Enabling a Public Key Infrastructure in SmartFusion2 Devices

UG0626 User Guide
Contents

1 Preface ........................................................................................................... 7
  1.1 Purpose ....................................................................................................... 7
  1.2 Intended Audience ................................................................................... 7
  1.3 References ................................................................................................. 7
    1.3.1 Microsemi Publications .................................................................. 7

2 Enabling a Public Key Infrastructure in SmartFusion2 Devices ................. 8
  2.1 Introduction ............................................................................................... 8
    2.1.1 PKI Objectives .............................................................................. 9
  2.2 Design Requirements ................................................................................ 10
  2.3 Design Files .............................................................................................. 10
  2.4 Design Description .................................................................................. 10
    2.4.1 Enrolling a SmartFusion2 SoC FPGA Device into PKI .................. 10
    2.4.2 LRA ............................................................................................... 11
    2.4.3 ESCRYPT CurvKEYS as CA ....................................................... 12
    2.4.4 PKI Enrollment Process ............................................................... 12
      2.4.4.1 SmartFusion2 Device Authentication ..................................... 12
      2.4.4.2 Proof-of-possession of Private Keys ...................................... 13
      2.4.4.3 Acquiring User Public Key Certificate and Signature Verification 14
      2.4.4.4 Certificate Request Message ............................................... 15
      2.4.4.5 Digital Signature and Signature Verification .............................. 15
    2.4.5 M2M Communication using Digital Certificates .............................. 15
      2.4.5.1 TLS/SSL Protocol ................................................................. 16
  2.5 Hardware Implementation ......................................................................... 18
    2.5.0.1 Package Pin Assignments ....................................................... 19
    2.5.0.2 User Clients in eNVM ................................................................ 20
  2.6 Software Implementation ........................................................................... 20
  2.7 Setting Up the Demo for PKI Enrollment .................................................. 22
    2.7.1 Installing PKI-Authenticated M2M Communications Demo Utility 22
    2.7.2 Installing User Root CA Public Key Certificate into Windows Certificate Store 22
    2.7.3 Board Setup for PKI Enrollment ..................................................... 24
    2.7.4 Programming the SmartFusion2 Device ........................................ 31
  2.8 Running the Demo for Enrolling SmartFusion2 SoC FPGA Device in PKI 32
    2.8.1 Detection of Certificate Requests from Non-authentic Machines .... 47
  2.9 Setting Up the Demo for M2M Communication using the Digital Certificates 48
    2.9.1 Enrolling the two SmartFusion2 SoC FPGA Devices in PKI ............ 48
    2.9.2 Setting up the Boards ...................................................................... 48
  2.10 Running the Demo for M2M Communication using the Digital Certificates 51
    2.10.1 Secure M2M Communication using Client-server Model .............. 55
      2.10.1.1 Transferring the Data Securely to the Server ....................... 55
      2.10.1.2 Sending a Command Securely to the Server to blink LEDs ..... 60
      2.10.1.3 Sending a Command Securely to the Server to get DIP Switches Status 61
    2.10.2 Conclusion ....................................................................................... 61

Revision 2 2
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Block Diagram of Public Key Infrastructure</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>LRA Host PC Application</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>Device Authentication and Proof-of-Possession of Private Keys</td>
<td>14</td>
</tr>
<tr>
<td>4</td>
<td>Acquiring User Public Key Certificate for SmartFusion2 Device</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>Secure Communication between PKI Enrolled SmartFusion2 Devices</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>TLS Handshake Procedure</td>
<td>17</td>
</tr>
<tr>
<td>7</td>
<td>Libero SoC Hardware Design</td>
<td>18</td>
</tr>
<tr>
<td>8</td>
<td>High-Speed Serial Interface 2 Configurator</td>
<td>19</td>
</tr>
<tr>
<td>9</td>
<td>eNVM Configurator - User Clients in eNVM</td>
<td>20</td>
</tr>
<tr>
<td>10</td>
<td>SoftConsole Project Explorer Window</td>
<td>21</td>
</tr>
<tr>
<td>11</td>
<td>Windows Security Warning</td>
<td>22</td>
</tr>
<tr>
<td>12</td>
<td>Certificate Window</td>
<td>23</td>
</tr>
<tr>
<td>13</td>
<td>Certificate Store Selection</td>
<td>23</td>
</tr>
<tr>
<td>14</td>
<td>Windows Security Warning to Install the Certificate as Root Certificate</td>
<td>24</td>
</tr>
<tr>
<td>15</td>
<td>Windows Device Manager - COM Port Detection</td>
<td>25</td>
</tr>
<tr>
<td>16</td>
<td>Windows Firewall with Advanced Security</td>
<td>26</td>
</tr>
<tr>
<td>17</td>
<td>Active Network Profile and Windows Firewall State</td>
<td>27</td>
</tr>
<tr>
<td>18</td>
<td>Active Profile with Windows Firewall ON</td>
<td>28</td>
</tr>
<tr>
<td>19</td>
<td>Launch Windows Firewall Properties</td>
<td>29</td>
</tr>
<tr>
<td>20</td>
<td>Active Profile Tab Selection</td>
<td>29</td>
</tr>
<tr>
<td>21</td>
<td>Allow Inbound Connections through Windows Firewall - 1</td>
<td>30</td>
</tr>
<tr>
<td>22</td>
<td>Allow Inbound Connections through Windows Firewall - 2</td>
<td>30</td>
</tr>
<tr>
<td>23</td>
<td>Current State of Inbound Connections in the Active Profile</td>
<td>31</td>
</tr>
<tr>
<td>24</td>
<td>Launching Host PC PKI-Authenticated M2M Communications Demo Utility</td>
<td>32</td>
</tr>
<tr>
<td>25</td>
<td>SmartFusion2 User PKI - Local Registration Authority Utility</td>
<td>33</td>
</tr>
<tr>
<td>26</td>
<td>Selecting COM Port Number for SmartFusion2 Device UART Interface</td>
<td>34</td>
</tr>
<tr>
<td>27</td>
<td>SmartFusion2 Device Connecting to Host PC</td>
<td>35</td>
</tr>
<tr>
<td>28</td>
<td>SmartFusion2 Device Serial Number</td>
<td>36</td>
</tr>
<tr>
<td>29</td>
<td>SmartFusion2 Device Authentication</td>
<td>37</td>
</tr>
<tr>
<td>30</td>
<td>SmartFusion2 Device Certificate Signature Check at LRA</td>
<td>38</td>
</tr>
<tr>
<td>31</td>
<td>SmartFusion2 Device Certificate</td>
<td>39</td>
</tr>
<tr>
<td>32</td>
<td>Log Messages</td>
<td>40</td>
</tr>
<tr>
<td>33</td>
<td>Proof-of-Possession of Private Keys</td>
<td>41</td>
</tr>
<tr>
<td>34</td>
<td>Approved User Public Key Certificate Request Sent to CA</td>
<td>42</td>
</tr>
<tr>
<td>35</td>
<td>CA Issued User Public Key Certificate to Device</td>
<td>43</td>
</tr>
<tr>
<td>36</td>
<td>User Public Key Certificate General Information</td>
<td>44</td>
</tr>
<tr>
<td>37</td>
<td>User Public Key Certificate Details</td>
<td>45</td>
</tr>
<tr>
<td>38</td>
<td>Certification Path</td>
<td>46</td>
</tr>
<tr>
<td>39</td>
<td>Log Information</td>
<td>47</td>
</tr>
<tr>
<td>40</td>
<td>Windows Device Manager - COM Port Detection</td>
<td>48</td>
</tr>
<tr>
<td>41</td>
<td>COM Ports of Two SmartFusion2 Devices</td>
<td>49</td>
</tr>
<tr>
<td>42</td>
<td>Router Demo Setup</td>
<td>50</td>
</tr>
<tr>
<td>43</td>
<td>M2M Demo Setup</td>
<td>50</td>
</tr>
<tr>
<td>44</td>
<td>Launching Host PC PKI-Authenticated M2M Communications Demo Utility</td>
<td>51</td>
</tr>
<tr>
<td>45</td>
<td>Selecting COM Ports for the Machines</td>
<td>52</td>
</tr>
<tr>
<td>46</td>
<td>Selecting SmartFusion2 Machine as Server</td>
<td>53</td>
</tr>
<tr>
<td>47</td>
<td>Selecting SmartFusion2 Machine as Client</td>
<td>54</td>
</tr>
<tr>
<td>48</td>
<td>Log Messages</td>
<td>55</td>
</tr>
<tr>
<td>49</td>
<td>TLS Handshaking</td>
<td>56</td>
</tr>
<tr>
<td>50</td>
<td>TLS Protocol and Cipher Suite</td>
<td>57</td>
</tr>
<tr>
<td>51</td>
<td>Transferring Data Securely to Server - 1</td>
<td>58</td>
</tr>
<tr>
<td>52</td>
<td>Transferring Data Securely to Server - 2</td>
<td>59</td>
</tr>
</tbody>
</table>
Figure 53. Server Machine’s LED Blinking ......................................................... 60
Figure 54. DIP Switches Status ................................................................. 61
Figure 55. SmartFusion2 Security Evaluation Board ................................. 62
Figure 56. SC_SPI Signals Location ............................................................ 63
Figure 57. SC_SPI Signal Names ............................................................... 63
Figure 58. Location of the Resistors (R105, R111, R112 and R113) to be De-populated ................................. 64
Figure 59. Example Digital Certificate ....................................................... 65
Figure 60. PKI enrollment Setup using Wi-Fi Repeater and Smartphone ......................... 66
Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 1</td>
<td>Design Requirements</td>
<td>10</td>
</tr>
<tr>
<td>Table 2</td>
<td>Pins Assignments</td>
<td>19</td>
</tr>
</tbody>
</table>
Preface

1.1 Purpose

This user guide is for enabling a public key infrastructure in the SmartFusion®2 SoC FPGA devices and also describes secure machine-to-machine (M2M) communication using PKI-enrolled SmartFusion2 SoC FPGA devices. It provides instructions about how to use the corresponding reference design.

This document contains the following sections:

- Enabling a Public Key Infrastructure in SmartFusion2 Devices

1.2 Intended Audience

The following designers use the SmartFusion2 SoC FPGA devices:

- FPGA designers
- Embedded designers
- System-level designers

1.3 References

The following documents are referred in this user guide.

1.3.1 Microsemi Publications

UG0331: SmartFusion2 Microcontroller Subsystem User Guide
UG0450: SmartFusion2 SoC and IGLOO2 FPGA System Controller User Guide
UG0443: SmartFusion2 and IGLOO2 FPGA Security and Reliability User Guide

The following web-page provides a complete and up-to-date listing of SmartFusion2 SoC FPGA device documentation:


Refer to the following web-page for more information about secure connectivity:

2 Enabling a Public Key Infrastructure in SmartFusion2 Devices

2.1 Introduction

In the public network (Internet), digital certificates or public key certificates are often used to secure communication links. A public key certificate is an electronic document that contains both an identity and a public key, binding them together by a digital signature. Public key certificates are used to identify people, organizations, machines, and software applications over public networks. They also enable secure, confidential communication between two parties using encryption. The public key certificates are an important part of transport layer security (TLS), where they prevent an attacker from impersonating as a trusted client or server, commonly called a man in the middle attack. They are also used in other important applications, such as e-mail encryption and code signing.

Public key infrastructure (PKI) is an arrangement that binds public keys with respective user or device identities by means of a certificate authority (CA). CA issues a public key certificate after verifying the identity of the requester at local registration authority (LRA). The LRA verifies the identity of users or devices that request the public key certificate from the CA. Anybody in possession of the CA's public key can verify the signature on the public key certificate. In this way, the CA guarantees that the public key present in the certificate belongs to the user or device whose identity is present in the same certificate. The most common certificate format is defined by the X.509 standard.

Figure 1 • Block Diagram of Public Key Infrastructure

In a PKI, the process of submitting a certificate request to the CA or LRA is known as enrollment. After enrollment, the CA issues a public key certificate to the enrolled public key. During the enrollment, the
device that submits the public key is required to prove that it knows the associated private key and that it controls the use of this private key. This is commonly referred as proof-of-possession.

Elliptic curve cryptography (ECC) is emerging as an attractive public-key cryptosystem, in particular for mobile (that is, wireless) environments. Compared to currently prevalent cryptosystems such as RSA, ECC offers equivalent security with smaller key sizes. Smaller key sizes result in savings for power, memory, bandwidth, and computational cost that makes ECC especially attractive for constrained environments. The digital certificates based on the ECC cryptographic algorithms improve the performance and security strength over RSA based digital certificates. The premium S grade SmartFusion2 devices (M2S060S, M2S090S, and M2S150S) have a built-in ECC hardware accelerator (that is, NIST-defined P-384 curve) to support public-key cryptographic techniques for key establishment. SmartFusion2 SoC FPGA devices also contain the Quiddikey™, an static random-access memory (SRAM or static RAM) physically unclonable function (SRAM-PUF) licensed from Intrinsic-ID for secure key storage and a true random number generator.

This reference design demonstrates the ability of the SmartFusion2 SoC FPGA device to self-enroll in PKI and obtain a digital certificate to securely exchange messages with another device in the PKI.

### 2.1.1 PKI Objectives

PKI supports important requirements of Internet security. PKI provides the solution to security problems in the form of:

- **Data Authentication**
  Is achieved by providing an environment where users or devices can verify the identify of each other.

- **Data Confidentiality**
  Is achieved by not disclosing the transmitted information to any unauthorized communicating parties.

- **Data Integrity**
  Is achieved when the transmitted information is not tampered in any way.

- **Non-repudiation**
  Is achieved by ensuring that the sender cannot refuse that they signed or encrypted a particular message once it has been sent.

The security features of the SmartFusion2 devices make it an ideal platform to use them in PKI enabled applications.

The following SmartFusion2 built-in capabilities fulfill and accomplish the PKI objectives:

- ECC to generate user key pairs.
- SRAM physically unclonable function (SRAM-PUF) for enrolling and retrieving the ECC secret keys.
- Non-deterministic random bit generator (NRBG) for preparing the secret keys and Nonces.
- User Cryptographic services such as AES-128, AES-256, SHA-256, Key-Tree, and keyed-hash message authentication code (HMAC).
- Large on-chip non-volatile memory for storing the digitally signed user public key certificates.


2.2 Design Requirements

Table 1 shows the design requirements.

<table>
<thead>
<tr>
<th>Design Requirements</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Hardware Requirements</strong></td>
<td></td>
</tr>
<tr>
<td>Two reworked SmartFusion2 Security Evaluation Kits: M2S090TS-EVAL-KIT-REWORKED&lt;sup&gt;1&lt;/sup&gt;</td>
<td>Rev D or later</td>
</tr>
<tr>
<td>1. 12 V Power adapter</td>
<td></td>
</tr>
<tr>
<td>2. FlashPro4 JTAG programmer</td>
<td></td>
</tr>
<tr>
<td>3. Two USB A to mini-B cables</td>
<td></td>
</tr>
<tr>
<td>Host PC or Laptop</td>
<td></td>
</tr>
<tr>
<td>Ethernet Router</td>
<td>--</td>
</tr>
<tr>
<td>Three Ethernet Cables</td>
<td>--</td>
</tr>
<tr>
<td>Operating System</td>
<td>64-bit Windows 7 or Windows 10</td>
</tr>
<tr>
<td><strong>Software Requirements</strong></td>
<td></td>
</tr>
<tr>
<td>Libero® System-on-Chip (SoC)</td>
<td>v11.7 or later</td>
</tr>
<tr>
<td>PKI-Authenticated M2M Communications Demo Utility</td>
<td>Host PC Software</td>
</tr>
<tr>
<td>Host PC Drivers</td>
<td>USB to UART drivers FT232R</td>
</tr>
<tr>
<td>Framework to run demonstration</td>
<td>Microsoft .NET framework 4.5.1 or later</td>
</tr>
<tr>
<td>Java Runtime Environment</td>
<td>Java SE Runtime Environment 8u25 or later</td>
</tr>
</tbody>
</table>

1. SmartFusion2 Security Evaluation Kit (M2S090TS-EVAL-KIT) is reworked to connect the SmartFusion2 system controller SPI to MSS SPI1 (MSS_SPI1) for running key verification protocol. With this rework, ARM® Cortex®-M3 processor can access the system controller SPI interface. Refer to "Appendix: SmartFusion2 Security Evaluation Kit" section on page 62 for more information on the rework.

2.3 Design Files

To get the reference design files send an e-mail request to pkidemo@microsemi.com.

The design files include:

- SmartFusion2 PKI Demo Programming File
- Libero SoC project
- User Root CA Public Key Certificate
- PKI-Authenticated M2M Communications Demo Utility
- Readme file

Refer to the readme.txt file provided in the design files for the complete directory structure.

2.4 Design Description

This reference design leverages the built-in security capabilities of the SmartFusion2 devices to demonstrate the PKI enrollment process. The design also enables the SmartFusion2 devices to securely communicate with each other using the digital certificates and TLS protocol.

2.4.1 Enrolling a SmartFusion2 SoC FPGA Device into PKI

The PKI enrollment process uses a CA to issue a digital certificate to the SmartFusion2 device through the LRA. The SmartFusion2 SoC FPGA device is configured as an HTTP client and the LRA acts as an HTTP server.
The following SmartFusion2 built-in capabilities are used in the PKI enrollment process.

### 2.4.1.1 Elliptic Curve Cryptography

The reference design utilizes ECC feature that is available in the SmartFusion2 device to use public-key methods for device authentication. The reference design uses DPA-resistant hardware ECC engine, which is available through system services to perform key establishment, and to generate or verify digital signatures. The SmartFusion2 system controller’s ECC point multiplication system service, computes the ECC public key using the point P on the NIST P-384 curve and the NRBG based ECC private key. For more information about ECC, refer to the *UG0443: SmartFusion2 and IGLOO2 FPGA Security and Reliability User Guide*.

### 2.4.1.2 Key Verification Protocol

SmartFusion2 devices have a built-in on-line challenge-response type key verification protocol. This can be used to have a device prove that it knows a particular secret key, without exposing the value of the key in the external communications used in the protocol. The key verification protocol binds the X.509 certificates to the device by requiring them to prove the possession of the secret key. The device calculates a shared secret by executing elliptic-curve diffie-hellman (ECDH) protocol and a DPA-safe key derivation function on device private key and LRA public key. The key verification protocol service is not available as an internal system service. It can only be requested either through JTAG interface or system controller SPI interface. This reference design uses system controller SPI interface to execute the key verification protocol. A connection is made between MSS SPI (master) to system controller SPI (slave) for executing key verification protocol.

### 2.4.1.3 SRAM-PUF

The SRAM-PUF is a novel key storage mechanism called Quiddikey™ licensed from Intrinsic-ID, B.V., with superior security attributes. It combines the passive zeroization feature of volatile memory with tamper-resistant non-volatile key storage, without requiring batteries.

The reference design uses the SRAM-PUF feature to enroll the ECC private key. For more information, refer to the *UG0443: SmartFusion2 and IGLOO2 FPGA Security and Reliability User Guide*.

### 2.4.1.4 NRBG

The reference design uses the NRBG to prepare ECC private key, and generate random numbers such as Nonces (numbers used only once). Nonces are used in key verification protocols to prevent replay attacks by ensuring that the response received in the protocol exchange is fresh and not a replay of a similar previous session.

### 2.4.1.5 SHA-256 Secure Hash Engine

The reference design uses the hardware SHA-256 accelerator to compute the shared secret key as part of the key verification protocol.

### 2.4.2 LRA

In this reference design, the LRA functionality is emulated on a Host PC using the PKI-Authenticated M2M Communications Demo Utility. *Figure 2* shows the PKI enrollment service running on the Host PC.
2.4.3 ESCRYP'T CycurKEYS as CA

ESCRYP'T CycurKEYS® is a security server hosted in the cloud that offers the management of cryptographic keys and certificates, especially designed and implemented for automotive, industrial embedded and cyber physical system applications. This reference design uses the ESCRYP'T CycurKEYS CA services for issuing user public key certificates to the SmartFusion2 devices.

2.4.4 PKI Enrollment Process

The PKI enrollment process is divided into the following three tasks:

- SmartFusion2 device authentication
- Proof-of-possession of private keys
- Acquiring user public key certificate and signature verification

2.4.4.1 SmartFusion2 Device Authentication

1. The SmartFusion2 SoC FPGA device fetches the device certificate using system controller services and exports the device certificate to the LRA for device authentication.
2. The LRA verifies the device certificate signature with the trusted Microsemi Public Key. If the digital signature is valid, then the public key that is present in the SmartFusion2 device certificate is used for the device authentication. As part of the device authentication, the LRA challenges the device to prove that it knows the corresponding device secret key.
2.4.4.2 Proof-of-possession of Private Keys

3. The SmartFusion2 SoC FPGA device generates the user ECC key pair (U-PK and U-SK) using built-in ECC hardware accelerator and sends the U-PK to the LRA. The LRA uses the U-PK as part of key verification protocol to prove that the U-PK comes from the SmartFusion2 device, and not from a man-in-middle attack.

4. The LRA generates ephemeral ECC key pair 1 (E-PK1 and E-SK1) and Nonce (LRA-Nonce1).

5. The LRA sends E-PK1 and LRA-Nonce1 to the SmartFusion2 SoC FPGA device. The LRA challenges the SmartFusion2 device to prove that it knows the device secret key (D-SK).

6. The device calculates a shared secret by executing ECDH protocol using its private key (D-SK) and LRA public key (E-PK1). The device generates a root key by executing a key derivation function using the shared secret. The device generates the device nonce1 using NRBG. The device uses the root key, device nonce1, LRA-Nonce1, and U-PK1 for executing a key verification protocol and calculates the validator1.

The key verification protocol is summarized as:
- Validator1 = \( F(\text{Root key}, \text{Device nonce1}, \text{LRA nonce1}, \text{U-PK1}) \)
- Root Key = Key Derivative Function (Shared Secret)
- Shared Secret = ECDH (D-SK, E-PK1)

7. The device sends the validator1 and Device Nonce1 to the LRA.

8. LRA uses E-SK1 and D-PK to establish a shared secret using the ECDH protocol. From this, LRA derives the same root key, which is the actual key used along with the device nonce1, LRA – nonce1, U-PK in the challenge-response portion of the protocol for calculating the Validator1.

9. If the calculated LRA validator1 is proved to match with the received device validator1, this implies that the device must have known the device private key used on its side of the ECDH protocol. As a result, the device not only authenticates with the validator that it knows the device secret key, but also that it knows the valid U-PK before an adversary can modify it in a man-in-the-middle attack on the communicated values.

10. LRA generates another pair of ephemeral ECC key (E-PK2 and E-SK2) and LRA Nonce2.

11. The LRA sends E-PK2 and LRA-Nonce2 to the SmartFusion2 SoC FPGA device. The LRA challenges the SmartFusion2 SoC FPGA device to prove that it knows the user secret key (U-SK).

12. The device executes the key verification protocol and generates validator2 as explained in step 6.
- Validator2 = \( F(\text{Root key}, \text{Device nonce2}, \text{LRA nonce2}) \)
- Root Key = Key Derivative Function (Shared Secret)
- Shared Secret = ECDH (U-SK and E-PK2)

13. The device sends the validator2 and Device Nonce2 to the LRA.

14. LRA uses E-SK2 and U-PK to establish a shared secret using the ECDH protocol. From this, LRA derives Root Key, which is the key actually used along with the Device Nonce2, LRA – nonce2 in the challenge-response portion of the protocol for calculating the Validator2.

15. If the calculated LRA validator2 is proved to match with the received device validator2, this implies that the device must have known the user private key (U-SK). This step completes the proof-of-possession of private keys and it implies that the device authentication is successful.
2.4.4.3 Acquiring User Public Key Certificate and Signature Verification

1. The SmartFusion2 device sends signed certificate request message to the LRA. Refer to "Certificate Request Message" section on page 15 for more information.

2. The LRA verifies the certificate request message signature using the user public key and compares the user public key of the certificate message request that is available with U-PK to prevent the man-in-the-middle attack.

3. The LRA approves and forwards the certificate request to the CA.

4. The CA processes the certificate request and signs the X.509 formatted user public key certificate with root CA secret key.

5. The CA issues the user public key certificate to the LRA.

6. The LRA sends the user public key certificate to the SmartFusion2 device.

7. The SmartFusion2 SoC FPGA device checks the signature of the received user public key certificate using trusted Root CA public key certificate and stores the certificate into eNVM on successful signature verification.
2.4.4.4 Certificate Request Message

The certificate management protocol (CMP) is an Internet protocol used to get an X.509 digital certificate from a CA in PKI. The CMP is used between a CA and LRA or LRA and end-user. The certificate request message mainly contains the following fields:

- **CertReqId**
  Contains an integer value that is used by the certificate requester to associate a specific certificate request with a certificate response.

- **certTemplate**
  Contains a template of an X.509 certificate. The requester fills in certificate template fields for which specific values are desired. The main fields of the certificate template are subject name and public key. The public key is filled with user ECC P-384 public key for which the digital certificate is issued by the CA. The subject name includes mainly Country, Organization, the Organizational Unit name, and Common Name. The Common Name field is filled with a portion of the SmartFusion2 device serial number (DSN). For more information on all the certificate request message fields, refer to https://tools.ietf.org/html/rfc4211.

2.4.4.5 Digital Signature and Signature Verification

It is a mathematical process where hash of the public key certificate is calculated and encrypted using the private key. The encrypted hash code is attached to the original public key certificate as a digital signature. The digital signature is then verified by decrypting the signature using an associated public key and comparing the same with a calculated hash of the public key certificate. The public key certificate remains integral during its transit, if the hash code comparison is successful.

2.4.5 M2M Communication using Digital Certificates

The successful PKI enrollment process fetches the user public key digital certificate from the CA and stores the same to SmartFusion2 eNVM. The user ECC private key is enrolled into the SRAM-PUF hardware block. The M2M communication demo uses the TLS protocol to transmit the messages securely across the machines. The TLS protocol is used to authenticate the server and the client to establish the secure communication between authenticated parties. The TLS protocol uses the user public key certificates stored in eNVM to authenticate the server and the client machines as part of the TLS handshake procedure. Once the server and client machines trust each other by successfully verifying the certificates, then they start transmitting the encrypted messages using shared session secret key.
Figure 5 • Secure Communication between PKI Enrolled SmartFusion2 Devices

The server and client applications support TLS/SSL security protocol that encrypts and decrypts the messages to secure the communication against message tampering. The communication from the TLS server and the TLS client ensures that the sensitive data is translated into a secret code that is difficult to tamper the data. The SmartFusion2 TLS server handles the HTTPS request messages from the SmartFusion2 TLS client.

2.4.5.1 TLS/SSL Protocol

The TLS/SSL protocol is divided into the following two protocol layers:

2.4.5.1.1 Handshake Protocol Layer

This layer consists of the following sub protocols:

- **Handshake**
  Used to negotiate session information between the server and the client. The session information includes a session ID, peer certificates, the cipher spec, the compression algorithm, and a shared secret code that is used to generate required keys.

- **Change Cipher spec**
  Used to change the key used for encryption between the client and the server. The key is computed from the information exchanged during the client-server handshake.

- **Alert**
  Alert messages are generated during the client-server handshake to report an error or a change in status to the peer.
Figure 6 shows the overview of the TLS/SSL handshake procedure. Refer to http://tools.ietf.org/html/rfc5246 for detailed information on handshake protocol, record protocol, and cryptographic algorithms.

**Figure 6 • TLS Handshake Procedure**

2.4.5.1.2 Record Protocol Layer

The record protocol receives and encrypts data from the application and transfers to the transport layer. The record protocol fragments the received data to a size appropriate to the cryptographic algorithm and optionally compresses the data. The protocol applies HMAC and encrypts or decrypts the data using the information negotiated during the handshake protocol.
2.5 Hardware Implementation

The demo design is implemented using an serial gigabit media independent interface (SGMII) PHY interface by configuring the TSEMAC for the ten-bit interface (TBI) operation. For more information on the TSEMAC TBI interface, refer to the UG0331: SmartFusion2 Microcontroller Subsystem User Guide.

Figure 7 shows the Libero SoC hardware design implementation for this design.

The Libero hardware project uses the following SmartFusion2 MSS resources and IPs:

- **TSEMAC TBI**
  This interface is used for Ethernet communication. In this design, the MSS TSEMAC interface is configured for ten-bit interface and SERDES_IF is configured for serial gigabit media independent interface (SGMII) mode.

- **MMUART_1**
  It is used for RS-232 communications on the SmartFusion2 Security Evaluation Kit.

- **MSS_SPI_1**
  It is used for executing key verification protocol using system controller services.

- **General purpose input and output (GPIO)**
  Interfaces with the light emitting diodes (LEDs) and dip switches.

- **System controller services**
To generate ECC key pair and enrolling the key and implement TLS/SSL protocol, PUF, SHA, and Key-Tree protocol. High-speed serial interface (SERDESIF) SERDES_IF IP, configured for SERDESIF_3 EPCS lane3, as shown in Figure 8 on page 19.

For more information on high-speed serial interfaces, refer to the UG0447: IGLOO2 and SmartFusion2 High Speed Serial Interfaces User Guide.

Figure 8 • High-Speed Serial Interface 2 Configurator

2.5.0.1 Package Pin Assignments

Package pin assignments for SPI, LEDs, DIP switches, and Ethernet PHY interface signals are shown in Table 2.

Table 2 • Pins Assignments

<table>
<thead>
<tr>
<th>Port Name</th>
<th>Package Pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED 1 to LED 8</td>
<td>E1, F4, F3, G7, H7, J6, H6, H5</td>
</tr>
<tr>
<td>DIP Switch 1 to DIP Switch 4</td>
<td>L19, L18, K21, K20</td>
</tr>
<tr>
<td>PHY_MDC</td>
<td>J3</td>
</tr>
<tr>
<td>PHY_MDI</td>
<td>J4</td>
</tr>
<tr>
<td>PHY_RST</td>
<td>K6</td>
</tr>
<tr>
<td>SPI_1_CLK_M2F</td>
<td>E4</td>
</tr>
<tr>
<td>SPI_1_DI_F2M</td>
<td>C3</td>
</tr>
<tr>
<td>SPI_1_DO_M2F</td>
<td>C1</td>
</tr>
<tr>
<td>SPI_1_SSO_M2F</td>
<td>D3</td>
</tr>
</tbody>
</table>
2.5.0.2 User Clients in eNVM

The eNVM configurator in the Libero tool is shown in Figure 9. The following user clients are stored in eNVM with different start addresses.

1. **Boot loader**
   - Executes on device power-up from eNVM and loads the demo application into LPDDR to execute the demo application from the LPDDR memory.

2. **PKI_Demo**
   - Demo application, which contains the PKI enrollment, TLS server, and TLS client implementations.

3. **Cert_loc**
   - Place holder for the user public key digital certificate.

4. **Root_CA_Cert**
   - The Root CA public key certificate is stored in eNVM. This is used to verify the signature of the received user public key digital certificate.

5. **Cert_mitm**
   - This SmartFusion2 device certificate is used to emulate the MITM attack.

![Figure 9 • eNVM Configurator - User Clients in eNVM](image)

2.6 Software Implementation

Invoke the SoftConsole project using standalone SoftConsole IDE and open the SoftConsole firmware project located at:

<design files folder>/M2S_PKI_Ref_DF\PKI_Demo_HW\SoftConsole\Webserver_TCP_MSS_CM3 to view the SoftConsole work space.

The following stacks are used for this demo design:

- **PolarSSL library version 1.3.8 (mbed TLS)**
  - To implement the TLS/SSL Protocol.

- **lwIP TCP/IP stack version 1.4.1**
  - To implement light-weight TCP/IP protocol suite.

- **FreeRTOS**
  - To schedule and manage the tasks.
Figure 10 shows the directory structure of the SoftConsole software demo design.

The SoftConsole workspace consists of three projects.

- **Boot loader**
  The boot loader runs from the eNVM. The boot loader copies the demo application from eNVM to the LPDDR memory and executes the application from the LPDDR memory.

- **Webserver_TCP_MSS_CM3_app**
  Contains the PKI enrollment, TLS server, and TLS client implementations using PolarSSL, LWIP, and FreeRTOS.

- **Webserver_TCP_MSS_CM3_hw_platform**
  Contains all the firmware and hardware abstraction layers that correspond to the hardware design. This project is configured as a library and is referenced by the Webserver_TCP_MSS_CM3_app application project. The contents of this folder get over-written by regenerating the root design every time and exporting the SoftConsole firmware project in the Libero SoC software.
2.7 Setting Up the Demo for PKI Enrollment

2.7.1 Installing PKI-Authenticated M2M Communications Demo Utility

The design files include the PKI-Authenticated M2M Communications Demo Utility installer.

1. Browse and navigate to the location where the setup file of the utility installer is located (<design files folder>M2S_PKI_Ref_DF\Utility\setup.exe).
2. Select and double-click the setup file. Follow the on-screen instructions to install the utility on Host PC.

Note: Microsemi recommends to choose a destination folder or drive that has Windows administrative permissions.

2.7.2 Installing User Root CA Public Key Certificate into Windows Certificate Store

The design files include the User Root CA Public Key Certificate.

1. Browse and navigate to the location where the certificate is located (<design files folder>M2S_PKI_Ref_DF\CA_Root_Certificate\Root_CA_Cert.crt).
2. Select and double-click the certificate file (Root_CA_Cert.crt). Follow the instructions provided in the following procedure to install the certificate to the windows certificate store.
3. Accept Windows Security Warning to open the certificate file. Click Open in the Windows Security Warning window to open the certificate.

Figure 11 • Windows Security Warning

4. Click Install Certificate in the Certificate window. A certificate import wizard is displayed, as shown in Figure 12.
5. Click **Next** in the certificate import wizard window.
6. Specify the Certificate Store as Trusted Root Certification Authorities using **Browse** option, as shown in Figure 13.

**Figure 13 • Certificate Store Selection**

7. Follow the on-screen certificate import wizard instructions to install the certificate in Trusted Root Certification Authorities Certificate Store.
8. Click **Yes** to accept the Windows Security Warning to install the certificate as root certificate.
2.7.3 Board Setup for PKI Enrollment

The following steps describe how to setup the SmartFusion2 Security Evaluation Kit to run the demo design:

1. Connect the power supply to the J6 connector, switch on the power supply switch, SW7. **CAUTION:** Ensure that the power supply switch SW7 is switched OFF while connecting the jumpers.

2. Connect the J18 connector (USB mini-B port) of the SmartFusion2 Security Evaluation Kit to the Host PC USB port using the USB A to mini-B cable. This connection establishes the UART communication between the SmartFusion2 device and the Host PC.

3. Install the USB to UART bridge drivers on the Host PC, if the USB to UART bridge drivers are not detected automatically. The drivers are available for download at: http://www.ftdichip.com/Drivers/VCP.htm.

4. Ensure that the USB to UART bridge drivers are detected (you can verify in the Device Manager).

9. Click **OK** in the Certificate Window after importing successfully.
5. From the Device Manager window, identify the COM port that is assigned to the SmartFusion2 UART interface. Figure 15 shows four COM ports with COM27 highlighted. **Note:** Use the highest numbered COM port to run the PKI demo. The COM port numbers vary from system to system.

6. Connect the wall Ethernet port that has Internet connection to the WAN port on the Ethernet Router using an Ethernet cable. Configure the Ethernet Router to provide Internet access to the devices connected to it. Ignore this step, if an Ethernet Router is already configured for this purpose. If the wall Ethernet port is not available, you can use wireless range extender (Wireless repeater) and personal Smartphone to provide Internet access to the SmartFusion2 device and Host PC or Laptop. For more information about using Wireless repeater and Smartphone, refer to “Appendix: Ethernet Settings using Wireless Repeater and Smartphone” on page 66.

7. Connect one of the Ethernet ports of Wireless Router/Repeater to J13 on SmartFusion2 Security Evaluation Kit using an Ethernet cable. This connection provides Internet access to the SmartFusion2 device.

8. Connect the Host PC or Laptop to the Wireless Router/Repeater using Wi-Fi or Ethernet cable for accessing Internet on the Host PC or Laptop.

9. Identify the active network profile (Domain, Private, or Public) and state of the windows firewall in that profile by launching “Windows Firewall with Advanced Security” program. You can search for the word firewall in the Start Menu search box and click **Windows Firewall with Advanced Security** result, as shown in Figure 16.
10. If the Windows firewall state is OFF for the active profile, then skip the following steps and proceed to program the device. For more information, refer to "Programming the SmartFusion2 Device" on page 31.
11. If the Windows firewall state is ON for the active profile, then you need to modify the Windows Firewall Properties to allow inbound connections to the Host PC.
12. Click **Windows Firewall Properties** hyperlink on the Windows Firewall with Advanced Security window and Open Windows Firewall Properties.
13. Click **Active Profile** tab in the Windows Firewall Properties window. In this case, click **Public Profile** tab, as shown in Figure 20.

**Figure 20** • Active Profile Tab Selection
14. In the **Public Profile** tab, change **Inbound connections** setting from **Block (default)** to **Allow** and click **Apply**.

*Figure 21* • Allow Inbound Connections through Windows Firewall - 1

*Figure 22* • Allow Inbound Connections through Windows Firewall - 2

**Note:** Updated state of the Inbound Connections for the active profile is verified in the Windows Firewall with Advanced Security window, as shown in *Figure 22*. 
2.7.4 Programming the SmartFusion2 Device

The following steps describe how to program the SmartFusion2 device with the provided programming file:

1. Connect the FlashPro4 programmer to the J5 connector of the SmartFusion2 Security Evaluation Kit.
2. Connect the Host PC USB port to the J18 USB connector on the SmartFusion2 Security Evaluation Kit board using the USB mini-B cable.
3. Power ON the SmartFusion2 Security Evaluation Kit by changing SW7 switch position to ON.
4. Launch the FlashPro software.
5. Click New Project.
6. In the New Project window, enter a project name.
7. Click Browse and navigate to the location where you want to save the project.
8. Select Single Device as the Programming Mode.
9. Click OK to save the project.
10. Click Configure Device.
11. Click Browse and navigate to the location where M2S_PKI_Demo_Enrollmet_and_M2M.stp file is located. The file is located in <design files folder>\M2S_PKI_Ref_DF\Programming_File and the file name is M2S_PKI_Demo_Enrollmet_and_M2M.stp.
12. The required programming file is selected and is ready to be programmed in the device.
13. Click PROGRAM to start programming the device. An orange LED blinks as programming is in progress. Wait until a message is displayed, which indicates that the program is passed.
14. Disconnect the FlashPro4 programmer and close the FlashPro software.
2.8 Running the Demo for Enrolling SmartFusion2 SoC FPGA Device in PKI

This section describes how to run the demo to enroll the SmartFusion2 device in PKI.

1. Launch the PKI-Authenticated M2M Communications Demo Utility from Start > All Programs > PKI_M2M_GUI > PKI_Demo as Administrator.

2. Right-click and select Run as administrator option, as shown in Figure 24.

Figure 24 • Launching Host PC PKI-Authenticated M2M Communications Demo Utility

The M2M Authentication Using PKI demo GUI contains the PKI Enrollment Service tab. The PKI enrollment service shows the overall PKI enrollment process in seven stages with a status mark of each stage. A green colored tick mark (√) is displayed on successful completion of the stage.
3. Make a note of the LRA IP Address (that is, 10.61.11.66) and click **Start Server** to start the LRA PKI enrollment server. Different LRA IP addresses are provided depending on the Internet network.

4. Click **OK** in the Confirmation dialog. An affirmation message: **HTTP Server Started successfully** is displayed.

5. Reset the SmartFusion2 Security Evaluation Kit using the switch **SW6**.

6. Select COM port number assigned to the SmartFusion2 UART interface, refer to the COM port number that was recorded in the previous section, as shown in Figure 26.
7. Click **Connect** to connect to the SmartFusion2 device using UART interface from Host PC. This step performs the following functions:
   - Acquires a dynamic IP address to the SmartFusion2 device from the Ethernet Router.
   - Creates a SmartFusion2 SRAM-PUF User Activation code, if you are running the demo for the first time.
   - Turns ON the LED G7 and LED F3 on the SmartFusion2 Security Evaluation Kit board.
8. Refer to the log window for SmartFusion2 device serial number (DSN). The least significant 52 bits of the DSN is used as a common name in the public key certificate request. The log window also shows all the messages that the SmartFusion2 device performs during the enrollment process.
9. Click **Start Device Authentication Check**. This step performs the following functions:
   - Generates a random ECC P-384 user key pair using SmartFusion2 built-in ECC hardware engine. It generates a public key and a private key. The SmartFusion2 requests the public key certificate for this user public key using CMP.
   - Enrolls the user public key into SRAM-PUF hardware as key code #2 and it is used to provide the PoP of the user public key.
   - Sends SmartFusion2 Device Certificate to LRA for authentication. LRA checks the authenticity of the SmartFusion2 Device Certificate by verifying the signature of the SmartFusion2 device certificate using the Microsemi Public Key.
   - Displays a green colored tick marks, if the signature check is successful. Else, the PKI enrollment process is stopped when the signature check fails.
Figure 29 • SmartFusion2 Device Authentication
10. Click **Display Device Certificate** on the utility and view the SmartFusion2 device certificate. It displays the SmartFusion2 device certificate that was received using the Windows default security certificate viewer.

11. In the Certificate window, click **Details** tab to see more information such as, the subject and identifier information, certificate validity period, issuer identity information, and digital signature of the issuer, as shown in **Figure 31**.
Figure 31 • SmartFusion2 Device Certificate

12. Click **Log** buttons of the SmartFusion2 User Interface and Host PC LRA User Interface to see all the actions that are performed at SmartFusion2 device and LRA, as shown in **Figure 32**.
13. Click **Provide Proof-of-possession of Private Keys** to provide proof-of-possession of Device Private Key and User Private Key to the LRA. This procedure performs the following functions:

- The LRA generates an ephemeral ECC key pair and sends the public key (E-PK1) to the SmartFusion2 device. Both the LRA and device computes the ECDH shared secret key (SSK1) using a key derivation function.
- The LRA challenges the device to prove it has the same shared secret, thus proving it possesses the device private key and user public key required to compute it by computing Validator1 from the SSK1 and the hash of user public key. The LRA matches Validator1.
- The Utility displays a green colored tick mark, if the validators are matching. Else, the PKI enrollment process is stopped when the Validator1 check fails.
- The LRA generates a new ephemeral ECC key pair and sends the public key (E-PK2) to the SmartFusion2 device. Both the LRA and device computes the ECDH Shared Secret Key (SSK2) using User ECC Key pair and the LRA ephemeral ECC key pair.
- The LRA challenges the device to prove it has the same shared secret, thus proving it possesses the user private key required to compute it by computing Validator2 from the SSK2. The LRA matches Validator2.
- The Utility displays a tick mark if the validators are matching. If the Validator2 check fails, then the PKI enrollment process is stopped.
14. Click **Request User Public Key Certificate** to send the Certificate Request to LRA.

This procedure performs the following functions:

- The LRA checks the user public key present in the certificate request with the one used for proof-of-possession checks and also validates the certificate request signature. If it matches, it approves the request and transfers the approved certificate request to the User Root CA.
- The Utility displays a tick mark, if the user public keys are matching. Else, the PKI enrollment process is stopped when the user public key check fails.
- The User Root CA signs the approved user public key certificate request using the User Root CA's signing key. The signed user public key certificate will be sent back to LRA. This process takes a few seconds to get the response from CA. You can view the signed user public key certificate. Click **Display User Public Key Certificate** on the utility.
- **Note:** The LRA utility connects to CA using the URL 10.testkeys.escrypt.com through the port #80. Ensure that your Internet network has the access rights to connect to the CA. If your Internet network does not allow the LRA utility to connect to the CA, the LRA gives an error message and the PKI enrollment process is stopped.
- SmartFusion2 User Interface shows **Waiting to receive user public key certificate from LRA** message.
SmartFusion2 checks the authenticity of the received certificate using inherently trusted root CA public certificate and stores it in the SmartFusion2 eNVM, if it is a valid certificate. This step also enrolls the user private key into SRAM-PUF hardware as key code #3 for secure key storage.
• This step completes the User PKI enrollment process. The acquired User Public Key Certificate is used for secure communication. SmartFusion2 User Interface shows **PKI enrollment process completed successfully** message, as shown in Figure 35.

**Figure 35**  CA Issued User Public Key Certificate to Device

15. Click **Display User Public Key Certificate** to open the received User Public Key Certificate.
16. In the Certificate window, click **Details** to see more information such as the subject's public key value and identifier information, certificate validity period, issuer identity information, and digital signature of the issuer.

**Note:** The prefix 04 at public key value is used to distinguish uncompressed public keys from compressed public keys that begin with a 02 or a 03.
17. In the Certificate window, click **Certification Path** tab to view the path from the selected certificate to the certification authority (Root CA) that issue the certificate.
18. The SmartFusion2 User Interface and Host PC LRA User Interface, provide Log tabs that captures the information exchanged with the SmartFusion2 device. This information includes SmartFusion2 device public key, device serial number, LRA ECC key pairs, and validators.
2.8.1 Detection of Certificate Requests from Non-authentic Machines

This demo has the feature to emulate an attack that targets PKI enrollment of a non-authentic machine with an authentic SmartFusion2 device certificate. As the SmartFusion2 device certificate of an authentic machine can be exported, a non-authentic machine can send a user public key certificate request to the LRA with the exported authentic device certificate. The security protocol defined between the machine and the LRA detects this attack during the security check of proof-of-possession of private keys.

On the SmartFusion2 Security Evaluation Kit, which acts as a machine, press the SW3 switch before clicking **Start Device Authentication Check (Step )** on the utility and continue with the rest of the instructions, as described in "Running the Demo for Enrolling SmartFusion2 SoC FPGA Device in PKI" section on page 32.

When the user presses the SW3 switch, it turns the machine as a non-authentic machine and sends a certificate request with the device certificate of another authentic machine. The private key corresponding to the public key that is present in the device certificate is not known to the certificate requesting machine. As a result, the security check for proof-of-possession of device private key fails and the enrollment process is stopped.
2.9 **Setting Up the Demo for M2M Communication using the Digital Certificates**

In this part of the demo, two SmartFusion2 devices communicate with each other securely. Figure 40 shows the two SmartFusion2 devices with Ethernet Router setup for M2M communication.

2.9.1 **Enrolling the two SmartFusion2 SoC FPGA Devices in PKI**

The following steps describe enrolling the two SmartFusion2 SoC FPGA devices in PKI:

1. Program the SmartFusion2 device-1 and SmartFusion2 device-2 as explained in "Programming the SmartFusion2 Device" section on page 31.
2. Follow "Running the Demo for Enrolling SmartFusion2 SoC FPGA Device in PKI" section on page 32 to enroll the SmartFusion2 device-1 and SmartFusion2 device-2 separately.

2.9.2 **Setting up the Boards**

The following steps describe how to setup two SmartFusion2 Security Evaluation Kits to run the M2M demo design:

1. Connect the J18 connector (USB mini-B port) of the SmartFusion2 Security Evaluation Kit to the Host PC USB port using the USB A to mini-B cable. This connection establishes the UART communication between the SmartFusion2 device and the Host PC. Install the USB to UART bridge drivers on the Host PC, if the USB to UART bridge drivers are not detected automatically. The drivers are available for download at: [http://www.ftdichip.com/Drivers/VCP.htm](http://www.ftdichip.com/Drivers/VCP.htm).
2. Ensure that the USB to UART bridge drivers are detected (verify in the Device Manager).

*Figure 40 • Windows Device Manager - COM Port Detection*

3. From the Device Manager window, identify the COM port that is assigned to the SmartFusion2 UART interface. Figure 40 shows four COM ports with COM6 highlighted. Make a note of the highest numbered COM port to run the PKI demo. The COM port numbers vary from system to system.
4. Connect the wall Ethernet port that has Internet connection to the WAN port on the Ethernet Router using an Ethernet cable. Configure the Ethernet Router to provide Internet access to the devices connected to it. Ignore this step, if an Ethernet Router is already configured for this purpose. If the wall Ethernet port is not available, you can use wireless range extender (Wireless repeater) and personal Smartphone to provide Internet access to the SmartFusion2 device and Host PC or Laptop. For more information about using Wireless repeater and Smartphone, refer to "Appendix: Ethernet Settings using Wireless Repeater and Smartphone" on page 66.

5. Connect one of the Ethernet ports of Wireless Router/Repeater to J13 on SmartFusion2 Security Evaluation Kit using an Ethernet cable. This connection provides Internet access to the SmartFusion2 device.

6. Connect the Host PC or Laptop to the Wireless Router/Repeater using Wi-Fi or Ethernet cable for accessing Internet on the Host PC or Laptop. Connect the Power supply to the J6 connector, switch on the power supply switch, SW7.

7. Repeat steps 1, 2, 3, 5, and 7 for another SmartFusion2 device. From the Device Manager window, identify the COM port that is assigned to the SmartFusion2 UART interface. Figure 41 shows 8 COM ports for two SmartFusion2 devices with COM6 and COM49 highlighted.

8. See Figure 42 for demo setup. Ensure, that the two SmartFusion2 devices are connected to the Host PC or laptop using USB A to Mini-B cables for serial communication.
Figure 42 • Router Demo Setup

Figure 43 • M2M Demo Setup
2.10 Running the Demo for M2M Communication using the Digital Certificates

This section describes how to run the demo for M2M communication.

1. Launch the PKI-Authenticated M2M Communications Demo Utility from Start > All Programs > PKI_M2M_GUI>PKI_Demo as Administrator (Right-click and select Run as administrator option). The PKI-Authenticated M2M Communications Demo Utility has M2M Communication tab.

2. Click M2M Communication tab. Power-cycle both the SmartFusion2 Security Evaluation Kit boards, and select COM port numbers for both the boards in the demo user interface.
Figure 45 • Selecting COM Ports for the Machines

Note: In this example, SmartFusion2 Machine, which has COM port number 6 is selected for TLS server and another SmartFusion2 Machine having COM port number 49 is selected for TLS Client.

3. Click Connect in the Server Machine User Interface to configure the SmartFusion2 Machine as TLS server. This step performs the following functions:
   • Changes the Connect button to Connected.
   • Acquires a dynamic IP address to the SmartFusion2 Server Machine from the Ethernet Router.
   • Retrieves User Private Key using SRAM-PUF service.
   • Turns ON the LED H7 and LED J6 on the SmartFusion2 Security Evaluation Kit board.
   • Displays the DIP switch status of the SmartFusion2 Server Machine on GUI.
4. Click **Connect** of the Client Machine window to configure the SmartFusion2 Machine as TLS client. This step performs the following functions:
   - Changes the Connect button to **Connected**.
   - Acquires a dynamic IP address to the SmartFusion2 client Machine from the Ethernet Router.
   - Retrieves User Private Key using SRAM-PUF service.
   - Turns ON the LED H6 and LED H5 on the SmartFusion2 Security Evaluation Kit board.
   - Client machine gets the server machine IP address from GUI.
Figure 47 • Selecting SmartFusion2 Machine as Client

Note: Power-cycle the boards, relaunch the GUI and select the com ports again if the roles of the machines have to be changed.

5. Click Log buttons to display the log messages of the server machine and client machine, as shown in Figure 48.
2.10.1 Secure M2M Communication using Client-server Model

In M2M communications, one of the two SmartFusion2 devices is configured as TLS server and the other device as TLS client. The TLS server responds to the trusted TLS client requests with the encrypted messages.

The TLS Client Machine performs the following operations:

- Transfers the data securely to the server.
- Sends a command securely to the server to blink LEDs.
- Sends a command securely to the server to get dip switches status.

2.10.1.1 Transferring the Data Securely to the Server

This section describes how to transfer the data securely to the server.

1. Click **Send Data to Server** to communicate with the server securely. Selecting TLS client option for the first time enables both the machines to perform the TLS authentication to agree upon cipher
Enabling a Public Key Infrastructure in SmartFusion2 Devices

suite, as shown in Figure 49. The TLS session information is cached and saved for future communication between server and client.

**Figure 49 • TLS Handshaking**

2. On successful TLS handshake, the log window displays the TLS protocol version and cipher suite algorithm, as shown in Figure 50.
Figure 50 • TLS Protocol and Cipher Suite
The server machine displays the data received from the trusted client.

**Note:** You can enter the information in **Data to be sent to Server** to send the data from the client to sever securely. Ensure that the data is more than four characters. The fixed data gets transferred from the client to server securely when the text box is empty. See Figure 52, the text SmartFusion2 gets transferred securely from the client to the server.
Figure 52 • Transferring Data Securely to Server - 2
2.10.1.2 Sending a Command Securely to the Server to blink LEDs

This section describes how to send a command securely to the server to blink LEDs.

1. Click Blink Server LEDs to send the command securely to the server to blink the LEDs.

Figure 53 • Server Machine’s LED Blinking

The Server Machine’s LEDs start blinking on receiving the command from the trusted client. The server Machine’s LEDs blink for approximately 12 seconds and stop automatically. The user interface also shows the LEDs blink, as shown in Figure 53.
2.10.1.3 Sending a Command Securely to the Server to get DIP Switches Status

This section describes how to send a command securely to the server to get DIP switches status.

1. Click **Get Server DIP Switch Status** to send the command securely to the server to get the DIP switches status.

![Figure 54 • DIP Switches Status](image)

2. Change the Server machine’s DIP switches position (SW5) manually to get the corresponding DIP switches status message at the Client Machine.

3. Switch OFF the SmartFusion2 Security Evaluation Kit boards when you are finished with the demonstration.

2.11 Conclusion

This reference design describes how to enable a public key infrastructure using built-in security capabilities in the SmartFusion2 SoC FPGA devices. It also explains the secure M2M communication using PKI-enrolled SmartFusion2 SoC FPGA devices.
Figure 55 shows the SmartFusion2 Security Evaluation Kit board. You need to do the following rework on the SmartFusion2 Security Evaluation Kit Rev D for running the demo.

**Figure 55 • SmartFusion2 Security Evaluation Board**

Make the following connections on the board for connecting MSS_SPI1 interface to the system controller SPI (SC_SPI) interface. The MSS_SPI1 signals are routed to the MSIOs through the FPGA fabric. These signals are connected to the GPIO header (J1) available on the board. SC_SPI signals can be accessed through Vias as shown in the Figure 56 and Figure 57 on page 63.
Figure 56 • SC_SPI Signals Location

Figure 57 • SC_SPI Signal Names
1. Make the following connections between GPIO header (J1) and SC_SPI signals.
   • J1-48 to SC_SPI_SDO
   • J1-42 to SC_SPI_SCK
   • J1-54 to SC_SPI_SDI
   • J1-60 to SC_SPI_SS

2. De-populate the following resistors: R105, R111, R112, and R113. These resistors are available next to FTDI chip (FT4232H), as shown in Figure 58.

Figure 58 • Location of the Resistors (R105, R111, R112 and R113) to be De-populated
Appendix: Structure of an X.509 v3 Digital Certificate

The following is the structure of an X.509 v3 digital certificate:

- Certificate
  - Version
  - Serial Number
  - Algorithm ID
  - Issuer
  - Validity
  - Not Before
  - Not After
- Subject
  - Subject Public Key Info
  - Public Key Algorithm
  - Subject Public Key
  - Issuer Unique Identifier (optional)
  - Subject Unique Identifier (optional)
  - Extensions (optional)
- Certificate Signature Algorithm
- Certificate Signature

The SmartFusion2 device example user public key digital certificate is, as shown in Figure 59.

Figure 59 • Example Digital Certificate
5 Appendix: Ethernet Settings using Wireless Repeater and Smartphone

**Note:** Ensure the Wireless repeater has the Ethernet ports.

1. Enable Portable Wi-Fi hotspot feature in your personal smartphone to share smartphones 3G/4G data connection.
2. Configure wireless repeater to extend the smartphone's data connection to its Ethernet ports.

*Figure 60* shows the PKI enrollment setup using Wi-Fi repeater and smartphone.

*Figure 60* • PKI enrollment Setup using Wi-Fi Repeater and Smartphone
## Revision History

The following table shows important changes made in this document for each revision.

<table>
<thead>
<tr>
<th>Revision</th>
<th>Changes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Revision 2</td>
<td>Updated the document for Libero v11.7 software release (79705).</td>
</tr>
<tr>
<td>(May 2016)</td>
<td></td>
</tr>
<tr>
<td>Revision 1</td>
<td>Initial release.</td>
</tr>
<tr>
<td>(November 2015)</td>
<td></td>
</tr>
</tbody>
</table>

**Note:** The revision number is located in the part number after the hyphen. The part number is displayed at the bottom of the last page of the document. The digits following the slash indicate the month and year of publication.
7 Product Support

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group and using these support services.

7.1 Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

- From North America, call 800.262.1060
- From the rest of the world, call 650.318.4460
- Fax, from anywhere in the world, 408.643.6913

7.2 Customer Technical Support Center

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues, and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

7.3 Technical Support


7.4 Website


7.5 Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

7.5.1 Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is soc_tech@microsemi.com.

7.5.2 My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.
7.5.3  **Outside the U.S.**

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office. Visit About Us for sales office listings and corporate contacts.

7.6  **ITAR Technical Support**

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech@microsemi.com. Alternatively, within My Cases, select Yes in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.