# jForté/HAT
# Cryptographic Module Security Policy
# Version 3.3

Litronic, Inc.

17861 Cartwright Road

Irvine, CA 92614

litronic

a sāflink company

# Table of Contents

# Table of Contents (continued)

# List of Tables

# List of Figures

# 1. Introduction

## 1.1 Purpose

This document describes the security policies implemented by the jForté/HAT cryptographic module and how its design enforces them. These security policies and enforcement methods apply specifically to the following configuration of the jForté/HAT cryptographic module:

- Firmware version 3.1
- Module version J002
- Part number 078-2010-02

## 1.2 Scope

The Cryptographic Security Policy specifies the security rules under which the cryptographic module operates. It does not describe the requirements for the entire system.

## 1.3 References

- Atmel Corporation, *High Security Chip (AT91SC321RC) Specification*, Revision 2009A, June, 2001.
- Department of Defense, *DoD PKI and KMI Token Protection Profile (Medium Robustness)*, Version 3.0, March 22, 2002.
- National Institute of Standards and Technology (NIST), FIPS Pub 46-3, *Data Encryption Standards*, October 25, 1999.
- National Institute of Standards and Technology (NIST), FIPS Pub 140-2, *Security Requirements for Cryptographic Modules*, December 3, 2002.
- National Institute of Standards and Technology (NIST), FIPS Pub 180-2, *Secure Hash Standard*, August 1, 2002.
- National Institute of Standards and Technology (NIST), FIPS Pub 186-2, *Digital Signature Standard (DSS)*, January 27, 2000.
- National Institute of Standards and Technology (NIST), *SKIPJACK and KEA Algorithm Specifications*, Version 2.1, June 10, 1998.
- RSA Security Inc., *Public-Key Cryptographic Standard PKCS #1*, Version 2.1, June 14, 2002.

## 1.4 Definitions

**Transient Key**—A cryptographic key that is automatically generated by the jForté/HAT on start-up and zeroized upon loss of power.

In addition, Table 15 on page 32 provides a list of abbreviations and acronyms.

## 1.5 Module Overview

The jForté/High Assurance Token (jForté/HAT) device, as shown in Figure 1, is a single-chip cryptographic device that supports multiple identities and applications. The cryptographic core is defined when the CPU executes instructions at the highest possible level (Kernel Mode), and has exclusive access to all the available hardware registers. This includes the ASIC registers and the various support components and circuits.

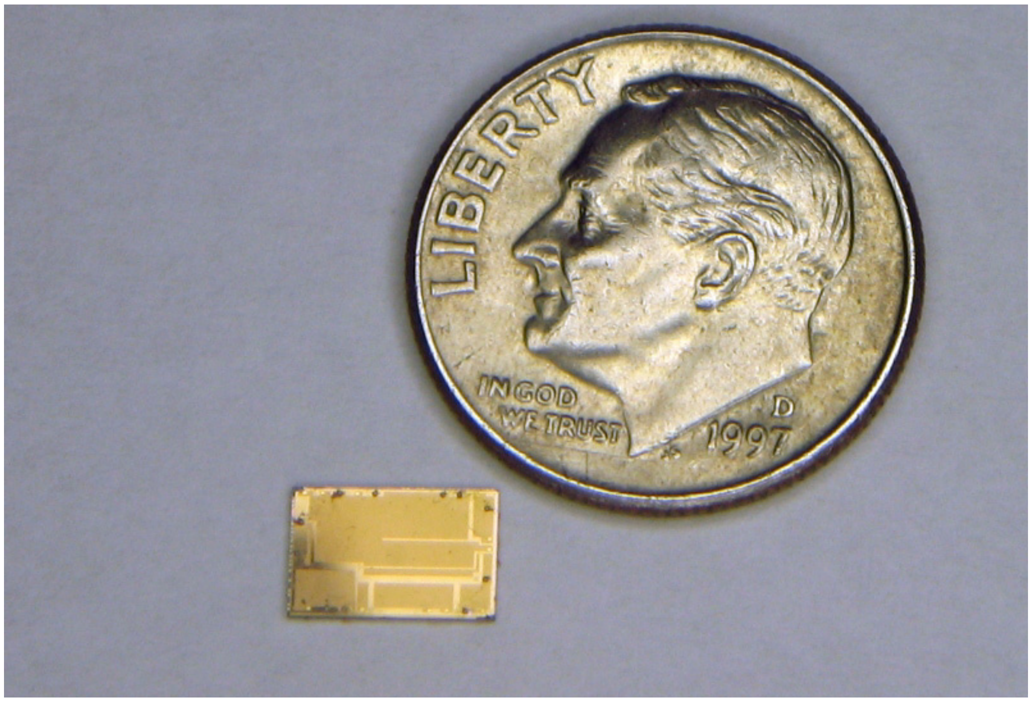


**Figure 1 - The jForté/HAT Cryptographic Module**

Table 1 and Figure 2 on page 4 show the die pad assignments.

**Table 1 - jForté/HAT Die Pad Coordinates**

| Pad Number | Signal Name | Pad Coordinates | | | |
|---|---|---|---|---|---|
| | | 4.837 x 6.81 mm Die<br>125.02 um Pad | | 190.4 x 270.9 mil Die<br>4.9 mill Pad | |
| | | X<br>(in micrometers) | Y<br>(in micrometers) | X<br>(in millimeters) | Y<br>(in millimeters) |
| 1 | SPI_MOSI | –1895.64 | 2822.92 | –74.631 | 111.139 |
| 2 | GND | –1895.67 | 2216.10 | –74.633 | 87.248 |
| 3 | USB_CLK48 | –1895.67 | 1621.62 | –74.633 | 63.843 |
| 4 | SPI_CLK | –1895.64 | 1008.28 | –74.631 | 39.696 |

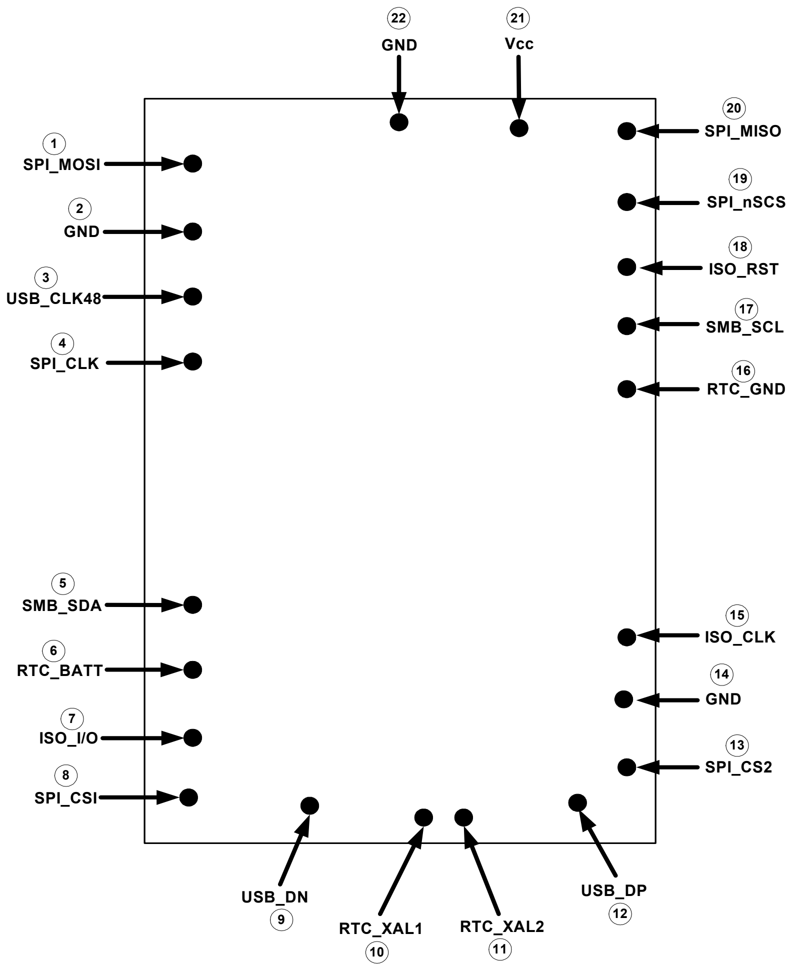
**Table 1 - jForté/HAT Die Pad Coordinates (continued)**

| Pad Number | Signal Name | Pad Coordinates | | | |
|---|---|---|---|---|---|
| | | 4.837 x 6.81 mm Die 125.02 um Pad | | 190.4 x 270.9 mil Die 4.9 mill Pad | |
| | | X (in micrometers) | Y (in micrometers) | X (in millimeters) | Y (in millimeters) |
| 5 | SMB_SDA | –1895.46 | –1248.56 | –74.624 | –49.156 |
| 6 | RTC_BATT | –1895.67 | –1848.49 | –74.633 | –72.775 |
| 7 | ISO_I/O | –1895.46 | –2468.20 | –74.624 | –97.173 |
| 8 | SPI_CS1 | –1895.60 | –3070.17 | –74.630 | –120.873 |
| 9 | USB_DN | –771.23 | –3094.49 | –30.363 | –121.830 |
| 10 | RTX_XAL1 | 288.96 | –3214.37 | 11.376 | –126.550 |
| 11 | RTX_XAL2 | 650.19 | –3214.37 | 25.598 | –126.550 |
| 12 | USB_DP | 1729.24 | –3093.13 | 68.081 | –121.777 |
| 13 | SPI_CS2 | 2198.94 | –2748.24 | 86.573 | –108.198 |
| 14 | GND | 2199.02 | –2135.77 | 86.575 | –84.085 |
| 15 | ISO_CLK | 2199.01 | 1580.46 | 86.575 | –62.223 |
| 16 | RTC_GND | 2199.05 | 728.00 | 86.577 | 28.661 |
| 17 | SMB_SCL | 2199.02 | 1333.29 | 86.575 | 52.492 |
| 18 | ISO_RST | 2199.02 | 1920.14 | 86.575 | 75.596 |
| 19 | SPI_nSCS | 2198.98 | 2516.01 | 86.574 | 99.056 |
| 20 | SPI_MISO | 2198.98 | 3128.69 | 86.574 | 123.177 |
| 21 | Vcc | 1144.78 | 3159.83 | 45.070 | 124.403 |
| 22 | GND | 40.60 | 3258.75 | 1.598 | 128.691 |

The primary purpose of the jForté/HAT is to provide a platform for Java applets to perform on-device cryptographic operations in a FIPS 140-2 compliant environment. The jForté/HAT supports commercial RSA, DES (transitional phase only – valid until May 19, 2007), and TDES algorithms.

The dual-mode interface feature for data input and output supports interoperability with either:
- The standard ISO/IEC 7816 smart card interface
- A Universal Serial Bus (USB) smart card reader

The jForté/HAT incorporates a USB 1.1-compliant macro-cell that allows data transmission rates of 12 megabits per second, which is fast enough for performing bulk cryptographic processing.



**Wafer Flat (this side of die)**

**Figure 2 - jForté/HAT Die Signal Pad Approximate Locations**

### 1.5.1 Cryptographic Module Specifications

The jForté/HAT cryptographic module supports the set of commands defined by Global Platform 2.0.1 that allows for two-factor authentication of a personal security device to application infrastructures supporting them. This protocol allows for strong cryptographic algorithms (TDES) for authentication and data protection. This environment supports multiple identities and a secure channel of communication to the jForté/HAT device.

The cryptographic module consists of a combination of firmware and hardware to support an environment where applets can be securely loaded and executed. The main security kernel and virtual runtime engine is stored in ROM. Applets, key material, and critical security parameters are stored in 64K bytes of available EEPROM.

The focus of the cryptographic validation effort will be on the kernel layer of code that manages all keying material and the hardware cryptographic engine. The virtual machine for executing applets and running the Global Platform

Card Manager applets is built on top of this secure kernel. If applets are loaded onto the jForté/HAT device, the presence of the applets will compromise the current FIPS 140-2 validation of the cryptographic module.

**Note:** While the jForté/HAT cryptographic module *supports* applets that may be loaded onto the module, its FIPS-validated configuration does not include applets. A jForté/HAT cryptographic module with applets would require reevaluation for FIPS validation.

### 1.5.2 External Ports

The external ports are:

- USB
- ISO 7816
- SPI

When the jForté/HAT is presented in the optional USB token form factor, the external port pins that provide the USB interface are connected to a standard USB four-contact connector as shown in Table 2.

When the jForté/HAT is presented in the optional smart card form factor, the external port pins that provide both the USB and ISO 7816 interfaces are connected to the ISO 7816-3 card contacts as shown in Figure 3 and Table 3 on page 6.

**Interface Exceptions:**

1. While the System Management Bus (SMB) data and clock signals are present in the application-specific integrated circuit (ASIC), neither is used by the jForté/HAT cryptographic module.
2. While the Serial Peripheral Interface (SPI) signal pads are present in the application-specific integrated circuit (ASIC), this interface is not supported by the current product firmware of the jForté/HAT cryptographic module. It is reserved for future enhancements.

**Note:** The universal synchronous/asynchronous receiver transmitter (USART) circuits for the SMB and SPI interfaces employ a technique called *clock gating*. Clock signals are not routed to those devices unless they are initiated by the system kernel. Since the jForté/HAT firmware does not initialize the SMB and SPI interfaces, both devices are disabled and cannot be activated by application software.

**Table 2 - Optional USB Token Four-Wire Contact Assignments**

| Contact | | Signal | | Typical Wiring Assignment |
|---|---|---|---|---|
| **Die** | **USB** | **Signal Name** | **Description** | |
| 21 | 1 | **VBus** | Power to the chip | Red |
| 9 | 2 | **D-** | D– component of the differential data bus pair | White |
| 12 | 3 | **D+** | D+ component of the differential data bus pair | Green |
| 2, 14, & 22 | 4 | **GND** | Ground | Black |
| | Shell | **Shield** | Shield | Drain wire |

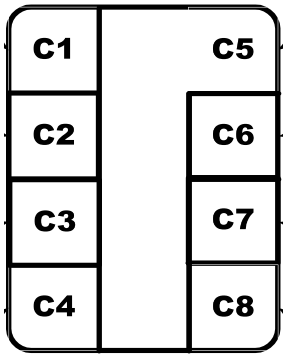


**Figure 3 - Optional Smart Card Contact Configuration**

**Table 3 - Optional ISO 7816 and USB 1.1 Smart Card Contact Assignments**

| **Contact** | | **Signal Name** | **Signal Description** |
|---|---|---|---|
| **Die** | **Card** | | |
| 21 | C1 | **Vcc** | Power to the card: +5 VDC |
| 18 | C2 | **RST** | RESET: A proprietary Litronic, Inc. RESET contact with a threefold purpose:<br>• Used by the module to automatically sense either ISO 7816-3 or Litronic, Inc. USB 1.1 reader mode of operation:<br>  • Logic-low = ISO 7816-3 mode<br>  • Logic-high = USB 1.1 mode<br>• Request for software reset by reader—for ISO 7816 mode only<br>• The module can assert a logic-low to turn on an activity LED on the smart card reader—for Litronic, Inc. USB 1.1 mode only |
| 15 | C3 | **ISO CLK** | ISO 7816 Clock: 3.579 MHz, nominal |
| 12 | C4 | **D+** | USB data (+) pull-up resistor employed on the Litronic, Inc. reader—for Litronic, Inc. USB 1.1 mode only |
| 2, 14, & 22 | C5 | **GND** | Ground |
| 3 | C6 | **USB CLK** | USB clock, 48 MHz, nominal—for Litronic, Inc. USB 1.1 mode only |
| 7 | C7 | **ISO I/O** | ISO 7816 input/output, the data input/output contact |
| 9 | C8 | **D–** | USB data (–) pull-up resistor employed on the Litronic, Inc. reader—for Litronic, Inc. USB 1.1 mode only |

### 1.5.3 Logical Interfaces

The four logical interfaces are:

- Data Input interface
- Data Output interface
- Control Input interface
- Status Output interface

### 1.5.4 Power Interface

The jForté/HAT cryptographic module obtains all of its power from external sources. The 5-volt Vcc power is used by the internal (on-chip) power regulator to supply switched power to some of the I/O pins. The power interface pins on the die product are identified in Table 4 on page 8 as pads 2, 14, 21, and 22.

| When optionally package as . . . | Vcc is provided on . . . | And ground is provided on . . . |
|---|---|---|
| An ISO 7816 smart card configuration | Contact C1 | Contact C5 |
| A 24-pin SOIC | Pin 24 | Pins 1, 4, and 16 |

The jForté/HAT does not provide power to any other device.

### 1.5.5 Mapping Logical Interfaces to Physical Interfaces

The logical interfaces are mapped from the physical die pads as shown in Table 4 on page 8.

**Note:** All of the die interface signals are available when the jForté/HAT is optionally packaged in the standard SOIC form factor.

### 1.5.6 Interface Security

At no time are private and secret keys, authentication data, or CSPs imported or exported in plaintext. A separate path for these items is provided through the GlobalPlatform Secure Channel facility.

### 1.5.7 Interface Disconnection During Key-Generation

At start-up, either the ISO 7816 or the USB interface is established as the communications interface with the host for the duration of the session. During a session, all logical paths are carried on the chosen physical path.

The interface with the host computer is a command-response interface, where all communication with the host is inhibited until a command is executed. In particular, a command to generate a key or key pair causes communication with the host to be suspended until the key generation is either completed successfully or aborted. At the conclusion of the operation, a response message is returned to the host containing only the return code.

**Table 4 - Map from Die Pads to Logical Interfaces**

| Interface Classification | | | | Die Pad No. | Signal Name | Description | ISO | USB | SPI |
|---|---|---|---|---|---|---|---|---|---|
| Control Input | Data In | Data Out | Status Out | | | | | | |
| | √ | | | 1 | SPI_MOSI | SPI data in | | | √ |
| | | | | 2 | GND | Ground | | | |
| √ | | | | 3 | USB_CLK48 | USB clock | | √ | |
| √ | | | | 4 | SPI_CLK | SPI clock | | | √ |
| √ | √ | √ | √ | 5 | SMB_SDA | SMB data I/O (deactivated by Firmware) | | | |
| | | | | 6 | RTC_BATT | Real time clock supply voltage | | | |
| √ | √ | √ | √ | 7 | ISO_I/O | USART I/O for ISO mode of operation | √ | | |
| | | | | 8 | SPI_CS1 | SPI chip select (output) | | | √ |
| √ | √ | √ | √ | 9 | USB_DN | USB data I/O | | √ | |
| √ | | | | 10 | RTX_XAL1 | Real time clock crystal | | | |
| √ | | | | 11 | RTX_XAL2 | Real time clock crystal | | | |
| √ | | √ | √ | 12 | USB_DP | USB data I/O | | √ | |
| | | | | 13 | SPI_CS2 | SPI chip select (output) | | | √ |
| | | | | 14 | GND | Ground | | | |
| √ | | | | 15 | ISO_CLK | Clock for ISO mode of operation | √ | | |
| | | | | 16 | RTC_GND | Real time clock ground | | | |
| √ | | | | 17 | SMB_SCL | SMB clock (deactivated by Firmware) | | | |
| √ | | | √ | 18 | ISO_RST | Chip reset for ISO mode of operation. If in USB mode, the module can assert logic low to turn on an activity LED on a smart card reader. | √ | √ | |
| | | | | 19 | SPI_nSCS | Negative assertion, SPI chip select (output) | | | √ |
| | | √ | | 20 | SPI_MISO | SPI data out | | | √ |
| | | | | 21 | Vcc | Supply voltage | √ | √ | √ |
| | | | | 22 | GND | Ground | √ | √ | √ |

**Note:** The SPI interface is not supported by the current jForté/HAT product firmware. The SPI interface is reserved for future product enhancements.

### 1.5.8 Built-in Self-test Functionality

The built-in self-test (BIST) is automatically executed when power is applied. Cryptographic services to the user are inhibited until all self tests have completed successfully.

## 2. Security Level

The jForté/HAT cryptographic module meets the overall requirements applicable to Level 3 security of FIPS 140-2, as shown in Table 5.

**Table 5 - Module Security Level Specifications**

| Security Requirements Section | Level |
|---|---|
| Cryptographic Module Specification | 3 |
| Module Ports and Interfaces | 3 |
| Roles, Services and Authentication | 3 |
| Finite State Model | 3 |
| Physical Security | 4 |
| Operational Environment | N/A |
| Cryptographic Key Management | 3 |
| EMI/EMC | 3 |
| Self-Tests | 3 |
| Design Assurance | 3 |
| Mitigation of Other Attacks | 3 |

## 3. Identification and Authentication Policy

Table 6 shows the roles that are supported by the jForté/HAT cryptographic module.

**Table 6 - Roles and Required Identification and Authentication**

| Role | Authentication | |
|---|---|---|
| | **Identification** | **Authentication** |
| **Cryptographic Officer (CO)—**This role is assumed to perform cryptographic initialization or a management function such as module initialization, input/output of cryptographic keys and CSPs. | The identity of the Cryptographic Officer is specified by providing the Cryptographic Officer's key-ID. | Uses INITIALIZE UPDATE and EXTERNAL AUTHENTICATE commands to prove knowledge of a TDES key set. |
| **User—**The User is the developer of applets that will be loaded onto the jForté/HAT. The User has the suite of Java and Global Platform APIs supported by the jForté/HAT virtual machine. | The identity of the User is specified by providing the User's key-ID. | Uses INITIALIZE UPDATE and EXTERNAL AUTHENTICATE commands to prove knowledge of a TDES key set. |
| **Unauthenticated—**This role can have access to administrative services that do not perform cryptographic operations. | None. | None. |
| **Firmware Loader—**This role updates, modifies, or enhances the firmware that executes on the jForté/HAT. To maintain its validation, the jForté/HAT must be revalidated after the firmware has been updated. | The identity of the Firmware Loader role is specified by selecting the Load Package service. | Uses the on-chip RSA Manufacturer Download Authentication Key (MDAK) public key to prove that the downloaded firmware package was issued by the authorized agency. |

The jForté/HAT uses identity-based operator authentication through the selection of security domains to enforce the separation of roles.

Table 7 presents the probabilities of success by random access attempts.

**Table 7 - Strengths of Authentication Mechanisms**

| Authentication Mechanism | Strength of Mechanism |
|---|---|
| INITIALIZE UPDATE and EXTERNAL AUTHENTICATE | This technique uses the strength of a TDES key to validate that the operator can encrypt a known challenge sent by the jForté/HAT. If the jForté/HAT can decrypt the cipher text from the operator and retrieve the original known challenge, the operator is assumed to have knowledge of the TDES key.<br>The INITIALIZE UPDATE and EXTERNAL AUTHENTICATE sequence can be issued at 5.5 times per second, for a maximum of 330 random attempts per minute.<br>The probability of a single random attempt succeeding is much less than 1 in 1,000,000. The probability of a successful random attempt during a one-minute period is less than 1 in 100,000. |

(continued)

**Table 7 - Strengths of Authentication Mechanisms (continued)**

| Authentication Mechanism | Strength of Mechanism |
|---|---|
| LOAD PACKAGE | This technique uses the strength of MDAK 1024-bit RSA key to authenticate the Firmware Loader role.<br>The LOAD PACKAGE service can be attempted up to 8.6 times per second, for a maximum of 517 attempts during a one-minute period.<br>The probability of a single random attempt succeeding is much less than 1 in 1,000,000. The probability of a successful random attempt during a one-minute period is less than 1 in 100,000. |

# 4. Access Control Policy

## 4.1 Roles and Services

**Table 8 - Services Authorized for Roles**

| | | **For These Role Types** | | | |
|---|---|---|---|---|---|
| **This Service** | **Performs This Function** | **Firmware Loader** | **Unauthenticated** | **User** | **CO** |
| DELETE | Deletes a uniquely identifiable object such as an executable load file or an application. | | | | √ |
| ECHO COMPLEMENT | Tests the communication path to the jForté/HAT.<br>This command returns the same buffer given as input. | | √ | | |
| EXTERNAL AUTHENTICATE | Used by the jForté/HAT to authenticate the host and determine the level of security required for all subsequent commands. | | | √ | √ |
| GET DATA | Retrieves a single data object.<br>The data object may contain information pertaining to a key. Key material is not permitted to be extracted from the jForté/HAT.<br>Also obtains the Version Number and Approved Mode indicator. | | | √ | √ |
| GET jFORTE STATUS | Returns the jForté/HAT cryptographic status including the current state. | | √ | | |
| GET RESPONSE | Retrieves the output of the previously executed command. | | √ | √ | √ |
| GET STATUS | Retrieves status information according to a given match/search criteria. | | | √ | √ |
| INITIALIZE UPDATE | Transmits jForté/HAT and secure channel session data between the jForté/HAT and the host.<br>This command initiates the initiation of a secure channel session. | | √ | √ | √ |
| INSTALL | Issued to a security domain to initiate or perform the various steps required for jForté/HAT content management.<br>Requires a secure channel. | | | | √ |
| LOAD | Transfers a load file to the jForté/HAT, as identified by the previous INSTALL command. The load file is divided into smaller components for transmission.<br>Requires a secure channel. | | | | √ |

(continued)

**Table 8 - Services Authorized for Roles (continued)**

| This Service | Performs This Function | For These Role Types | | | |
|---|---|---|---|---|---|
| | | Firmware Loader | Unauthenticated | User | CO |
| LOAD PACKAGE | Loads, authenticates and installs digitally signed firmware packages on the jForté/HAT cryptographic module. The firmware packages are PKCS#1 signed by the token manufacturer (in this case, Litronic, Inc.).<br>The command is used to patch firmware bugs.<br>If the authentication fails, the package is not installed and the service returns an error. | √ | | | |
| PUT KEY | Adds either:<br>• A single new key<br>• Multiple new keys<br>Requires a secure channel. | | | √ | √ |
| SELECT | Selects an application.<br>This is the selection process that is used to identify what application (or applet) should be active and therefore what identity is selected. | | √ | √ | √ |
| SET STATUS | Modifies the card Life Cycle State or the Application Life Cycle State. | | | √ | √ |
| STORE DATA | Transfers data to an application or the security domain processing the command. | | | √ | √ |

## 4.2 Cryptographic Keys and Critical Security Parameters

Table 9 describes the cryptographic keys and critical security parameters (CSPs) contained in the module.

**Table 9 - Cryptographic Key and Critical Security Parameter Descriptions**

| | | | **By These Role Types** | | | |
|---|---|---|---|---|---|---|
| | | | **Firmware Loader** | | | |
| | | | | **Unauthenticated** | | |
| | | | | | **User** | |
| **Key/CSP Name** | **Abbreviation** | **Description** | | | | **CO** |
| Application Domain Encryption Key | Aenc | This TDES key is used to authenticate during secure channel setup via INITIALIZE UPDATE and EXTERNAL AUTHENTICATE. The Senc key is generated from this key. | | | √ | √ |
| Application Domain Key Encryption Key | Akek | This TDES key is used to unwrap inbound keys. This key is copied to the working Skek key location after the secure channel is established. | | | √ | √ |
| Application Domain Message Authentication Key | Amac | This TDES key is used in the User authentication process to generate the Smac key. | | | √ | √ |
| Card Manager Encryption Key | Kenc | This TDES key is used to authenticate during secure channel setup via INITIALIZE UPDATE and EXTERNAL AUTHENTICATE. | | | | √ |
| Card Manager Key Encryption Key | Kkek | This TDES key is used to unwrap inbound keys. This key is copied to the working Skek key location after the secure channel is established. | | | | √ |
| Card Manager Message Authentication Key | Kmac | This TDES key is used in the CO authentication process. | | | | √ |
| Download Transport Key | DTK | This SKIPJACK key is used to decrypt the body of download packages. The purpose of this scheme is to safeguard the confidentiality of the download. | √ | | | |
| Session Encryption Key | Senc | This transient TDES key is used to decrypt data (APDUs) transmitted over the interface. | | | √ | √ |
| Session Key Encryption Key | Skek | This transient TDES key is used to unwrap inbound keys transmitted over the secure channel interface (logical). | | | √ | √ |
| Session Message Authentication Key | Smac | This transient TDES key is used to verify the message authentication code (MAC) of inbound messages transmitted over the secure channel interface (logical). | | | √ | √ |

**Note:** The Global Platform PIN (GPIN), can be used to authenticate the card holder after the jForté/HAT is deployed with applets. GPIN is 4 to 8 bytes and there is a retry counter that is set to 3.

A GPIN value is not loaded or utilized by the current product configuration. A Java Card utility applet would be required to load a value for GPIN onto the cryptographic module. This GPIN and retry counter may be used by future applets to authenticate the user.

The persistent Download Transport Key (DTK) is loaded into the module during manufacturing. All other persistent keys (Aenc, Amac, Akek, and Kenc, Kmac, and Kkek) are loaded into the module using the PUT KEY service.

**Note:** While the DTK is persistent, and is not destroyed by a loss of power, it is not permanent. It can be zeroized, as described in Table 12 on page 18.

The transient keys (Senc, Smac, and Skek) are generated within the cryptographic module (copied into RAM) from the appropriate persistent key .

## 4.3 Other Cryptographic Keys

The Manufacturer Download Authentication Key (MDAK), described in Table 10, is a nonsecret public key that is also stored as a persistent key on the cryptographic module. The MDAK public key is loaded into the cryptographic module in manufacturing, prior to the jForté/HAT being issued.

**Table 10 - MDAK Public Key Description**

| Key Name | Abbreviation | Description |
|---|---|---|
| Manufacturer Download Authentication Key | MDAK | This 1024-bit RSA public key is used to verify the:<br>• Integrity of the firmware update<br>• Signature of the Firmware Loader, which authenticates him/her into his/her role<br>**Note:** The MDAK is never replaced or modified. |

## 4.4 CSPs Modes of Access

Table 11 describes the relationships between the different module services and access to CSPs.

**Table 11 - Access Rights Within Services**

| Service | Cryptographic Key or CSP | Type of Access (RWE) | | |
|---|---|---|---|---|
| | | **Read** | **Write** | **Execute** |
| DELETE | Senc | | | √ |
| | Smac | | | √ |
| ECHO COMPLEMENT | None | | | |
| EXTERNAL AUTHENTICATE | Kenc, Aenc | | | √ |
| | Kmac, Amac | | | √ |
| | Kkek, Akek | √ | | |
| | Senc | | √ | |
| | Smac | | √ | |
| | Skek | | √ | |
| GET DATA | Senc | | | √ |
| | Smac | | | √ |
| GET jFORTE STATUS | None | | | |
| GET RESPONSE | Senc | | | √ |
| | Smac | | | √ |
| GET STATUS | None | | | |
| INITIALIZE UPDATE | Kenc, Aenc | | | √ |
| | Kmac, Amac | | | √ |
| INSTALL | Senc | | | √ |
| | Smac | | | √ |
| LOAD | Senc | | | √ |
| | Smac | | | √ |

(continued)

**Table 11 - Access Rights Within Services (continued)**

| Service | Cryptographic Key or CSP | Type of Access (RWE) | | |
|---|---|---|---|---|
| | | Read | Write | Execute |
| LOAD PACKAGE | MDAK | | | √ |
| | DTK | | | √ |
| PUT KEY | Senc | | | √ |
| | Smac | | | √ |
| | Skek | | | √ |
| | Kenc, Aenc | √ | √ | √ |
| | Kmac, Amac | √ | √ | √ |
| | Kkek, Akek | √ | √ | √ |
| SELECT | None | | | |
| SET STATUS | None | | | |
| STORE DATA | Senc | | | √ |
| | Smac | | | √ |

## 4.5 Methods for Zeroizing Cryptographic Keys

Table 12 lists and describes the methods that are used to zeroize cryptographic keys that are stored on the cryptographic module.

**Table 12 - Cryptographic Key Zeroization Methods**

| Key Type | Description |
|---|---|
| User Security Domain Keys | The CO role uses the PUT KEY service command to replace all of the persistent USER Security Domain keys (Aenc, Amac, and Akek) with all-zero values. |
| CO Security Domain Keys | The CO role uses the PUT KEY service command to replace all of the the persistent CO Security Domain keys (Kenc, Kmac, and Kkek) with all-zero values. |
| Download Transport Key | The Firmware Loader role uses the LOAD PACKAGE service command to load a special firmware utility package—the jForté DTK-zeroization Package—that replaces the persistent Download Transport Key in EEPROM memory with an all-zero value. |
| Transient Keys | Removing the jForté/HAT from power zeroizes all of the transient keys in RAM (Senc, Smac, and Skek). |

# 5. Security Rules

This section describes the security rules enforced by the jForté/HAT cryptographic module to implement the security requirements of a FIPS 140-2 Level 3 device.

## 5.1 Identities and Authentication

The Cryptographic Officer (CO) must prove to have the TDES key set stored in the jForté/HAT associated with the Global Platform Card Manager application. Through the INITIALIZE UPDATE and EXTERNAL AUTHENTICATE commands, the CO and jForté/HAT will mutually authenticate.

The User must prove to have the TDES key set stored in the jForté/HAT associated with the User security domain. Through the INITIALIZE UPDATE and EXTERNAL AUTHENTICATE commands, the User and jForté/HAT will mutually authenticate.

Authenticated log-in by either the Cryptographic Officer or by the User is cleared upon removal or power down of of the jForté/HAT.

Authentication of the Firmware Loader role uses the on-chip RSA Manufacturer Download Authentication Key (MDAK) public key to prove that a downloaded firmware package was issued by the authorized agency.

## 5.2 Access Control Security

- Keys must be loaded over a secure channel established by the Cryptographic Officer. The keys must be encrypted with the Skek.
- The Global Platform PIN can only be loaded over an encrypted secure channel.

## 5.3 Applet Loading Security

Applets are loaded over a secure channel to the jForté/HAT.

Applets that will run on the jForté/HAT must go through their own validation effort. Only FIPS 140-2 Level 3 applets should be loaded onto the jForté/HAT, and then the combined jForté/HAT and applets would require reevaluation for FIPS 140-2 validation.

# 6. Physical Security Policy

## 6.1 Physical Security Mechanisms

The jForté/HAT cryptographic module includes the following physical security mechanisms when engaging power:

- Environmental failure protection (EFP) features for temperature, voltage, internal clock frequency, and duty cycle are provided by immediate shutdown circuitry.
- A hard, opaque removal-resistant coating with hardness and adhesion characteristics covers the single-chip cryptographic module, and attempts to peel or pry the coating from the module results in serious damage to the module. (Not shown in Figure 1 on page 2.)
- The removal-resistant metal-alloy "Tin Roof" that coats the single-chip cryptographic module has solvency characteristics such that attempting to dissolve the coating would seriously damage the module (i.e., the module would not function). (The "Tin Roof" is visible through the transparent Silicon dioxide passivation layer shown in Figure 1 on page 2.)

### 6.1.1 Temperature Environmental Failure Protection

The Temperature Attack is a type of fault induction attack that utilizes temperature manipulation to cause processing errors within the cryptographic module.

An analysis of these errors and their patterns is an attempt to reverse engineer the cryptographic module, revealing certain features and implementations of cryptographic algorithms and, subsequently, revealing the values of cryptographic keys.

The jForté/HAT protects against the threat of noninvasive attacks based on temperature by providing a pair of autonomous analog circuits that continuously compares the output of a proportional-to-absolute-temperature (PTAT) transistor against an upper limit reference and against a lower limit reference.

If the output of the PTAT is higher than the upper limit reference or below the lower limit reference, an EFP shutdown is triggered.

The EFP shutdown disables the power regulator and closes a crowbar circuit from the regulated internal power distribution bus (VDD) to ground, immediately zeroizing all registers, SRAM, and plaintext keys stored in volatile memory. The EFP shutdown thereby causes the jForté/HAT to enter the Mute Mode error state until power is removed.

The jForté/HAT Application-Specific Integrated Circuit (ASIC) operates reliably over a temperature range of –20°C to +80°C. A reasonable working temperature range for humans is 0°C to +60°C. Therefore:

- An under-temperature EFP shutdown will be triggered between –20°C and 0°C
- An over-temperature EFP shutdown will be triggered between +60°C and +80°C

### 6.1.2 Voltage Environmental Failure Protection

The Voltage Attack is a type of fault induction attack that utilizes voltage manipulation to cause processing errors within the cryptographic module.

An analysis of these errors and their patterns is an attempt to reverse engineer the cryptographic module, revealing certain features and implementations of cryptographic algorithms and, subsequently, revealing the values of cryptographic keys.

The jForté/HAT protects against the threat of noninvasive attacks based on voltage by providing no external point to monitor or control the regulated internal power distribution bus (VDD).

In addition, a pair of voltage references are provided so that the voltage supplied to the chip can be compared against a low-voltage limit as well as a high-voltage limit. If the voltage is either higher than the upper limit reference or below the lower limit reference, an EFP shutdown is triggered.

The EFP shutdown disables the power regulator and closes a crowbar circuit from the regulated internal power distribution bus (VDD) to ground, immediately zeroizing all registers, SRAM, and plaintext keys stored in volatile memory. The EFP shutdown thereby causes the jForté/HAT to enter the Mute Mode error state until power is removed.

These thresholds have been chosen to be outside the normal voltage range of 5.0 ±0.5 V. It is necessary for the under voltage threshold to be high enough so that the 3.3 V internal power bus cannot be directly monitored through the supply voltage. Therefore:

- An over-voltage EFP shutdown will be triggered between 5.5 V and 8.0 V
- An under-voltage EFP shutdown will be triggered between 4.0 V and 4.5 V

## 6.2 Operator Approved Mode

The jForté/HAT supports only a FIPS 140-2 compliant mode of operation. To determine the version information of the module, the operator should execute the GET DATA command to obtain the CPLC data. The operating system release level within the CPLC data is 0301.

## 6.3 Inspection and Testing of Physical Security Mechanism

Only destructive tampering or “most invasive” tampering can breach the physical security mechanisms of the jForté/HAT. Since most invasive tampering would have the probable result of rendering the jForté/HAT inoperable, no regular regimen of visual or instrumented inspection is required for the jForté/HAT. The cryptographic module comprises a single integrated circuit chip. Physical or electrical probing would require extraction of the chip from its jForté/HAT body (card or fob) and destructive removal of one or more manufacturing layers. This would have the almost-certain affect of rendering the chip inoperable. In the unlikely event that the chip was invasively probed without disabling the chip, reassembly of the jForté/HAT without evidence to the casual observer would be extremely difficult.

**Note:** This document, therefore, does not include a table of "Inspection & Testing of Physical Security Mechanisms," as recommended in Appendix-C, Section-C.4 of the FIPS 140-2 document.

## 7. Self-Test Specifications

The jForté/HAT cryptographic module uses power-up and conditional self-tests to ensure that the module is functioning properly.

Upon power-up, the full set of built-in power-up self-tests (see Subsection 7.1 on page 25) are conducted. The power-up sequence concludes with termination of the internally generated reset. The response to power-on (which is, in effect, a command) is issuance of the Answer-to-Reset (ATR). The results of the power-on self-tests can subsequently be determined by issuing a GetjForteStatus command.

Power-on is the only method provided for initiating the self-tests:

| These self-tests . . . | Are performed when . . . |
|---|---|
| Power-up | The cryptographic module is powered up. |
| Conditional | An applicable security function or operation is invoked (i.e., security functions for which self-tests are required). |

| If the jForté/HAT cryptographic module fails . . . | The module will . . . |
|---|---|
| Any power-up self-test except:<br>• Patch update firmware integrity test<br>• DRNG K-A-T<br>• Initial DRNG test | 1. Enter an error state called Abort Mode.<br>2. Output an error indicator (ABORT MODE) via the status output (command) interface.<br>3. Only allow the following administrative service commands to be operational while the module is in the Abort Mode state:<br>• GetjForteStatus<br>• EchoComplement |
| The following power-up self-tests:<br>• Patch update firmware integrity test<br>• DRNG K-A-T<br>• Initial DRNG test | 1. Enter an error state called Mute Mode.<br>2. Inhibit communications with the module while in Mute Mode state.<br>3. Not provide any status indication of the error. |
| Any of the following conditional self-tests:<br>• Continuous DRNG test<br>• Continuous hardware randomizer test<br>• RSA key generation (pair-wise consistency) test | 1. Repeat the failed operation one time.<br>2. Enter an error state called Mute Mode if the operation fails a second time.<br>3. Inhibit communications with the module while in Mute Mode state.<br>4. Not provide any status indication of the error. |

(continued)

| If the jForté/HAT cryptographic module fails . . . | The module will . . . |
|---|---|
| Either of the following conditional self-tests:<br>• Applet load test<br>• Firmware load test | 1. Terminate the load operation that triggered the test.<br>2. Output an error indicator via the status output (command) interface.<br>In this case, the module does not enter the ABORT MODE state, and the operation can be repeated. |

**Note:** Due to the command-response implementation of the jForté/HAT cryptographic module, a GetjForteStatus command must be asserted over the command interface to ascertain the Abort Mode status. The jForté/HAT cryptographic module does not perform any cryptographic operations while in Abort Mode or Mute Mode state. Only a "power-cycle" (the removal and subsequent reapplication of power) applied to the jForté/HAT causes the module to exit from the Abort Mode or Mute Mode state.

Table 13 lists all of the power-up and conditional self-tests performed by the jForté/HAT cryptographic module.

**Table 13 - Power-up and Conditional Self-tests**

| | |
|---|---|
| **Power-up self-tests**<br>(See Subsection 7.1 on page 25) | • Firmware integrity test<br>• Randomizer test<br>• Initial DRNG test<br>• SRAM test<br>• The following cryptographic algorithm known-answer tests:<br>  • DES CBC MAC sign test<br>  • DES CBC mode encryption/decryption test<br>  • Deterministic random number generation test<br>  • RSA sign/verify test<br>  • SHA-1 HASH test<br>  • SKIPJACK ECB64 test<br>  • Triple DES CBC MAC sign test<br>  • Triple DES CBC mode encryption/decryption test |
| **Conditional self-tests**<br>(See Subsection 7.2 on page 27) | • Applet load test<br>• Continuous DRNG test<br>• Continuous hardware randomizer test<br>• Firmware load test<br>• RSA key generation (pair-wise consistency) test |

## 7.1 Power-Up Self-Tests

The jForté/HAT cryptographic module performs the power-up self-tests listed in Table 13 whenever the module is powered up after being either:

- Powered off
- Reset
- Rebooted

The power-up self-tests are initiated automatically, and do not require operator intervention.

The jForté/HAT power-up self tests include a known-answer cryptographic algorithm test for each cryptographic function, such as encryption, decryption, authentication, and deterministic random number generation, of each FIPS-approved cryptographic algorithm.

A known-answer test:

- Applies the cryptographic algorithm to data for which the correct output is already known
- Compares the calculated output with the stored previously generated output (the known answer)

The known-answer test fails if the calculated output does not equal the known answer.

**Note:** All data output via the data output interface is inhibited while the power-up self-tests are being performed.

The jForté/HAT cryptographic module also permits operators to initiate the power-up self-tests on demand for periodic testing of the module. Resetting, rebooting, and power cycling are described by FIPS 140-2 as acceptable means for the on-demand initiation of power-up self-tests.

The following subsections briefly describe each power-up self-test.

### 7.1.1 Firmware Integrity Test

When the jForté/HAT cryptographic module is powered up, an integrity self-test is applied to all of its validated firmware components using a CRC-16 algorithm. The firmware in nonvolatile memory (NVM) that is tested includes:

- Runtime firmware patch code updates
- Applet components (if any)

The firmware integrity test fails if the calculated result does not equal the previously generated checksum.

**Note:** The jForté/HAT performs the CRC-16 check over each section of NVM that contains firmware.

### 7.1.2 Randomizer Test

The randomizer test checks the hardware random engine for proper operation by inspecting two 32-bit output blocks during power-up self-test and confirming that the randomizer output blocks are different.

The hardware random engine provides a seed value to the deterministic random number generator.

### 7.1.3 Initial DRNG Test

The Initial DRNG test checks the DRNG mechanism for proper operation by inspecting the first two consecutive 160-bit output blocks during power-up self-test and confirming that the DRNG output blocks are different.

### 7.1.4 SRAM Test

The SRAM test performs a nondestructive bit test over the entire SRAM address range for both SRAM-1 and SRAM-2.

### 7.1.5 DES CBC MAC Sign Test

Performs a DES CBC MAC sign operation on two blocks of plaintext data with a nonzero Initialization Vector and compares the resulting eight-byte signature with a known signature stored in ROM.

### 7.1.6 DES CBC Mode Encryption/Decryption Test

Performs a DES CBC encryption on one block of plaintext data with a nonzero Initialization Vector and compares the resulting ciphertext to known ciphertext. The test then performs a DES CBC decryption on the resulting ciphertext and compares the result to the original plaintext data.

A failure occurs if at any point these compares do not match.

### 7.1.7 Deterministic Random Number Generation Known-Answer Test

The deterministic random number generation test is a known-answer test, using 160-bit blocks, that is executed on power-up to confirm the correctness of the DRNG algorithm.

### 7.1.8 RSA Sign/Verify Known-Answer Test

The RSA sign/verify known-answer test uses an RSA key pair that was precomputed and included in the jForté/HAT ROM image. It uses this key pair to make sure that the RSA public and private key translation code is working correctly by comparing each output against known answers.

### 7.1.9 SHA-1 HASH Test

Performs a SHA-1 hash operation on the text “abc” and compares the resulting hash digest to the known-answer hash digest stored in ROM.

#### 7.1.10 SKIPJACK Test

Performs a SKIPJACK ECB64 encrypt and decrypt operation on one block of plaintext data with comparisons against known-answer ciphertext after the encrypt and the original plaintext after decrypt. Note that the SKIPJACK mechanism is only used for decrypting firmware downloads and cannot be accessed from the applet interface or by any external operators of the module.

#### 7.1.11 Triple DES CBC MAC Test

Performs a Triple DES CBC MAC operation on two blocks of plaintext data with a nonzero Initialization Vector and compares the resulting eight-byte MAC with a known MAC stored in ROM.

#### 7.1.12 Triple DES CBC Mode Encryption/Decryption Test

Performs a Triple DES CBC encryption operation on one block of plaintext data with a nonzero Initialization Vector and compares the resulting ciphertext to known ciphertext. The test then performs a Triple DES CBC decryption on the resulting ciphertext and compares the result to the original plaintext data.

A failure occurs if at any point these compares do not match.

## 7.2 Conditional Self-Tests

The jForté/HAT cryptographic module performs the conditional tests listed in Table 13 on page 24 whenever the conditions specified for the tests occur:

- Applet load
- Continuous deterministic random number generation
- Continuous hardware randomizer
- Firmware load
- RSA key generation

The following subsections briefly describe each conditional self-test.

### 7.2.1 Applet Load Test

The jForté/HAT performs the following two-step applet load test whenever applets are externally loaded:

| Step | Action |
|---|---|
| 1 | Determines the authentication using TDES MAC for the candidate applet component being loaded. |
| 2 | Compares the calculated result with a previously generated result. |

The test fails and the jForté/HAT outputs an error indicator if the calculated result does not equal the previously generated result.

### 7.2.2 Continuous Deterministic Random Number Generator Test

The jForté/HAT cryptographic module employs a deterministic random number generator (DRNG) that functions in an approved mode of operation (FIPS 186-2, Appendix 3.1). The module performs the following two-step continuous random number generator test on each DRNG block to test for failure to a constant value:

| Step | Action |
|---|---|
| 1 | Saves the first *n*-bit block generated after power-up, initialization, or reset.<br>Since each call to a DRNG produces blocks of *n* bits, where $n > 15$, the first *n*-bit block generated after power-up, initialization, or reset is not immediately used, but saved for comparison with the next generated *n*-bit block. |
| 2 | Compares each subsequently generated *n*-bit block with the previously generated block. |

The test fails and the jForté/HAT enters the Mute Mode state if any two compared *n*-bit blocks are equal.

**Note:** For the jForté/HAT, $n = 160$ bits.

### 7.2.3 Continuous Hardware Randomizer Test

Since the jForté/HAT cryptographic module employs a hardware random number generator (RNG) that provides a seed to the DRNG algorithm, the module performs the following two-step continuous random number generator test on each RNG block to test for failure to a constant value:

| Step | Action |
|---|---|
| 1 | Saves the first 32-bit block generated after power-up, initialization, or reset.<br>The first 32-bit block generated after power-up, initialization, or reset is not immediately used, but saved for comparison with the next generated 32-bit block. |
| 2 | Compares each subsequently generated 32-bit block with the previously generated block. |

The test fails and the jForté/HAT enters the Mute Mode state if any two compared 32-bit blocks are equal.

### 7.2.4 Firmware Load Test

The jForté/HAT performs the following two-step firmware load test whenever firmware is externally loaded:

| Step | Action |
|---|---|
| 1 | Determines the authentication by verifying the 1024-bit RSA signature on the candidate runtime engine firmware update being loaded. |
| 2 | Compares the calculated result with a previously generated result. |

The test fails and the jForté/HAT outputs an error indicator if the calculated result does not equal the previously generated result.

### 7.2.5 RSA Key Generation (Pair-wise Consistency) Test

Two RSA pair-wise consistency checks are run during RSA key pair generation to validate the consistency of an RSA key pair candidate—which has just been generated—before it is accepted.

One pair-wise test validates signature/verify mode by signing test data with the private-key and then verifying the signature with public key. The other pair-wise test validates key transport mode by encrypting the test data with a public-key and then decrypting the resulting ciphertext with the private key. A more detailed description of this self-test is provided in a separate document.

## 8. Mitigation of Other Attacks Policy

Cryptographic modules are susceptible to other attacks for which testable security requirements were not available when issuing this version of the standard (e.g., power analysis, timing analysis, and/or fault induction) or the attacks were outside of the scope of the standard (e.g., TEMPEST). Susceptibility of a cryptographic module to such attacks depends on module type, implementation, and implementation environment. Such attacks may be of particular concern for cryptographic modules implemented in hostile environments (e.g., where the attackers may be the authorized operators of the module). Such types of attacks generally rely on the analysis of information obtained from sources physically external to the module. In all cases, the attacks attempt to determine some knowledge about the cryptographic keys and CSPs within the cryptographic module.

**Note:** Any event that triggers an environmental failure protection (EFP) shutdown causes the jForté/HAT to enter the Mute Mode error state. This error state can only be cleared by the power-on reset (POR) that occurs as a result of initial power application.

**Table 14 - Mitigation of Other Attacks**

| Other Attacks | Mitigation Mechanism | Specific Limitations |
|---|---|---|
| **Power Analysis**<br>Attacks based on the analysis of power consumption divides into two general categories:<br>• Simple Power Analysis (SPA)<br>• Differential Power Analysis (DPA)<br>SPA involves a direct (primarily visual) analysis of electrical power consumption patterns and timings derived from the execution of individual instructions carried out by a cryptographic module during a cryptographic process.<br>The patterns are obtained through monitoring the variations in electrical power consumption of a cryptographic module for the purpose of revealing the features and implementations of cryptographic algorithms and subsequently values of cryptographic keys.<br>DPA has the same goals, but utilizes advanced statistical methods and/or other techniques to analyze the variations of the electrical power consumption of a cryptographic module. Cryptographic modules that use external power (direct current) sources appear to be at greatest risk.<br>Methods that may reduce the overall risk of Power Analysis attacks include the use of capacitors to level the power consumption, the use of internal power sources, and the manipulation of the individual operations of the algorithms or processes to level the rate of power consumption during cryptographic processing. | The AT91SC321RC contains circuitry to thwart DPA and SPA attacks.<br>The internal power supply regulates the input from 5 to 3.3 volts and holds the internal input voltage (for the operation of the chip) steady even while executing a cryptographic operation. | N/A |

**Table 14 - Mitigation of Other Attacks (continued)**

| Other Attacks | Mitigation Mechanism | Specific Limitations |
|---|---|---|
| **Timing Analysis**<br>Timing Analysis attacks rely on precisely measuring the time required by a cryptographic module to perform specific mathematical operations associated with a cryptographic algorithm or process.<br>The timing information collected is analyzed to determine the relationship between the inputs to the module and the cryptographic keys used by the underlying algorithms or processes. The analysis of the relationship may be used to exploit the timing measurements to reveal the cryptographic keys or CSPs.<br>Timing Analysis attacks assume that the attacker has knowledge of the design of the cryptographic module. Manipulation of the individual operations of the algorithms or processes to reduce timing fluctuations during processing is one method to reduce the risk of this attack. | The AT91SC321RC contains independent oscillating clock domains for the various components within the device. Thus, the speed of one device is completely out of synch and phase with another.<br>The chip also implements data and address signal “scrambling” when accessing information off the primary system bus. | N/A |
| **Fault Induction**<br>Fault Induction attacks use external forces such as clock frequency extremes to cause processing errors within the cryptographic module.<br>An analysis of these errors and their patterns are an attempt to reverse engineer the cryptographic module, revealing certain features and implementations of cryptographic algorithms and subsequently revealing the values of cryptographic keys.<br>Cryptographic modules with limited physical security appear to be at greatest risk. Proper selection of physical security features may be used to reduce these risks. | The power-on-reset recognition circuit detects and halts upon invalid startup sequences. | N/A |
| **Frequency Attack**<br>The Frequency Attack is a type of fault induction attack that utilizes clock frequency manipulation to cause processing errors within the cryptographic module.<br>An analysis of these errors and their patterns is an attempt to reverse engineer the cryptographic module, revealing certain features and implementations of cryptographic algorithms and subsequently revealing the values of cryptographic keys. | The jForté/HAT mitigates the threat of invasive attacks based on input frequency by providing a circuit (called the pulse piranha) that eliminates any clock edge not separated from the previous edge by at least 20 nanoseconds, thereby establishing a maximum frequency of 25 MHz.<br>In addition, an EFP shutdown is issued if the clock frequency becomes less than 400 KHz. | N/A |

## 9. Abbreviations and Acronyms

**Table 15 - List of Abbreviations and Acronyms**

| Acronyms | Definitions |
|---|---|
| ASIC | Application-Specific Integrated Circuit |
| CBC | Cipher Block Chaining (block encryption mode) |
| Clk | Clock |
| CO | Cryptographic Officer |
| CSP | Critical Security Parameter (FIPS 140) |
| DES | Data Encryption Standard |
| DRNG | Deterministic Random Number Generator |
| ECB | Electronic Code Book (block encryption mode) |
| EEPROM | Electrically Erasable Programmable Read Only Memory |
| EFP | Environmental Failure Protection |
| FIPS | Federal Information Processing Standard |
| HAT | High Assurance Token |
| ISO | International Organization for Standardization |
| jForté | Java (enabled) Forté |
| MAC | Message Authentication Code |
| MSM | (token) Manufacturing Security Model |
| PIN | Personal Identification Number |
| RNG | Random Number Generator (hardware) |
| ROM | Read Only Memory |
| RSA | Rivest Shamir Adleman (Public Key Algorithm) |
| SHA-1 | Secure Hash Algorithm |
| USB | Universal Serial Bus |

# Appendix A. Supported Cryptographic Algorithms

The jForté/HAT device provides the following services to the applet developers. These services are not accessible externally and are only available to applet developers. The applets (which will go through a separate FIPS140-2 validation) can offer these services externally to the card holders. Table 16 lists the security services available by the jForté/HAT.

**Note:** This information is provided strictly for future applet developers, and is not relevant to validation of the jForté/HAT device itself.

**Note also:** DES algorithms (DES and DES MAC) are supported for transitional phase only – valid until May 19, 2007.

**Table 16 - Supported Cryptographic Algorithms**

| Algorithm | FIPS Approved | Encrypt/Decrypt | Digest | Sign/Verify | Generate Key | Generate Key Pair | Wrap/Unwrap |
|---|---|---|---|---|---|---|---|
| RSA-1024 as specified in "Public-Key Cryptographic Standard PKCS #1," Version 2.1, for 1024-bit modulus prime key lengths. | √ | | | √ | | 1024 | |
| Data Encryption Standard (DES) ECB mode, Single key generation as specified in FIPS Pub 46-3 "Data Encryption Standards," October 25, 1999. For legacy purposes only. | √ | √ | | | 56 | | √ |
| Data Encryption Standard (DES) CBC mode, Single key generation as specified in FIPS Pub 46-3 "Data Encryption Standards," October 25, 1999. For legacy purposes only. | √ | √ | | | 56 | | √ |
| Data Encryption Standard (DES) MAC mode, as specified in FIPS Pub 113 "Computer Data Authentication," May 30, 1985. For legacy purposes only. | √ | | | √ | 56 | | |
| Deterministic Random Number Generator (DRNG) as specified in FIPS Pub 186-2 "Digital Signature Standard (DSS)," Appendix 3.1 - Algorithm for Computing m values of x (using DEA), January 27, 2000. | √ | | | | | | |
| Triple Data Encryption Standard (TDES) ECB mode, Double key generation as specified in FIPS Pub 46-3 "Data Encryption Standards," October 25, 1999. | √ | √ | | | 112 | | √ |
| Triple Data Encryption Standard (TDES) CBC mode, Double key generation as specified in FIPS Pub 46-3 "Data Encryption Standards," October 25, 1999. | √ | √ | | | 112 | | √ |

(continued)

**Table 16 - Supported Cryptographic Algorithms (continued)**

| Algorithm | FIPS Approved | Encrypt/Decrypt | Digest | Sign/Verify | Generate Key | Generate Key Pair | Wrap/Unwrap |
|---|---|---|---|---|---|---|---|
| Triple Data Encryption Standard (TDES) ECB mode, Triple key generation as specified in FIPS Pub 46-3 "Data Encryption Standards," October 25, 1999. | √ | √ | | | 168 | | √ |
| Triple Data Encryption Standard (TDES) CBC mode, Triple key generation as specified in FIPS Pub 46-3 "Data Encryption Standards," October 25, 1999. | √ | √ | | | 168 | | √ |
| Triple Data Encryption Standard (TDES) MAC mode, Double key as specified in FIPS Pub 81 "DES Modes of Operation," 1980 December 2 | √ | | | √ | 112 | | |
| Triple Data Encryption Standard (TDES) MAC mode, Triple key as specified in FIPS Pub 81 "DES Modes of Operation," 1980 December 2 | √ | | | √ | 168 | | |
| Secure Hash Algorithm (SHA-1) as specified in FIPS Pub 180-2 "Secure Hash Standard," August 1, 2002. | √ | | √ | | | | |