STMICROELECTRONICS

Trusted Platform Module ST33TPHF2XSPI, ST33TPHF2XI2C, ST33GTPMASPI & ST33GTPMAI2C

FIPS 140-2 Security Policy Level 2

Firmware revision:	00.01.01.00 / 00.01.01.01 / 00.01.01.02 / 00.02.01.00 / 00.02.01.10 / 00.03.01.00 / 00.06.01.00
HW version:	ST33HTPH revision A / ST33G1M2A revision F

Date: 2020/05/08
Document Version: 02-00

NON-PROPRIETARY DOCUMENT

Table of Contents

1 MODULE DESCRIPTION ... 3

1.1 DEFINITION ... 3
1.2 MODULE IDENTIFICATION ... 3
1.2.1 AHC4 ... 4
1.2.2 AHC5 ... 5
1.2.3 FAE5 ... 5
1.2.4 FAE6 ... 5
1.2.5 AHD4 ... 6
1.2.6 AHD5 ... 6
1.2.7 AHD8 ... 7
1.3 PINOUT DESCRIPTION ... 8
1.3.1 ST33TPHF2XSPI configuration ... 8
1.3.2 ST33TPHF2XI2C configuration ... 9
1.3.3 ST33GTPMASPI configuration ... 10
1.3.4 ST33GTPMAI2C configuration ... 10
1.4 BLOCK DIAGRAMS ... 11
1.4.1 HW block diagram ... 11
1.4.2 FW block diagram ... 13
1.5 SECURITY LEVELS ... 14
1.6 CRYPTOGRAPHIC FUNCTIONS ... 15
1.7 MODES OF OPERATION ... 17
1.7.1 Approved modes of operation ... 17
1.7.2 FIPS mode recommendations ... 17
1.7.3 Limited and error modes ... 18
1.8 PORTS AND INTERFACES ... 19

2 IDENTIFICATION AND AUTHENTICATION POLICY ... 21

2.1 ROLES ... 21
2.2 AUTHENTICATION ... 21
2.2.1 Description ... 21
2.2.2 Authorization strength ... 22
2.2.3 Authorization protection ... 23

3 ACCESS CONTROL POLICY ... 24

3.1 LIST OF KEYS AND CSPS ... 24
3.2 SERVICES ... 29
3.2.1 Services list ... 29
3.2.2 Authorization ... 35
3.3 KEY MANAGEMENT ... 37
3.3.1 Key entry and output ... 37
3.3.2 Key transport ... 38

4 SELF-TESTS ... 39

4.1 SELF-TESTS ON FIRST BOOT OF A FW ... 39
4.1.1 Power-up tests list ... 39
4.1.2 Asymmetric cryptography self-tests list ... 40
4.2 SELF-TESTS ON SUBSEQUENT BOOTS OF A FW ... 40
4.3 CONDITIONAL TESTS LIST ... 41
4.4 VERIFICATION ... 41

5 PHYSICAL SECURITY POLICY ... 42

6 OPERATIONAL ENVIRONMENT ... 43

7 MITIGATIONS OF OTHER ATTACKS ... 44

7.1 INTERNAL TAMPER DETECTION ... 44
7.2 ENVIRONMENTAL PROTECTION ... 44

8 REFERENCES ... 45

9 ACRONYMS ... 47

IMPORTANT NOTICE – PLEASE READ CAREFULLY ... 48

1 MODULE DESCRIPTION

1.1 Definition

The ST33TPHF2XSPI / ST33TPHF2XI2C / ST33GTPMASPI / ST33GTPMAI2C Trusted Platform Module are fully integrated security modules designed to be integrated into personal computers and other embedded systems. The security module is used primarily for cryptographic key generation, key storage, key management as well as generation and secure storage for digital certificates.

The TPM is a single chip cryptographic HW module as defined in **[FIPS 140-2]**. The single silicon chip is encapsulated in a hard, opaque, production grade integrated circuit (IC) package.

The cryptographic boundary is defined as the perimeter of the IC package. The security module supports SPI and I^2C interfaces compliant with the Trusted Computing Group (TCG) specification for PC Client [PTP 1.03]. The HW and FW cryptographic boundaries are indicated in §1.4 of the current document.

The security module implements version 2.0 (revision 1.38) of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM).

1.2 Module identification

The hardware and firmware versions covered by the FIPS evaluation are identified as follow:

- Hardware version: ST33HTPH revision A / ST33G1M2A revision F
- Firmware version: 00.01.01.00 / 00.01.01.01 / 00.02.01.00 / 00.03.01.00 / 00.06.01.00 / 00.01.01.02 / 00.02.01.10

Correspondence is summarized in the next table.

Table 1: Module identification table

	Module Identification			
Module Name	ST33TPHF2XSPI	ST33TPHF2XI2C	ST33GTPMASPI	ST33GTPMAI2C
Hardware Version	ST33HTPH revision A		ST33G1M2A revision F	
Interface	SPI	I^2C	SPI	I^2C
Firmware Version	00.01.01.00	00.02.01.00	00.03.01.00	00.06.01.00
	00.01.01.01	00.02.01.10		
	00.01.01.02			

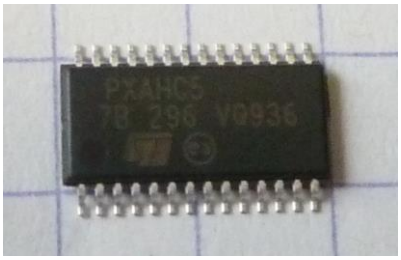
FW version can be retrieved via response to the command TPM2_GetCapability with property set to TPM_PT_FIRMWARE_VERSION_1.

The cryptographic services are provided by the cryptographic library "**NesLib 6.3.3 for ST33**".

The product ST33HTPH is manufactured in two packages:

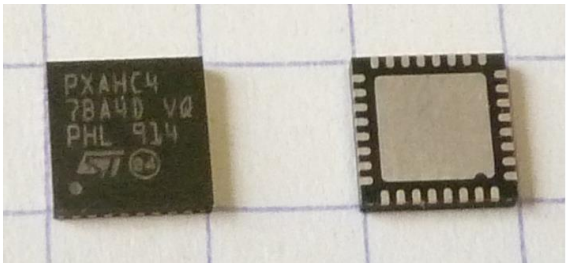
- TSSOP28
  - TSSOP 28-pin
  - 4.4 x 9.7 mm

Figure 1: TSSOP28 package (PXAHC5 marking)



- vQFN32
  - Very thin pitch Quad Flat No-lead 32-pin
  - 5 x 5 mm

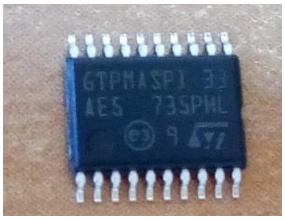
Figure 2: vQFN32 package (PXAHC4 marking)



The product ST33G1M2A is manufactured in one package:

- TSSOP20
  - TSSOP 20-pin
  - 6.5 x 4.4 mm

Figure 3: TSSOP20 package (GTPMASPI AE5 marking)



The security module is available in the configurations listed hereafter.

1.2.1 *AHC4*

This configuration of the security module implement only the version 2.0 of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM). The current FIPS 140-2 level 2 security policy always applies (no mode lock requested) to this security module configuration.

Table 2: Security module configuration

	Module configuration					
Product name / HW version	ST33TPHF2XSPI/ ST33HTPH revision A					
Package	TSSOP28	vQFN32	TSSOP28	vQFN32	TSSOP28	vQFN32
Marking	PXAHC4					
FW version	00.01.01.00		00.01.01.01[1]		00.01.01.02[2]	
Libraries version	00.02.01.00 (HWINTF library)		00.02.01.01 (HWINTF library)		00.02.01.02 (HWINTF library)	

[1] The default version of this configuration is 00.01.01.00. To operate with FW version 00.01.01.01, module must be first field upgraded from 00.01.01.00 to 00.01.01.01.
[2] The default version of this configuration is 00.01.01.00. To operate with FW version 00.01.01.02, module must be first field upgraded from 00.01.01.00 to 00.01.01.02. If module has been first field upgraded to 00.01.01.01, it must then be field upgraded from 00.01.01.01 to 00.01.01.02.

	00.01.01.00 (TPM2.0 library)	00.01.01.02 (TPM2.0 library)	00.01.01.02 (TPM2.0 library)

1.2.2 *AHC5*

This configuration of the security module implement only the version 2.0 of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM). The current FIPS 140-2 level 2 security policy always applies (no mode lock requested) to this security module configuration.

Table 3: Security module configuration

	Module configuration			
Product name / HW version	ST33TPHF2XI2C/ ST33HTPH revision A			
Package	TSSOP28	vQFN32	TSSOP28	vQFN32
Marking	PXAHC5			
FW version	00.02.01.00		00.02.01.10[1]	
Libraries version	00.06.01.00 (HWINTF library) 00.01.01.03 (TPM2.0 library)		00.06.01.10 (HWINTF library) 00.01.01.10 (TPM2.0 library)	

1.2.3 *FAE5*

This configuration of the security module implement only the version 2.0 of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM). The current FIPS 140-2 level 2 security policy always applies (no mode lock requested) to this security module configuration.

Table 4: Security module configuration

	Module configuration
Product name / HW version	ST33GTPMASPI/ ST33G1M2A revision F
Package	TSSOP20
Marking	GTPMASPI AE5
FW version	00.03.01.00
Libraries version	00.03.01.00 (HWINTF library) 00.01.01.02 (TPM2.0 library)

1.2.4 *FAE6*

This configuration of the security module implement only the version 2.0 of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM). The current FIPS 140-2 level 2 security policy always applies (no mode lock requested) to this security module configuration.

[1] The default version of this configuration is 00.02.01.00. To operate with FW version 00.02.01.10, module must be first field upgraded from 00.02.01.00 to 00.02.01.10.

Table 5: Security module configuration

	Module configuration
Product name / HW version	ST33GTPMAI2C/ ST33G1M2A revision F
Package	TSSOP20
Marking	GTPMAI2C AE6
FW version	00.06.01.00
Libraries version	00.05.01.00 (HWINTF library) 00.01.01.03 (TPM2.0 library)

1.2.5 *AHD4*

This configuration of the security module implement only the version 2.0 of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM). The current FIPS 140-2 level 2 security policy always applies (no mode lock requested) to this security module configuration.

Table 6: Security module configuration

	Module configuration			
Product name / HW version	ST33TPHF2XSPI/ ST33HTPH revision A			
Package	TSSOP28	vQFN32	TSSOP28	vQFN32
Marking	PXAHD4			
FW version	00.01.01.01		00.01.01.02[1]	
Libraries version	00.02.01.01 (HWINTF library) 00.01.01.02 (TPM2.0 library)		00.02.01.02 (HWINTF library) 00.01.01.02 (TPM2.0 library)	

1.2.6 *AHD5*

This configuration of the security module implement only the version 2.0 of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM). The current FIPS 140-2 level 2 security policy always applies (no mode lock requested) to this security module configuration.

Table 7: Security module configuration

	Module configuration	
Product name / HW version	ST33TPHF2XI2C/ ST33HTPH revision A	
Package	TSSOP28	vQFN32
Marking	PXAHD5	
FW version	00.02.01.10	
Libraries version	00.06.01.10 (HWINTF library) 00.01.01.10 (TPM2.0 library)	

[1] The default version of this configuration is 00.01.01.01. To operate with FW version 00.01.01.02, module must be first field upgraded from 00.01.01.01 to 00.01.01.02.

1.2.7 *AHD8*

This configuration of the security module implement only the version 2.0 of the Trusted Computing Group (TCG) specification for Trusted Platform Modules (TPM). The current FIPS 140-2 level 2 security policy always applies (no mode lock requested) to this security module configuration.

Table 8: Security module configuration

	Module configuration	
Product name / HW version	ST33TPHF2XSPI/ ST33HTPH revision A	
Package	TSSOP28	vQFN32
Marking	PXAHD8	
FW version	00.01.01.02	
Libraries version	00.02.01.02 (HWINTF library) 00.01.01.02 (TPM2.0 library)	

1.3 Pinout description

The pin layouts for the ST33TPHF2XSPI are shown in Figure 4 and Figure 5.

The pin layouts for the ST33TPHF2XI2C are shown in Figure 6 and Figure 7.

The pin layouts for the ST33GTPMASPI are shown in Figure 8.

The pin layouts for the ST33GTPMAI2C are shown in Figure 9.

1.3.1 ST33TPHF2XSPI configuration

Figure 4: TSSOP28 Pinout Diagram

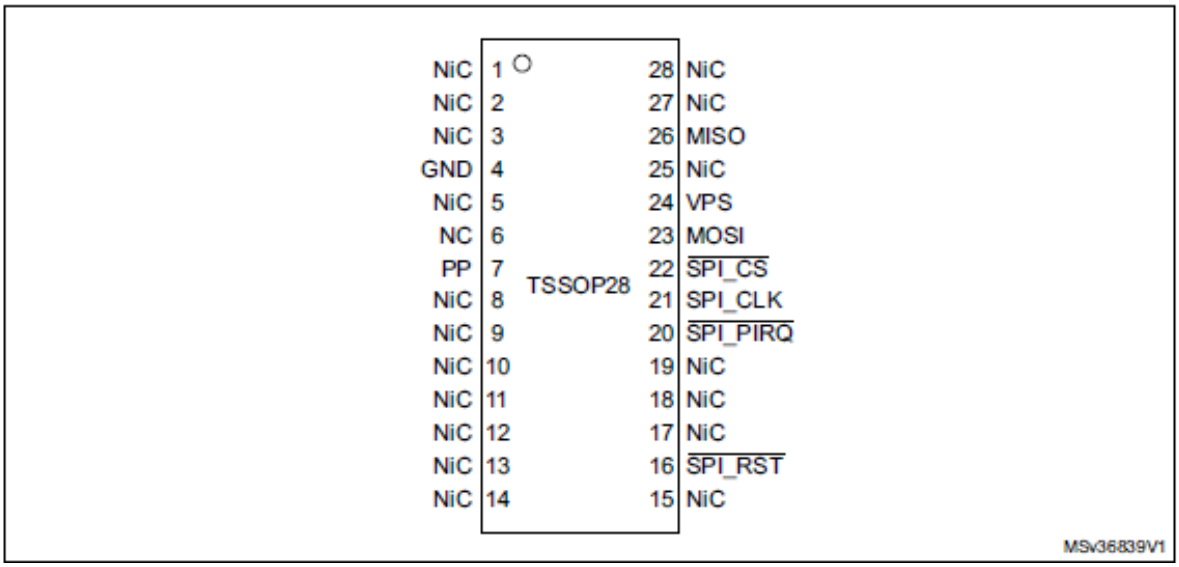
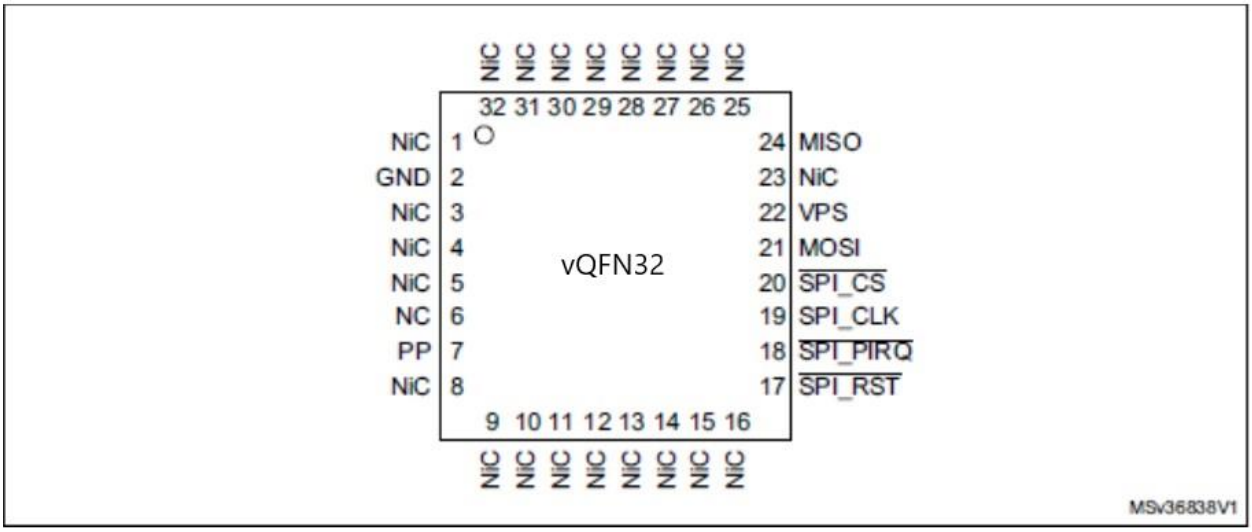


Figure 5: vQFN32 Pinout Diagram



Next table gives a description of the products pins.

Table 9: ST33TPHF2XSPI pins definition

Signal	Type	Description
VPS	Input	**Power supply**. This pin must be connected to 1.8V or 3.3V DC power rail supplied by the motherboard.
GND	Input	GND has to be connected to the main motherboard ground.
$\overline{\text{SPI_RST}}$	Input	**SPI Reset** used to re-initialize the device
MISO	Output	**SPI** Master Input, Slave Output (output from slave)

MOSI	Input	**SPI** Master Output, Slave Input (output from master)
SPI_CLK	Input	**SPI** serial clock (output from master)
$\overline{\text{SPI_CS}}$	Input	**SPI** slave select (active low; output from master)
$\overline{\text{SPI_PIRQ}}$	Output	**SPI IRQ** used by TPM to generate an interrupt
PP	Input	**Physical presence**, active high, internal pull-down. Used to indicate Physical Presence to the TPM.
NiC	-	**Not internally connected:** not connected to the die. May be left unconnected but no impact on TPM if connected.
NC	-	**Not Connected:** connected to the die but not usable. May be left unconnected. Internal pull-down.

1.3.2 *ST33TPHF2XI2C configuration*

Figure 6: TSSOP28 Pinout Diagram

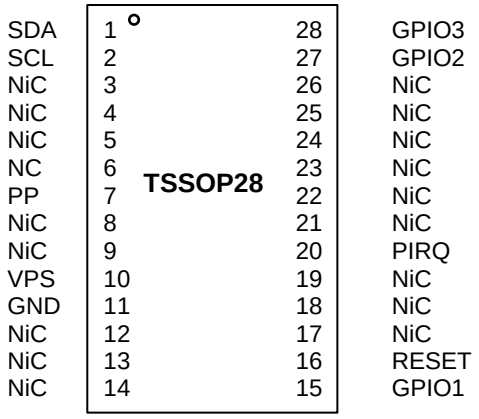
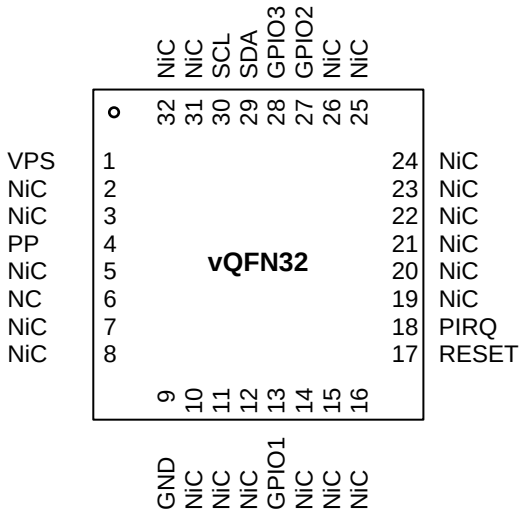


Figure 7: vQFN32 Pinout Diagram



Next table gives a description of the products pins.

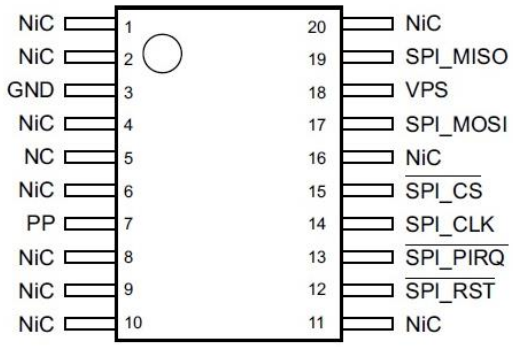
Table 10: ST33TPHF2XI2C pins definition

Signal	Type	Description
VPS	Input	**Power supply**. This pin must be connected to 1.8V or 3.3V DC power rail supplied by the motherboard.
GND	Input	GND has to be connected to the main motherboard ground.

RESET	Input	Reset used to re-initialize the device
SCL	Input	I²C serial clock (Open drain with no weak pull-up resistor)
SDA	Input/Output	I²C serial data (Open drain with no weak pull-up resistor)
PIRQ	Output	IRQ used by TPM to generate an interrupt
GPIO1	Input/Output	GPIO default to low (not used, reserved for future use)
GPIO2	Input/Output	GPIO default to low (not used, reserved for future use)
GPIO3	Input/Output	GPIO default to low (not used, reserved for future use)
PP	Input	**Physical presence**, active high, internal pull-down. Used to indicate Physical Presence to the TPM.
NiC	-	**Not internally connected:** not connected to the die. May be left unconnected but no impact on TPM if connected.
NC	-	**Not Connected:** connected to the die but not usable. May be left unconnected. Internal pull-down.

1.3.3 *ST33GTPMASPI configuration*

Figure 8: TSSOP20 Pinout Diagram



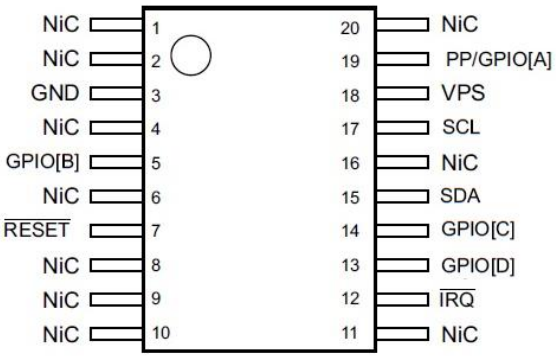
Next table gives a description of the products pins.

Table 11: ST33GTPMASPI pins definition

Signal	Type	Description
VCC	Input	**Power supply**. This pin must be connected to 1.8V or 3.3V DC power rail supplied by the motherboard.
GND	Input	GND has to be connected to the main motherboard ground.
$\overline{\text{RESET}}$	Input	**SPI Reset** used to re-initialize the device
MISO	Output	**SPI** Master Input, Slave Output (output from slave)
MOSI	Input	**SPI** Master Output, Slave Input (output from master)
SPI_CLK	Input	**SPI** serial clock (output from master)
$\overline{\text{SPI_CS}}$	Input	**SPI** slave select (active low; output from master)
$\overline{\text{SPI_PIRQ}}$	Output	**SPI IRQ** used by TPM to generate an interrupt
PP	Input	**Physical presence**, active high, internal pull-down. Used to indicate Physical Presence to the TPM.
NiC	-	**Not internally connected:** not connected to the die. May be left unconnected but no impact on TPM if connected.
NC	-	**Not Connected:** connected to the die but not usable. May be left unconnected. Internal pull-down.

1.3.4 *ST33GTPMAI2C configuration*

Figure 9: TSSOP20 Pinout Diagram



Next table gives a description of the products pins.

Table 12: ST33GTPMAI2C pins definition

Signal	Type	Description
VPS	Input	**Power supply**. This pin must be connected to 1.8V or 3.3V DC power rail supplied by the motherboard.
GND	Input	GND has to be connected to the main motherboard ground.
$\overline{\text{RESET}}$	Input	Reset used to re-initialize the device
$\overline{\text{IRQ}}$	Output	IRQ used by TPM to generate an interrupt
SCL	Input	I²C serial clock (Open drain with no weak pull-up resistor)
SDA	Input/Output	I²C serial data (Open drain with no weak pull-up resistor)
GPIO[B]	Input/Output	GPIO default to low (not used, reserved for future use)
GPIO[C]	Input/Output	GPIO default to low (not used, reserved for future use)
GPIO[D]	Input/Output	GPIO default to low (not used, reserved for future use)
PP	Input	**Physical presence**, active high, internal pull-down. Used to indicate Physical Presence to the TPM.
NiC	-	**Not internally connected:** not connected to the die. May be left unconnected but no impact on TPM if connected.
NC	-	**Not Connected:** connected to the die but not usable. May be left unconnected. Internal pull-down.

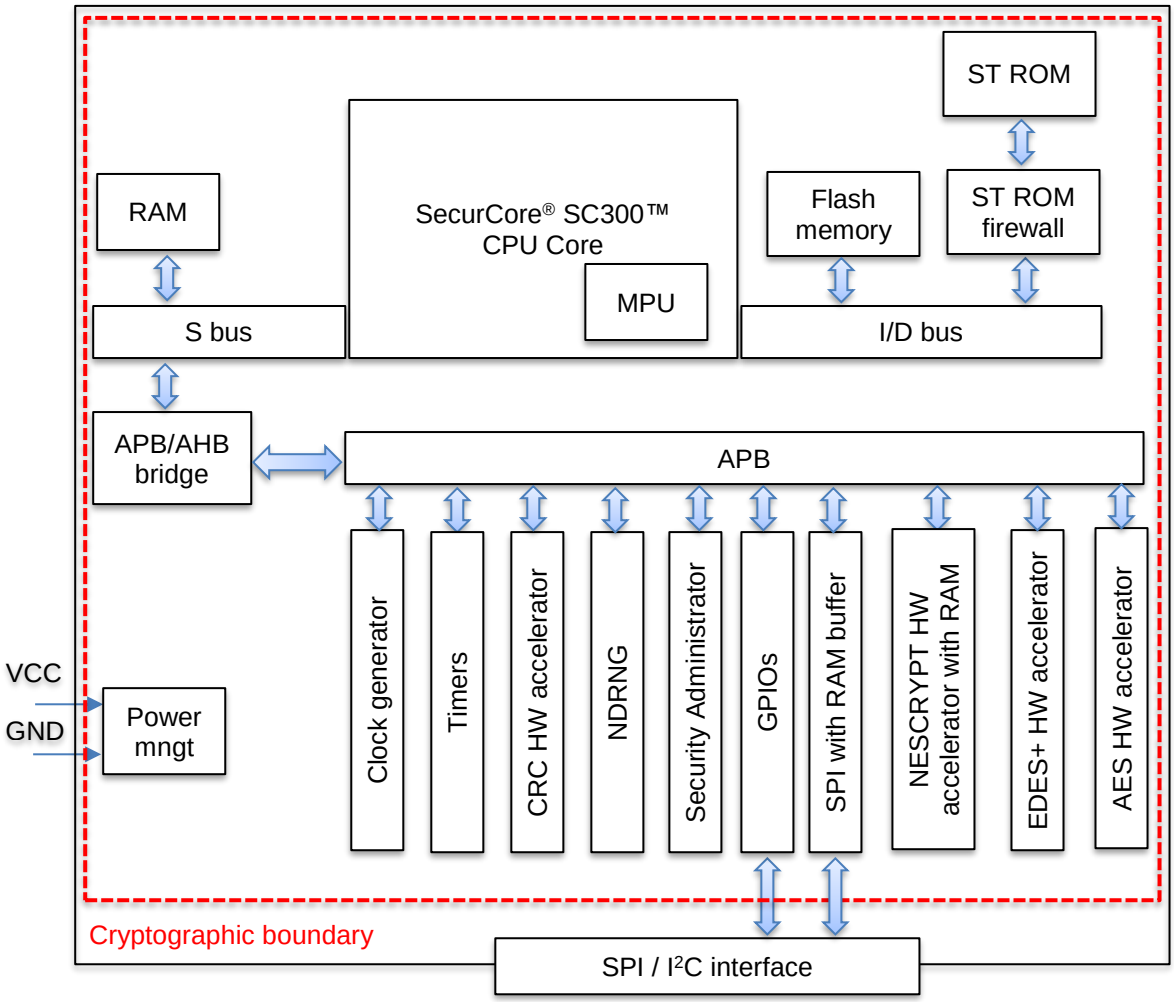
1.4 Block diagrams

1.4.1 HW block diagram

A block diagram of both the ST33HTPH hardware and the ST33G1M2A hardware with the associated cryptographic boundary are provided in Figure 10. Both hardware share the same block diagram. They are composed of:

- A SecurCore® SC300™ CPU core including a MPU (Memory Protection Unit)
- Memories (RAMs, Flash and ROM)
- HW accelerators for CRC (16 and 32-bits) and cryptographic operations (symmetric with EDES+ and AES and asymmetric with NESCRYPT)
- A clock generator and three 16-bit timers
- NDRNG (non-deterministic random bit generator)
- SPI master/slave block
- A security administration block dedicated to chip security configuration and alarms detection
- FW and data stored in the memory areas

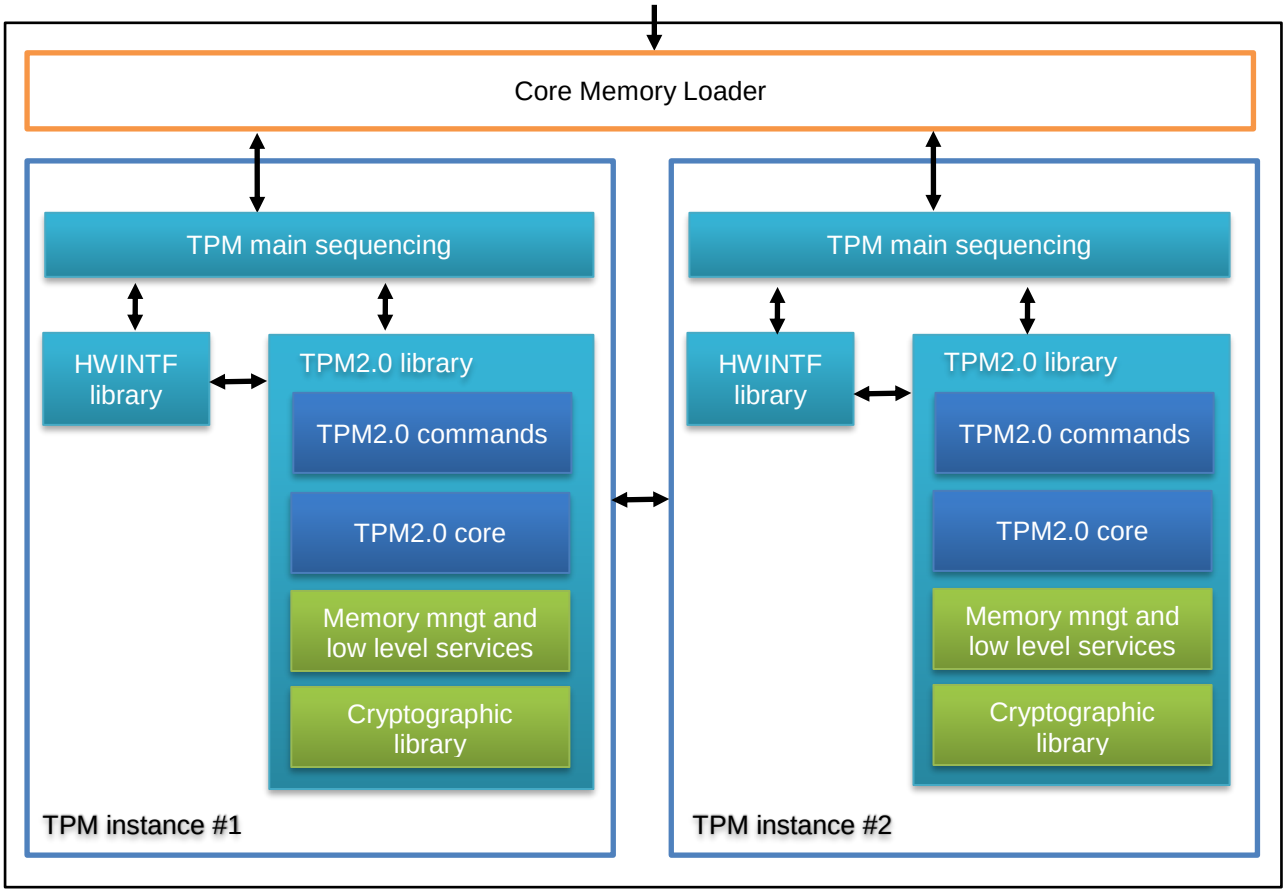
Figure 10: ST33HTPH/ST33G1M2A block diagram



1.4.2 *FW block diagram*

The block diagram of the firmware, valid for all configurations (00.01.01.00 / 00.01.01.01 / 00.02.01.00 / 00.03.01.00 / 00.06.01.00 / 00.01.01.02 / 00.02.01.10), is provided at the next figure.

Figure 11: FW block diagram



FW is composed of three independent blocks:

- A non-upgradable code block located in ROM & flash memories (orange box)
  - Core memory loader (CML) in charge of verifying integrity of the TPM instance to be executed.
- Two independent code blocks upgradable via secure field upgrade mechanism (TPM instances #1 and #2). They are composed of:
  - TPM2.0 commands code
  - TPM2.0 core
  - Memory management and low-level services
  - Cryptographic library

1.5 Security levels

The cryptographic module meets the overall requirements applicable to Level 2 security of FIPS 140-2.

Table 13: Module Security Level Specification

Security Requirements Section	Level
Cryptographic Module Specification	2
Cryptographic Module Ports and Interfaces	2
Roles, Services and Authentication	2
Finite State Model	2
Physical Security	3
Operational Environment	N/A
Cryptographic Key Management	2
EMI/EMC	2
Self-Tests	2
Design Assurance	2
Mitigation of Other Attacks	2
Overall	**2**

1.6 Cryptographic functions

The security module supports the following cryptographic algorithms (both approved and non-approved). Algorithm certificate numbers for each approved algorithm are listed below. All algorithms, keys size or curve lengths listed below are part of services offered by the module.

Table 14: Approved algorithms

CAVP Cert	Algorithm	Standard	Mode / Method	Key lengths, curves or moduli	Use
C957, C958, C959, C960, C961, C1658	RSA	FIPS 186-4	SHA-256, SHA-384, RSASSA-PKCS-v1.5, RSASSA-PSS	2048	Digital signature generation
		FIPS 186-4	SHA-1[1], SHA-256, SHA-384, RSASSA-PKCS-v1.5, RSASSA-PSS	1024[2], 2048	Digital signature verification
		FIPS 186-4	Appendix C3.1	2048	Key generation
C955, C956	CVL RSADP	SP800-56B	RSA decryption primitive	2048	Key transport
C955, C956	ECDSA	FIPS 186-4	SHA-256, SHA-384	P-256, P-384	Digital signature generation
		FIPS 186-4	SHA-1, SHA-256, SHA-384	P-256, P-384	Digital signature verification
		FIPS 186-4	-	P-256, P-384	Key verification
C957, C958, C959, C960, C961, C1658	ECDSA	FIPS 186-4	Appendix B.4.2	P-256, P-384	Key generation
C951, C952, C954, C1656	HMAC (single call)	FIPS 198-1	SHA-1, SHA-256, SHA-384, SHA3-256, SHA3-384	160, 256, 384	Message authentication
C962, C963, C964, C1657	HMAC (sequence)	FIPS 198-1	SHA-1, SHA-256, SHA-384, SHA3-256, SHA3-384	160, 256, 384	Message authentication
C951, C952, C954, C1656	KBKDF	SP 800-108	CTR		Key derivation
C928	DRBG	SP 800-90A	HASH_based		Deterministic random bit generation
C957, C958, C959, C960, C961, C1658	AES	FIPS 197, SP 800-38A	ECB, CFB128, OFB, CBC, CTR	128, 192, 256	Data encryption/decryption
C957, C958, C959, C960, C961, C1658	Triple-DES	SP 800-67, SP 800-38A	TECB, TCBC, TCFB64, TOFB, CTR	192	Data encryption/decryption
C957+C954, C958+C952, C959+C951, C960+C954, C961+C952, C1658+C1656	KTS (AES cert + HMAC cert)	SP 800-38F	CFB	128, 256	Key transport

[1] Legacy use only
[2] Legacy use only

C928	SHS	FIPS 180-4	SHA-1, SHA-256, SHA-384		Message digest
C928	SHA-3	FIPS 202	SHA3-256, SHA3-384		Message digest
Vendor affirmation	CKG	SP800-133 (per IG D.12)	Direct generation, Generation		Key generation[1]
	KTS RSA	SP800-56B	KTS-OAEP-basic	2048	Key transport
	RSA	FIPS 186-4	SHA3-256, SHA3-384	2048	Digital signature generation
					Digital signature verification
	ECDSA	FIPS 186-4	SHA3-256, SHA3-384	P-256, P-384	Digital signature generation
					Digital signature verification
	KAS SSC	SP 800-56A Rev3	ECC	P-256, P-384	Key agreement scheme – shared secret computation
	KDA	SP800-56C Rev 1			Key derivation

Table 15: Allowed algorithms

Algorithm	Caveat	Use
NDRNG	NA	Seed or reseed DRBG 800-90A (with a minimum of 0.9x384=345 bits of entropy). Generate random numbers not dedicated to be used as cryptographic material.

Table 16: Non-approved algorithms

Algorithm	Use
RSA (key length = 1024 bits)	Key and digital signature generation
SHA-1	Digital signature generation
ECSchnorr	Digital signature generation and verification
ECDAA	Digital signature generation
ECC derived keys	Secret exchange or digital signature generation/verification

[1] Symmetric keys and seeds used for generating the asymmetric keys are either generated by using KBKDF or DRBG methods. Methods are detailed per CSPs in Table 22.

1.7 Modes of Operation

This security policy only applies to the security module when TPM operator follows the recommendations from:

- §1.7.1 to execute all self-tests required in a FIPS 140-2 approved mode of operation
- §1.7.2 to use the security module in a FIPS 140-2 approved mode of operation

1.7.1 Approved modes of operation

TPM supports two sequential approved modes of operation.

1.7.1.1 Approved mode 1

This mode is the default mode when TPM starts. This mode is limited to a subset of TPM services.

Table 17: Approved mode 1

Properties	Description
Definition	Transient mode of operation when TPM is power-up the very first time or after a field upgrade. This mode is valid until TPM2_SelfTest(full=YES) command is completed
Configuration	No configuration required
Services available	List of available services is indicated in last column of Table 23: Command support table.
Algorithms used	SHA / SHA3 / HMAC / AES / DRBG / KDF / TDES
CSPs used	List of CSPs that might be accessed in this mode is indicated in Table 23: Command support table.
Self-tests	SHA / SHA3 / HMAC / AES / DRBG / KDF / TDES / HW integrity / FW integrity / NDRNG

1.7.1.2 Approved mode 2

This mode is the full FIPS approved mode of operation. This mode is reached after completion of TPM2_SelfTest(full=YES) command if this is the very first boot or after a field upgrade. This mode is also reached on all subsequent boots after completion of the boot sequence that performs the HW and SW integrity tests (FIPS 140-2 IG 9.11).

Table 18: Approved mode 2

Properties	Description
Definition	Full approved mode of operation
Configuration	TPM2_SelfTest(full=YES) execution
Services available	All services
Algorithms used	All supported algorithms (cf. §1.6)
CSPs used	All CSPs
Self-tests	SHS / SHA-3 / HMAC / AES / DRBG / KDF / TDES / RSA / ECDH / ECDSA / HW integrity / FW integrity / NDRNG

1.7.2 FIPS mode recommendations

To use the TPM in a FIPS approved mode of operation (valid for mode1 and mode2), the TPM operator **shall**:

- Use an encryption session for the commands that inputs/outputs CSPs (List is indicated at §3.3.1). For commands without authorization, encryptedSalt used in TPM_StartAuthSession on encryption session creation must be different from the empty buffer.
- Use an approved symmetric algorithm (AES) for encryption sessions
- Use authorization session based on HMAC or policy (no password allowed, cf. §2.2.1).
- Set the attribute noDA to 0 for objects to benefit from DAM protection (§2.2.2.1).
- Duplicate only objects with *encryptedDuplication* attribute set.
- Not use FIPS 140-2 non-approved algorithms:
  - SHA-1 for RSA digital signature generation
  - EC Schnorr for ECC digital signature generation
  - ECDAA for ECC digital signature generation
  - Use ECC key derived from a parent key for ECC cryptographic operations

  For the following services:
  - TPM2_Sign, TPM2_Certify, TPM2_CertifyCreation, TPM2_Quote, TPM2_GetSessionAuditDigest, TPM2_GetCommandAuditDigest, TPM2_GetTime, TPM2_NV_Certify, TPM2_Commit
- Not use TPM2_LoadExternal service to load TDES keys into the TPM.
- Limit number of encryptions with a same TDES key to a maximum of 2^{16} encryptions.
- Use a policy including TPM2_PolicyAuthValue as a minimum in the policy sequence in case authorization is ensured by policy (authorization by policy must be at least as secure as authorization by HMAC).
- Use TPM2_HierarchyChangeAuth after first TPM init or after each TPM2_Clear to set the authorization value for the endorsement, platform, owner and lockout hierarchies.
- Use TPM2_CreatePrimary command only for RSA and ECC key with default template.

If operator does not strictly follow the FIPS approved mode recommendations (ex: use of XOR instead of AES in encryption session), TPM is considered as being in a FIPS non-approved mode of operation.

To use the TPM in a FIPS approved mode if it was previously used in a FIPS non-approved mode, the operator shall:

- Zeroize all data listed in Table 22: Keys and CSPs list that could potentially be reused as CSPs in FIPS approved mode

To use the TPM in a FIPS non-approved mode if it was previously used in a FIPS approved mode, the operator shall:

- Zeroize all CSPs listed in Table 22: Keys and CSPs list that could potentially be used by FIPS non-approved algorithms in FIPS approved mode

1.7.3 Limited and error modes

TPM may reach specific states depending on sequence of operation that occurred.

1.7.3.1 Shutdown mode

The shutdown mode is an infinite HW reset loop that may be exit only by a power-off/power-on sequence. This state is entered when TPM detects that a FW integrity check failed during the TPM boot sequence.

1.7.3.2 *Failure mode*

Failure mode is a state of TPM that restricts the commands that can be executed to TPM2_GetCapability / TPM2_GetTestResult. TPM answers to all other commands with a specific error code: TPM_RC_FAILURE (0x101). This state is entered when one (except FW integrity test) of the self-tests fails.

1.8 Ports and interfaces

The physical port of the security module is the SPI bus or I²C Bus. The logical interfaces and their mapping to physical ports of the module are described below:

Table 19 : Ports and interfaces

Logical interface	Description	Physical port
Control Input Interface	Control Input commands issued to the security module	**SPI :** $\overline{\text{SPI_CS}}$ / SPI_CLK / MOSI / SPI_RST / PP **I²C :** SCL / SDA / RESET / PP
Status Output Interface	Status data output by the chip	**SPI :** $\overline{\text{SPI_CS}}$ / SPI_CLK / MISO / SPI_PIRQ **I²C :** SCL / SDA / PIRQ
Data Input Interface	Data provided to the chip as part of the data processing commands	**SPI :** $\overline{\text{SPI_CS}}$ / SPI_CLK / MOSI **I²C :** SCL / SDA
Data Output Interface	Data output by the chip as part of the data processing command	**SPI :** $\overline{\text{SPI_CS}}$ / SPI_CLK / MISO **I²C :** SCL / SDA
Power interface	Power interface of the chip	VPS / GND

Here are some details concerning the ports and interfaces of TPM:

1. The module does not include a maintenance interface.
2. Control and data inputs are multiplexed over the same physical interface. Control and data are distinguished by properly parsing input TPM command parameters according to input structures description, indicated for each command in **[TPM2.0 Part3 r1.38]**[1].
3. Status and data output are multiplexed over the same physical interface. Status and data are distinguished by properly setting output TPM response parameters according to output structures description, indicated for each command in **[TPM2.0 Part3 r1.38]**.
4. The logical state machine and the command structure parsing of the module prevent from using input data externally from the “data input path” and prevent from outputting data externally from the “data output path”.
5. While performing key generation or key zeroization (no manual key entry on TPM), the output data path is logically disconnected while the output status path remains connected to report any possible failure during command processing. Generally, the output data path is only connected when TPM outputs response containing data.
6. Plaintext data can be output through usage of:
   - TPM2_Unseal, TPM2_RSA_Decrypt, TPM2_EncryptDecrypt

   To prevent inadvertent release of the plaintext data, command performs:
   - Check of command input structure
   - Check of command authorization

[1] Some commands only deal with control input and status output parameters

- Decryption of the input blob with private part of specified key

However an encryption session might be used with these commands to avoid releasing plaintext data.

7. The logical state machine and command structure of the module guarantees the inhibition of all data output via the data output interface whenever an error state exists and while doing self-tests.

2 IDENTIFICATION AND AUTHENTICATION POLICY

This chapter gives details about the roles managed by TPM.

2.1 Roles

Services proposed by TPM are accessible under different roles. Next table defines the different roles supported by the TPM.

Table 20: Roles

Role	Description	Type of authentication	Authentication data
Crypto officer (CO)	Role that requires knowledge of the authValue or authPolicy associated to one of the hierarchy (incl. lockout).	Role based	384-bit secret data (authValue and/or authPolicy)
User (U)	Role that requires knowledge of the authValue or authPolicy associated to one object or NV index.	Role based	160-bit, 256-bit or 384-bit secret data (authValue and/or authPolicy). Authorization depends on userWithAuth object attribute.
Admin (A)	The object Administrator controls the certification of an object (TPM2_Certify and TPM2_ActivateCredential) and controls changing of the authValue of an object (TPM2_ObjectChangeAuth).	Role based	160-bit, 256-bit or 384-bit (authValue and/or authPolicy). Authorization depends on adminWithPolicy object attribute.
DUP (D)	This authorization role is only used for TPM2_Duplicate(). If duplication is allowed, authorization must always be provided by a policy session and the authPolicy equation of the object must contain a command that sets the policy command code to TPM_CC_Duplicate.	Role based	160-bit, 256-bit or 384-bit secret data (authPolicy).

Some commands can also be executed without any authorization role. Those commands are marked as NA in the service list table (Table 23: Command support table).

The security module does NOT provide a Maintenance Role or Maintenance Interface and does NOT support concurrent operators.

Roles are implicitly selected by TPM operator on command execution (cf. Table 23 for correspondence between service and supported role) by proving knowledge of the authorization value or knowledge of the policy sequence (nature of authorization session indicates the type of authorization) that are associated with the object the command is operating on.

An operator might switch from one role to another by executing commands requiring different roles and proving knowledge of the authorization value or policy sequence of objects the role is associated to.

2.2 Authentication

2.2.1 Description

In FIPS approved mode of operation, TPM uses a mechanism for authorization that consists in:

1. Opening an authorization session that may be of the following types:
    a. HMAC
    b. Policy

2. Executing the expected policy commands sequence in case of policy authorization session (defined policy must follow recommendations listed in §1.7.2).
3. Do the comparison between reference value and computed value. If both match, command execution is authorized.

More details on HMAC and policy sessions can be found in §19 of [TPM2.0 Part1 r1.38].

2.2.2 *Authorization strength*

As minimum value of authorization or policy values might be 160-bit random values (based on unbiased distribution of '0' and '1'), the probability for an attacker to guess the authorization data is:

$$\frac{1}{2^{160}} = 6.84 * 10^{-49}$$

This value is then higher than the minimum of $1*10^{-6}$ required by **[FIPS140-2]**.

The number of attempts per minute that an attacker can make is limited by the DAM (Dictionary Attack Mechanism) or by the PIN fail mechanism (they are concurrent mechanisms).

2.2.2.1 DAM

DAM consists in counting the number of failed authentication. When this counter reaches a pre-defined threshold, a lockout period is started. During this period, no authorized command execution is allowed and a specific error is returned in TPM response until period expires. Next table indicates the threshold values and the lockout durations:

Table 21: DAM lockout durations

Failed authentication counter	>31
Lockout period (in seconds)	7200

This table indicates that an attacker can do a maximum (during the first minute) of 32 trials per minute before DAM being active. As a result the probability per minute that a random attempt will lead to a successful authorization matches FIPS requirements. Value is equal to:

$$32 * \frac{1}{2^{160}} \approx 2.19 * 10^{-47}$$

This value is then higher than the minimum of $1*10^{-5}$ required by **[FIPS140-2]**.

NB: commands handling (reception, processing and response sending) is negligible compared to the lockout periods and not taken into account in the above computation.

NB2: DAM parameters might be changed by using TPM2_DictionnaryAttackParameters command. However to operate in a FIPS approved mode, they shall not be changed in order not to decrease the authorization strength computed above.

2.2.2.2 PIN fail

The PIN fail mechanism is based on the use of an NV index (named PIN index) that contains two 32-bits values: *pinCount* and *pinLimit.* Mechanism consists in proving, during a policy session (through use of TPM2_PolicySecret command), the knowledge of the *authValue* of this PIN index. If it fails, *pinCount* is incremented. If *pinCount* >= *pinLimit*, authorization is locked.

The best case for an attacker is a *pinCount* and *pinLimit* set to the maximum possible value (2^{32}-1). So probability of successful authorization is equal to:

$$(2^{32} - 1) * \frac{1}{2^{160}} \approx 2.94 * 10^{-39}$$

This value is then higher than the minimum of $1*10^{-5}$ required by **[FIPS140-2]**.

NB: commands handling (reception, processing and response sending) is negligible compared to the lockout periods and not taken into account in the above computation.

2.2.2.3 *Hierarchies authValue*

The *ownerAuth*, *platformAuth* and *endorsementAuth* associated to the three hierarchies are not subject to DAM or PIN fail protection. They are 384-bit random values (based on unbiased distribution of '0' and '1'). Probability of guess per minute can be expressed as:

$$n * \frac{1}{2^{384}}$$

Where n is the number of attempts per minute. If we consider a maximum of 2^{32}-1 trials per minute (value being much higher than what processing timings of any command permit), the probability of successful authorization per minute is equal to:

$$(2^{32} - 1) * \frac{1}{2^{384}} \approx 1.09 * 10^{-106}$$

This value is then higher than the minimum of $1*10^{-5}$ required by **[FIPS140-2]**.

Total number of trials to decrease the probability to $1*10^{-5}$ is equal to:

$$1.10^{-5} * 2^{384} \approx 3.94 * 10^{110}$$

By still considering 2^{32}-1 trials per minute, this means that the total number of minutes necessary to decrease the probability of hierarchy *authValue* guess to $1*10^{-5}$ is equal to:

$$(3.94 * 10^{110})/(2^{32} - 1) \approx 9.17 * 10^{100}$$

This value justifies not having DAM or PIN fail protection for hierarchies' *authValue*.

2.2.3 <u>*Authorization protection*</u>

By following recommendations to operate in FIPS mode of operation, authorization data associated to objects, NV indexes or hierarchies are never output from TPM in plaintext form and thus are protected from unauthorized disclosure.

Authorization can be changed via the following services:

- TPM2_ObjectChangeAuth
- TPM2_HierarchyChangeAuth
- TPM2_NV_ChangeAuth

As indicated in Table 23, roles that imply authentication are associated with these services meaning that authentication are protected against unauthorized modification and substitution.

TPM authorization mechanism (HMAC or policy digest comparison) does not provide any information about authentication data or policy sequence. Authentication indicates pass (command executed) or fail (command not executed) and does not provide feedback that could weaken the strength of authentication.

3 ACCESS CONTROL POLICY

This chapter gives details about the services, keys and CSPs that the TPM manages.

3.1 List of Keys and CSPs

Table 22: Keys and CSPs list

Keys/CSPs		Description	Zeroized
Index	**Name**		
Hierarchies			
1	nullSeed	64 bytes primary seed values resp. for NULL, platform, endorsement and storage hierarchies. nullSeed is a random value generated by HDRBG at each TPM power-up. ppSeed / epSeed / spSeed are random values generated by HDRBG before its first use. They are used as seeds for: • DRBG to generate random used for sensitive part creation of primary keys (prime numbers for RSA and private key for ECC/KEYEDHASH/SYMCIPHER) and seedValue creation for all types of primary keys.	TPM reset
2	ppSeed		TPM2_Change PPS
3	epSeed		TPM2_Change EPS
4	spSeed		TPM2_Clear
5	nullProof	64 bytes secret values resp. for NULL, platform, endorsement and storage hierarchies. nullProof is a random value generated by HDRBG at each TPM power-up. phProof / ehProof / shProof are random values generated by HDRBG before its first use. They are used as keys for: • KDFa to generate context encryption key and IV (cf. [TPM2.0 Part1 r1.38] §30.3.1) • HMAC to compute context blob integrity (cf. [TPM2.0 Part1 r1.38] §30.3.2) • HMAC to compute/verify tickets shProof is used also as key (or part of key) for: • KDFa to generate obfuscation value used in attestation commands (cf. [TPM2.0 Part1 r1.38] §36.7) • KDFa to generate CSP #30. shProof is also used as source of entropy for: • DRBG reseed before generating seedValue (CSP #20) in the endorsement hierarchy (cf. [TPM2.0 Part1 r1.38] §27.7.4)	TPM reset
6	phProof		TPM2_Change PPS
7	ehProof		TPM2_Clear / TPM2_Change EPS
8	shProof		TPM2_Clear
9	platformAuth	48 bytes authorization data (authValue) used in authorization session based resp. on platform, endorsement, and storage or lockout hierarchy authorization. PlatformAuth is set to 0 at each TPM2_Startup (CLEAR). EndorsementAuth / ownerAuth / lockoutAuth are set to 0 at first TPM power-up. Primary auth values can be changed with command TPM2_HierarchyChangeAuth. They are used as keys for: • HMAC authorization in case of unsalted and unbound session • KDFa to generate session key used in HMAC authorization in case of bound session They are used as part of keys for: • HMAC authorization in case of salted or bound session (key is concatenation of sessionKey and authValue) • KDFa to generate session key used in HMAC authorization in case of salted and bound session (key is concatenation of authValue and salt) They are used as reference values for comparison in case of password authorization session.	TPM2_Startup
10	endorsementAuth		TPM2_Clear / TPM2_Change EPS
11	ownerAuth		TPM2_Clear
12	lockoutAuth		TPM2_Clear

13	platformPolicy	48 bytes authorization data (authPolicy) used in authorization session based resp. on platform, endorsement, storage or lockout hierarchy policy. platformPolicy is set to 0 at each TPM2_Startup (CLEAR). endorsementPolicy / ownerPolicy / lockoutPolicy are set to 0 at first TPM power-up. Primary policies can be changed with command TPM2_SetPrimaryPolicy. They are used as reference values for a comparison in case of policy session.	TPM2_Change PPS / TPM reset
14	endorsementPolicy		TPM2_Clear / TPM2_Change EPS
15	ownerPolicy		TPM2_Clear
16	lockoutPolicy		TPM2_Clear
Objects			
17	authValue	0 to 48 bytes authorization data defined during object creation (TPM2_Create/TPM2_CreatePrimary/TPM2_CreateLoaded) used to authorize commands based on this object. Value can be changed with command TPM2_ObjectChangeAuth. It is used as: • HMAC and/or KDFa keys or part of keys in authorization session based on HMAC or password (usage is the same than for CSPs #9/10/11/12) • Secret value extended into policyDigest on TPM2_PolicySecret command	TPM2_Clear (owner & endorsement) TPM2_Change PPS (platform) TPM2_Change EPS (endorsement)
18	authPolicy	0 to 48 bytes authorization data defined during object creation (TPM2_Create/TPM2_CreatePrimary/TPM2_CreateLoaded) used to authorize commands based on this object. It is used as reference value for a comparison in case of policy session	TPM2_Clear (owner & endorsement) TPM2_Change PPS (platform) TPM2_Change EPS (endorsement)
20	seedValue	48 bytes generated from: • DRBG (cf. CSP #47) instantiated with CSP #1/2/3/4, a template hash, data and a string in case of primary object • TPM2.0 DRBG instance (cf. CSP #38) for ordinary objects • KDFa using parent's seed in case of derived objects. It is used (when not set to 0) as: • Data in SHA computation to generate object's unique value (HMAC and symmetric key creation) • Key in KDFa to generate a symmetric encryption key used in TPM2B_PRIVATE structure en/decryption. • Key in KDFa to generate HMAC key used in TPM2B_PRIVATE integrity protection generation or verification	TPM2_Clear (owner & endorsement) TPM2_Change PPS (platform) TPM2_Change EPS (endorsement)
21	symKey	16 bytes generated from derivation of seedValue through KDFa usage. It is used as key for: • Symmetric en/decryption of TPM2B_PRIVATE structure	Transient value only available during command processing
22	hmacKey	20 to 48 bytes generated from derivation of seedValue through KDFa usage. It is used as key for: • HMAC used in TPM2B_PRIVATE integrity protection generation or verification	Transient value only available during command processing
23	sensitive	Object sensitive part might be passed as encrypted parameter to TPM2_Create/TPM2_CreateLoaded commands or generated by: • DRBG (cf. CSP #47) instantiated with CSP #1/2/3/4, a template hash and a string in case of primary object • TPM2.0 DRBG instance (cf. CSP #38) for ordinary objects	TPM2_Clear (owner & endorsement) TPM2_Change PPS (platform)

		• KDFa using parent's sensitive value in case of derived objects (type symcipher, keyedhash or ECC objects) . For RSA or ECC key, sensitive corresponds to the private key. Depending on object's nature, sensitive is used as key for: • en/decryption (RSA, AES, TDES) • signature generation (RSA, ECDSA, HMAC) • secret value exchange (ECDH) • key for derivation through KDFa of derived objects Available key lengths correspond to the ones listed in Table 14: Approved algorithms (Key nature and length are selected by user thanks to the interface of the keys creation commands).	TPM2_Change EPS (endorsement)
NV Indexes			
24	authValue	0 to 48 bytes authorization data defined during NV index creation (TPM2_NV_DefineSpace) used to authorize commands based on this object. Value can be changed with command TPM2_NV_ChangeAuth. It is used as: • HMAC and/or KDFa keys or part of keys in authorization session based on HMAC or password. • Secret value extended into policyDigest on TPM2_PolicySecret command	TPM2_NV_UndefineSpace / TPM2_NV_UndefineSpaceSpecial
25	authPolicy	0 to 48 bytes authorization data defined during NV index creation (TPM2_NV_DefineSpace) used to authorize commands based on this object. It is used as reference value for a comparison in case of policy session	TPM2_NV_UndefineSpace / TPM2_NV_UndefineSpaceSpecial
Sessions			
26	salt	Value passed encrypted (with a loaded decrypt key) to TPM2_StartAuthSession. It is used as: • Part of KDFa key to generate the sessionKey (cf. [TPM2.0 Part1 r1.38] §19.6)	Transient value only available during TPM2_StartAuthSession processing
27	sessionKey	Key generated by KDFa (cf. [TPM2.0 Part1 r1.38] §19.6) and whose value depends on salt and bind parameters of TPM2_StartAuthSession command (size depends on symmetric algorithm used). It is used as: • HMAC key used to generate and verify command authorization • Part of KDFa key used to generate encryption key and IV of encryption-based session	TPM2_FlushContext
28	encryption key and IV of encryption-based session	Symmetric key and IV generated by KDFa (cf. [TPM2.0 Part1 r1.38] §21.3) from sessionKey and object's authValue. It is used as key and IV for: • Symmetric en/decryption of first parameter of command/response if parameter structure is of type TPM2B_	TPM2_FlushContext
Context			
29	contextKey	16 bytes randomly generated by HDRBG at each TPM reset. It is used as: • 1st part of key used in KDFa to generate a symmetric encryption key and IV used in context blob en/decryption.	TPM reset

30	symKey, IV	2*16 bytes derived from the concatenation of contextKey and one of the proof (CSP 5, 6, 7, 8). It is used as key and IV for: • Symmetric en/decryption of context blob	Transient value only available during TPM2_ContextSave / TPM2_ContextLoad processing
Duplication			
31	inner symKey	Symmetric key passed as input to duplication commands or generated by HDRBG (size depends on symmetric algorithm used). It is used as: • Symmetric en/decryption key to protect TPM2B_PRIVATE output structure	Transient value only available during command processing
32	seed	20 to 48 bytes value randomly generated by HDRBG if new parent is a RSA key or via KDFe if new parent is an ECC key. It is used as key for : • KDFa to generate a symmetric en/decryption key for outer protection • KDFa to generate a HMAC key for outer integrity protection	
33	outer symKey	Symmetric key generated via KDFa from seed. It is used as key for: • Symmetric en/decryption key for outer protection of TPM2B_PRIVATE output structure	
34	outer hmacKey	HMAC key generated via KDFa from seed. It is used as key for: • HMAC integrity key for outer protection of TPM2B_PRIVATE output structure	
Credential			
35	seed	20 to 48 bytes value randomly generated by HDRBG if new parent is a RSA key or via KDFe if new parent is an ECC key. It is used as key for : • KDFa to generate a symmetric en/decryption key for outer protection • KDFa to generate a HMAC key for outer integrity protection	Transient value only available during command processing
36	symKey	Symmetric key generated via KDFa from seed. It is used as key for: • Symmetric en/decryption key for outer protection of credentialBlob	
37	hmacKey	HMAC key generated via KDFa from seed. It is used as key for: • HMAC integrity key for outer protection of credentialBlob	
DRBG			
38	DRBG state	Internal state (V and C secret values) of the HDRBG (based on SHA256) stored in RAM. This is the state of the main DRBG instance used to produce random numbers.	TPM2_Clear
ECC			
39	commitNonce	48 bytes value randomly generated by HDRBG at each TPM2_Startup (CLEAR). It is used as key for : • KDFa to generate an ECC ephemeral private key used in TPM2_EC_Ephemeral command / TPM2_ZGen_2Phase	Transient value only available during command processing
40	ephemeral key – derived from commitNonce	ECC private key (size depends on curve selected) generated with KDFa from commitNonce. It is used as ephemeral private key in: • TPM2_Ephemeral command (scalar multiplication) to generate the associated ephemeral public key • TPM2_Zgen_2Phase (ECDH scheme) to generate outZ2 (output point)	
41	ephemeral key	ECC private key (size depends on curve selected) generated with HDRBG. It is used as ephemeral private key in: • TPM2_ECDH_KeyGen command (ECDH scheme) to generate zPoint (output point)	

Static keys			
42	Endorsement key - RSA primes	2 primes of 1024 bits used to construct EK if parameters in TPM2_CreatePrimary or TPM2_CreateLoaded (if not derivation parent and not a parent object) command match the default EK RSA template. Generated by a FIPS140-2 compliant HSM.	TPM2_Change EPS
43	Endorsement key - ECC private key	ECC 256 bits or 384 bits private key used to construct EK if in TPM2_CreatePrimary or TPM2_CreateLoaded (if not derivation parent and not a parent object) command match the default EK ECC template. Generated by a FIPS140-2 compliant HSM.	TPM2_Change EPS
Field upgrade keys			
44	Field upgrade RSA verification key	2048 bits permanent RSA key unique per TPM product line. Only public part of the key is stored in the TPM (modulus, exponent).	No (public key)
45	Field upgrade ECC verification key	384 bits permanent ECC key unique per TPM product line. Only public part of the key is stored in the TPM.[1]	No (public key)
Transient DRBG			
47	Transient DRBG state	Internal state (V and C secret values) of a HDRBG instance (based on SHA256) stored in RAM. HDRBG is instantiated from primary seeds and used only in TPM2_CreatePrimary and TPM2_CreateLoaded (if not derivation parent and not a parent object) to generate prime numbers for primary RSA keys.	Transient DRBG state cleared at the end of random numbers generation
DRBG input seed			
48	DRBG input seed	48-bytes value output from a NDRNG.	Transient value
First boot identifier			
49	First boot identifier	32-bits value indicating if this is the first boot of the security module or the first boot after a FW installation.	NA

[1] Key is only provisioned in TPM for future use and is currently not manipulated by any TPM service.

3.2 Services

Next table lists all services supported by the TPM and indicates for each service, the role that can use this service and the keys/CSPs that can be accessed.

3.2.1 *Services list*

Table 23: Command support table

Services		Role	Keys and CSP access W = write, O = output, Z = zeroize C = used as key in cryptographic operation R = read (and not used as C)	Authorized in limited approved mode
Start-up				
1	_TPM_Init	**NA**	W (first boot only) : 2, 3, 4, 6, 7, 8, 10, 11, 12, 14, 15, 16 W : 29, 38, 48, 49 R : 49	X
2	TPM2_Startup	**NA**	W : 1, 5, 9, 13, 39	X
3	TPM2_Shutdown	**NA**	**-**	X
Testing				
4	TPM2_SelfTest	**NA**	**-**	X
5	TPM2_IncrementalSelfTest	**NA**	**-**	X
6	TPM2_GetTestResult	**NA**	**-**	X
Session commands				
7	TPM2_StartAuthSession	**NA**	W : 26, 27 C : 9, 10, 11, 12, 17, 24, 26, 28	
8	TPM2_PolicyRestart	**NA**	-	
Objects commands				
9	TPM2_Create	**U**	R : 18 W : 20, 21, 22, 23, 28, 38, 48 C : 5, 6, 7, 8, 17, 20, 21, 22, 27, 28, 38, 48 O : 17, 18, 20, 23	
10	TPM2_Load	**U**	R : 18 W : 17, 18, 20, 21, 22, 23, 28, 38, 48 C : 17, 20, 21, 22, 27, 28, 38, 48	
11	TPM2_LoadExternal	**NA**	W : 17, 18, 20, 21, 22, 23, 28, 38, 48 C : 17, 20, 21, 22, 27, 28, 38, 48	X
12	TPM2_ReadPublic	**NA**	R : 23 W : 28 C : 28	X
13	TPM2_ActivateCredential	**A, U**	R : 18, 23, 35 W : 28, 36, 37 C : 27, 28, 35, 36, 37	
14	TPM2_MakeCredential	**NA**	R : 23 W : 28, 35, 36, 37 C : 28, 36, 37 O : 35	

Services		Role	Keys and CSP access W = write, O = output, Z = zeroize C = used as key in cryptographic operation R = read (and not used as C)	Authorized in limited approved mode
15	TPM2_Unseal	**U**	R : 18, 23 W : 28 C : 27, 28 O : 23	
16	TPM2_ObjectChangeAuth	**A**	R : 18 W : 17, 28, 38, 48 C : 27, 28, 38, 48	
17	TPM2_CreateLoaded	**CO, U**	R : 18, 42, 43, 47 W : 20, 21, 22, 23, 28, 38, 47, 48 C : 1, 2, 3, 4, 8, 17, 20, 21, 22, 27, 28, 38, 47, 48 O : 20, 23	
Duplication commands				
18	TPM2_Duplicate	**D**	R : 18 W : 28, 31, 32, 33, 34, 38, 48 C : 27, 28, 31, 32, 33, 34, 38, 48 O : 23, 31, 32	
19	TPM2_Rewrap	**U**	R : 18, 32 W : 28, 31, 32, 33, 34, 38, 48 C : 27, 28, 31, 32, 33, 34, 38, 48 O : 23, 31, 32	
20	TPM2_Import	**U**	R : 18, 32 W : 28, 31, 33, 34, 38, 48 C : 27, 28, 31, 32, 33, 34, 38, 48 O : 23	
Asymmetric primitives				
21	TPM2_RSA_Encrypt	**NA**	C: 28	
22	TPM2_RSA_Decrypt	**U**	R : 18 W : 28 C : 23, 27, 28	
23	TPM2_ECDH_KeyGen	**NA**	W : 28, 41 C : 28, 41	
24	TPM2_ECDH_ZGen	**U**	R : 18 W : 28 C : 23, 27, 28	
25	TPM2_ECC_Parameters	**NA**	-	X
26	TPM2_ZGen_2Phase	**U**	R : 18 W : 28, 38, 48 C : 23, 27, 28, 38, 40, 48	
Symmetric primitives				
27	TPM2_EncryptDecrypt	**U**	R : 18 W : 28 C : 23, 27, 28	

Services		Role	Keys and CSP access W = write, O = output, Z = zeroize C = used as key in cryptographic operation R = read (and not used as C)	Authorized in limited approved mode
28	TPM2_EncryptDecrypt2	**U**	R : 18 W : 28 C : 23, 27, 28	
29	TPM2_Hash	**NA**	W: 28 C: 5, 6, 7, 8, 28	
30	TPM2_HMAC	**U**	R : 18 W : 28 C : 23, 27, 28	
Random number generator				
31	TPM2_GetRandom	**NA**	C : 28, 38, 48	X
32	TPM2_StirRandom	**NA**	W : 28, 38, 48 C: 28	X
Hash/HMAC/Event sequences				
33	TPM2_HMAC_Start	**U**	R : 18 W : 17, 28 C : 23, 27, 28	
34	TPM2_HashSequenceStart	**NA**	W: 17, 28 C: 28	X
35	TPM2_SequenceUpdate	**U**	R : 18 W : 28 C : 23, 27, 28	
36	TPM2_SequenceComplete	**U**	R : 18 W : 28 C : 5, 6, 7, 8, 23, 27, 28	
37	TPM2_EventSequenceComplete	**U**	R : 18 W : 28 C : 23, 27, 28	
Attestation commands				
38	TPM2_Certify	**A, U**	R : 18 W : 28, 38, 48 C : 8, 23, 27, 28, 38, 48	
39	TPM2_CertifyCreation	**U**	R : 18 W : 28, 38, 48 C : 5, 6, 7, 8, 23, 27, 28, 38, 48	
40	TPM2_Quote	**U**	R : 18 W : 28, 38, 48 C : 8, 23, 27, 28, 38, 48	
41	TPM2_GetSessionAuditDigest	**CO**	R : 18 W : 28, 38, 48 C : 8, 23, 27, 28, 38, 48	
42	TPM2_GetCommandAuditDigest	**CO**	R : 18 W : 28, 38, 48 C : 8, 23, 27, 28, 38, 48	

Services		Role	Keys and CSP access W = write, O = output, Z = zeroize C = used as key in cryptographic operation R = read (and not used as C)	Authorized in limited approved mode
43	TPM2_GetTime	**CO**	R : 18 W : 28, 38, 48 C : 8, 23, 27, 28, 38, 48	
Ephemeral EC keys				
44	TPM2_EC_Ephemeral	**NA**	W : 28, 40 C : 28, 39	
Signing and signature verification				
45	TPM2_VerifySignature	**NA**	R : 23 W : 28 C : 5, 6, 7, 8, 28	
46	TPM2_Sign	**U**	R : 18 W : 28, 38, 48 C : 5, 6, 7, 8, 23, 27, 28, 38, 48	
Command audit				
47	TPM2_SetCommandCodeAuditStatus	**CO**	R : 13, 18 C : 9, 11, 15, 27	
Integrity collection (PCR)				
48	TPM2_PCR_Extend	**U**	R : 18 C : 27	
49	TPM2_PCR_Event	**U**	R : 18 W : 28 C : 27, 28	
50	TPM2_PCR_Read	**NA**	**-**	X
51	TPM2_PCR_Allocate	**CO**	R : 13, 18 C : 9, 27	
52	TPM2_PCR_Reset	**NA**	**-**	
53	_TPM_Hash_Start	**NA**	**-**	X
54	_TPM_Hash_Data	**NA**	**-**	X
55	_TPM_Hash_End	**NA**	**-**	X
Enhanced authorization commands				
56	TPM2_PolicySigned	**NA**	C: 5, 6, 7, 8, 28	
57	TPM2_PolicySecret	**U**	R : 18 W : 28, 38, 48 C : 5, 6, 7, 8, 9, 10, 11, 12, 17, 24, 27, 28, 38, 48	
58	TPM2_PolicyTicket	**NA**	W : 28 C: 5, 6, 7, 8, 28	
59	TPM2_PolicyOR	**NA**	-	
60	TPM2_PolicyPCR	**NA**	W : 28 C: 28	
61	TPM2_PolicyLocality	**NA**	-	
62	TPM2_PolicyNV	**U**	R : 18 W : 28 C : 27, 28	

Services		Role	Keys and CSP access W = write, O = output, Z = zeroize C = used as key in cryptographic operation R = read (and not used as C)	Authorized in limited approved mode
63	TPM2_PolicyCounterTimer	**NA**	W : 28 C: 28	
64	TPM2_PolicyCommandCode	**NA**	-	
65	TPM2_PolicyPhysicalPresence	**NA**	-	
66	TPM2_PolicyCpHash	**NA**	W : 28 C: 28	
67	TPM2_PolicyNameHash	**NA**	W : 28 C: 28	
68	TPM2_PolicyDuplicationSelect	**NA**	W : 28 C: 28	
69	TPM2_PolicyAuthorize	**NA**	W : 28 C: 5, 6, 7, 8, 28	
70	TPM2_PolicyAuthValue	**NA**	-	
71	TPM2_PolicyPassword	**NA**	-	
72	TPM2_PolicyGetDigest	**NA**	W : 28 C: 28	
73	TPM2_PolicyNvWritten	**NA**	-	
74	TPM2_PolicyTemplate	**NA**	-	
75	TPM2_PolicyAuthorizeNV	**U**	R : 25 C : 24	
Hierarchy commands				
76	TPM2_CreatePrimary	**CO**	R : 13, 14, 15, 16, 18, 42, 43, 47 W : 20, 21, 22, 23, 28, 38, 47, 48 C : 1, 2, 3, 4, 8, 17, 20, 21, 22, 27, 28, 38, 42, 43, 47, 48 Z : 47	
77	TPM2_HierarchyControl	**CO**	C : 9, 10, 11, 27	
78	TPM2_SetPrimaryPolicy	**CO**	W : 13, 14, 15, 16, 28 C : 9, 10, 11, 12, 27, 28	
79	TPM2_ChangePPS	**CO**	Z : 2, 6, 13,14, 17, 18, 20, 23, 43	
80	TPM2_ChangeEPS	**CO**	Z : 3, 7, 10, 14, 17, 18, 20, 23, 42	
81	TPM2_Clear	**CO**	R : 13, 16 Z : 4, 7, 8, 9, 10, 11, 12, 14, 15, 16, 17, 18, 20, 23, 24, 25 C : 38	
82	TPM2_ClearControl	**CO**	R : 13, 16 C : 9, 12	
83	TPM2_HierarchyChangeAuth	**CO**	R : 13, 16 W : 9, 10, 11, 12, 28, 38, 48 C : 9, 10, 11, 12, 28, 38, 48	
Non-Volatile Storage				
84	TPM2_DictionaryAttackLockReset	**CO**	R : 16 C : 12	
85	TPM2_DictionaryAttackParameters	**CO**	R : 16 C : 12	
Field Upgrade (proprietary commands)				
86	TPM2_VendorFieldUpgradeStart	**CO**	W : 28 C : 9, 13, 28, 44	

Services		Role	Keys and CSP access W = write, O = output, Z = zeroize C = used as key in cryptographic operation R = read (and not used as C)	Authorized in limited approved mode
87	TPM2_VendorFieldUpgradeData	**NA**	W : 49	
Context Management				
88	TPM2_ContextSave	**NA**	W : 30 C : 5, 6, 7, 8, 29, 30	
89	TPM2_ContextLoad	**NA**	W : 30 C : 5, 6, 7, 8, 29, 30	
90	TPM2_FlushContext	**NA**	Z : 17, 18, 20, 23, 27, 28	X
91	TPM2_EvictControl	**CO**	R : 13, 15 C : 9, 11	
Clock and Timers				
92	TPM2_ReadClock	**NA**	**-**	X
93	TPM2_ClockSet	**CO**	R : 13, 15 C : 9, 11	
94	TPM2_ClockRateAdjust	**CO**	R : 13, 15 C : 9, 11	
Capability Commands				
95	TPM2_GetCapability	**NA**	**-**	X
96	TPM2_TestParms	**NA**	**-**	X
Non-volatile storage				
97	TPM2_NV_DefineSpace	**CO**	R : 13, 15, 18 W : 24, 25, 28 C : 9, 11, 27, 28	
98	TPM2_NV_UndefineSpace	**CO**	R : 13, 15, 18 C : 9, 11, 27 Z : 24, 25	
99	TPM2_NV_UndefineSpaceSpecial	**CO, A**	R : 9, 13, 17, 18 C : 9, 11, 27 Z : 24, 25	
100	TPM2_NV_ReadPublic	**NA**	C: 28	X
101	TPM2_NV_Write	**U**	W : 28 R : 25 C : 24, 27, 28	
102	TPM2_NV_Increment	**U**	R : 25 C : 24, 27	
103	TPM2_NV_Extend	**U**	W : 28 R : 25 C : 24, 27, 28	
104	TPM2_NV_SetBits	**U**	R : 25 C : 24, 27	
105	TPM2_NV_WriteLock	**U**	R : 25 C : 24, 27	
106	TPM2_NV_GlobalWriteLock	**CO**	C : 27	

Services		Role	Keys and CSP access W = write, O = output, Z = zeroize C = used as key in cryptographic operation R = read (and not used as C)	Authorized in limited approved mode
107	TPM2_NV_Read	**U**	W : 28 R : 25 C : 24, 27, 28	
108	TPM2_NV_ReadLock	**U**	R : 25 C : 24, 27	
109	TPM2_NV_ChangeAuth	**A**	W : 24, 28 C : 24, 27, 28	
110	TPM2_NV_Certify	**U**	W : 28, 38, 48 R : 25 C : 24, 27, 28, 38, 48	
Proprietary commands				
111	TPM2_VendorCmdSetMode	**CO**	W : 28 C : 27, 28	
112	TPM2_VendorCmdSetCommandSet	**CO**	W : 28 C : 27, 28	
113	TPM2_VendorCmdRestoreEK	**CO**	Z : 3, 7, 10, 14, 17, 18, 20 R : 42, 43 W : 23, 28 C : 27, 28	
114	TPM2_VendorCmdSetCommandSet Lock	**CO**	W : 28 C : 27, 28	
115	TPM2_VendorCmdGetRandom2	**NA**	C : 28, 38, 48	
116	TPM2_VendorCmdGPIOConfig	**NA**	**-**	
Misc. commands				
117	TPM2_PP_Commands	**CO**	**-**	
Non FIPS services				
118	Field upgrade de-obfuscation[1]	**NA**	**-**	

3.2.2 *Authorization*

Some of the services listed above manipulate CSPs without requiring the operator to assume an authorized role:

- Services restricted to use of SHS:

  TPM2_Hash, TPM2_HashSequenceStart

- Services using DRNG (read, state update without manipulation):

  TPM2_GetRandom, TPM2_GetRandom2, TPM2_StirRandom

- Services used for authentication mechanism:

[1] This service is not callable from TPM interface but is only used internally by TPM2_VendorFieldUpgradeData command. It consists of de-obfuscating data received by the TPM2_VendorFieldUpgradeData command with a non-FIPS approved algorithm.

TPM2_StartAuthSession, TPM2_PolicySigned, TPM2_PolicyTicket, TPM2_PolicyPCR, TPM2_PolicyCounterTimer TPM2_PolicyLocality, TPM2_PolicyCpHash, TPM2_PolicyNameHash, TPM2_PolicyAuthorize, TPM2_PolicyAuthorizeNV, TPM2_PolicyDuplicationSelect, TPM2_PolicyGetDigest

- Services using (read, cryptographic operation) only public part of objects:

  TPM2_ReadPublic, TPM2_RSA_Encrypt, TPM2_NV_ReadPublic

- Specific services that do not affect security of the module:

  TPM2_LoadExternal: loaded object not considered as protected object (specific attribute).

  TPM2_MakeCredential: convenience function that do not use TPM secrets.

  TPM2_ECDH_KeyGen: ephemeral ECC key generation

  TPM2_EC_Ephemeral: ephemeral ECC key generation

  TPM2_FieldUpgradeData: transport command for field upgrade. Can be used only if TPM2_FieldUpgradeStart command has been successfully executed (authorized command)

  TPM2_ContextSave: save objects under an encrypted and integrity protected format

  TPM2_ContextLoad: load encrypted and integrity protected objects into TPM

  TPM2_FlushContext: flush loaded object/session from TPM volatile memory

3.3 Key management

3.3.1 *Key entry and output*

Next table indicates the approved method used to encrypt all secret, private keys and data (indicated by S for secret value, P for private key and D for user defined data in type column), entered into or output from the cryptographic module.

Table 24 : Encrypted methods for secret and private keys input

Service	Parameter name	Type	Input or output	Encryption algorithm
TPM2_ActivateCredential	credentialBlob	S	Input	AES CFB
	secret	S	Input	RSA OAEP or ECDH
TPM2_ContextSave	context	D	Output	AES CFB
TPM2_ContextLoad	context	D	Input	AES CFB
TPM2_Create	inSensitive	P / S	Input	AES CFB (*)
	outPrivate	P / S	Output	AES CFB
TPM2_CreateLoaded	inSensitive	P / S	Input	AES CFB (*)
	outPrivate	P / S	Output	AES CFB
TPM2_CreatePrimary	inSensitive	P / S	Input	AES CFB (*)
TPM2_Duplicate	encryptionKeyIn (if present)	S	Input	AES CFB (*)
	encryptionKeyOut	S	Output	AES CFB (*)
	duplicate	S	Output	AES CFB
	outSymSeed	S	Output	RSA OAEP or ECDH
TPM2_EventSequenceComplete	buffer	D	Input	AES CFB (*)
TPM2_GetRandom	randomBytes	D	Output	AES CFB (**)
TPM2_Hash	data	D	Input	AES CFB (*)
TPM2_HashSequenceStart	auth	S	Input	AES CFB (*)
TPM2_HierarchyChangeAuth	newAuth	S	Input	AES CFB (*)
TPM2_HMAC	buffer	D	Input	AES CFB (*)
TPM2_HMACStart	auth	S	Input	AES CFB (*)
TPM2_Import	encryptionKeyIn (if present)	S	Input	AES CFB (*)
	duplicate	S	Input	AES CFB
	inSymSeed	S	Input	RSA OAEP or ECDH
	outPrivate	S	Output	AES CFB
TPM2_Load	inPrivate	P / S	Input	AES CFB
TPM2_LoadExternal	inPrivate	P / S	Input	AES CFB (*)
TPM2_MakeCredential	credentialBlob	S	Output	AES CFB
	secret	S	Output	RSA OAEP or ECDH
TPM2_NV_ChangeAuth	newAuth	S	Input	AES CFB (*)

TPM2_NV_DefineSpace	auth	S	Input	AES CFB (*)
TPM2_NV_Extend	data	D	Input	AES CFB (*)
TPM2_NV_Read	data	D	Output	AES CFB (**)
TPM2_NV_Write	data	D	Input	AES CFB (*)
TPM2_ObjectChangeAuth	newAuth	S	Input	AES CFB (*)
	outPrivate	S	Output	AES CFB
TPM2_PCR_Event	eventData	D	Input	AES CFB (*)
TPM2_Rewrap	inDuplicate	S	Input	AES CFB
	inSymSeed	S	Input	RSA OAEP or ECDH
	outDuplicate	S	Output	AES CFB
	outSymSeed	S	Output	RSA OAEP or ECDH
TPM2_RSA_Decrypt	message	D	Output	AES CFB (**)
TPM2_RSA_Encrypt	message	D	Input	AES CFB (*)
TPM2_SequenceComplete	buffer	D	Input	AES CFB (*)
TPM2_SequenceUpdate	buffer	D	Input	AES CFB (*)
TPM2_SetPrimaryPolicy	authPolicy	S	Input	AES CFB (*)
TPM2_StirRandom	inData	D	Input	AES CFB (*)
TPM2_Unseal	outData	D	Output	AES CFB (**)
TPM2_EncryptDecrypt	outData	D	Output	AES CFB (**)
TPM2_EncryptDecrypt2	inData	D	Input	AES CFB (*)
	outData	D	Output	AES CFB (**)

(*): Parameter decryption is ensured by use of a decryption session (attribute DECRYPT set)

(**): Parameter encryption is ensured by use of an encryption session (attribute ENCRYPT set). This is mandatory if output data is a CSP.

3.3.2 *Key transport*

Relative security strength has been calculated for each cryptographic algorithm supported by the module and used for key transport. TPM FW prevents use of key in a transport scheme with lower strength than the transported key.

Table 25: Cryptographic Functions

Algorithm	Comparable number of bits of security
RSA OAEP (2048 bits)	112
ECDH (P-256 curve)	128
ECDH (P-384 curve)	192
AES CFB (128 bits)[1]	128
AES CFB (256 bits)[2]	256

[1] AES is used in conjunction with HMAC approved authentication method (**[SP800-38F]** compliant)
[2] AES is used in conjunction with HMAC approved authentication method (**[SP800-38F]** compliant)

4 SELF-TESTS

Self-tests run by the cryptographic module are split into three categories:

- Self-tests on first boot of a FW (security module first boot or post-field upgrade)
- Self-tests on subsequent boots of a FW
- Conditional self-tests

The security module takes benefit of the IG 9.11 from **[FIPS IG]** to reduce the number of self-tests to run at each start-up. The power-on self-tests do not require operator intervention in order to run. The security module outputs an "error" Return Code via the status interface when the error state is entered due to a failed self-test. While in error state, security module does not perform any cryptographic functions and all data output via the data output interface are inhibited.

If power-on self-tests have passed successfully, no status is indicated but commands that require self-tests to be completed can be successfully executed.

4.1 Self-tests on first boot of a FW

Power-on self-tests execution completes all tests except KATs on asymmetric algorithms (RSA, ECDSA, ECDH). Completion of power-on self-tests allows the TPM to be in a limited approved mode allowing to process only a subset of TPM commands (see §1.7.1.1).

To switch from limited approved mode to full approved mode, operator shall execute TPM_SelfTest command with parameter full set to YES. This command requests the module to switch mode by executing all self-tests listed in Table 27 : Asymmetric cryptography self-tests list (power-up self-tests and the remaining self-tests, that mainly concern asymmetric cryptography).

4.1.1 Power-up tests list

Table 26 : Power-up self-tests list

Algorithm tested	Test description
SHA1	SHA1 computation on known data (16 bytes) and comparison of output to the expected digest (20 bytes)
SHA256	SHA256 computation on known data (16 bytes) and comparison of output to the expected digest (32 bytes)
SHA384	SHA384 computation on known data (16 bytes) and comparison of output to the expected digest (48 bytes)
SHA3_256	SHA3_256 computation on known data (16 bytes) and comparison of output to the expected digest (32 bytes)
HMAC SHA1	HMAC-SHA1 computation on known data (16 bytes) / known key (16 bytes, same value as data) and comparison of output to the expected MAC (20 bytes). Self-test allows validating the secure SHA algorithm also used in standalone (out of HMAC context).
KDF SP800-108	KDF (based on SHA1) computation on known data (16 bytes) / known label ("TEST") and comparison of output to the expected value (32 bytes).
Hash-DRBG	Instantiate, Generate and Reseed API are tested in a single test sequence in accordance with §11.3 of **[SP800-90A]**. Output of HDRBG (55 bytes) is compared to a reference value.
AES	AES CFB encryption is done on known data (32 bytes) / known key (16 bytes) and known IV (16 bytes, same value as key). The 32 bytes output data are compared to the expected reference data. If comparison succeeds, AES CFB decryption is done on encrypted data with same key & same IV as encryption. 32 bytes output are compared to the initial plaintext data.

Triple-DES	Triple-DES CFB encryption is done on known data (32 bytes) / known key (24 bytes) and known IV (8 bytes). The 32 bytes output data are compared to the expected reference data. If comparison succeeds, Triple-DES CFB decryption is done on encrypted data with same key & same IV as encryption. 32 bytes output are compared to the initial plaintext data.
FW integrity	FW integrity is verified by computing an EDC (CRC-16 ISO 13239) and comparing it to reference values. FW integrity is verified during boot sequence before execution of one of the code block (CML and TPM) and during full self-tests execution. If failure is detected during boot sequence, TPM enters an infinite reset loop that can be exit only by power-off/power-on sequence. In failure is detected during self-tests, status is set to FAIL and error is returned.
HW integrity	HW integrity is guaranteed via check of HW sensors. If failure is detected during boot sequence, status is set to FAIL and error is returned.

4.1.2 *Asymmetric cryptography self-tests list*

Table 27 : Asymmetric cryptography self-tests list

Algorithm tested	Test description
RSA	A known key is loaded (2048 bits length). Signature RSASSA-PKCS1-v1_5 is generated on known data (20 bytes). Output of signature is compared to a reference signature. Signature verification is performed on the generated signature.
ECDH	Primitive "Z" Computation KAT is implemented: a known private key d (32 bytes length) is used with a known point P of NIST P-256 curve to compute P = dQ. Q is compare to known reference point.
ECDSA	A known private key (256 bits) is used to generate ECDSA signature based on NIST P-256 curve. Output of signature is compared to a reference signature. Signature verification is performed on the generated signature.

4.2 Self-tests on subsequent boots of a FW

On all subsequent boots of a FW, only the integrity of the FW (CML and TPM) as well as the integrity of the HW are verified as authorized per IG9.11 **([FIPS IG]).**

Table 28 : Power-up self-tests list

Algorithm tested	Test description
FW integrity	FW integrity is verified by computing an EDC (CRC-16 ISO 13239) and comparing it to reference values. FW integrity is verified during boot sequence before execution of one of the code block (CML and TPM) and during full self-tests execution. If failure is detected during boot sequence, TPM enters an infinite reset loop that can be exit only by power-off/power-on sequence. If failure is detected during self-tests, status is set to FAIL and error is returned.
HW integrity	HW integrity is guaranteed via check of HW sensors. If failure is detected during boot sequence, status is set to FAIL and error is returned.

4.3 Conditional tests list

Table 29 : TPM conditional tests

Algorithm tested	Test description
FW integrity	FW integrity is verified by computing an EDC (CRC-16 ISO 13239) and comparing it to reference values.
Hash-DRBG	Each 32 bytes of generated data are compared to the previous generated data. If data are equal, status is set to FAIL and error is returned.
NDRNG	TPM performs AIS31 statistical test verification on NDRNG output and continuous HW self-tests (AS09.42) on NDRNG 48-bits output sequence. If test fails, TRNG_ERR bit is raised in SEC_STAT register. Status is set to FAIL and error is returned.
FW load	During field upgrade procedure, several checks are performed before authorizing the FW to be upgraded: - Verification of signature (RSASSA-PSS) on the first data blob to ensure authentication of the FW - Verification of digest (SHA256) on each subsequent blob to guarantee integrity of the full FW.
RSA key generation	A new RSA key is generated or retrieved from pre-computed keys (done in BKG). Depending on the key purpose (signing or encrypting) indicated in sign attribute of the key, en/decryption or signing/verification is done on known data (16 bytes).
ECC key generation	On each ECC key generation, an ECDSA signature is generated (k is fixed and m varies) and verified on curve NIST P-256 or NIST P-384.
TDES key generation	TDES key generation process consists in generating a pseudo-random value from KDFa and checking that this value passes the following conditional tests to be considered and next used as a functional TDES key. Conditional tests are: 1. Check that the 3 TDES cryptographic keys are different: Key_1 != Key_2, Key_2 != Key_3, Key_1 != Key_3 (Keying option 1 from §3.2 of **[SP800-67]**) 2. Key is not one of the weak key listed in §3.4.2 of **[SP800-67]** In case of failure, new pseudo-random values are generated until a valid TDES key is found.

4.4 Verification

Successful completion of self-tests can be verified through use of TPM2_GetTestResult command. The first 4 bytes of response indicate self-tests status. If they are equal to 0, self-tests completed successfully. If not, the subsequent 4 bytes indicate the list of algorithms not fully self-tested.

5 PHYSICAL SECURITY POLICY

The security module meets Physical Security protection requirements for FIPS level 3.

CSPs are physically protected from unexpected disclosure and modification. Security module is tamper evident, encapsulated in a hard opaque package to prevent direct observation of internal security components. Regular visual inspection must be conducted by user to check that HW integrity of the chip has not been damaged.

Physical security protection mechanisms that assure that CSPs remain protected from unauthorized disclosure, usage, modification or deletion, are described in “Mitigations of other attacks” section.

Nominal operating conditions for the security module are:

- **Voltage:** 1.8V or 3.3V (±10%).
- **Frequency:** System clock is created by an internal oscillator.

Hardness testing was only performed at ambient temperature. No assurance is provided for Level 3 hardness conformance at any other temperature.

6 OPERATIONAL ENVIRONMENT

Module operational environment is “limited modifiable” because TPM FW can only be modified through field upgrade service (use of TPM2_VendorFieldUpgradeStart and TPM2_VendorFieldUpgradeData commands). The non-upgradable code block (CML) is non-modifiable.

FIPS 140-2 level 1 & 2 operational environment requirements of **[FIPS 140-2]** section 4.6.1 are then not applicable to the security module.

New firmware versions within the scope of this validation must be validated through the FIPS 140-2 CMVP. Any other firmware loaded into this module is out of the scope of this validation and require a separate FIPS 140-2 validation.

7 MITIGATIONS OF OTHER ATTACKS

The security module meets Physical Security protection requirements for FIPS level 3.

7.1 Internal Tamper Detection

The security module contains an active metal shield that covers the internal TPM circuitry and memory components. Cutting, removing or modifying the shield layer will cause the TPM to Reset and enter a SHUTDOWN mode.

7.2 Environmental protection

The security module contains circuitry which will detect environmental conditions outside the range described in the product datasheet. Power supply voltage is continuously monitored. If conditions exist outside the range determined by the TPM tamper detection circuitry, the chip will reset and will enter a FAILURE mode. The chip will remain Reset and in FAIL mode as long as the environmental condition causing the tamper event persists.

8 REFERENCES

Reference	Document
[ST33TPHF2XSPI DS]	ST33TPHF2XSPI Datasheet, STMicroelectronics, July 2019
[ST33TPHF2XI2C DS]	ST33TPHF2XI2C Datasheet, STMicroelectronics, July 2019
[ST33GTPMASPI DS]	ST33GTPMASPI Datasheet, STMicroelectronics, July 2019
[ST33GTPMAI2C DS]	ST33GTPMAI2C Datasheet, STMicroelectronics, July 2019
[TPM2.0 Part1 r1.38]	TPM2.0 Main, Part 1, Architecture, rev 1.38, TCG
[TPM2.0 Part2 r1. 38]	TPM2.0 Main, Part 2, Structures, rev 1. 38, TCG
[TPM2.0 Part3 r1. 38]	TPM2.0 Main, Part 3, Commands, rev 1. 38, TCG
[TPM2.0 Part4 r1. 38]	TPM2.0 Main, Part 4, Supporting routines, rev 1. 38, TCG
[PTP 1.03]	TCG PC Client Platform TPM Profile (PTP) Specification, rev. 1.03 with errata
[FIPS 140-2]	FIPS PUB 140-2, Security Requirements for Cryptographic Modules / National Institute of Standards and Technology (NIST), CHANGE NOTICES (12-03-2002)
[FIPS DTR]	National Institute of Standards and Technology and Communications Security, *Derived Test Requirements(DTR) for FIPS PUB 140-2, Security Requirements for Cryptographic Modules*
[FIPS IG]	National Institute of Standards and Technology and Communications Security, *Implementation Guidance for FIPS PUB 140-2 and the Cryptographic Module Validation Program*
[FIPS 180-4]	National Institute of Standards and Technology, *Secure Hash Standard*, Federal Information Processing Standards Publication 180-4, March 2012
[FIPS 186-4]	National Institute of Standards and Technology, *Digital Signature Standard (DSS),* Federal Information Processing Standards Publication 186-4, July 2013
[FIPS 197]	National Institute of Standards and Technology, *Advanced Encryption Standard (AES),* Federal Information Processing Standards Publication 197, November 2001
[SP800-135]	National Institute of Standards and Technology, *Existing Application-Specific Key Derivation Function Validation System*, September 2015.
[SP800-108]	National Institute of Standards and Technology, *Recommendation for Key Derivation Using Pseudorandom Functions*, October 2009.
[SP800-131Ar2]	National Institute of Standards and Technology, *Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths*, March 2019.

Reference	Document
[FIPS 198-1]	National Institute of Standards and Technology, *The Keyed-Hash Message Authentication Code,* NIST Computer Security Division Page 3 07/26/2011, *(HMAC)*, Federal Information Processing Standards Publication 198-1, July, 2008
[SP800-90A]	National Institute of Standards and Technology, *Recommendation for Random Number Generation Using Deterministic Random Bit Generators*, January 2012.
[SP800-38F]	National Institute of Standards and Technology, *Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping,* December 2012.
[SP800-56A] Rev 3	National Institute of Standards and Technology, *Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography,* April 2018.
[SP800-56B]	National Institute of Standards and Technology, *Recommendation for Pair-Wise Key-Establishment Using Integer Factorization Cryptography,* August 2009
[SP800-56C] Rev 1	National Institute of Standards and Technology, *Recommendation for Key-Derivation Methods in Key-Establishment Schemes,* April 2018
[SP800-67]	National Institute of Standards and Technology, *Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher,* January 2012.
[FIPS 140-2 TPM]	TCG FIPS 140-2 Guidance for TPM2.0, v1.0, TCG

9 ACRONYMS

Term	Definition
AES	Advanced Encryption Standard
CO	Crypto Officer
DES	Data Encryption Standard
DSAP	Delegate Specific Authorization Protocol
EK	Endorsement Key
FIPS	Federal Information Processing Standard
FUM	Field Upgrade Mode
GPIO	General Purpose I/O
HMAC	Keyed-Hashing for Message Authentication
NIST	National Institute of Standards and Technology
NV	Non-volatile (memory)
OIAP	Object-Independent Authorization Protocol
OSAP	Object Specific Authorization Protocol
PCR	Platform Configuration Register
RSA	Rivest Shamir Adelman
RTM	Root of Trust for Measurement
RTR	Root of Trust for Reporting
SHA	Secure Hash Algorithm
SPI	Serial Peripheral Interface
SRK	Storage Root Key
TCG	Trusted Computed Group
TPM	Trusted Platform Module
TSS	TPM Software Stack

IMPORTANT NOTICE – PLEASE READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgement.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of Purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2020 STMicroelectronics - All rights reserved
www.st.com