

# Using Device Certificate System Service in SmartFusion2 - Libero SoC v11.6

## Table of Contents

Purpose . . . . .	1
Introduction . . . . .	1
Device Certificate Application . . . . .	2
SmartFusion2 Device Certificate . . . . .	2
References . . . . .	3
SmartFusion2 Device Certificate Service . . . . .	3
Using Device Certificate Service . . . . .	5
Design Requirements . . . . .	7
Design Description . . . . .	7
Hardware Implementation . . . . .	8
Software Implementation . . . . .	9
Setting up the Design . . . . .	9
Running the Design . . . . .	11
Viewing Fields in Device Certificate . . . . .	12
Conclusion . . . . .	16
Appendix A: Design and Programming Files . . . . .	17
Appendix B: Decoding Device Certificate Using ASN.1 JavaScript Decoder Open Source Tool . . . . .	18
Appendix C: Feature Activation . . . . .	21
List of Changes . . . . .	22

## Purpose

This application note describes how to read and export device certificate using system services and analyze the content of the device certificate in the SmartFusion<sup>®</sup>2 system-on-chip (SoC) field programmable gate array (FPGA) devices.

## Introduction

The device certificate includes a digital signature, an electronic analogue of a written signature. The digital signature assures that the claimed signatory signed the information. In addition, a digital signature detects whether or not the information was modified after it was signed.

Microsemi<sup>®</sup> SmartFusion2 device certificate is an X.509 complaint digital certificate that is digitally signed with a Microsemi private key. The X.509 is an ITU-T standard for a public key infrastructure (PKI) and privilege management infrastructure (PMI). It specifies standard formats for public key certificates, certificate revocation lists, attribute certificates, and a certification path validation algorithm.

The device certificate in the SmartFusion2 devices cryptographically binds the device serial number, date code, its model or part number, the device's secret factory key, and a digital signature from Microsemi that is validated internally by the device and externally by the user.

The digital certificate is stored in the device's embedded non-volatile memory (eNVM). The bigger devices (M2S150 and M2S090) support elliptic curve cryptography (ECC). Therefore, the factory ECC public keys are also certified and included in the device certificates.

## Device Certificate Application

The primary advantage of the device certificate application is to prevent counterfeiting and fraud. Counterfeiting in electronic parts can take various forms, such as:

- Cloning designs at the transistor level
- Black-topping and re-marking devices to misrepresent used devices as new
- Changing the date codes
- Improving the speed grade or the temperature grade, and increasing the alleged screening level

As a result, any mismatch between how the device is represented by its shipping paperwork or the label printed on its surface and the digital certificate indicates the possibility of counterfeiting fraud.

One application for a SmartFusion2 device certificate is that if a counterfeiter remarks a device with a faster speed grade, the model number authenticated in the device certificate still reflects the true speed grade. When the user attempts to program such a device with a design that was compiled for the faster speed grade device, the programmer observes that the speed grade reflected in the certificate is incorrect for the design.

## SmartFusion2 Device Certificate

The SmartFusion2 device certificate is encoded in the abstract syntax notation one format: ASN.1. It is a standard and notation that describes rules and structures for representing, encoding, transmitting, and decoding data in telecommunications and computer networking. The formal rules enable representation of objects that are independent of machine-specific encoding techniques. Formal notation makes it possible to automate the task of validating whether a specific instance of data representation abides by the specifications.

**Table 1 • SmartFusion2 Device Certificate Fields and Descriptions**

Field Name	Description
Version	Contains the version information
Serial Number	Contains the serial number information
Signature Algorithm	Provides information about the algorithm that is used to generate the signature
Issuer	Provides information about certificate issuers information like: Country Name, Organization Unit Name, Organization Name, and Common Name information
Validity	Provides information about validity of the certificate <ul style="list-style-type: none"><li>• Not Before (start time specified for the certificate validity)</li><li>• Not After (end time specified for the certificate validity)</li></ul> <i>Note: The certificate is only valid between these specified time fields.</i>
Subject	Provides information about generation qualifier, surname, and given name
Subject Public Key Info	Provides the information about the public key generation algorithm and public key information <ul style="list-style-type: none"><li>• Public Key Algorithm</li><li>• Subject Public Key</li></ul>
Issuer Unique Identifier	It contains issuer unique identification string of 9 bytes size
Subject Unique Identifier	It contains 0x00+factory serial number (FSN) + serial number modifier (SNM). For more information about FSN and SNM descriptions, refer to the <a href="#">UG0443: SmartFusion2 and IGLOO2 FPGA Security and Reliability User Guide</a> .

**Table 1 • SmartFusion2 Device Certificate Fields and Descriptions (continued)**

Field Name	Description
Extensions	There are six extensions present in this device certificate. These extensions provide information about padding, date code, temp/speed/voltage grade, feature activation, pricing ID, and certificate validator information.
Certificate Signature Algorithm	Provides information about the algorithm that is being used.
Certificate Signature	Provides the certificate signature information. The signature of the SmartFusion2 device certificate can be verified using Microsemi public key.

## References

The following documents complement and help in understanding the device flows and features:

- [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](#)

## SmartFusion2 Device Certificate Service

SmartFusion2 device certificate service is a part of device and design information services of the system services. These system services are performed by the system controller block.

The device certificate service provides access to the system controller's device and design information services. This service is accessed through the communication block (COMM\_BLK).

There are two COMM\_BLK instances:

- Located in the microcontroller sub system (MSS)
- Located in the system controller

The COMM\_BLK consists of an APB interface, eight byte transmit FIFO, and eight byte receive FIFO. The COMM\_BLK provides a bi-directional message passing facility between the MSS and the system controller.

The device certificate service is initiated using the COMM\_BLK in the MSS, which can be read or written by any master on the AMBA high performance bus (AHB) matrix; typically either the ARM® Cortex®-M3 processor or a design in the FPGA fabric (also known as a fabric master).

The system controller receives the command through the COMM\_BLK in the system controller. On completion of the requested service, the system controller returns a status message through the COMM\_BLK. The responses generated are based on the selected command.

Figure 1 shows the system controller block in SmartFusion2.

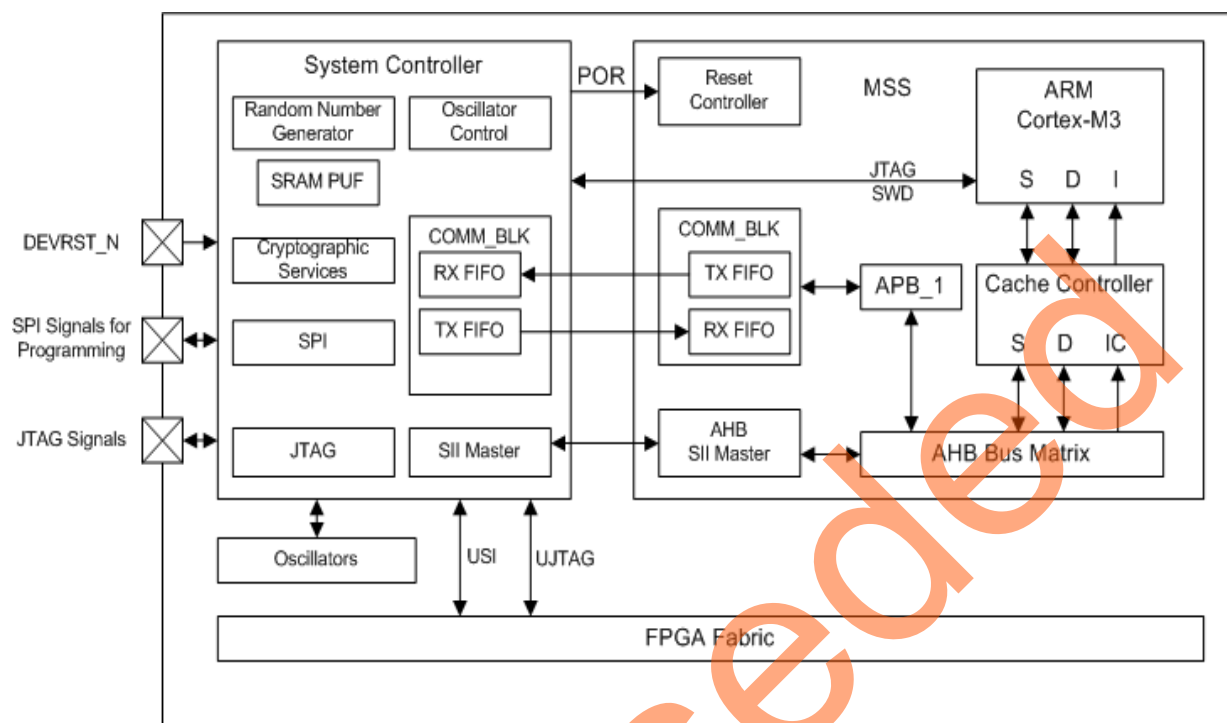


Figure 1 • System Controller Interface with MSS and FPGA Fabric

## Using Device Certificate Service

The device certificate service is initiated using the COMM\_BLK. The COMM\_BLK base address resides at 0x40016000 and extends to address 0x40016FFF in the Cortex-M3 processor memory map. [Table 2](#) summarizes the control and status registers for the COMM\_BLK.

For more information about COMM\_BLK registers description, refer to the "Communication Block" chapter in the [UG0331: SmartFusion2 Microcontroller Subsystem User Guide](#).

**Table 2 • COMM\_BLK Register Map**

Register Name	Address Offset	R/W	Reset Value	Description
CONTROL	0x00	R/W	0x00	Control Register
STATUS	0x04	R/W	0x00	Status Register
INT_ENABLE	0x08	R/W	0x00	Interrupt Enable
DATA8	0x10	R/W	0x00	Byte Data Register
DATA32	0x14	R/W	0x00000000	Word Data Register
FRAME_START8	0x18	R/W	0x00	Frame/Command Byte Register
FRAME_START32	0x1c	R/W	0x00000000	Frame/Command Word Register

The following are the basic steps to use the device certificate system service in the SmartFusion2 devices:

1. Disable the COMM BLOCK loop back mode by writing "1" to LOOPBACK bit (bit-5) of the CONTROL register (0x40016000).
2. Enable the COMM block by writing "1" to ENABLE bit (bit-4) of the CONTROL register.
3. Enable the receive interrupt by writing "1" to RCVOKAY bit (bit-1) of the INT\_ENABLE register (0x40016008).
4. Enable the COMM\_BLK\_INTR (INTISR[19]) in Cortex-M3 Processor interrupts.
5. Set up the COMM\_BLK register in byte mode by writing "0" to SIZETX bit (bit-2) of the CONTROL register.
6. Wait for the TXTOKAY bit (bit-0) of STATUS register(0x40016004) to become 1.
7. Send the device certificate command by writing the register FRAME\_START8 with the command value. The command value of the device certificate service is 0x00.
8. Set up the COMM\_BLK in 4 bytes mode by writing "1" to SIZETX bit (bit-2) of the CONTROL register.
9. Wait for the TXTOKAY status bit to become 1.
10. Send the DEVICECERTPTR address, by writing the register DATA32 with the DEVICECERTPTR value. For more information about the device certificate service request details, refer to [Table 3](#).
11. After completion of the device certificate service, system controller returns a response through the COMM\_BLK instance.
12. The service response includes the 1 byte command, 1 byte STATUS, and 4 bytes DEVICECERTPTR value. The 768 Bytes device certificate is stored in the location pointed by the DEVICECERTPTR pointer. For more information about device certificate service response details, refer to [Table 4 on page 6](#).

**Note:** Microsemi recommends using system services driver provided in the firmware core configurator for detailed implementation of the device certificate service.

Table 3 shows the device certificate service request details.

**Table 3 • Device Certificate Service Request**

Offset	Length (bytes)	Field	Description
0	1	CMD = 0	Command
1	4	DEVICECERTPTR	Pointer to 768-byte buffer to receive the device certificate.

Table 4 shows the device certificate service response details.

**Table 4 • Device Certificate Service Response**

Offset	Length (bytes)	Field	Description
0	1	CMD = 0	Command
1	1	STATUS	Command status
2	4	DEVICECERTPTR	Pointer to original buffer from request

For more information about System Controller, refer to the [UG0450: SmartFusion2 SoC and IGLOO2 FPGA System Controller User Guide](#).

For more information about COMM\_BLK, refer to the Communication Block chapter in the [UG0331: SmartFusion2 Microcontroller Subsystem User Guide](#).

# Design Requirements

Table 5 shows the design requirements.

**Table 5 • Design Requirements**

Design Requirements	Description
<b>Hardware Requirements</b>	
SmartFusion2 Security Evaluation Kit: <ul style="list-style-type: none"> <li>12 V adapter</li> <li>FlashPro4 programmer</li> <li>USB A to Mini-B cable</li> </ul>	Rev D or later
Host PC or Laptop	Any 64-bit Windows Operating System
<b>Software Requirements</b>	
Libero® System-on-Chip (SoC)	v11.6
FlashPro programming software	v11.6
USB to UART drivers	–
One of the following serial terminal emulation programs: <ul style="list-style-type: none"> <li>HyperTerminal</li> <li>TeraTerm</li> <li>PuTTY</li> </ul>	–

## Design Description

The design is implemented on the SmartFusion2 Security Evaluation Kit board using the M2S090TS-1FGG484 device.

The design example consists of:

- RC oscillator
- Fabric CCC
- CORERESET
- MSS (DeviceCertificate\_MSS\_0)

The fabric PLL is used to provide the base clock for the MSS. The system services are run using various C routines in the MSS, as shown in the following sections. In addition, a universal asynchronous receiver/transmitter (UART1) in the MSS is used to display the device certificate information.

## Hardware Implementation

Figure 2 shows a block diagram of the design example. The RC oscillator generates a 50 MHz input clock and the fabric PLL generates a 100 MHz clock from the RC oscillator. This 100 MHz clock is used as the base clock for the MSS (DeviceCertificate\_MSS\_0).

The MMUART\_1 signals are used for communicating with the host PC serial terminal program.

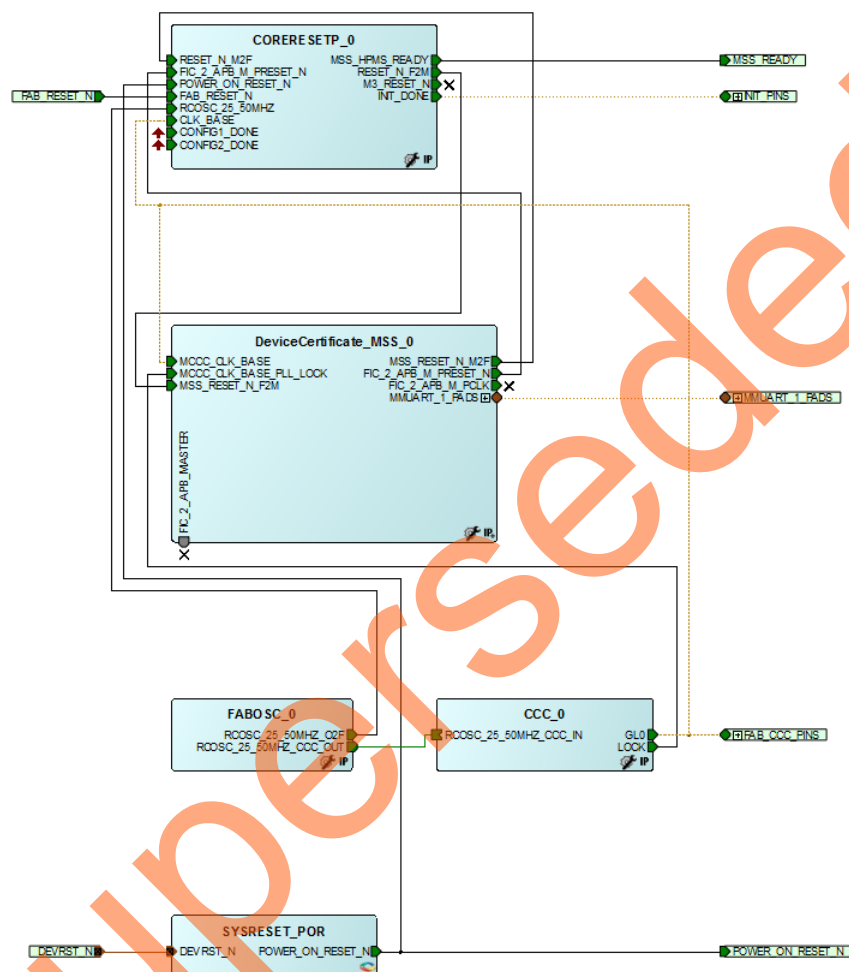


Figure 2 • Block Diagram of SmartFusion2 Device Certificate Design Example



## Software Implementation

The software design example is used to display the device certificate information.

### Firmware Drivers

The following firmware drivers are used in this application:

- MSS MMUART driver: To communicate with serial terminal program on the host PC.
- MSS system services driver: Provides access to SmartFusion2 system services.

### API to Access the Device Certificate Service

The MSS\_SYS\_get\_device\_certificate() API is used in software design to access the device certificate service.

## Setting up the Design

Ensure that power supply switch SW7 is switched OFF before setting up the SmartFusion2 Security Evaluation Kit, then proceed with the following steps:

1. Plug the FlashPro4 ribbon cable into the connector J5 (JTAG Programming Header) of the SmartFusion2 Security Evaluation Kit board.
2. Connect FlashPro4 and the USB port of the PC using the mini USB cable.
3. Connect the power supply to the J6 connector.
4. Connect the J18 connector provided on the SmartFusion2 Security Evaluation Kit to the host PC using the USB mini cable.
5. Ensure that the USB to UART bridge drivers are automatically detected by verifying the Device Manager.

Figure 3 shows an example device manager window. If USB to UART bridge drivers are not installed, download and install the drivers from the following location:

[www.microsemi.com/soc/documents/CDM\\_2.08.24\\_WHQL\\_Certified.zip](http://www.microsemi.com/soc/documents/CDM_2.08.24_WHQL_Certified.zip)

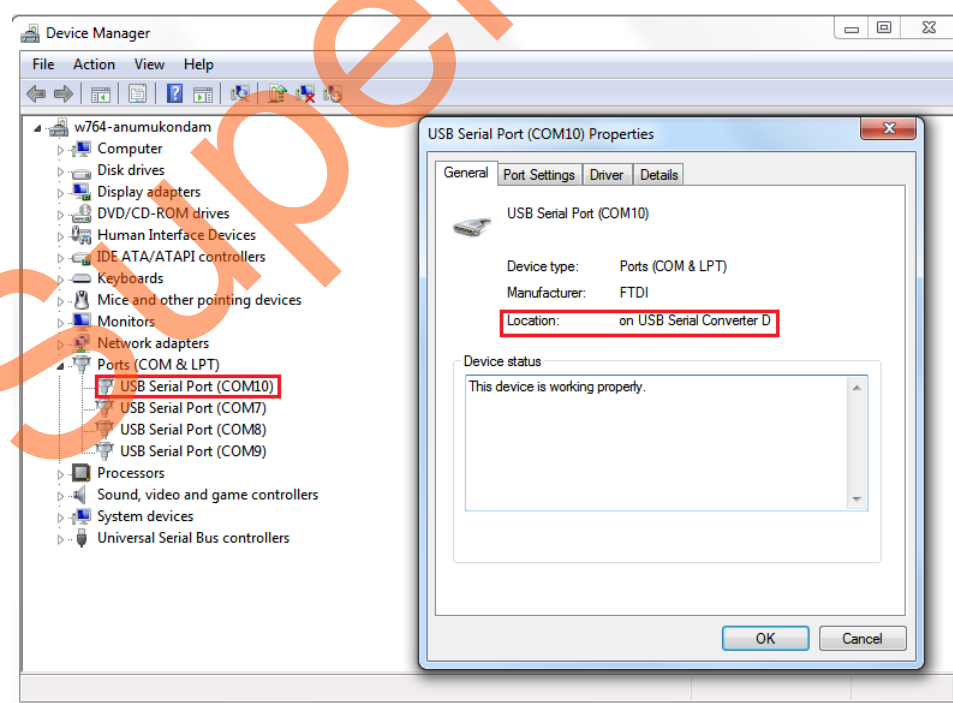


Figure 3 • Device Manager Window

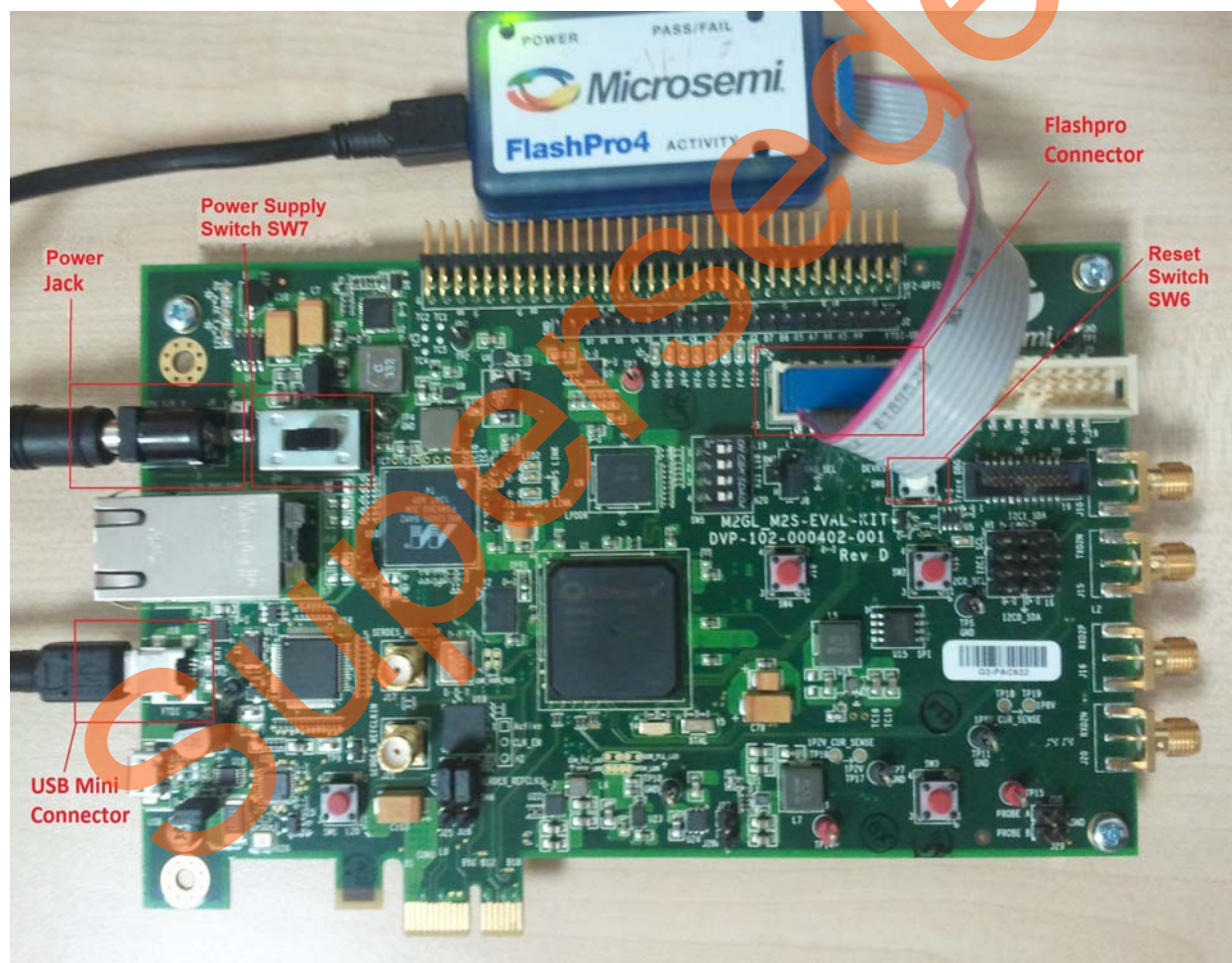
6. Connect the jumpers on the SmartFusion2 Security Evaluation Kit as shown in Table 6.

**Note:** Ensure that power supply switch **SW7** is switched OFF while connecting the jumpers on the SmartFusion2 Security Evaluation Kit.

**Table 6 • SmartFusion2 SoC FPGA Security Evaluation Kit Jumper Settings**

Jumper	Pin (from)	Pin (to)	Comments
J22	1	2	default
J23	1	2	default
J24	1	2	default
J8	1	2	default
J3	1	2	default

Figure 4 shows the board setup for running the ECC services design on the SmartFusion2 Security Evaluation Kit.



**Figure 4 • SmartFusion2 Security Evaluation Kit**

## Running the Design

The following steps describes how to run the design on the SmartFusion2 Security Evaluation Kit board using the M2S090TS-1FGG484 device.

1. Switch ON the power supply switch, **SW7**.
2. Start a PuTTY session with 115200 baud rate, 8 data bits, 1 stop bit, no parity, and no flow control. Use any free serial terminal emulation program such as: HyperTerminal or TeraTerm, if the computer does not have the PuTTY program. For more information about configuring HyperTerminal, TeraTerm, or PuTTY, refer to the [Configuring Serial Terminal Emulation Programs Tutorial](#).
3. Program the SmartFusion2 Security Evaluation Kit board with the provided STAPL file using FlashPro4. Refer to "Appendix A: Design and Programming Files" on page 17 for more information.
4. After programming, press the reset switch, **SW6** (DEVRST), then HyperTerminal displays a message and the device certificate information as shown in Figure 5.



```

COM20 - PuTTY
***** SmartFusion2 System Services Device Certificate Example *****
*****
This example project displays information about the Device Certificate
retrieved using the SmartFusion2 System Services.
It uses the following System Services driver functions:
- MSS_SYS_get_device_certificate()
-----
Device certificate:
30 82 02 e9 30 82 02 6f a0 03 02 01 02 02 12 40
92 ad 6f 0a a6 85 6a fe e4 f6 05 20 02 9b c3 e6
01 30 0a 06 08 2a 86 48 ce 3d 04 03 04 30 43 31
41 30 09 06 03 55 04 06 13 02 55 53 30 0a 06 03
55 04 0b 0c 03 53 6f 43 30 0b 06 03 55 04 0a 0c
04 4d 53 43 43 30 1b 06 03 55 04 03 0c 14 36 39
36 37 34 36 33 36 34 36 63 34 37 36 37 34 66 32
37 65 30 20 17 0d 31 32 31 32 30 31 30 30 30 30
30 30 5a 18 0f 32 31 39 39 31 32 33 31 30 30 30
30 30 30 5a 30 4e 31 4c 30 0b 06 03 55 04 2c 0c
04 31 20 20 20 20 30 13 06 03 55 04 04 0c 73 6d
61 72 74 46 75 73 69 6f 6e 32 30 28 06 03 55 04
2a 0c 21 20 20 20 20 20 4d 32 53 30 39 30 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 30 76 30 10 06 07 2a 86 48 ce 3d 02
01 06 05 2b 81 04 00 22 03 62 00 04 12 e4 04 4e
65 9d 39 85 99 17 57 38 31 19 5d 4d 07 ca 5e 1a
10 cb de 46 0b eb 2d 5e e5 04 36 f0 f9 5a a9 89
1e 5d 87 be b9 89 6e de 03 22 52 b3 59 b9 13 b1
51 47 89 80 24 34 32 81 41 ef a5 06 70 84 e9 ca
4b 8e 58 7e 1b 35 57 56 4c a1 36 08 78 71 a9 a5
20 68 6e 7a 61 c7 71 88 9c 41 a8 55 81 09 00 c1
93 d0 b2 cf 6b 40 00 82 11 00 32 b8 a9 66 a4 07
4b 1b 88 5a 54 7a e6 64 8a 02 a3 81 fa 30 81 f7
30 69 06 0a 2b 06 01 04 01 82 bd 64 01 00 04 5b
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
06 01 04 01 82 bd 64 01 01 04 04 36 32 34 31 30
1e 06 0a 2b 06 01 04 01 82 bd 64 01 02 04 10 ff
ff ff ff ff ff ff ff ff ff ff ff ff 00 00 30
12 06 0a 2b 06 01 04 01 82 bd 64 01 03 04 04 13
00 08 c0 30 12 06 0a 2b 06 01 04 01 82 bd 64 01
04 04 04 00 00 00 00 30 2e 06 0a 2b 06 01 04 01
82 bd 64 01 0a 04 20 c7 c3 af 18 45 8f c1 8d 8f
3c e8 88 2e 42 ea d3 a6 e1 b2 52 9d 19 cd 1e b6
f8 96 63 08 16 6c a4 30 0a 06 08 2a 86 48 ce 3d
04 03 04 03 68 00 30 65 02 30 74 d7 c4 68 c1 9b
63 4a a5 fe a5 9a ba e4 af 4b f5 df 25 87 bf 28
79 c4 c6 05 49 5e c7 89 a7 ce e2 ad 5c 78 38 75
3a ca 3c cc 44 74 62 83 a2 bc 02 31 00 d8 f8 d4
e4 b7 36 71 c3 a9 e5 1a 5b 2f 34 41 17 ac 2b a6
68 fe a9 a1 c1 eb 72 d4 ce 4f 8a e1 2d eb fc 55
cb b8 17 9a 80 da 20 ba 1c 81 30 4b ee 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
-----
  
```

Figure 5 • Device Certificate in Hexadecimal Format (also known as base 16 or hex)

## Viewing Fields in Device Certificate

The SmartFusion2 device certificates are encoded in the ASN.1 format. To view the content, the certificates need to be decoded to a user readable format. The content of a device certificate can be decoded in many ways, such as:

- Use windows utility certutil.exe
- Use open source or other third-party online tools

For more information about how to decode device certificate using ASN.1 JavaScript decoder online tool, refer to ["Appendix B: Decoding Device Certificate Using ASN.1 JavaScript Decoder Open Source Tool" on page 18](#).

This application note uses the `certutil.exe` windows command tool utility to decode the device certificate. The following steps describe how to decode the device certificate:

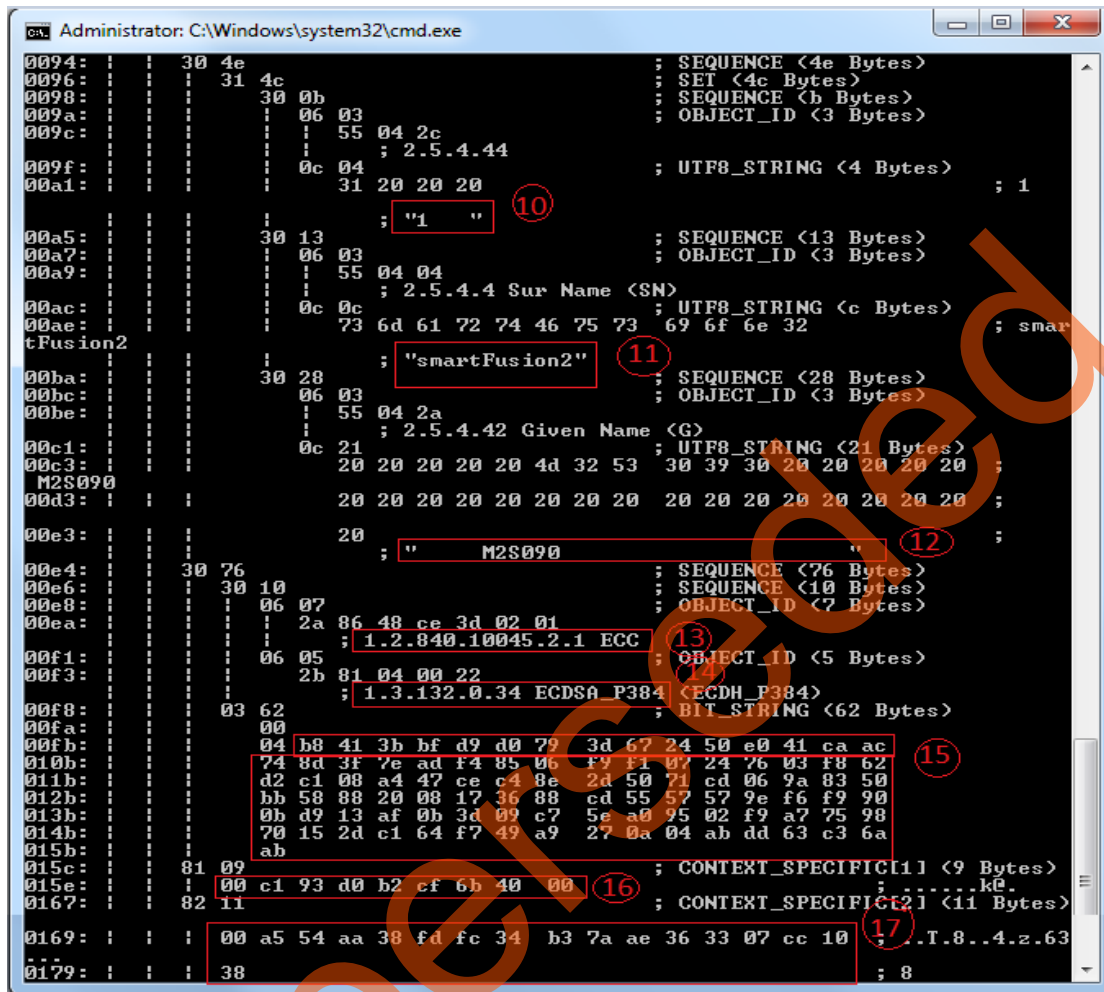
1. Copy the device certificate HEX values (768 bytes) from the serial terminal program (PuTTY/HyperTerminal) to a text file. For example, copy the device certificate HEX values to the DC\_HEX.txt file and save the file in the C:\ drive.
2. Open the command prompt window and type the command `C:\>certutil.exe -asn C:\DC_HEX.txt` and click Enter key.
3. The command prompt displays the decoded device certificate in a user readable format.

The device certificate fields are highlighted in [Figure 6 on page 13](#), [Figure 7 on page 14](#), [Figure 8 on page 15](#), and [Figure 9 on page 16](#).





Figure 7 shows the device certificate - screen 2.



```

0094: 30 4e ; SEQUENCE <4e Bytes>
0096: 31 4c ; SET <4c Bytes>
0098: 30 0b ; SEQUENCE <b Bytes>
009a: 06 03 ; OBJECT_ID <3 Bytes>
009c: 55 04 2c ; 2.5.4.44
009f: 0c 04 ; UTF8_STRING <4 Bytes>
00a1: 31 20 20 20 ; 1
00a5: 30 13 ; SEQUENCE <13 Bytes>
00a7: 06 03 ; OBJECT_ID <3 Bytes>
00a9: 55 04 04 ; 2.5.4.4 Surname <SN>
00ac: 0c 0c ; UTF8_STRING <c Bytes>
00ae: 73 6d 61 72 74 46 75 73 69 6f 6e 32 ; smartFusion2
00ba: 30 28 ; SEQUENCE <28 Bytes>
00bc: 06 03 ; OBJECT_ID <3 Bytes>
00be: 55 04 2a ; 2.5.4.42 Given Name <G>
00c1: 0c 21 ; UTF8_STRING <21 Bytes>
00c3: 20 20 20 20 20 4d 32 53 30 39 30 20 20 20 20 ;
M2S090
00d3: 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 ;
00e3: 20 ; " M2S090 "
00e4: 30 76 ; SEQUENCE <76 Bytes>
00e6: 30 10 ; SEQUENCE <10 Bytes>
00e8: 06 07 ; OBJECT_ID <7 Bytes>
00ea: 2a 86 48 ce 3d 02 01 ; 1.2.840.10045.2.1 ECC
00f1: 06 05 ; OBJECT_ID <5 Bytes>
00f3: 2b 81 04 00 22 ; 1.3.132.0.34 ECDSA_P384
00f8: 03 62 ; BIT_STRING <62 Bytes>
00fa: 00
00fb: 04 b8 41 3b bf d9 d0 79 3d 67 24 50 e0 41 ca ac
010b: 74 8d 3f 7e ad f4 85 06 f9 f1 07 24 76 03 f8 62
011b: d2 c1 08 a4 47 ce c4 8e 2d 50 71 cd 06 9a 83 50
012b: bb 58 88 20 08 17 36 88 cd 55 57 57 9e f6 f9 90
013b: 0b d9 13 af 0b 3d 09 c7 5e a0 95 02 f9 a7 75 98
014b: 70 15 2d c1 64 f7 49 a9 27 0a 04 ab dd 63 c3 6a
015b: ab
015c: 81 09 ; CONTEXT_SPECIFIC11 <9 Bytes>
015e: 00 c1 93 d0 b2 cf 6b 40 00 ; k
0167: 82 11 ; CONTEXT_SPECIFIC11 <11 Bytes>
0169: 00 a5 54 aa 38 fd fc 34 b3 7a ae 36 33 07 cc 10 ; T.8..4.z.63
0179: 07 cc 10 38 ; 8
  
```

Figure 7 • Device Certificate - Screen 2

The following is the description of labels in Figure 7:

#### Subject Information

10. Rev (Generation Qualifier) 4 bytes fixed length: "1 "

11. Family (Surname): " SmartFusion2"

12. Product ID 33 characters (Given name): " M2S090 "

#### Subject Public Key

##### Public Key Algorithm Information

13. ECC

14. ECDSA\_P384

##### Subject Public Key information

15. 96 Byte ECC public key

#### Issuer Unique Identifier

16. 9 byte bit string: 00 c1 93 d0 b2 cf 6b 40 00

#### Subject Unique Identifier

17. (0x00+Factory Serial Number + Serial Number Modifier): 00 a5 54 aa 38 fd fc 34 b3 7a ae 36 33 07 cc 10 07 cc 10 38

Figure 8 shows the device certificate - screen 3.



```

0179: 38 ; 8
017a: a3 81 fa ; OPTIONAL[31 (fa Bytes)
017d: 30 81 f7 ; SEQUENCE (f7 Bytes)
0180: 30 69 ; SEQUENCE (69 Bytes)
0184: 06 0a ; OBJECT_ID (a Bytes)
      2b 06 01 04 01 82 bd 64 01 00 ; 1.3.6.1.4.1.40676.1.0 18
018e: 04 5b ; OCTET_STRING (5b Bytes) 19
0190: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ....
01a0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ....
01b0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ....
01c0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ....
01d0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ....
01e0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ....
01eb: 30 12 ; SEQUENCE (12 Bytes)
01ed: 06 0a ; OBJECT_ID (a Bytes)
01ef: 2b 06 01 04 01 82 bd 64 01 01 ; 1.3.6.1.4.1.40676.1.1 20
01f9: 04 04 ; OCTET_STRING (4 Bytes)
01fb: 36 32 34 31 ; 6241 21
01ff: 30 1e ; SEQUENCE (1e Bytes)
0201: 06 0a ; OBJECT_ID (a Bytes)
0203: 2b 06 01 04 01 82 bd 64 01 02 ; 1.3.6.1.4.1.40676.1.2 22
020d: 04 1a ; OCTET_STRING (1a Bytes)
020f: ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff 00 00 ; .... 23
021f: 30 12 ; SEQUENCE (12 Bytes)
0221: 06 0a ; OBJECT_ID (a Bytes)
0223: 2b 06 01 04 01 82 bd 64 01 03 ; 1.3.6.1.4.1.40676.1.3 24
022d: 04 04 ; OCTET_STRING (4 Bytes)
022f: 13 00 08 c0 ; 25
0233: 30 12 ; SEQUENCE (12 Bytes)
0235: 06 0a ; OBJECT_ID (a Bytes)
0237: 2b 06 01 04 01 82 bd 64 01 04 ; 1.3.6.1.4.1.40676.1.4 26
0241: 04 04 ; OCTET_STRING (4 Bytes)
0243: 00 00 00 00 ; 27
0247: 30 2e ; SEQUENCE (2e Bytes)
0249: 06 0a ; OBJECT_ID (a Bytes)
024b: 2b 06 01 04 01 82 bd 64 01 0a ; 1.3.6.1.4.1.40676.1.10 28
0255: 04 20 ; OCTET_STRING (20 Bytes) 29
0257: 00 31 f0 ab f6 6c 0a 63 c5 27 ef 1f 12 8e 5a 20 ; .1..
0267: 8a a8 6c c0 b8 3e 12 ec 19 d0 87 51 1f e0 7c 45 ; ..1.
.>.....Q..!E
  
```

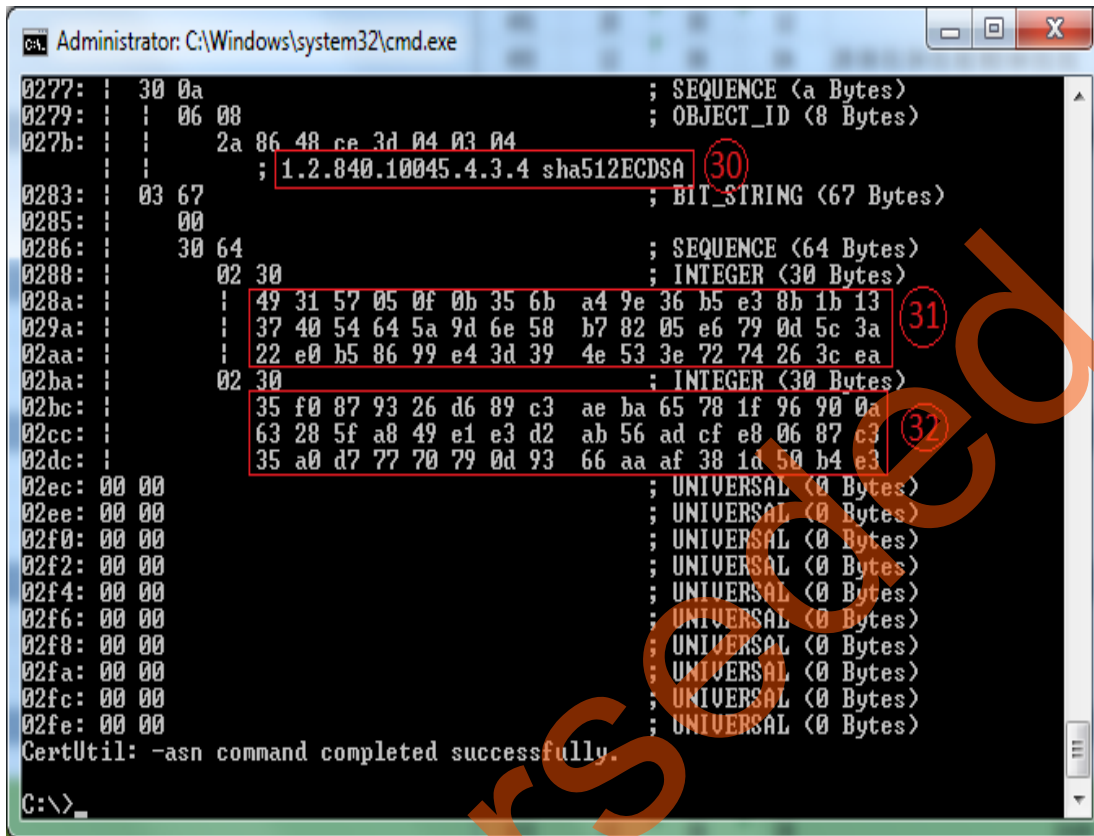
Figure 8 • Device Certificate - Screen 3

The following is the description of labels in Figure 8:

#### Extension Information

18. Object ID 1.3.6.1.4.1.40676.1.0 : **padding**
19. 91 bytes zero padding
20. Object ID 1.3.6.1.4.1.40676.1.1: **Date code**
21. Date code value : **36 32 34 31**
22. Object ID 1.3.6.1.4.1.40676.1.2: **Temp, Speed, and Voltage Grade**
23. Temp, Speed, and Voltage Grade <Generic Field Value>: **ff ff ff ff ff ff ff ff ff ff ff ff ff ff ff 00 00**
24. Object ID 1.3.6.1.4.1.40676.1.3: **Feature Activation**
25. Feature Activation Value: **13 00 08 c0**. This field indicates the features that are available in a particular device. For more information on how to interpret this field values, refer to "[Appendix C: Feature Activation](#)" section on page 21.
26. Object ID 1.3.6.1.4.1.40676.1.4: **Pricing ID**
27. Pricing ID value : **00 00 00 00**
28. Object ID 1.3.6.1.4.1.40676.1.10: **Certificate Validator**. This field is used by the Libero SoC/Flashpro software to validate the Device certificate
29. Certificate Validator Value ( 256 bit validator ): **"00 31 f0 ab f6 6c 0a 63 c5 27 ef 1f 12 8e 5a 20 8a a8 6c c0 b8 3e 12 ec 19 d0 87 51 1f e0 7c 45"**

Figure 9 shows the device certificate - screen 4.



```

Administrator: C:\Windows\system32\cmd.exe
0277: 30 0a ; SEQUENCE (a Bytes)
0279: 06 08 ; OBJECT_ID (8 Bytes)
027b: 2a 86 48 ce 3d 04 03 04 ; 1.2.840.10045.4.3.4 sha512ECDSA (30)
0283: 03 67 ; BIT_STRING (67 Bytes)
0285: 00
0286: 30 64 ; SEQUENCE (64 Bytes)
0288: 02 30 ; INTEGER (30 Bytes)
028a: 49 31 57 05 0f 0b 35 6b a4 9e 36 b5 e3 8b 1b 13 (31)
029a: 37 40 54 64 5a 9d 6e 58 b7 82 05 e6 79 0d 5c 3a
02aa: 22 e0 b5 86 99 e4 3d 39 4e 53 3e 72 74 26 3c ea
02ba: 02 30 ; INTEGER (30 Bytes)
02bc: 35 f0 87 93 26 d6 89 c3 ae ba 65 78 1f 96 90 0a (32)
02cc: 63 28 5f a8 49 e1 e3 d2 ab 56 ad cf e8 06 87 c3
02dc: 35 a0 d7 77 70 79 0d 93 66 aa af 38 1d 50 b4 e3
02ec: 00 00 ; UNIVERSAL (0 Bytes)
02ee: 00 00 ; UNIVERSAL (0 Bytes)
02f0: 00 00 ; UNIVERSAL (0 Bytes)
02f2: 00 00 ; UNIVERSAL (0 Bytes)
02f4: 00 00 ; UNIVERSAL (0 Bytes)
02f6: 00 00 ; UNIVERSAL (0 Bytes)
02f8: 00 00 ; UNIVERSAL (0 Bytes)
02fa: 00 00 ; UNIVERSAL (0 Bytes)
02fc: 00 00 ; UNIVERSAL (0 Bytes)
02fe: 00 00 ; UNIVERSAL (0 Bytes)
CertUtil: -asn command completed successfully.
C:\>

```

Figure 9 • Device Certificate - Screen 4

The following is the description of labels in Figure 9:

#### Certificate Signature Algorithm Information

30. Algorithm: **sha512ECDSA** (Object ID 1.2.840.10045.4.3.4)

**Certificate Signature Information:** As shown in Figure 9, the highlighted circles 31 and 32 display the certificate signature that is stored in the bit string format.

## Conclusion

This application note describes how to implement the device certificate using the system services in the SmartFusion2 SoC FPGAs and view the content of the device certificate.



## Appendix A: Design and Programming Files

Download the design files from the Microsemi SoC Products Group website:

[http://soc.microsemi.com/download/rsc/?f=m2s\\_ac436\\_liberov11p6\\_df](http://soc.microsemi.com/download/rsc/?f=m2s_ac436_liberov11p6_df)

The design file consists of Libero SoC Verilog project, SoftConsole software project, and programming files (\*.stp) for the SmartFusion2 Security Evaluation Kit board. Refer to the Readme.txt file included in the design file for the directory structure and description.

Download the programming files from the Microsemi SoC Products Group website:

[http://soc.microsemi.com/download/rsc/?f=m2s\\_ac436\\_liberov11p6\\_pf](http://soc.microsemi.com/download/rsc/?f=m2s_ac436_liberov11p6_pf)

The programming file consists of STAPL programming file (\*.stp) for the SmartFusion2 Security Evaluation Kit board.

Superseded

## Appendix B: Decoding Device Certificate Using ASN.1 JavaScript Decoder Open Source Tool

ASN.1 JavaScript decoder is a web tool capable of parsing and showing any valid ASN.1 DER or BER data structure as both a tree and a cross-linked hex-dump.

1. Open any standard web browser (for example, Internet Explorer) and enter the following URL in the address bar: <http://lapo.it/asn1js/#>  
ASN.1 online decoder page is displayed as shown in Figure 10.
2. Click **clear**.

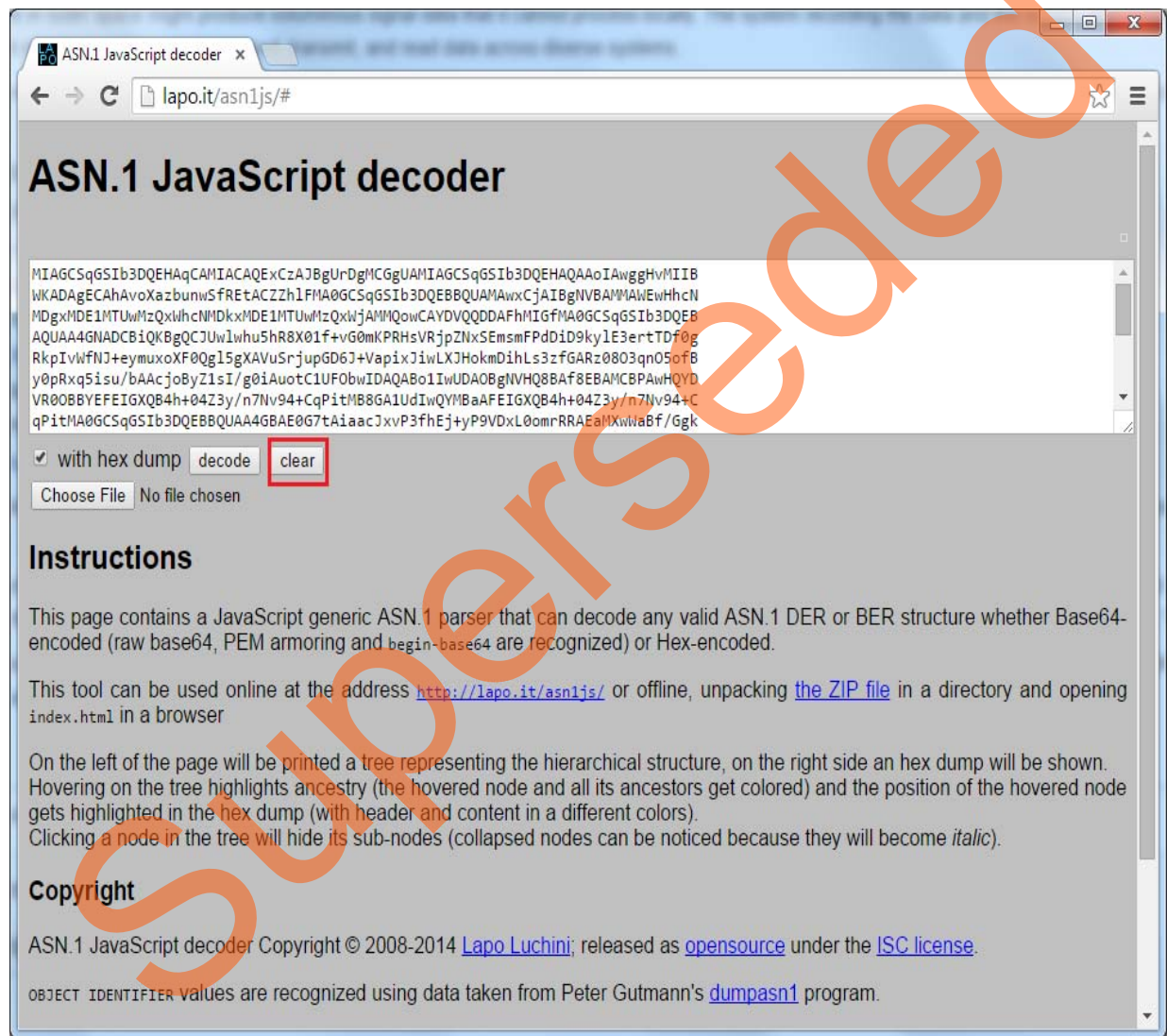


Figure 10 • Online Decoder

3. Copy the 768 Bytes HEX format device certificate from any serial terminal program and paste it in the online decoder as shown in the Figure 11.
4. Click **decode**.

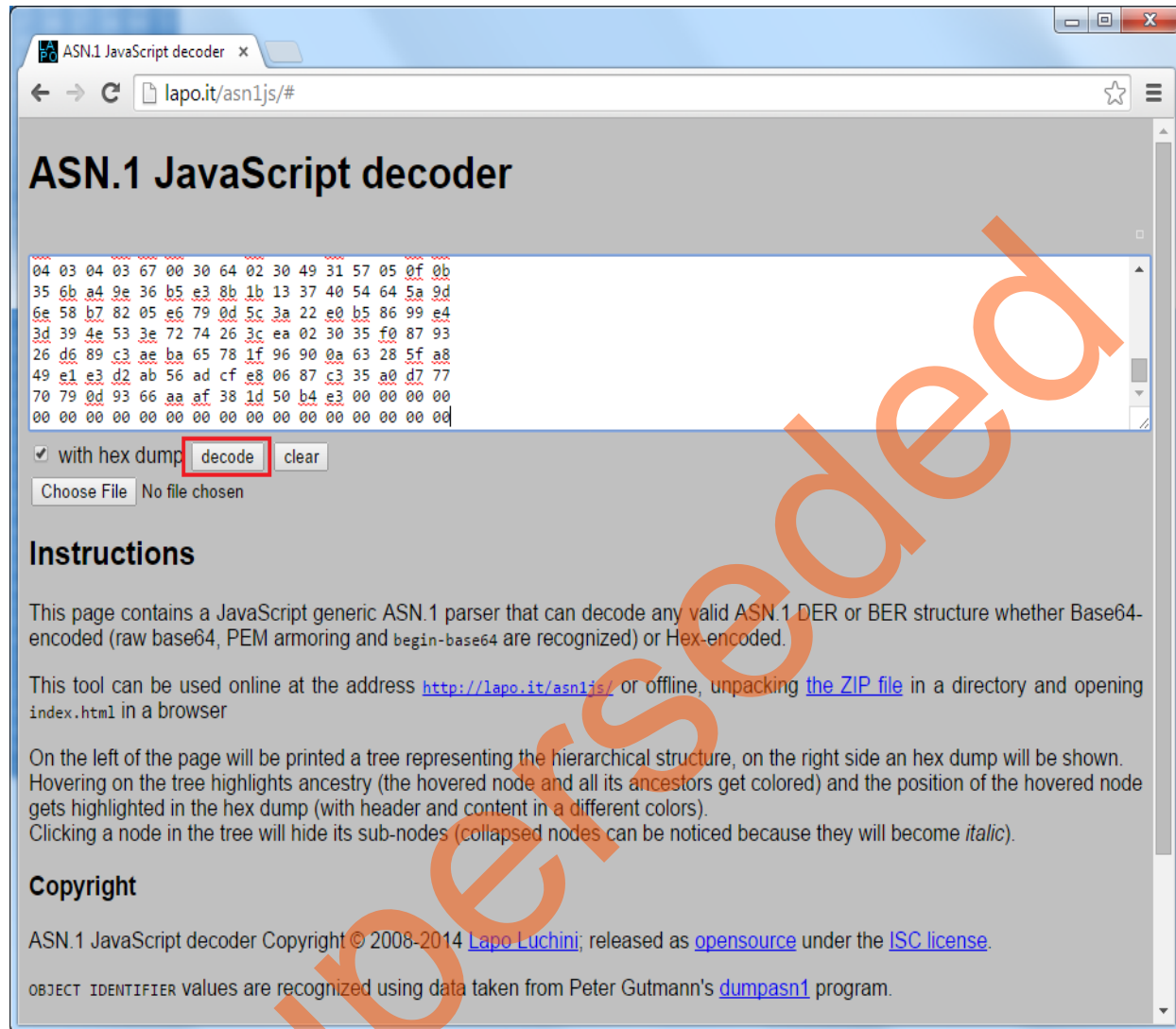


Figure 11 • Online Decoder with Decode Option

[illegible]

### Figure 12 • Decoded Device Certificate

The decoded certificate field values in a tree format are on the left and the HEX dump values are on the right.

For more information about the decoded field values, refer to ["Viewing Fields in Device Certificate" section on page 12](#).

## Appendix C: Feature Activation

The feature activation value provides information about the features that are available in a particular device.

Arrange the feature activation field value in the Byte reverse order. For example, if the field value is: "13 00 08 c0", then write it in the Byte reverse order format "c0 08 00 13" and then convert it to the Binary format" 1100 0000 0000 1000 0000 0000 0001 0011" (MSB to LSB).

The value "1" in a particular position indicates that the feature is available in the device.

Table 7 shows the feature activation array. For example, if the Cortex -M3 processor is enabled within the device, then the array contains "1" in the 0th bit position.

**Note:** Reserved field values and grade field values are programmed with zeros.

**Table 7 • Feature Activation Array**

Bits	Capability	Description	Comments
0	M3	The device features the Cortex-M3	
1	CAN	The device features the MSS CAN peripheral	
3:2	RESERVED		Unused (set to 0)
4	SERDESGEN2	The device features hi-speed SERDES	
15:5	RESERVED		Unused (set to 0)
19	ADVANCED	Device supports advanced security features	
29:20	RESERVED		Unused (set to 0)
30	DENSITY	The device is a high density device	
31	PRODUCTION	The device is a production device	

## List of Changes

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

Revision*	Changes	Page
Revision 2 (October 2015)	Updated the document for Libero SoC v11.6 software release (SAR 71682).	N/A
Revision 1 (February 2015)	Initial release.	N/A

**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.



**Microsemi Corporate Headquarters**  
One Enterprise, Aliso Viejo,  
CA 92656 USA

**Within the USA:** +1 (800) 713-4113  
**Outside the USA:** +1 (949) 380-6100  
**Sales:** +1 (949) 380-6136  
**Fax:** +1 (949) 215-4996

**E-mail:** [sales.support@microsemi.com](mailto:sales.support@microsemi.com)

© 2015 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Ethernet Solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,600 employees globally. Learn more at [www.microsemi.com](http://www.microsemi.com).

Microsemi makes no warranty, representation, or guarantee regarding the information contained herein or the suitability of its products and services for any particular purpose, nor does Microsemi assume any liability whatsoever arising out of the application or use of any product or circuit. The products sold hereunder and any other products sold by Microsemi have been subject to limited testing and should not be used in conjunction with mission-critical equipment or applications. Any performance specifications are believed to be reliable but are not verified, and Buyer must conduct and complete all performance and other testing of the products, alone and together with, or installed in, any end-products. Buyer shall not rely on any data and performance specifications or parameters provided by Microsemi. It is the Buyer's responsibility to independently determine suitability of any products and to test and verify the same. The information provided by Microsemi hereunder is provided "as is, where is" and with all faults, and the entire risk associated with such information is entirely with the Buyer. Microsemi does not grant, explicitly or implicitly, to any party any patent rights, licenses, or any other IP rights, whether with regard to such information itself or anything described by such information. Information provided in this document is proprietary to Microsemi, and Microsemi reserves the right to make any changes to the information in this document or to any products and services at any time without notice.