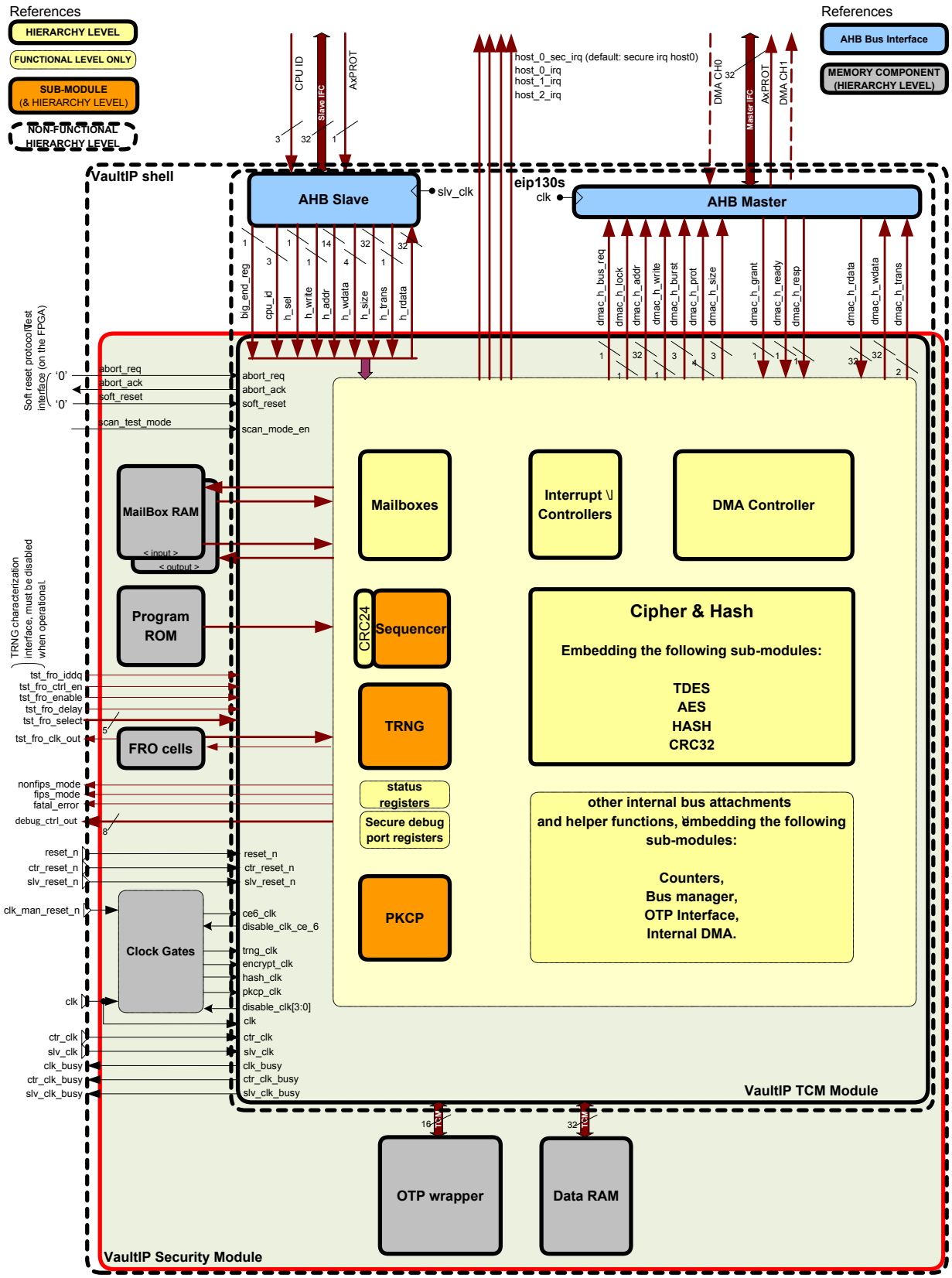


**Figure 2- Hardware Block Diagram**

## 2.7 MS1201 Security Sub-system module breakdown

The next list shows all (sub-)module levels of MS1201 Security Sub-system and their corresponding version numbers. The levels provide an overview starting at the MS1201 Security Sub-system shell level.

Top-level MS1201 Security Sub-system and sub-modules HW2.2.0.

- MS1201 Security Sub-system Module

  Implementation of MS1201 Security Sub-system Tightly Coupled Memory (TCM) level. The TCM level embeds many sub-modules. Several components are not implemented as dedicated sub-modules from a design perspective, but do perform a specific operation and have their own hierarchy level:

  - CRC32
  - CRC24
  - Counters
  - Mailboxes
  - OTP Interface
  - Bus manager
  - Internal DMA

  The individual sub-modules from a hardware design perspective are listed below.

  - AES with ECB, CBC, CTR, CMAC, and CCM.
    - AES data path (ECB)
  - PKCP (public key co-processor), available for the Internal Controller to offload computationally intensive Public Key operations, such as modular exponentiations and Elliptic Curve Cryptography operations.
  - HASH, SHA-1 and SHA-2, including SHA-224, SHA-256, SHA-384 and SHA-512.
  - Multi-input 32-bit wide adders
  - TRNG (generate and capture the entropy).
  - Sequencer (‘tiny’ RISC processor), running the Firmware code.
  - DMA controller, requesting data reads or writes to or from the MS1201 Security Sub-system.
  - Interrupt controller, captures the various interrupt sources and manages these to a single host interrupt. Multiple instantiations.
  - The following are available, but not FIPS approved: AES-XTS, AES-GCM, Triple-DES with ECB and CBC.

Technology specific cells in the MS1201 Security Sub-system shell module.

- Memories
  - Mailbox RAM (2 instantiations, one input mailbox and one output mailbox)
  - Program ROM
  - OTP wrapper
  - Data RAM
- Clock gates
- FRO cells and related components / standard cells

Firmware FW2.5.3

- The MS1201 Security Sub-system Boot Firmware located in the Program ROM.

  This includes token management from and to the mailboxes (external interface), higher-level crypto operations, key generation, DRBG engine, asset management, DMA setup and generic engine control.

  - PKA Firmware

Interfaces located outside the security logical boundary, but inside the MS1201 Security Sub-system top level.

- AHB Slave
- AHB Master: Interface module converts a DMA request into a data transaction from TCM to external AHB or reverse.

# 3 Ports and Interfaces

The MS1201 Security Sub-system module embeds a single slave and master interfaces. The slave interface is used to receive commands from one or more host CPUs and send the appropriate response. The master interface is used for autonomous data reads and writes from and to an external memory, flash or interface.

Additionally, the MS1201 Security Sub-system includes physical ports for showing the crypto module status, establishing the role that is requesting services, and resetting the crypto module.

## 3.1 Physical ports

The summary of interface pins located on the physical boundary of the MS1201 Security Sub-system is given in the tables below and shown in Figure 2. For clarity, signals are grouped by function in separate tables. Each port pin provides its name, direction, clock domain and function.

The first set of signals in the table below is hardware related and drives the various clock and reset signals.

| Port Name | Direction | Clock Domain | Function |
|---|---|---|---|
| **Clocks** | | | |
| slv_clk | IN | slv_clk | Host interface clock. |
| ctr_clk | IN | ctr_clk | System counter clock signal. This clock signal may not be gated and must be connected to a fixed frequency, while the other clock speed could vary. |
| clk | IN | clk | Internal crypto-module clock. |
| **Reset** | | | |
| slv_reset_n | IN | slv_clk | Host interface reset. |
| ctr_reset_n | IN | ctr_clk | Counter reset. This reset signal may only be active (set to ‘0’) when the system is reset and must remain inactive after that, such that the counters remain counting. |
| reset_n | IN | clk | Module reset. |
| clk_man_reset_n | IN | n/a | This signal provides a means to reset all flip-flops inside the clock gate modules, e.g. for DFT and for the simulation processes to start in a known state. The clock gates are typically not connected to the global **reset_n** signal because it may be required to have the clocks running during a system reset. |
| **External clock signals for dynamic clock control** | | | |
| slv_clk_busy | OUT | clk | Indicates that the Host interface is busy with Host bus transfers. When 1b, indicates active transfer on Host bus. |
| ctr_clk_busy | OUT | ctr_clk | When 1b, indicates that the counter clock domain is active. This signal is always asserted (set to ‘1’), except when the counter module is in reset (**ctr_reset_n** set to ‘0’). |

| | | | |
|---|---|---|---|
| clk_busy | OUT | clk | When 1b, indicates that the module is active and busy with processing data and tokens. |

**Table 2 - Clock and Reset ports**

The second group is related to software reset and is designed only for testing purposes: this functionality is tied-off in the production cryptographic module and is only provided for reference1.

| Port Name | Direction | Clock Domain | Function |
|---|---|---|---|
| soft_reset<br>[For FIPS: tied-off to zero: 1'b0] | IN | slv_clk | Soft reset input. When this signal is made high, internal (state) registers is cleared and crypto engines are reset. |
| abort_req<br>[For FIPS: tied-off to zero: 1'b0] | IN | slv_clk | Abort request signal. A high level of this signal indicates a request for a soft reset of the MS1201 Security Sub-system module. |
| abort_ack<br>[For FIPS: unconnected] | OUT | slv_clk | Abort acknowledge signal. A high level of this signal indicates that the MS1201 Security Sub-system module is ready to receive a soft reset. |

**Table 3 - Soft reset ports**

The third group provides signals to indicate the status of the MS1201 Security Sub-system:

| Port Name | Direction | Clock Domain | Function |
|---|---|---|---|
| fatal_error | OUT | clk | When active (set to '1'), the MS1201 Security Sub-system detected a fatal error and stops operation. Fatal errors can happen when the CRC on the Firmware Program ROM fails or when a self-test fails. The value of this bit equals the value of the corresponding bit in the MODULE_STATUS register (bit 31). |
| debug_ctrl_out[7:0] | OUT | clk | Secure debug control output bus. |

**Table 4 - Status Signal ports**

The fourth group provides signals to indicate the Power mode of the MS1201 Security Sub-system:

| Port Name | Direction | Clock Domain | Function |
|---|---|---|---|
| power_mode_out | OUT | clk | 0b: MS1201 Security Sub-system is operational.<br>1b: MS1201 Security Sub-system is ready to enter 'Sleep Mode'.<br>Only valid in combination with an active **power_mode_write** signal. |
| power_mode_write | OUT | clk | When active it indicates the status of **power_mode_out** is valid. |

[1] Can be completely reset by a hard reset. The firmware can be reset using a reset token, provided via the Control Input Interface.

| Port Name | Direction | Clock Domain | Function |
|---|---|---|---|
| power_mode_in | IN | clk | The status of this signal is checked by the core during power up after Sleep Mode to see if the state information from Data RAM must be restored.<br>0b: MS1201 Security Sub-system must not restore the state information from Data RAM after reset.<br>1b: MS1201 Security Sub-system must restore the state information from Data RAM after reset. |

**Table 5 - Power Mode Control Signals**

The following table provides the signals of the target Host interface. Besides global configuration, this physical interface provides the token interface to the mailboxes. Writing to the mailbox triggers processing. After processing, the results can be read from the output mailbox using this same physical interface.

| Port Name | Direction | Clk.Dom. | Function |
|---|---|---|---|
| cpu_id[2:0] | IN | slv_clk | Processor Host ID bits. These bits must be valid together with h_addr. |
| h_sel | IN | slv_clk | AHB Select. Selects the TCM port for a bus transfer. |
| h_write | IN | slv_clk | AHB Write. Indicates the direction of a bus transfer. |
| h_ready_in | IN | slv_clk | AHB Ready Input. |
| h_trans | IN | slv_clk | AHB Transfer type control bus bit [1], bit [0] is not used. Indicates the type of current bus transfer |
| h_size[2:0] | IN | slv_clk | AHB Size. Indicates the size of the current bus transfer |
| h_addr[13:0] | IN | slv_clk | AHB Address bus. Indicates the target address for the bus transfer (byte addressable). |
| h_wdata[31:0] | IN | slv_clk | AHB Write Data bus. Transfers data from Master to Slave. |
| h_ready_out | OUT | slv_clk | AHB Ready Output. Indicates extension of the bus transfer, when de-asserted (‘0’). |
| h_resp[1:0] | OUT | slv_clk | AHB Response. Indicates the status of the bus transfer. |
| h_rdata[31:0] | OUT | slv_clk | AHB Read Data bus. Transfers data from Slave to Master. |
| big_end_reg | IN | slv_clk | Big-endian. Indicates the byte order of the transfers.<br>This input should not change while transfers are active. |

**Table 6 – AHB Slave interface (Host processor bus)**

The next table provides the signals of the master data interface. This physical interface provides DMA signals and a target TCM interface. A request for data is initiated via the DMA interface. It is a requirement from the external system to provide the requested data on the TCM interface. Depending on the indicated direction of the DMA, input data is requested, or result data is available to be read.

| Port Name | Direction | Clock Domain | Function |
|---|---|---|---|
| **Read Command channel** | | | |
| dmac_h_grant | IN | clk | The AHB Master is granted the bus and is currently the highest priority master on the bus |
| dmac_h_ready | IN | clk | The current bus transfer has finished. |
| dmac_h_rdata[31:0] | IN | clk | Transfers read data from an AHB slave to the AHB Master during read operations. |
| dmac_h_resp[1:0] | IN | clk | Transfer response; provides additional transfer status information. |
| dmac_h_bus_req | OUT | clk | The AHB Master requests the bus. |
| dmac_h_lock | OUT | clk | The AHB Master requires access to the bus for a locked transfer. |
| dmac_h_write | OUT | clk | Direction of the current bus transfer. |
| dmac_h_addr[31:0] | OUT | clk | Target address for the bus transfer (byte address) |
| dmac_h_burst[2:0] | OUT | clk | The current bus transfer forms part of a burst |
| dmac_h_prot[3:0] | OUT | clk | AHB protection level |
| dmac_h_size[2:0] | OUT | clk | Size of the current bus transfer |
| dmac_h_trans[1:0] | OUT | clk | Type of the current transfer. |
| dmac_h_wdata[31:0] | OUT | clk | Transfers write data from the AHB Master to an AHB Slave during write operations |

**Table 7 - Master DMA interface ports (AHB)**

The next table provides an overview of the interrupt outputs. By default, the enabled number of interrupts equals the number of mailboxes. If the slave interface supports secure accesses, host 0 has a dedicated IRQ for that.

When accessing the MS1201 Security Sub-system module securely (prot_acc_n = 0b), it can also configure the mailboxes and interrupts. During integration, other optional interrupts can be enabled in the module, dependent on the system host CPUs and access requirements to the MS1201 Security Sub-system.

| Port Name | Direction | Clock Domain | Function |
|---|---|---|---|
| host_0_sec_irq | OUT | slv_clk | Combined interrupt output (active HIGH) for one or more Hosts and Domain.<br>Represents Host0, secure interrupt. The interrupt controller is only accessed when Host ID equals 0 and PROT is zero (secure).<br>This interrupt is only available when the slave interface has a prot signal. |
| host_0_irq | OUT | slv_clk | Combined interrupt output (active HIGH) for one or more Hosts and Domain.<br>Represents Host0, non-secure interrupt. The interrupt controller is only accessed when Host ID equals 0 and PROT is one (non-secure). |
| host_1_irq | OUT | slv_clk | Combined interrupt output (active HIGH) for one or more Hosts and Domain.<br>Represents Host1, non-secure interrupt. The interrupt controller is only accessed when Host ID equals 1 and PROT is one (non-secure). |
| host_2_irq | OUT | slv_clk | Combined interrupt output (active HIGH) for one or more Hosts and Domain.<br>Represents Host2, non-secure interrupt. The interrupt controller is only accessed when Host ID equals 2 and PROT is one (non-secure). |

**Table 8 - Interrupt signal ports**

Another set of signals available on the FIPS boundary is the SCAN and FRO characterization signals. Only the *scan_mode_*en and *tst_fro_iddq* signals are connected for device production test purposes. The direct FRO input and output characterization signals are tied-off (inputs) or left unconnected (output)2.

| Port Name | Direction | Clock Domain | Function |
|---|---|---|---|
| **Characterization / FRO Characterization** | | | |
| scan_mode_en | IN | None | Active HIGH enable signal for scan_test_mode. This signal typically comes from a testmode controller and is used to break unwanted combinatorial loops during scan test. |
| tst_fro_iddq | IN | None | Active HIGH enable signal for IDDq testing – this forces all fro_enable outputs LOW to shut down all FROs, overruling all other control signals. This is a combinatorial function, TRNG module clocks don't need to run for this to work. |
| tst_fro_ctrl_en <= 1'b0 | IN | None | Active HIGH enable signal for FRO characterization (enables the tst_fro_select, |

[2] MS1201 Security Sub-system provides a token that can be used to return sampled TRNG outputs. The requested number of bits is returned over the RxT data output interface using DMA.

| | | | |
|---|---|---|---|
| | | | tst_fro_enable and tst_fro_delay inputs). This is a combinatorial function, TRNG module clocks don't need to run for this to work. |
| tst_fro_select[4:0] <= 5'b00000 | IN | None | FRO selection input (valid values 0-7). A selected FRO will have its fro_testin input forced LOW. |
| tst_fro_enable <= 1'b0 | IN | None | Active HIGH enable signal for FRO selected by tst_fro_select. |
| tst_fro_delay <= 1'b0 | IN | None | Delay chain length selection for FRO selected by tst_fro_select. This input should only be changed while tst_fro_enable is LOW. |
| tst_fro_clk_out (unconnected) | OUT | None | Output clock signal on the FRO shell module from the FRO selected by tst_fro_select, forced 'low' when tst_fro_ctrl_en is 'low'. This output can also be activated for register-controlled characterization. |

**Table 9 - TRNG control ports**

## 3.2 Logical Interfaces

The Slave Interface ports communicate with the host through the AHB slave interface, providing the token interface to the mailboxes. Writing to the mailbox triggers processing. Once the input token is processed completely, the results can be read from the output mailbox using this same interface. The Firmware running on the embedded sequencer reads the input tokens, starts processing and writes the output token triggering the corresponding interrupt. Based on the interrupt, an external host can read the result output token. Output is not available in the output mailbox until the input token is fully processed.

Input tokens sent through the Slave Interface constitutes data input and control input interfaces; output tokens constitute data output and status output interfaces.

The Master DMA/TCM interface communicates with the host through the AHB master interface, providing DMA signals and a target TCM interface. A request for data is initiated via the DMA interface. It is a requirement from the external system to provide the requested data on the TCM interface. Depending on the direction of the DMA, input data is requested, or result data is available to be read. The Master DMA/TCM interface provides support for data input and data output interfaces.

The MS1201 Security Sub-system also includes additional ports for Control Input and Status Output:

- The Clock and Reset ports provide Control Input and Status Output interfaces.
- The Soft Reset ports (testing only) provide Control Input and Status Output interfaces.
- The Status Signal ports provide the Status Output interface.
- Interrupt signal ports provide the Status Output interface.
- The TRNG control ports (testing only) provide Control Input and Status Output.

Finally, the power port provides power input to the MS1201 Security Sub- system.

The following table shows the mapping between the ports available in the MS1201 Security Sub-system and the logical interfaces:

| Port Groups | Data Input | Data Output | Control Input | Status Output | Power Input |
|---|---|---|---|---|---|
| Clock and Reset ports | | | ✓ | ✓ | |
| Soft Reset ports | | | ✓ | ✓ | |
| Status Signal ports | | | | ✓ | |
| Slave interface ports (AHB Slave Interface) | ✓ | ✓ | ✓ | ✓ | |
| Master DMA interface ports (AHB Master Interface) | ✓ | ✓ | | | |
| Interrupt signal ports | | | | ✓ | |
| TRNG control ports | | | ✓ | ✓ | |
| Power port | | | | | ✓ |

**Table 10 - Logical Interfaces**

# 4 Roles, Services and Authentication

## 4.1 Roles

The MS1201 Security Sub-system module is a part of a System on Chip (SoC), where applications running on the system can use the MS1201 Security Sub-system cryptographic services. Applications must identify and authenticate to the MS1201 Security Sub-system through one of the following roles:

- User Role: This role performs general services, cryptographic operations, and asset managements.
- Crypto Officer Role: This role can perform the same functionality as the user role, but it also performs initialization services in the cryptographic module (e.g. configuration of the TRNG engine, write operations in OTP and registers).

The MS1201 Security Sub-system module implements a role-based authentication method (See section 4.3 ).

Internal mechanisms of the MS1201 Security Sub-system ensure that User and Crypto Officer service requests and assets (key material and other CSPs) are properly separated and protected.

Next section shows the services provided by the MS1201 Security Sub-system and the roles that can request them. All services require an authorized role.

## 4.2 Services

The following table presents the services provided by the MS1201 Security Sub-system, including:

- The input token through which the service is requested.
- The authorized roles: a checkmark in the role column indicates that the user authenticated with that role can perform the service.
- The cryptographic algorithms involved in the service. The algorithm can be split in several roles for different modes or key lengths as they may not be available in FIPS mode of operation.
- Whether the service is available in FIPS mode
- The Keys and Critical Security Parameters (CSPs) involved in the service.
- How the service accesses the Keys and CSPs: (C)reate (create and fill the CSP), (R)ead (read an existing CSP), (U)pdate (write an existing CSP), (D)elete (delete and zeroize memory where the CSP was stored). An asterisk (*) indicates that the CSP is transported via the token or the Host memory DMA (assets cannot be used).

| Service | Input Token | Roles | | Cryptographic Algorithms | FIPS | Keys and CSPs | Access |
|---|---|---|---|---|---|---|---|
| | | User | CO | | | | |
| Encryption and Decryption | Encryption | ✓ | ✓ | AES (ECB, CBC, CTR) | ✓ | AES key | R |
| | | ✓ | ✓ | AES (CCM) | ✓ | AES key | R |
| | | ✓ | ✓ | AES (XTS) | | AES key | R |

| Service | Input Token | Roles | | Cryptographic Algorithms | FIPS | Keys and CSPs | Access |
|---|---|---|---|---|---|---|---|
| | | User | CO | | | | |
| | | ✓ | ✓ | AES (GCM) | | AES Key | R |
| | | ✓ | ✓ | Triple-DES (ECB, CBC) | | Triple-DES keys | R |
| Message Digest | Hash | ✓ | ✓ | SHA-1 | ✓ | n/a | |
| | | ✓ | ✓ | SHA-224, SHA-256 | ✓ | n/a | |
| | | ✓ | ✓ | SHA-384, SHA-512 | ✓ | n/a | |
| MAC Generation | MAC | ✓ | ✓ | HMAC-SHA-224, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 | ✓ | HMAC key | R |
| | | ✓ | ✓ | AES-CMAC | ✓ | AES key | R |
| MAC Verification | MAC | ✓ | ✓ | HMAC-SHA-1 | ✓ | HMAC key | R |
| | | ✓ | ✓ | HMAC-SHA-224, HMAC-SHA-256 | ✓ | HMAC key | R |
| | | ✓ | ✓ | HMAC-SHA-384, HMAC-SHA-512 | ✓ | HMAC key | R |
| | | ✓ | ✓ | AES-CMAC | ✓ | AES key | R |
| | | ✓ | ✓ | AES-CBC-MAC | | AES key | R |
| ECDH/ECDSA key verification | Public Key | ✓ | ✓ | ECDH | ✓ | EC parameters<br>ECDH private key *(optional)*<br>ECDH public key *(optional)* | R<br>R<br>R |
| | | ✓ | ✓ | ECDSA | ✓ | EC parameters<br>ECDSA private key *(optional)*<br>ECDSA public key *(optional)* | R<br>R<br>R |
| ECDSA Sig Gen | Public Key | ✓ | ✓ | ECDSA (P-224, P-256, P-384, P-521), SHA-224, SHA-256, SHA-384, SHA-512 | ✓ | EC parameters<br>ECDSA private key | R<br>R |
| ECDSA Sig Verify | Public Key | ✓ | ✓ | ECDSA (P-192, P-224, P-256, P-384, P-521)<br>SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 | ✓ | EC parameters<br>ECDSA public key | R<br>R |
| RSA Sig Gen | Public Key | ✓ | ✓ | RSA-PSS and RSA-PKCS#1-v1.5 (n=2048, 3072), SHA-224, SHA-256, SHA-384, SHA-512 | ✓ | RSA-PSS private key<br>RSA-v1.5 private key | R |

| Service | Input Token | Roles | | Cryptographic Algorithms | FIPS | Keys and CSPs | Access |
|---|---|---|---|---|---|---|---|
| | | User | CO | | | | |
| RSA Sig Verify | Public Key | ✓ | ✓ | RSA-PSS and RSA-PKCS#1-v1.5<br>(n=1024 to 3072),<br>SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 | ✓ | RSA-PSS public key<br>RSA-v1.5 public key | R |
| ECDH/ECDSA generate public key | Public Key | ✓ | ✓ | ECDSA (P-224, P-256, P-384, P-521) | | ECDH/ECDSA private key<br>EC parameters<br>ECDH/ECDSA public key | R<br>R<br>C |
| ECDH/ECDSA generate private and public key | Public Key | ✓ | ✓ | ECDSA (P-224, P-256, P-384, P-521), | | ECDH/ECDSA private key<br>EC parameters<br>ECDH/ECDSA public key | C<br>R<br>C |
| ECDH generate shared secrets (single key-pair) | Public Key | ✓ | ✓ | ECDH (P-224, P-256, P-384, P-521),<br>CTR_DRBG | ✓ | ECDH private key<br>EC parameters<br>ECDH public key<br>ECDH shared secret | R<br>R<br>R<br>C |
| ECDH key agreement (single key-pair) | Public Key | ✓ | ✓ | ECDH (P-224, P-256, P-384, P-521), CTR_DRBG, one-step KDF [SP800-56C], section 4.1 (SHA-256) | ✓ | ECDH private key<br>EC parameters<br>ECDH public key<br>Derived Key | R<br>R<br>R<br>C |
| ECDH generate shared secrets (dual key-pair) | Public Key | ✓ | ✓ | ECDH (P-224, P-256, P-384, P-521),<br>CTR_DRBG | ✓ | ECDH private key (2x)<br>EC parameters<br>ECDH public key (2x)<br>ECDH shared secret | R<br>R<br>R<br>C |
| ECDH key agreement (dual key-pair) | Public Key | ✓ | ✓ | ECDH (P-224, P-256, P-384, P-521), CTR_DRBG, one-step KDF [SP800-56C], section 4.1 (SHA-256) | ✓ | ECDH private key (2x)<br>EC parameters<br>ECDH public key (2x)<br>Derived Key | R<br>R<br>R<br>C |
| AES Key wrapping | AES (Un)Wrap | ✓ | ✓ | AES Key Wrap with or without padding as specified in [SP800-38F] | ✓ | AES key | R/R* |
| AES Key unwrapping | AES (Un)Wrap | ✓ | ✓ | AES Key Wrap with or without padding as specified in [SP800-38F] | ✓ | AES key | R/R* |
| TRNG Configuration | TRNG Configuration | | ✓ | | ✓ | n/a | |
| Get TRNG Random Number | TRNG Get Random Number | ✓ | ✓ | TRNG | ✓ | Raw random data to seed DRBG | C* |

| Service | Input Token | Roles | | Cryptographic Algorithms | FIPS | Keys and CSPs | Access |
|---|---|---|---|---|---|---|---|
| | | User | CO | | | | |
| Get DRBG Random Number | TRNG Get Random Number | ✓ | ✓ | CTR_DRBG | ✓ | Entropy Input<br>Internal DRBG state (seed, V and key) | R<br>RU |
| TRNG Post-processing Verification | TRNG Post-Processing Verification | ✓ | ✓ | | ✓ | n/a | |
| TRNG Hardware Self-test Verification | TRNG Hardware Self-test Verification | ✓ | ✓ | | ✓ | n/a | |
| Dynamic Asset Creation | Asset Create | ✓ | ✓ | | ✓ | Asset | C |
| Static Asset Search | Static Asset Search | ✓ | ✓ | | ✓ | n/a | |
| Dynamic Asset Key Derivation | Asset Load (derive) | ✓ | ✓ | SP800-108 KDF<br>SP800-56C KDF | ✓ | Asset<br>Key Derivation Key (KDK) | U<br>R |
| Dynamic Asset Import as Key Blob | Asset Load (import) | ✓ | ✓ | AES-CMAC, AES-CTR | ✓ | Asset<br>Key Encryption Key (KEK) | U<br>R |
| Dynamic Asset Import as AES wrapped Key Blob | Asset Load (AES Unwrap) | ✓ | ✓ | AES-KW, AES-KWP | ✓ | Asset<br>AES Wrap Key | U<br>R |
| Dynamic Asset Generation | Asset Load (random) | ✓ | ✓ | CTR_DRBG, AES-CMAC, AES-CTR | ✓ | Asset<br>Key Encryption Key (KEK)[3] | C<br>R |
| Dynamic Asset Import as plaintext | Asset Load (plaintext) | ✓ | ✓ | AES-CMAC, AES-CTR | ✓ | Asset<br>Key Encryption Key (KEK)[3] | U<br>R |
| Dynamic Asset Deletion | Asset Delete | ✓ | ✓ | | ✓ | Asset | D |
| Public Data Read | Public Data Read | ✓ | ✓ | | ✓ | Asset | R* |
| Monotonic Counter Read | Monotonic Counter Read | ✓ | ✓ | | ✓ | Asset | R* |
| Monotonic Counter Increment | Monotonic Counter Increment | ✓ | ✓ | | ✓ | Asset | RU |
| OTP Data Write | One-Time-Programmable Data Write | | ✓ | | ✓ | Asset<br>Provisioning Key | CU<br>R |

[3] The Key Encryption Key (KEK) is required when the asset needs to be exported as a Key Blob.

| Service | Input Token | Roles | | Cryptographic Algorithms | FIPS | Keys and CSPs | Access |
|---|---|---|---|---|---|---|---|
| | | User | CO | | | | |
| Generate Random HUK | Provision Random HUK | | ✓ | DRBG | ✓ | Trusted Root Key (HUK)<br>Crypto Officer Identity | C<br>R |
| System Information | System Information | ✓ | ✓ | | ✓ | n/a | |
| On Demand Self-Tests | Self-Test | ✓ | ✓ | All relevant FIPS algorithms | ✓ | Asset Store | D |
| Module Reset | Hard Reset | ✓ | ✓ | | ✓ | Asset Store | D |
| Authenticated Unlock Start | Authenticated Unlock Start | ✓ | ✓ | DRBG | ✓ | RSA Authentication Key<br>Authentication State | R<br>U |
| Authenticated Unlock Verify | Authenticated Unlock Verify | ✓ | ✓ | ECDSA P-256 | ✓ | RSA Authentication Key<br>Authentication State | R<br>RU |
| Activate / Deactivate Debug port signals | Set Secure Debug | ✓ | ✓ | | ✓ | RSA Authentication Key<br>Authentication State | R<br>R |
| Register Read | Register Read | ✓ | ✓ | | ✓ | n/a | |
| Register Write | Register Write | | ✓ | | ✓ | n/a | |
| Zeroize Output Mailbox | Zeroize Output Mailbox | ✓ | ✓ | | ✓ | CSPs in output mailbox | D |
| Create / Update User Identity | Define Users | | ✓ | | ✓ | User Identity | CU |
| Select OTP Zeroize | Select One-Time-Programmable Zeroize | | ✓ | | ✓ | n/a | |
| Zeroize OTP | Zeroize One-Time-Programmable | | ✓ | | ✓ | CSPs in the OTP and Asset Store | D |
| Go to Sleep mode | Sleep Mode | ✓ | ✓ | | ✓ | n/a | |
| Resume from Sleep mode | Resume from Sleep | ✓ | ✓ | | ✓ | n/a | |
| Set the System timer | Set System Time | | ✓ | | ✓ | n/a | |

**Table 11 - MS1201 Security Sub-system Services**

The following table shows the FIPS-Approved algorithms. All FIPS-approved algorithms are validated under the CAVP with certificate number C1900.

*Note: Not all of the algorithms/modes verified through the CAVP certificates are available from the module.*

| Algorithm | Usage | Key lengths | Modes | Standards |
|---|---|---|---|---|
| AES | Encryption<br>Decryption | 128, 192, 256 bits | ECB, CBC, CTR | [FIPS197]<br>[SP800-38A] |
| | Encryption<br>Decryption | 128, 192, 256 bits | CCM | [SP800-38C] |
| | MAC | 128, 192, 256 bits | CMAC | [SP800-38B]<br>[SP800-38D] |
| AES Key Wrapping | Key Wrapping (KTS) | 128, 192, 256 bits | KW, KWP | [SP800-38F]<br>[RFC3394]<br>[RFC5649] |
| AES Key Wrapping with AES-CMAC and AES-CTR | Key Wrapping (KTS) | 256-bit AES keys | AES-CMAC and AES-CTR | [SP800-38B]<br>[SP800-38A] |
| DRBG | Random Number Generation | | AES-256 in CTR mode, no derivation function, prediction resistance disabled | [SP800-90A]<br>[SP800-38A] |
| ECDSA | Key Verification<br>Signature Verification | P-192, P-224, P-256, P-384, P-521 | | [FIPS186-4] |
| | Signature Generation | P-224, P-256, P-384, P-521 | | [FIPS186-4] |
| KAS ECC CDH Component (CVL)<br>(approved per IG D.8 scenario 5) | Shared Secret Computation used in Key Agreement Scheme | P-224, P-256, P-384, P-521 | | [SP800-56A] Section 5.7.1.2 |
| HMAC-SHA-1 | MAC Verification | 112-512 bits | | [FIPS198-1] |
| HMAC-SHA-224<br>HMAC-SHA-256<br>HMAC-SHA-384<br>HMAC-SHA-512 | MAC Generation<br>MAC Verification | 112-512 bits<br>128-512 bits<br>128-512 bits<br>128-512 bits | | [FIPS198-1] |
| SP800-108 KDF | Key Derivation | 128-512 bits | Counter and feedback modes using CMAC-AES-256 and HMAC-SHA-256 | [SP800-108]<br>[FIPS198-1]<br>[SP800-38B] |
| SP800-56C KDF<br>(KDA, vendor affirmed) | Key Derivation | 128-512 bits | two-step KDF, Counter and feedback modes using CMAC-AES-256 and HMAC-SHA-256 | [SP800-56C]<br>[SP800-108]<br>[FIPS198-1]<br>[SP800-38B] |
| Key Generation<br>(vendor affirmed) | Cryptographic Key Generation (CKG) | 128, 192, 256 bit AES key<br>112-512 bit HMAC key | | [SP800-133] |

| Algorithm | Usage | Key lengths | Modes | Standards |
|---|---|---|---|---|
| RSA | Signature Verification | n=(1024 to 3072) | RSA-PSS (no CRT) | [PKCS#1] [FIPS186-4] |
| | Signature Generation | n=(2048 and 3072) | | |
| | Signature Verification | n=(1024 to 3072) | RSA-PKCS#1v1.5 (no CRT) | |
| | Signature Generation | n=(2048 to 3072) | | |
| SHA-1 | Message Digest | | | [FIPS180-4] |
| SHA-224, SHA-256, SHA-384, SHA-512 | Message Digest | | | [FIPS180-4] |

**Table 12 - FIPS approved cryptographic algorithms**

The following table shows the FIPS-Allowed algorithms.

| Algorithm | Usage | Key lengths | Modes | Standards |
|---|---|---|---|---|
| ECDH (allowed per IG D.8 scenario 3) | Key Agreement | P-224, P-256, P-384, P-521 | one-step KDF using SHA-256 | [SP800-56A] section 5.7.1.2 [SP800-56C Rev1] section 4.1 |
| NDRNG (non-approved but allowed per IG 7.15) | Non-Deterministic Random Number Generator | N/A | N/A | N/A |

**Table 12a: FIPS Non-Approved but Allowed algorithms**

The following table shows the cryptographic algorithms and their key lengths that are not FIPS-Approved.

| Algorithm | Usage | Key lengths | Modes (CAVP Cert. ) |
|---|---|---|---|
| AES | Encryption Decryption | 128, 192, 256 bits | GCM |
| | | 128, 256 bits | XTS (C1900) |
| ECDSA | Key Generation | P-224, P-256, P-384, P-521 | (C1900) |
| ECDH | Key Agreement with Ephemeral Unified, Full Unified, One Pass DH, One Pass Unified, Static Unified | P-224, P-256 | SHA-256 (C1900) |
| Triple-DES | Encryption Decryption | 192 bits | ECB, CBC (C1900) |

**Table 13 – Non-Approved FIPS algorithms**

The MS1201 Security Sub-system also supports the following algorithms that are used for firmware and data integrity:

| Algorithm | Usage |
|---|---|
| CRC-24 | ROM Firmware Integrity Test |
| CRC-32 | Asset Integrity Test |

**Table 14 - Other algorithms**

When a CSP is loaded or updated in the asset store, the MS1201 Security Sub-system generates a CRC-32 checksum. Whenever the CSP is used by a service (referenced through its asset ID), the MS1201 Security Sub-system verifies the integrity of the CSP comparing the existing and re-calculated checksums.

## 4.3 Identification and Authentication

Role-based identification is indicated through the use of a single bit hardware signal (prot_acc_n) which is provided as side band information for all services. When the input token is written (the token data is available on the data input bus) the sideband signal must be valid.

Role-based authentication is performed through the use of a 32-bit identity value provided in the input token for all services. The MS1201 Security Sub-system requires the identification and authentication of the user and Crypto Officer roles for each service request; role identification and authentication status is not internally maintained in the cryptographic module.

The module is initialized via the “Provision Random HUK” token, which instructs the module to generate the Trusted Root key (HUK) and assigns the Crypto Officer 32-bit ID. The module supports only one Crypto Officer role identity; this value is stored in One Time Programmable (OTP) memory and cannot be altered unless the whole module is zeroized.

To access the module to request services, the Crypto Officer role is explicitly selected using the role selection bit (prot_acc_n=0b). For each individual token received with the Crypto Officer role selected, the MS1201 Security Sub- system compares the 32-bit ID field provided in the input token with a predefined 32-bit value stored inside the OTP of the MS1201 Security Sub-system (this identity value is pre-defined by vendor and loaded to OTP during the module initialization). If the comparison succeeds, then the input token is processed. Otherwise, the service request is rejected indicating the error in the output token.

Likewise, the User role is explicitly selected using the role selection bit (prot_acc_n=1b). For each individual token received with the user role selected, the MS1201 Security Sub-system compares the 32-bit ID field provided in the input token with the stored user IDs (up to four) inside the MS1201 Security Sub-system. If the identity matches one of the stored user IDs, access is granted and the input token is processed. Otherwise, the request is rejected indicating the error in the output token. User role identities are created by the Crypto Officer role and reside in volatile memory.

If the authentication fails, the MS1201 Security Sub-system waits at least 15ms (minimum value set for the highest chip frequency) before it returns the output token rejecting the service request.

For the User role, the Define Users token allows the Crypto Officer to create or modify up to four identities, whose values are stored internally in MS1201 Security Sub-system's internal memory. These identities do not survive a power-cycle; the Crypto Officer must define the users again anytime the MS1201 Security Sub-system is reset or powered-up and invoke “Self-Test” token to initialize user authentication.

## 4.4 Mechanism and Strength of Authentication

The probability of successfully guessing the Crypto Officer Identity is: $P_{co} = \frac{1}{2^{32}}$.

and the probability of successfully guessing one of the four User Identities is: $P_u = \left(4 \times \frac{1}{2^{32}}\right)$ .

In both cases, the FIPS 140-2 requirements are satisfied: the probability is less than $1/1{,}000{,}000$ that a random attempt will succeed or a false acceptance occurring since

$$P_{co} = \frac{1}{2^{32}} < \frac{1}{1{,}000{,}000} \quad \text{and} \quad P_u = \frac{4}{2^{32}} < \frac{1}{1{,}000{,}000}$$

In addition, since there is a $15ms$ delay for a failed authentication, the MS1201 Security Sub-system can process at most 4000 consecutive failed authentication attempts every minute since

$$4000\ attempts = {}^{60s}/_{0.015s}$$

This means, in the case of the Crypto Officer identity, the overall success rate is equal to

$$4000 \times P_{co} = \frac{4000}{2^{32}} \approx \frac{1}{1{,}073{,}742} \quad \text{which is } strictly\ less\ than\ \frac{1}{100{,}000}.$$

Likewise in the case of the User identity, the overall success rate is

$$4000 \times P_u = \frac{16000}{2^{32}} \approx \frac{1}{268{,}435} \text{ which is } strictly\ less\ than\ \frac{1}{100{,}000}.$$

In both scenarios, the FIPS 140-2 requirement is meet: the probability of multiple attempts within a minute is less than $1/100{,}000$.

## 4.5 Authentication Data protection

The Crypto Officer Identity is stored in the OTP and cannot be modified. The four identities assigned to users with the User role reside in internal memory and can be only created or updated by the Crypto Officer, who needs to authenticate with the correct Identity.

The user role identities are zeroized when the MS1201 Security Sub-system is powered-off; the Crypto Officer Identity remains in the OTP but can be zeroized through OTP zeroization.

No authentication data can be output by any of the available services.

# 5 Physical Security

For the purpose of this validation, the MS1201 Security Sub-system was synthesized in silicon as part of the MS1201 single-chip embodiment. This single chip that includes standard passivation layer is covered by a production-grade opaque coating hard material. The coating material serves as a protective shell around the processed silicon providing opacity in the visible spectrum and preventing any access to the interior of the chip. The single-chip conforms to Level 3 requirements for physical security.

# 6 Operational Environment

The module operates in a non-modifiable environment.

# 7 Cryptographic Key and CSP Management

The Asset Store provides the core mechanism for secure handling of key material and other sensitive data like the IV, digest, MAC and state. The Asset Store is designed to store asset objects and allows them to be used, while never revealing their contents outside of the MS1201 Security Sub-system. The Asset Store identifies the following three types of assets:

- *Static Asset*: key material that is available immediately after power-up and is located in the OTP of the module. This asset cannot be modified or output (nevertheless, the whole OTP can be zeroized by the Crypto-Officer, see section 7.6).
- *Dynamic Asset*: key material or any other sensitive data like the IV, digest, MAC and state. This asset needs to be loaded into the Asset Store before its use because it is located in the internal Data RAM memory of module and set to zero at powered-off. This asset cannot be modified, only deleted from the asset store.
- *Public data object*: data that can be retrieved from the Asset Store in plaintext for use outside the module and can reside either in the OTP or the RAM. This asset cannot be modified, only deleted from the asset store.

Static assets are used in the same way as dynamic assets, the only difference is their lifespan. Both are stored in plaintext within the MS1201 Security Sub-system and protected from disclosure and modification.

Whenever an asset is loaded or updated, the MS1201 Security Sub-system generates a CRC-32 checksum for that asset; a read of an asset will verify its integrity re-computing the CRC-32 checksum and comparing it to the stored checksum.

The following table summarizes the cryptographic keys used in the MS1201 Security Sub-system with the key lengths supported, the available methods for key generation, entry and output and the way they are stored. Notice that for Key Entry:

- "Firmware" means that the keys are determined during the delivery process and are unique to a given customer. See section 7.2.2 for more information.
- "Key Blob" is a block of binary data encrypted through a key wrapping method using the AES-CMAC and AES-CTR algorithms. See section 7.2.1 for more information.

| Name | Key Length[4] / CSP Size | Generation | | | | Entry | | | | | Storage | | Output |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | KDF SP800-108 | KDF SP800-56C | Random | Key Pair | Firmware | OTP | Plaintext | AES- WrapKey | Key Blob | Static | Dynamic | Key Blob |
| Trusted Root Key (HUK) | 128, 256 bits | | | ✓ | | | ✓ | | | ✓ | ✓ | | ✓ |
| Trusted Key Encryption Keys (KEK) | 256 bits | ✓ | | | | | | | | | ✓ | ✓ | |
| Trusted Key Derivation Keys (KDK) | 128, 256 bits | ✓ | | | | | | | | | ✓ | ✓ | |
| Authentication Keys (RSA public key) | 2048-3072 bits | | | | | ✓ | | | | | ✓ | | |

[4] Not all key lengths are allowed in FIPS mode of operation. See section 4.2 for more information.

| | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Provisioning Key | 2x256 bits | | | | | ✓ | | | | | ✓ | | |
| AES keys | 128, 192, 256 bits | ✓ | ✓ | ✓ | | | | ✓ | ✓ | ✓ | | ✓ | ✓ |
| HMAC keys | SHA-1: 112-512 bits<br>SHA-224: 112-512 bits<br>SHA-256: 128-512 bits<br>SHA-384: 1024 bits<br>SHA-512: 1024 bits | ✓ | ✓ | ✓ | | | | ✓ | ✓ | ✓ | | ✓ | ✓ |
| RSA key pairs | 2048-3072 bits | | | | | | | ✓ | ✓ | ✓ | | ✓ | ✓ |
| ECDSA key pairs | 224-521 bits | | | | | | | ✓ | ✓ | ✓ | | ✓ | ✓ |
| EC Diffie-Hellman key pairs | 224-521 bits | | | | | | | ✓ | ✓ | ✓ | | ✓ | ✓ |
| Crypto Officer Role | 32 bits | | | | | ✓ | | ✓ | | | ✓ | | |
| User Identity | 32 bits | | | | | | | ✓ | | | | ✓ | |
| Entropy input string | | | | | | | | | | | | ✓ | |
| DRBG internal state (seed, V and key) | | | | | | | | | | | | ✓ | |

**Table 15 - Life Cycle of Keys and Critical Security Parameters (CSPs)**

The Provisioning Key is a key that is available for OTP initialization only. When the complete OTP is initialized the Provisioning Key cannot be used anymore.

# 7.1 Key Generation

The MS1201 Security Sub-system provides services for generating symmetric keys. The key generation methods implemented in the module are compliant with [SP800-133] (CKG - vendor affirmed).

The MS1201 Security Sub-system implements symmetric key generation for AES and HMAC keys using random data obtained from a Deterministic Random Bit Generator (DRBG) compliant with [SP800-90A].

Intermediate key generation values are not output from the cryptographic module during or after processing the service.

## 7.1.1 Key Derivation

The MS1201 Security Sub-system provides services for deriving keys from the Trusted Root Key (HUK), or any asset indicated as a Trusted Key Derivation Key (KDK). The MS1201 Security Sub-system provides key-based derivation in compliance with [SP800-108] and [SP800-56C] (vendor affirmed).

# 7.2 Key Entry and Output

Electronic key entry method is used to import the private/secret keys; there is no manual key import or export method used in the MS1201 Security Sub-system.

### 7.2.1 Dynamic assets

Dynamic assets can be input into the MS1201 Security Sub-system through the following methods:

- Load plaintext from the Host (*Asset Load Plaintext* Token).
- AES Key Wrapping as specified in NIST SP 800-38F (*Asset Load AES-Wrap* Token).
- AES Key Wrapping with AES-CMAC, AES-CTR (*Asset Load Import* Token).

Keys can also be initialized in the asset store using the built-in key generation features provided by the MS1201 Security Sub-system (see section 7.1).

When plaintext or random number loaded assets must survive a power cycle, they must be stored outside the module. The Asset Store is able to generate a block of binary data known as a Key Blob in which the asset is exported to the Host. MS1201 Security Sub-system utilizes the AES-CMAC, AES-CTR algorithms to protect the asset inside the Key Blob from disclosure and modification, using a Trusted Key Encryption Key (KEK). An important part of the protection provided by the Asset Store is to allow a Key Blob to be created only once, when the asset is filled using the Asset Load Plaintext (key provided in plaintext by the host) or Asset Load Random (key generated by the module using the DRBG) services.

### 7.2.2 Static assets

The MS1201 Security Sub-system provides functionality to program static assets or public data objects into the OTP via a special service only available to the Crypto Officer role and using an AES Key Blob utilizing AES-CMAC and AES-CTR that is intended for provisioning. The provisioning Key Blob can be created outside MS1201 Security Sub-system using a special shared secret. Asset Ownership (see section 7.3) is covered through the policy of assets that are intended to be written to OTP.

Static assets stored in OTP cannot be output. The OTP can also contain Public data objects which are accessible through its identifier.

Additionally, the MS1201 Security Sub-system includes two types of keys that are stored in the Firmware and can be input during the integration of the cryptographic module:

- Two RSA Authentication Public Keys with 2048 and 3072 bits.
- One 512-bit AES Provisioning Key (AES-CMAC, AES-CTR).

Note: The program RAM area for these keys in the Firmware is not considered for the integrity verification of the Firmware component of the MS1201 Security Sub-system.

## 7.3 Key access control and usage

The MS1201 Security Sub-system manages the concept of asset ownership and usage policy based on the following information provided during asset creation:

- Host ID: host that owns the asset
- Protection bit: indicates whether the user is a Crypto Officer or User role
- Identity: user ID that owns the asset
- Usage Policy: defines how the asset may be used (i.e. algorithm, mode and cryptographic operation)

Once created, the ownership and usage attributes of the key remains until the key is deleted from the asset store or the asset store is zeroized.

## 7.4 Key Agreement / Key Transport

The MS1201 Security Sub-system provides:

- ECDH shared secret computation [SP800-56A] followed by one-step key derivation function compliant with section 4.1 of [SP800-56C] key agreement scheme in compliance with scenario 3 of IG D.8.
- EC Diffie-Hellman shared secret computation, compliant with SP800-56A and scenario 5 primitive only of IG D.8.

These key agreement schemes provide between 112 and 256 bits of security strength.

The MS1201 Security Sub-system also provides key wrapping. As shown in Table 15 and explained in section Dynamic assets, keys can be entered in encrypted form through the following key wrapping methods and key lengths:

- Key Wrapping with AES in KW and KWP modes with 128, 192 or 256-bit keys, compliant with [SP800- 38F]. Security strength ranges from 128 to 256 bits, according to the AES key length.
- Key Wrapping using AES in CMAC and CTR modes with two 256-bit keys (the first key for the AES-CMAC operation and the second key for the AES-CTR operation), compliant with [SP800-38F]. It provides a security strength of 256 bits.

The following table shows the maximum lengths and security strength of the keys that can be imported to and exported from the MS1201 Security Sub-system using the key agreement or wrapping services:

| Key | Maximum Length | Security Strength |
|---|---|---|
| Trusted Root Key | 256 bits | 256 bits |
| AES key | 256 bits | 256 bits |
| HMAC key | 1024 bits | 256 bits |
| RSA key pair | 3072 bits | 128 bits |
| ECDSA key pair | 521 bits | 256 bits |
| EC Diffie-Hellman key pair | 521 bits | 256 bits |

**Table 16 - Security Strength of Cryptographic Keys**

The maximum strength of the key wrapping schemes provided by the MS1201 Security Sub-system is greater than or equal to the security strength of the keys that need to be wrapped while entering or exiting the module. The maximum strength of the key agreement scheme provided by the MS1201 Security Sub-system is 256 bits.

Nevertheless, it is the user’s responsibility to use the establishment method with an appropriate key size to ensure the FIPS compliance. Using an insufficient AES key size for AES Key Wrapping or an insufficient ECDH key size for key agreement will reduce the security strength of the wrapped, agreed upon key.

## 7.5 Key / CSP Zeroization

A dynamic asset (key or CSP) is deleted from the asset store using the *Asset Delete* service and the memory where the asset was stored is zeroized.

Additionally, the whole asset store is zeroized when the MS1201 Security Sub-system transitions to the error state or when the module is powered-off.

Keys and CSPs considered static assets are stored in OTP; the MS1201 Security Sub-system provides a two-step process to zeroize the OTP memory. First, the service *Select One-Time-Programmable Zeroize* must be called to enable OTP zeroization. After OTP zeroization is enabled, the Zeroize *One-Time-Programmable* service fills the complete OTP with ones, and zeroizes the asset store.

The MS1201 Security Sub-system also provides the *Zeroize Output Mailbox* service can be used to zeroize the output mailbox before the mailbox is unlinked. This operation prevents sensitive material leaking to other Hosts applications.

Zeroization is performed filling the memory area with zeroes. The operation is performed in a time that is not sufficient to compromise CSPs. Additionally:

- The MS1201 Security Sub-system processes one input message at a time, when a zeroization service (Asset Delete, Zeroize Output Mailbox) is processed no other input message accessing the asset store can be executed.
- No information is output when the MS1201 Security Sub-system transitions to or is in the Error state.

## 7.6 Random Number Generation

The MS1201 Security Sub-system includes a Deterministic Random Bit Generator (DRBG) based on the CTR_DRBG (without prediction resistance; without derivation function) algorithm and AES-256 as the underlying cipher according to [SP800-90A]. The MS1201 Security Sub-system uses this engine to:

- Create symmetric keys in the asset store for the *Asset Load Random* Token.
- Provide random data for the *TRNG Get Random* service.

The MS1201 Security Sub-system includes a True Random Number Generator (TRNG) engine to provide entropy input to the DRBG. The Crypto Officer can seed or re-seed the DRBG using the *TRNG Configuration* service (the service can also start the TRNG engine in the same service).

It is also possible to trigger an automatic re-seed of the DRBG using the same service when requesting sufficient random data; in this case the operation can be performed by both the Crypto Officer and the User roles.

The DRBG is seeded with 384 bits from NDRNG and provides 256 bits of security strength .

MS1201 Security Sub-system performs continuous tests in the DRBG and TRNG engines, verifying that the previous and current generated blocks of random data are not equal (section 9.3). The module also performs DRBG health tests according to section 11.3 of SP 800-90A.

## 7.7 True Random Number Generation

The True Random Number Generator (TRNG) engine is a Non-Deterministic Random Number generator (NDRNG) containing a hardware entropy source based on free-running ring oscillators. This TRNG utilizes eight Free-Running Ring Oscillators (FROs) to supply the entropy needed to generate true random numbers.

The set of FROs provided in the Verilog RTL is ready for synthesis after instantiating cells from the chosen target technology. However, it is possible to optimize the FROs for the specific target technology such that more entropy is generated. Customers should follow the instructions provided in the MS1201 Security Sub-system Integration Manual for this purpose.

Using the *TRNG Configuration* service, the Crypto Officer can configure and start the TRNG engine to initialize the DRBG.

# 8 Electromagnetic Interference / Electromagnetic Compatibility (EMI/EMC)

According to 47 Code of Federal Regulations, Part 15, Subpart B, Unintentional Radiators, the MS1201 Security Sub- system is not subject to EMI/EMC regulations because it is a subassembly that is sold to an equipment manufacturer for further fabrication. That manufacturer is responsible for obtaining the necessary authorization for the equipment with the MS1201 Security Sub-system embedded prior to further marketing to a vendor or to a user.

# 9 Self-Tests

## 9.1 Power-Up Tests

After successful initialization of the SoC in which MS1201 Security Sub-system is integrated, the MS1201 Security Sub-system automatically performs power-up tests without user intervention to ensure that the module is not corrupted and that the cryptographic algorithms within the module work as expected. During the execution of the power-up tests, services are not available and no data output is possible.

If the power-up tests succeed, then the MS1201 Security Sub-system becomes operational with the following status signals: *fatal_error* =0b.

If the power-up tests fail, then the MS1201 Security Sub-system transitions to the Error state and becomes non-operational with status signals: *fatal_error* =1b.

### 9.1.1 Integrity tests

The MS1201 Security Sub-system verifies the integrity of the firmware in ROM using a dedicated 24-bit CRC algorithm. After power-up, the MS1201 Security Sub-system calculates the CRC of the firmware image and compares it against the last word of the firmware code image. If the CRC verification succeeds, the MS1201 Security Sub- system continues with the rest of the power-up tests; if the CRC verification fails, the cryptographic module sets the fatal_error signal and transitions into the Error state, becoming non-operational.

### 9.1.2 Cryptographic Algorithm tests

Once the firmware integrity have been successfully verified, the self-tests (listed below) are automatically executed in preparation for the execution of the boot process. If any of the known answer tests fail (i.e. the calculated output does not equal the known answer), the MS1201 Security Sub-system transitions to the Error state and becomes non-operational.

| Cryptographic Algorithm | Test |
|---|---|
| AES | KAT AES-CBC, 128-bit, encryption<br>KAT AES-CBC, 128-bit, decryption<br>KAT AES-CCM, 192-bit, encryption<br>KAT AES-CCM, 192-bit, decryption<br>KAT AES-CMAC, 256-bit, MAC generation |
| SHS | KAT SHA-1<br>KAT SHA-224<br>KAT SHA-384<br>KAT SHA-512 |
| HMAC | KAT HMAC-SHA-256 |
| RSA | KAT RSA 2048-bit (PKCS#1 v1.5), signature generation<br>KAT RSA 2048-bit (PKCS#1 v1.5), signature verification |
| ECDH | KAT for Z computation (NIST P-224)<br>HMAC-SHA-256, AES-CMAC KAT (for Key Derivation Function). |

| | |
|---|---|
| ECDSA | KAT ECDSA (NIST P-224) signature generation<br>KAT ECDSA (NIST P-224) signature verification |
| DRBG | KAT AES-CTR-256 DRBG |
| KBKDF | KAT SP800-108 KDF (PRF HMAC-SHA-256, AES-CMAC KAT) |
| CRC32 | KAT CRC32 |

**Table 17 - Power-Up Tests**

## 9.2 On-demand self-tests

In order to perform the on-demand self-tests as required by FIPS 140-2, the MS1201 Security Sub-system shall be power-off and power-on by doing a hard reset.

## 9.3 Conditional Tests

MS1201 Security Sub-system performs conditional tests on the cryptographic algorithms shown in the following table:

| Algorithm | Test |
|---|---|
| DRBG | Continuous test |
| TRNG | Continuous test |

**Table 18 - Conditional Tests**

If any of these tests fail, MS1201 Security Sub-system transitions to the Error state and becomes non-operational with the following status signals: *fatal_error* =1b

## 9.4 Module status

MS1201 Security Sub-system provides different interfaces to show its status:

- The *fatal_error* output signal (equivalent to bits 31 of the MODULE_STATUS register accessible by hosts) indicates whether the cryptographic module is in an Error state. This information is available to any user.
- The *System Info* service provides hardware and firmware version information and eventual OTP anomalies. This service can be requested by both the User and Crypto-Officer roles.
- The output token returned by MS1201 Security Sub-system provides the result of processing the service requested by the User or Crypto-Officer role.

## 9.5 Error state

When a fatal error occurs, MS1201 Security Sub-system transitions to the Error state and becomes non-operational, and the Asset Store is zeroized. The module can transition to this state for the following reasons:

- Failure of the power-up tests (including failure of the integrity test) or on-demand self-tests.
- Failure when resuming from Sleep.
- Failure of the conditional tests.
- Failure of the True Random Number Generator (TRNG).
- DMA errors.

In the Error state, services are not available and no data output is possible with the exception of the System Info and Reset services. If the Error state is caused by integrity tests, the System Info and Reset services cannot be used.

There are two options to clear the Error state and bring MS1201 Security Sub-system back to an operational state:

- Reset the module.
- Power-off and power-on the module.

# 10 Design Assurance

## 10.1 Configuration Management

eWBM uses a variety of tools for configuration management. These are listed below.

| Version Control Tool | Objects |
|---|---|
| Git version control | Software/firmware, source code, test tools and verification scripts |
| SVN (Subversion version control) | RTL source |
| Google Drive built-in version control feature | All Documentation such as user manuals, integration manuals, etc. |

**Table 19 - Change and Version Control Tools**

### 10.1.1 Cryptographic Module Identification

MS1201 Security Sub-system is uniquely identified by the filenames used to ship the IP core Verilog code and the firmware image. The following convention is used:

<nn>[<o>]_HW<x.y.z> [_(alpha|beta|final)]

<nn>[<o>]_Firmware[_cfg<O>]_FW<a.b.c> [_(alpha|beta|final)]

where:

- <nn> is the “EIP number” in eWBM’s IP catalog.
- <o> or <O> is a code for specific configuration options, if any. Please see the Data Sheet or Hardware Reference Manual for a detailed explanation of this code and the Release Notes to see the configuration details for this delivery.
- <x.y.z> is the hardware version number of the IP in this delivery. The 3rd digit is only used for patches to the original <x.y> version.
- <a.b.c> is the firmware version number included in this delivery. The 3rd digit is only used for patches to the original <a.b> version.
- (alpha|beta|final) is used to distinguish intermediate drops leading towards the final release. “_final” is omitted if no intermediate drops are done or if it is a generic release.

### 10.1.2 Guidance Identification

eWBM uniquely identifies each document with the document name, document number (e.g. 007-130300-201) and Revision (RevA, RevB, etc.).

### 10.1.3 Source Code Identification

The source code is identified and controlled by the configuration management tools as listed in Table 19 . Proper keywords are included in each source code file to provide the filename and revision configuration item.

## 10.2 Delivery and Operation

MS1201 Security Sub-system is synthesized in silicon within the MS1201 embodiment. It is a single chip hardware module. The chip is delivered from the vendor via a trusted delivery courier. Upon reception of MS1201 Security Sub-system, the customer should verify that the package does not have any irregular tears or openings.

The chip comes preloaded with the following code packages:

- 915-130017-220_VaultIP-130-018_HW2.2.0.zip
- 914-130017-230_VaultIP-130-018_Firmware-cfgA_FW2.5.3.zip

The Crypto Officer ID, which is embedded in the Firmware, will be provided in the package and the *Define User* token can be used to create the user roles. On power-up, the module automatically performs power-up self-tests; successful completion of the integrity checks within the power-up tests ensures the integrity of MS1201 Security Sub-system. The *System Info* service can be used to verify the version number of the HW and FW components of the module.

## 10.3 Guidance

For the purpose of this cryptographic module validation, MS1201 Security Sub-system is synthesized in silicon within the MS1201 embodiment and validated as a single-chip hardware module.

eWBM provides the following documentation describing the functionality of MS1201 Security Sub-system:

| Document Name | Document Number | Revision |
|---|---|---|
| EIP-130-HW2.2 Security Module, Integration Manual | 007-130220-200 | RevA |
| VaultIP Security Module, Software Integration Manual | 007-130250-312 | RevA |
| VaultIP Security Module, Hardware Reference Manual | 007-130220-201 | RevA |
| VaultIP Security Module, Firmware Reference Manual | 007-130250-204 | RevB |
| VaultIP Security Module, Secure Debug Application Note | 007-130250-401/2 | RevA |
| SafeXcel IP True Random Number Generator Hardware Reference and Programmer Manual | 007-076210-207 | RevA |
| SafeXcel IP True Random Number Generator Noise and Entropy Discussion | 007-076210-210 | RevA |

**Table 20 - MS1201 Security Sub-system Documentation**

### 10.3.1 Crypto Officer Guidance

For the validated module, at each initialization, the CO should verify the module is operating in FIPS validated configuration by performing the following steps:

- The CO should verify that the MODULE_STATUS register value shows 0x00000202. This indicates the successful completion of the power-up tests. If the value is shown as 0x80000202, this indicates the power-up tests failure and the module in Error state.

- Next the CO must send a Self-Test token to initialize the User role authentication. The successful completion of this step must be verified confirming that the MODULE_STATUS register shows 0x00000201 value.

The use of the module without these steps will be considered as a non-validated module.

# 11 Mitigation of Other Attacks

The cryptographic module does not implement security mechanisms to mitigate other attacks.

# A Appendixes

## A.1 Glossary and Abbreviations

| | |
|---|---|
| **AES** | Advanced Encryption Specification |
| **CAVP** | Cryptographic Algorithm Validation Program |
| **CBC** | Cipher Block Chaining |
| **CCM** | Counter with Cipher Block Chaining-Message Authentication Code |
| **CFB** | Cipher Feedback |
| **CMVP** | Cryptographic Module Validation Program |
| **CSP** | Critical Security Parameter |
| **CTR** | Counter Mode |
| **CVL** | Component Verification List |
| **DES** | Data Encryption Standard |
| **DMA** | Direct Memory Access |
| **ECDSA** | Elliptic Curve Digital Signature Algorithm |
| **FIPS** | Federal Information Processing Standards Publication |
| **FRO** | Free Running Oscillator |
| **FSM** | Finite State Model |
| **GCM** | Galois/Counter Mode |
| **HMAC** | Hash Message Authentication Code |
| **IP** | (semiconductor) Intellectual Property (core) |
| **IRQ** | Interrupt ReQuest |
| **KAT** | Known Answer Test |
| **KBKDF** | Key-based Key Derivation Function |
| **MAC** | Message Authentication Code |
| **NIST** | National Institute of Science and Technology |
| **NVLAP** | National Voluntary Laboratory Accreditation Program |
| **OFB** | Output Feedback |
| **OTP** | One-Time-Programmable memory |
| **O/S** | Operating System |
| **PSS** | Probabilistic Signature Scheme |
| **RNG** | Random Number Generator |
| **RSA** | Rivest, Shamir, Addleman |
| **RTL** | Register Transfer Level |
| **SHA** | Secure Hash Algorithm |
| **SHS** | Secure Hash Standard |
| **SOC** | System on a Chip |
| **SSH** | Secure Shell |
| **TCM** | Tightly Coupled Memory |
| **TEE** | Trusted Execution Environment |

# A.2 References


**FIPS140-2** **FIPS PUB 140-2 - Security Requirements For Cryptographic Modules**
**May 2001**
http://csrc.nist.gov/publications/fips/fips140-2/fips1402.pdf

**FIPS140-2_IG** **Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program**
http://csrc.nist.gov/groups/STM/cmvp/documents/fips140-2/FIPS1402IG.pdf

**FIPS180-4** **Secure Hash Standard (SHS)**
March 2012 http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

**FIPS186-4** **Digital Signature Standard (DSS)**
July 2013 http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-4.pdf

**FIPS197** **Advanced Encryption Standard**
November 2001
http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

**FIPS198-1** **The Keyed-Hash Message Authentication Code (HMAC)**
July 2008
http://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf

**PKCS#1** **Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1**
February 2003
http://www.ietf.org/rfc/rfc3447.txt

**RFC3394** **Advanced Encryption Standard (AES) Key Wrap Algorithm** September 2002
http://www.ietf.org/rfc/rfc3394.txt

**RFC5649** **Advanced Encryption Standard (AES) Key Wrap with Padding Algorithm**
September 2009
http://www.ietf.org/rfc/rfc5649.txt

**SP800-38A** **NIST Special Publication 800-38A - Recommendation for Block Cipher Modes of Operation - Methods and Techniques**
December 2001
http://csrc.nist.gov/publications/nistpubs/800-38a/sp800-38a.pdf

**SP800-38B** **NIST Special Publication 800-38B - Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication**
May 2005
http://csrc.nist.gov/publications/nistpubs/800-38B/SP_800-38B.pdf

**SP800-38C** **NIST Special Publication 800-38C - Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality**
May 2004
http://csrc.nist.gov/publications/nistpubs/800-38C/SP800-38C_updated-July20_2007.pdf

**SP800-38D** **NIST Special Publication 800-38D - Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC**
November 2007
http://csrc.nist.gov/publications/nistpubs/800-38D/SP-800-38D.pdf

**SP800-38E** **NIST Special Publication 800-38E - Recommendation for Block Cipher Modes of Operation: The XTS-AES Mode for Confidentiality on Storage Devices**
January 2010
http://csrc.nist.gov/publications/nistpubs/800-38E/nist-sp-800-38E.pdf

**SP800-38F** **NIST Special Publication 800-38F - Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping**
December 2012
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf

**SP800-56A** **NIST Special Publication 800-56A Revision 2 - Recommendation for Pair-Wise Key-Establishment Schemes Using Discrete Logarithm Cryptography**
May 2013
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar2.pdf

**SP800-56C** **NIST Special Publication 800-56C Revision 1 - Recommendation for Key-Derivation Methods in Key- Establishment Schemes**
April 2018
https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr1.pdf

**SP800-67** **NIST Special Publication 800-67 Revision 1 - Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher**
January 2012
http://csrc.nist.gov/publications/nistpubs/800-67-Rev1/SP-800-67-Rev1.pdf

**SP800-90A** **NIST Special Publication 800-90A – Revision 1 - Recommendation for Random Number Generation Using Deterministic Random Bit Generators**
January 2015
http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf

**SP800-108** **NIST Special Publication 800-108 - Recommendation for Key Derivation Using Pseudorandom Functions**
October 2009
http://csrc.nist.gov/publications/nistpubs/800-108/sp800-108.pdf