
***SmartFusion2 SoC FPGA PCIe Control
Plane Demo For Advanced Development
Kit - Libero SoC v11.5***

DG0566 Demo Guide

Superseded

January 2015



Revision History

Date	Revision	Change
29 January 2015	2	Second release
27 August 2014	1	First Release

Confidentiality Status

This is a non-confidential document.

Superseded

Table of Contents

Preface	4
About this document	4
Intended Audience	4
References	4
 SmartFusion2 SoC FPGA PCIe Control Plane Demo For Advanced Development Kit	5
Introduction	5
Design Requirements	6
Demo Design	7
Introduction	7
Demo Design Features	8
Demo Design Description	8
Setting up the Demo Design	11
Board Setup	12
Programming the Board	12
Connecting the Board to the Host PC	15
Running the Demo Design	17
Running the Demo Design on Windows	17
Running the Demo Design on Linux	29
Conclusion	38
Appendix 1: SmartFusion2 Advanced Development Kit Board	39
 A List of Changes	40
 B Product Support	41
Customer Service	41
Customer Technical Support Center	41
Technical Support	41
Website	41
Contacting the Customer Technical Support Center	41
Email	41
My Cases	42
Outside the U.S.	42
ITAR Technical Support	42

Preface

About this document

This demo is for SmartFusion[®]2 system-on-chip (SoC) field programmable gate array (FPGA) devices. It provides instructions on how to use the corresponding reference design.

Intended Audience

SmartFusion2 devices are used by:

- FPGA designers
- System-level designers

References

Microsemi Publications

The following references are used in this document:

- [SmartFusion2 Microcontroller Subsystem User Guide](#)
- [SmartFusion2 and IGLOO2 High Speed Serial Interfaces User Guide](#)

Refer to the following web page for a complete and up-to-date listing of SmartFusion2 device documentation: <http://www.microsemi.com/products/fpga-soc/soc-fpga/smartfusion2>

SmartFusion2 SoC FPGA PCIe Control Plane Demo For Advanced Development Kit

Introduction

The SmartFusion2 SoC FPGA devices integrate a fourth generation flash-based FPGA fabric and an ARM® Cortex®-M3 processor, along with high performance communication interfaces on a single chip. The SmartFusion2 high speed serial interface (SERDESIF) provides a fully hardened PCIe endpoint (EP) implementation and is compliant with PCIe Base Specification Revision 2.0 and 1.1. For more details, refer to the [SmartFusion2 and IGLOO2 High Speed Serial Interfaces User Guide](#).

The demo explains the SmartFusion2 embedded PCI Express feature and how this can be used as a low bandwidth control plane interface using the SmartFusion2 Advanced Development Kit Board.

The demo provides a simple design to access the SmartFusion2 PCIe EP from a Host PC. A GUI is provided for read and write access to the SmartFusion2 PCIe configuration space and memory space of BAR0 and BAR1. It also provides the Host PC device drivers for the SmartFusion2 PCIe EP. It can run on both Windows and Red Hat Linux operating system (OS).

Figure 1 shows the top-level block diagram for the PCIe control plane demo. The demo design uses a SmartFusion2 PCIe interface with a maximum link width of x4 to interface with a Host PC PCIe Gen 2 slot. If the Host PC does not support Gen 2 slot, the design automatically changes to Gen 1 slot. The SmartFusion2 microcontroller subsystem (MSS) GPIOs control the LEDs and switches on the SmartFusion2 Advanced Development Kit Board using the PCIe interface. The Host PC can also read memory and writes to the SmartFusion2 eSRAM through GUI. The Host PC can also be interrupted by using the push button on the SmartFusion2 Advanced Development Kit Board.

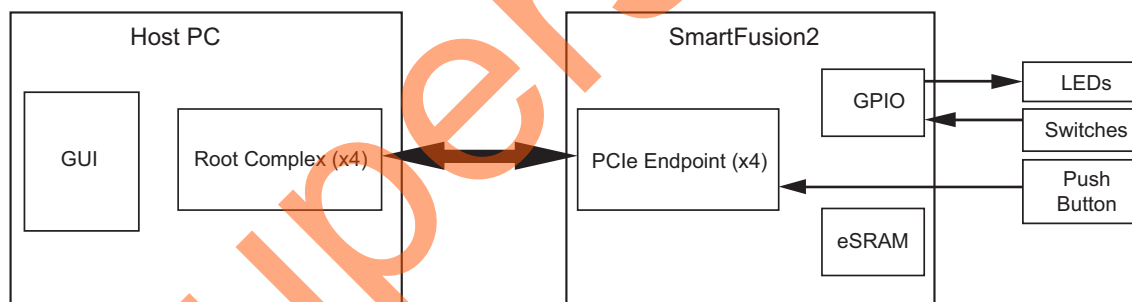


Figure 1 • PCIe Control Plane Demo Top-Level Block Diagram

Design Requirements

Table 1 lists the SmartFusion2 Advanced Development Kit Board design requirements details.

Table 1 • SmartFusion2 Advanced Development Kit Board Design Requirements

Design Requirements	Version
Hardware	
SmartFusion2 Advanced Development Kit Board <ul style="list-style-type: none">12 V adapterFlashPro5USB A to Mini-B cable	Rev A or later
Host PC with an available PCIe 2.0 Gen 1 or Gen 2 compliant slot	64-bit Windows 7 OS or 64-bit Red Hat Linux OS (Kernel Version: 2.6.18-308)
Software	
Libero [®] System-on-Chip (SoC) for viewing the design files <ul style="list-style-type: none">FlashPro Programming Software	v11.5
Host PC Drivers (provided along with the design files)	-
GUI executable (provided along with the design files)	-

Demo Design

Introduction

The design files for this demo can be downloaded from the Microsemi® website:
http://soc.microsemi.com/download/rsc/?f=m2s_dg0566_libero11p5_df

Design files include:

- LiberoProject
- ProgrammingFile
- Linux_64bit
- Windows_64bit
- Source Files
- Readme.txt

Figure 2 shows the top-level structure of the design files. For further details, refer to the [readme.txt](#) file.

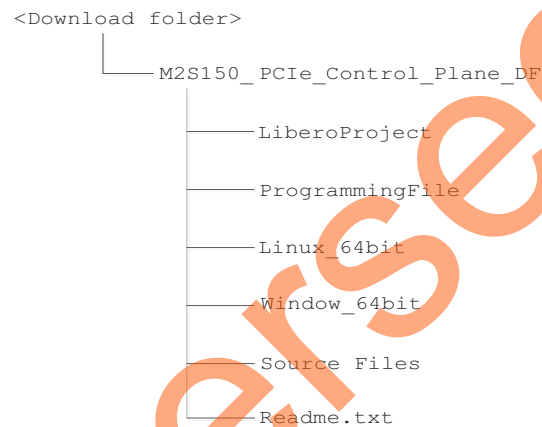


Figure 2 • Demo Design Files Top-Level Structure

Demo Design Features

The demo design performs the following tasks:

- Displays PCIe link enable/disable, negotiated link width, and the link speed.
- Controls the status of LEDs on the SmartFusion2 Advanced Development Kit Board
- Displays the position of DIP switches on the SmartFusion2 Advanced Development Kit Board
- Enables read and write to LSRAM
- Accepts and displays interrupts from the push button on the SmartFusion2 Advanced Development Kit Board
- Displays the SmartFusion2 PCIe Configuration space

Demo Design Description

The demo design helps to access the SmartFusion2 PCIe EP from the Host PC. Figure 3 shows a detailed block diagram of the design implementation.

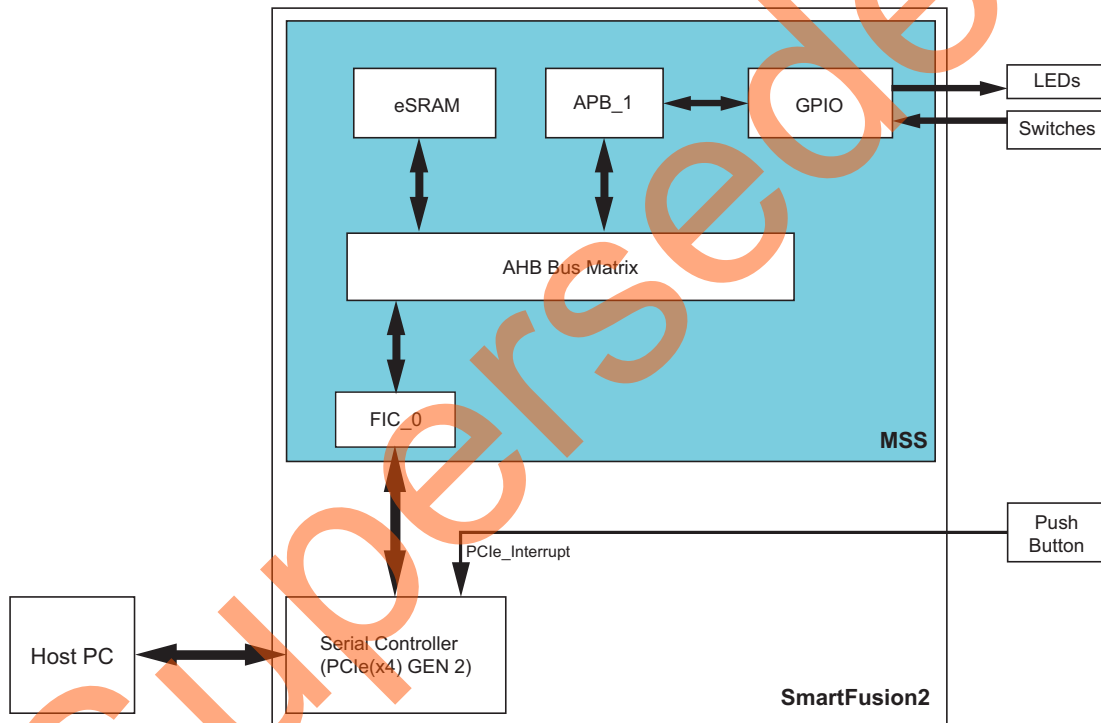


Figure 3 • PCIe Control Plane Demo Block Diagram

This demo design implements the SmartFusion2 embedded PCI Express interface as a low bandwidth control plane interface. This design provides Host PC drivers and a Host PC interface over PCIe to control the SmartFusion2 device. Figure 3 shows a detailed block diagram of the design implementation. The PCIe EP device receives commands from the Host PC through GUI and does corresponding memory writes to the SmartFusion2 MSS address space. The MSS address space provides a GPIO block and eSRAM memory block, which is accessed through a fabric interface controller (FIC_0).

The SERDES_IF_0 is configured for a PCIe 2.0, x4 link width with GEN2 speed for SmartFusion2 Advanced Development Kit Board. The PCIe interface to the fabric uses an AMBA High-speed Bus (AHB). The AHB master interface of SERDESIF is enabled and connected to the AHB slave interface of FIC_0 to access the MSS peripherals. The SmartFusion2 PCIe BAR0 and BAR1 are configured in 32-bit memory mapped memory mode.

The AXI master windows of the SERDESIF PCIe provide address translation for accessing one address space from another address space as the PCIe address is different from SmartFusion2 AHB bus matrix address space. The AXI master window 0 is enabled and configured to translate the BAR0 memory address space to the MSS GPIO address space to control the MSS GPIOs. The AXI master window 1 is enabled and configured to translate the BAR1 memory address space to the eSRAM address space to perform read and writes from PCIe.

MSS GPIO block is enabled and configured as below:

- GPIO_0 to GPIO_7 as outputs and connected to LEDs
- GPIO_8 to GPIO_11 as inputs and connected to DIP switches

The PCIe interrupt line is connected to the SW1 push button on the SmartFusion2 Advanced Development Kit. The FPGA clocks are configured to run the FPGA fabric and MSS at 100 MHz.

Simulating the Design

The design supports the BFM_P PCIe simulation level to communicate with the High Speed Serial Interface block through the master AXI bus interface. Though, the serial communication does not actually go through the High Speed Serial Interface block, this scenario allows validating the fabric interface connections. The *SERDESIF_0_user.bfm* file under the *<LiberoProject>/simulation* folder contains the BFM commands to verify the read or write access to MSS GPIOs and eSRAM.

BFM commands added in the *SERDESIF_0_user.bfm* file do the following:

- Write to GPIO_OUT[7:0]
- Write to LSRAM
- Read-check from LSRAM

To run the simulation, double-click **Simulate** under **Verify Pre-Synthesized Design** in the **Design Flow** window of the Libero project. ModelSim runs the design for about 250 us. The ModelSim **Transcript** window displays the BFM commands and the BFM simulation completed with no errors, as shown in Figure 4.



```

# BFM:43:write w 20000004 87654321 at 233550 ns
# BFM: Data Write 40013088 00000080
# BFM:44:write w 20000008 9abcdef0 at 233850 ns
# ESRAM0: Word Write 00000000=12345678 at 233950 ns
# BFM: Data Write 20000000 12345678
# BFM:45:write w 2000000c 0fedcba9 at 234050 ns
# ESRAM0: Word Write 00000004=87654321 at 234150 ns
# BFM: Data Write 20000004 87654321
# BFM:47:readcheck w 20000000 12345678 at 234250 ns
# ESRAM0: Word Write 00000008=9abcdef0 at 234350 ns
# BFM: Data Write 20000008 9abcdef0
# ESRAM0: Word Write 0000000c=0fedcba9 at 234550 ns
# BFM: Data Write 2000000c 0fedcba9
# BFM:48:readcheck w 20000004 87654321 at 234650 ns
# ESRAM0: Word Read 00000000=12345678 at 234700 ns
# BFM: Data Read 20000000 12345678 MASK:ffffff at 234800.010000ns
# ESRAM0: Word Read 00000004=87654321 at 234900 ns
# SFM: Data Read 20000004 87654321 at 235000.010000ns
# BFM:49:readcheck w 20000008 9abcdef0 at 235050 ns
# ESRAM0: Word Read 00000004=87654321 at 235100 ns
# BFM: Data Read 20000004 87654321 MASK:ffffff at 235200.010000ns
# ESRAM0: Word Read 00000008=9abcdef0 at 235300 ns
# SFM: Data Read 20000008 9abcdef0 at 235400.010000ns
# BFM:50:readcheck w 2000000c 0fedcba9 at 235450 ns
# ESRAM0: Word Read 00000008=9abcdef0 at 235500 ns
# BFM: Data Read 20000008 9abcdef0 MASK:ffffff at 235600.010000ns
# ESRAM0: Word Read 0000000c=0fedcba9 at 235700 ns
# SFM: Data Read 2000000c 0fedcba9 at 235800.010000ns
# BFM:52:return
# ESRAM0: Word Read 0000000c=0fedcba9 at 235900 ns
# BFM: Data Read 2000000c 0fedcba9 MASK:ffffff at 236000.010000ns
# ESRAM0: Word Read 00000010=xxxxxxxx at 236100 ns
# SFM: Data Read 20000010 0xxxxxxxxx at 236200.010000ns
#####
#
# SERDESIF_0 BFM Simulation Complete - 28 Instructions - NO ERRORS
#
#####
[VSIM 2>]

```

Figure 4 • SERDES BFM Simulation

Figure 5 shows the **Wave** window with GPIO_OUT signals.

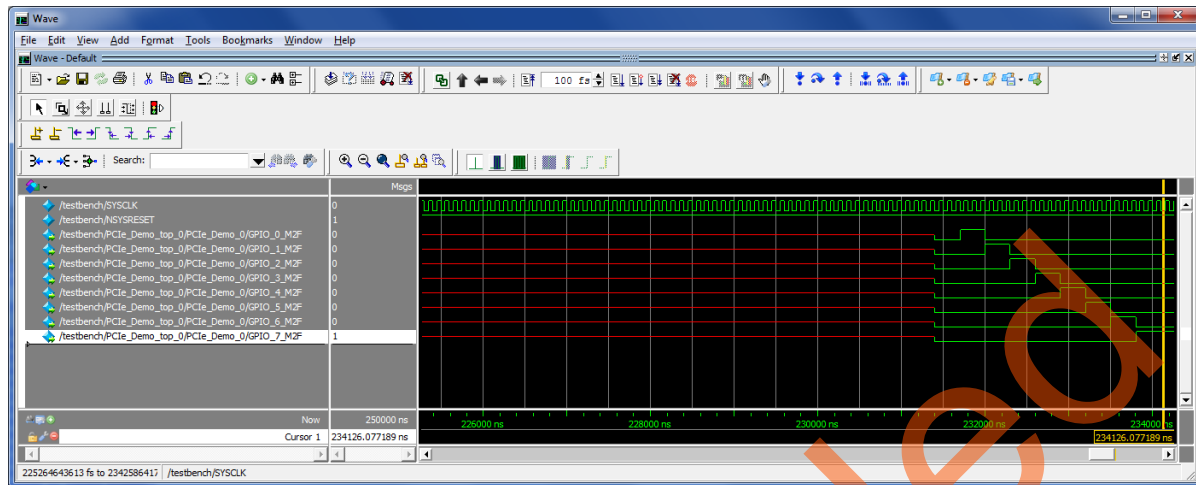


Figure 5 • Simulation Result with GPIO_OUT Signals

Setting up the Demo Design

The following steps describe how to setup the demo for SmartFusion2 Advanced Development Kit Board:

1. Connect the Host PC to the J33 Connector using the USB A to mini-B cable. The USB to UART bridge drivers are automatically detected. Verify, if the detection is made in the device manager as shown in Figure 6.

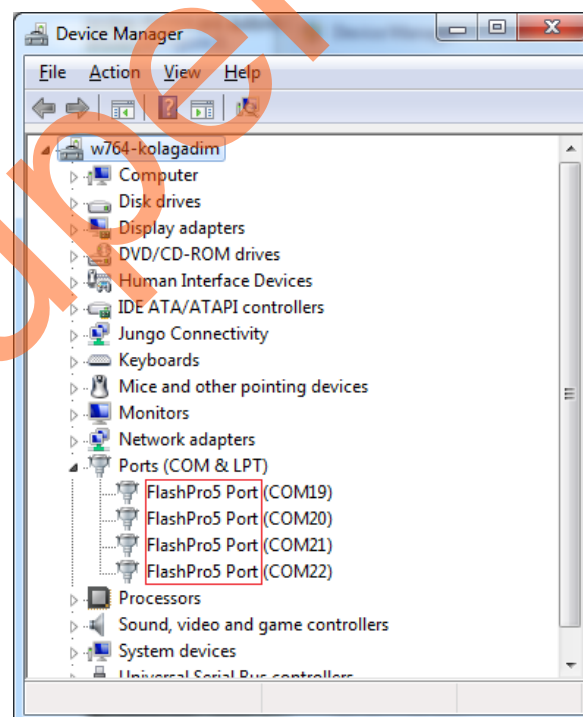


Figure 6 • Device Manager

2. Connect the jumpers on the SmartFusion2 Advanced Development Kit Board as shown in [Table 2](#).

CAUTION: While making the jumper connections, the power supply switch **SW7** on the board should be in OFF position.

Table 2 • SmartFusion2 FPGA Advanced Kit Jumper Settings

Jumper	Pin (from)	Pin (to)	Comments
J116, J353, J354, J54	1	2	These are the default jumper settings of the Advanced Dev Kit Board. Make sure these jumpers are set accordingly.
J123	2	3	
J124, J121, J32	1	2	JTAG programming via FTDI

3. Connect the power supply to the J42 Connector on the SmartFusion2 Advanced Development Kit Board.

Board Setup

Snapshots of the SmartFusion2 Advanced Development Kit Board with the complete set up is given in the ["Appendix 1: SmartFusion2 Advanced Development Kit Board"](#) on page 39.

Programming the Board

The following steps describe how to program the board.

1. Download the demo design from:
http://soc.microsemi.com/download/rsc/?f=m2s_dg0566_libero11p5_df
2. Switch **ON** the **SW7** power supply switch.
3. Launch the **FlashPro** software.
4. Click **New Project**.

5. In the **New Project** window, enter the **Project Name** as PCIe_Control_Plane.

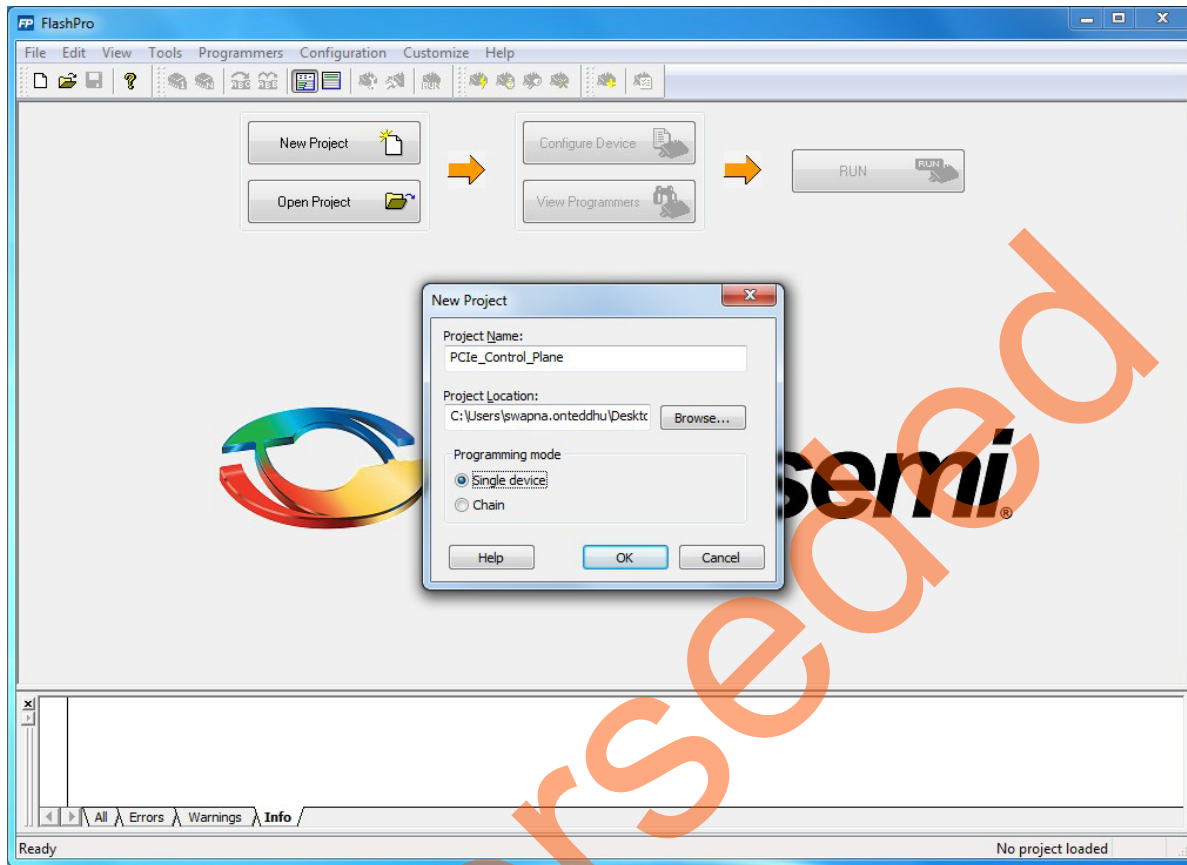


Figure 7 • FlashPro New Project

6. Click **Browse** and navigate to the location where you want to save the project.
7. Click **Single device** as the **Programming mode**.

8. Click **OK** to save the project.

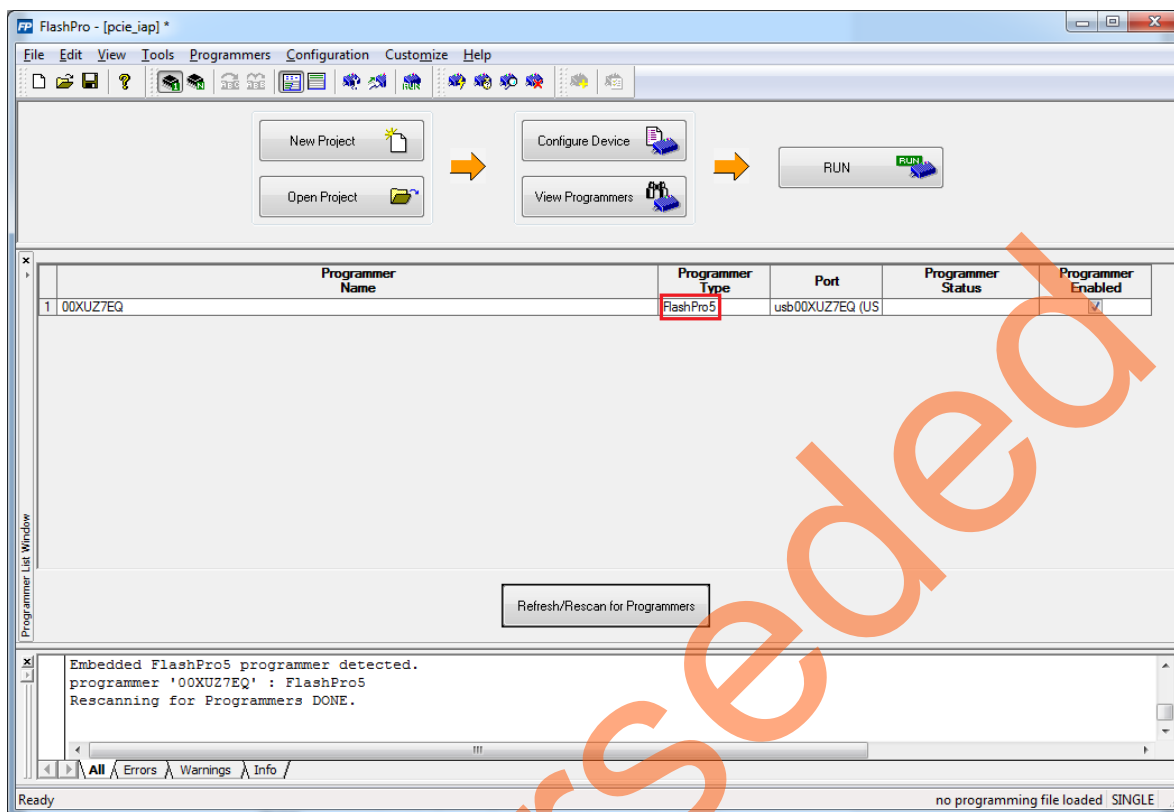


Figure 8 • FlashPro5 Programmer Type

9. Click **Configure Device** on the FlashPro GUI.
10. Click **Browse** and navigate to the location where the `PCIE_Demo_top.stp` file is located and select the file. The location for SmartFusion2 Advanced Development Kit Board is:
`<download_folder>\M2S150_PClE_Control_Plane_DF\programmingFile\SF2_Advanced_Dev_Kit.`

11. Click **Open**. The required programming file is selected and is ready to be programmed in the device.

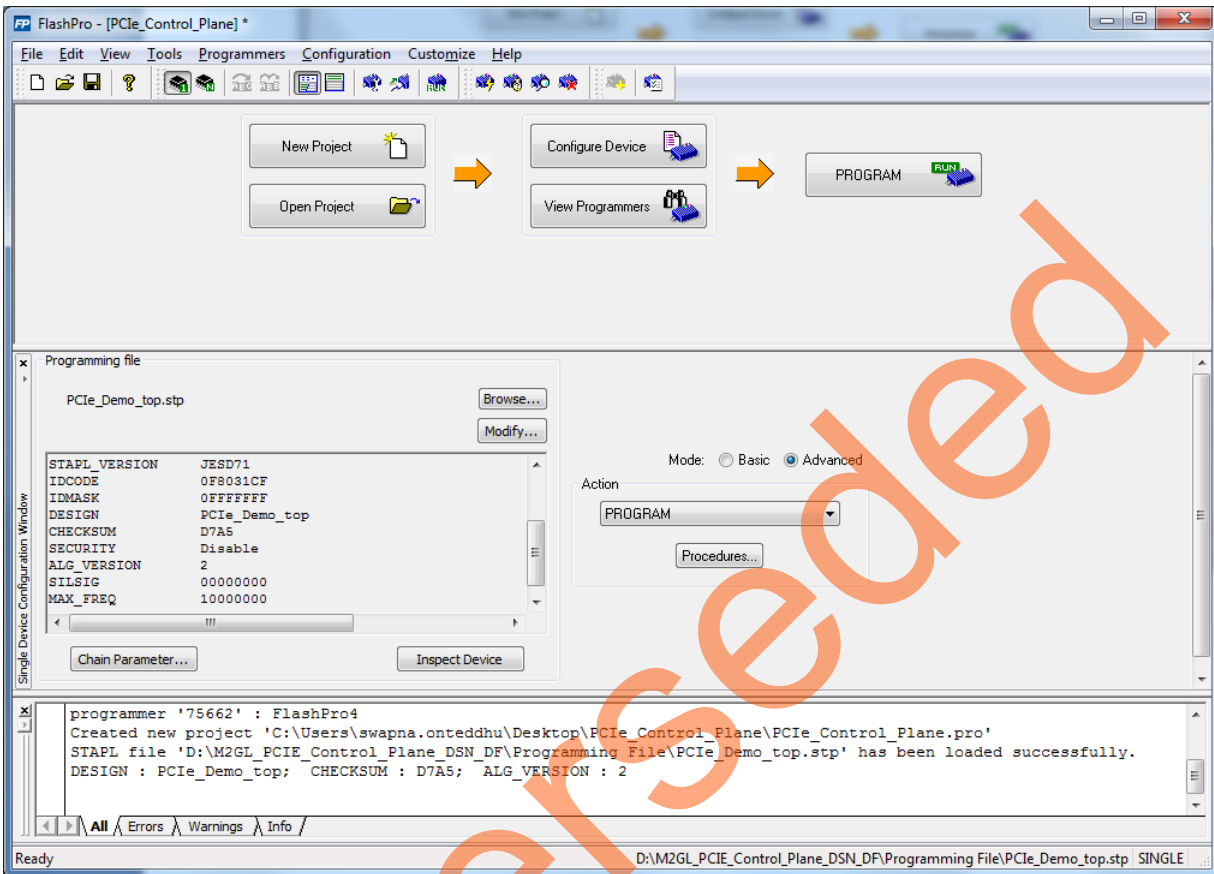


Figure 9 • FlashPro Project Configured

12. Click **PROGRAM** to start programming the device. Wait until a message is displayed indicating that the **PROGRAM PASSED**.

Connecting the Board to the Host PC

The following steps describe how to connect the board to the Host PC.

1. After successful programming, power **OFF** the SmartFusion2 Advanced Kit Board and shut down the Host PC.
This demo is designed to run in any PCIe Gen 2 compliant slot. If the Host PC does not support Gen 2 compliant slot, the demo switches to Gen 1 slot.
2. Connect the **CON1 - PCIe Edge Card Ribbon** cable to **Host PC PCIe Gen 2** slot or **Gen 1** slot as applicable.

CAUTION: Host PC must be powered OFF while inserting the PCIe Edge connector. If it is not, the PCIe device detection and selection of Gen 1 or Gen 2 slot may not occur properly. This is very dependent on the Host PC PCIe configuration. It is recommended that the Host PC is powered OFF before inserting the PCIe card.

Figure 10 shows the board setup for the Host PC in which SmartFusion2 Advanced Kit Board is connected to the Host PC PCIe slot.

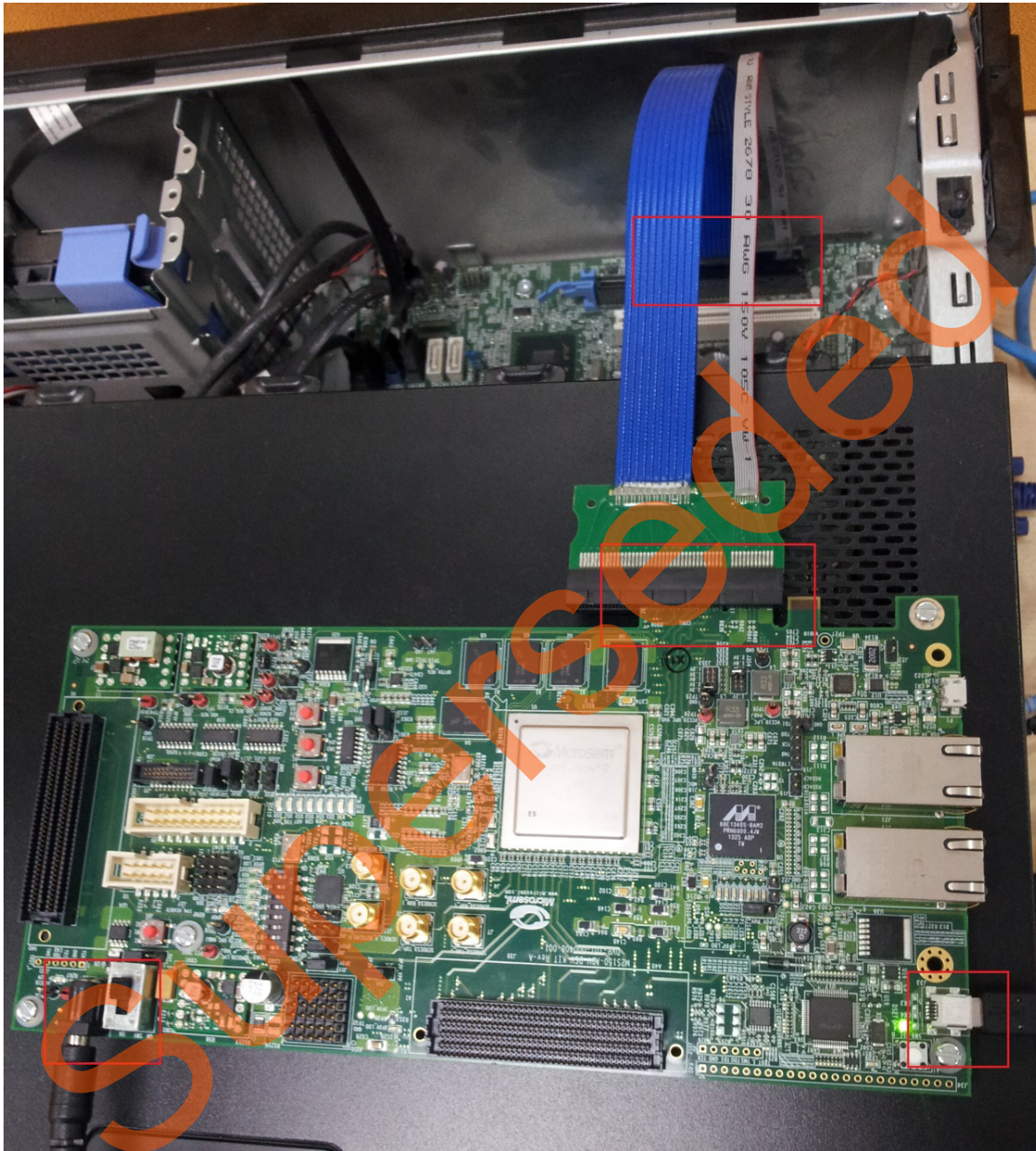


Figure 10 • SmartFusion2 Advanced Development Kit Setup for Host PC

Running the Demo Design

This demo can run on both Windows and RedHat Linux OS.

- To run the demo on Windows OS GUI, Jungo drivers are provided. Refer to ["Running the Demo Design on Windows" on page 17](#).
- To run the demo on Linux OS, native RedHat Linux drivers and command line scripts are provided. Refer to ["Running the Demo Design on Linux" on page 29](#)

Running the Demo Design on Windows

The following steps describe how to run the demo design on windows,

1. Switch **ON** the power supply switch, **SW7**.
2. Power on the Host PC and open the Host PC Device Manager for PCIe device, as shown in [Figure 11](#). If the PCIe device is not detected, power cycle the SmartFusion2 Advanced Development Kit Board. Right-click **PCIe Device** > **Scan for hardware changes** in Device Manager.

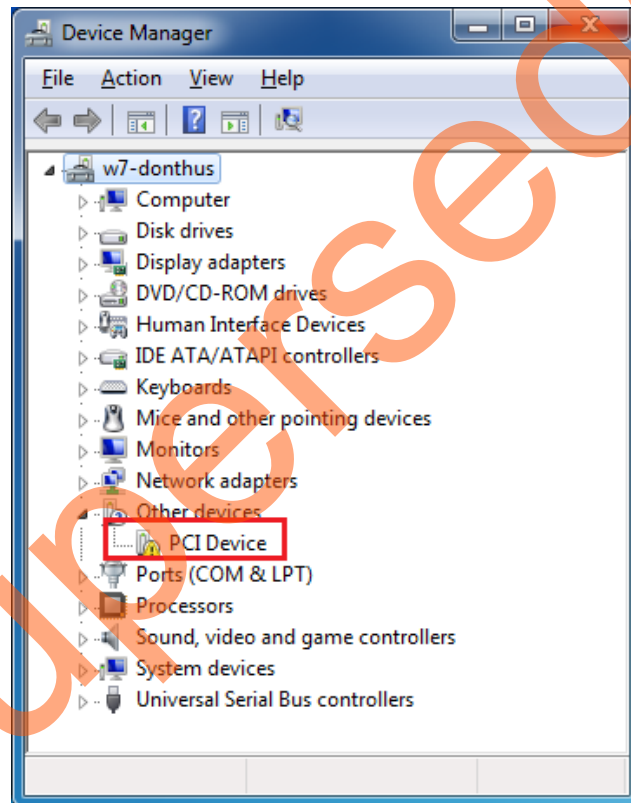


Figure 11 • Device Manager

Note: If the device is still not detected, check whether or not the BIOS version in Host PC is the latest, and if PCIe is enabled in the Host PC BIOS.

If the Host PC has any other installed drivers (previous versions of Jungo drivers) for the SmartFusion2 PCIe device, uninstall them.

The following steps describe how to uninstall previous versions of Jungo drivers:

- a. Navigate to device manager and right-click **DEVICE** and select **Uninstall** as shown in Figure 12. The **Confirm Device Uninstall** dialog box is displayed.

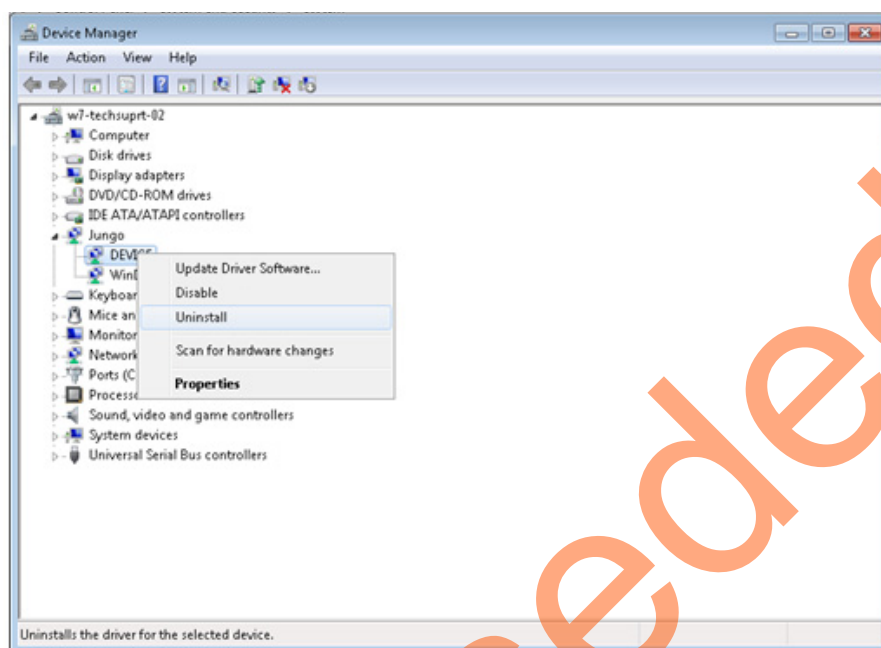


Figure 12 • Device Uninstall

- b. Select the **Delete the driver software for this device** check box as shown in Figure 13.
- c. Click **OK**.

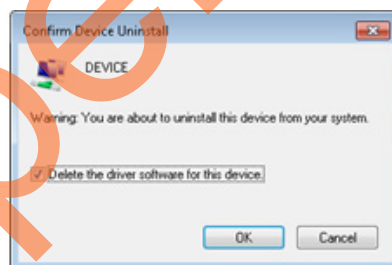


Figure 13 • Confirm Device Uninstall

After uninstalling previous Jungo drivers, make sure that the PCIe device is detected in the **Device Manager** window as shown in Figure 11.

Drivers Installation

The PCIe Demo uses a driver framework provided by Jungo WinDriverPro. To install the PCIe drivers on Host PC for SmartFusion2 Advanced Development Kit Board, use the following steps:

1. Extract the **PCle_Demo.rar** to C:\ drive. The **PCle_Demo.rar** is located in the provided design files:
 - M2S150_PcIe_Control_Plane_DF\Windows_64bit\Drivers\PCle_Demo.rar

Note: Installing these drivers requires the Host PC administration rights.

2. Run the batch file **C:\PCle_Demo\DriverInstall\Jungo_KP_install.bat**.

3. Click **Install**, if the window is displayed as shown in Figure 14.

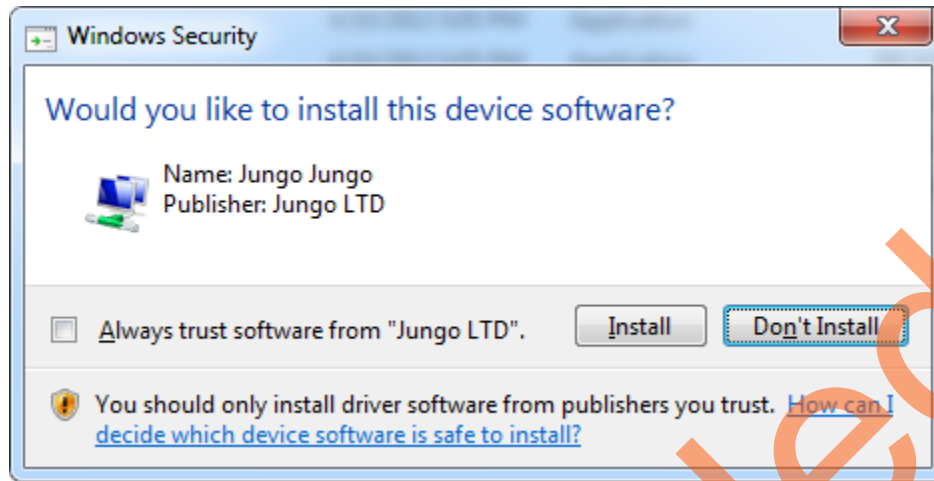


Figure 14 • Jungo Driver Installation

Note: If the installation is not in progress, right-click on the command prompt and select Run as administrator. Run the batch file `C:\PCIe_Demo\DriverInstall\Jungo_KP_install.bat` from command prompt.

4. Click **Install this driver software anyway** if the window appears as shown in Figure 15.

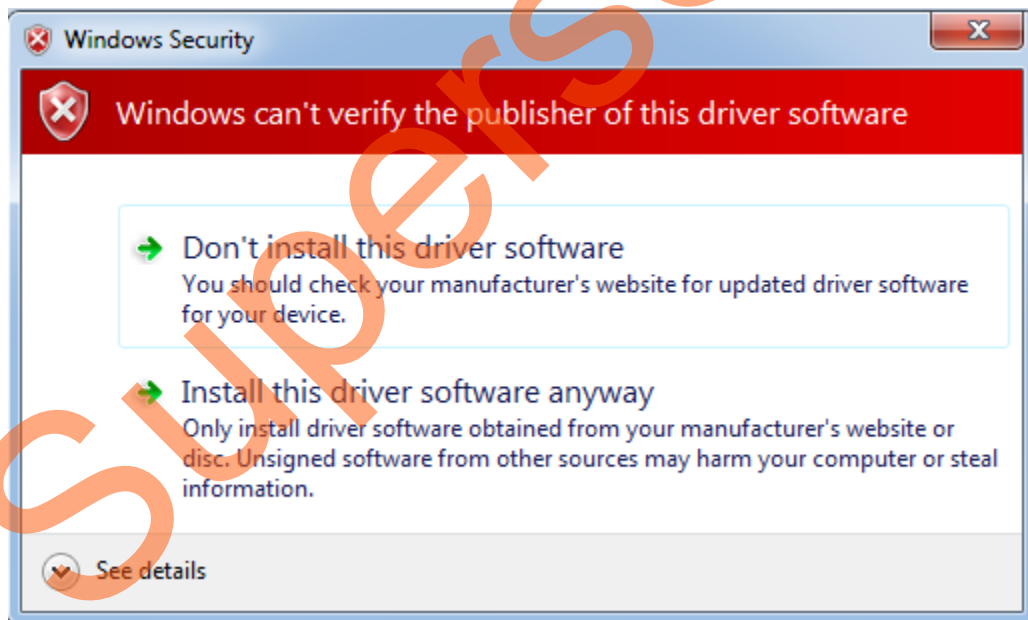


Figure 15 • Windows Security

PCIe Demo GUI Installation

The SmartFusion2 PCIe demo GUI is a simple GUI that runs on the Host PC to communicate with the SmartFusion2 PCIe EP device. The GUI provides the PCIe link status, driver information, and demo controls. The GUI invokes the PCIe driver installed on the Host PC and provides commands to the driver according to the user selection.

Use the following steps to install the GUI:

1. Download the PCIe demo GUI installer from
http://soc.microsemi.com/download/rsc/?f=PCIe_Demo_GUI_Installer
2. Extract the **PCIe_Demo_GUI_Installer.rar**.
3. Double-click the **setup.exe** in the provided GUI installation
(*PCIe_Demo_GUI_Installer\setup.exe*). Apply default options as shown in Figure 16.

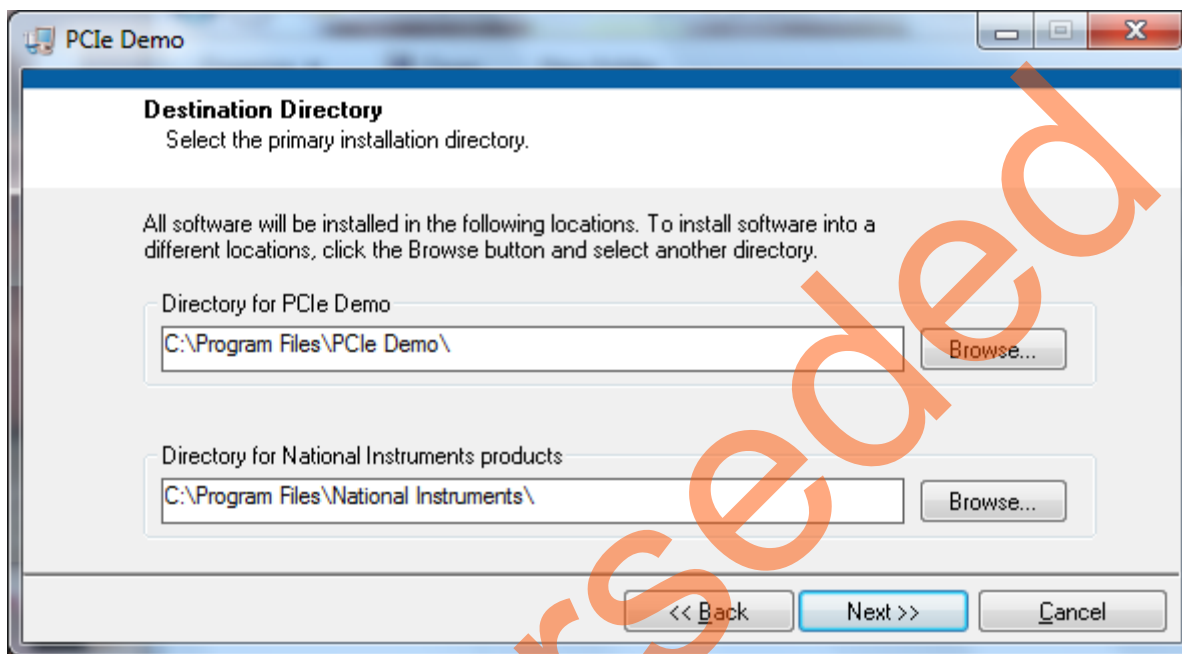


Figure 16 • GUI Installation

- Click **Next** and **Finish** to complete the installation. Figure 17 shows the **Successful GUI Installation** window.

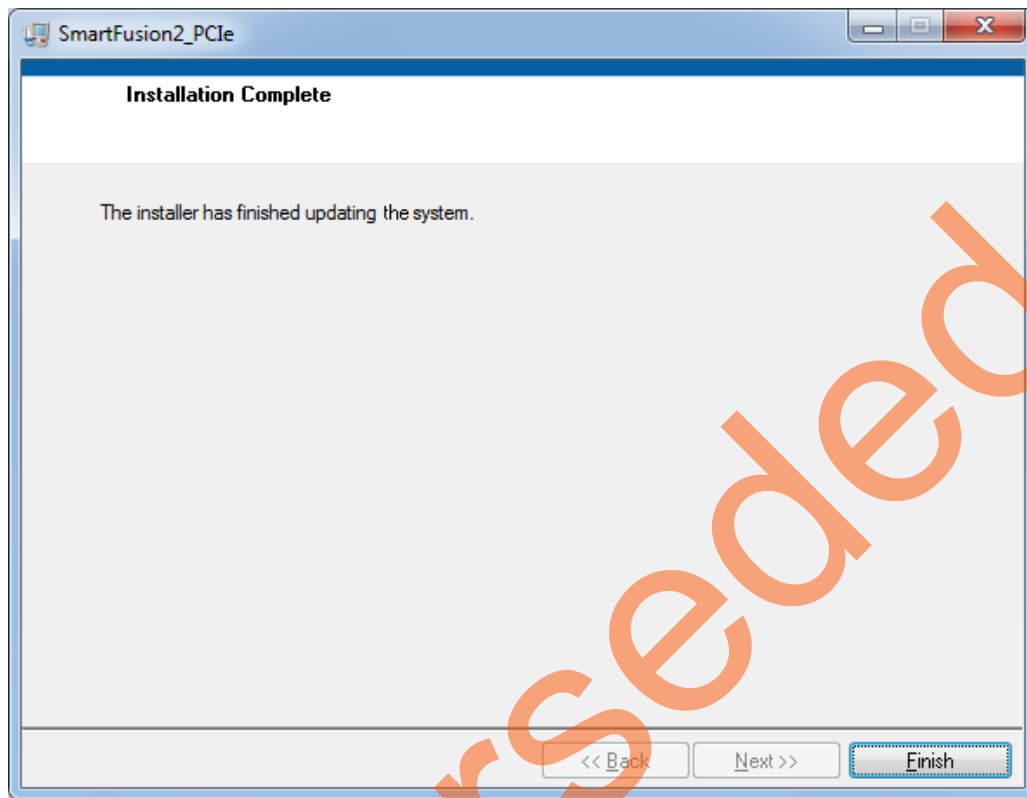


Figure 17 • Successful GUI Installation

- Restart the Host PC.

Running the PCIe GUI

The following steps describe how to run the PCIe GUI.

1. Check the Host PC Device Manager for the drivers. If the device is not detected, power cycle the SmartFusion2 Advanced Development Kit Board.
2. Right-click **DEVICE** > **Scan for hardware changes** in Device Manager. Ensure that the board is switched ON.

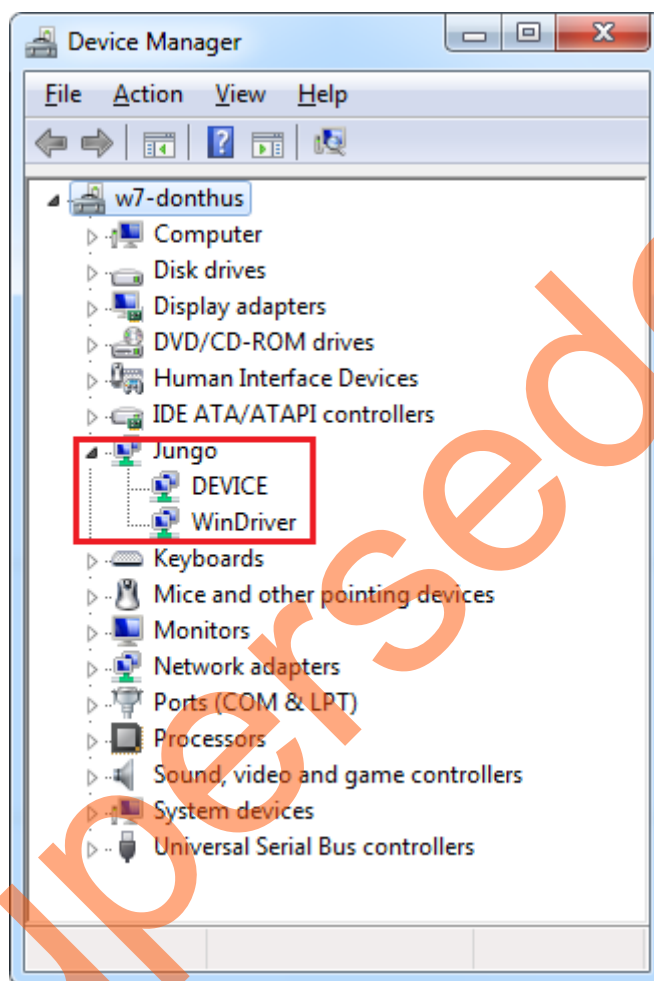


Figure 18 • Device Manager - PCIe Device Detection

Note: If a warning symbol is displayed on the **DEVICE** or **WinDriver** icon in the **Device Manager**, uninstall them and start from step 1 of "Drivers Installation" on page 29.

3. Invoke the GUI from **ALL Programs > PCIeDemo > PCIe Demo GUI**. Figure 19 shows the **PCIe Demo GUI** window.

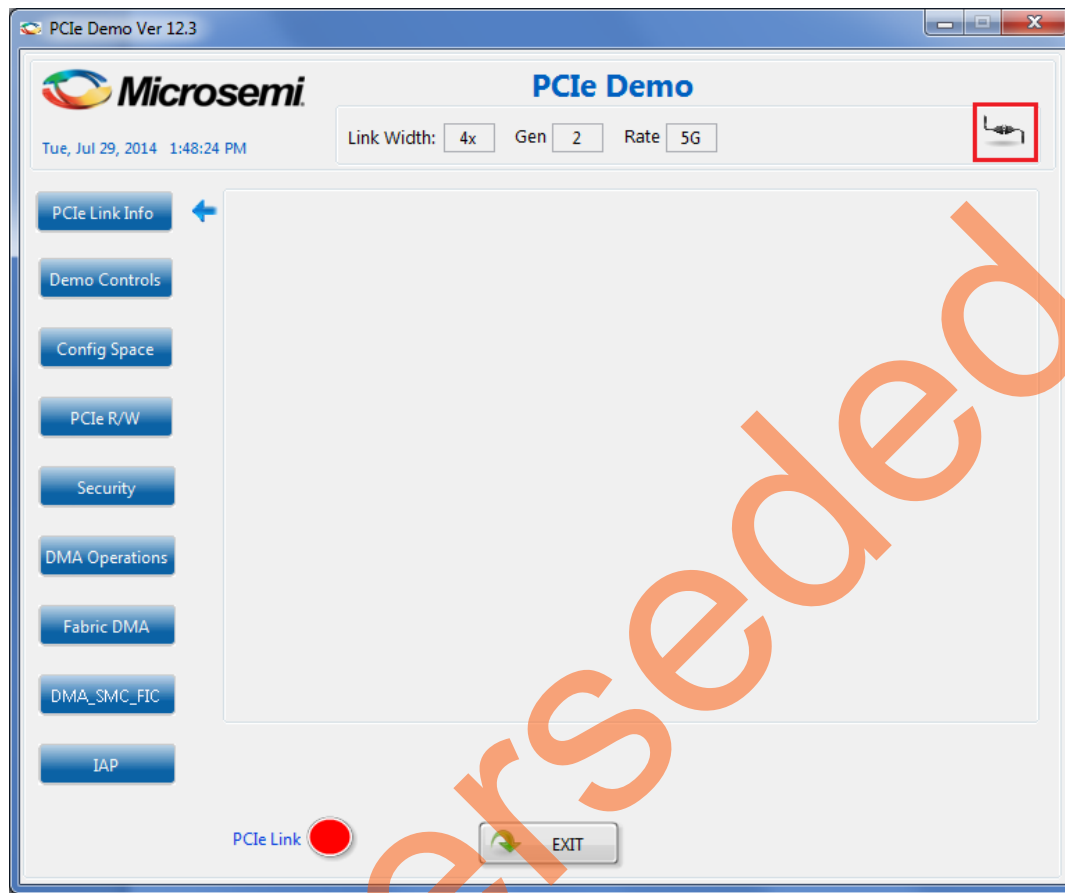
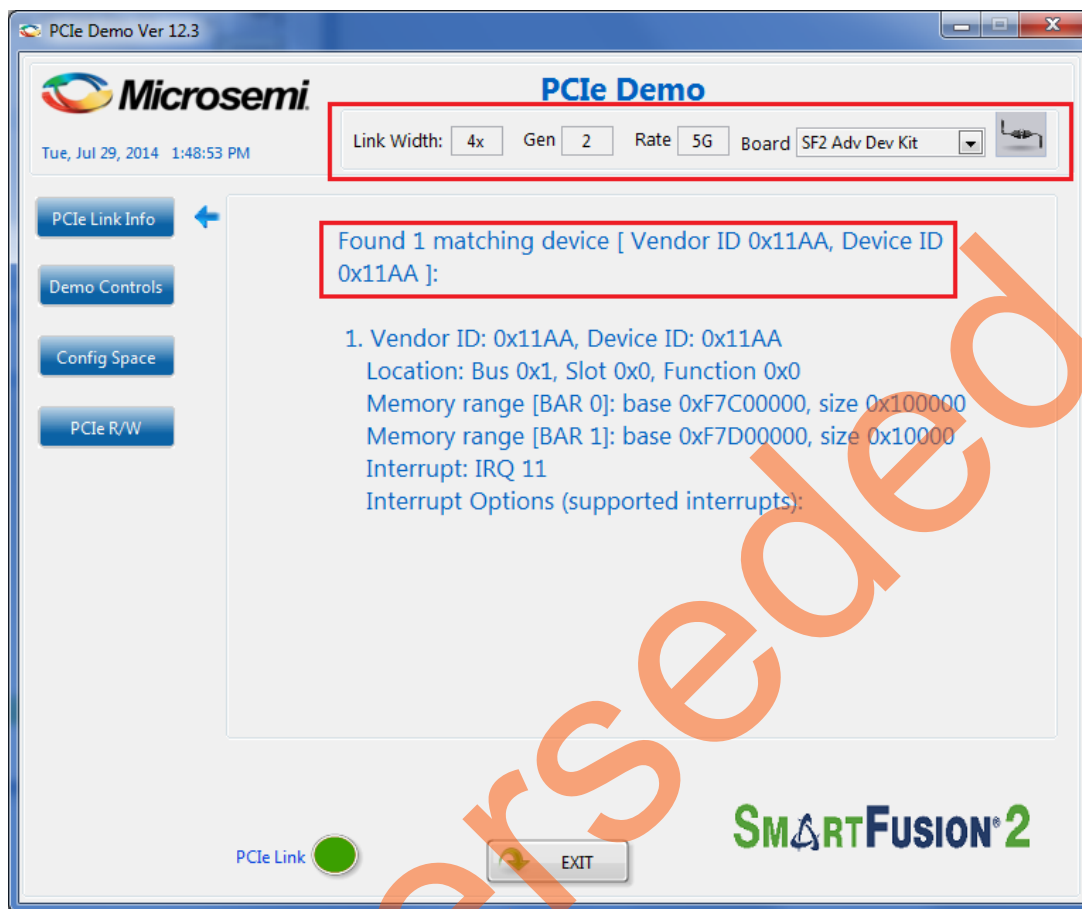


Figure 19 • PCIe Demo GUI

4. Click **Connect** icon highlighted in Figure 19. The Link Width, Gen, Rate, and Type of Kit are displayed on the GUI as shown in Figure 20 on page 24.

**Figure 20 • Version Information**

Note: If the Host PC does not support Gen 2 slot, the design automatically changes to Gen 1 slot.

5. Click **Demo Controls**. Figure 21 shows the LED options and DIP switch status.

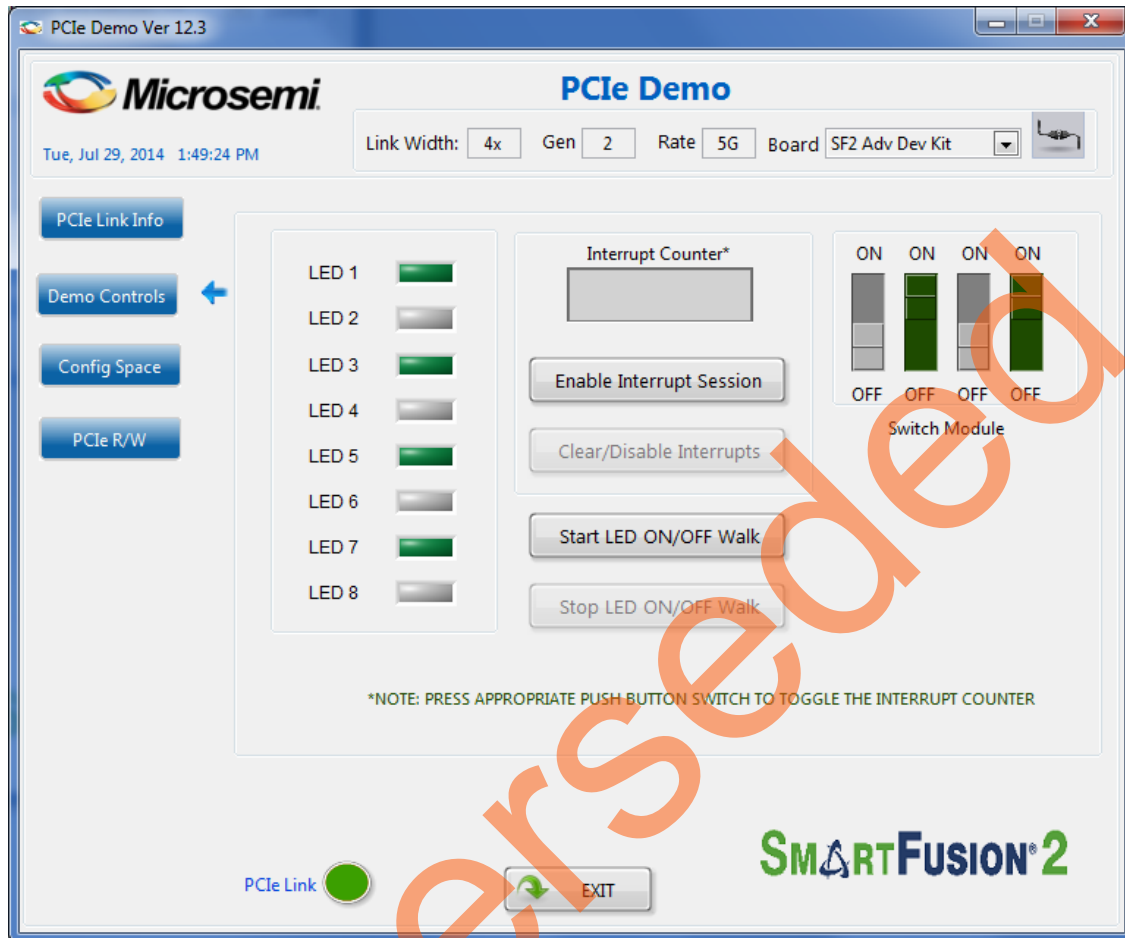


Figure 21 • Demo Controls

6. Click LEDs on GUI to **ON/OFF** the LEDs on the SmartFusion2 Advanced Development Kit Board.
7. Click **Start LED ON/OFF Walk** to blink the LEDs on SmartFusion2 Advanced Development Kit Board.
8. Click **Stop LED ON/OFF Walk** to stop the LEDs blinking.
9. Change the DIP switch positions (**1 to 4**) on the SmartFusion2 Advanced Development Kit Board (**SW5**) and observe the similar position of switches in **GUI SWITCH MODULE**.
10. Click **Enable Interrupt Session** to enable the PCIe interrupt.

11. Press the push button **SW1** on the SmartFusion2 Advanced Development Kit Board and observe the interrupt count in the **Interrupt Counter** field, as shown in [Figure 22](#).

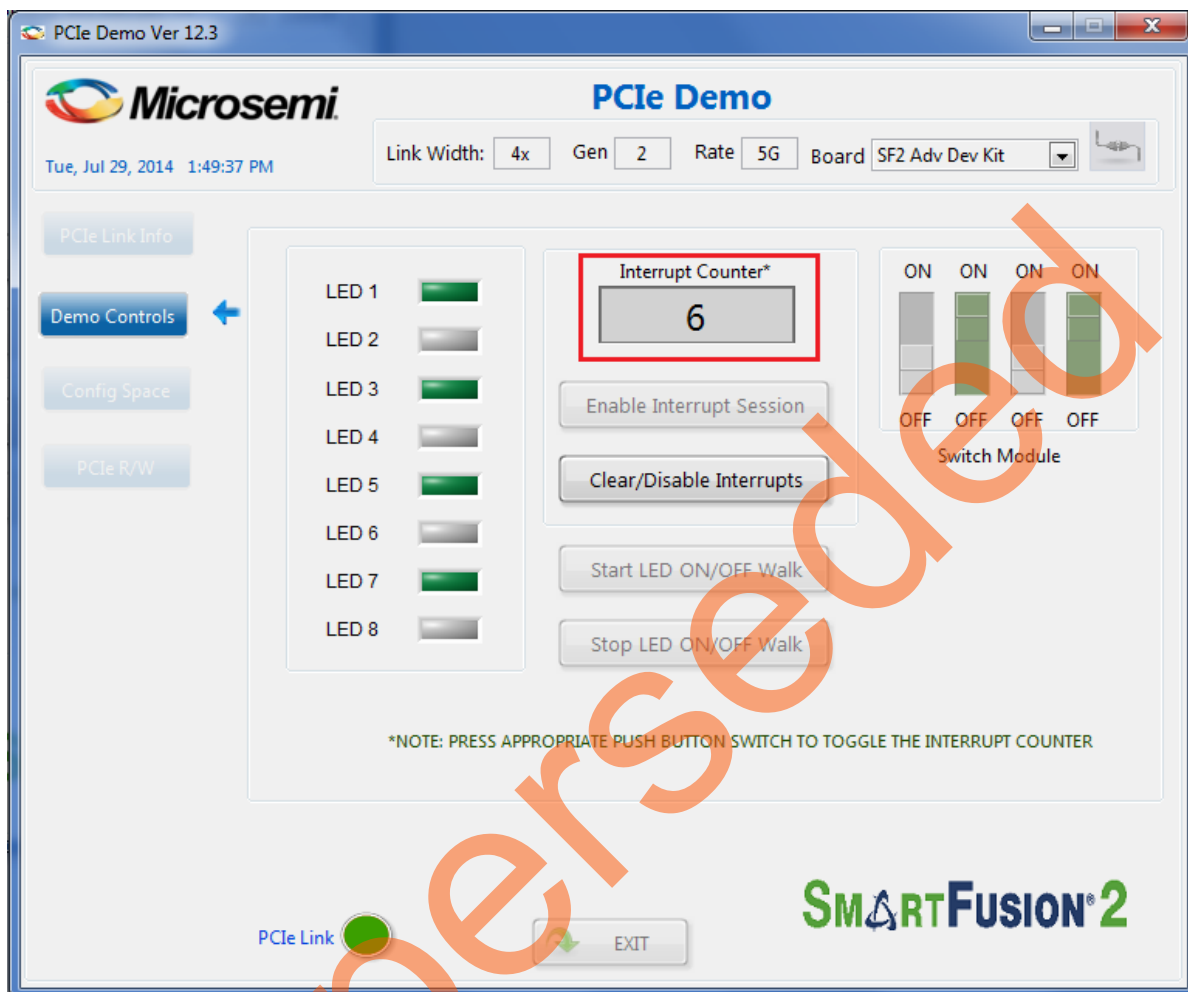


Figure 22 • Interrupt Counter

12. Click **Clear/Disable Interrupts** to clear and disable the PCIe interrupts.

13. Click **Config Space** to read details about the PCIe configuration space. Figure 23 shows the PCIe configuration space.

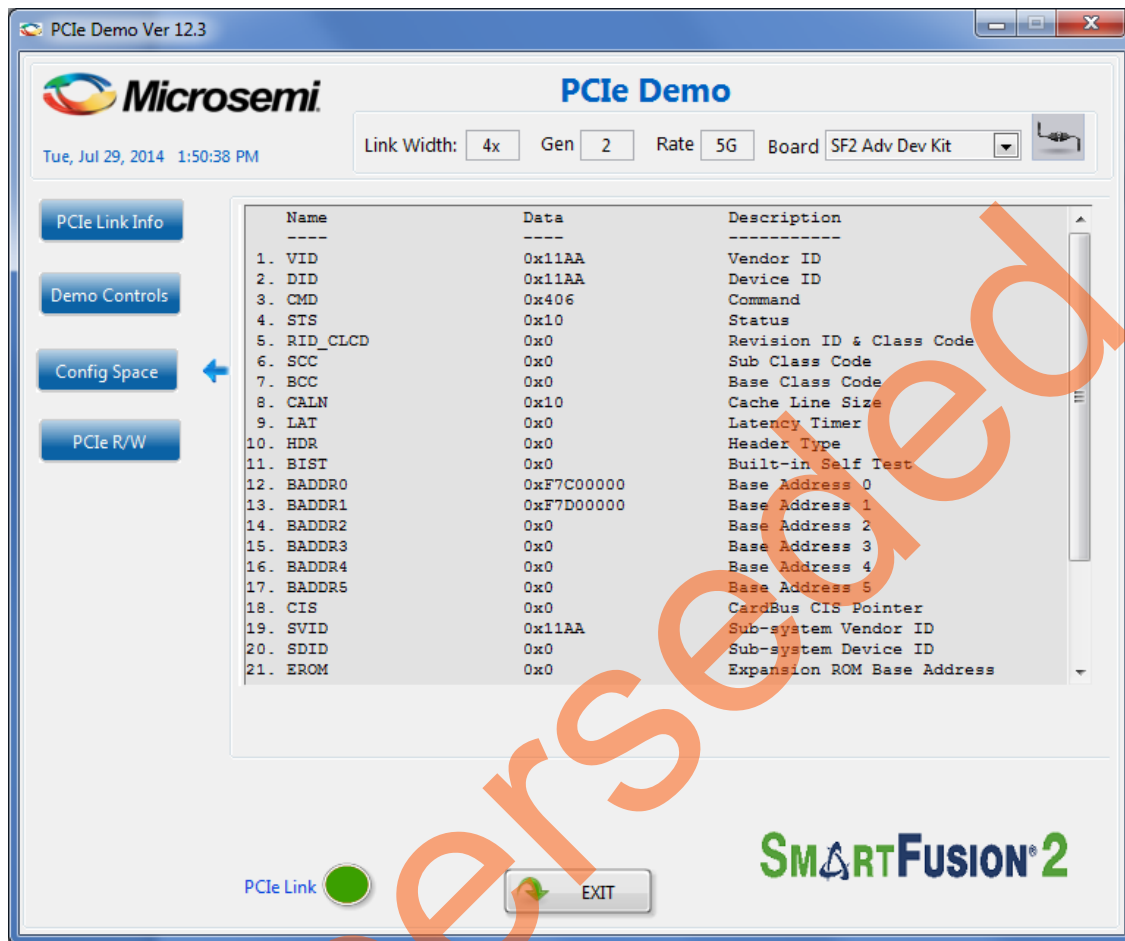


Figure 23 • Configuration Space

14. Click **PCIe R/W** to perform read and writes to LSRAM memory through **BAR1** space. Figure 24 on page 28 shows the **PCIe R/W** window.
15. Enter **Address** between 0x0000 to 0xFFFFC range.

16. Enter **Data**. The data field accepts a 32-bit hexadecimal value.

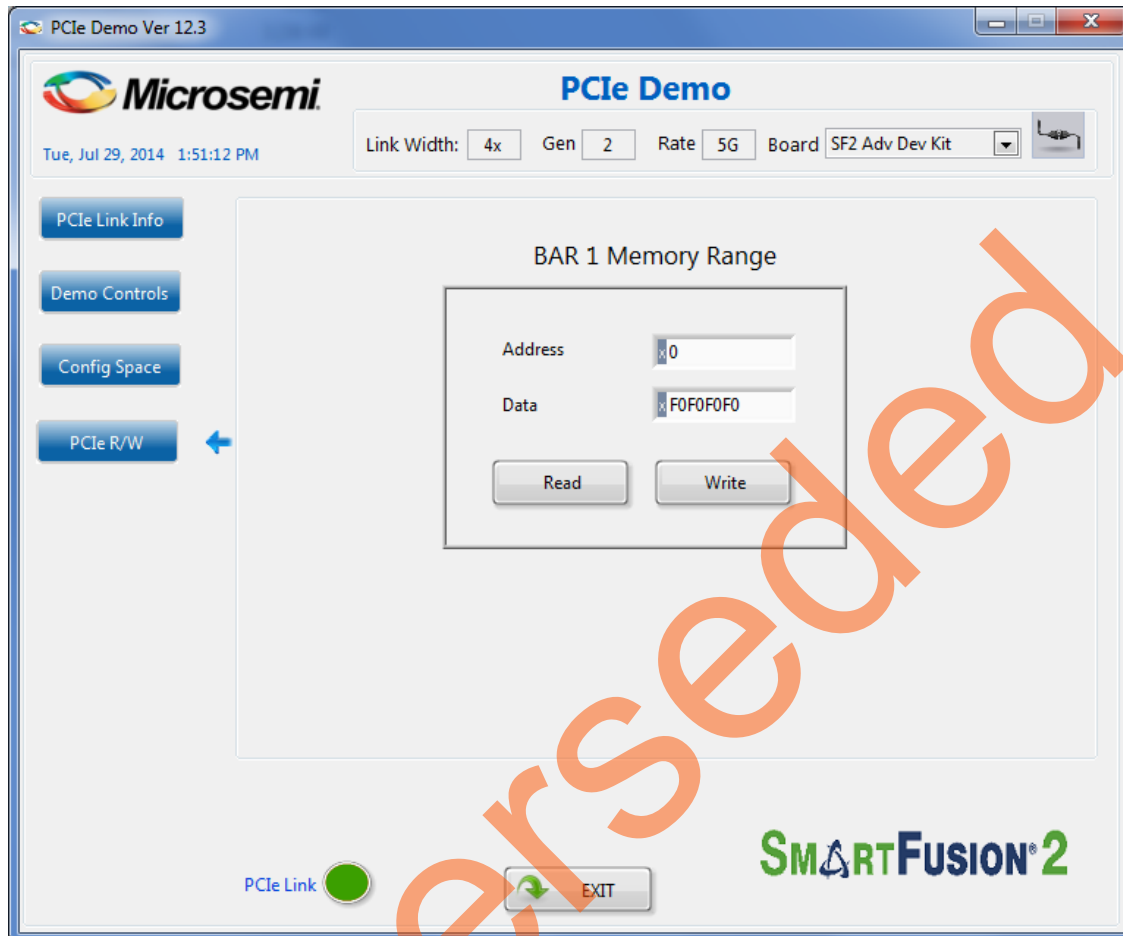


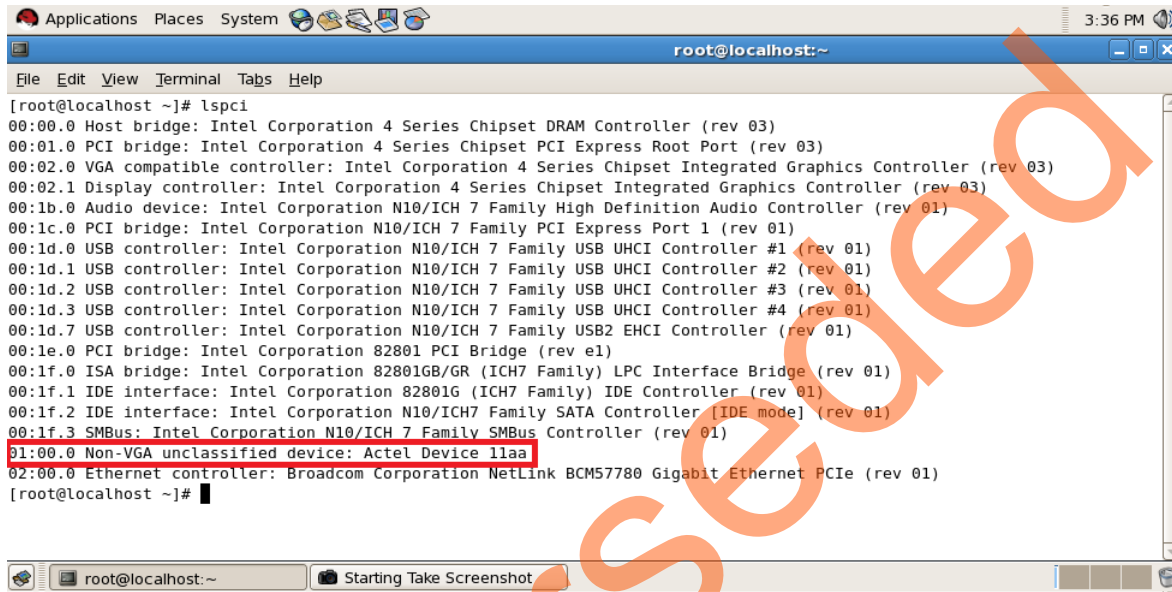
Figure 24 • Perform Read and Write to LSRAM Using PCIe

17. Click **Exit** to quit the demo.

Running the Demo Design on Linux

1. Switch **ON** the power supply switch **SW7** on the SmartFusion2 Advanced Development Kit Board.
2. Switch **ON** the Red Hat Linux Host PC.
3. Red Hat Linux Kernel detects the SmartFusion2 PCIe end point as Actel Device.
4. On Linux Command Prompt Use `lspci` command to display the PCIe info.

`lspci`



```
[root@localhost ~]# lspci
00:00.0 Host bridge: Intel Corporation 4 Series Chipset DRAM Controller (rev 03)
00:01.0 PCI bridge: Intel Corporation 4 Series Chipset PCI Express Root Port (rev 03)
00:02.0 VGA compatible controller: Intel Corporation 4 Series Chipset Integrated Graphics Controller (rev 03)
00:02.1 Display controller: Intel Corporation 4 Series Chipset Integrated Graphics Controller (rev 03)
00:1b.0 Audio device: Intel Corporation N10/ICH 7 Family High Definition Audio Controller (rev 01)
00:1c.0 PCI bridge: Intel Corporation N10/ICH 7 Family PCI Express Port 1 (rev 01)
00:1d.0 USB controller: Intel Corporation N10/ICH 7 Family USB UHCI Controller #1 (rev 01)
00:1d.1 USB controller: Intel Corporation N10/ICH 7 Family USB UHCI Controller #2 (rev 01)
00:1d.2 USB controller: Intel Corporation N10/ICH 7 Family USB UHCI Controller #3 (rev 01)
00:1d.3 USB controller: Intel Corporation N10/ICH 7 Family USB UHCI Controller #4 (rev 01)
00:1d.7 USB controller: Intel Corporation N10/ICH 7 Family USB2 EHCI Controller (rev 01)
00:1e.0 PCI bridge: Intel Corporation 82801 PCI Bridge (rev e1)
00:1f.0 ISA bridge: Intel Corporation 82801GB/GR (ICH7 Family) LPC Interface Bridge (rev 01)
00:1f.1 IDE interface: Intel Corporation 82801G (ICH7 Family) IDE Controller (rev 01)
00:1f.2 IDE interface: Intel Corporation N10/ICH7 Family SATA Controller [IDE mode] (rev 01)
00:1f.3 SMBus: Intel Corporation N10/ICH 7 Family SMBus Controller (rev 01)
01:00.0 Non-VGA unclassified device: Actel Device 11aa
02:00.0 Ethernet controller: Broadcom Corporation NetLink BCM57780 Gigabit Ethernet PCIe (rev 01)
[root@localhost ~]#
```

Figure 25 • PCIe Device Detection

Drivers Installation

Enter the following commands in the Linux command prompt to install the PCIe drivers:

1. Create the **sf2** directory under the **home/** directory using the following command:
`mkdir /home/sf2`
2. Copy the **M2S150_PcIe_Control_Plane_DF/** design files folder under **/home/sf2** directory, which contains the Linux PCIe device driver files and Linux PCIe application utility files.
3. Copy the Linux PCIe Device Driver file (**PCie_Driver.zip**) from **M2S150_PcIe_Control_Plane_DF/** design files folder.
`cp -rf /home/sf2/M2S150_PcIe_Control_Plane_DF/Linux_64bit/Drivers/PCie_Driver.rar /home/sf2`
`unzip PCie_Driver.rar`
/home/sf2 directory must contain the **PCie_Driver/ inc/** folders.
4. Execute `ls` command to display the contents of **/home/sf2** directory.
`ls`
5. Change to **inc/** directory by using the following command:
`cd /home/sf2/inc`

6. Edit the `board.h` file for SmartFusion2 Advanced Development Kit Board as shown in Figure 26.

```
#vi board.h
#define SF2_ADV_KIT
#undef IGL2
#undef SF2_DEV_KIT
#undef SF2_EVAL_KIT
```

7. Enter `[:wq]` command to save the selected file.
8. Enter the following command to change the directory:

```
#cd /home/sf2/PCie_Driver
```

9. Enter the `make` command on Linux Command Prompt to compile the Linux PCIe device driver code.

```
#make clean [To clean any *.o, *.ko files]
#make
```

The kernel module, `pci_chr_drv_ctrlpln.ko`, is created in the same directory.

10. Enter `insmod` command to insert the Linux PCIe device driver as a module.

```
#insmod pci_chr_drv_ctrlpln.ko
```

Note: Root privileges are required to execute this command.

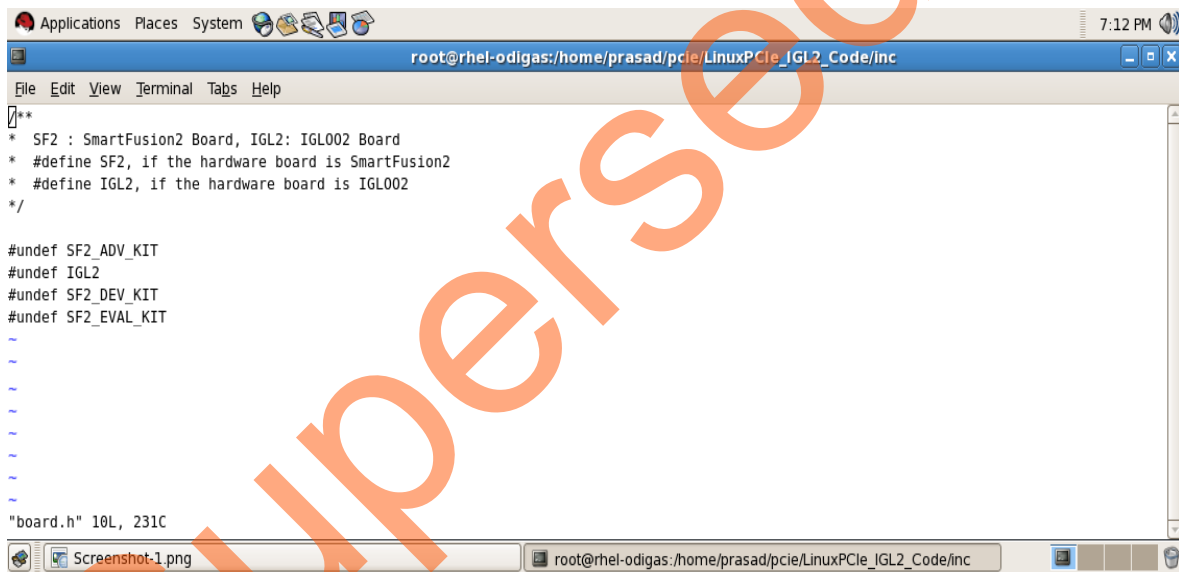
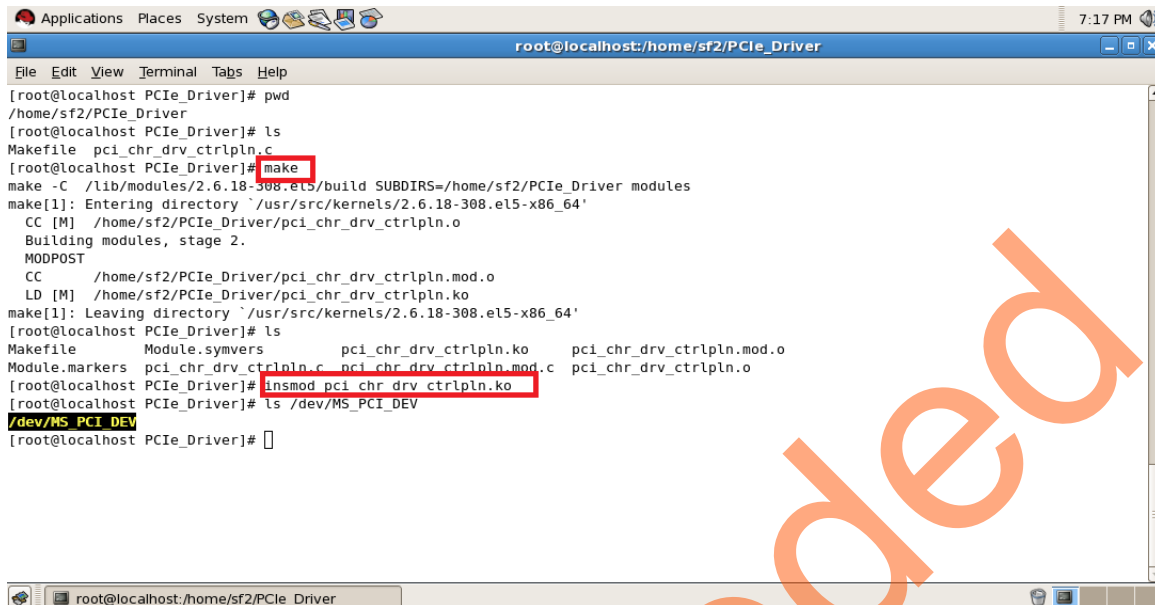


Figure 26 • Edit `board.h` File



```

root@localhost: /home/sf2/PCie_Driver
File Edit View Terminal Tabs Help
[root@localhost PCie_Driver]# pwd
/home/sf2/PCie_Driver
[root@localhost PCie_Driver]# ls
Makefile pci_chr_drv_ctrlpln.c
[root@localhost PCie_Driver]# make
make -C /lib/modules/2.6.18-308.el5/build SUBDIRS=/home/sf2/PCie_Driver modules
make[1]: Entering directory `/usr/src/kernels/2.6.18-308.el5-x86_64'
CC [M] /home/sf2/PCie_Driver/pci_chr_drv_ctrlpln.o
Building modules, stage 2.
MODPOST
CC /home/sf2/PCie_Driver/pci_chr_drv_ctrlpln.mod.o
LD [M] /home/sf2/PCie_Driver/pci_chr_drv_ctrlpln.ko
make[1]: Leaving directory `/usr/src/kernels/2.6.18-308.el5-x86_64'
[root@localhost PCie_Driver]# ls
Module.symvers pci_chr_drv_ctrlpln.ko pci_chr_drv_ctrlpln.mod.o
Module.markers pci_chr_drv_ctrlpln.c pci_chr_drv_ctrlpln.mod.c pci_chr_drv_ctrlpln.o
[root@localhost PCie_Driver]# insmod pci_chr_drv_ctrlpln.ko
[root@localhost PCie_Driver]# ls /dev/MS_PCI_DEV
/dev/MS_PCI_DEV
[root@localhost PCie_Driver]#

```

Figure 27 • PCIe Device Driver Installation

11. After successful Linux PCIe device driver installation, check `/dev/MS_PCI_DEV` got created by using the following Linux command:

```
#ls /dev/MS_PCI_DEV
```

Note: `/dev/MS_PCI_DEV` interface is used to access the SmartFusion2 PCIe end point from Linux user space.

Linux PCIe Application Compilation and PCIe Control Plane Utility Creation

1. Change to the `/home/sf2/` directory using the following command:


```
#cd /home/sf2
```
2. Copy the Linux PCIe application utility file (`PCie_App.zip`) from `M2S150_Control_Plane_DF/design files` folder.


```
# cp -rf /home/sf2/M2S150_Pcie_Control_Plane_DF/Linux_64bit/Util/PCie_App.rar /home/sf2
```

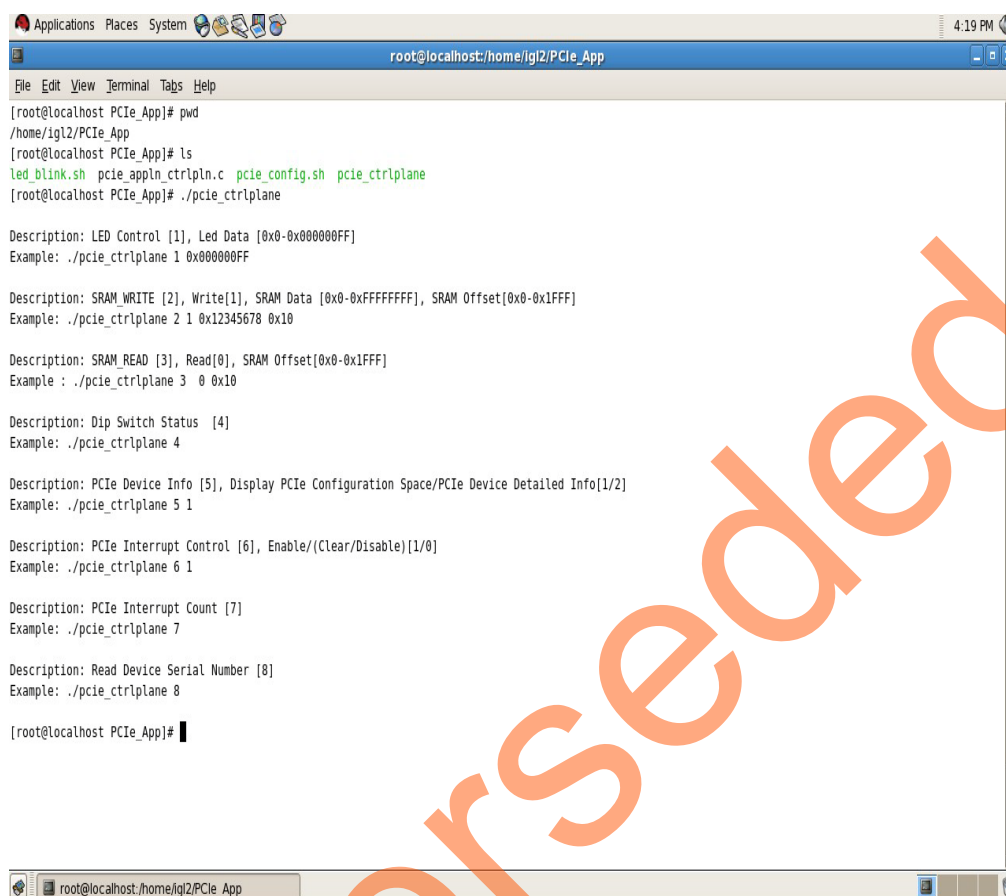
`/home/sf2` directory must contain `PCie_App/` folder along with `led_blink.sh` and `pcie_config.sh` scripts.
3. Execute `ls` command to display the contents of `/home/sf2` directory.


```
# ls
```
4. Compile the Linux user space application `pcie_appln_ctrlpln.c` in `/home/sf2/PCie_App` folder by using `gcc` command.


```
# cd /home/sf2/PCie_App
# gcc -o pcie_ctrlplane pcie_appln_ctrlpln.c
```

After successful compilation, Linux PCIe application utility `pcie_ctrlplane` creates in the same directory.
5. On Linux Command Prompt, run the `pcie_ctrlplane` utility as:


```
#./pcie_ctrlplane
```
6. Help menu is displayed as shown in [Figure 28 on page 32](#).



```
Applications Places System 4:19 PM
root@localhost:/home/igl2/PCie_App
File Edit View Terminal Tabs Help
[root@localhost PCie_App]# pwd
/home/igl2/PCie_App
[root@localhost PCie_App]# ls
led_blink.sh pcie_appln_ctrlpln.c pcie_config.sh pcie_ctrlplane
[root@localhost PCie_App]# ./pcie_ctrlplane

Description: LED Control [1], Led Data [0x0-0x000000FF]
Example: ./pcie_ctrlplane 1 0x000000FF

Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFFF], SRAM Offset[0x0-0x1FFF]
Example: ./pcie_ctrlplane 2 1 0x12345678 0x10

Description: SRAM_READ [3], Read[0], SRAM Offset[0x0-0x1FFF]
Example : ./pcie_ctrlplane 3 0 0x10

Description: Dip Switch Status [4]
Example: ./pcie_ctrlplane 4

Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2]
Example: ./pcie_ctrlplane 5 1

Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0]
Example: ./pcie_ctrlplane 6 1

Description: PCIe Interrupt Count [7]
Example: ./pcie_ctrlplane 7

Description: Read Device Serial Number [8]
Example: ./pcie_ctrlplane 8

[root@localhost PCie_App]#
```

Figure 28 • Linux PCIe Application Utility

Execution of Linux PCIe Control Plane Features

LED Control

LED1 to LED8 is controlled by writing data to SmartFusion2 LED Control Registers.

```
#./pcie_ctrlplane 1 0x000000FF [LED OFF]
#./pcie_ctrlplane 1 0x00000000 [LED ON]
```

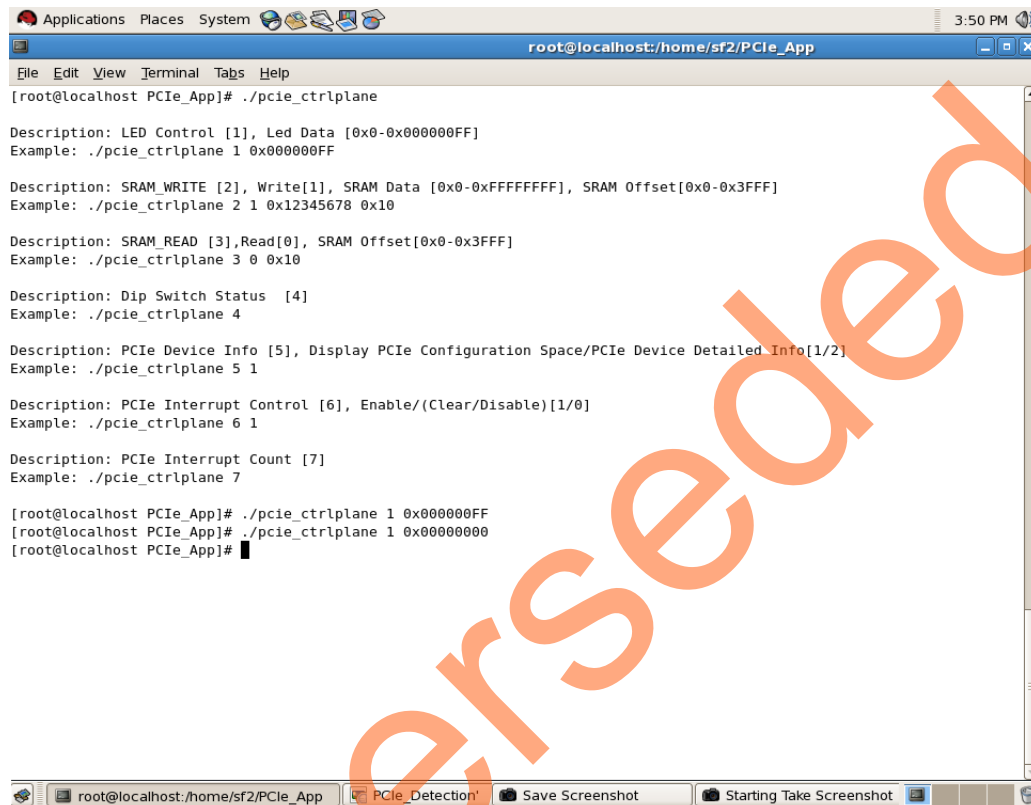


Figure 29 • Linux Command - LED Control

led_blink.sh contains the shell script code to perform the LED Walk ON, whereas Ctrl C exits the shell script and LED Walk turns OFF.

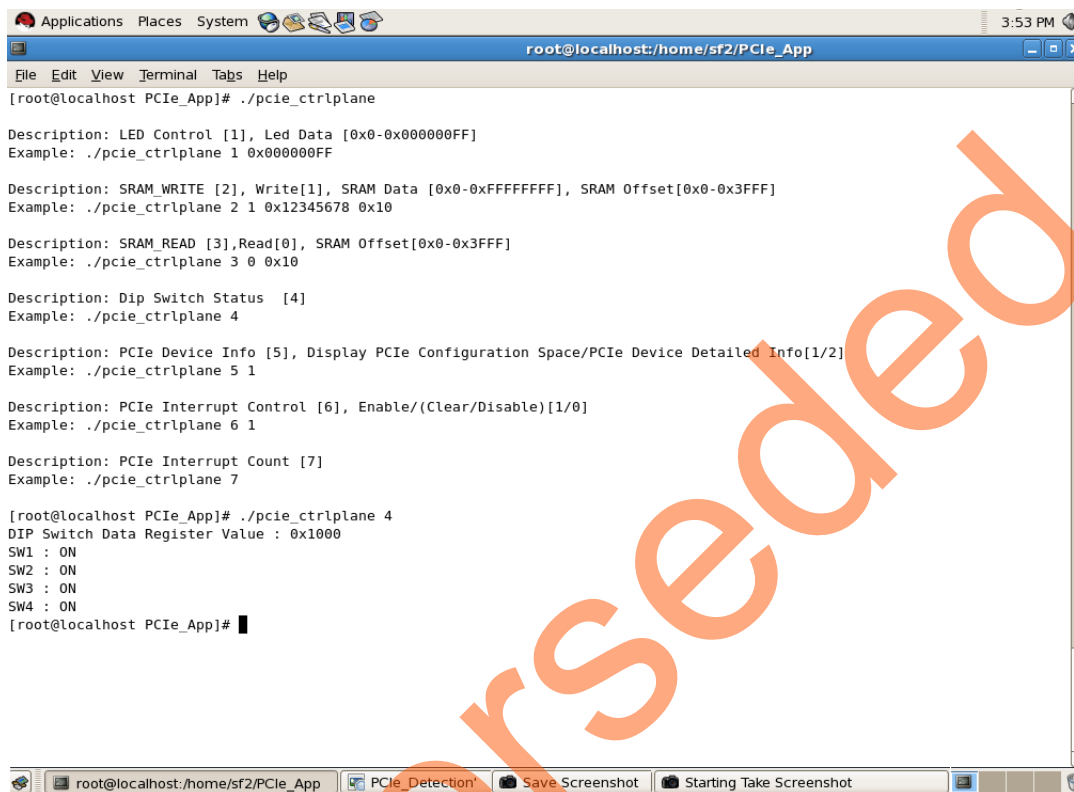
```
#sh led_blink.sh
```

Run the led_blink.sh shell script using the sh command.

DIP Switch Status

Dip Switch on SmartFusion2 Advanced Development Kit Board has four electric switches to hold the device configurations. Linux PCIe utility reads the corresponding switches (ON/OFF) state.

```
#./pcie_ctrlplane 4 [DIP Switch Status]
```



```
Applications Places System root@localhost:/home/sf2/PCie_App 3:53 PM
File Edit View Terminal Tabs Help
[root@localhost PCie_App]# ./pcie_ctrlplane

Description: LED Control [1], Led Data [0x0-0x000000FF]
Example: ./pcie_ctrlplane 1 0x000000FF

Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFFF], SRAM Offset[0x0-0x3FFF]
Example: ./pcie_ctrlplane 2 1 0x12345678 0x10

Description: SRAM_READ [3],Read[0], SRAM Offset[0x0-0x3FFF]
Example: ./pcie_ctrlplane 3 0 0x10

Description: Dip Switch Status [4]
Example: ./pcie_ctrlplane 4

Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2]
Example: ./pcie_ctrlplane 5 1

Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0]
Example: ./pcie_ctrlplane 6 1

Description: PCIe Interrupt Count [7]
Example: ./pcie_ctrlplane 7

[root@localhost PCie_App]# ./pcie_ctrlplane 4
DIP Switch Data Register Value : 0x1000
SW1 : ON
SW2 : ON
SW3 : ON
SW4 : ON
[root@localhost PCie_App]#
```

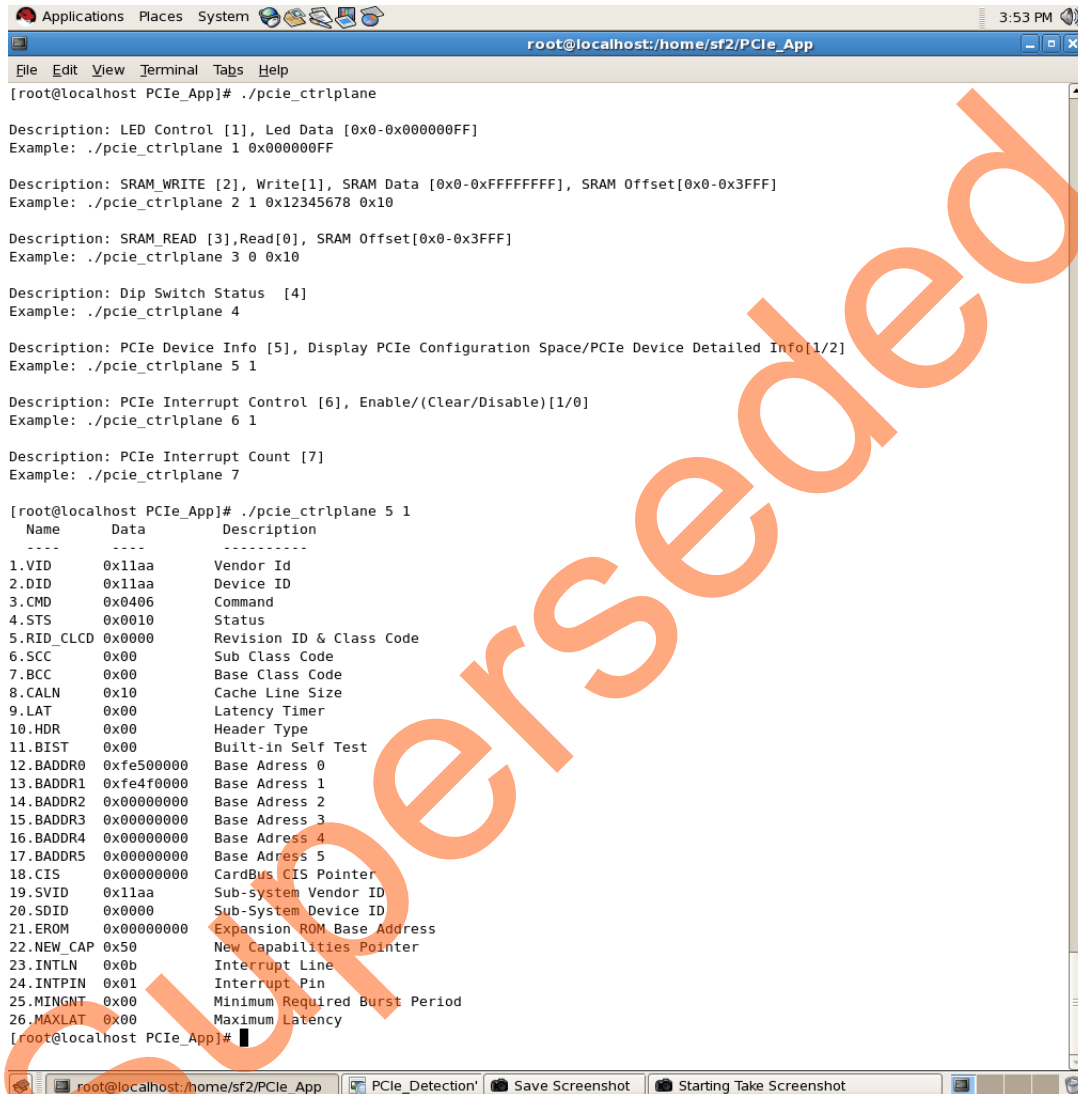
Figure 30 • Linux Command - DIP Switch

PCIe Configuration Space Display

PCIe configuration space contains the PCIe device data such as Vendor ID, Device ID, and Base Address 0.

Note: Root Privileges are required to execute this command.

```
#./pcie_ctrlplane 5 1 [Read PCIe Configuration Space]
```



```

root@localhost:~/home/sf2/PCIE_App
[root@localhost PCIE_App]# ./pcie_ctrlplane

Description: LED Control [1], Led Data [0x0-0x000000FF]
Example: ./pcie_ctrlplane 1 0x000000FF

Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFFF], SRAM Offset[0x0-0x3FFF]
Example: ./pcie_ctrlplane 2 1 0x12345678 0x10

Description: SRAM_READ [3],Read[0], SRAM Offset[0x0-0x3FFF]
Example: ./pcie_ctrlplane 3 0 0x10

Description: Dip Switch Status [4]
Example: ./pcie_ctrlplane 4

Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2]
Example: ./pcie_ctrlplane 5 1

Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0]
Example: ./pcie_ctrlplane 6 1

Description: PCIe Interrupt Count [7]
Example: ./pcie_ctrlplane 7

[root@localhost PCIE_App]# ./pcie_ctrlplane 5 1
  Name      Data      Description
  ----      -
1.VID      0x11aa    Vendor Id
2.DID      0x11aa    Device ID
3.CMD      0x0406    Command
4.STS      0x0010    Status
5.RID_CLCD 0x0000    Revision ID & Class Code
6.SCC      0x00      Sub Class Code
7.BCC      0x00      Base Class Code
8.CALN     0x10      Cache Line Size
9.LAT      0x00      Latency Timer
10.HDR     0x00      Header Type
11.BIST     0x00      Built-in Self Test
12.BADDR0  0xfe500000 Base Address 0
13.BADDR1  0xfe4f0000 Base Address 1
14.BADDR2  0x00000000 Base Address 2
15.BADDR3  0x00000000 Base Address 3
16.BADDR4  0x00000000 Base Address 4
17.BADDR5  0x00000000 Base Address 5
18.CIS     0x00000000 CardBus CIS Pointer
19.SVID     0x11aa    Sub-system Vendor ID
20.SDID     0x0000    Sub-System Device ID
21.EROM    0x00000000 Expansion ROM Base Address
22.NEW_CAP 0x50      New Capabilities Pointer
23.INTLN   0x0b      Interrupt Line
24.INTPIN   0x01      Interrupt Pin
25.MINGNT   0x00      Minimum Required Burst Period
26.MAXLAT   0x00      Maximum Latency
[root@localhost PCIE_App]#

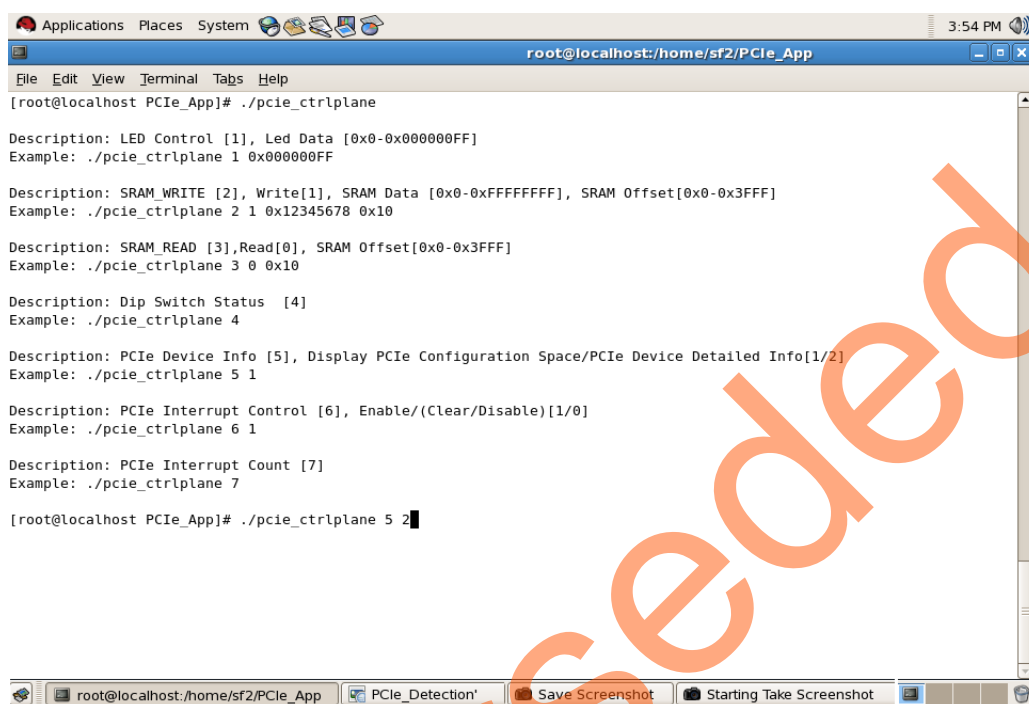
```

Figure 31 • Linux Command - PCIe Configuration Space Display

PCIe Link Speed and Width

Note: Root Privileges are required to execute this command.

```
#./pcie_ctrlplane 5 2 [Read PCIe Link Speed and Link Width]
```



```
root@localhost:/home/sf2/PCie_App
[root@localhost PCie_App]# ./pcie_ctrlplane

Description: LED Control [1], Led Data [0x0-0x000000FF]
Example: ./pcie_ctrlplane 1 0x000000FF

Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFFF], SRAM Offset[0x0-0x3FFF]
Example: ./pcie_ctrlplane 2 1 0x12345678 0x10

Description: SRAM_READ [3], Read[0], SRAM Offset[0x0-0x3FFF]
Example: ./pcie_ctrlplane 3 0 0x10

Description: Dip Switch Status [4]
Example: ./pcie_ctrlplane 4

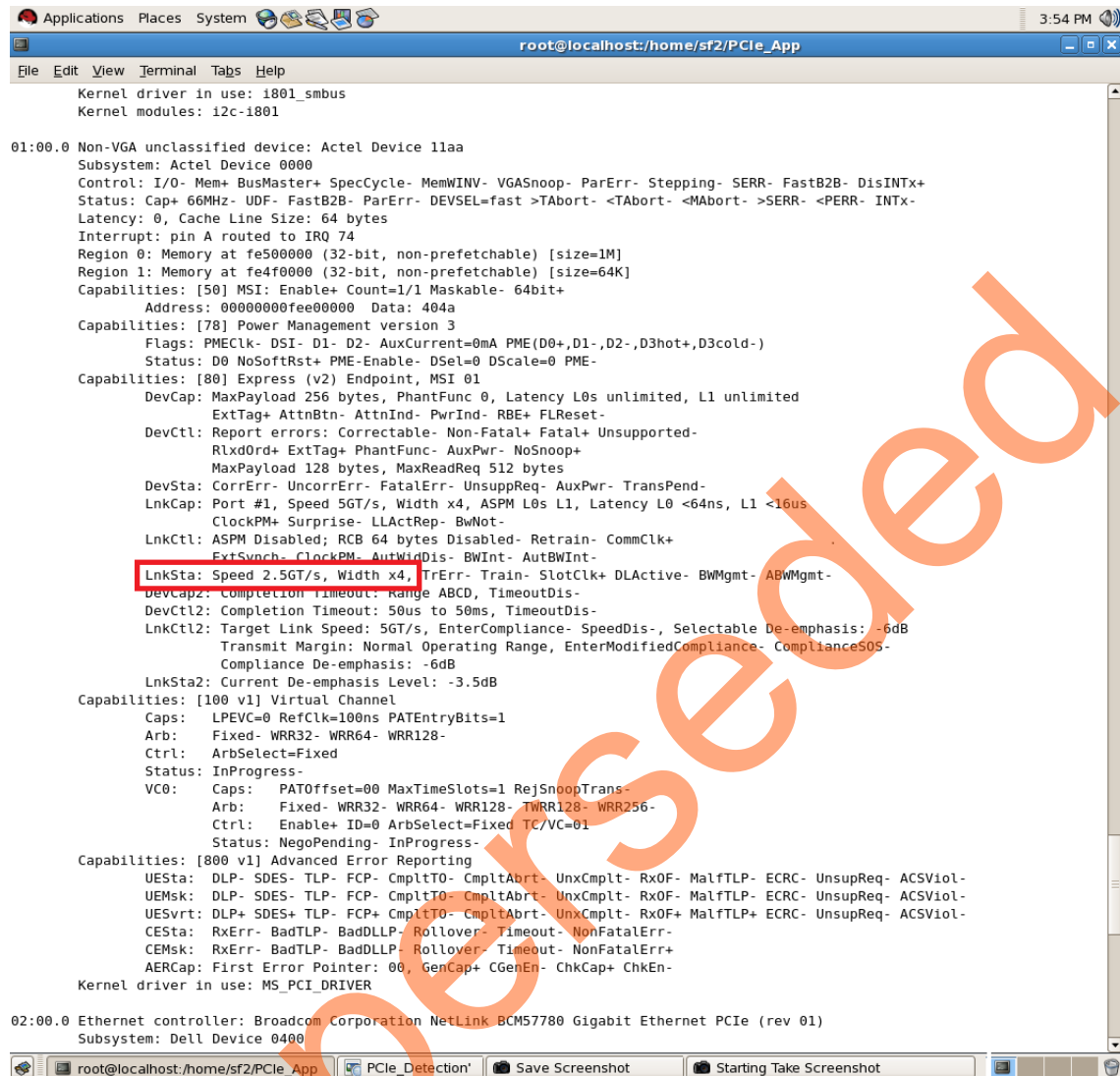
Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2]
Example: ./pcie_ctrlplane 5 1

Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0]
Example: ./pcie_ctrlplane 6 1

Description: PCIe Interrupt Count [7]
Example: ./pcie_ctrlplane 7

[root@localhost PCie_App]# ./pcie_ctrlplane 5 2
```

Figure 32 • Linux Command - PCIe Link Speed and Width



```

Applications Places System 3:54 PM
root@localhost:/home/sf2/PCle_App
File Edit View Terminal Tabs Help
Kernel driver in use: i801_smbus
Kernel modules: i2c-i801

01:00.0 Non-VGA unclassified device: Actel Device 11aa
Subsystem: Actel Device 0000
Control: I/O- Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx+
Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
Latency: 0, Cache Line Size: 64 bytes
Interrupt: pin A routed to IRQ 74
Region 0: Memory at fe500000 (32-bit, non-prefetchable) [size=1M]
Region 1: Memory at fe4f0000 (32-bit, non-prefetchable) [size=64K]
Capabilities: [50] MSI: Enable+ Count=1/1 Maskable- 64bit+
Address: 00000000fee00000 Data: 404a
Capabilities: [78] Power Management version 3
Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0+,D1-,D2-,D3hot+,D3cold-)
Status: D0 NoSoftRst+ PME-Enable- DSel=0 DScale=0 PME-
Capabilities: [80] Express (v2) Endpoint, MSI 01
DevCap: MaxPayload 256 bytes, PhantFunc 0, Latency L0s unlimited, L1 unlimited
ExtTag+ AttnBtn- AttnInd- PwrInd- RBE+ FLReset-
DevCtl: Report errors: Correctable- Non-Fatal+ Fatal+ Unsupported-
Rlxd0rd+ ExtTag+ PhantFunc- AuxPwr- NoSnoop+
MaxPayload 128 bytes, MaxReadReq 512 bytes
DevSta: CorrErr- UncorrErr- FatalErr- UnsuppReq- AuxPwr- TransPend-
LnkCap: Port #1, Speed 5GT/s, Width x4, ASPM L0s L1, Latency L0 <64ns, L1 <16us
ClockPM+ Surprise- LLActRep- BwNot-
LnkCtl: ASPM Disabled; RCB 64 bytes Disabled- Retrain- CommClk+
ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
LnkSta: Speed 2.5GT/s, Width x4, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
DevCap2: Completion Timeout: Range ABCD, TimeoutDis-
DevCtl2: Completion Timeout: 50us to 50ms, TimeoutDis-
LnkCtl2: Target Link Speed: 5GT/s, EnterCompliance- SpeedDis-, Selectable De-emphasis: -6dB
Transmit Margin: Normal Operating Range, EnterModifiedCompliance- ComplianceSOS-
Compliance De-emphasis: -6dB
LnkSta2: Current De-emphasis Level: -3.5dB
Capabilities: [100 v1] Virtual Channel
Caps: LPEVC=0 RefClk=100ns PATEntryBits=1
Arb: Fixed- WRR32- WRR64- WRR128-
Ctrl: ArbSelect=Fixed
Status: InProgress-
VC0:
Caps: PATOffset=00 MaxTimeSlots=1 RejSnoopTrans-
Arb: Fixed- WRR32- WRR64- WRR128- TWRR128- WRR256-
Ctrl: Enable+ ID=0 ArbSelect=Fixed TC/VC=01
Status: NegoPending- InProgress-
Capabilities: [800 v1] Advanced Error Reporting
UESta: DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq- ACSViol-
UEmsk: DLP- SDES- TLP- FCP- CmpltTO- CmpltAbrt- UnxCmplt- RxOF- MalfTLP- ECRC- UnsupReq- ACSViol-
UESvrt: DLP+ SDES+ TLP- FCP+ CmpltTO- CmpltAbrt- UnxCmplt- RxOF+ MalfTLP+ ECRC- UnsupReq- ACSViol-
CESta: RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr-
CEmsk: RxErr- BadTLP- BadDLLP- Rollover- Timeout- NonFatalErr+
AERCap: First Error Pointer: 00, GenCap+ CGenEn- ChkCap+ ChkEn-
Kernel driver in use: MS_PCI_DRIVER

02:00.0 Ethernet controller: Broadcom Corporation NetLink BCM57780 Gigabit Ethernet PCIe (rev 01)
Subsystem: Dell Device 0400
root@localhost:/home/sf2/PCle_App [PCle_Detection] Save Screenshot Starting Take Screenshot

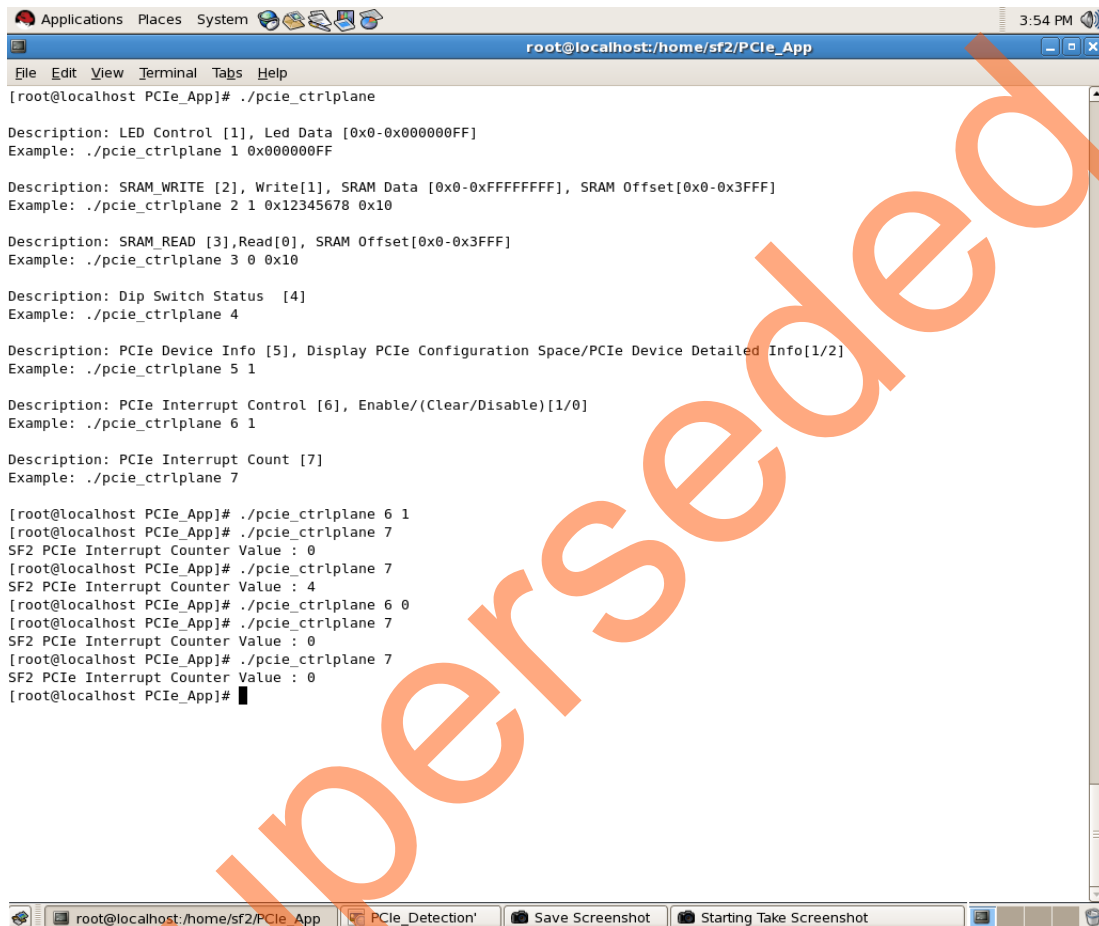
```

Figure 33 • Linux Command - PCIe Link Speed and Width

PCIe Interrupt Control (Enable/Disable) and Interrupt Counter

SmartFusion2 Advanced Development Kit Board enables or disables the MSI interrupts by writing data to its PCIe configuration space. Interrupt Counter holds the number of MSI interrupts got triggered by pressing the **SW1** push button.

```
#. /pcie_ctrlplane 6 0 [Disable Interrupts]
#. /pcie_ctrlplane 6 1 [Enable Interrupts]
#. /pcie_ctrlplane 7 [Interrupt Counter Value]
```



```
root@localhost: /home/sf2/PCie_App
File Edit View Terminal Tabs Help
[root@localhost PCie_App]# ./pcie_ctrlplane

Description: LED Control [1], Led Data [0x0-0x000000FF]
Example: ./pcie_ctrlplane 1 0x000000FF

Description: SRAM_WRITE [2], Write[1], SRAM Data [0x0-0xFFFFFFFF], SRAM Offset[0x0-0x3FFF]
Example: ./pcie_ctrlplane 2 1 0x12345678 0x10

Description: SRAM_READ [3], Read[0], SRAM Offset[0x0-0x3FFF]
Example: ./pcie_ctrlplane 3 0 0x10

Description: Dip Switch Status [4]
Example: ./pcie_ctrlplane 4

Description: PCIe Device Info [5], Display PCIe Configuration Space/PCIe Device Detailed Info[1/2]
Example: ./pcie_ctrlplane 5 1

Description: PCIe Interrupt Control [6], Enable/(Clear/Disable)[1/0]
Example: ./pcie_ctrlplane 6 1

Description: PCIe Interrupt Count [7]
Example: ./pcie_ctrlplane 7

[root@localhost PCie_App]# ./pcie_ctrlplane 6 1
[root@localhost PCie_App]# ./pcie_ctrlplane 7
SF2 PCIe Interrupt Counter Value : 0
[root@localhost PCie_App]# ./pcie_ctrlplane 7
SF2 PCIe Interrupt Counter Value : 4
[root@localhost PCie_App]# ./pcie_ctrlplane 6 0
[root@localhost PCie_App]# ./pcie_ctrlplane 7
SF2 PCIe Interrupt Counter Value : 0
[root@localhost PCie_App]# ./pcie_ctrlplane 7
SF2 PCIe Interrupt Counter Value : 0
[root@localhost PCie_App]#
```

Figure 34 • Linux Command - PCIe Interrupt Control

Conclusion

This demo describes how to access the PCIe EP and displays the device serial number feature of SmartFusion2 by implementing a low bandwidth control plane design with BFM simulation. It provides a GUI for easy control of PCIe EP device through Jungo drivers for windows platform. It also provides a Linux PCIe application for easy control of PCIe EP device through Linux PCIe Device Driver.

Appendix 1: SmartFusion2 Advanced Development Kit Board

Figure 35 shows the SmartFusion2 Advanced Development Kit Board.

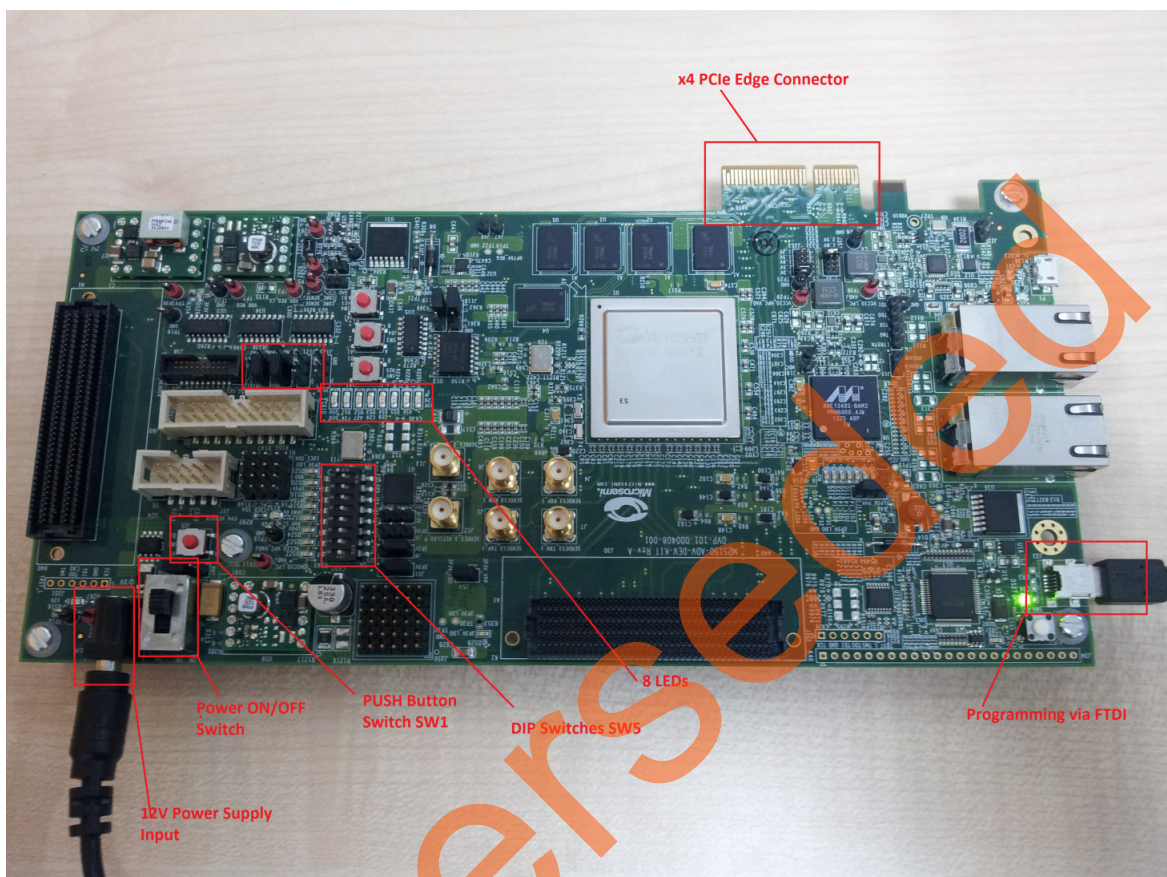


Figure 35 • SmartFusion2 Advanced Development Kit Board

A – List of Changes

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

Date	Changes	Page
Revision 2 (January 2015)	Updated the document for Libero v11.5 software release (SAR 63979).	NA
Revision 1 (August 2014)	Initial release	NA

Superseded

B – Product Support

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

Customer Service

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

From North America, call 800.262.1060

From the rest of the world, call 650.318.4460

Fax, from anywhere in the world, 408.643.6913

Customer Technical Support Center

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

Technical Support

For Microsemi SoC Products Support, visit

<http://www.microsemi.com/products/fpga-soc/designsupport/fpga-soc-support>

Website

You can browse a variety of technical and non-technical information on the SoC home page, at

www.microsemi.com/soc.

Contacting the Customer Technical Support Center

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

Email

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

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

My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to [My Cases](#).

Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office. [Sales office listings](#) can be found at www.microsemi.com/soc/company/contact/default.aspx.

ITAR Technical Support

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

Superseded



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

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

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; security technologies and scalable anti-tamper products; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 3,400 employees globally. Learn more at www.microsemi.com.

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