

<b>Unit / Module Description:</b>	Installation and first step
<b>Unit / Module Number:</b>	SMT6058
<b>Document Issue Number:</b>	2.0
<b>Issue Date:</b>	23/09/09
<b>Original Author:</b>	F.S

# **Application Note for SMT6058**

## **Ethernet support for the SMT148- FX standalone carrier board**

Sundance Multiprocessor Technology Ltd, Chiltern House,  
Waterside, Chesham, Bucks. HP5 1PS.

This document is the property of Sundance and may not be copied  
nor communicated to a third party without prior written  
permission.

© Sundance Multiprocessor Technology Limited 2009



Certificate Number FM 55022

## Revision History

<b>Issue</b>	<b>Changes Made</b>	<b>Date</b>	<b>Initials</b>
1.0	First release	30/07/09	FS
2.0	SMT6058 BootLoader and release modification	23/09/09	FS

# Table of Contents

<b>1</b>	<b>Introduction</b> .....	<b>5</b>
<b>2</b>	<b>Related Documents</b> .....	<b>5</b>
2.1	Referenced Documents .....	5
<b>3</b>	<b>Acronyms, Abbreviations and Definitions</b> .....	<b>5</b>
3.1	Acronyms and Abbreviations .....	5
<b>4</b>	<b>Installation</b> .....	<b>6</b>
4.1	Installation specification .....	6
4.2	EDK patch.....	6
<b>5</b>	<b>Development procedure</b> .....	<b>7</b>
5.1	SMT148FX configuration .....	7
5.2	The IP configuration.....	8
5.3	The BootLoader Application .....	9
5.3.1	File generation .....	9
5.3.2	Programming the flash with the SMT6002 .....	11
5.3.3	Programming the flash with the SMT6058 Tool .....	11
5.4	The Applications .....	13
5.4.1	The Web Server .....	14
5.4.2	Modify the Web Server .....	16
5.4.3	The Ethernet with SMT362 comport loopback.....	18
5.4.4	TFTP.....	20
5.4.5	ECHO.....	21

## Table of Figures

Figure 1 : Impact CPLD programming .....	7
Figure 2 : TCP/IP configuration .....	8
Figure 3 : Linker Script Generator .....	9
Figure 4 : Flash SREC file generation .....	10
Figure 5 : SMT6002 Flash Programming Utility for FPGA .....	11
Figure 6 : SMT6058 Flash Programming Tool .....	12
Figure 7 : image.mfs initialization .....	14
Figure 8 : SMT148-FX Web Server .....	15
Figure 9 : HyperTerminal .....	16
Figure 10 : image.mfs generation .....	17
Figure 11 : Web Server modified .....	17
Figure 12 : Virtex 4 comports connections .....	18
Figure 13 : Host application .....	18
Figure 14 : Debug and 3L server result .....	19
Figure 15 : TFTP example .....	21
Figure 16 : Echo example .....	21

# 1 Introduction

This document specifies the SMT6058 installation procedure and the first step to run the design examples.

The SMT6058 product will allow accessing the SMT148-FX60 carrier boards via its Gigabit Ethernet port (onboard RJ45 connector). The SMT6058 includes a software TCP/IP stack for the Gigabit Ethernet interface, a default firmware implementing a PowerPC core for the Virtex-4 FX60 FPGA device, the software functions to access the board resources (flash programming) and on-board modules via Rocket-IO Serial Links and/or Comport link from the gigabit Ethernet port (TCP/IP stack).

## 2 Related Documents

### 2.1 Referenced Documents

[SMT6058 : Product specification](#)

[SMT148FX : 4 site stand alone TIM carrier](#)

[SMT362 : Dual 'C6455 DSP Module](#)

## 3 Acronyms, Abbreviations and Definitions

### 3.1 Acronyms and Abbreviations

[A list of acronyms etc](#)

## 4 Installation

### 4.1 Installation specification

Xilinx EDK does NOT support 'spaces' in the project path.

Copy the SMT6058 folder and make sure that you have no space in your project path. (good path example : C:\SMT6058)

The folder contains one EDK project:

“..\SMT6058\Applications\EDK\_project\system.xmp”

This project includes two software application projects named:

- SMT6058\_BootLoader
- Applications

The SMT6058\_BootLoader application is used to load your application from flash to the SMT148FX zbt memory.

The project named Applications includes all the applications:

- APP\_WEBSERVER
- APP\_ETH2CP
- APP\_TFTPSERVER
- APP\_ECHOSERVER

### 4.2 EDK patch

The EDK patch, in the EDK\_Sources\_Patch folder, is the replacement files for the IITEMAC (driver for the Ethernet) and the LWiP TCP/IP stack.

- The archive litemac\_v1\_00\_b should replace the one installed in :

%%:\Xilinx\10.1\EDK\sw\XilinxProcessorIPLib\drivers

- The archive lwip\_v3\_00\_a should replace the one installed in :

%%:\Xilinx\10.1\EDK\sw\ThirdParty\sw\_services

This patch initializes properly the PHY in the correct speed. This is required to adjust the chipset's delay as well.

This would not be needed if the Xilinx ethernet auto-negotiation worked as expected.

If you haven't yet installed the patch and already use the SMT6058, just clean and rebuild your software application to make sure that you will use the right drivers.

## 5 Development procedure

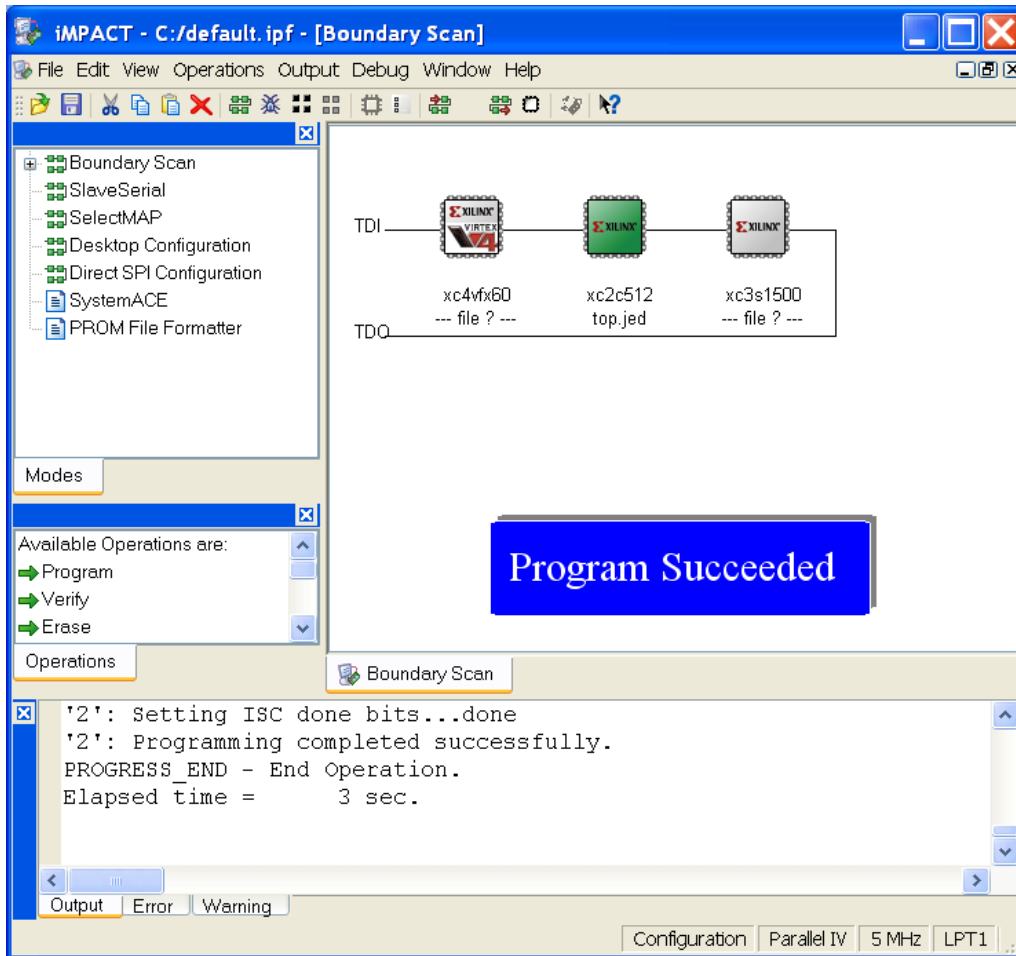
### 5.1 SMT148FX configuration

The SMT148-FX has to be configured to run the SMT6058 applications.

First update the CPLD firmware with the firmware:

“..\SMT6058\Hardware\CPLD\top.jed”

This new firmware gives the possibility to reset the SMT148FX from the comport (i.e. from the TIMs or from the SMT148FX Virtex4).



**Figure 1 : Impact CPLD programming**

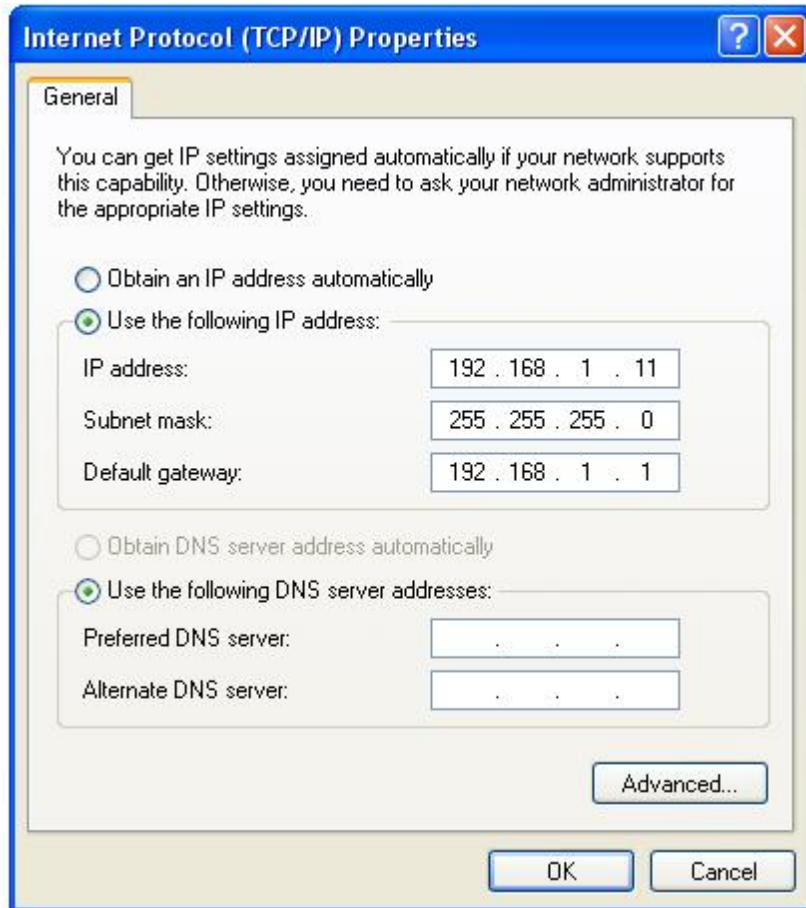
The SMT6058 project uses the comport of the SMT148-FX Virtex4, so the Spartan has to be programmed with the firmware provided in the SMT6058 package.

“..\SMT6058\Hardware\FPGA\XCS1500\com.sundance.smt148-fx.sc3s1500.fx60toflash\ com.sundance.smt148-fx.sc3s1500.fx60toflash.app”

This firmware connects the SMT148-FX Virtex4 to its flash through the comport and to the T1CP4.

## 5.2 The IP configuration

Make sure that you have no firewall on port 80 of your network. For a direct link between your computer and the SMT148-FX Ethernet connector configure your IP like the following picture, or make sure that you have a good configuration depending on your project application.



**Figure 2 : TCP/IP configuration**



## 5.3 The BootLoader Application

### 5.3.1 File generation

The PPC applications (.elf) and data (Webserver page) are store in the external memory (all the sections are defined in the linker script of your application).

To use the bootloader, make sure all the section are in the external memory, but the boot and vector sections must be in BRAM. If a section is in BRAM, the bootloader application (which is stored in BRAM) will probably write on its own code and the loading will fail.

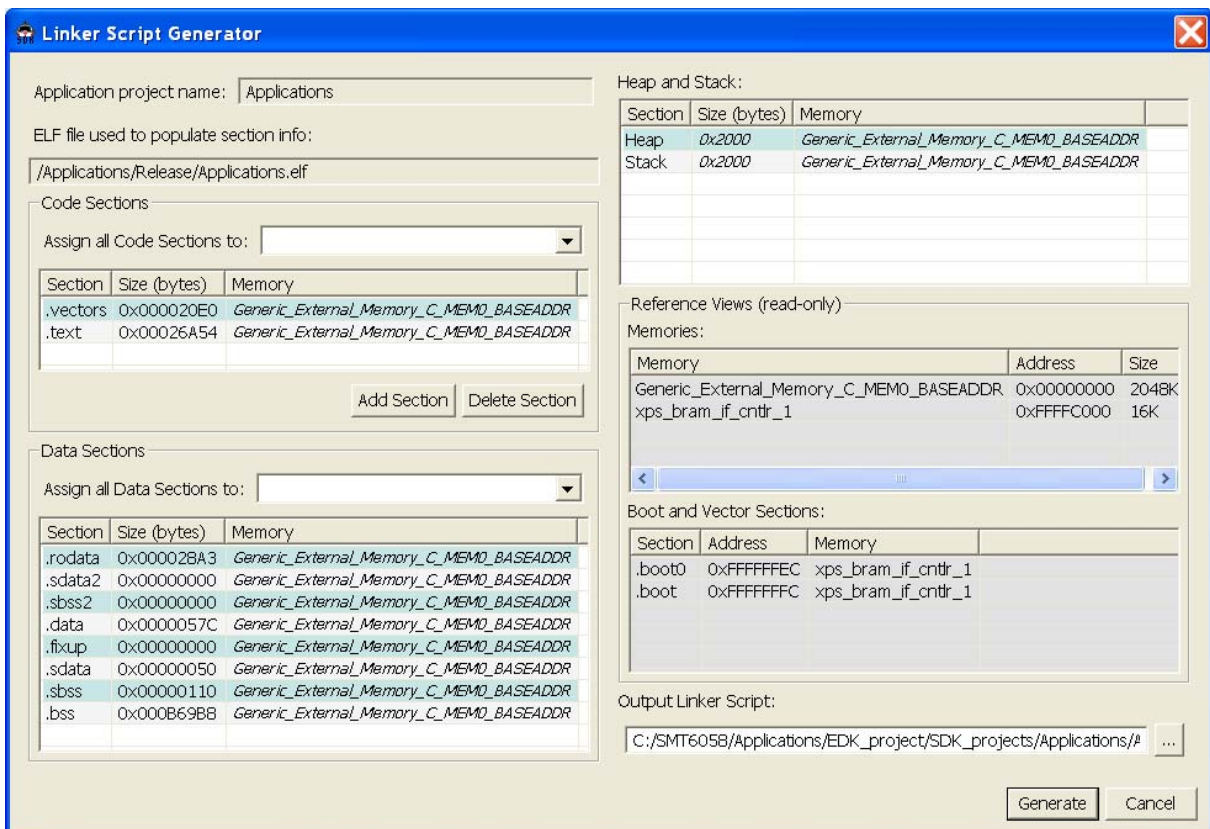

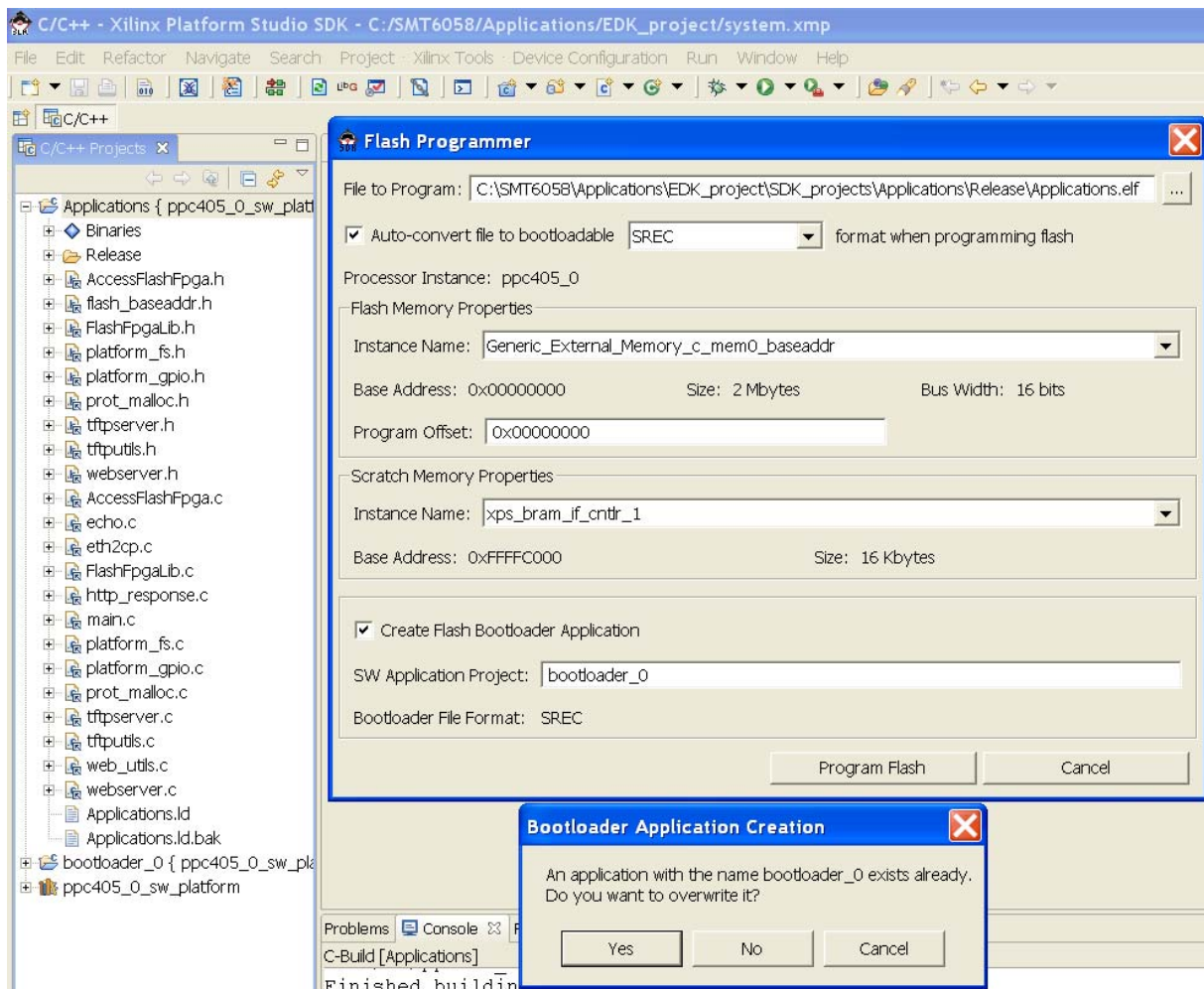


Figure 3 : Linker Script Generator

You need to generate the .srec file for the Bootloader. Select Device Configuration and  Program Flash Memory..., then select the file to program, here ..\SDK\_projects\Applications\Release\Applications.elf

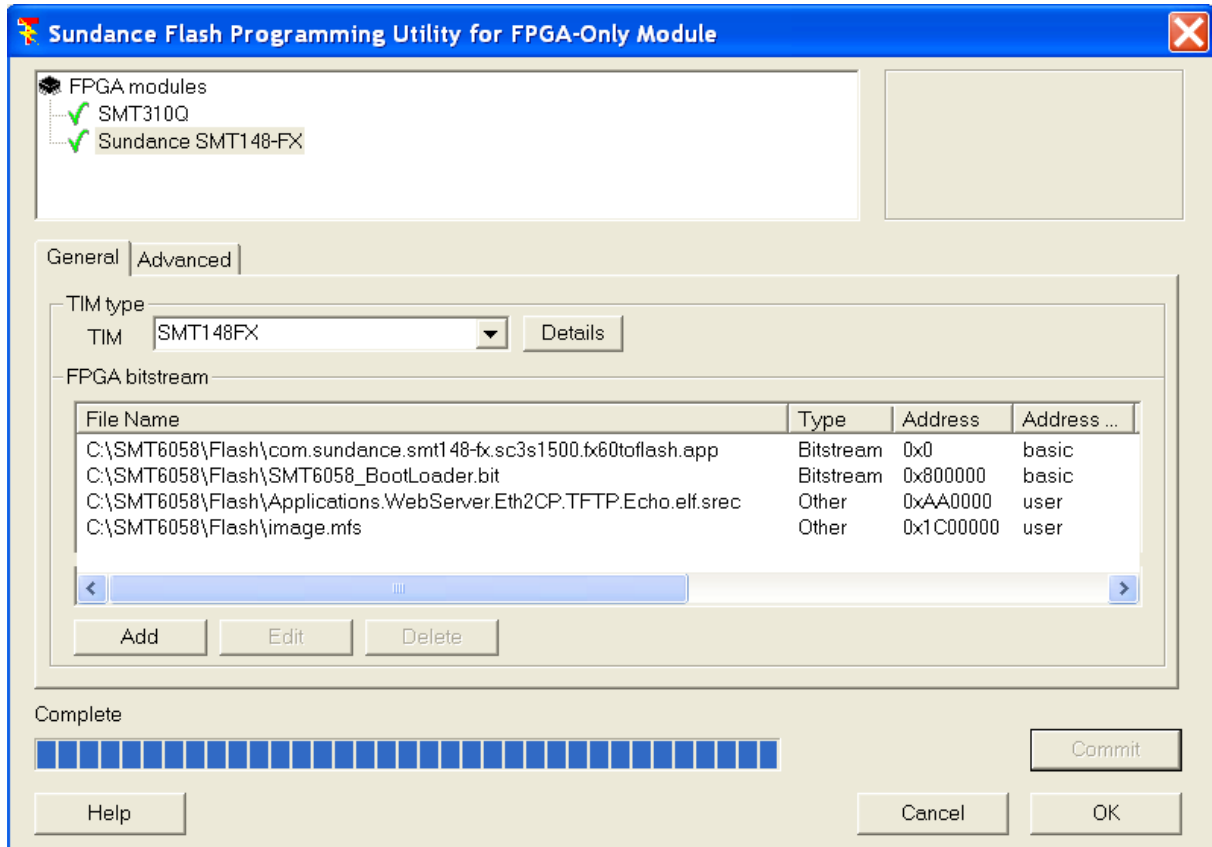
Make sure that all is selected like the following picture; don't pay attention to the flash memory program offset, because when you will click "Program Flash" it will not program the SMT148FX flash, you have to put the .srec file in flash yourself with the SMT6002 or the SMT6058 Flash Programming Tool.

This step will create a new bootloader application every time, but you have to use the SMT6058\_Bootloader application. Remember to always write for the SW Application project : bootloader\_0 and overwrite the project bootloader\_0.



**Figure 4 : Flash SREC file generation**

### 5.3.2 Programming the flash with the SMT6002



**Figure 5 : SMT6002 Flash Programming Utility for FPGA**

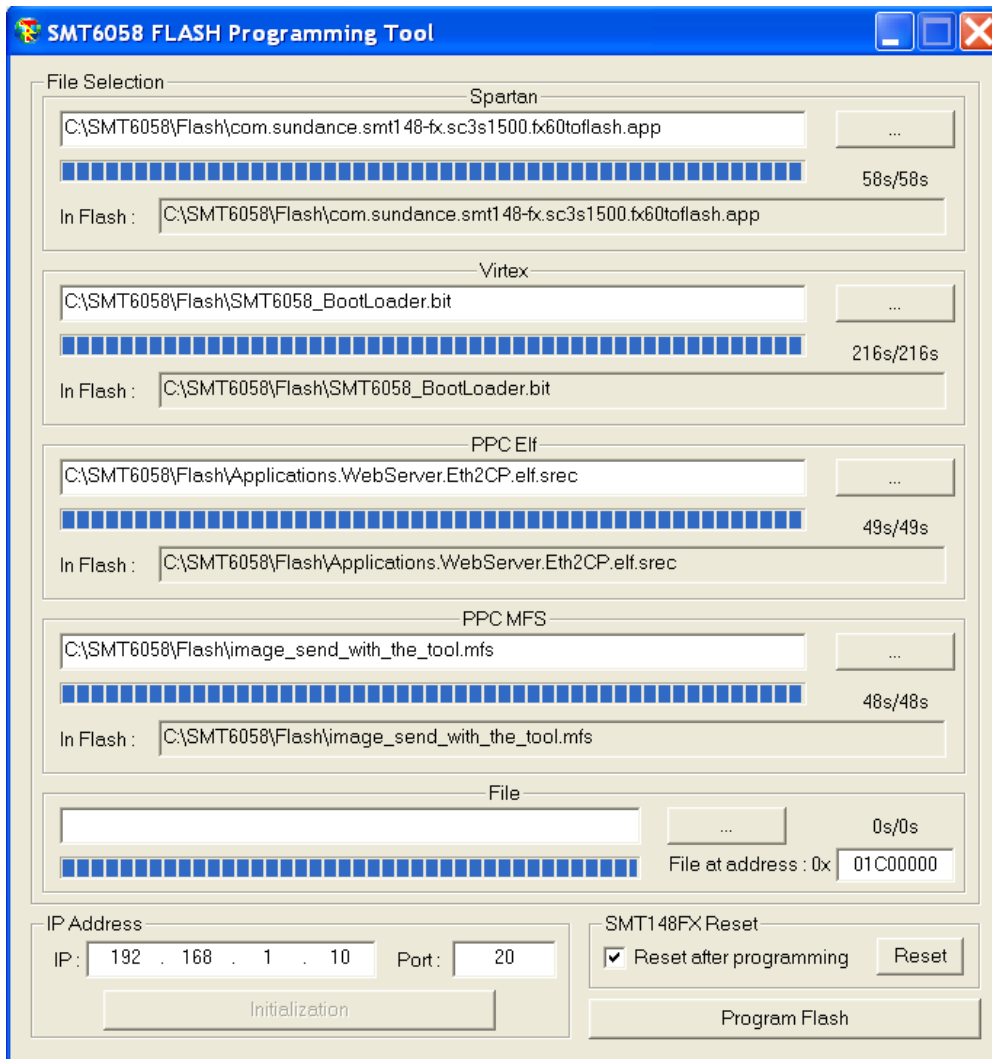
All the data in flash needs to be loaded to a specific Address; for the Spartan bistream it must be 0x0, and for the Virtex4 it must be 0x800000. The PowerPc Application and Memory File System flash address are defined in the application at the address 0xAA0000 for the SREC file and 0X1C00000 for the MFS file. If you change the address be sure that they are defined, and the same in your host and PPC application.

### 5.3.3 Programming the flash with the SMT6058 Tool

You need to use the SMT6002 at least one time before you can use the SMT6058 Flash Programming Tool. Program the flash as explain in the previous section. To be able to use the SMT6058 Flash Programming Tool you need at least the Ethernet to Comport application in the PPC to get the flash access from the PPC.

The SMT6058\_Flash\_Programming\_Tool.exe tool is in the directory:

..\SMT6058\Host\SMT6058\_Flash\_Programming\_Tool\



**Figure 6 : SMT6058 Flash Programming Tool**

First check the IP address and the port, and then initialize the connection. The initialization just tries to connect the host to the board and get the flash information.

You can select the file that you want to store in flash and program it. By default there is a reset request sent to the board after programming to reload all the bitstreams and application. Alternatively, you can uncheck this option and send the reset request with the Reset button.

You don't have to upload all the files every time, just try to replace the Power PC elf and mfs file in example with:

PPC Elf :       “..\SMT6058\Flash\Applications.WebServer.Eth2CP.elf.srec”

PPC MFS:       “..\SMT6058\Flash\image\_send\_with\_the\_tool.mfs”

This elf application is just the WebServer and Ethernet to Comport threads.

The mfs file is another html page with some different text.

Uploading firmware in the flash may prevent the board to communicate over ethernet. If an erroneous bitstream is placed in the flash, the virtex will have to be uploaded using JTAG, USB or the SMT6002.

## 5.4 The Applications

What is in the design?

- PPC running at 300MHz,
- Kernel working on the PowerPC with cache memory,
- Multithreading fonctionnality,
- NO DMA ENGINES
- ZBT RAM implemented
- On-board UART for debug output (if the RS-232 is connected to the board while the design is executed, some feedback from the PPC is sent. This can be observed with HyperTerminal)
- WebServer (GUI/LED control)
- Ethernet Link connection at 10/100/1000Mbps

There are four application (threads) available in the project.

Open the “..\SMT6058\Applications\EDK\_project\system.xmp” project, Launch Platform Studio SDK, and open the file “main.c” .

You just have to define which application you want to include in you project at the line 33.

Example:

```
/* list of applications to be included */
#define APP_WEBSERVER
#define APP_ETH2CP
//#define APP_TFTPSERVER
//#define APP_ECHOSERVER
```

This definition will start the WebServer thread and the Ethernet to Comport thread.

Two pre-build applications are available to test the BootLoader.

“..\SMT6058\Flash\ Applications.WebServer.Eth2CP.elf.srec”

“..\SMT6058\Flash\Applications.WebServer.Eth2CP.TFTP.Echo.elf.srec”

**To execute all the following instruction use:**

Applications.WebServer.Eth2CP.TFTP.Echo.elf.srec

### 5.4.1 The Web Server

```
/* list of applications to be included */  
#define APP_WEBSERVER
```

This project provides an html page to control the SMT148FX LEDs from a web browser. This page is saved in the image.mfs file.

If you want to initialize this file when you run your application from SDK, comment the line 23 in the file "platform\_fs.c".

```
#define load_mfs_from_flash_to_memory
```

This file has to be added during the initialization at the address 0x130000.

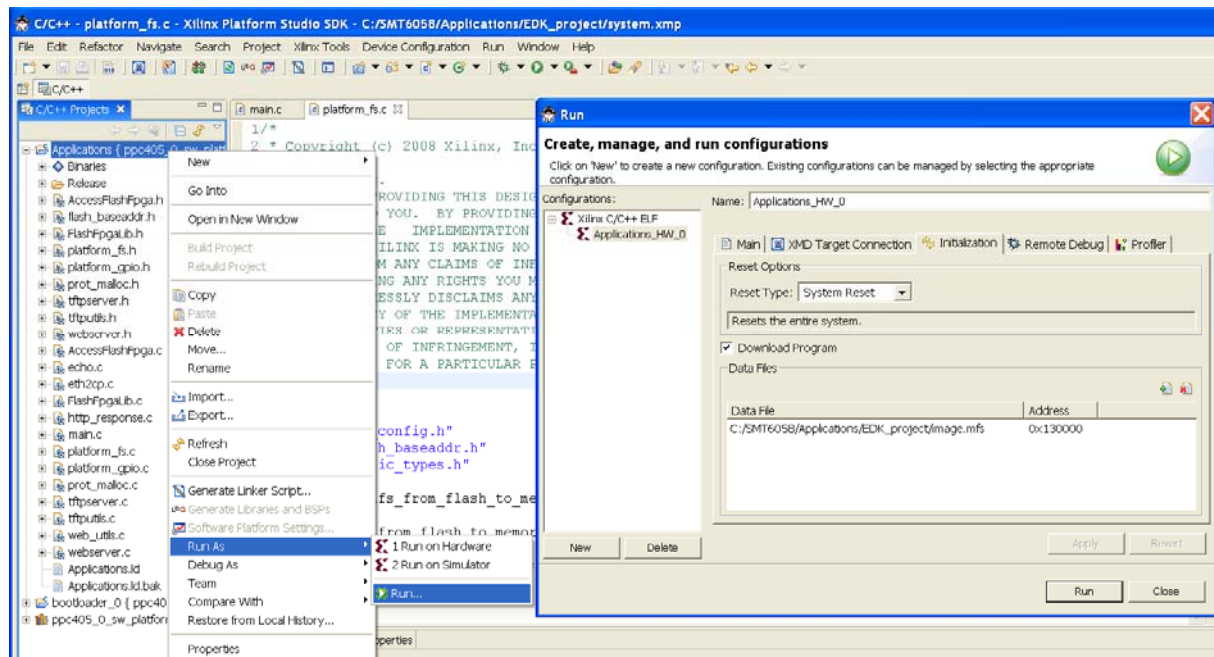


Figure 7 : image.mfs initialization

Before running the application, make sure that you have connected the SMT148-FX60 FPGA via the FPGA JTAG chain using the Xilinx JTAG pod connected to the JP6 header on the SMT148-FX.

Connect the SMT148-FX RJ45 connector to the PC ethernet card via an Ethernet cable. If the RS-232 is connected to the board while the design is executed, some feedback from the PPC is sent. This can be observed with HyperTerminal.

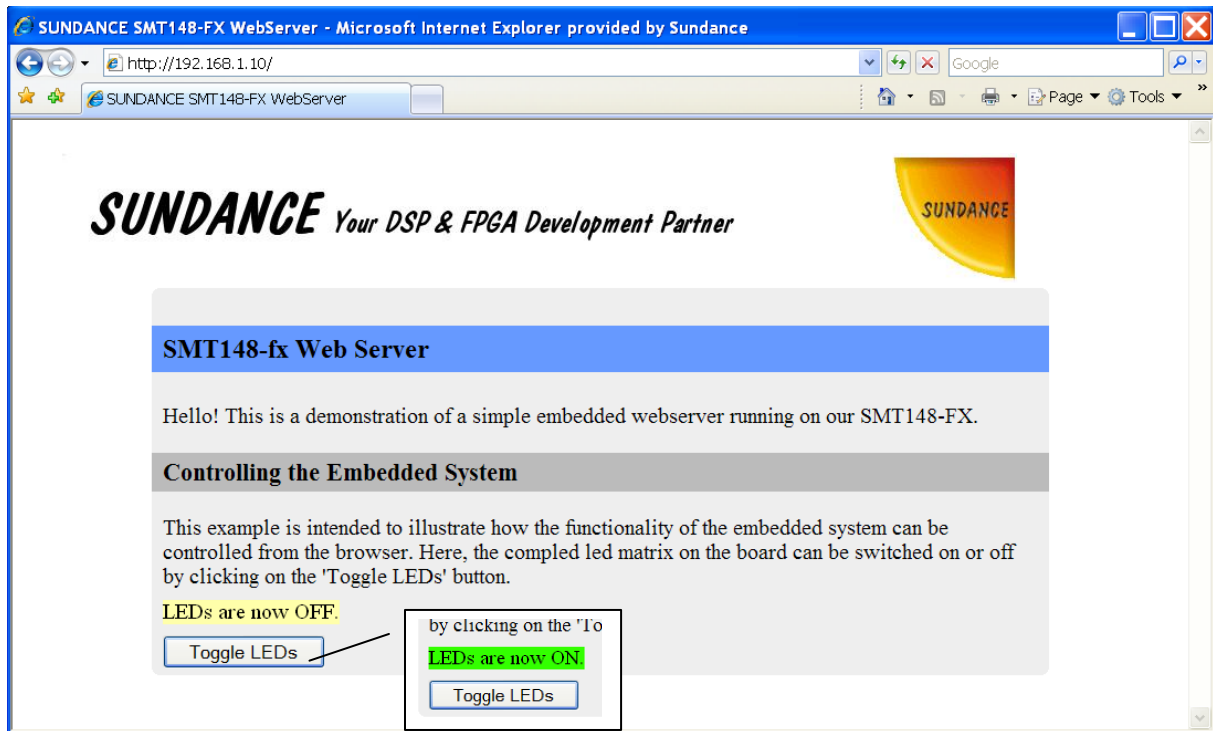
Now program the FPGA  and after run the Webserver application .

You can test the connection between the PC and the PowerPC (EMAC connections). Open a command prompt window and type the instruction: PING 192.168.1.10

And in a web browser, type: 192.168.1.10

A simple web page is displayed, to demonstrate that the Webserver is running on the PowerPC.

LEDs can be controlled from the web server to demonstrate the communication of system Ethernet-PPC and peripherals.



**Figure 8 : SMT148- FX Web Server**

When you click on the 'Toggle LEDs' button, the LEDs status is shown on the Web Server and you can observe that all the SMT148FX LEDs matrix are ON or OFF on the board.

If you have plugged the RS232, you should have the same result as the following picture.

```
-----lwIP test WebServer -----
Open up your favorite browser and type:
http://192.168.1.10

Board IP: 192.168.1.10
Netmask : 255.255.255.0

Gateway : 192.168.1.1
XL11emac detect_phy: No PHY detected. Assuming a PHY at address 0
auto-negotiated link speed: 1000
PhySetup_Marvell_88e111: Try to set speed of 1000 Mbps
PhySetup_Marvell_88e111: Retries 4
PhySetup_Marvell_88e111: Link is fine

Memory File System initialized
http GET: index.html
http GET: yui/yahoo.js
http GET: yui/dom.js
http GET: yui/event.js
http GET: yui/conn.js
http GET: yui/anim.js
http GET: js/main.js
http GET: css/main.css
http GET: images/sundance.JPG
http POST: switch state: 0
http POST: ledstatus: FFFFFFFF
http POST: ledstatus: 0
http POST: ledstatus: FFFFFFFF
```

**Figure 9 : HyperTerminal**


#### **5.4.2 Modify the Web Server**

The html page is “..\SMT6058\Applications\EDK\_project\memfs\index.html”.

To show how to modify the page and make the image.mfs, start by editing the index.html code.

For a little example we will centre the Sundance image: line 17 should become:

```
<center></img></center>
```

Now you need to generate the image.mfs file. Under EDK or SDK open the Shell .

Go inside your memfs folder, it should be with :

- cd memfs

And now generate the image.mfs with :

- make



```

/cygdrive/c/SMT6058/Example/Webserver/memfs
EDK Shell
Xilinx EDK 10.1.03 Build EDK_K_SP3.6
Copyright (c) 1995-2008 Xilinx, Inc. All rights reserved.

Analyzing Cygwin versions...
Xilinx EDK detected Cygwin installation v1.5.17(0.129/4/2) on your machine.
This Cygwin (C:\Xilinx\10.1\EDK\cygwin\bin) will be used to run Xilinx EDK
tools.

Fabiens@SudacaXP /cygdrive/c/SMT6058/Example/Webserver
$ cd memfs


Fabiens@SudacaXP /cygdrive/c/SMT6058/Example/Webserver/memfs
$ make
mfsgen -cubfs ../image.mfs 1000 x
mfsgen
Xilinx EDK 10.1.03 EDK_K_SP3.6
Copyright (c) 2004 Xilinx, Inc. All rights reserved.

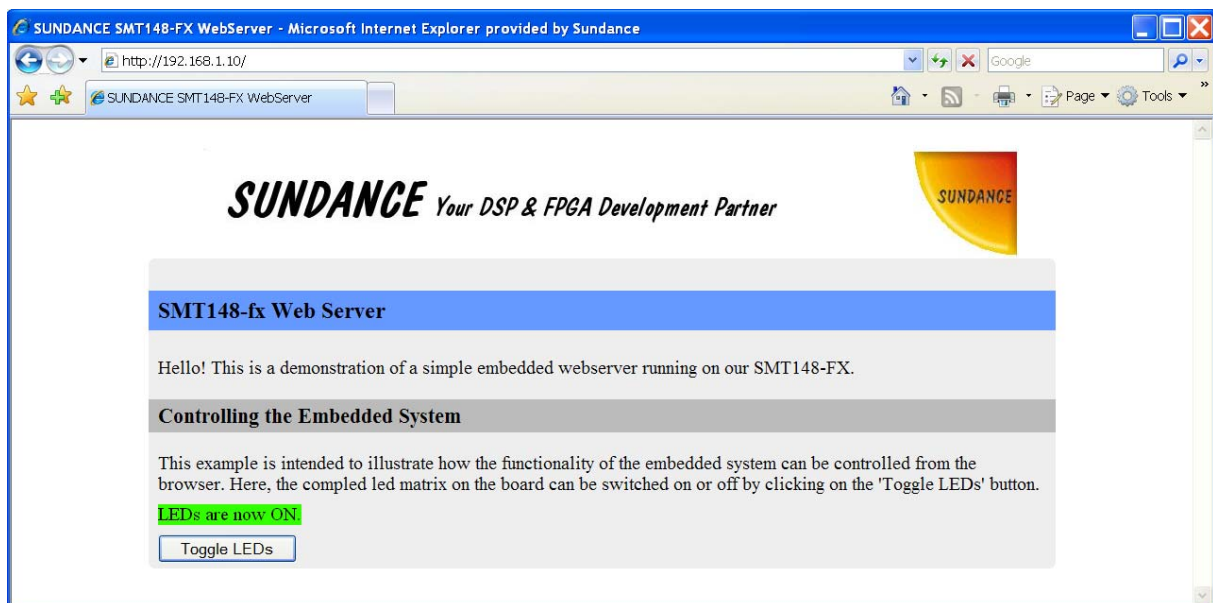
Makefile 43
css:
main.css 762
images:
sundance.JPG 12497
index.html 1608
js:
main.js 7284
yui:
anim.js 12588
conn.js 11639
dom.js 10862
event.js 14317
yahoo.js 5360
MFS block usage (used / free / total) = 161 / 839 / 1000
Size of memory is 532000 bytes
Block size is 532
mfsgen done!

Fabiens@SudacaXP /cygdrive/c/SMT6058/Example/Webserver/memfs
$ =

```

**Figure 10 : image.mfs generation**

When the image.mfs is generated, just run the application  under SDK and you should see the difference with the figure 3, the Sundance image is now in the centre of the browser.

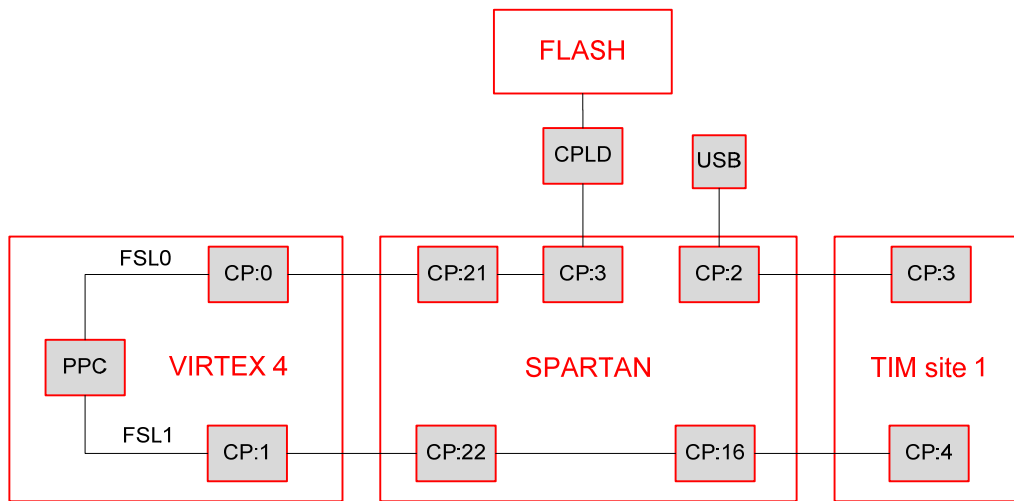


**Figure 11 : Web Server modified**

### 5.4.3 The Ethernet with SMT362 comport loopback

```
/* list of applications to be included */
#define APP_ETH2CP
```

This project provides two comport connections between the PowerPC and the Spartan.



**Figure 12 : Virtex 4 comport connections**

The comport CP0 is connected to have the flash access from the Virtex4 and the CP1 is connected to the T1CP4.

A host application is provide to test the TIM link, first you need to run the 3L SMT362 loopback application with the TIM on site 1.

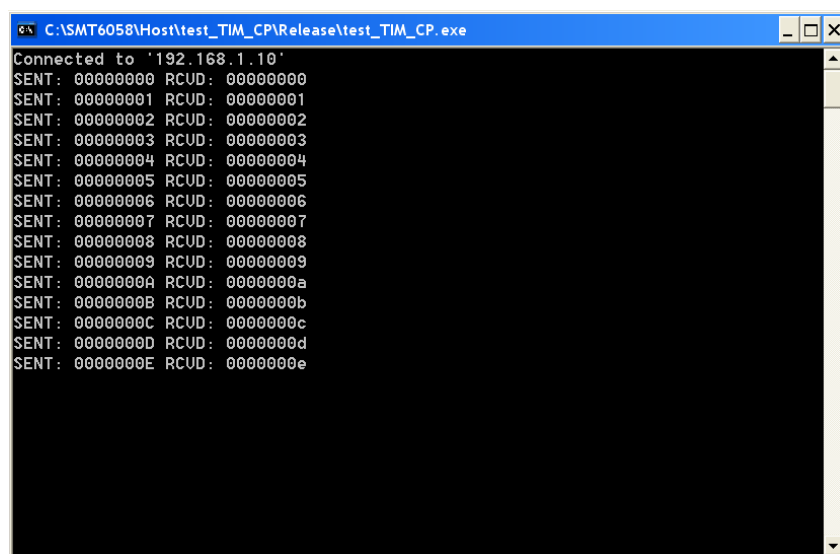
“\SMT6058\Hardware\DSP\SMT362\CpLoopback\output\CpLoopback.app”

If the Diamond server has reset the SMT148FX, wait until the end of the BootLoader and application loading, if you are loading the application from flash.

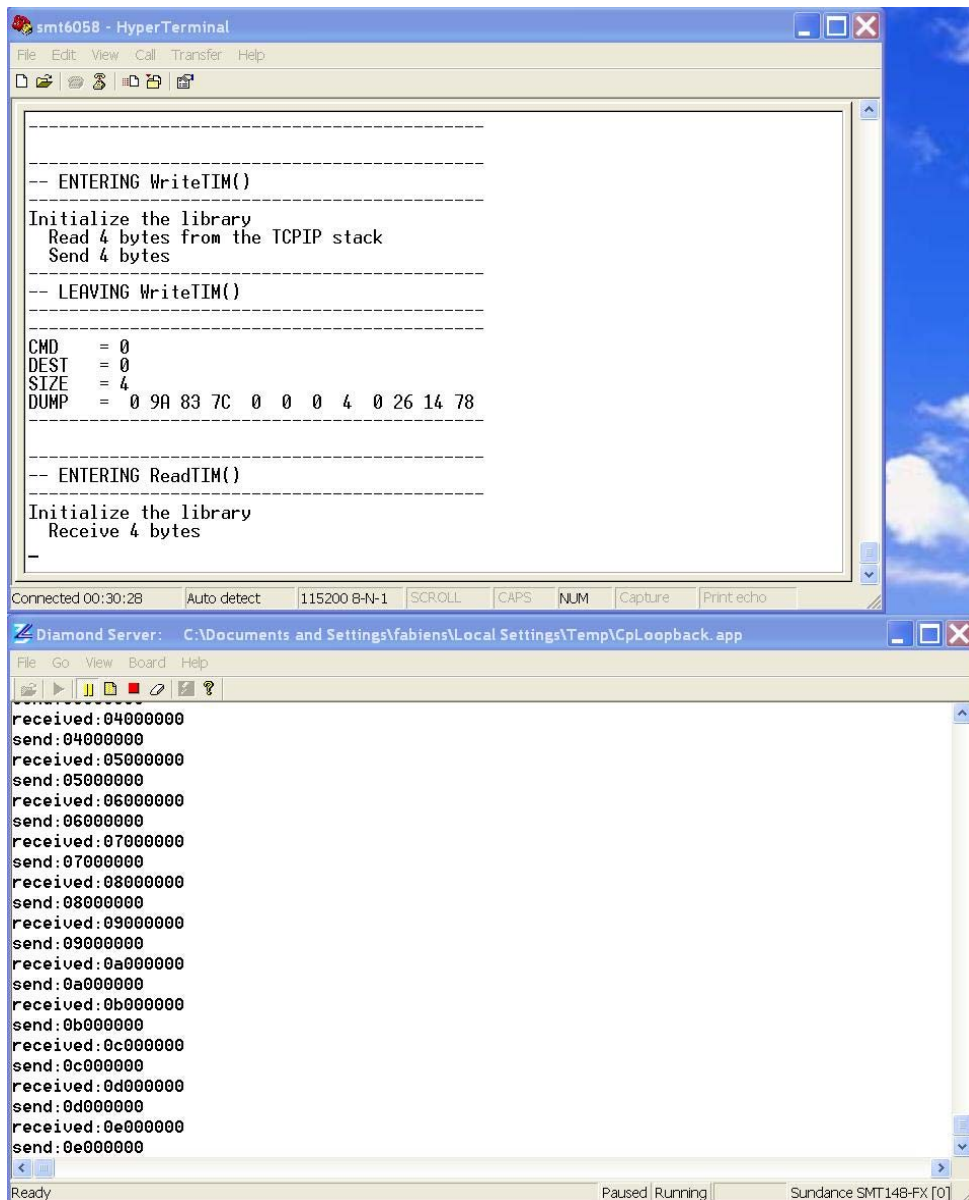
Then execute the host application:

“..\SMT6058\Host\test\_TIM\_CP\Release\test\_TIM\_CP.exe”

You should get the following result.



**Figure 13 : Host application**



**Figure 14 : Debug and 3L server result**

To restart the PPC software you need first to reset the system, due to the direction of the comports being changed.

The PPC FSL interfaces to a comport peripheral that always starts as a transmitter at reset.

If the comport is in receiver mode when you re-launch the software, the system will hang. One way of doing it is to use the Diamond server and start the Diamond app again, as a reset is issued by default by the server before loading the app.

## 5.4.4 TFTP

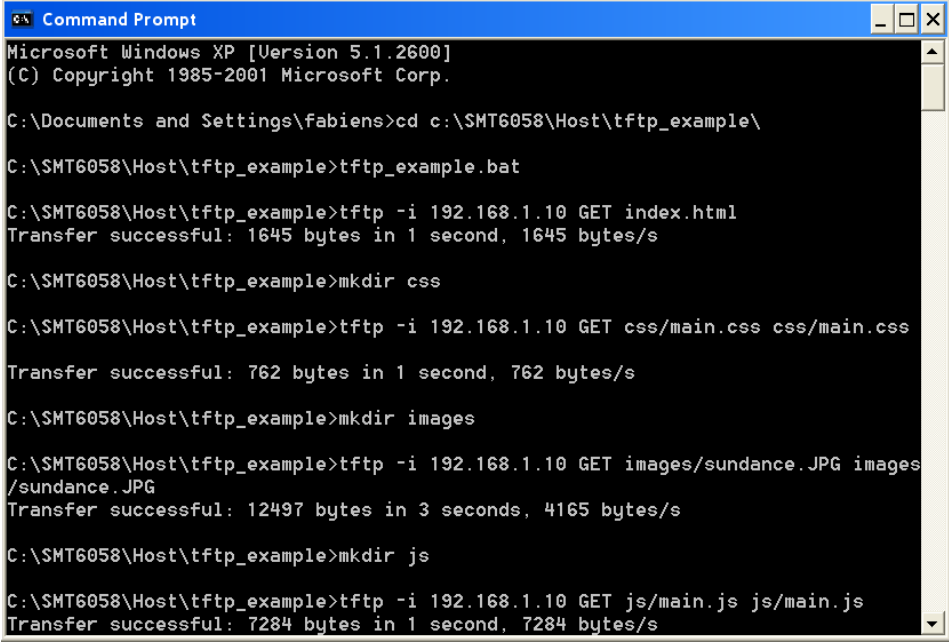
```
/* list of applications to be included */  
#define APP_TFTP_SERVER
```

Trivial File Transfer Protocol (TFTP) is a file transfer protocol, with the functionality of a very basic form of File Transfer Protocol (FTP).

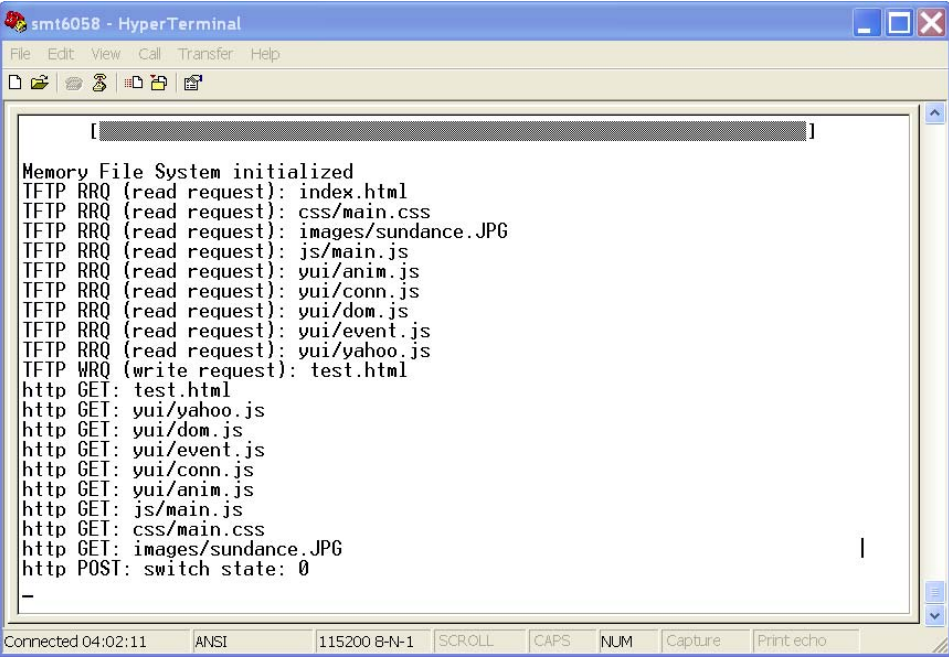
Open a Command Prompt window and type tftp to get some help.

Run “..\SMT6058\Host\tftp\_example\tftp\_example.bat” as example, it will send the tftp request to download all the html page and after upload the page test.html.

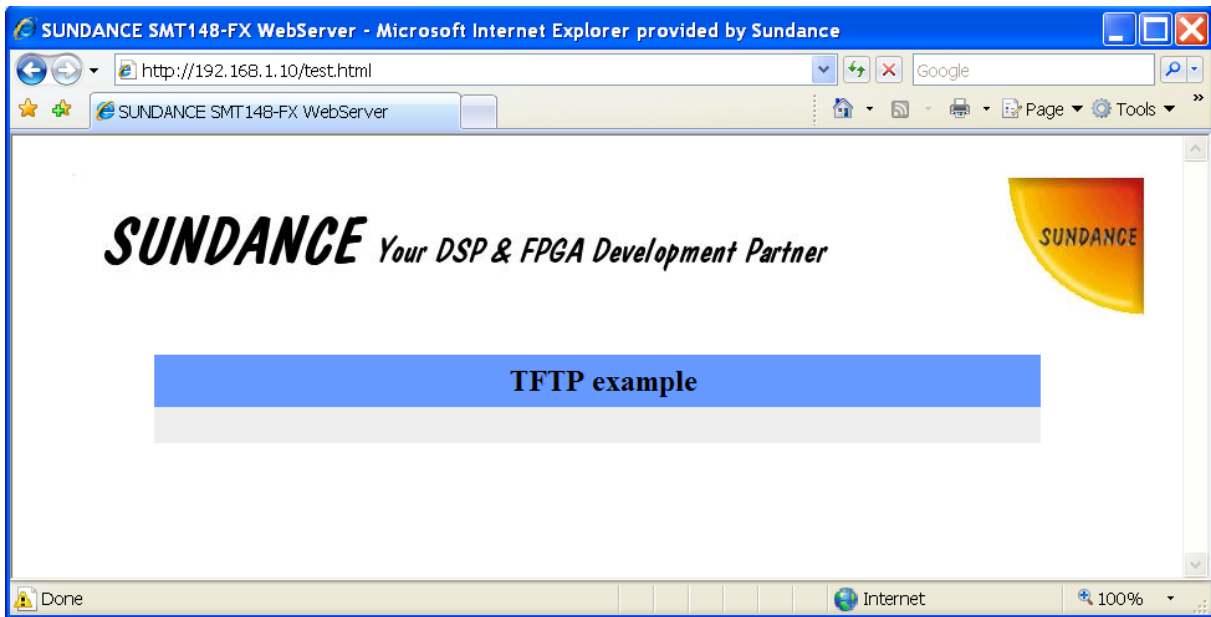
Open <http://192.168.1.10/test.html> to see the page that you just upload in the SMT148FX memory.



```
Command Prompt  
Microsoft Windows XP [Version 5.1.2600]  
(C) Copyright 1985-2001 Microsoft Corp.  
  
C:\Documents and Settings\fabiens>cd c:\SMT6058\Host\tftp_example\  
  
C:\SMT6058\Host\tftp_example>tftp_example.bat  
  
C:\SMT6058\Host\tftp_example>tftp -i 192.168.1.10 GET index.html  
Transfer successful: 1645 bytes in 1 second, 1645 bytes/s  
  
C:\SMT6058\Host\tftp_example>mkdir css  
  
C:\SMT6058\Host\tftp_example>tftp -i 192.168.1.10 GET css/main.css css/main.css  
Transfer successful: 762 bytes in 1 second, 762 bytes/s  
  
C:\SMT6058\Host\tftp_example>mkdir images  
  
C:\SMT6058\Host\tftp_example>tftp -i 192.168.1.10 GET images/sundance.JPG images/  
/sundance.JPG  
Transfer successful: 12497 bytes in 3 seconds, 4165 bytes/s  
  
C:\SMT6058\Host\tftp_example>mkdir js  
  
C:\SMT6058\Host\tftp_example>tftp -i 192.168.1.10 GET js/main.js js/main.js  
Transfer successful: 7284 bytes in 1 second, 7284 bytes/s
```



```
smt6058 - HyperTerminal  
File Edit View Call Transfer Help  
Memory File System initialized  
TFTP RRQ (read request): index.html  
TFTP RRQ (read request): css/main.css  
TFTP RRQ (read request): images/sundance.JPG  
TFTP RRQ (read request): js/main.js  
TFTP RRQ (read request): yui/anim.js  
TFTP RRQ (read request): yui/conn.js  
TFTP RRQ (read request): yui/dom.js  
TFTP RRQ (read request): yui/event.js  
TFTP RRQ (read request): yui/yahoo.js  
TFTP WRQ (write request): test.html  
http GET: test.html  
http GET: yui/yahoo.js  
http GET: yui/dom.js  
http GET: yui/event.js  
http GET: yui/conn.js  
http GET: yui/anim.js  
http GET: js/main.js  
http GET: css/main.css  
http GET: images/sundance.JPG  
http POST: switch state: 0  
-  
Connected 04:02:11 ANSI 115200 8-N-1 SCROLL CAPS NUM Capture Print echo
```



**Figure 15 : TFTP example**

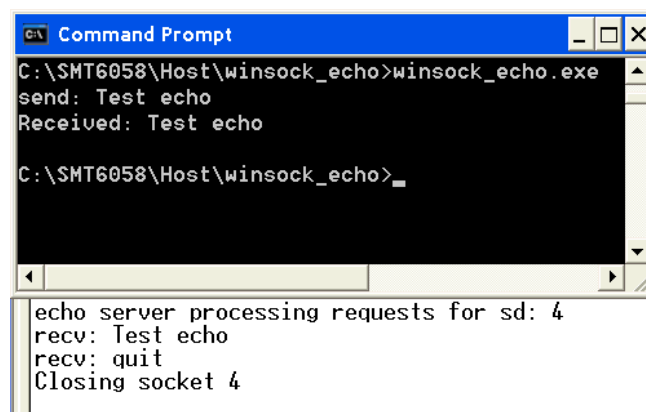
#### 5.4.5 ECHO

```
/* list of applications to be included */
#define APP_ECHOSERVER
```

The “..\SMT6058\Host\test\_TIM\_CP\Include\smt6058.h” file gives you all the functions needed to use the entire project and design your host application.

A simple winsock echo example is available to show how to use winsock if you want to make your own host function.

Run “..\SMT6058\Host\winsock\_echo\winsock\_echo.exe” and you should get this result.



**Figure 16 : Echo example**