Using ECC System Service in SmartFusion2 - Libero SoC v11.7

Table of Contents

Purpose

This application note explains how to access an elliptic curve cryptography (ECC) service in the SmartFusion®2 system-on-chip (SoC) field programmable gate array (FPGA) devices.

Introduction

ECC is a public key encryption technique based on the elliptic curve theory. ECC is used to encrypt and is based on a one-way property in which it is easy to perform a calculation, but not feasible to invert the results of the calculation to find the original numbers.

ECC is used:

- In implementing faster, smaller size, and efficient cryptographic keys.
- In digital signatures through elliptic curve digital signature algorithm (ECDSA) and in key exchange through elliptic curve diffie-hellman (ECDH).

ECC feature is available in the larger SmartFusion2 devices: M2S090TS and M2S150TS. ECC has two mathematical services such as: scalar point multiplication and point addition, based on the national institute of standards and technology (NIST) recommended P-384 elliptic curve domain parameters.

In the SmartFusion2 SoC FPGA devices, ECC services are accessible through the system services. The system services are system controller actions initiated by asynchronous events from the ARM® Cortex®-M3 processor or a fabric master in the SmartFusion2 device. ECC is used for data and design security applications.

This application note provides a design example to implement and access the following ECC services:
- ECC point addition
- Scalar point multiplication
- Generate public key

References

The following are the references used:
- *UG0331: SmartFusion2 Microcontroller Subsystem User Guide*
- *UG0450: SmartFusion2 SoC FPGA and IGLOO2 FPGA System Controller User Guide*
- *UG0443: SmartFusion2 SoC FPGA and IGLOO2 FPGA Security and Reliability User Guide*

System Controller Block in SmartFusion2 Device

The ECC services provide access to the system controller's ECC core. The ECC core block is accessed through the communication block (COMM_BLK).

There are two COMM_BLK instances:
- Located in the MSS
- Located in the system controller

The COMM_BLK consists an APB interface, eight byte transmits FIFO, and eight byte receives FIFO. The COMM_BLK provides a bi-directional message passing facility between the microcontroller subsystem (MSS) and system controller.

The ECC system services are 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 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 are generated based on the selected command.

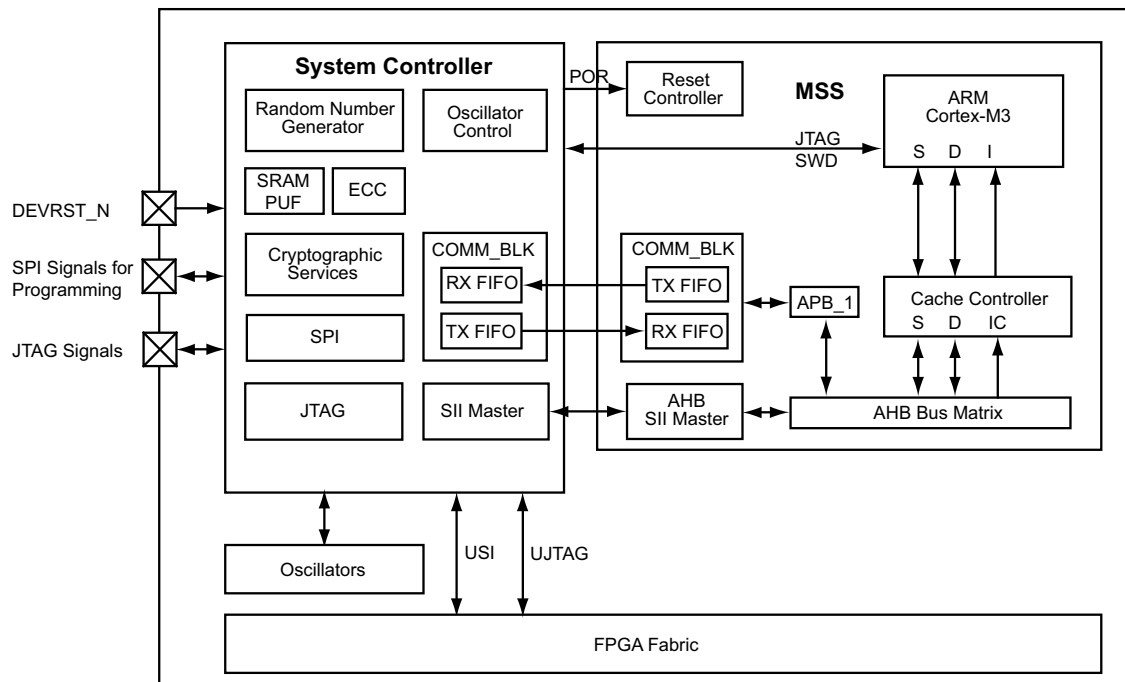Figure 1 shows the system controller block in the SmartFusion2 device.

Figure 1 • **System Controller Block in the SmartFusion2 Device**

For more information about system controller. refer to the *UG0450: SmartFusion2 SoC FPGA 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*.

ECC and Services

An elliptic curve is defined as the set of points (X, Y), which satisfy an elliptic curve equation of the form:

$Y^2 = X^3 + aX + b;$

EQ 1

Where a and b are the elements of a finite field with p^n (n = 192, 224, 256, or 384), where p is a prime number larger than 3. An example of the Elliptical curve is shown in Figure 2.
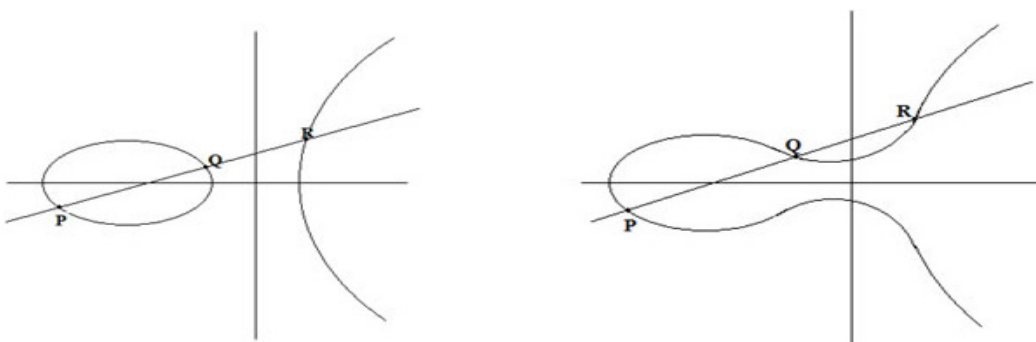

Figure 2 • **Elliptical Curves**

If X^3+aX+b contains no repeated factors, or equivalently if $4a^3+27b^2$ is not 0 (zero), then the elliptic curve $Y^2 = X^3+aX+b$ can be used to form a group.

The elliptic curve groups are additive groups.

This application note uses NIST recommended P-384 curve to implement the ECC system services. The domain parameters for the curve P-384 are the prime, a, b, and base point G. The recommended values for these parameters are:

The Prime $P^{384} = 2^{384} - 2^{128} - 2^{96} + 2^{32} - 1$:

In hexadecimal form:

```
p384 = ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff fffffffe ffffffff
00000000 00000000 ffffffff
```

The parameter a in hexadecimal form:

```
a = ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff fffffffe ffffffff
00000000 00000000 fffffffc
```

The parameter b in hexadecimal form:

```
b = b3312fa7 e23ee7e4 988e056b e3f82d19 181d9c6e fe814112
0314088f 5013875a c656398d 8a2ed19d 2a85c8ed d3ec2aef
```

The base point G in hexadecimal form:

```
Gx = aa87ca22 be8b0537 8eb1c71e f320ad74 6e1d3b62 8ba79b98
59f741e0 82542a38 5502f25d bf55296c 3a545e38 72760aB7
Gy = 3617de4a 96262c6f 5d9e98bf 9292dc29 f8f41dbd 289a147c
e9da3113 b5f0b8c0 0a60b1ce 1d7e819d 7a431d7c 90ea0e5F
```

The following ECC services are implemented in this application note:

Point Addition

The elliptic point addition service adds two points according to the definition of elliptic curve point addition. The inputs are two points (x, y), each lying on the P-384 curve and the resulting point (x, y), which is guaranteed to be on the curve.

Point addition service computes the results as the following:

$\quad$ R(Rx, Ry) = P(Px, Py) + Q(Qx, Qy)

EQ 2

P and Q are two 384 bit input points (P_x, P_y) and (Q_x, Q_y) on the P-384 elliptic curve.

To add the points P and Q, a line is drawn through the two points. This line intersects the elliptic curve in exactly one more point, call -R. The point -R is reflected in the x-axis to the point R. The law for addition in an elliptic curve group is P + Q = R, where R is the result of addition, as shown in Figure 3.
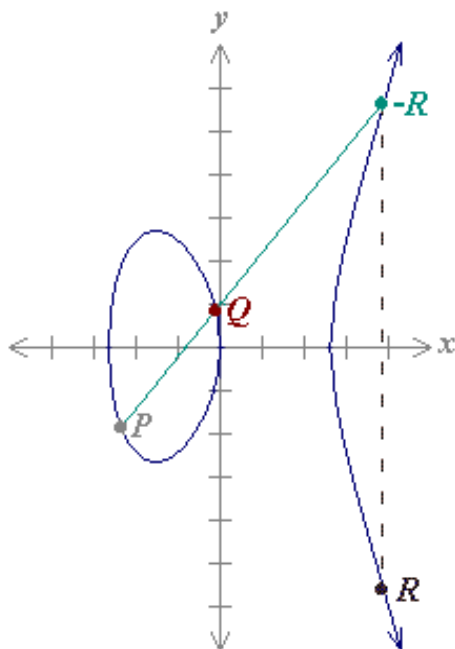

Figure 3 • **Addition on an Elliptic Curve**

The example input points P and Q are provided in this application note for P-384 elliptic curve, you can enter any P and Q points on the curve to perform the addition operation.

Note: Input points are not tested for validity by the ECC. If an input point is not a valid point on the P-384 curve (including the point at infinity), then the result will be undefined and no warning message are displayed. You can find NIST recommended input test vectors at the following location:

http://csrc.nist.gov/groups/STM/cavp/documents/components/ecccdhtestvectors.zip.

Table 1 shows the command value to perform the ECC Point Addition in the SmartFusion2 device.

Table 1 • **ECC Point Addition**

System Service Name	Command Value	Response Status
Point Addition	17	0: Success completion
		127: HRESP error occurred during MSS transfer
		253: License not available in device
		254: Service disabled by factory security
		255: Service disabled by user security

Scalar Point Multiplication

The scalar point multiplication service multiplies the scalar point with the point on the P-384 curve.

The multiplication service computes the result as per the standard:

Q = d x P

d is the scalar input (384 bit) and P is the 384 bit point (P_x, P_y) on the curve.

Table 2 shows the command value to perform the ECC point multiplication.

Table 2 • **ECC Point Multiplication**

System Service Name	Command Value	Response Status
Point Multiplication	16	0: Success completion
		127: HRESP error occurred during MSS transfer
		253: License not available in device
		254: Service disabled by factory security
		255: Service disabled by user security

Generating a Public Key

The point multiplication is used for generating a public key with provided private key (384 bit integer). The domain parameter G, specifies the base point on the curve P-384 to use. This domain base point is built into hardware accelerator, which has a special form of the scalar multiplication command that uses base point without the user having to enter.

The generated public key is used further to encrypt the data or an application. The data or the application can be decrypted using the private key.

Design Requirements

Table 3 shows the design requirements.

Table 3 • **Design Requirements**

Design Requirements	Description
Hardware Requirements	
SmartFusion2 Security Evaluation Kit: • 12 V adapter • FlashPro4 programmer • USB A to Mini-B cable	Rev D or later
Host PC or Laptop	Any 64-bit Windows Operating System
Software Requirements	
Libero® System-on-Chip (SoC)	v11.7
SoftConsole	v3.4 SP1*
FlashPro Programming Software	v11.7
USB to UART drivers	–
*Note: * For this application, SoftConsole v4.0 is used, see the TU0546: SoftConsole v4.0 and Libero SoC v11.7 Tutorial.*	

Design Description

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

The design example consists:

- RC oscillator
- Fabric CCC
- CORERESET
- MSS

The fabric PLL provides the base clock for the MSS. The system services are run using various C routine in the MSS, as shown in the following sections. In addition, a universal asynchronous receiver/transmitter (UART1) in the MSS displays the operation of the ECC system service.

Hardware Implementation

Figure 4 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. The 100 MHz clock is used as the base clock for the MSS.

The MMUART_1 signals are used to communicate with the serial terminal program.
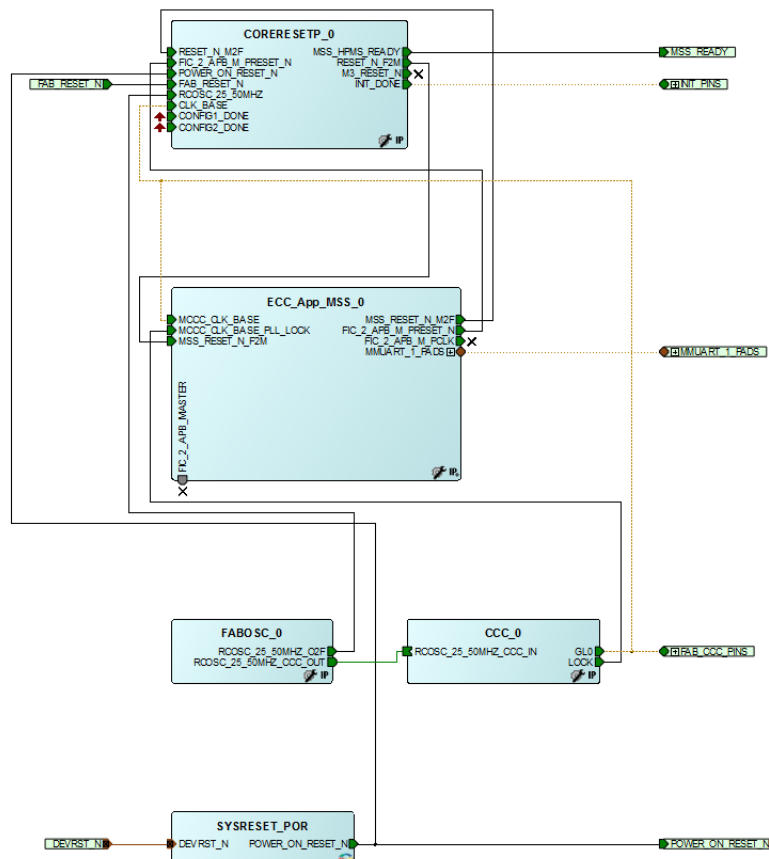
Figure 4 • **Block Diagram of SmartFusion2 ECC Design Example**

Software Implementation

The software design example performs the following operations:

- ECC point addition
- Scalar point multiplication
- Public key generation

Firmware Drivers

The following firmware drivers are used in this application:

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

List of APIs

The following APIs in Table 4 are used in software design to access the ECC Services.

Table 4 • **APIs to Access the ECC System Services**

API	Description
MSS_SYS_ecc_point_addition()	Point addition
MSS_SYS_ecc_point_multiplication()	Scalar point multiplication
MSS_SYS_ecc_get_base_point()	To get base point

Setting Up the Design

The following steps describe how to set up the design:

1. Plug the FlashPro4 ribbon cable to the **J5** connector (JTAG Programming Header) on the SmartFusion2 Security Evaluation Kit board.
2. Connect the mini USB cable between the FlashPro4 and the USB port of the host PC.
3. Connect the power supply to the **J6** connector.
4. Connect the Host PC to the **J18** connector using the USB min-B cable.
5. Ensure that the USB to UART bridge drivers are automatically detected. This can be verified in the Device Manager.

Figure 5 shows the example device manager window. If USB to UART bridge drivers are not installed, download and install the drivers from:

www.microsemi.com/soc/documents/CDM_2.08.24_WHQL_Certified.zip
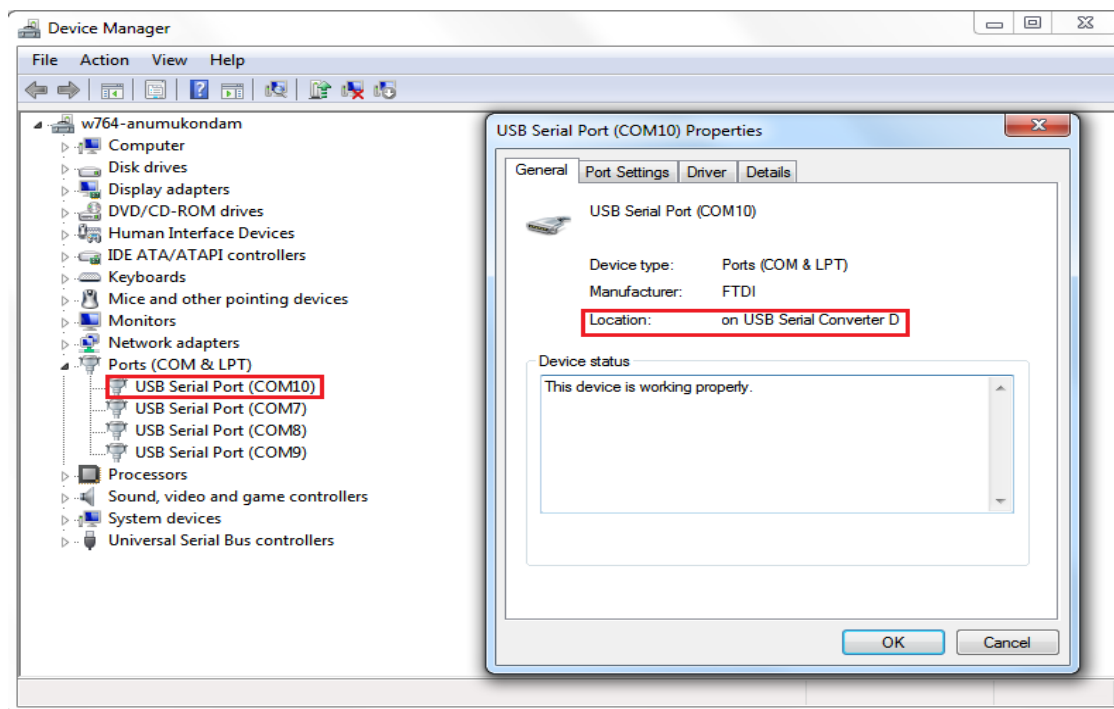
***Figure 5 •* Device Manager Window**

6. Connect the jumpers on the SmartFusion2 Security Evaluation Kit, as shown in Table 5.

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

Table 5 • **SmartFusion2 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 6 shows the board setup for running the ECC services design on the SmartFusion2 Security Evaluation Kit.

Figure 6 • **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 or HyperTerminal 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 the FlashPro4 programmer. Refer to "Appendix: Design and Programming Files" on page 21" for more information.

4. After programming, HyperTerminal displays a message to run the ECC Services as shown in Figure 7.
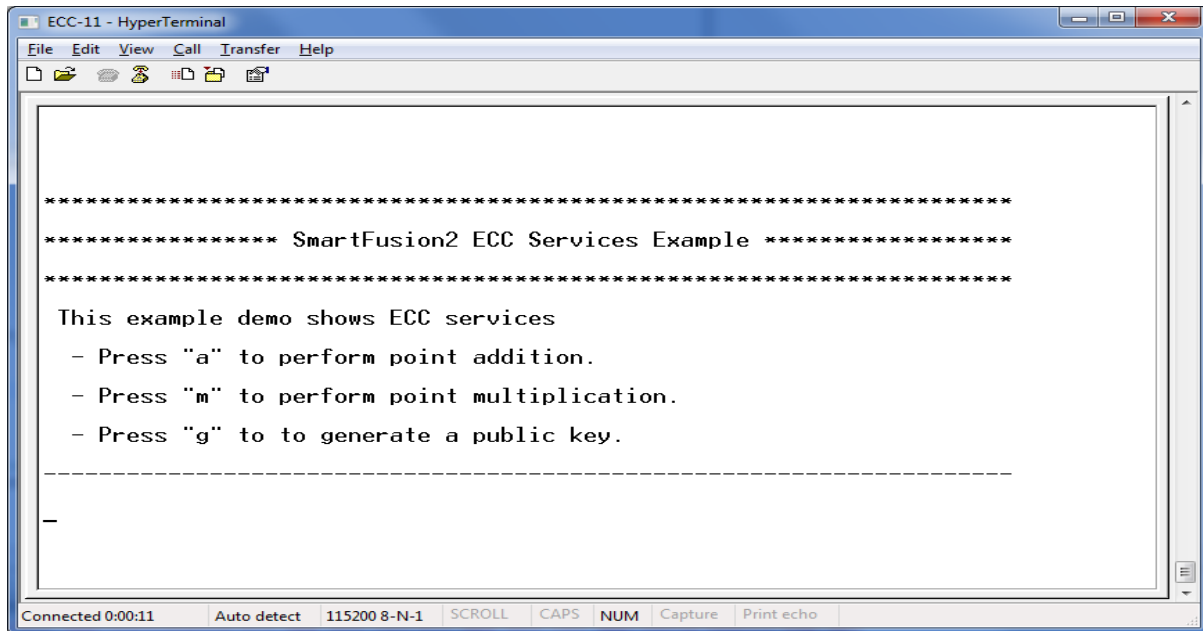
Figure 7 • **Welcome Message**

1. Enter **a** to perform point addition. HyperTerminal displays "Enter the X Coordinate of input point P". Enter the following X coordinate of P in HyperTerminal or copy each line.

2. Right-click HyperTerminal and select **Paste to Host** to paste the coordinate each line once as shown in Figure 8. Do not enter after copying each line.
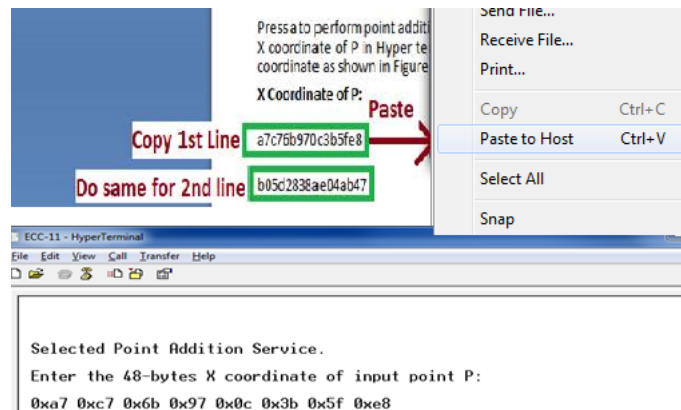
Figure 8 • **Pasting Coordinate**

X Coordinate of P, as shown in Figure 9.

```
a7c76b970c3b5fe8
b05d2838ae04ab47
697b9eaf52e76459
2efda27fe7513272
734466b400091adb
f2d68c58e0c50066
```
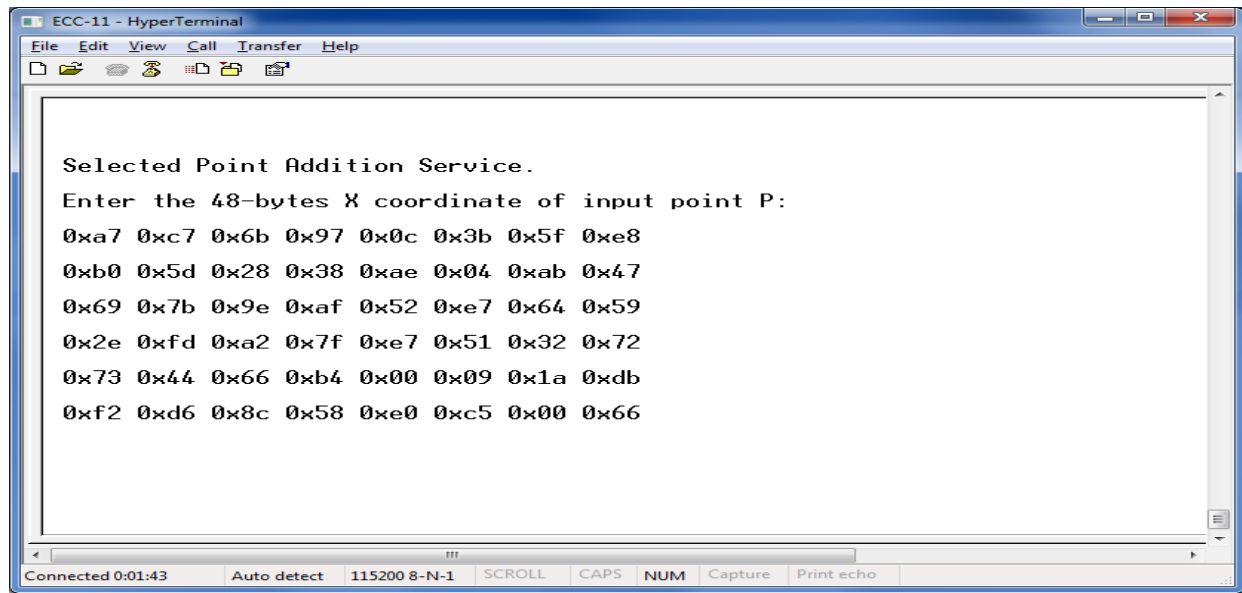

Figure 9 • **Point Addition Input**

3. Enter the Y Coordinate of P (Py), X and Y coordinates of Q (Qx, Qy) as shown in Figure 10 on page 13, Figure 11 on page 13, and Figure 12 on page 14.

Y Coordinate of P:

```
ac68f19f2e1cb879
aed43a9969b91a08
39c4c38a49749b66
1efedf243451915e
d0905a32b060992b
468c64766fc8437a
```

Figure 10 shows the Y Coordinate of P.
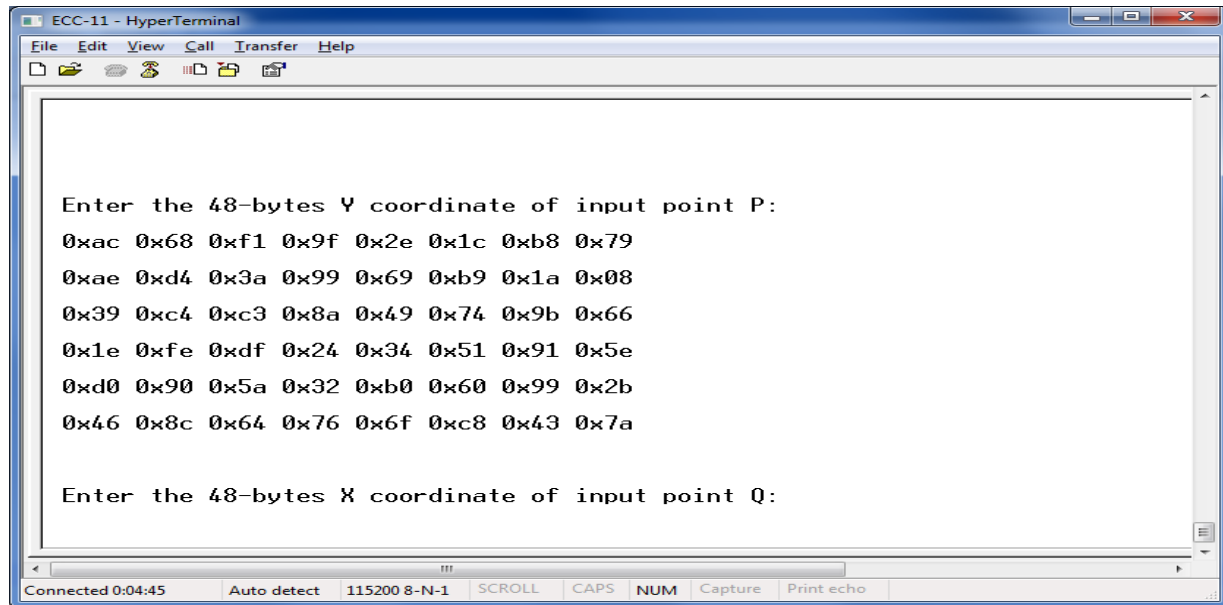
Figure 10 • **Point Addition Input P**

X Coordinate of Q:
```
9803807f2f6d2fd9
66cdd0290bd410c0
190352fbec7ff624
7de1302df86f25d3
4fe4a97bef60cff5
48355c015dbb3e5f
```
Figure 11 shows the X Coordinate of Q.
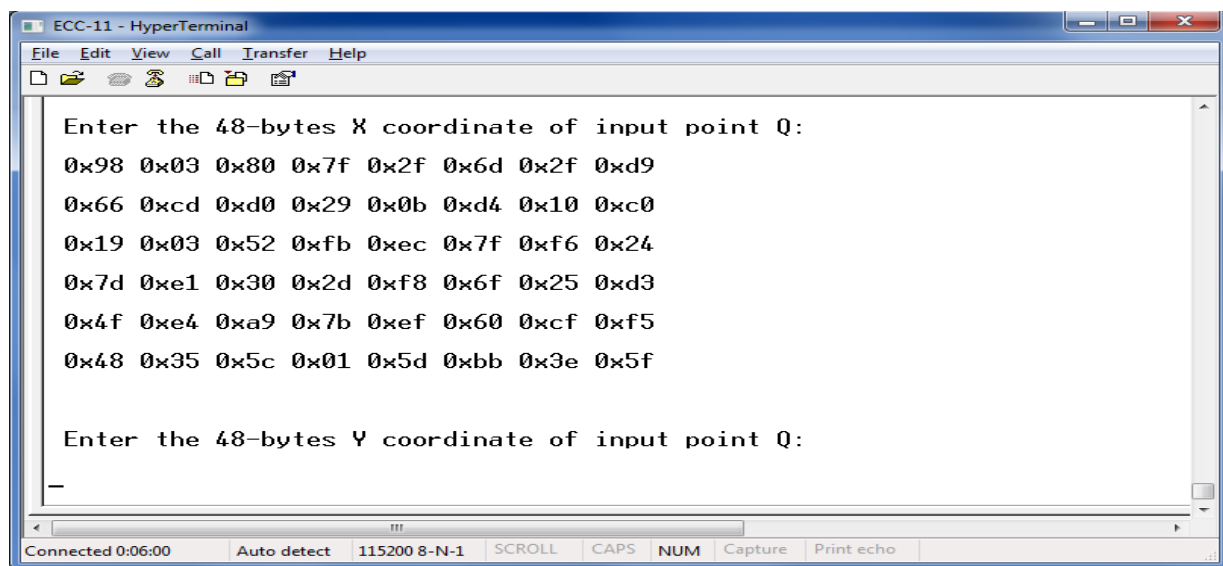
Figure 11 • **Point Addition Input**

Y Coordinate of Q:

```
ba26ca69ec2f5b5d
9dad20cc9da71138
3a9dbe34ea3fa5a2
af75b46502629ad5
4dd8b7d73a8abb06
a3a3be47d650cc99
```

Figure 12 shows the Y Coordinate of Q.
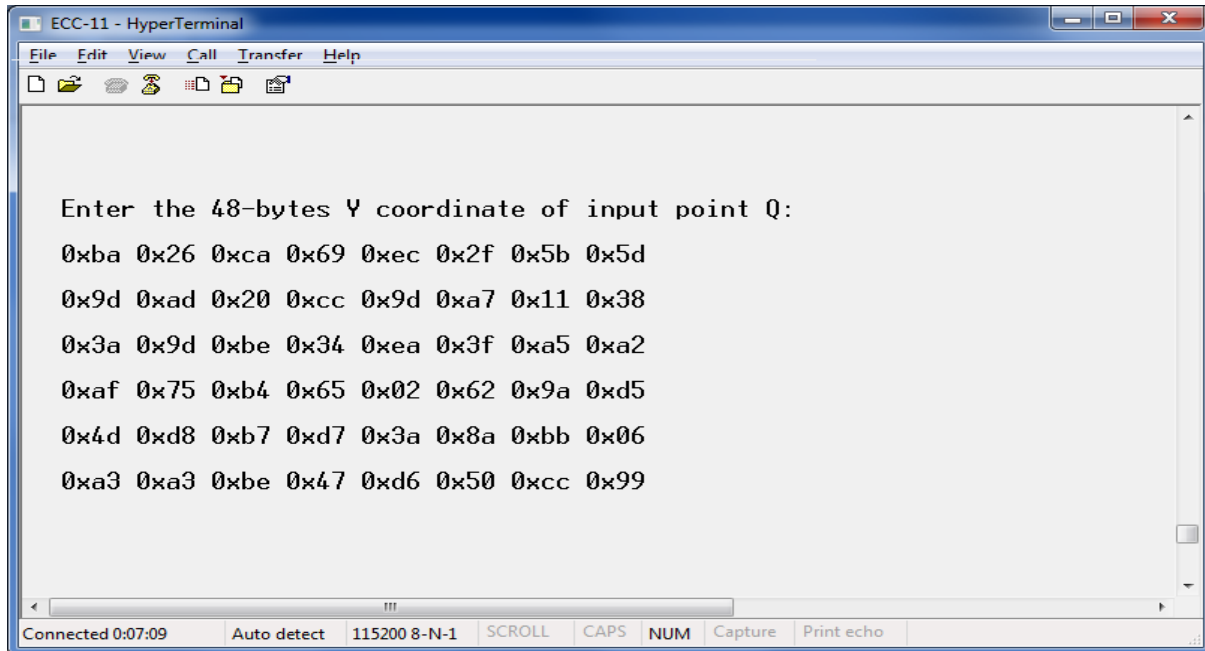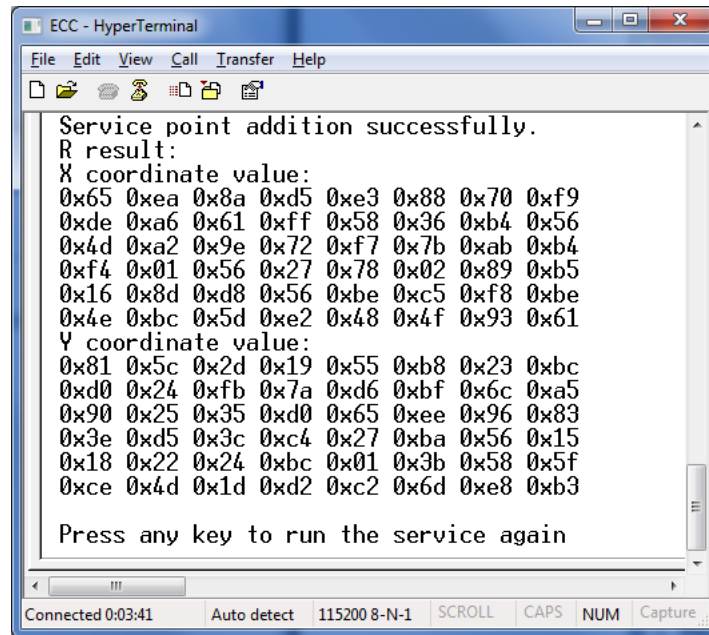
Figure 12 • **Point Addition Input**

4. Point addition result is displayed on HyperTerminal, as shown in Figure 13.

Figure 13 • **Point Addition Result**

5. Enter **m** to perform point multiplication and enter the 384 bit input scalar D, as shown in Figure 14 on page 16.

Input Scalar D:

D =

```
b7847b54eb40d602
dbe18b5386ac9a99
e0a584e3d01e2ef5
a5700be26e7076ca
c390d423a0033b07
e4ecbe709001fc39
```

```
ECC-11 - HyperTerminal
File  Edit  View  Call  Transfer  Help

  Selected Point Multiplication Service.

  Enter the 384 bit input scaler d:

  0xb7 0x84 0x7b 0x54 0xeb 0x40 0xd6 0x02

  0xdb 0xe1 0x8b 0x53 0x86 0xac 0x9a 0x99

  0xe0 0xa5 0x84 0xe3 0xd0 0x1e 0x2e 0xf5

  0xa5 0x70 0x0b 0xe2 0x6e 0x70 0x76 0xca

  0xc3 0x90 0xd4 0x23 0xa0 0x03 0x3b 0x07

  0xe4 0xec 0xbe 0x70 0x90 0x01 0xfc 0x39



  Enter the 48-bytes X coordinate of input point P:

  _

Connected 0:09:59   Auto detect   115200 8-N-1   SCROLL   CAPS   NUM   Capture   Print echo
```
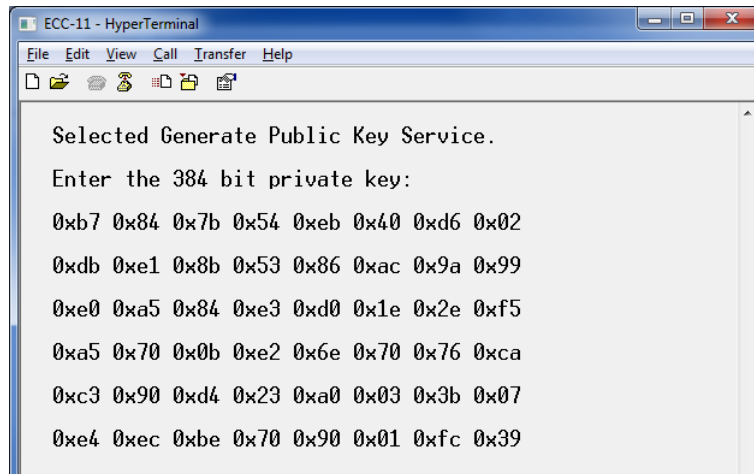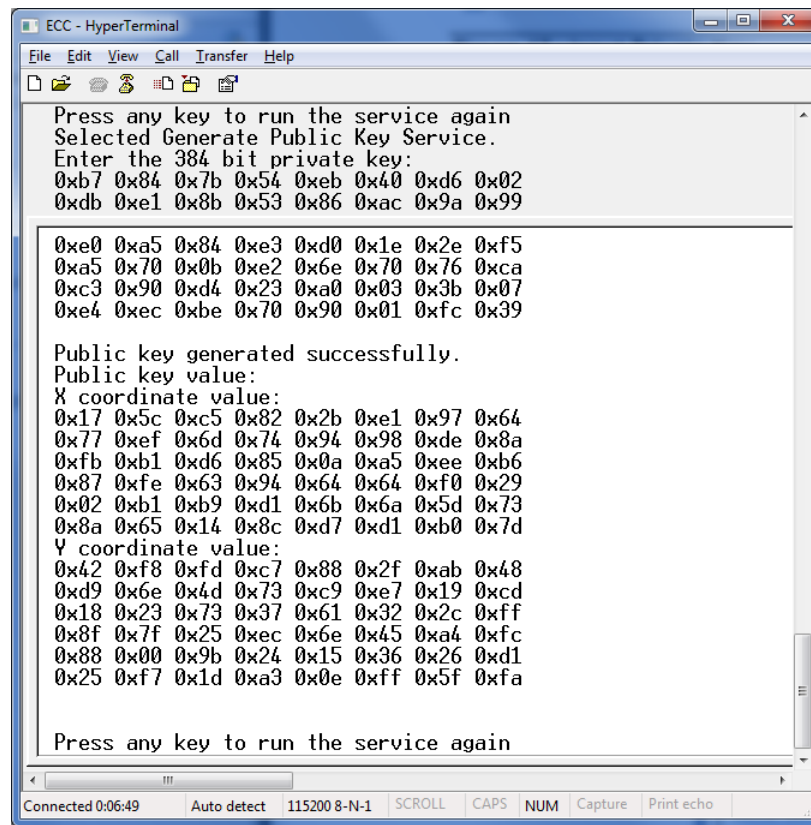
Figure 14 • **Point Multiplication**

6. Enter 48 byte X, Y coordinates of Input point P (Px, Py) as shown in and .

X and Y coordinates of the point P on the elliptic curve are:

X Coordinate of P:

```
8c19da9344ffda65
b72b2f83da40d806
40da85a220a0ccea
03788ef0b395b2ed
9a6a7cc5c2c60a80
54268c9d33dda7fe
```

Y Coordinate of P:

```
a86d11a4ab265d8d
c4aa0d86e16bdbdb
8c78914f4ef6aef0
9c382b689460eacb
363fc795ec1914c0
276a7b75b562fe6b
```

Figure 15 shows the X Coordinate of P.


```
0xe4 0xec 0xbe 0x70 0x90 0x01 0xfc 0x39


Enter the 48-bytes X coordinate of input point P:

0x8c 0x19 0xda 0x93 0x44 0xff 0xda 0x65

0xb7 0x2b 0x2f 0x83 0xda 0x40 0xd8 0x06

0x40 0xda 0x85 0xa2 0x20 0xa0 0xcc 0xea

0x03 0x78 0x8e 0xf0 0xb3 0x95 0xb2 0xed

0x9a 0x6a 0x7c 0xc5 0xc2 0xc6 0x0a 0x80

0x54 0x26 0x8c 0x9d 0x33 0xdd 0xa7 0xfe


Enter the 48-bytes Y coordinate of input point P:
```

Figure 15 • **X Input of Point P**

Figure 16 shows the Y Coordinate of P.


```
0x40 0xda 0x85 0xa2 0x20 0xa0 0xcc 0xea

0x03 0x78 0x8e 0xf0 0xb3 0x95 0xb2 0xed

0x9a 0x6a 0x7c 0xc5 0xc2 0xc6 0x0a 0x80

0x54 0x26 0x8c 0x9d 0x33 0xdd 0xa7 0xfe


Enter the 48-bytes Y coordinate of input point P:

0xa8 0x6d 0x11 0xa4 0xab 0x26 0x5d 0x8d

0xc4 0xaa 0x0d 0x86 0xe1 0x6b 0xdb 0xdb

0x8c 0x78 0x91 0x4f 0x4e 0xf6 0xae 0xf0

0x9c 0x38 0x2b 0x68 0x94 0x60 0xea 0xcb

0x36 0x3f 0xc7 0x95 0xec 0x19 0x14 0xc0

0x27 0x6a 0x7b 0x75 0xb5 0x62 0xfe 0x6b
```

Figure 16 • **Y Input of Point P**

7. Point Multiplication result is displayed on the HyperTerminal, as shown in Figure 17.

Figure 17 • **Point Multiplication Result**

8. Enter **g** to generate a public key and enter 384 bit private key as shown in Figure 18.

   Private Key:

   ```
   b7847b54eb40d602
   dbe18b5386ac9a99
   e0a584e3d01e2ef5
   a5700be26e7076ca
   c390d423a0033b07
   e4ecbe709001fc39
   ```


Figure 18 • **Input Private Key**

This service uses the internal 384 bit base point and performs the point multiplication with a private key to generate the public key. Generated public key X, Y coordinates are displayed on the HyperTerminal window as shown in Figure 19.

Figure 19 • **X, Y Coordinates of Public Key**

Conclusion

This application note explains how to access the ECC services in the SmartFusion2 SoC FPGAs. It describes how to perform the ECC point addition, scalar point multiplication, and public key generation.

Appendix: Design and Programming Files

Download the design files from the Microsemi website:

http://soc.microsemi.com/download/rsc/?f=m2s_ac435_liberov11p7_df

The design files consist a Libero Verilog project and programming files (*.stp) for the SmartFusion2 Security Evaluation Kit.

Refer to the `readme.txt` file included in the design files for the directory structure and description.

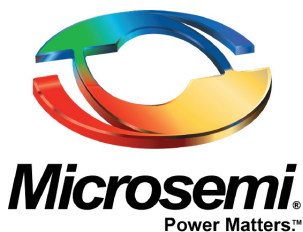Download the programming files (*.stp) in release mode from the Microsemi website:

http://soc.microsemi.com/download/rsc/?f=m2s_ac435_liberov11p7_pf

The programming .zip file contains the STAPL programming file (*.stp) for the SmartFusion2 Security Evaluation Kit.

List of Changes

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

Date*	Changes	Page
Revision 4 (April 2016)	Updated the document for Libero v11.7 software release (SAR 77403).	NA
Revision 3 (October, 2015)	Updated the document for Libero v11.6 software release (SAR 71998).	NA
Revision 2 (February, 2015)	Updated the document for Libero v11.5 software release (SAR 64496).	NA
Revision 1 (October, 2014)	Initial release.	N/A

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; Enterprise Storage and Communication solutions,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 4,800 employees globally. Learn more at **www.microsemi.com**.

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

51900303-4/04.16