DG0566
Demo Guide
SmartFusion2 SoC FPGA PCIe Control Plane Demo For Advanced Development Kit - Libero SoC v11.8 SP1

Microsemi
Power Matters.™

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
www.microsemi.com

About Microsemi

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for aerospace & defense, communications, data center and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; enterprise storage and communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif., and has approximately 4,800 employees globally. Learn more at www.microsemi.com.

50200566. 5.0 9/17

Contents

Figures

Tables

1 Revision History

The revision history describes the changes that were implemented in the document. The changes are listed by revision, starting with the most current publication.

1.1 Revision 5.0

Updated the document for Libero v11.8 SP1 software release.

1.2 Revision 4.0

Updated the document for Libero v11.7 software release.

1.3 Revision 3.0

Updated the document for Libero v11.6 software release.

1.4 Revision 2.0

Updated the document for Libero v11.5 software release.

1.5 Revision 1.0

Revision 1.0 was the first publication of this document.

2 SmartFusion2 SoC FPGA PCIe Control Plane Demo For Advanced Development Kit

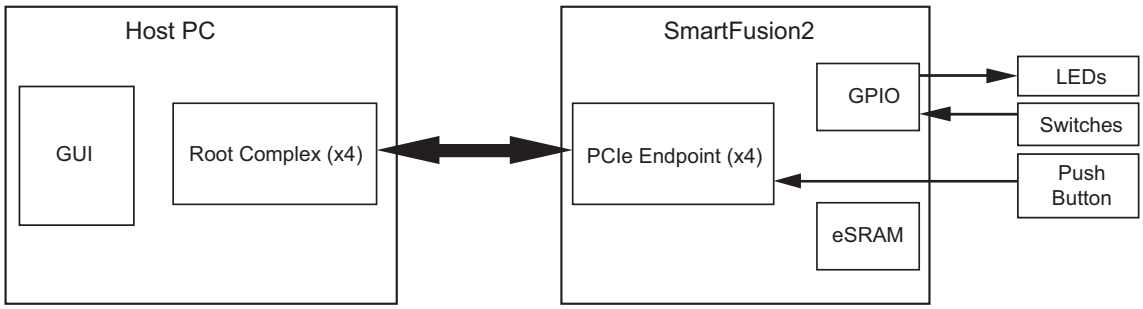
The SmartFusion2 SoC FPGA devices integrate a fourth generation flash-based FPGA fabric and an ARM Cortex-M3 processor, along with high performance communication interfaces on a single chip. The SmartFusion2 high speed serial interface (SERDESIF) provides a fully hardened PCIe endpoint (EP) implementation and is compliant with PCIe Base Specification Revision 2.0, 1.1, and 1.0. For more details, refer to the *UG0447: 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).

The following figure 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**



2.1 Design Requirements

The following table lists the design requirements details.

Table 1 • **Design Requirements**

Design Requirements	Version
Hardware	
SmartFusion2 Advanced Development Kit: – 12 V adapter – FlashPro5 – USB A to Mini-B cable	Rev E 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)

Table 1 • **Design Requirements *(continued)***

Software	
Libero SoC Design Suite	v11.8 SP1
FlashPro Programming Software	v11.8 SP1
Host PC Drivers (provided along with the design files)	–
GUI executable (provided along with the design files)	–
SoftConsole	v4.0

2.2 Demo Design

The design files for this demo can be downloaded from the Microsemi website: *http://soc.microsemi.com/download/rsc/?f=m2s_dg0566_libero11p8_sp1_df*

The following figure 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**

```
<Download folder>
└── M2S150_PCIe_Control_Plane_DF
    ├── GUI
    ├── Libero_Project
    ├── Linux_64bit
    ├── ProgrammingFile
    ├── Source Files
    ├── Windows_64bit
    └── Readme.txt
```

2.2.1 Demo Design Features

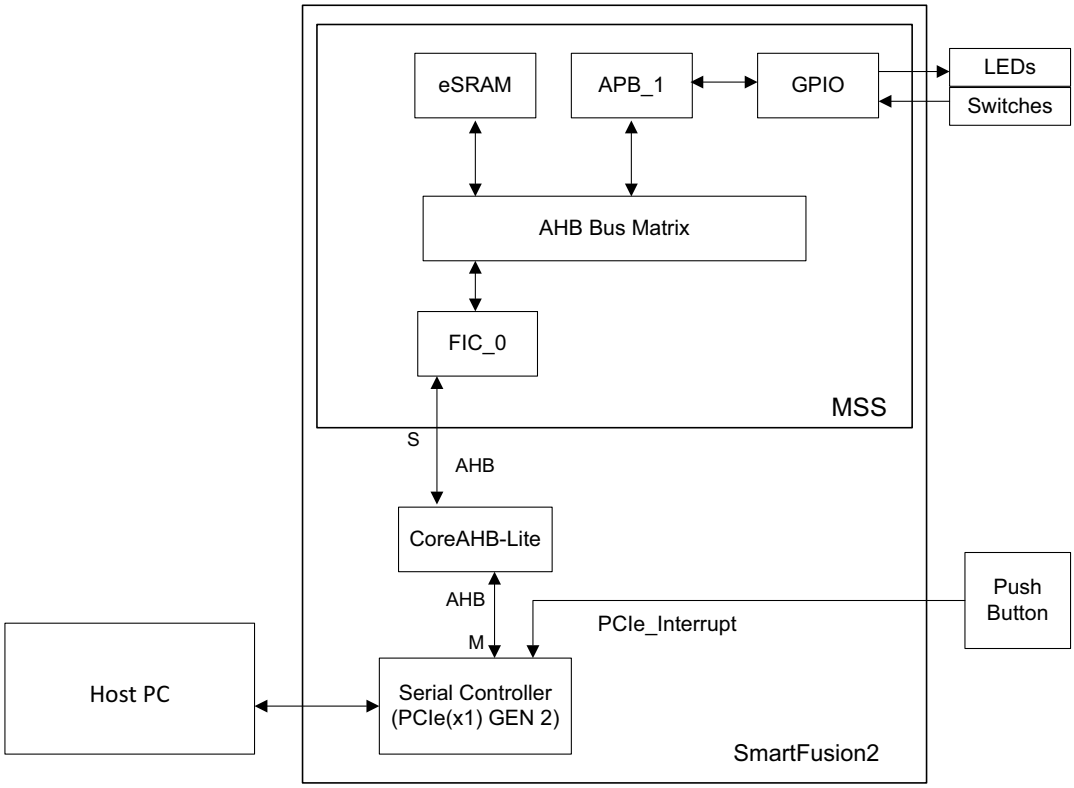
The demo design performs the following tasks:

- 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 eSRAM
- Accepts and displays interrupts from the push button on the SmartFusion2 Advanced Development Kit board
- Displays the SmartFusion2 PCIe Configuration space

2.2.2 Demo Design Description

The demo design helps to access the SmartFusion2 PCIe EP from the Host PC. The following figure 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. Preceding figure 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 advanced eXtensible interface (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 70 MHz.

2.2.2.1 Simulating the Design

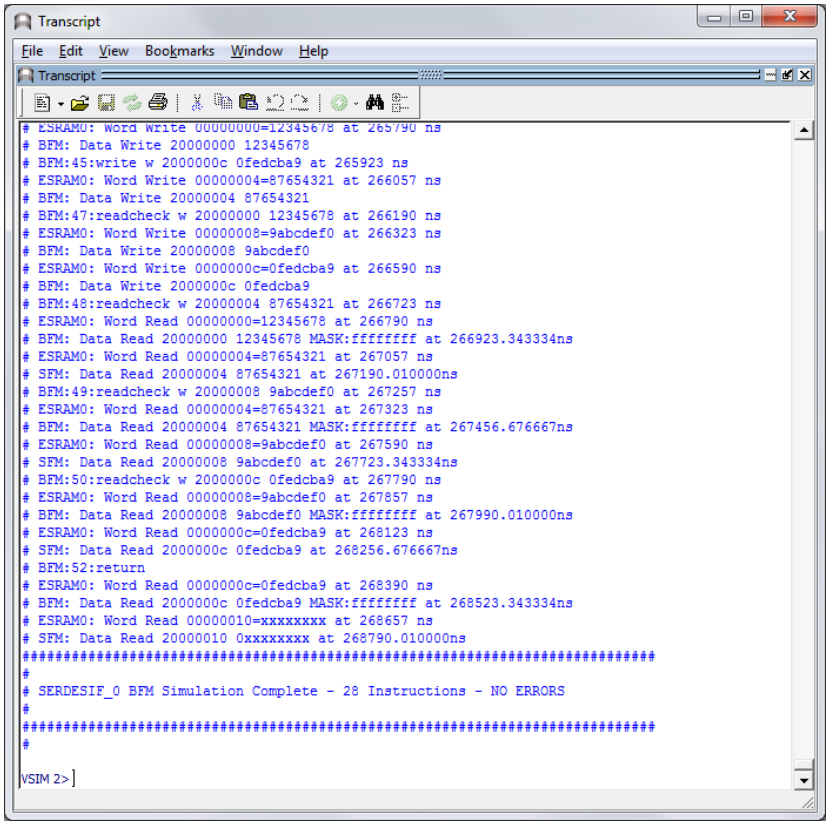
The design supports the BFM_PCIe simulation level to communicate with the High Speed Serial Interface block through the master AXI bus interface. Though, the serial communication does not actually go through the High Speed Serial Interface block, this scenario allows validating the fabric interface connections. 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 eSRAM
- Read-check from eSRAM

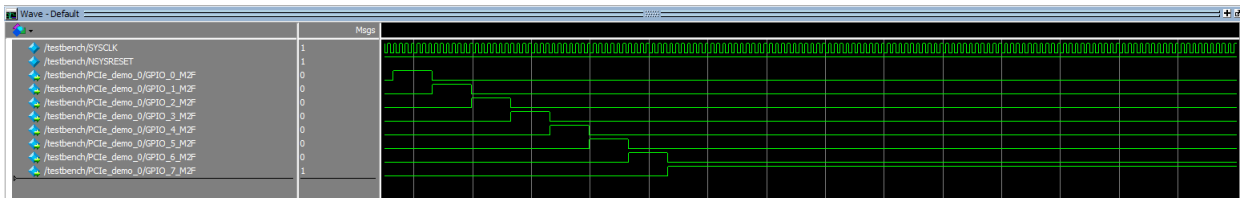
To run the simulation, double-click **Simulate** under **Verify Pre-Synthesized Design** in the **Design Flow** window of the Libero project. ModelSim runs the design for about 270 µs. The ModelSim **Transcript** window displays the BFM commands and the BFM simulation completed with no errors, as shown in the following figure.

Figure 4 • **SERDES BFM Simulation**



The following figure shows the **Wave** window with GPIO_OUT signals.

Figure 5 • **Simulation Result with GPIO_OUT Signals**

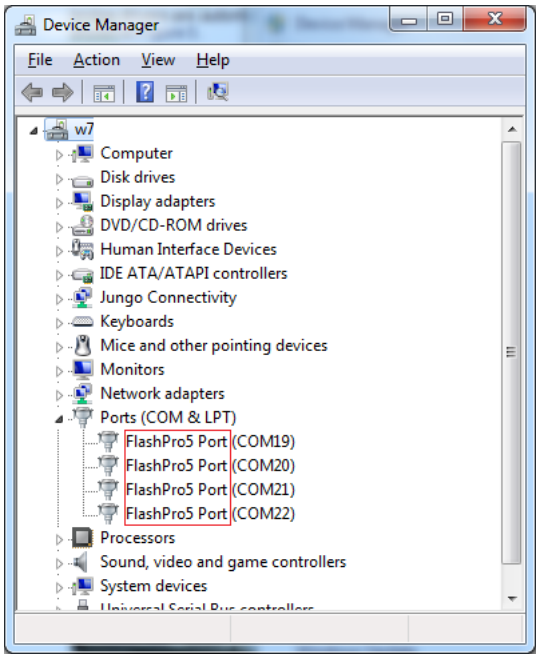


2.3 Setting Up the Demo Design

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 the following figure.

Figure 6 • **Device Manager**



2. Connect the jumpers on the SmartFusion2 Advanced Development Kit board, as listed in the following table.
   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 Development 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.

2.3.1 Board Setup

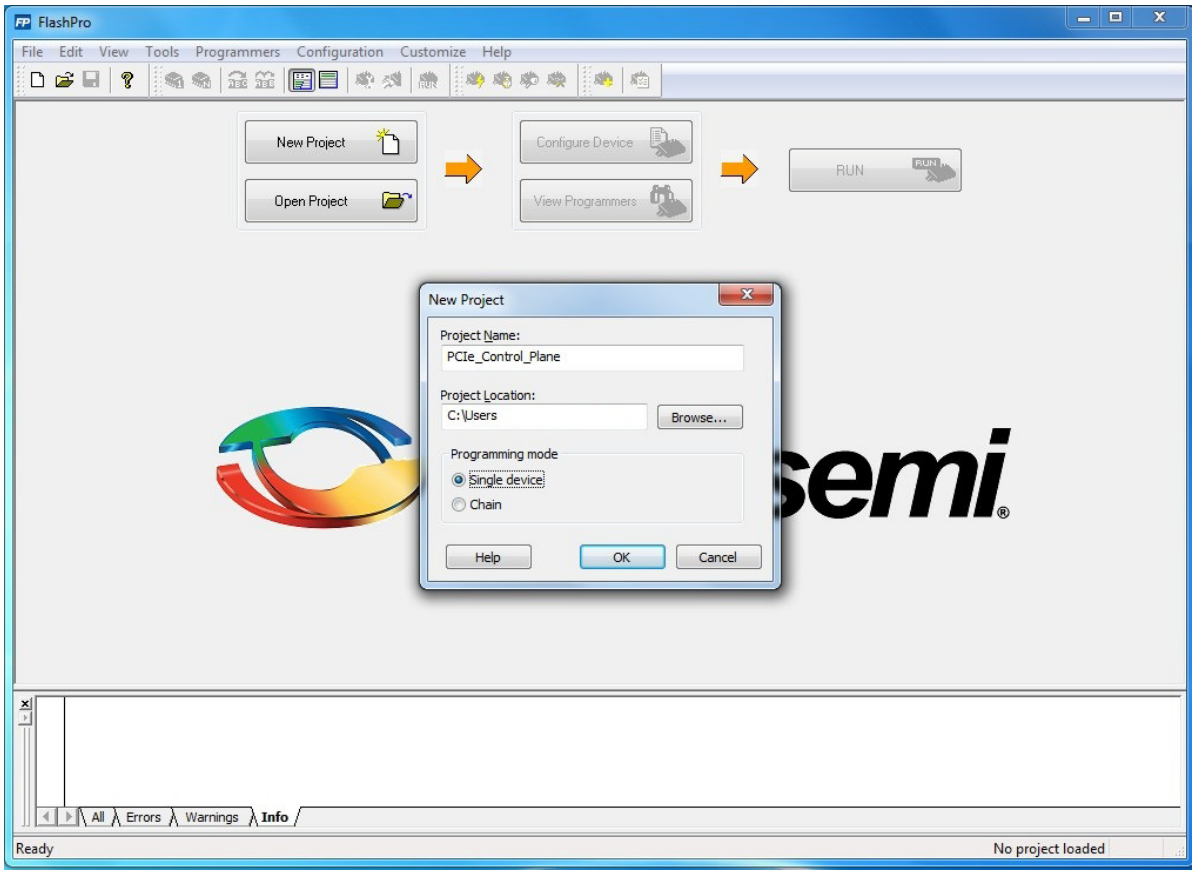
Snapshots of the SmartFusion2 Advanced Development Kit Board with the complete set up are given in the Appendix: SmartFusion2 Advanced Development Kit Board, page 28.

2.3.2 Programming the Board

The following steps describe how to program the board.

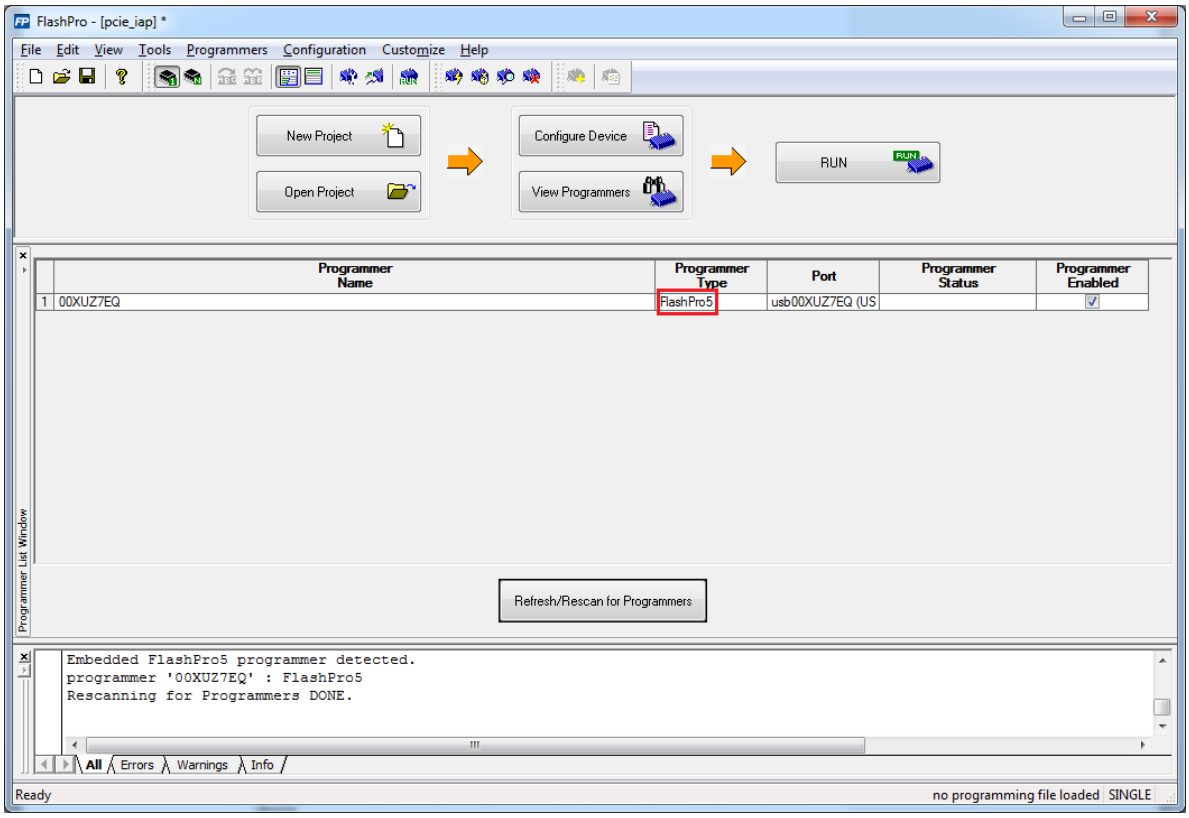
1. Download the demo design from:
   http://soc.microsemi.com/download/rsc/?f=m2s_dg0566_liberov11p8_sp1_df
2. Switch ON the power supply switch, **SW7**.
3. Launch the **FlashPro** software.
4. Click **New Project**.
5. In the **New Project** window, enter the **Project Name** as PCIe_Control_Plane.

Figure 7 • **FlashPro New Project**



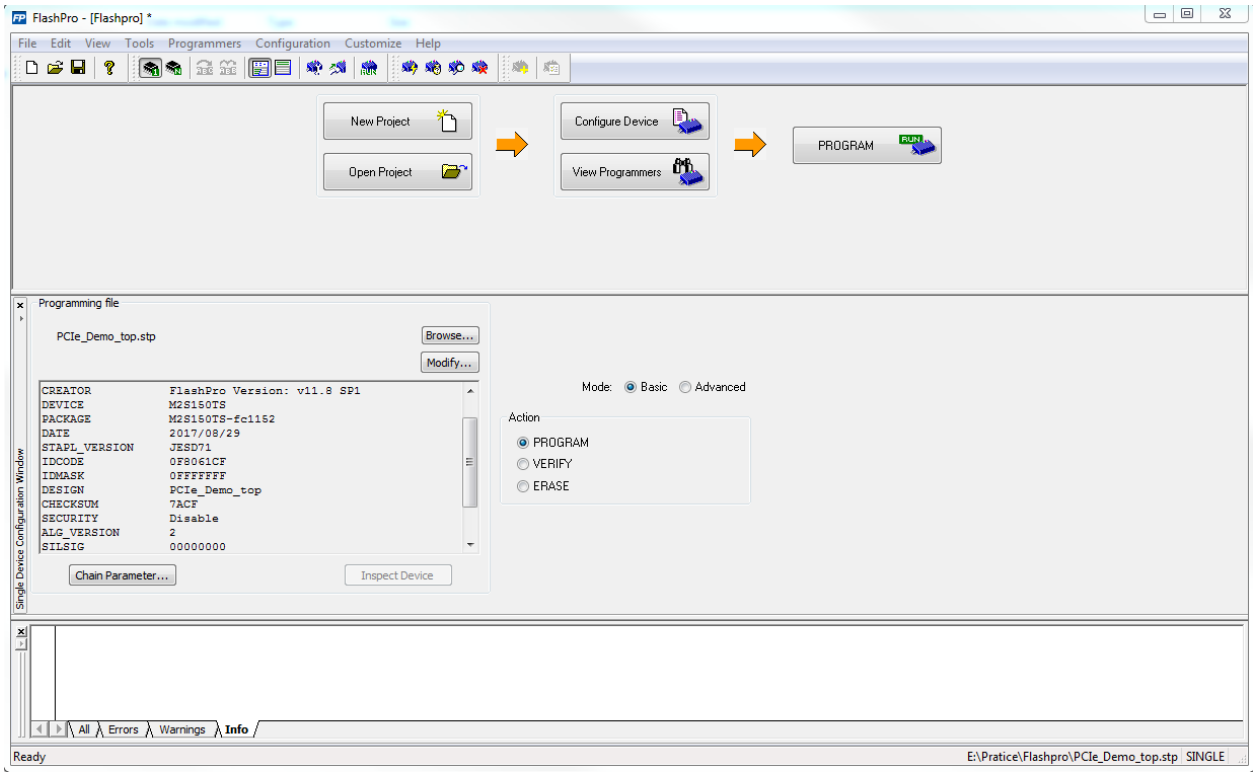
6. Click **Browse** and navigate to the location where you want to save the project.
7. Click **Single device** as the **Programming mode**.
8. Click **OK** to save the project.

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_PCIe_Control_Plane_DF\programmingFile*.
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**.

2.3.3 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.

The following figure 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**

2.4 Running the Demo Design

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

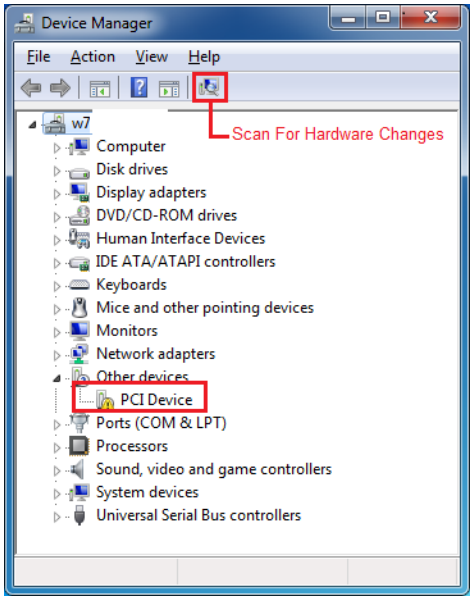
- To run the demo on Windows OS GUI, Microsemi PCIe are provided. Refer to Running the Demo Design on Windows, page 11.
- To run the demo on Linux OS, native RedHat Linux drivers and command line scripts are provided. Refer to Running the Demo Design on Linux, page 18.

2.4.1 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 the following figure. If the PCIe device is not detected, power cycle the SmartFusion2 Advanced Development Kit board.
3. Right-click **PCIe Device > Scan for hardware changes** in Device Manager.

Figure 11 • **Device Manager**



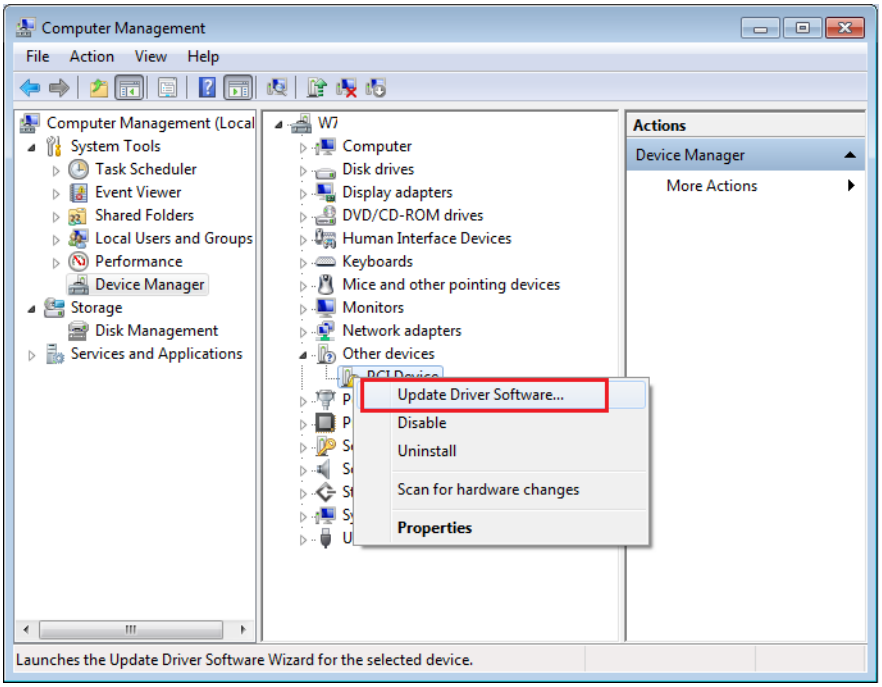
Note: If the device is still not detected, check whether or not the BIOS version in Host PC is the latest, and if PCIe is enabled in the Host PC BIOS.

2.4.1.1 Drivers Installation

Perform the following steps to install the PCIe drivers on the host PC:

1. Right-click **PCI Device** in Device Manager and select **Update Driver Software...** as shown in the following figure. To install the drivers, administrative rights are required.

Figure 12 • **Update Driver Software**



2. In the **Update Driver Software - PCIe Device** window, select the **Browse my computer for driver software** option as shown in the following figure.

Figure 13 • **Browse for Driver Software**

3. Browse the drivers folder: M2S150_PCIe_Control_Plane_DF\Windows_64bit\Drivers\PCIe_Demo and click **Next** as shown in the following figure.

Figure 14 • **Browse for Driver Software Continued**

4. The **Windows Security** dialog box is displayed. Click **Install** as shown in the following figure. After successful driver installation, a message appears. See Figure 16, page 13.

Figure 15 • **Windows Security**

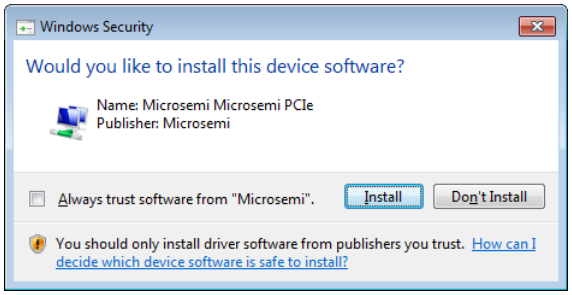
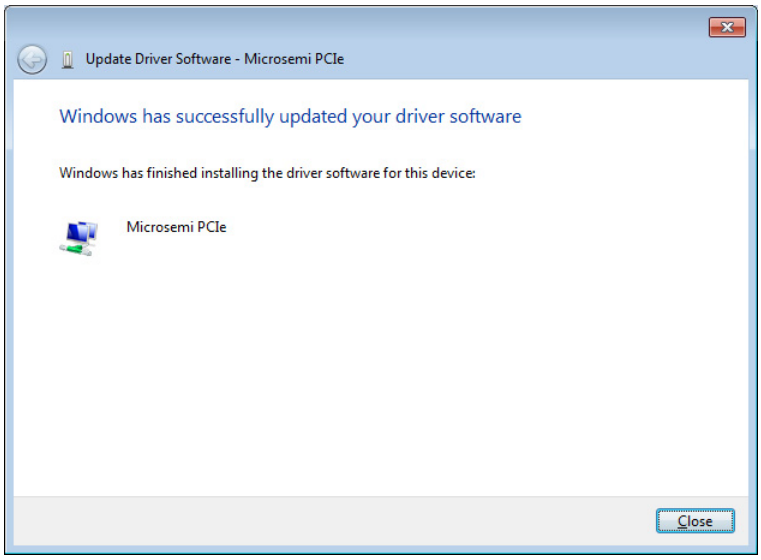


Figure 16 • **Successful Driver Installation**



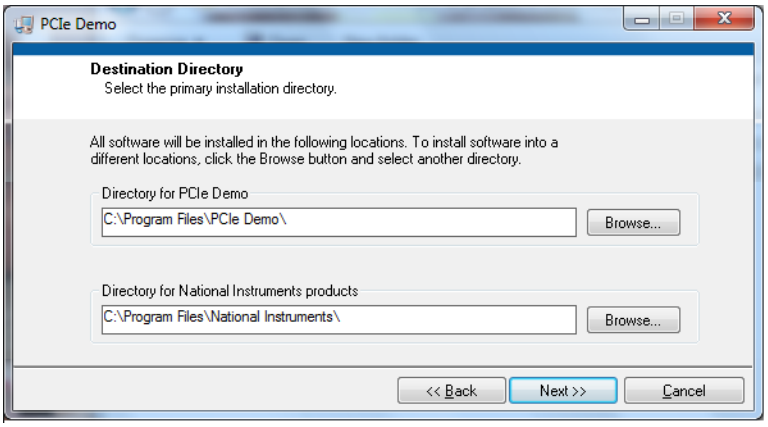
2.4.1.2 PCIe Demo GUI Installation

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.

To install the GUI:

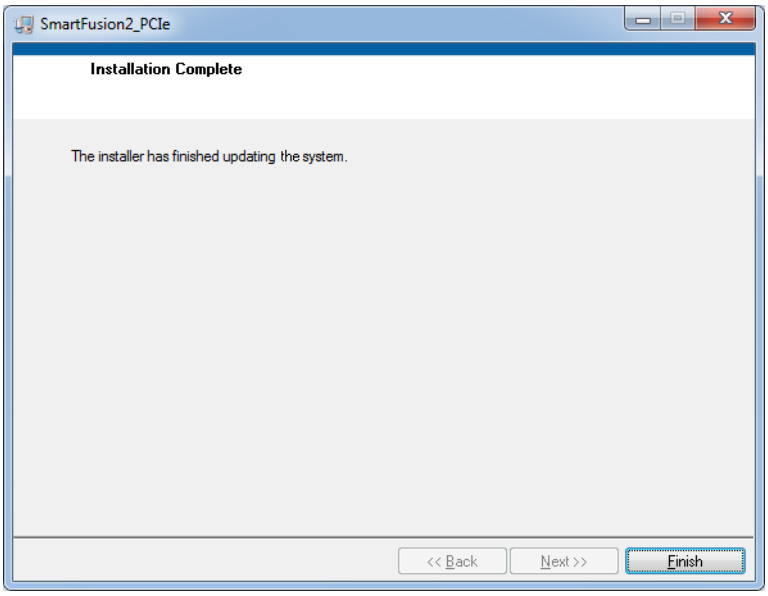
1. Extract the PCIe_Demo_GUI_Installer.rar and locate the files at **M2S150_PCIe_Control_Plane_DF\GUI**
2. Double-click the `setup.exe` in the provided GUI installation (*PCIe_Demo_GUI_Installer\setup.exe*). Apply default options as shown in the following figure.
3. To start the installation, click **Next**.

Figure 17 • **GUI Installation**



4. Click **Finish** to complete the installation.

Figure 18 • **Successful GUI Installation**



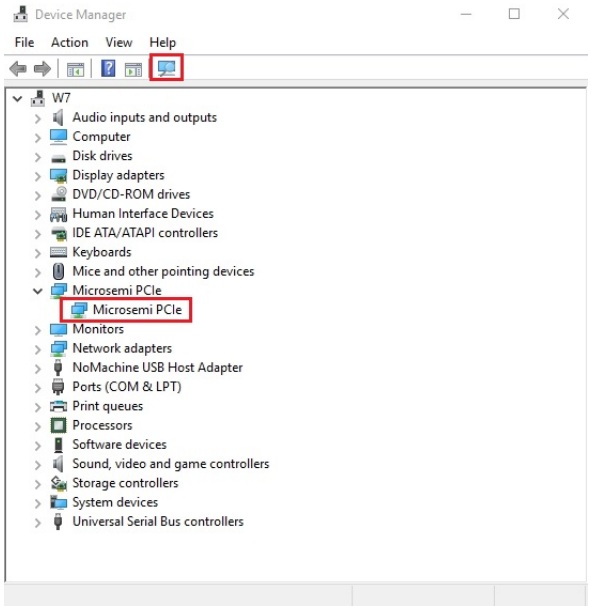
5. Restart the Host PC.

2.4.1.3 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. Click **Scan for hardware changes** in Device Manager. Ensure that the board is switched on.

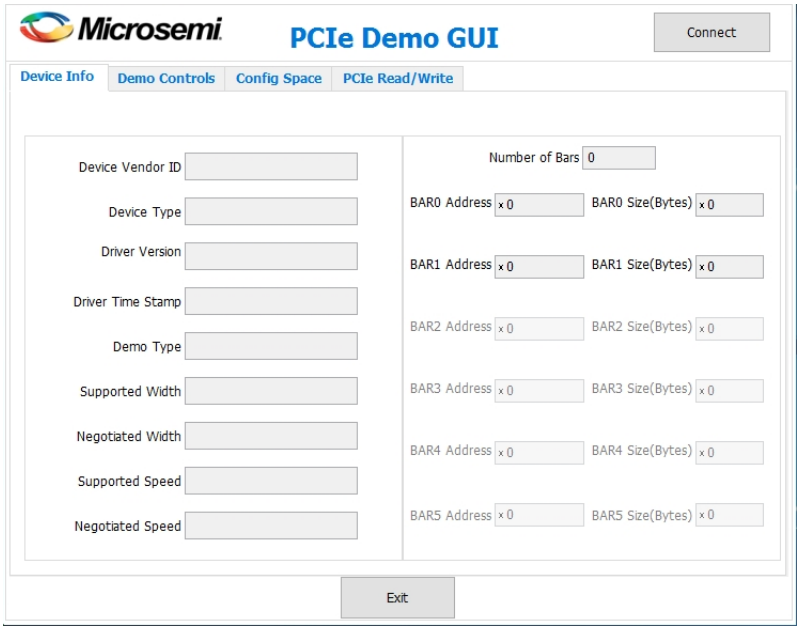
Figure 19 • **Device Manager—PCIe Device Detection**



Note: If a warning symbol is displayed on the **Microsemi PCIe** in the **Device Manager**, uninstall them and start from step1 of Drivers Installation, page 12.

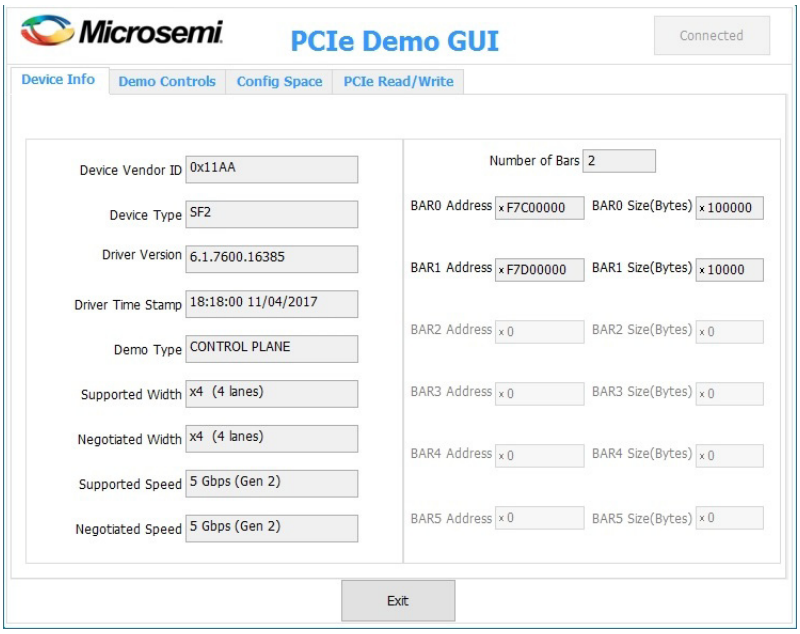
3. Invoke the GUI from **ALL Programs > PCIe Control Plane Demo**. The following figure shows the **PCIe Demo** GUI window.

Figure 20 • **PCIe Demo GUI**



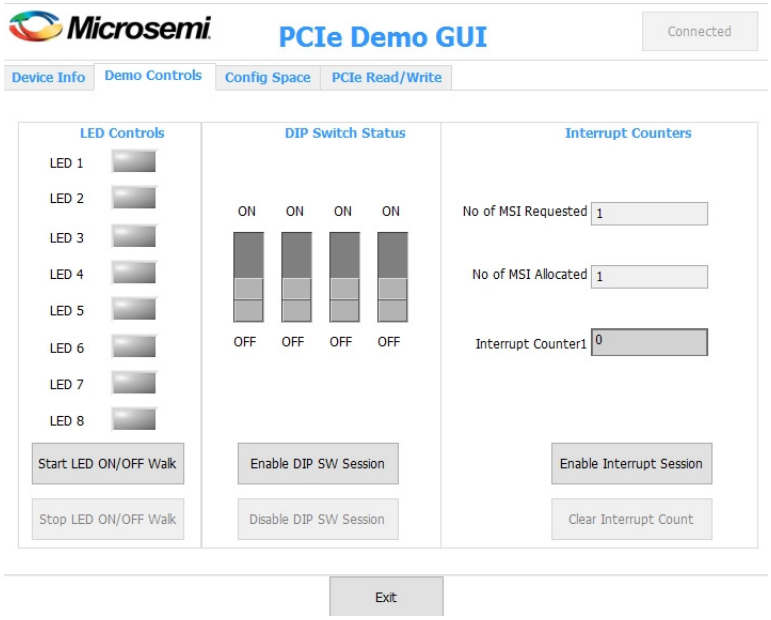
4. Click **Connect**. The application detects and displays the information related to the connected kit such as Device Vendor ID, Device Type, Driver Version, Driver Time Stamp, Demo Type, Supported Width, Negotiated Width, Supported Speed, Negotiated Speed, Number of Bars, and BAR Address as shown in the following figure.

Figure 21 • **Device Info**



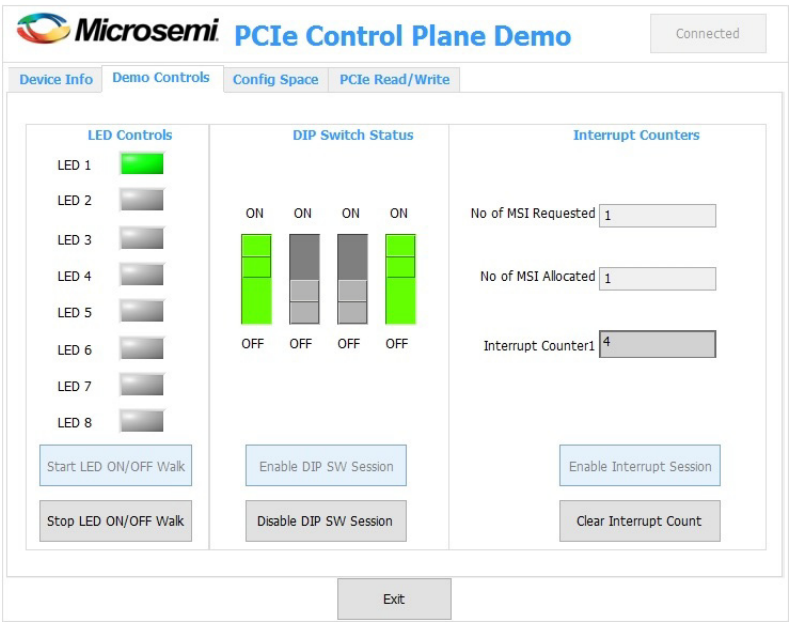
5. Click the **Demo Controls** tab to display the **LED Controls**, **DIP Switch Status**, and **Interrupt Counters** as shown in the following figure.

Figure 22 • **Demo Controls**



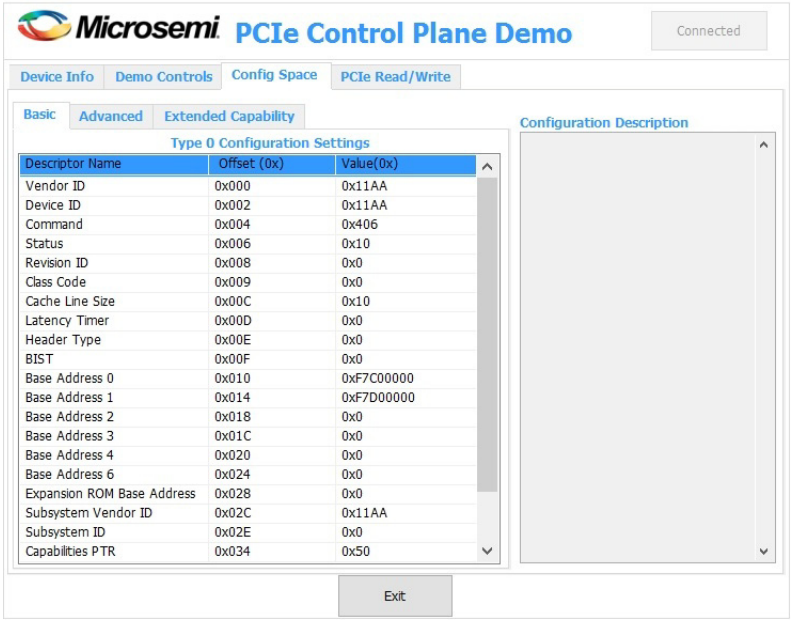
6. Click **Start LED ON/OFF Walk**, **Enable DIP SW Session**, and **Enable Interrupt Session** to view controlling LEDs, getting the DIP switch status, and monitoring the interrupts simultaneously as shown in the following figure.

Figure 23 • **Demo Controls—Continued**



7. Click **Config Space** to view details about the PCIe configuration space. The following figure shows the PCIe configuration space.

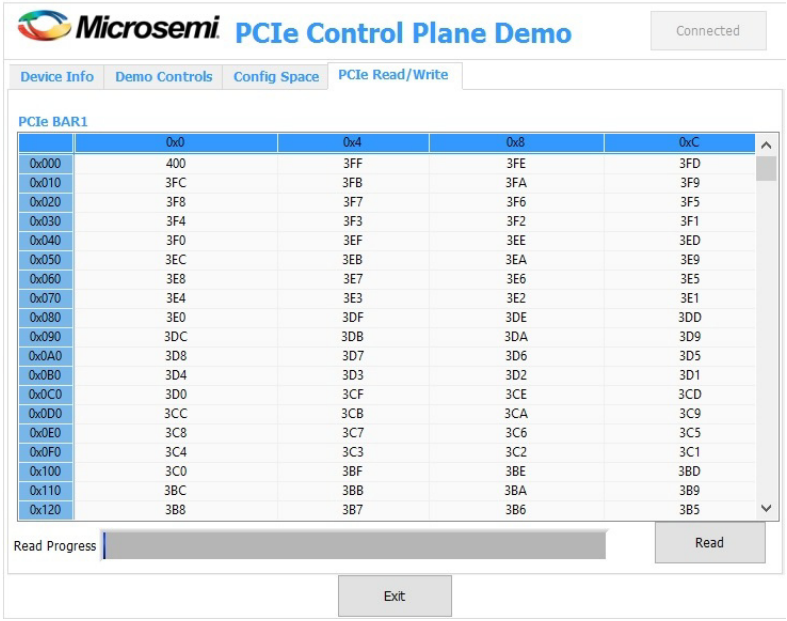
Figure 24 • **Configuration Space**



Descriptor Name	Offset (0x)	Value(0x)
Vendor ID	0x000	0x11AA
Device ID	0x002	0x11AA
Command	0x004	0x406
Status	0x006	0x10
Revision ID	0x008	0x0
Class Code	0x009	0x0
Cache Line Size	0x00C	0x10
Latency Timer	0x00D	0x0
Header Type	0x00E	0x0
BIST	0x00F	0x0
Base Address 0	0x010	0xF7C00000
Base Address 1	0x014	0xF7D00000
Base Address 2	0x018	0x0
Base Address 3	0x01C	0x0
Base Address 4	0x020	0x0
Base Address 6	0x024	0x0
Expansion ROM Base Address	0x028	0x0
Subsystem Vendor ID	0x02C	0x11AA
Subsystem ID	0x02E	0x0
Capabilities PTR	0x034	0x50

8. Click the **PCIe Read/Write** tab to perform read and writes to LSRAM memory through **BAR1** space. Click **Read** to read the 4 KB memory mapped to BAR1 space, as shown in the following figure.

Figure 25 • **PCIe BAR1 Memory Access**



	0x0	0x4	0x8	0xC
0x000	400	3FF	3FE	3FD
0x010	3FC	3FB	3FA	3F9
0x020	3F8	3F7	3F6	3F5
0x030	3F4	3F3	3F2	3F1
0x040	3F0	3EF	3EE	3ED
0x050	3EC	3EB	3EA	3E9
0x060	3E8	3E7	3E6	3E5
0x070	3E4	3E3	3E2	3E1
0x080	3E0	3DF	3DE	3DD
0x090	3DC	3DB	3DA	3D9
0x0A0	3D8	3D7	3D6	3D5
0x0B0	3D4	3D3	3D2	3D1
0x0C0	3D0	3CF	3CE	3CD
0x0D0	3CC	3CB	3CA	3C9
0x0E0	3C8	3C7	3C6	3C5
0x0F0	3C4	3C3	3C2	3C1
0x100	3C0	3BF	3BE	3BD
0x110	3BC	3BB	3BA	3B9
0x120	3B8	3B7	3B6	3B5

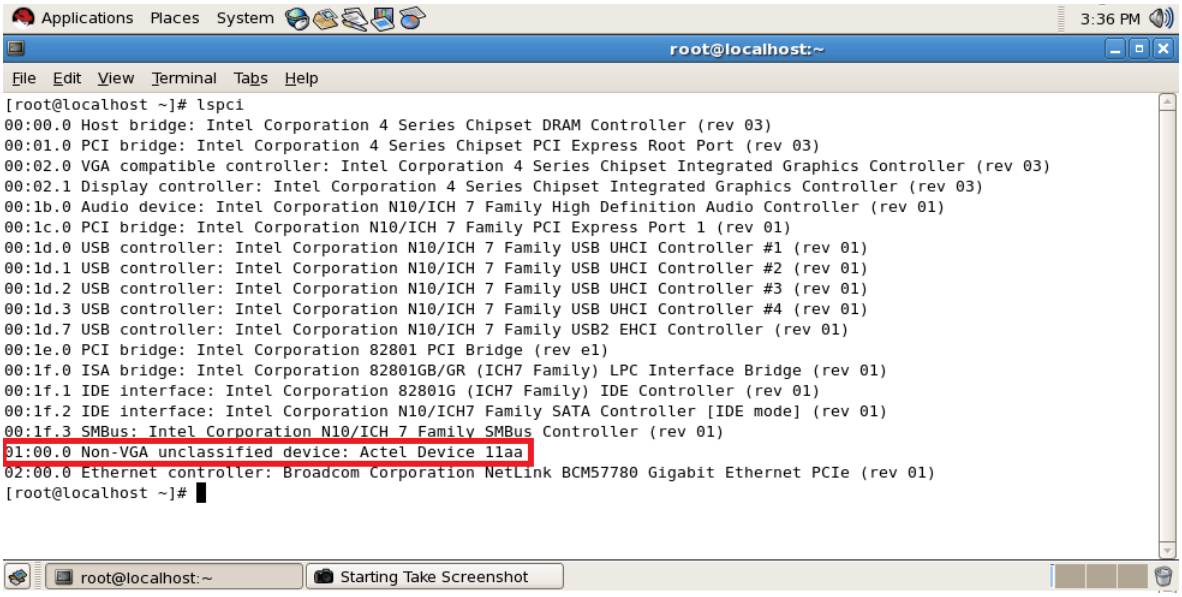
9. Click **Exit** to quit the demo.

2.4.2 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
   ```

Figure 26 • **PCIe Device Detection**



2.4.2.1 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/Driv-
   ers/PCIe_Driver.rar
   /home/sf2
   # unrar -e 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 27, page 20.
   ```
   #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 27 • **Edit board.h File**

```
Applications  Places  System                                                    7:12 PM
root@rhel-odigas:/home/prasad/pcie/LinuxPCIe_IGL2_Code/inc
File  Edit  View  Terminal  Tabs  Help
/**
*  SF2 : SmartFusion2 Board, IGL2: IGLOO2 Board
*  #define SF2, if the hardware board is SmartFusion2
*  #define IGL2, if the hardware board is IGLOO2
*/

#undef SF2_ADV_KIT
#undef IGL2
#undef SF2_DEV_KIT
#undef SF2_EVAL_KIT
~
~
~
~
~
~
~
~
"board.h" 10L, 231C
Screenshot-1.png    root@rhel-odigas:/home/prasad/pcie/LinuxPCIe_IGL2_Code/inc
```

Figure 28 • **PCIe Device Driver Installation**

```
Applications  Places  System                                                    7:17 PM
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
Makefile        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]#
root@localhost:/home/sf2/PCIe_Driver
```

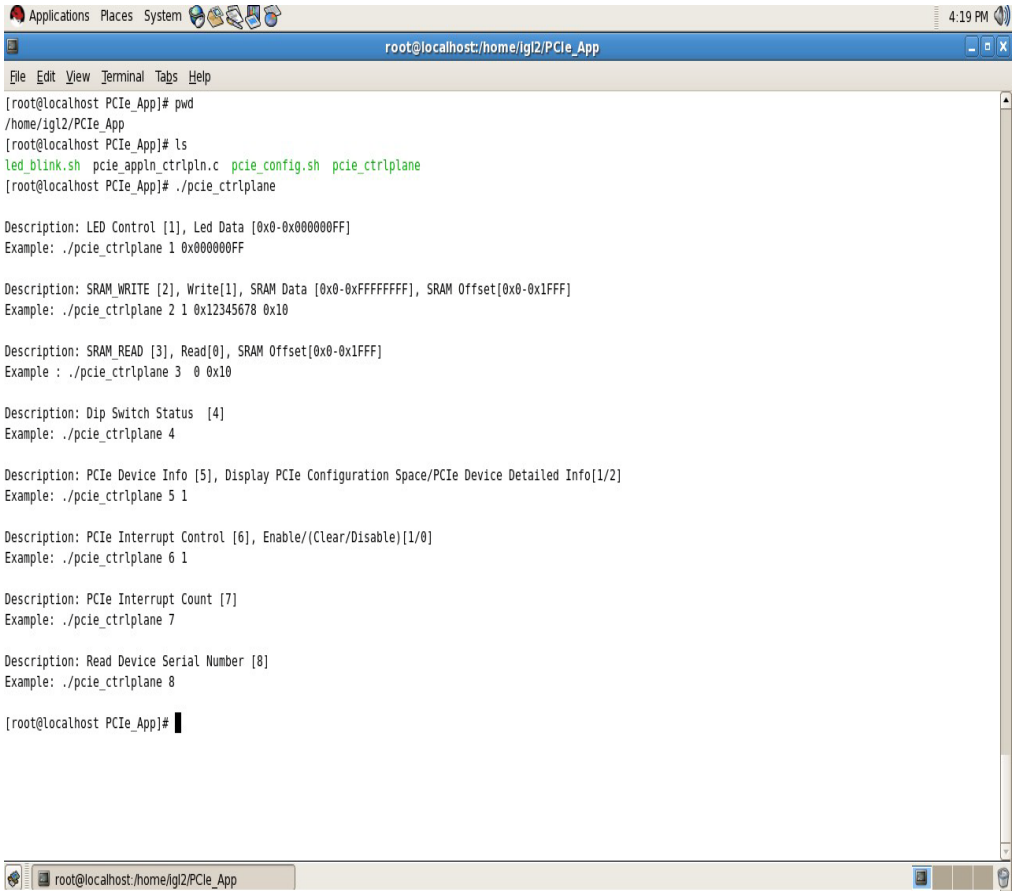
11. After successful Linux PCIe device driver installation, check `/dev/MS_PCI_DEV` got created by using the following Linux command:
    `#ls /dev/MS_PCI_DEV`

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

2.4.2.2 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
   # unrar -e PCIe_App.rar
   ```
   /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 the following figure.

Figure 29 • **Linux PCIe Application Utility**



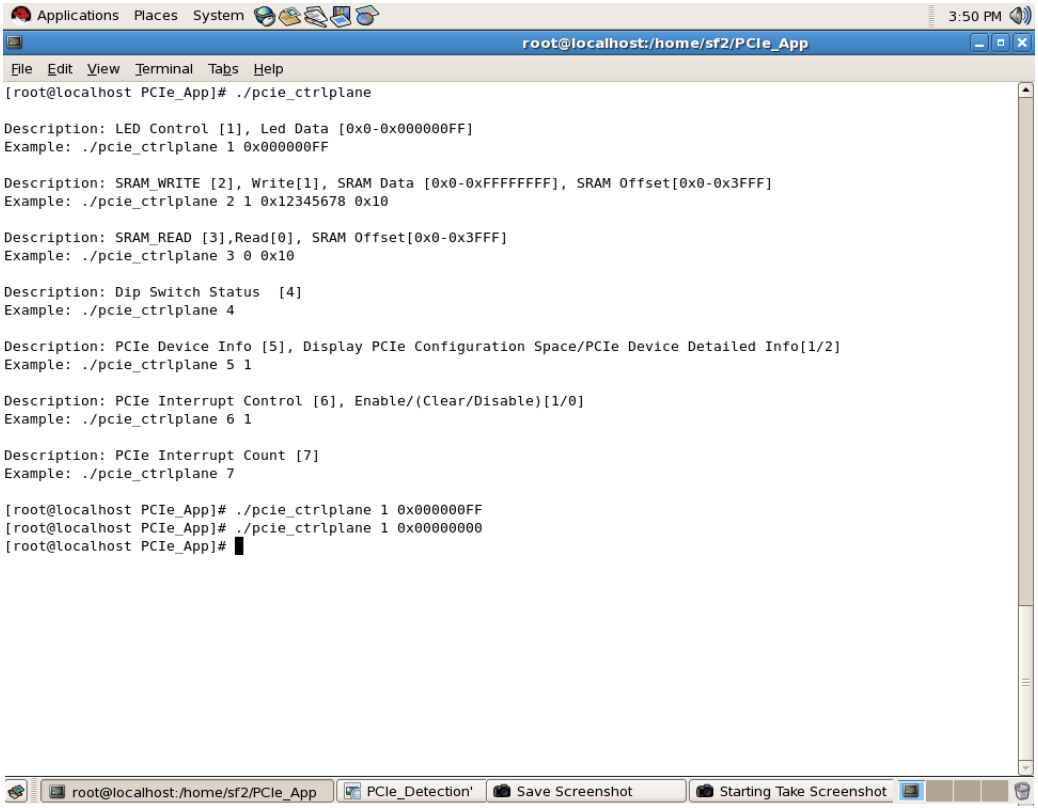
2.4.2.3 Execution of Linux PCIe Control Plane Features

2.4.2.3.1 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 30 • **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
```

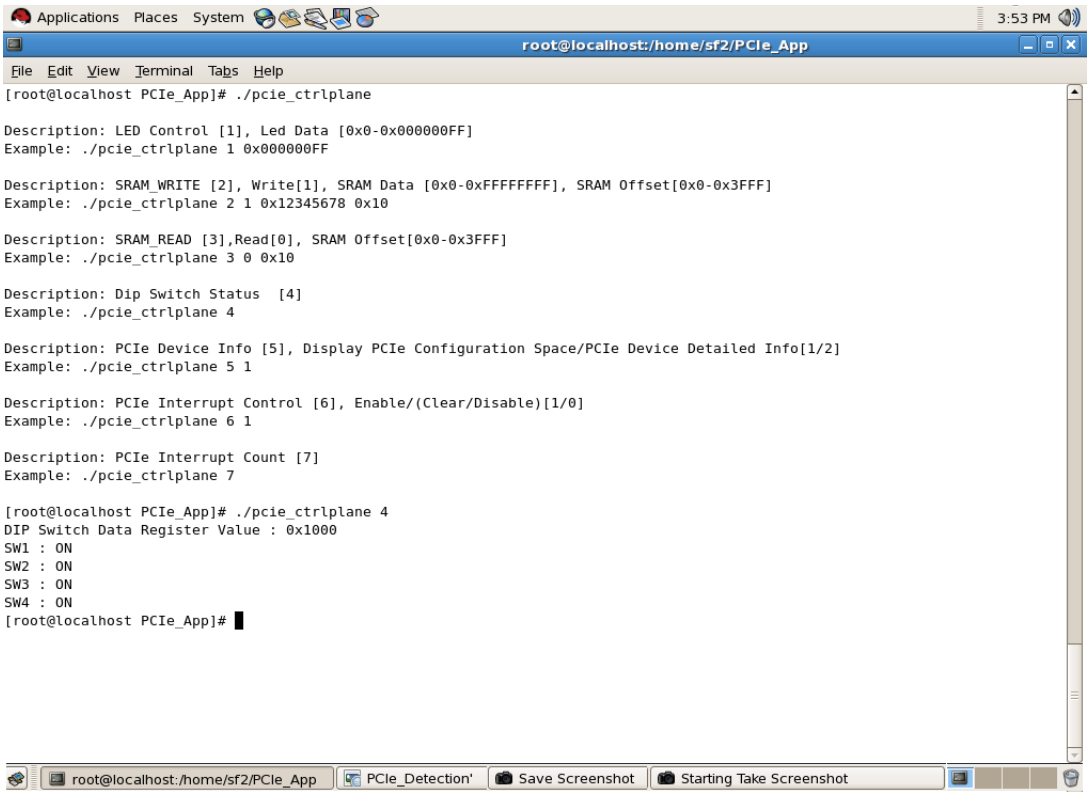
Run the `led_blink.sh` shell script using the `sh` command.

2.4.2.3.2 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]
```

Figure 31 • **Linux Command—DIP Switch**



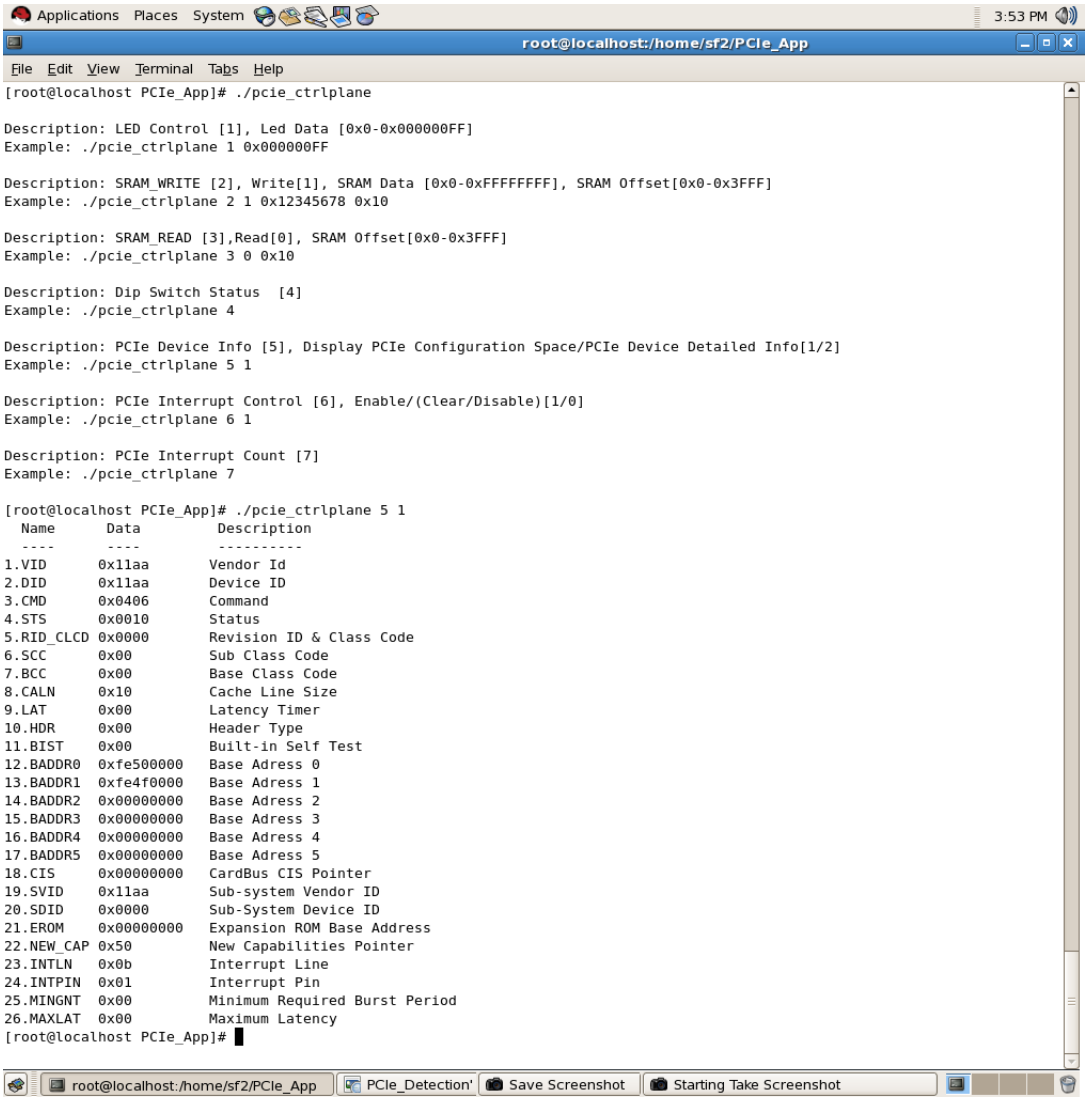
2.4.2.3.3 PCIe Configuration Space Display

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]
```

Figure 32 • **Linux Command—PCIe Configuration Space Display**



2.4.2.3.4 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]
```

Figure 33 • **Linux Command—PCIe Link Speed and Width**

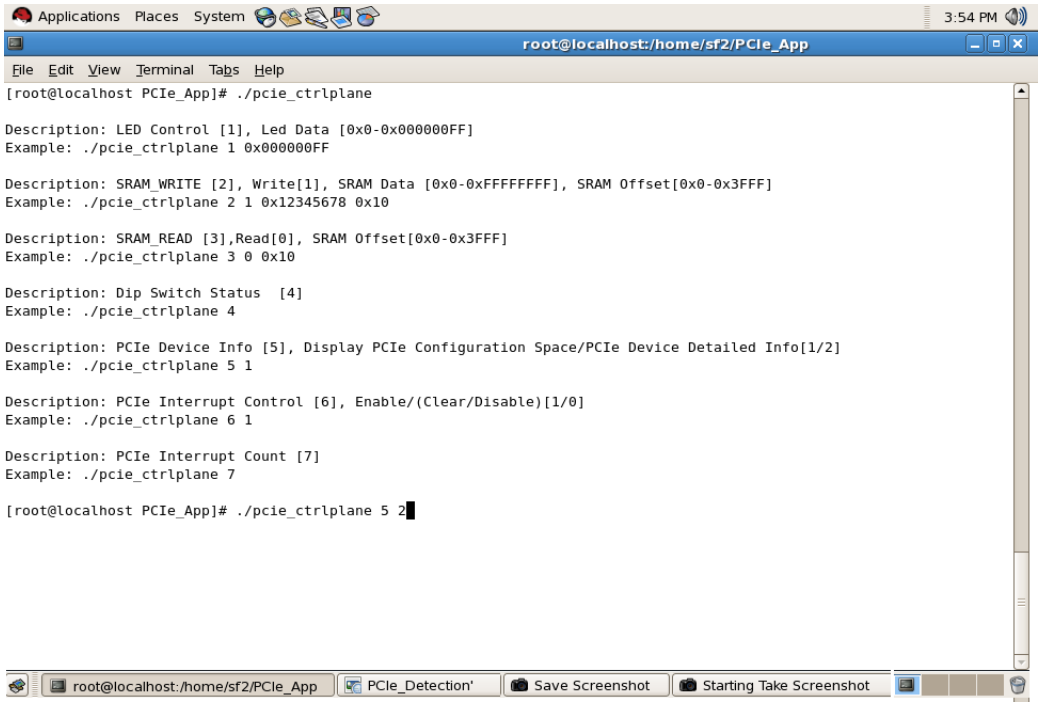


Figure 34 • **Linux Command—PCIe Link Speed and Width**

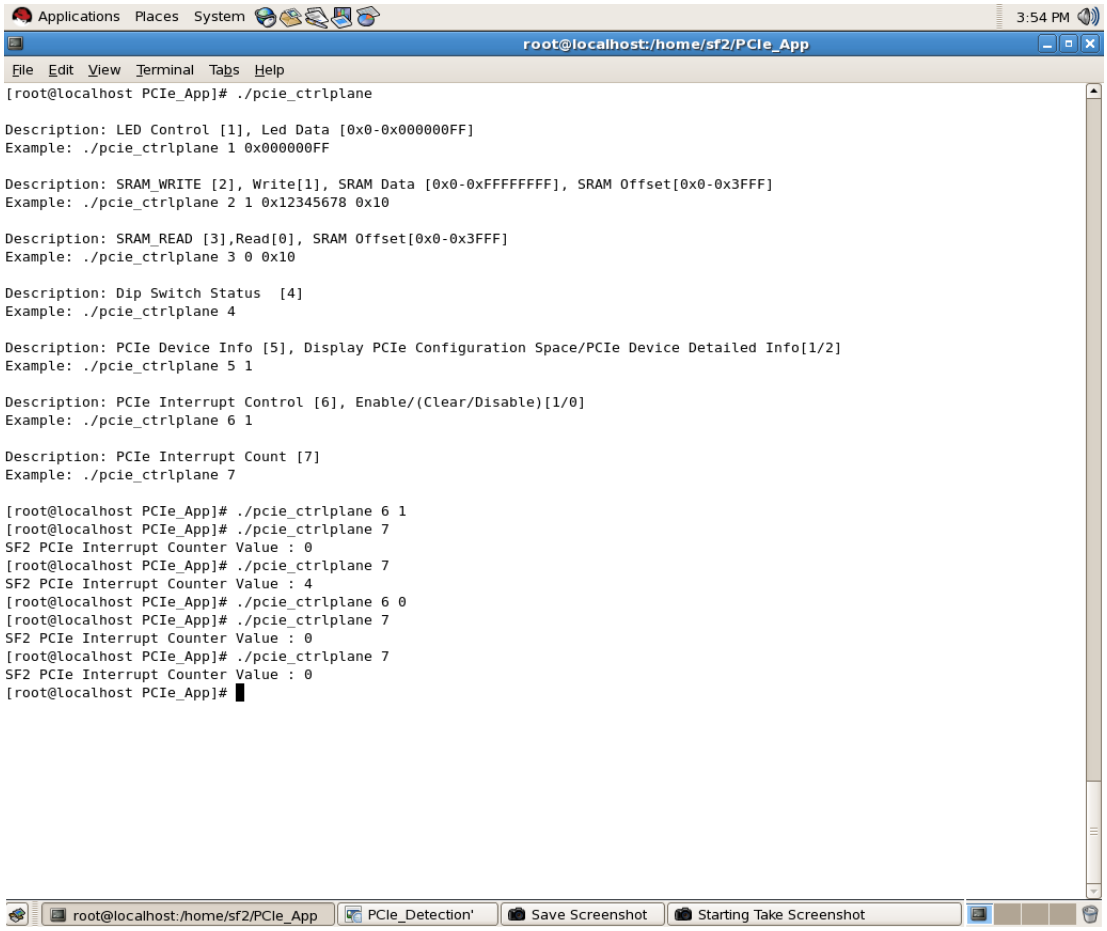
```
Applications  Places  System                                                    3:54 PM
root@localhost:/home/sf2/PCIe_App
File  Edit  View  Terminal  Tabs  Help
        Kernel driver in use: i801_smbus
        Kernel modules: i2c-i801

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-
                        RlxdOrd+ 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/PCIe_App   PCIe_Detection'   Save Screenshot   Starting Take Screenshot
```

2.4.2.3.5 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]
```

Figure 35 • **Linux Command—PCIe Interrupt Control**



2.5 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 Microsemi PCIe drivers for windows platform. It also provides a Linux PCIe application for easy control of PCIe EP device through Linux PCIe Device Driver.

3 Appendix: SmartFusion2 Advanced Development Kit Board

The following figure shows the SmartFusion2 Advanced Development Kit board.

Figure 1 • **SmartFusion2 Advanced Development Kit Board.**