MaxCore™ MC3000 Platform Software

Installation and Use

P/N: 6806800U97B

March 2020

SMART™ Embedded Computing

Trademarks

The stylized "S" and "SMART" is a registered trademark of SMART Modular Technologies, Inc. and “SMART Embedded Computing” and the SMART Embedded Computing logo are trademarks of SMART Modular Technologies, Inc. All other names and logos referred to are trade names, trademarks, or registered trademarks of their respective owners. These materials are provided by SMART Embedded Computing as a service to its customers and may be used for informational purposes only.

Disclaimer*

SMART Embedded Computing (SMART EC) assumes no responsibility for errors or omissions in these materials. **These materials are provided "AS IS" without warranty of any kind, either expressed or implied, including but not limited to, the implied warranties of merchantability, fitness for a particular purpose, or non-infringement.** SMART EC further does not warrant the accuracy or completeness of the information, text, graphics, links or other items contained within these materials. SMART EC shall not be liable for any special, indirect, incidental, or consequential damages, including without limitation, lost revenues or lost profits, which may result from the use of these materials. SMART EC may make changes to these materials, or to the products described therein, at any time without notice. SMART EC makes no commitment to update the information contained within these materials.

Electronic versions of this material may be read online, downloaded for personal use, or referenced in another document as a URL to a SMART EC website. The text itself may not be published commercially in print or electronic form, edited, translated, or otherwise altered without the permission of SMART EC.

It is possible that this publication may contain reference to or information about SMART EC products, programming, or services that are not available in your country. Such references or information must not be construed to mean that SMART EC intends to announce such SMART EC products, programming, or services in your country.

Limited and Restricted Rights Legend

If the documentation contained herein is supplied, directly or indirectly, to the U.S. Government, the following notice shall apply unless otherwise agreed to in writing by SMART Embedded Computing.

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data clause at DFARS 252.227-7013 (Nov. 1995) and of the Rights in Noncommercial Computer Software and Documentation clause at DFARS 252.227-7014 (Jun. 1995).

SMART Embedded Computing, Inc.

2900 S. Diablo Way, Suite 190

Tempe, Arizona 85282

USA

*For full legal terms and conditions, visit *www.smartembedded.com/ec/legal*

Table of Contents

Table of Contents

List of Figures

List of Figures

List of Tables

List of Tables

About this Manual

Overview of Contents

This manual provides information on how to install MaxCore™ MC3000 Platform Software on SMART Embedded Computing MaxCore PCI Express (PCIe) cards. This manual contains the following chapters and appendices.

Chapter 1, Introduction on page 17 provides an overview of this manual.

Chapter 2, Software Installation on page 19 provides procedures on how to create a MaxCore MC3000 ISO, different types of booting options, and how to install the software on MaxCore PCIe cards.

Chapter 3, Accessing the MaxCore MC3000 Platform on page 33 provides brief information about various types of interfaces to access MaxCore MC3000 Platform.

Chapter 4, Managing and Configuring MaxCore Platform on page 39 provides additional information that you need to know while working with MaxCore MC3000 Platform.

Chapter 5, Linux Command-line Utilities on page 57 provides additional information about Linux command line utilities.

Appendix A, Troubleshooting and FAQ on page 65 provides a set of troubleshooting tips and frequently asked questions that are useful while working with MaxCore MC3000 Platform.

Appendix B, Related Documentation on page 71 provides the list of the relevant manuals that you may need to access while working with MaxCore MC3000 Platform.

Abbreviations

The following abbreviations are used in this manual.

Abbreviation	Definition
aCPU	applicationCPU (also referred to as applicationHost)
BCSIM	Blade Common System Information Model
BIOS	Basic Input/Output System
BMC	Baseboard Management Controller
CLI	Command Line Interface
CPLD	Complex Programmable Logic Device
CPU	Central Processing Unit
CSIM	Common System Information Model

Abbreviation	Definition
DHCP	Dynamic Host Configuration Protocol
FCU	Firmware Command-line Utility
FRU	Field Replaceable Unit
GUI	Graphical User Interface
IP	Internet Protocol
IPMI	Intelligent Platform Management Interface
ISO	International Organization for Standardization
JRE	Java Runtime Environment
LAN	Local Area Network
LCU	Log Collection Utility
MAC	Media Access Control
mCPU	Management CPU
MIB	Management Information Base
NVMe	Non-volatile Memory Express
NTP	Network Time Protocol
OS	Operating System
OTG	On-the-Go
PCIe or PCIE	PCI Express
PEP	PCIe End Point
PF	Primary Function
RADIUS	Remote Authentication Dial-In User Service
RRC	Red Rock Canyon
RMCP	Remote Management and Control Protocol
SATA	Serial ATA
SMAN	Service Manager
SOL	Serial over LAN
SSH	Secure Shell
SSF	System Services Framework
SNMP	Simple Network Management Interface
TCP	Transmission Control Protocol

Abbreviation	Definition
TCP/IP	Transmission Control Protocol/Internet Protocol
TFTP	Trivial File Transfer Protocol
TL	Transport Layer
TLS	Transport Layer Server
UDP	User Datagram Protocol
UDS	Unix Domain Socket
USB	Universal Serial Bus
VF	Virtual Function
VGA	Video Graphics Array
VLAN	Virtual Local Area Network
XML	Extensible Markup Language

Conventions

The table below describes the conventions used throughout this manual.

Notation	Description
0x00000000	Typical notation for hexadecimal numbers (digits are 0 through F), for example used for addresses and offsets
0b0000	Same for binary numbers (digits are 0 and 1)
bold	Used to emphasize a word
`Screen`	Used for on-screen output and code related elements or commands. `Sample of Programming used in a table (9pt)`
`Courier + Bold`	Used to characterize user input and to separate it from system output
Reference	Used for references and for table and figure descriptions
File > Exit	Notation for selecting a submenu
<text>	Notation for variables and keys
[text]	Notation for software buttons to click on the screen and parameter description
...	Repeated item for example node 1, node 2, ..., node 12

Notation	Description
. . .	Omission of information from example/command that is not necessary at the time
..	Ranges, for example: 0..4 means one of the integers 0,1,2,3, and 4 (used in registers)
|	Logical OR
	Indicates a hazardous situation which, if not avoided, could result in death or serious injury
	Indicates a hazardous situation which, if not avoided, may result in minor or moderate injury
	Indicates a property damage message
	Indicates a hot surface that could result in moderate or serious injury
	Indicates an electrical situation that could result in moderate injury or death
	Indicates that when working in an ESD environment care should be taken to use proper ESD practices
	No danger encountered, pay attention to important information

Summary of Changes

Part Number	Date	Description
6806800U97B	March 2020	Rebranded to SMART Embedded Computing template. Updated list of acronyms; minor grammatical updates and clarifications made throughout document. Section 2.5 Upgrading Shelf using System Update was added per work item 7349.
6806800U97A	January 2018	Initial version.

Introduction

1.1 Overview

This manual describes the installation and use of the MaxCore™ MC3000 Platform Software on SMART Embedded Computing MaxCore™ PCI Express (PCIe) cards. This software contains board utilities and the System Services Framework (SSF) platform management interface software.

SSF is the primary interface to configure and manage the MaxCore platform. Certain command-line utilities are also documented in this manual. These details are provided in *Chapter 5, Linux Command-line Utilities on page 57*.

1.2 Terminology

This section provides explanation of commonly used terminology in a MaxCore system.

shelfHost

shelfHost is a central management entity in a MaxCore system with many CPUs. It manages the MaxCore infrastructure, such as power supplies, fans, and USB/SATA/PF/VF assignments. The term mCPU is also used in this manual.

applicationHost

An applicationHost processes the data it receives through the network functions assigned to it by the shelfHost. The term applicationCPU (aCPU) is also used in this manual.

networkCPU

This is the CPU on a PCIE-9205 card. It runs the switch management software with a user interface provided by the SSF core software on the shelfHost. The networkCPU may also run user applications of any kind.The term networkHost is also used in this manual.

systemHost

The systemHost can manage a single MaxCore system or a stack of multiple MaxCore systems, referred to as *SYSTEM* in the SSF terminology. The systemHost can be located on any CPU within the SSF network. It can also be a third-party server or a PC.

Software Installation

2.1 Overview

This chapter describes different installation methods to install the MaxCore™ MC3000 Platform Software on MaxCore PCIe cards.

2.2 Software Content

The MaxCore platform software is delivered as an ISO image `MC3K-COMPLETE-ISO_<Version number>.iso`. This ISO image contains the Linux operating system (OS), Linux command-line utilities, SSF, and other necessary files to create a bootable USB or PXE boot image to boot PCIe cards and perform disk installation.

This software can be installed on the shelfHost, applicationHost, and networkHost. On the shelfHost, this software has highly functional content with multiple internal dependencies. The purpose is to manage the MaxCore system and to provide graphical user interfaces. Packages may be added to the installed OS, but pre-installed packages should not be removed or updated independently.

NOTICE

Do not run `yum` update on the shelfHost.

The applicationCPUs (aCPUs) are primarily bare metal devices and the user can install any kind of OS. The SMART Embedded Computing package is also designed to be installed on the aCPUs and contains the following additional services:

1. A daemon which reports the CPU temperature to the MaxCore fan controller. The controller will adjust the fan speed accordingly.
2. An agent which connects SSF to the OS of the aCPU. Based on this, the SSF agent offers various services.
3. A method to exchange parameters between shelfHost and aCPU.

Some of these optional services require a proprietary kernel module. The use of these services is tied to the SMART EC OS distribution. Contrary to the shelfHost, users can update installed packages and install new packages with the OS package manager (`yum` for CentOS).

NOTICE

These packages are excluded from upgrade:

`postgresql, freeradius, libpqxx, mod_ssl, syslog, telnet, vsftpd, ftp, httpd, expect, eventlog, daemonize, ivykis`

SMART EC also provides the following software images:

- `mc3ksw_update_centos-7.3_<Version number>.iso`
- `SSFMAXCORE-R_<Version number>.iso`

These software images are released based on need. Contact your local SMART EC sales representative to obtain the latest software images.

The following table provides the directory structure of the `MC3K-COMPLETE-ISO_<Version number>.iso` software image.

Table 2-1 ISO Directory Structure

Directory	Files	Description
EFI	`BOOT`	Boot into UEFI mode
isolinux		Boot into legacy mode
images	`kernel`	Kernel image file used for USB, PXEboot or Disk booting
	`ramfs.xz`	`initramfs` based root file system
	`files.sha1sum`	Sha1sum of `vmlinuz` and `ramfs.xz` are used during the installation on to the disk
utils	`mkinitramfs`	Create `ramfs.xz` image
	`unpackinitramfs`	Unpack `ramfs.xz` image
	`create_bootusb.sh`	Install bootable `kernel` and `ramfs` onto disk

2.2.1 Linux Distribution

The distribution is based on CentOS 7.x.

The default runlevel is **3**. The root file system does not contain any graphical interface packages. The following table shows the default login credentials.

Table 2-2 Login Credentials

Username	Password
root	root

2.2.1.1 Ethernet Interfaces

This section provides information on Ethernet device (s) interfaces.

- Backplane Ethernet devices assigned to an aCPU have the following naming convention:
  `e_s<physical_slot_id>d<device_id>f<function>`
- aCPU (on-board Ethernet devices) and mCPU (all Ethernet devices) have the following naming convention:
  `enp<pci_bus_num>s<pci_device_num>f<pci_func_num>`

2.3 Console Access (BIOS and OS)

This section provides information about various methods for console access.

2.3.1 VGA over LAN to Access shelfHost Console

Start a VGA console session to access shelfHost (**JViewer**). This gives you guaranteed console access and you can also run software with graphical output.

NOTE: The shelfHost console can only be accessed using this method.

1. Install Java from www.java.com to your PC (if it is not already installed).
2. Go to **Java Control Panel > Security > Exception Site List** and add **172.26.0.1** to the list.
3. In MegaRAC GUI, go to **Remote Control > Console Redirection** and click **Launch** to view the **JViewer** window. If any security warning messages pop-up, accept them to view the **JViewer** window.

2.3.2 Serial over LAN

Open an SSH session on the BMC using the same IP address and login credentials as MegaRAC.

```
/opt/fru/bin/sol <slotID> <cpuID>
```

The slotID ranges from 1 to 15 and the cpuID is either 1 or 2. Press <ESC + t>, if you want to terminate the session and open a new one for another CPU. Be aware that the session terminates automatically after 1800 seconds. Re-open it when you see the BMC prompt.

To use SOL console, use the second option during the bootup in the GRUB prompt.

console=ttyS2,38400n8

You can use this method to access any of the hosts (shelfHost, applicationHost, networkHost).

2.3.3 Serial Console

CPU1 and/or **CPU2** serial consoles are exposed via face plate through Silicon Labs (CP2105) single-chip USB to Dual UART bridge. Connect a microUSB to USB cable with mircoUSB end to the **CONSOLE** on the faceplate and the other end to the PC or laptop.

For Windows 7 or Later: Teraterm
Drivers are automatically installed upon connecting the cable.

For Linux (any recent distribution): minicom
Serial USB driver *cp210x* should be available.

2.3.3.1 Serial Console Options

The following table provides serial port configuration information.

Table 2-3 Serial Port Configuration Parameters

Parameter	Default Settings
Baud Rate	38400
Data Bits	8
Parity	No
Stop Bit (s)	1
Flow Control	Off
Terminal Type	Teraterm

The output from the card can be viewed over the serial console. The default kernel command line uses the following console options.

console=tty0 console=ttyS1,38400n8

You can use this method to access any of the hosts (shelfHost, applicationHost, networkHost).

2.4 Installation

2.4.1 Installation Source and Target Media

This section provides information about different types of installation source media and types of target media for installation.

- Source image media for installation
  - CD or HDD over LAN
  - External TFTP server
  - USB drive at chassis front
  - USB drive at card bracket
- Target media for installation
  - SSD or HDD in drive bay
  - SSD on PCIE-600x card
  - SD on card
  - SSD on card (PCIE-721x only)
  - iSCSI drive on shelfHost (PCIE-9205 recommended)

2.4.2 Remote Installation of shelfHost

You can install the shelfHost remotely or locally. This section explains remote installation of the shelfHost from your notebook/PC using the CD or HDD over LAN as the source image media for installation.

Download the Complete ISO on to your notebook/PC and follow the below procedure:

1. Access MegaRAC GUI by providing the BMC IP address in a browser. The default IP address is 192.168.201.9. Use default login credentials **admin/admin**.

2. In the MegaRAC GUI, launch the Remote Console by going to **Remote Control > Console Redirection** and then clicking **Java Console**.

   This will load a Java application from the BMC and will also launch the JRE to run this application.The JRE will not run an application from an unknown source. In such case, use the **Java Control Panel** to change the security settings and add the BMC's IP address to the **Java Site List**.

   A new **JViewer** window pops up with set of pull-down menus and the remote console. Select the **JViewer** window and <Enter> to see the text or graphical message of your OS.
3. In the **JViewer** window, go to **Media > Virtual Media Wizard** to display **Virtual Media** window.
4. In **CD/DVD** Media, select **CD image.**
5. Click **Browse** and select the downloaded Complete ISO. Click **Connect CD/DVD** and then **Close**.
6. In the MegaRAC GUI, go to **Remote Control > Chassis Power & Reset**, select **shelfHost Reset** and then **Perform Action.**
7. Observe the **JViewer** session and press <F4> to enter the **Boot** menu. This may take several seconds.
8. In **Boot** menu, select **Virtual CD ROM** and press <Enter> to boot.
9. After boot up, log in using default credentials **root/root**.
10. Identify the storage device name for the microSD card using `parted -l`
    Note: We are currently shipping a **Model: Generic Ultra HS-COMBO** microSD card, but that may change. The assumption in this procedure is `/dev/sda`.
11. Execute the following command:
    ```
    $ disk_install.sh -d disk:sda -i bootcd -t
    ```

    The above command is preferred, because in addition to installing Complete ISO, it also sets up a TFTP server to enable PXEboot for applicationHost CPUs.
12. Disconnect remote storage.
    In **JViewer** menu, go to **Media > Virtual Media Wizard**, click **Disconnect CD/DVD** and close the window.
13. Reboot the shelfHost.
14. Enter BIOS and select the installed media as the new boot device.

For local installation procedure, refer to *Local Installation of shelfHost, applicationHost, and networkCPU (using bootable USB device) on page 26*.

2.4.3 Remote Installation of applicationCPU

You can install the aCPU either remotely or locally. This section explains the remote installation procedure.

The shelfHost must be up and running before you install any other CPU in the MaxCore system. Before you proceed with the next chapter, familiarize yourself with the SSF methods to power and reset the individual CPUs in the system. Login to SSF and select the CPU of your choice from the Navigation pane at the left. The GUI will then display the power and reset buttons of this specific CPU. For more information about options to power on/off a CPU, refer to section *Powering ON/OFF of a CPU on page 42*.

For local installation, refer to section *Local Installation of shelfHost, applicationHost, and networkCPU (using bootable USB device) on page 26*.

2.4.3.1 PXEBoot applicationHost CPU from shelfHost

To PXEBoot the applicationCPU from shelfHost:

1. Open TTY session to any applicationCPU. The following commands assume CPU2 in slot 1:
   `/opt/fru/bin/sol 1 2`
2. Execute the following commands on the shelfHost to power off and power on the applicationCPU.
   ```
   mccs_tool.py --method=set-cpu-power --cpu=1,2 --power=off
   mccs_tool.py --method=set-cpu-power --cpu=1,2 --power=on
   ```
3. Enter the BIOS boot console with <F4> key.
4. Boot from network interface with MAC address 02:01:00:10:02:xx (xx is the number of the assigned VF).
5. Log in using default credentials **root/root** and install the Complete ISO on to the disk.

2.4.3.2 Installing to an applicationCPU Disk

This installation procedure is similar for SATA disk, on-board SSD, SSD on PCIE-600x cards, and third-party NVMe cards.

To install to an applicationCPU disk:

1. Identify the storage device name for the microSD card. It is `/dev/sda` for this example.
   `parted -l`
2. Install Complete ISO to disk. The below command is framed with an assumption that shelf Id is 1 and disk is mounted on `/dev/`**`sda`**.
   `disk_install.sh -d disk:`**`sda`**` -i 172.27.`**`1`**`.2:/default/common/ images`
3. Reboot the applicationCPU.

4. Enter BIOS with <F2> key and modify the following parameters.
   `BIOS: BOOT > EFI Device First [Disabled]`
   `BIOS: BOOT > Legacy > Boot Type Order`

   Note: Move the USB (BIOS sees the microSD card as a USB device) to the top of the list, save with <F10> and let the shelfHost boot from its microSD card.

   Note: Above suggested BIOS configuration changes would disable SSF support for network boot. If you want to network boot the CPU using SSF, revert the changes. For more information about Network Boot Configuration, refer to *SSF for MaxCore MC3000 Platform GUI Help*.

2.4.4 Installation of networkCPU

You can only install the networkCPU locally. Refer to the following section *Local Installation of shelfHost, applicationHost, and networkCPU (using bootable USB device)* for local installation.

2.4.5 Local Installation of shelfHost, applicationHost, and networkCPU (using bootable USB device)

This is a common procedure that can be used to install on shelfHost, applicationHost, and networkCPU.

Prerequisites

- 1GB USB drive
- microUSB OTG adapter cable
- A PC with CentOS7.3 with extlinux, syslinux, sgdisk, and parted installed. The installation script will verify these utilities and abort, if not available. Alternatively, you can use shelfHost with USB connector at the front.
- Make sure the shelfHost must be up and running, before you install applicationHosts and networkCPUs.

2.4.5.1 Creating a Bootable USB on a Linux Machine

To create a bootable USB on a Linux machine, use the shelfHost with a USB connector at the front:

1. Connect the USB drive.

2. Mount the MC3K-COMPLETE-ISO_<Version number>.iso to a directory.
   ```
   $ mkdir -p /mnt
   $ mount -o loop MC3K-COMPLETE-ISO_<Version number>.iso /mnt
   ```
3. Run the create_bootusb.sh script available in the utils directory under mount point and then follow on-screen instructions.
   ```
   $ cd /mnt/utils
   $ sh create_bootusb.sh
   ```

You can now install the software using this bootable USB.

2.4.5.2 Install using Bootable USB

To install the software using a bootable USB drive:

1. Access the console. (For access methods, refer to section *Console Access (BIOS and OS) on page 21*).
2. Connect the bootable USB drive with an OTG cable to the microUSB connector at the card bracket and reboot the CPU using SSF.
3. Change the BIOS mode to UEFI. Refer to BIOS sections in the installation and use manuals of respective PCIE cards.
4. Boot the card with the bootable USB drive.
5. Login using default credentials **root/root**.
6. Identify the name of the target storage device. The target storage device could be microSD, on-board SSD, 2.5" SATA, or NVMe or iSCSI.
   ```
   parted -l
   ```
7. Execute disk_install.sh -d disk:<storage-device> -i bootusb

   If you want to install Complete ISO on shelfHost using a bootable USB and also want to setup a TFTP server to enable PXEboot for applicationCPUs, use the following command:
   ```
   disk_install.sh -d disk:<storage-device> -i bootusb -t
   ```

   For example, if /dev/**sda** is the storage device, the command will be:
   ```
   disk_install.sh -d disk:sda -i bootusb -t
   ```

2.5 Upgrading Shelf using System Update

SSF supports MaxCore System Update from release 1.1.0.28 (SP5). Using this feature, you can update a complete shelf with a single command.

SSF for MaxCore release package contains a System Update package, which includes upgrades for all SMART EC Cards (PCIE-7410, PCIE-7210, and PCIE-9205) for MaxCore platform. You can use System Update using CLI and XML interface. The following section describes the procedure in detail.

NOTICE

If any of the applicationHost CPU is booted with network boot, boot that CPU with the latest ramdisk image to avoid any upgrade to be initiated.

To perform a System Update, first connect to SSF through the CLI or XML interface through Serial access or over IP. For more information, refer to sections *Accessing the MaxCore MC3000 Platform using SSF XML Interface on page 35* and *Accessing the MaxCore MC3000 Platform using SSF CLI on page 35*.

The default credentials are:

Username: *Admin*

Password: *Admin*

NOTICE

The MaxCore System Update only updates the local storage, so you must copy the latest ramdisk image if you are booting any of the application hosts using netboot.

2.5.1 Checking Shelf Status

System Update upgrades the shelfHost and all participating applicationHost CPUs on the shelf. An applicationHost can only participate in System Update if the upgrade agent (part of SSF agent) is available and connected to the shelfHost.

The command example shown next lists the applicationHost CPUs which are participating in the System Update. Verify all of the applicationHost CPUs that are listed in the output of the command with the current version.

To check the current versions of all the available shelf components and to view the update status, use the status command as shown.

```
[root@pcie9205-s1-c1 ~]  telnet localhost 11001
  Trying ::1...
  telnet: connect to address ::1: Connection refused
  Trying 127.0.0.1...
  Connected to localhost.
  Escape character is '^]'.
```

```
  Welcome to SSF CLI
  Username: Admin
  Password:
  Access granted
  >en
  con t
  MaxCore(config)system 1
  MaxCore(system-1)shelf 22
  MaxCore(shelf-1-22)system-upgrade-status
================================================
Shelf# 22             status: Upgrade Success
================================================
  Slot# 1
  CPU# 1  status: Upgrade Success
  ------------------------------------------------------------
   Entity      Status      Curr Ver          Last Updated
  ------------------------------------------------------------
 BMC_Application          1.8.000000      Sun Mar 12 12:42:41 IST 2017
BMC_CPLD_Companion            0.03.03000000      Sun Mar 12 12:42:41 IST
2017
BMC_CPLD_Glue                    0.03.00000000       Sun Mar 12 12:42:41
IST 2017
BMC_CPLD_I/O_Module       0.06.01000000   Sun Mar 12 12:42:41 IST 2017
Basic_Board_Services              1.5.0               Sun Mar 12 12:42:41
IST 2017
System_Services_Framework Upgrade Success   1.1.0.29 Sun Mar 12 13:42:53
IST 2017
PCIe-9205_BIOS 1.4.00000002        Sun Mar 12 12:42:41 IST 2017
PCIe-9205_CPLD       01.00.00          Sun Mar 12 12:42:41 IST 2017
PCIe-9205_Switch_Management    Upgrade Success           1.0.2.24
Sun Mar 12 13:45:36 IST 2017
ViewCheck   Upgrade Success   1.0.2.15    Sun Mar 12 13:48:06 IST 2017
  Slot# 2
  Slot# 3
  Slot# 4
  Slot# 5
```

```
  Slot# 6
    CPU# 1 status:       --
     ------------------------------
    Entity        Status        Curr Ver        Last Updated
     ------------------------------    -------------------------
    CPU# 2 status: Upgrade Success
     ------------------------------    -------------------------
    Entity     Status           Curr Ver        Last Updated
     ------------------------------    -------------------------
Basic_Board_Services   1.5.0           Sun Mar 12 13:29:53 IST 2017
System_Services_Framework   Upgrade Success   1.1.0.29           Sun Mar
12 14:07:23 IST 2017
PCIe-7410_BIOS   1.4.00000002        Sun Mar 12 13:29:53 IST 2017
PCIe-7410_CPLD      01.00.00          Sun Mar 12 13:29:53 IST 2017
 ViewCheck Upgrade Success 1.1.0.15   Sun Mar 12 14:07:50 IST 2017
  Slot# 7
  Slot# 8
  Slot# 9
    CPU# 1  status:       --
     ------------------------------    -------------------------
    Entity    Status              Curr Ver       Last Updated
     ------------------------------    -------------------------
    CPU# 2  status: Upgrade Success
     ------------------------------    -------------------------
    Entity  Status              Curr Ver        Last Updated
     ------------------------------    -------------------------
  Slot# 10
  Slot# 11
  Slot# 12
  Slot# 13
  Slot# 14
  Slot# 15
    CPU# 1  status:       --
     ------------------------------    -------------------------
    Entity Status              Curr Ver        Last Updated
     ------------------------------    -------------------------
```

```
    CPU# 2  status:       --
     ------------------------------    ------------------------
  Entity   Status               Curr Ver        Last Updated
     ------------------------------     ------------------------
```

In the above output, notice that Slot6- CPU1, Slot9-CPU1, Slot9-CPU2 Slot15-CPU1 and Slot15-CPU2 are not participating in System Update but Slot6-CPU2 is participating.

2.5.2 Initiating System Update

To perform a System Update, follow these steps.

1. Download the System Update package from the delivery to the ShelfHost of the MaxCore for which you want to initiate the update.
2. Connect to SSF through the CLI or XML interface through Serial access or over IP.
3. Run the following command:
   `MaxCore(shelf-1-22)#system-upgrade-initiate filename "<System Update file with Absolute Path>"`

This initiates the upgrade on the shelf which includes multiple power cycles and reboots.

For more information about System Update command, refer to *SSF for MaxCore MC3000 Platform XML Interface Guide* and *SSF for MaxCore MC3000 Platform Command Line Interface Guide*.

Accessing the MaxCore MC3000 Platform

3.1 Overview

The following interfaces can be used to access the MaxCore MC3000 platform:

- Web interface
- XML interface
- CLI
- Simple Network Management Protocol (SNMP)

3.2 Accessing the MaxCore MC3000 Platform using SSF Web Interface

You can access the MaxCore MC3000 platform using the web interface for configuring, managing, and monitoring the platform equipped with multiple resources. You can use any of the following browsers to log on to SSF and access the MaxCore MC3000 platform:

- Internet Explorer version 10.0 and later
- Mozilla Firefox 12.0 and later
- Google Chrome version 23 and later

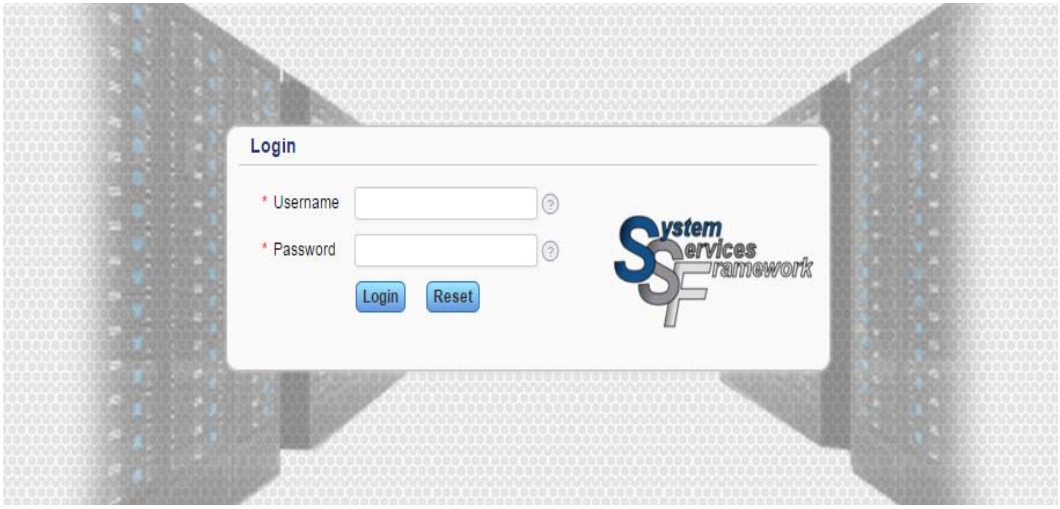
NOTE: You may get a blank screen after installation or system timeout. Cleanup the browser cache (CTRL + SHFT+ DEL) and reconnect the SSF using the web interface. This is a one-time activity.

To access the MaxCore MC3000 Platform using SSF Web interface:

1. Type the SSF web application URL in the address bar of the web browser and <Enter>. You can obtain the SSF URL from your administrator. The **SSF Login** dialog box opens, as shown in *Figure 3-1*.

   For example, `https://<IP Address>:<PortNumber>`

Figure 3-1 Login Page



2. In the **Login** dialog box, type your *User name* and *password*.

The default user name and password is Admin.

3. Click **Login** for logging into **SSF**. The SSF Home page opens.

For more information on Web interface, refer to Online Help integrated with the SSF application. Click the **Help** icon to access the Online help.

NOTICE
Make sure that the SSF Web Interface is initiated before accessing it. In order to access SSF GUI, the SSF discovery/initialization should be completed.

3.3 Accessing the MaxCore MC3000 Platform using SSF XML Interface

The XML interface of SSF passes management requests to the SSF framework for processing. It also handles responses and notifications/events from the SSF framework.

The XML interface facilitates access to SSF using an XML-based request protocol. The XML interface is intended for remote configuration of software and scripts. It can also be used via a remote GUI configuration tool. The XML-based requests are sent over a persistent connection to the XML agent, which processes the requests and returns XML-based responses. SSF also sends asynchronous responses/events over the XML interface such as alarm notifications, etc. By default, these events are disabled.

For more information about the XML commands, refer to *Appendix B, Related Documentation* for the *SSF for MaxCore™ Platform XML Interface Guide*.

3.4 Accessing the MaxCore MC3000 Platform using SSF CLI

You can access the MaxCore MC3000 platform using the SSF CLI. SSF provides a fully functional CLI.

To access the MaxCore MC3000 Platform using CLI:

1. Establish a secure shell connection to SSF host using SSH.
2. Start the **telnet** connection from an already established secure shell.

```
root@localhost ~]# telnet localhost 11001
Trying ::1...
telnet: connect to address ::1: Connection refused
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Welcome to Maxcore CLI
```

3. Type your *user name* and *password*.

```
Username: Admin
Password:
Access granted

>enable
#configure terminal
Maxcore(config)#
```

NOTICE

By default, the administrator user name and password are Admin. To change the existing password, see *Chapter 4, Managing and Configuring MaxCore Platform on page 39***.**

3.5 Accessing the MaxCore MC3000 Platform using SNMP

The SNMP is designed to provide a means of managing and monitoring diverse network devices. It has client-server architecture and uses unencrypted text known as community strings for authentication. Communication between the client and server is accomplished using a command. There are four commonly used commands: `snmpget`, `snmpset`, `snmpwalk`, and `snmptrapd` for receiving trap message.

You can access SSF on the PCIe card using port number `10165`, private enterprise number `26061` and text file `MAX-CORE-MIB` which will be available in `/usr/share/snmp/mibs` directory.

Example for SNMPget

```
#snmpget -v2c -cprivate -m /usr/share/snmp/mibs/MAX-CORE-MIB
snmp_agent_Ip_address:10165 systemName.1
MAX-CORE-MIB::systemName.1 = STRING: "MaxCore System Framework"
```

Example for SNMP walk

```
#snmpwalk -m /usr/share/snmp/mibs/MAX-CORE-MIB  -v2c -c private
snmp_agent_Ip_address:10165 1.3.6.1.4.1.26061
MAX-CORE-MIB::systemInfo.1 = STRING: "System Services Framework for
Configuring MaxCore shelves"
MAX-CORE-MIB::systemName.1 = STRING: "MaxCore System Framework"
MAX-CORE-MIB::maxNoEvents.1 = Gauge32: 1000000
MAX-CORE-MIB::eventFilterSeverity.1 = Gauge32: 1
MAX-CORE-MIB::eventFilterType.1 = Gauge32: 32
MAX-CORE-MIB::userConfig.1 = Gauge32: 0
MAX-CORE-MIB::shelfName.1.1 = STRING: "Maxcore (r1.s1.a1.b1)"
MAX-CORE-MIB::shelfAddr.1.1 = STRING: "r1.s1.a1.b1"
```

```
MAX-CORE-MIB::shelfInventoryInfo.1.1 = STRING: "Vendor: ARTESYN, Product:
MAXCore"
…
```

Example for SNMP set

```
#snmpset -m /usr/share/snmp/mibs/MAX-CORE-MIB  -v2c  -   cprivate
snmp_agent_Ip_address :10165 systemName.1 s "MAXCORE"
MAX-CORE-MIB::systemName.1 = STRING: "MAXCORE"
```

Example for SNMP trap

You can configure the PCIE card using the CLI to send notifications to SNMP managers as traps.

```
#snmp-server host snmp_traphost_Ip_address trap version 2c SSF udp-port
162 Admin
```

To receive traps, start snmptrapd application on configured manager that listen at port 162 and notifies the traps.

```
#snmptrapd  -f -Lo  -m MAX-CORE-MIB
```

Managing and Configuring MaxCore Platform

4.1 Overview

The System Services Framework (SSF) provides a management and configuration interface to the SMART Embedded Computing hardware and software products. It facilitates system level configuration and management access to SSF managed hardware and software components through Web, XML, and CLI protocol interfaces.

SSF represents all the managed hardware and software components in a simple and easily manageable hierarchal model. It also supports persistency and playback of the MaxCore™ configuration.

The following are the key features supported by SSF:

- Access, Authentication, and Authorization
- Configuration Management
- Hierarchal representation of System model
- Dynamic population of System model
- Remote system Configuration Management
- Application Management of Remote systems
- Event and Alarm management
- Graphical Monitoring of Sensors

4.2 PEX Modes and Network Configuration

The MaxCore platform has two different architecture modes:

- Single-Host Mode
- Multi-Host Mode

4.2.1 Single-Host Mode

In this mode, it is a single PCIe domain, where only one slot (slot1 or slot15) is populated with one host card and other slots are populated with PCIe endpoint cards. This mode can be compared with any other PCIe rack server with a host processor and multiple PCIe slots (14 slots).

4.2.2 Multi-Host Mode

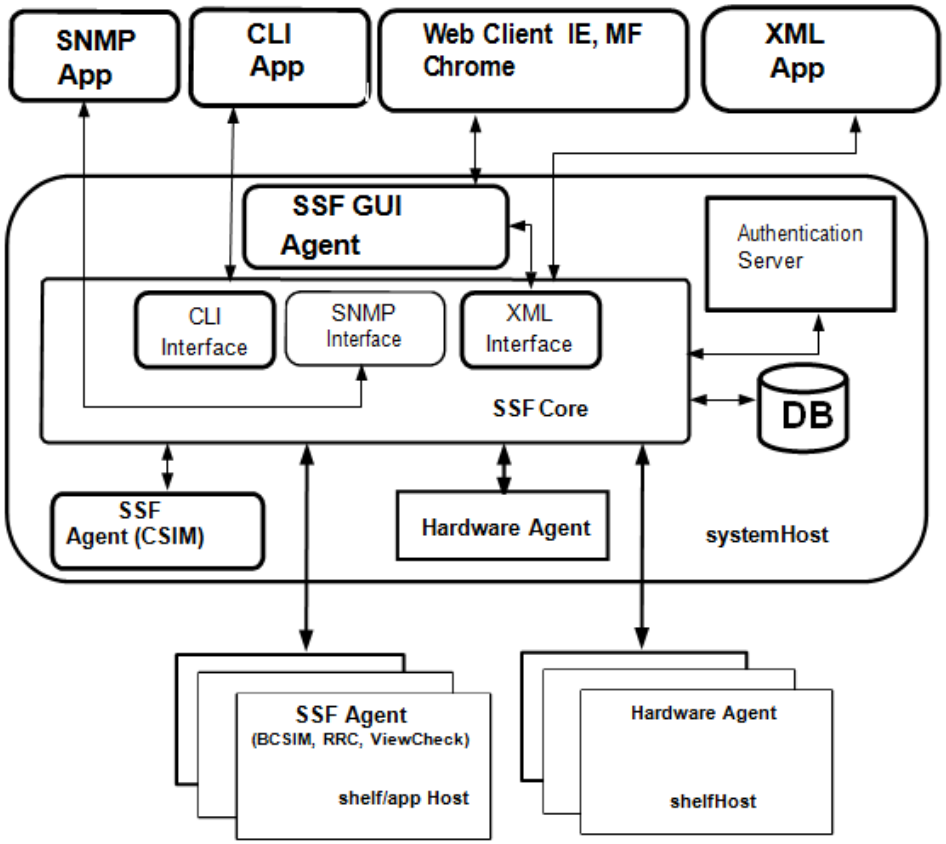
In Multi-Host mode, multiple slots are configured as Host Ports. There can be multiple PCIe hosts that contains PCIe endpoints associated with them. The shelfHost can either assign the primary function (PF) of a device or a virtual function (VF) of a device to an App Host CPU root port.

By default, the MaxCore system is configured in Base mode. You can switch the MaxCore system to ExpressFabric mode using SSF. This will result in the power cycle of the shelf host and will start the MaxCore system with the multi-host configuration as default.

4.3 SSF Components

The following figure illustrates various components of SSF.

Figure 4-1 SSF Components



SSF has the following components:

- **SSF Agent** - SSF Agent is the software component that models and represents underlying hardware or application to be managed. There is one SSF Agent that models the shelf hardware and runs on shelfHost, which is the called CSIM. Apart from CSIM, each applicationHost can run more than one SSF Agent, each modeling a particular software or hardware specific to the host. For example, an applicationHost can run one SSF Agent to configure and manage Linux Applications and another agent for diagnosing the hardware. Similarly, a networkHost can run an agent to manage the underlying RRC switch.
- **Hardware Agent** - This is a software component that runs on each shelfHost. This component understands the lower level languages like IPMI to communicate and discover complete hardware. The above mentioned CSIM models the hardware based on discovery of hardware agent.
- **SSF Core** - It is the central component of SSF. It provides single point access to the complete system. It uses north-bound external interface agents to communicate with the user. It receives requests from different external interfaces and forwards them to the one or many targeted SSF Agents. SSF Agents are the actual work horse that gets the job done for the user. SSF Agents in reply sends the response to SSF Core which in turn sends it to the user. It also receives asynchronous events from the SSF Agents and forwards them to the applications registered for the events.
- **SSF GUI Agent** - It is an application that interacts with the SSF Core to provide web interface for accessing SSF managed system. Similar to SSF core, SSF GUI Agent has to be installed on shelf host.

4.4 Resetting the Administrator Password

To reset the administrator password:

1. Log on to the card on which SSF and PostgreSQL are installed.
2. Log on as a PostgreSQL user with the `# su -l postgres` command.
3. Connect to the PostgreSQL database using the following command:
   `#/usr/local/postgres/bin/psql SSF`

   The following output is displayed:
   ```
   psql (9.1.3)
   Type "help" for help.
   SSF=#
   ```

4. List the available users, using the SSF=# `select * from "user";` command. The list of available users along with the passwords are displayed as shown as follows:

```
user | password | hash
-------+----------+------
Admin | Admin    |
(1 row)

SSF=#
```

5. Reset the administrator password using the following command:

```
SSF=# update "user" set password = 'test' where "user" = 'Admin';
UPDATE 1
SSF=#
```

After changing the administrator password, you can check whether the new password is reflected or not by listing the available users using the SSF=# `select * from "user";`command. The following output is displayed:

```
SSF=#
SSF=# select * from "user";
user | password | hash
-------+----------+------
Admin | test     |
(1 row)

SSF=#
```

4.5 Powering ON/OFF of a CPU

SSF provides the following options to power on/off a CPU:

- **Graceful Power Off** - Allows you to gracefully power off of the selected CPU. (The host can be a shelfHost CPU, or an applicationCPU or a networkCPU). If you want to gracefully power off shelfHost, it is strongly recommended to gracefully power off all applicationHosts. You can do this by clicking the **Power Off All App Hosts** button on **System > Shelf > Overview** screen.
- **Power Off** - This toggle button instantaneously powers off the selected CPU. This is not a graceful power off. It is suggested to use this option only when the graceful power off operation fails. If you want to power off shelfHost, it is strongly recommended to gracefully power off all applicationHosts. You can do this by clicking the **Power Off All App Hosts** button on **System > Shelf > Overview** screen.

- **Reboot** - Allows you to gracefully reboot the selected CPU. The rebooting of shelfHost results in power reset of applicationHosts. it is strongly recommended to gracefully power off all applicationHosts gracefully before rebooting shelfHost. You can do this by clicking the **Power Off All App Hosts** button on **System > Shelf > Overview** screen.
- **Reset** - This instantaneously resets the selected CPU. This is not a graceful reboot. It is suggested to use this option only when graceful reboot operation fails. If you want to perform shelfHost reset, it is strongly recommended to gracefully power off all applicationHosts gracefully. You can do this by clicking the **Power Off All App Hosts** button on **System > Shelf > Overview** screen. If SSF is running on the shelfHost, resetting the shelfHost will make SSF not accessible.

For more information, refer to *SSF for MaxCore MC3000 Platform GUI Online Help*.

4.6 Configuring a Non-SMART EC Card for SSF

If a non-SMART EC PCIe card is inserted into the MaxCore chassis, the card information will be shown as **Unknown** in the SSF GUI. Let us assume that a non-SMART EC card is inserted in slot 3.

To view the card information in the SSF GUI, you need to perform the following steps.

1. Obtain the vendor and device details of the card using **mccs_tool.py** tool. For this you need to run the following command on shelfHost:

   ```
   #mccs_tool.py --method=list-devices
   ```

NOTE: Only the output of the card in slot 3 is shown below for quick reference:

```
slot: 3 device: 1
  func: 1
      vendor          : 0x8086
      device          : 0x15a4
      class           : 0x20000
   pci location@mcpu  : 0f:00.0
   plx switch         : 0
   plx station        : 1
   plx port           : 4
```

2. Use the bus device function `0f:00.0` shown at `pci location@mcpu` in the above step to get device details as shown below:

   ```
   [root@pcie7410-s1-c1 ~]# lspci -s 0f:00.0 -vnn

   0f:00.0 Ethernet controller [0200]: Intel Corporation Ethernet Switch
   FM10000 Host Interface [8086:15a4]
           Subsystem: Artesyn Communication Products Device [1223:2020]
   ```

3. Add the retrieved device details in `/opt/ssf/etc/config/main/pcidb.csv` file in the following format:
   `<vendor, vendor desc, device, device desc <,sub vendor, sub vendor desc, sub device, sub device desc -- card name, device num, device name>`

   For example:
   `0x8086,"Intel Corporation",0x15a4,"Ethernet Switch FM10000 Host Interface",0x1223,"Artesyn Communication Products",0x2020,"PCIE-9205","--","PCIE-9205"`

   The above example shows the following details:
   Vendor id - 0x8086
   Vendor description - Intel Corporation
   Device id - 0x15a4,
   Device description - Ethernet Switch FM10000 Host Interface
   Sub-vendor id - 0x1223
   Sub-vendor description - Artesyn Communication Products
   Sub-device id - 0x2020
   Sub-device description - PCIE-9205
   SSF delimiter (for providing human readable strings for card and devices) – "—"
   Card name - PCIE-9205

   Note: SSF delimiter and elements following delimiter are optional. This is the interface for the user to give a human readable name to card for any third-party card that does not provide FRU information. These names are read and used in SSF GUI.

   If either the entry is not available in `pcidb.csv` or if human readable text is not provided, then the names would default to unknown.

4. Restart the SSF hardware agent.
   `#/opt/ssf/etc/config/S99mcagent.sh restart`

After completion of the procedure, the name of the card with its details are shown in SSF GUI.

4.7 Updating Shelf Address

The SSF allows you update the MaxCore shelf address through the GUI. The shelf ID is used to configure the internal default network. Several MAC and IP addresses will incorporate this number. Be cautious when you are going to change the shelf ID of an already running system. Shelf IDs must be unique within the same SSF system stack. The shelf name is a reference for management layers above SSF.

The SSF itself does not use this parameter. The shelf ID and shelf name can be uploaded using SSF. For more information on how to update a shelf address, refer to the *SSF for MaxCore MC3000 Platform GUI Online Help* menu.

4.8 Network Boot Configuration

The SSF allows you to enable or disable the network boot configuration of the applicationHost(s). The Network Boot Configuration dialog box in the SSF GUI shows the status and availability of images (Ramdisk and Kernel) of the CPU for the PCIE card (PCIE-7410, PCIE-9205, and PCIE-7210).

For more details about *Network Boot Configuration,* refer to the *SSF for MaxCore MC3000 Platform GUI Help* menu.

4.9 Setting Default Configuration

This option allows you to set the default assignments, ETH3 and ETH4 virtual functions, to applicationHosts. The pre-defined virtual functions are automatically assigned to the available applicationHosts and thus bringing applicationHosts into internal networking.

NOTICE

You do not need to assign these virtual functions to applicationHost manually. This operation will not disturb any other assignments made from other endpoint devices.

For more details on how to set default configuration, refer to the *SSF for MaxCore MC3000 Platform GUI Help* menu.

4.10 Configuration Management

The Configuration Editor is for editing a file with the MaxCore baseboard configuration. When this editor is active, the SSF GUI no longer displays the connected hardware. The GUI shows the content of the configuration file and is used to edit its content. A red frame below the System Alarms is visible when the GUI displays the editable file content. The content of the currently opened file can be applied to the connected system.

Be aware that the applied changes may result in a reboot of most CPUs in all shelves of the system. You can exit the editor to regain access to the connected system.

Make note that editing the saved file with a text editor is possible, but may result in an inconsistent configuration file.

For more information, refer to the *SSF for MaxCore MC3000 Platform GUI Help* menu.

4.11 Enabling Network Time Protocol

Using this feature you can enable or disable SSF Configured Network Time Protocol (NTP) Service. SSF provides this feature at the CPU level of a MaxCore system. This feature allows you to synchronize the date and time of application hosts with the shelfHost.

To enable or disable this feature, click the Off/On control button in line with the SSF Configured NTP Service label. This is a toggle button. When this service is enabled (ON), the time stamp of all the application hosts running in the MaxCore system gets aligned with shelfHost running time and when this service is disabled (OFF), the respective application hosts will be running independently irrespective of the shelfHost running time.

NOTICE

Disable SSF Configured NTP Service if you want to synchronize any of the applicationHosts date and time with an external source other than shelfHost.

4.12 Configuring Multiple MaxCore Shelves

To configure multiple MaxCore shelves, follow the procedure below.

NOTICE

This procedure assumes that shelf Ids/chassis numbers for the shelf are already configured as mentioned in *Updating Shelf Address on page 44***.**

On the System Host (that hosts all the MaxCore shelves)

1. Check the current shelf's configuration using CLI or XML Command:

```
MaxCore(config-HardwarePlatformManager)#listShelves
Shelf: Shelf
      rackID: 1
      ShelfId: 1
      Name: Shelf
      shelfHostIpAddr: 127.0.0.1
      isMaster: true;
```

   In this instance, the shelfHost IP address is 127.0.0.1 (localhost) and the shelf id is 1.

2. Add a new shelf using CLI or XML command. The shelf ID and shelfHost IP Address should be different from that of earlier configured shelves, and the shelfHost IP Address should be reachable bythe SSF core.

Syntax:

```
MaxCore(config-HardwarePlatformManager)#addShelf shelfID
<Shelf_Id> shelfHostIpAddr <Shelf_IP_Address> master false
shelfName <Shelf_Name>
```

Example:

```
MaxCore(config-HardwarePlatformManager)#addShelf shelfID 2
shelfHostIpAddr 172.27.2.2 master false shelfName Shelf2
```

For more information about add shelf command, refer to *SSF for MaxCore MC3000 Platform XML Interface Guide* and *SSF for MaxCore MC3000 Platform Command Line Interface Guide*.

On Shelf Hosts (other than System Host)

1. Disable loop-back for ETH3 while connecting multiple shelves to be on the default base network.
   ```
   # mccs_tool.py --method=set-loopback --mode=off --func=16,2
   ```
2. Start and Stop the following services:
   ```
   #systemctl stop ssfCore.service
   #systemctl disable ssfCore.service
   #systemctl start mcagent.service
   #systemctl enable mcagent.service
   ```

On both System Hosts and Shelf Hosts

1. Restart all SSF services.
   For more information on restarting SSF services, see *Starting SSF on page 65* and *Starting SSF Agent on page 65*.

4.13 Configuring firewalld to Allow SSF Communication

NOTICE

Assume that the following configurations are present:

- **`firewalld` is running both on SSF-Core and App-host**
- **`firewalld` is already started while executing the commands (Command: `systemctl start firewalld`)**

Follow the below procedure to configure `firewalld` to allow SSF internal communication.

On App-host

1. Identify the ports for which firewall rule needs to be added. Run the below command.
   ```
   # cat /opt/ssf/etc/config/bcsim/ssfApi.conf
   ```
   Output:
   ```
   # Transport type (one of: tcp, uds)
   #transport=tcp
    #listening address
   LocalTcpAddress=0.0.0.0:21215   <-  firewall rule to be added for
   this on App Host side
    # embeddedMIND process location
   emindTcpAddress=192.168.201.100:21212  <- firewall rule to be added
   for this on SSF Core side
   #emindUdsPath=/tmp/emind-tl-uds
    # Link health-check period (in seconds)
   healthcheckPeriod=10.0
    # Log settings
   #logEnabled=yes
   #logLevel=error # one of: error, info, debug
   #logFile=eMindApi.log
   [root@localhost ~]#
   ```
2. Add firewall rule on App Host. (port 21215 in this case). It allows traffic from SSF-Core to App-host.
   ```
   #firewall-cmd --zone=public --add-port=21215/tcp
   ```
   Change the port number in the above command as per your `ssfApi.conf` file.

On SSF Core-host

1. Add firewall rule on the SSF Core host. All app hosts usually connect to port 21212. So, this needs to be added for firewall exception.
   ```
   #firewall-cmd --zone=public --add-port=21212/tcp
   ```

4.14 Verifying SSF and BBS Versions Installed on the System

For verifying SSF version, execute the following command:

```
#cat /etc/ssf-release
```

For verifying BBS version, execute the following command:

```
#cat /etc/pcie-release
```

4.15 Changing Logging Configuration

By default, the log level configured is "info".

To change it to lower levels to avoid excessive logging, perform the following steps:

1. Login to SSF CLI and enter the config mode:

```
# telnet localhost 11001
Trying::1...
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
Welcome to SSF CLI
Username: Admin
Password:
Access granted
>enable
#configure terminal
 MaxCore(config)#
```

2. Go to `logfilter` class and select "syslog " instance and then enter **show** to get the current logging configuration.

```
MaxCore(config)#logfilter syslog
MaxCore(logfilter-syslog)#show
logfilter, syslog
type                = priority
priority            = info
modules             =
```

3. Enter priority from one of the following:

```
MaxCore(logfilter-syslog)#priority ?
  critical  Filter priority (applicable for priority log filter)
[debug]
  debug    Filter priority (applicable for priority log filter) [debug]
  error    Filter priority (applicable for priority log filter) [debug]
  info     Filter priority (applicable for priority log filter) [debug]
  warning   Filter priority (applicable for priority log filter)
[debug]
```

4.16 SSF Core Configuration

SSF core is a management and configuration interface between hardware and software. SSF core consists of two sets of configuration files. One for SSF-Core (`ssfMxcd`) and other for SSF-CSIM (`ssfcsimd`). All SSF-Core (`ssfMxcd`) executable configuration files are stored in `/opt/ssf/etc/config/main/`.

You can edit the `ssf.ini, maxcore.conf,` and `tl.ini` files stored at this location to configure the required parameters.

All CSIM (`ssfcsimd`) executable configuration files are stored in `/opt/ssf/etc/config/csim/`. You can edit the `ssfApi.conf` file stored at this location to configure the required parameters. *Table 4-1* describes the configuration options.

NOTICE

Misconfiguring any of the following may lead to an unusable system.

Table 4-1 SSF Core Configuration Files

File Name	Configuration Options	Description

`ssf.ini`
Directory:
`/opt/ssf/etc/config/main`
	`MaxSessions`	Default value: 100.
		You can edit it to any required value.

	`SessionTimeOut`	Default value: 1800 seconds. You can edit it to any required value.

Table 4-1 SSF Core Configuration Files (continued)

File Name	Configuration Options	Description

`maxcore.conf`
Directory:
`/opt/ssf/etc/config/main`
	`log_level`	Log level
		INFO - Notifications and important information.
		DEBUG - Verbose
		Default is INFO.

	`domain`	MaxCore ID in the system. This may become obsolete.
	`rack`	Rack ID in the system.
	`shelf`	Shelf ID or chassis number to uniquely identify the MaxCore system.
	`name`	Name of the MaxCore system. This is optional.
	`mcpu_ipaddr`	Shelf Host IP address. Default IP: `127.0.0.1`. You can edit it to any Shelf Host IP reachable by the system host.
	`mcpu_port`	TCP/IP port. Default value: 8888
	`mcpu_evt_port`	TCP/IP port for events. Default value: 8890
	Note: To access multiple MaxCore systems simultaneously, you can configure more than one domain in the same file.	

`tl.ini`
Directory:
`/opt/ssf/etc/config/main`
	`transport`	Default setting: TCP
		You can also change it to Unix Domain Sockets (UDS), if needed. SSF Server (SSF Host) listens on both TCP and UDS sockets for connection from the SSF agent.

	`emindTcpAddress`	Default setting: localhost You can replace with the IP address of the SSF Core.

Table 4-1 SSF Core Configuration Files (continued)

File Name	Configuration Options	Description

`ssfApi.conf`
Directory:
`/opt/ssf/etc/config/csim/`
	`transport`	Default setting: TCP
		You can also change it to UDS, if required.
		SSF Server (SSF Host) listens on both TCP and UDS sockets for connection from the SSF agent.

	`emindTcpAddress`	Default setting: localhost You can edit with the IP address of the SSF Core.
	`healthcheckPeriod`	Default setting: 1.0 second Every 1.0 second, the system checks the health link between CSIM core and SSF. You can modify this value to required duration.
	`logEnabled`	Default setting: enabled for log collection You can modify it to disable log collection.
	`logLevel`	Default setting: error. You can modify to either error, info or debug.
	`logFile`	Default setting: `eMindApi.log`

4.17 SSF Agent Configuration

The Server SSF core consists of SSF-BCSIM (`ssfbcsimd`) executable configuration files. These files are stored in `/opt/ssf/etc/config/bcsim/.` You can edit the `ssfApi.conf` file stored at this location to configure the required parameters.The following table provides the list of configuration options of `ssfApi.conf` file.

Table 4-2 SSF Server Configuration Files

File Name	Configuration Options	Description
`ssfApi.conf`	`transport`	Default setting: TCP You can change it to UDP, if needed.
	`emindTcpAddress`	Default setting: `172.27.1.2` You can edit with the IP address of SSF Core.
	`healthcheckPeriod`	Default setting: 1.0 second Every 1.0 second, the system checks the health link between CSIM core and SSF. You can edit this value to required duration.
	`logEnabled`	Default setting: enabled for log collection. You can edit it to disable log collection.
	`logLevel`	Default setting: error You can edit to either error, info or debug.
	`logFile`	Default setting: `eMindApi.log`
	`ShelfHostIPAddress`	Default setting: `172.27.1.2` You can edit with the IP address of the shelf host of the shelf in which the SSF agent is to be considered. **Note**: This option is only applicable to SSF Agent and no other TL servers.
	`localTcpAddress`	Default setting: `0.0.0.0:21215` You can edit the IP address and port used by the SSF agent to listen for incoming requests from SSF Core.

4.17.1 Service Manager Configuration

In the Service Manager (SMAN) configuration file `SMAN.conf`, you can add additional user-defined services. The `SMAN.conf` file is stored in `/opt/ssf/etc/config/bcsim/`.

Use the following guidelines to add a service:

- Service name should be less than 20 characters
- Service name should be the same as that of Linux daemon service, if there exists a Linux daemon
- Description of the service should be equal to or less than 128 characters
- Binary file path should be equal to or less than 128 characters
- Tabs should be given before the file list and space should be used between path and filename

Syntax

```
## SYSLOG-NG configuration ####

#service syslog-ng
{
        enable=y;
        desc="syslog-ng system logger application";
        binFilePath=/opt/ssf/etc/config/bcsim/etc/init.d/syslog-ng;
        numberOfConfigFiles=2;
        filename
        {
                /etc/syslog-ng/ syslog-ng.conf;
                /etc/syslog-ng/ scl.conf;
        }
}
```

NOTICE

To disable a particular service parsing, add '#' before the service.

4.17.2 Service Manager Configuration INI File

Service Manager (SMAN) configuration INI file (`SMAN.ini`) supports two modes:

- **ConfigMode**: Supports commit-config of configured applications. The services under SSF control should be separated by comma (,). For example: `[ConfigMode]:pcieBsnet,dhcpd,tftp,ntpd,syslog-ng,syslcu`
- **NonConfigMode**: Does not support commit-config of configured applications. For example: `[NonConfigMode]:syslog-ng,tftp`

NOTICE

Add an application under "configMode" in case you choose to perform 'Commit Config' for the application. Otherwise add it to "NonConfigMode".

4.18 Hardware Agent Configuration

The Hardware Agent RPM consists of the MaxCore Agent executable configuration files (`mxcagent.conf`). Configuration files are stored in `/opt/ssf/etc/config/agent.` You can edit the `mxcagent.conf` file stored at this location to configure the required parameters.The following table provides the list of configuration options of `mxcagent.conf` file.

Table 4-3 Hardware Agent Configuration

File Name	Configuration Options	Description

`mxcagent.conf`
Directory:
`/opt/ssf/etc/config/agent`
	`log_level`	Log level
		INFO - Notifications and important information.
		DEBUG - Verbose
		Default is INFO.

	`domain`	MaxCore ID in the system. This may become obsolete.
	`master`	Identifies if SSF core runs on this Shelf Host. Default: true
	`rack`	Rack ID in the system.
	`shelf`	Shelf ID or chassis number to uniquely identify the MaxCore system.
	`name`	Name of the MaxCore system (optional)
	`mcpu_ipaddr`	Shelf Host IP address. Default setting: `127.0.0.1`. You can edit it to any Shelf Host IP reachable by the system host.
	`agent_listening_port`	This is listening TCP/IP port to make a connection for SSF core. Default value: 8888.
	`agent_evt_listening_port`	This is listening TCP/IP port for event. T Default value: 8890.
	`con_type`	Connection type to BMC. Default value: smi.
	`ipmi_con_tmout`	IPMI connection timeout. Default value: 5000 msecs.
	Note: The below commands are not applicable if the connection type is (`con_type`) smi. It is applicable only if it is LAN type.	
	`bmc_ipaddr`	BMC IP address. Default IP: `192.168.201.9`. You can edit it to any BMC IP reachable by the shelf host.
	`port`	RMCP port. Default value: 623.
	`auth_type`	RMCP authentication type. Default type: md5.
	`privilege`	RMCP privilege. Default: admin.
	`username`	RMCP username. Default: admin.
	`password`	RMCP password. Default: admin.

Linux Command-line Utilities

5.1 Firmware Upgrade

The PCIe card has two firmware devices that need to be upgraded, when required. The first device is the CPU with the BIOS firmware and the second is the CPLD. The root file system includes the latest firmware images for both the devices and a firmware command-line utility (FCU) to execute the upgrade procedure. FCU provides the following operations:

- Query the device to return the current firmware version
- Show the version of a firmware image
- Validate the firmware image
- Verify whether the image is applicable on the target device
- Upgrade the device with the given firmware image

When you run the FCU with the help option `fcu –help or fcu –h`, a list of supported operations are displayed on the screen.

This section explains the firmware upgrade using the FCU tool, bundled as part of the Basic Blade Services (BBS).

Applicable: For mCPU and aCPU

Related packages: `fcu` and `pcie-firmware`

`pcie-firmware` images are available in the `/opt/bladeservices/rom/<PCIE Card Number>` directory and it contains BIOS and CPLD firmware.

The `fcu` utility is available in `/opt/bladeservices/bin` directory. It provides functionality to query and upgrade the BIOS and CPLD firmware.The following sections provide information about the commands used to query, upgrade, and verify the firmware.

5.1.1 BIOS Upgrade

5.1.1.1 Query Operation

Using the query operation, the FCU returns firmware information for a specific device (if used with -d) or information about all firmware devices.

If you want to know the current BIOS version before upgrading it with a new version, use the following command:

```
$ fcu -q
```

The following screen shows a typical output when the above command is executed.

```
[root@pcie7410-s1-c1 bin]# fcu -q
********************[[[[[REPORT BEGIN]]]]]********************
Operation: Query
Product Name: PCIE-7410

 #00 Device    : pcie7410-cpld
  Bank #0 -       Active Version: 01.00.00

 #01 Device    : pcie7410-cpu
  Bank #0 -       Active Version: 1.4.00000002

********************[[[[[ REPORT END ]]]]]********************
[root@pcie7410-s1-c1 bin]#
```

The above screen depicts the following information:

- Device #00 represents the CPLD and the firmware version is 01.00.00
- Device #01 represents BIOS and the firmware version is 1.4.00000002

The following screen shows the typical output of `fcu -q` command on a PCIE -721x card.

```
[root@pcie7210-s15-c2 ~]# fcu -q
********************[[[[[REPORT BEGIN]]]]]********************
Operation: Query
Product Name: PCIE-7210

 #00 Device    : pcie721X-cpld
  Bank #0 -       Active Version: 01.06.01

 #01 Device    : pcie721X-cpu
  Bank #0 -       Active Version: 2.2.00000004

********************[[[[[ REPORT END ]]]]]********************
[root@pcie7210-s15-c2 ~]#
```

5.1.1.2 Show Operation

Show operation does not access any device. It only operates with the firmware image and it shows the metadata, which is part of the image. Furthermore, it validates the firmware image to compare the checksum part of the metadata against the checksum of the raw image. The output of the show operation is similar to the output of the query operation.

Show contents of BIOS firmware image

```
[root@pcie7210-s15-c2 bios]# fcu -sf pcie_721x-2_bios_2.2.4.fri
********************[[[[[REPORT BEGIN]]]]]********************
Operation: Show
 Manufacturer : ARTESYN
 Board        : pcie721X
 #00 Device   : pcie721X-cpu
  Bank #0 -              Version: 2.2.00000004

 #01 Device   : pcie721X-cpu
  Bank #0 -              Version: 2.2.00000004

********************[[[[[ REPORT END ]]]]]********************
[root@pcie7210-s15-c2 bios]#
```

5.1.1.3 Verification Operation

The verify operation is used to check that the image is valid for this product by comparing the metadata in the image with the current on-board Flash device.

Verification of BIOS image

```
[root@pcie7210-s15-c2 bios]# fcu -vf pcie_721x-2_bios_2.2.4.fri
********************[[[[[REPORT BEGIN]]]]]********************
Operation: Verify
Result   : Success
********************[[[[[ REPORT END ]]]]]********************
[root@pcie7210-s15-c2 bios]#
```

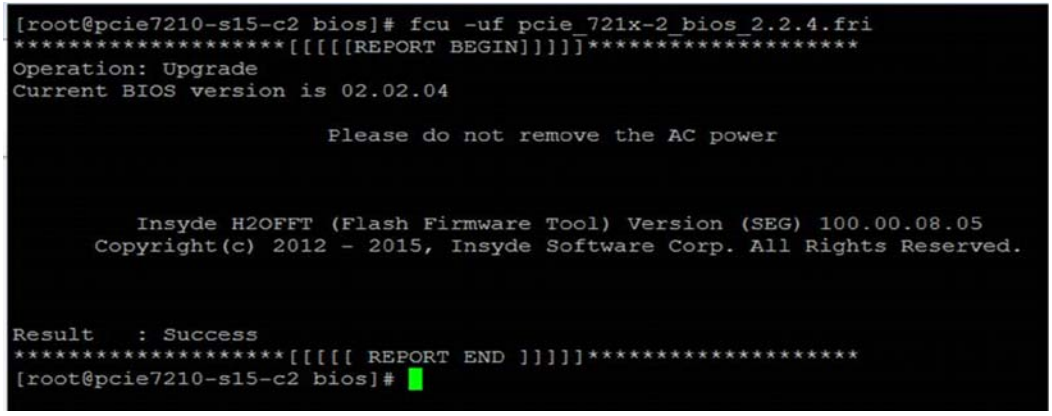
5.1.1.4 Upgrade Operation

The Upgrade operation uploads the firmware image to the device. Before the upload process, the firmware image is validated and the FCU verifies if the image is applicable to the firmware device.

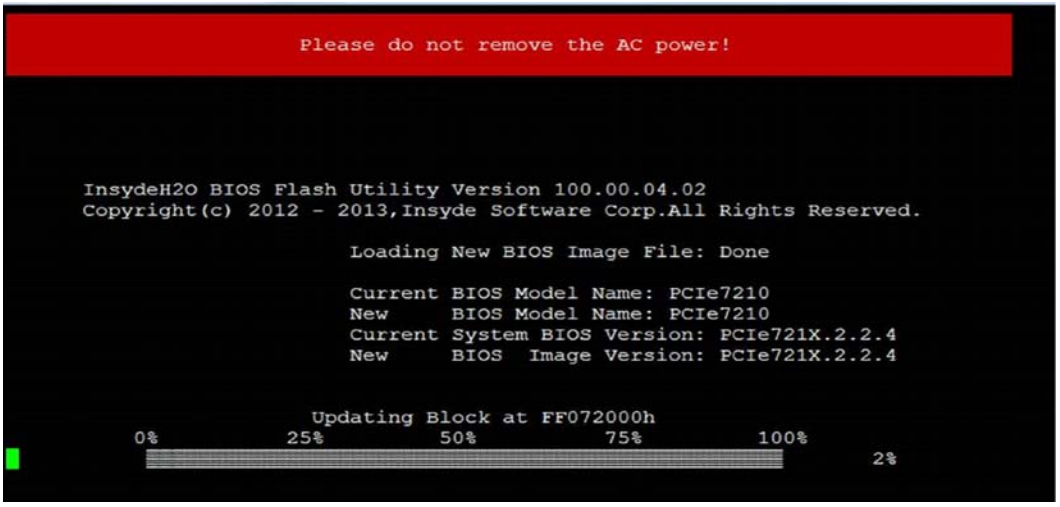
To upgrade the BIOS, use the following command:

$ **fcu -uf <BIOS image>.fri**

Here, *<BIOS image>.fri* is the firmware file to which BIOS will be upgraded. The following screen shows a typical output when the above command is executed.



After completion of this procedure, the image is copied to the flash temporary location. Once the upgrade procedure shows the result as **Success** (as notified in the above screen), the upgrade procedure would complete with the rebooting of the card. After the reboot, the following screen is displayed.



Wait until the above procedure is successful.

NOTICE

Do not reset or reboot the card at this point of time. This may corrupt the BIOS. Once the upgrade is successful, card will go for automatic reset and then boots with the upgraded BIOS.

After the card boots to Linux, to confirm whether the BIOS is upgraded, execute the `fcu –q` command.

This section contains screen shots of PCIE-7410 and PCIE-7210 cards as an example; the same command is applicable for the PCIE-9205 card.

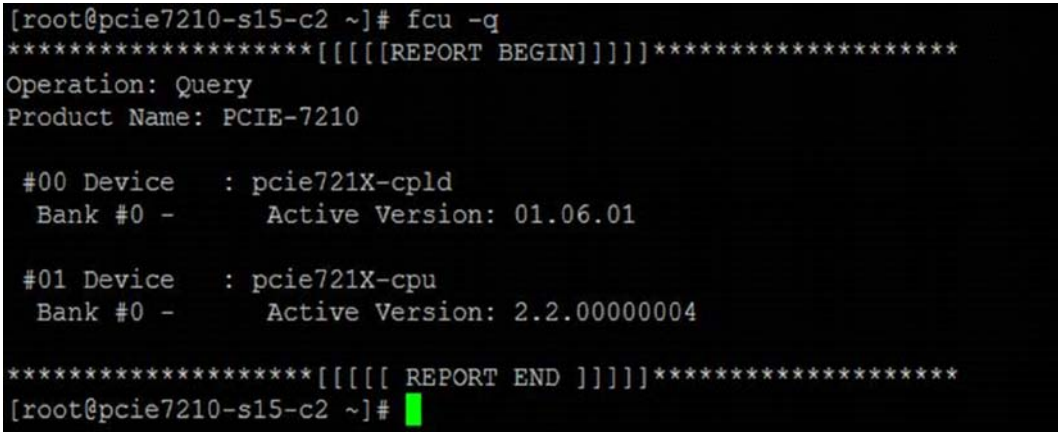
5.1.2 CPLD Upgrade

5.1.2.1 Query Operation

If you want know the current CPLD version before upgrading it with a new version, use the following command:

```
$ fcu -q
```

The following screen shows a typical output when the above command is executed.



5.1.2.2 Show Operation

To verify the CPLD firmware, use the following command:
`fcu-vf<Fri file image>`

The following screens show the typical outputs of the command when executed on the PCIE-721x card.

Show contents of CPLD firmware image

```
[root@pcie7210-s15-c2 cpld]# fcu -sf PCIe_721x-2_CPLD_1_6_0.fri
**********[[[[[REPORT BEGIN]]]]]**********
Operation: Show
 Manufacturer : ARTESYN
 Board        : pcie721X
 #00 Device   : pcie721X-cpld
  Bank #0 -             Version: 01.06.00

**********[[[[[ REPORT END ]]]]]**********
[root@pcie7210-s15-c2 cpld]#
```

Verify contents of CPLD firmware image

```
**********[[[[[ REPORT END ]]]]]**********
[root@pcie7210-s15-c2 ~]# cd /opt/bladeservices/rom/7210/cpld/
[root@pcie7210-s15-c2 cpld]# fcu -vf PCIe_721x-2_CPLD_1_6_0.fri
**********[[[[[REPORT BEGIN]]]]]**********
Operation: Verify
Result   : Success
**********[[[[[ REPORT END ]]]]]**********
[root@pcie7210-s15-c2 cpld]#
```

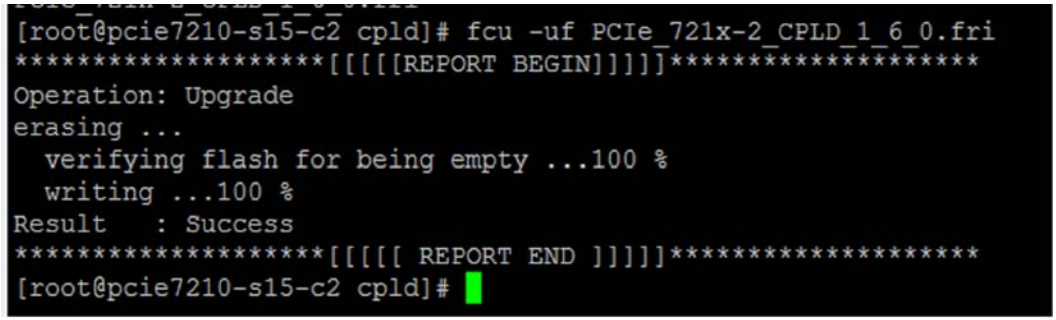
5.1.2.3 Upgrade Operation

To upgrade the CPLD, use the following command:

$ **`fcu -uf <CPLD image>.fri`**

Here, *<CPLD image>.fri* is the firmware file to which CPLD will be upgraded.

The following screen shows a typical output when the above command is executed.



NOTICE

For the newly upgraded CPLD image to be active, you need to PowerCycle the card.

After the card boots to Linux, to confirm whether the CPLD is upgraded, execute the `fcu -q` command.

Troubleshooting and FAQ

A.1 Overview

This section provides troubleshooting, frequently asked questions (FAQs), and their usual solutions on SSF for the MaxCore MC3000 platform.

A.2 Starting SSF

SSF starts automatically when the system restarts. The following sections describe how to start SSF Core, SSF Agent, Application agent, and SSF web interface manually.

A.2.1 Starting SSF Core

You can stop and start the SSF Services using the systemd scripts.

After you start the SSF Core, you can stop, restart, and check the status of the SSF using the following command:

```
#systemctl {stop|restart|status|enable|disable} ssfCore.service
```

A.2.2 Starting SSF Agent

You can stop, restart, and check the status of the SSF Agent using the following command:

```
#systemctl {stop|restart|status|enable|disable} ssfAgent.service
```

A.3 SSF Core Failure

Problem Description

SSF Core fails to start.

Root Cause and Solution

- PAM-postgres-SSF communication error. If so, wait for a while and retry starting SSF core.
- Radiusd service is not running
- User table in SSF got corrupted
- Insufficient persistence memory. Make sure sufficient physical disk space is available for database transactions being done by SSF.
- Meta.txt not is sync with the compiled SSF binaries - Make sure that `/opt/ssf/etc/config/main/meta.txt` is proper and not corrupted.

A.4 Host OS Not Displayed

Problem Description

Host OS is not visible under CPU in navigation pane.

Root Cause and Solution

- The CPU is not in the internal (base) network. Virtual functions from ETH3 (default base network) may not be assigned to the CPU. At least one VF from ETH3 should be assigned to the CPU.
- ssfAgent on that specific CPU is not running
- ssfAgent did not get shelf host IP address through mcparams
  - Verify `/opt/boardinfo/params` on that specific CPU to see if shelfhost IP address is populated
- If firewall is enabled on shelf host and or application host, then proper IP table rules should be added
- SSF discovery is in progress

A.5 Login Failure

Problem Description

Login fails.

Root Cause and Solution

- Login fails if wrong credentials are provided as input
- SSF failed to start or User table in SSF got corrupted

A.6 Switch Management Tab is Hidden

Problem Description

Switch management tab not seen.

Root Cause and Solution

See the reasons in *Host OS Not Displayed on page 66*.

A.7 Configuration Editor - Apply Failure

Problem Description

What are the various reasons for Configuration Editor - Apply Failure?

Root Cause and Solution

- If shelf ID of the shelf is different to what is present in the configuration
- If any of the IPMI or PEX commands fails
- Configuration file is corrupted
- If the PCIE card is PCIE-9205, then if any of RRC configurations set is failed
- If configuration is captured on a shelf which has different hardware configuration to the one on which it is applied

A.8 GUI Access and Logging Issues

Problem Description

User logged in after a long interval; Tree not getting loaded or not able to access GUI. It returns Error 500 or any http error.

Root Cause and Solution

- Linux may be responding slowly
- File system is corrupted

A.9 PCIE-9205 Switch Management is Not Populated in GUI

Problem Description

PCIE-9205 is defined as network CPU and PCIE-9205 Switch Management is not populated in GUI.

Root Cause and Solution

Perform the following steps:

1. Check if `172.27.<SHELF ID>.2` is reachable from PCIE-9205.

2. Check if PEP4 (ex. enp6s0) of PCIE-9205 is having DHCP IP on network 172.27.x.x.To find device name of PEP4, use the below commands. In this case, enp6s0 is the PEP4 interface.

```
# lspci -vv | grep "FM10000\|VP"
06:00.0 Ethernet controller: Intel Corporation Ethernet Switch FM10000
Host Interface
Product Name: FM1000
[VP] Vendor specific: 4
0a:00.0 Ethernet controller: Intel Corporation Ethernet Switch FM10000
Host Interface
Product Name: FM10000
[VP] Vendor specific: 8
# systool -c net
Class = "net"
Class Device = "enp0s20u2"
Device = "3-2:1.0"
Class Device = "enp10s0"
Device = "0000:0a:00.0"
Class Device = "enp4s0f0"
Device = "0000:04:00.0"
Class Device = "enp4s0f1"
Device = "0000:04:00.1"
Class Device = "enp6s0"
Device = "0000:06:00.0"
```

3. Check if DHCP client is running on PEP4 interface. emindTcpAddress and ShelfHostIPAddress in configuration file(s) of SSF BCSIM and SSF RRC TLS need to be set with the IP address of br0 on System Host. It would be of the format 172.27.<SHELFID>.2.

   For example:
   emindTcpAddress =172.27.44.2:21212 => This is to be modified in both rrc/ssfApi.conf and bcsim/ssfApi.conf
   ShelfHostIPAddress =172.27.44.2 => This is to be modified in bcsim/ssfApi.conf

   Note: Reboot PCIE-9205, if you have performed any changes on these files.

4. If an old configuration file is not loaded properly, ensure the following mentioned below:
   - Size of Port, VLAN, and Pool descriptions is less than 24 characters.
   - In the old configuration file, replace the trailing spaces at the end of each line using the below command in vim:
     %s/\s\+$//

A.10 Incorrect Device Id to PCIE-920x PEP Port Mapping

Problem Description

If the listed VF ports in the configuration interface are not matching as per the mapping listed by`/opt/switch_sw/etc/pcie9205_getpep.sh` script. This script lists the mapping between PEP devices and Red Rock Canyon (RRC) port, then there is a correction required in the mapping configuration file.

Root Cause and Solution

By default, all the four PEP devices are mapped in reverse to sw1p2* ports of RRC with EEPROM v10. However, if there is any change in the mapping or to confirm the mapping, copy “`/opt/switch_sw/etc/pcie9205_getpep.sh`” from PCIE-9205 to management CPU and run the script. This script will be listing the mapping between PEP devices and RRC ports.

If there is a different mapping between PEP devices and RRC ports, the correct mapping need to be updated in "`/opt/switch_sw/etc/pep_info.conf`" as below:

`<DEVICE ID> <RRC PORT>`

For example:

1. sw1p20
2. sw1p21
3. sw1p22
4. sw1p23

A.11 How to Check whether SSF Services are Running Fine

To confirm that SSF is running properly, run the following services in the following order:

1. `pciemgmt.service` – On shelfHost of SSF configured shelves
2. `mcagent.service` – On shelfHost of SSF configured shelves
3. `ssfCore.service` – On systemHost
4. `ssfAgent.service` – On all SSF configured hosts
5. `dhcpd.service` – On shelfHost of SSF configured shelves
6. `network.service` – On all SSF configured hosts

7. `ssfRRCAgent.service` – On all SSF configured network hosts
8. `zend-server.service` – On systemHost
9. `vsftpd.service` – On systemHost

Related Documentation

B.1 SMART Embedded Computing Documentation

The documentation listed is referenced in this manual. Technical documentation can be found by using the Documentation Search at *https://www.smartembedded.com/ec/support/* or you can obtain electronic copies of SMART EC documentation by contacting your local sales representative.

Table B-1 SMART EC Documentation

Document Title	Document Number
SSF for MaxCore™ MC3000 Platform XML Interface Guide	6806800T71
SSF for MaxCore™ MC3000 Platform Command Line Interface Guide	6806800T87
MaxCore™ MC3000 Platform Installation and Use	6806800T88
MaxCore™ MC3000 Platform Quick Start Guide	6806800T89
MaxCore™ MC3000 Platform Safety Notes Summary	6806800T90

Related Documentation