LAB 1: Introduction to the ZYNQ ZC702 “Hello World”

Introduction

This lab introduces the Xilinx ZC702 development board and the Xilinx software tools used to program the board. The board features a XC7Z020 integrated circuit which contains a dual-core ARM Cortex A9 processor (referred to as the PS) together with a field programmable gate array (referred to as the PL) which can be programmed to implement digital logic in hardware

Xilinx provides a diverse set of software tools which are used to develop applications for the ZC702. The primary tools used in the labs are:

• the PlanAhead tools is used to add design sources to the hardware. A design source may be in the form of a user-provided VHDL program to program the on-board FPGA, or it may consist of some custom Intellectual Property (IP) such as a floating point multiplier (created with the Xilinx Coregen tool). Another example of a design source would be a Xilinx User Constraint File (UCF).

• Xilinx Platform Studio (XPS) is used to customize the processor hardware and programmable logic. XPS configures the hardware and exports it to the Software Development Kit (SDK).

• SDK is used to program the processor. It consists of a C/C++ GNU compiler/debugger for the ARM Cortex A9 processor. SDK is based upon the Eclipse framework. It is within the SDK environment that an application program is written, compiled, downloaded and executed on the processor on the ZC702 development board. SDK allows for a terminal setup between the host computer and the ZC702 board so that allow the program to perform console (i.e. keyboard and monitor) input/output. [1]

Procedure

Setting up the user Linux environment:

1. Login in to an ENCS Linux system and type the following from the Linux command prompt:

source /CMC/tools/xilinx_14.7/14.7/ISE_DS/settings64_CMC_central_license.csh

The following will be displayed in the terminal window:

XILINXD_LICENSE_FILE is set to 6062@license-cadconnect:7062@license-cadconnect
source /nfs/sw_cmc/linux-64/tools/xilinx_14.7/14.7/ISE_DS/common/.settings64.csh /nfs/sw_cmc/linux-64/tools/xilinx_14.7/14.7/ISE_DS/common
source /nfs/sw_cmc/linux-64/tools/xilinx_14.7/14.7/ISE_DS/EDK/.settings64.csh /nfs/sw_cmc/linux-64/tools/xilinx_14.7/14.7/ISE_DS/EDK
source /nfs/sw_cmc/linux-64/tools/xilinx_14.7/14.7/ISE_DS/PlanAhead/.settings64.csh /nfs/sw_cmc/linux-64/tools/xilinx_14.7/14.7/ISE_DS/PlanAhead

source /nfs/sw_cmc/linux-64/tools/xilinx_14.7/14.7/ISE_DS/ISE/.settings64.csh /nfs/sw_cmc/linux-64/tools/xilinx_14.7/14.7/ISE_DS/ISE

This step sets up your Linux environment to run the Xilinx software tools. It needs only to be performed one time (as long as you terminal window is open).

Launching PlanAhead:

2. It is suggested that you create a Linux subdirectory to hold your COEN 317 lab related files. The following Linux commands will create a directory called COEN317 from within your Linux home directory and a Lab1 directory will be created within the COEN 317 directory:

mkdir COEN317
cd COEN 317
mkdir Lab1
cd Lab1

To launch the Xilinx PlanAhead software, type the following:

ted@happy Lab1 3:02pm > planAhead &

In the above, the ted@happy Lab1 3:02pm> is an example of a Linux prompt. Commands are entered at the command prompt. Your command prompt may appear different. Ask your TA for assistance if necessary. Once you enter the above command, the following will be displayed:

****** PlanAhead v14.4 (64-bit)
**** Build 222254 by xbuild on Tue Dec 18 05:17:28 MST 2012
** Copyright 1986-1999, 2001-2012 Xilinx, Inc. All Rights Reserved.

INFO: [Common 17-78] Attempting to get a license: PlanAhead
INFO: [Common 17-290] Got license for PlanAhead
INFO: [Common 17-86] Your PlanAhead license expires in -583 day(s)
INFO: [Device 21-36] Loading parts and site information from /nfs/sw_cmc/linux-64/tools/xilinx_14.4/14.4/ISE_DS/PlanAhead/data/parts/arch.xml
Parsing RTL primitives file [/nfs/sw_cmc/linux-64/tools/xilinx_14.4/14.4/ISE_DS/PlanAhead/data/parts/xilinx/rtl/prims/rtl_prims.xml]
Finished parsing RTL primitives file [/nfs/sw_cmc/linux-64/tools/xilinx_14.4/14.4/ISE_DS/PlanAhead/data/parts/xilinx/rtl/prims/rtl_prims.xml]
start_gui

The main PlanAhead window will appear as shown in Figure 1:

Figure 1: Main PlanAhead window.

Creating a new project from PlanAhead:

3.1. From the PlanAhead window select :

File -> New Project

The New Project window will appear, select Next to continue.

3.2. Change the default project name to lab1. The project location will be the Linux path of the directory from where you invoked planAhead from. Select Next to continue. Refer to Figure 2 for details.

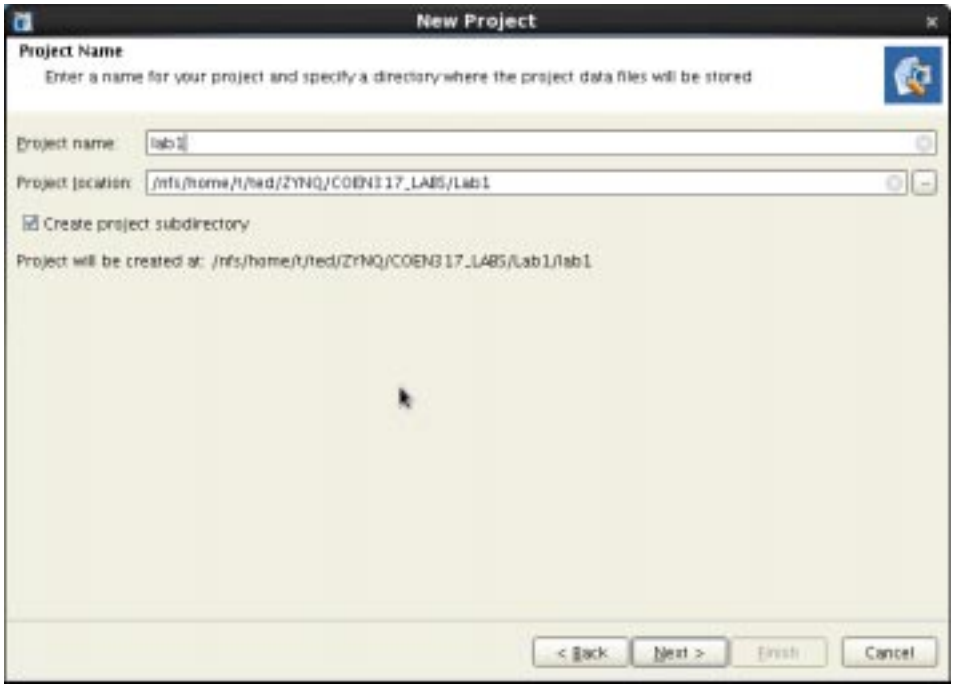

Figure 2: Specifying a project name and location.

3.3 In the Project type window, RTL Project should be already selected. Keep it as the selected project type and select Next to continue. Refer to Figure 3:

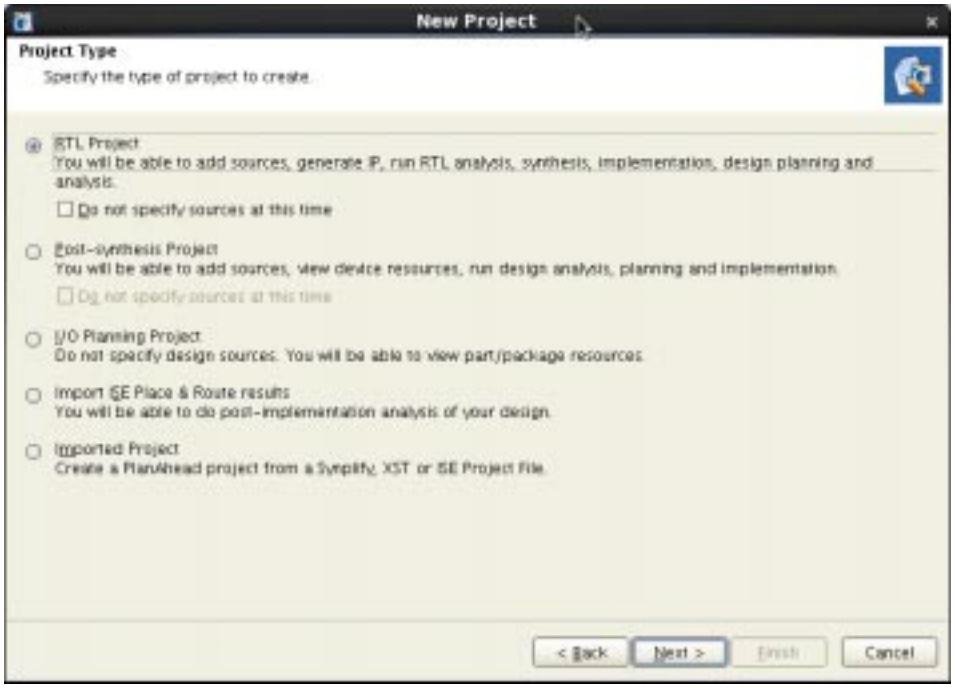

Figure 3: Specifying the Project Type.

3.4 Although no sources need to be added to this project (as it is purely software based), change the target language from Verilog to VHDL in the drop down box as shown in Figure 4. Future labs will involve using VHDL as the language used to program the FPGA and it is at this point in the flow that the hardware description language (HDL) needs to be specified. Select Next to continue.

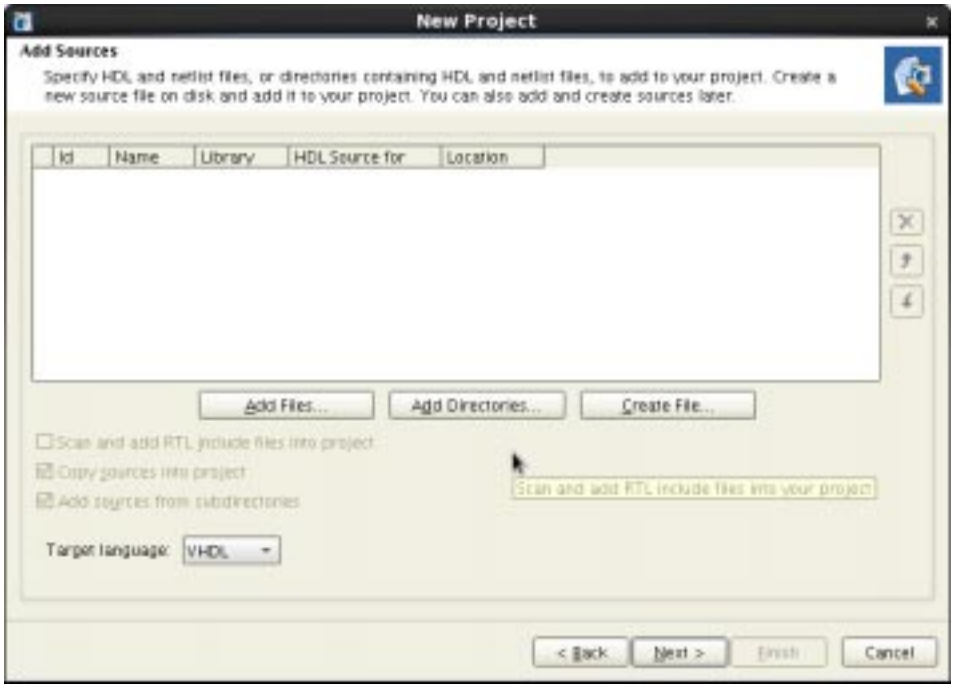

Figure 4: Add Sources.

3.5 This lab does not involve any IPs, select Next in the Add Existing IP to continue.

3.6 We do not need to specify any UCF file in this lab, select Next to continue to the Choose Default part screen.

3.7 Select Boards and scroll down to the list to select the ZYNQ-7 ZC702 Evaluation board as shown in Figure 5.

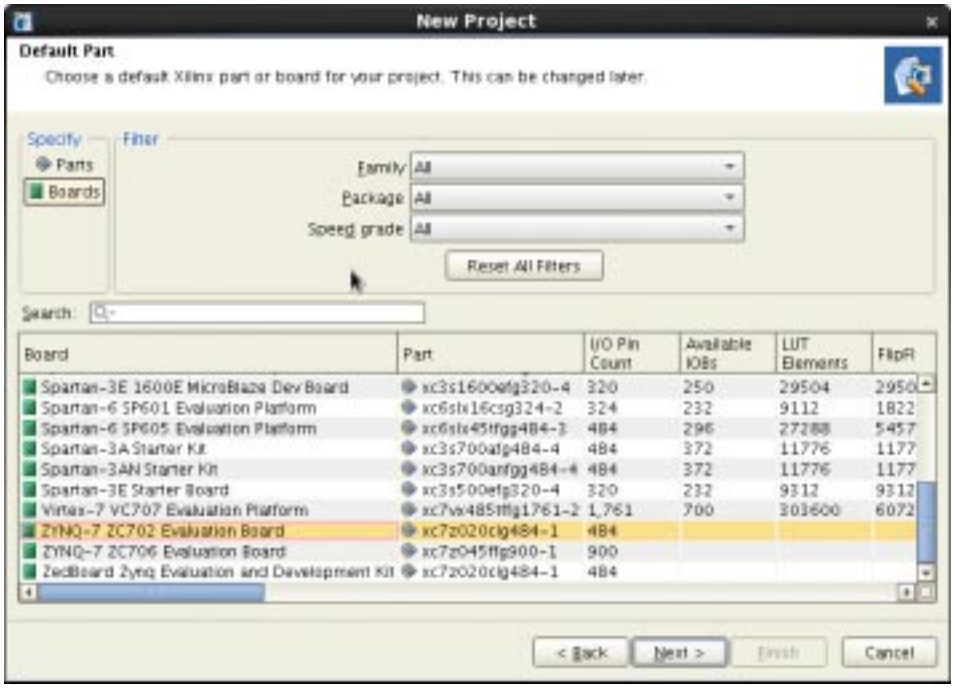

Figure 5: Choose Board.

3.8 In the New Project Summary, review the settings and select Finish to close the New Project wizard. The newly created project will open in the PlanAhead design tool.

Create an embedded processor project with the Add Sources wizard:

The above steps created a new project and opens the project from within the PlanAhead tool as shown in Figure 6.

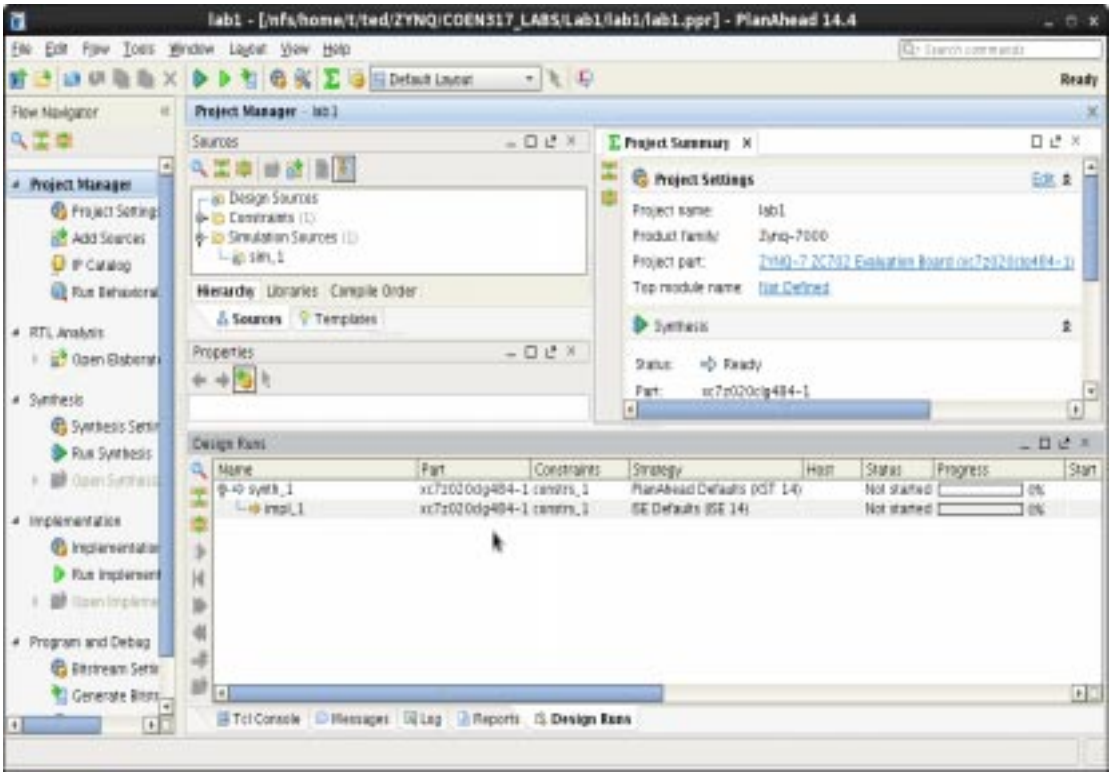

Figure 6: Newly opened project in PlanAhead.

4.1 From the left panel under the Project Manager tab, select Add Sources. The Add Sources wizard opens.

4.2 Select the Add or Create Embedded Sources and select next. Refer to Figure 7.

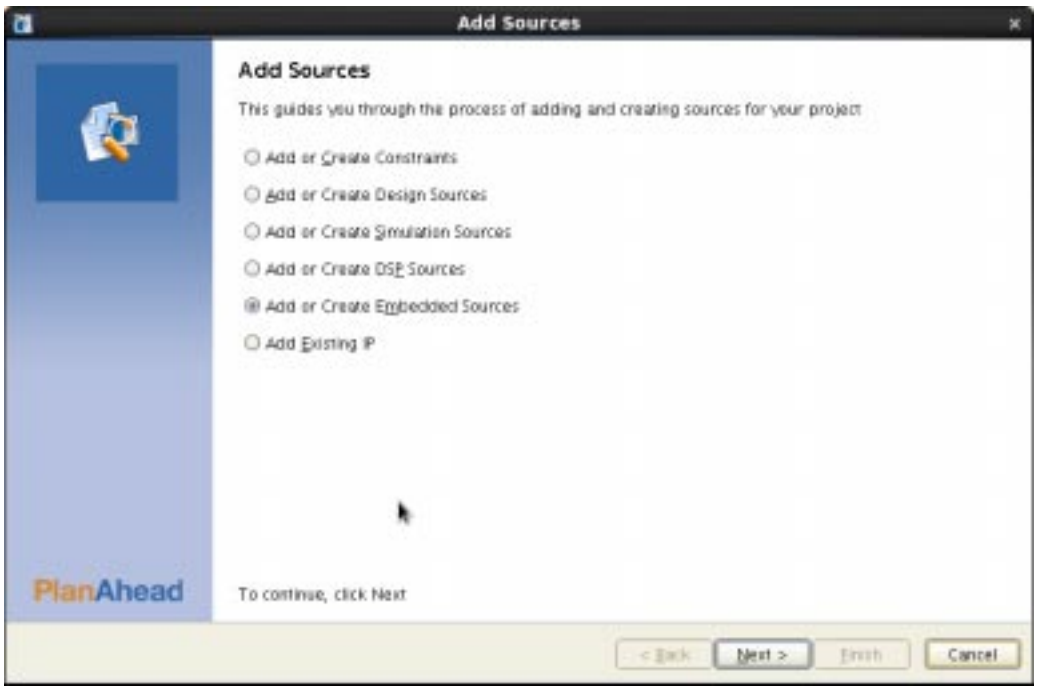

Figure 7: Add Sources.

4.3 In the Add or Create Embedded Sources, select Create Sub-design. A window will appear prompting you for a module name (Figure 8). Name the module "system" (any name may be chosen) and select Finish

Figure 8: Specifying the module name.

PlanAhead will create the embedded design source project and open up Xilinx Platform Studio (XPS) which is used to design the embedded system.

Designing the system in XPS:

5.1 A Platform studio window will appear asking:

“This project appears to be a blank zynq project. Do you want to create a Base System using the BSB Wizard?

Select Yes.

5.2 The Base System Builder (BSB) Wizard will appear with AXI System as the selected Inter-connect type. Select OK to continue. AXI is a bus protocol adopted by Xilinx as the interface between the ARM processor and the programmable logic.

5.2 In the Base System Builder - AXI Flow window, ensure that

Board Vendor = Xilinx
Board Name = Zynq ZC702 Evaluation Platform
Seelct A System = Zynq Processing System 7.

Select Next to continue.

Refer to Figure 9.

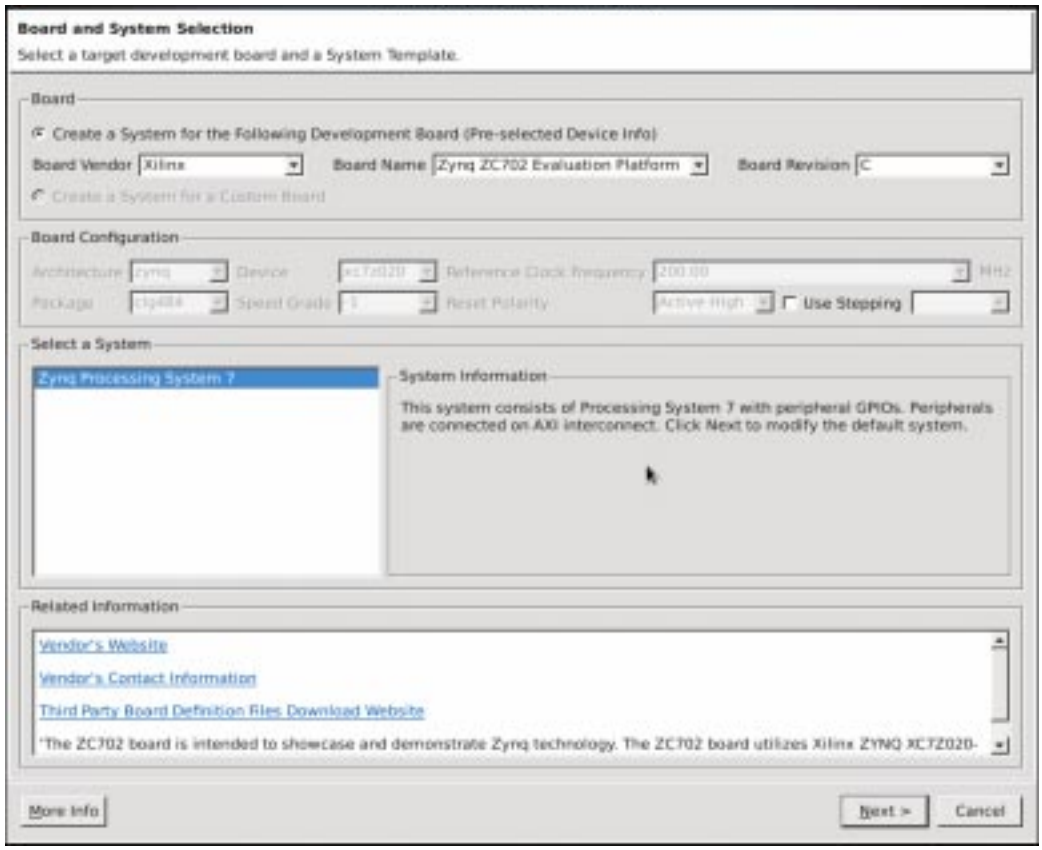

Figure 9: Board and System Selection.

In the Peripheral Configuration window, remove the GPIO_SW and LEDS_4Bits peripherals by first choosing Select All , followed by Remove

To build the design, select Finish in the Peripheral Configuration window.

5.3 Close the XPS tool by selecting File -> Exit. Closing XPS will update the active PlanAhead session with the new project settings.

Note: It is at this point in the flow that any peripherals which are required in the hardware (such as a General purpose IO , timer, etc) would be added within XPS. Since this lab involves only software running on the PS, no peripherals are required.

Exporting the Hardware to SDK:

6.1 In the PlanAhead window (Figure 10) , within the top middle Project Manager pane, right click the system(system.xmp) and select Create Top HDL.

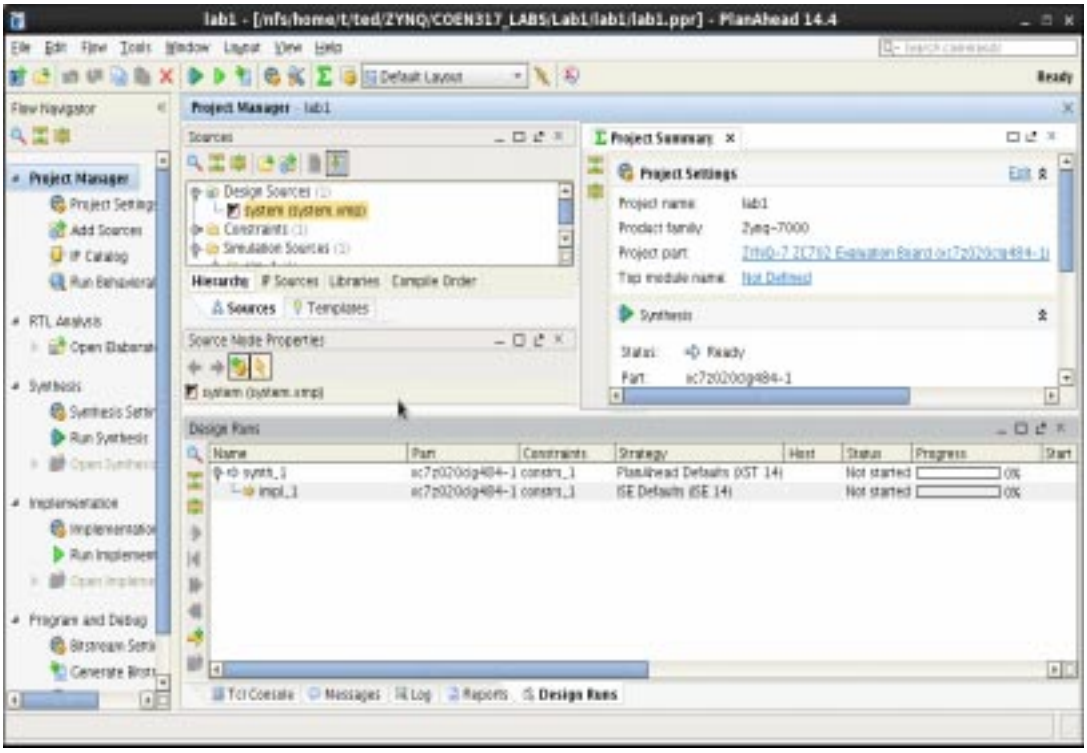

Figure 10. Creating the Top HDL.

The PlanAhead tool will create the system_stub.vhd file which is the top-level module for the system. For this lab, this file does not require any modifications (future labs will involve adding user-specified logic to this top-level module).

6.2 Synthesize the design by selecting Run Synthesis from the Synthesis tab in the Project Manager of PlanAhead. Wait until Synthesis is complete.

6.3 When the Synthesis Completed window appears, implement the design by selecting OK (make sure that Run Implementation has been selected). There may be 3 warnings concerning constraint locations of PS_PORB_IBUF which may be ignored as they do not affect the system. Click OK to close the warnings.

6.4 Upon completion of implementation, a window will appear prompting Generate Bitstream. Select OK.

6.5 Make sure that the power cable, the Platform Cable USB II and USB cable are attached to the board. Ask your TA to verify that the cables are properly connected. Turn on the power switch. Ensure that the status LED on the PLatform Cable USB II is green once power has been applied to the board (without power, it should be amber).

Choose Launch Impact and select OK. A window may appear prompting you to save the file. Select Cancel to close this window.

In the Impact window (Figure 11) right click the Target (xc7z020) and select Program to program the device with he system_stub.bit file.

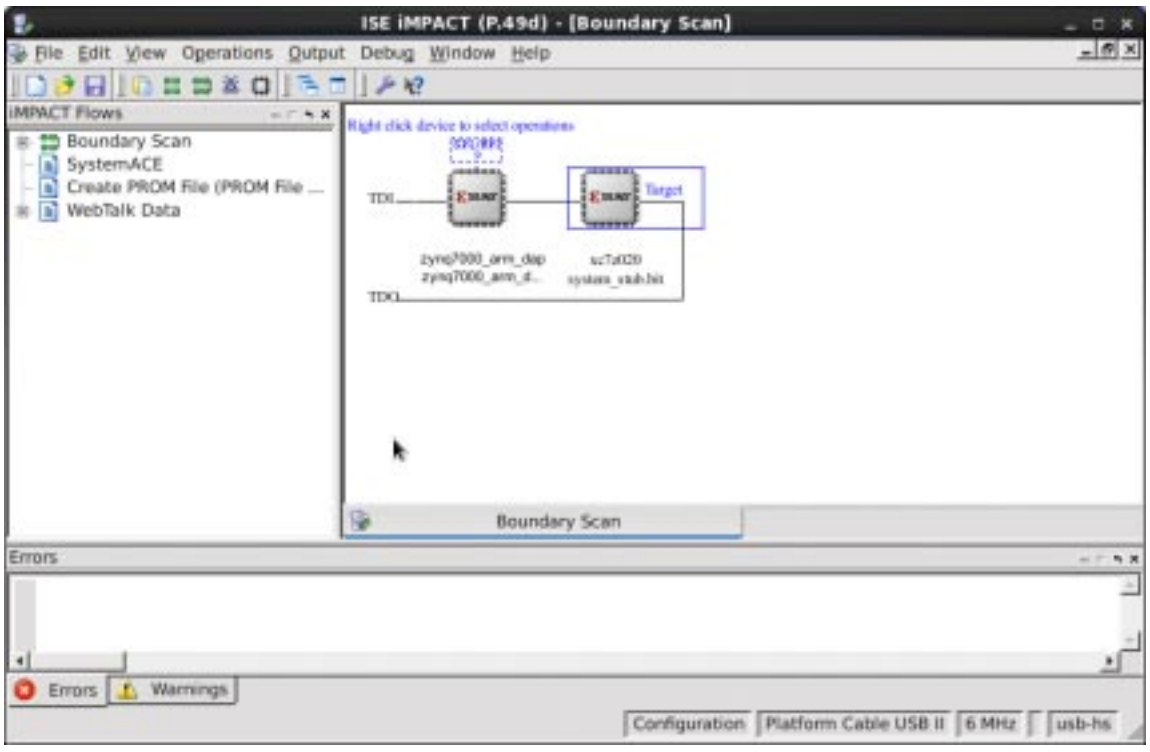

Figure 11: Impact window.

6.6 Wait until the message Program Succeeded is displayed in the Impact window and then select File -Exit You do not need the save the Impact project.

6.7 From the PlanAhead window, select File - > Export -> EXport Hardware for SDK. By default, the Export Hardware check box is enabled. Enable the Launch SDK check box and select OK as shown in Figure 12.

Figure 12. Exporting Hardware and Launching SDK.

SDK will open as shown in Figure 13.

Note: SDK may crash on the first the project is created. Check your terminal, you may see

```
#
# A fatal error has been detected by the Java Runtime Environment:
#
#  SIGSEGV (0xb) at pc=0x0000000000000000, pid=7396, tid=47440467388736)
```

If SDK crashed, repeat this step and overwrite the project.

Using SDK to create an application project:

We will create a new application project, compile it, and download and run it on the ZC702 board using SDK. SDK is based upon the Eclipse open-source program development environment.

7.1 From the main SDK window (Figure 13), select File -> New -> Application Project.

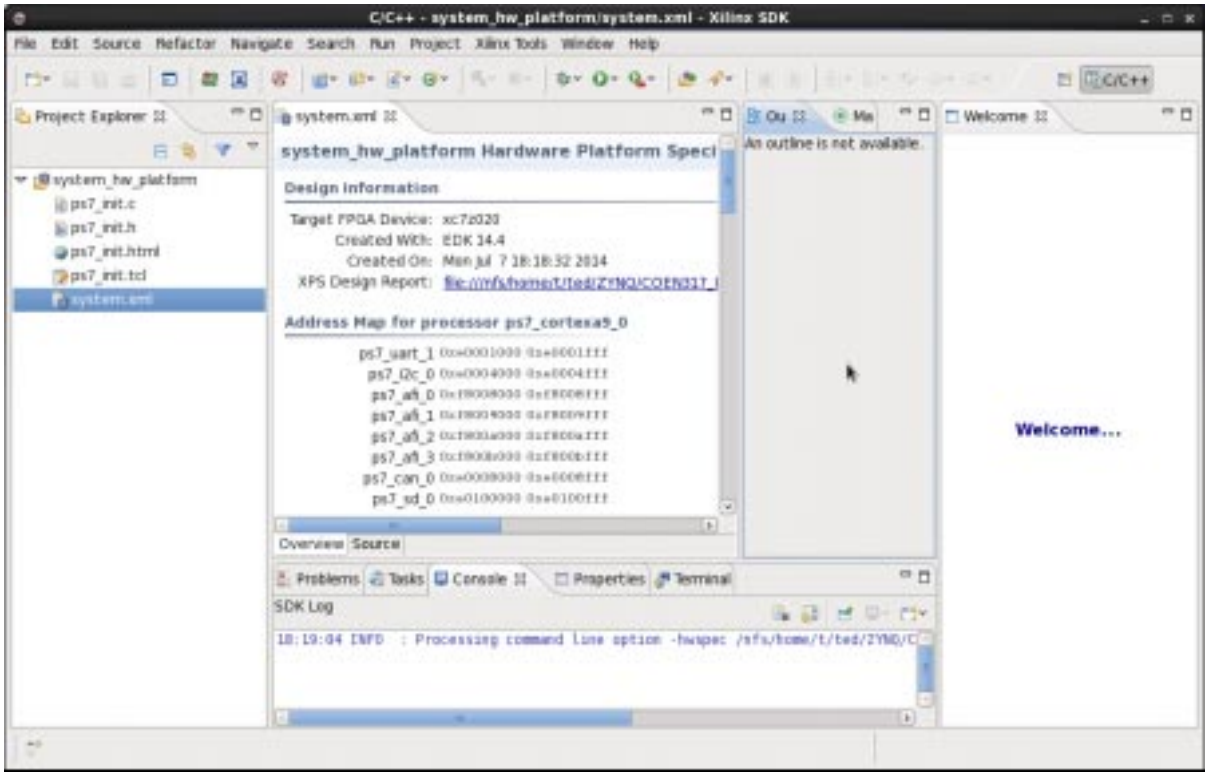

Figure 13: Main SDK window.

7.2 In the New Project window, specify a project name (HelloWorld). You can choose either C or C++ as the programming language. For the OS platform, select standalone. Refer to Figure 14. Select Finish to continue. The New Project Template window will appear listing the Empty Application as the available templates. SDK will create an empty C++ program called main.cc which you may then modify. Select Finish to complete the project.

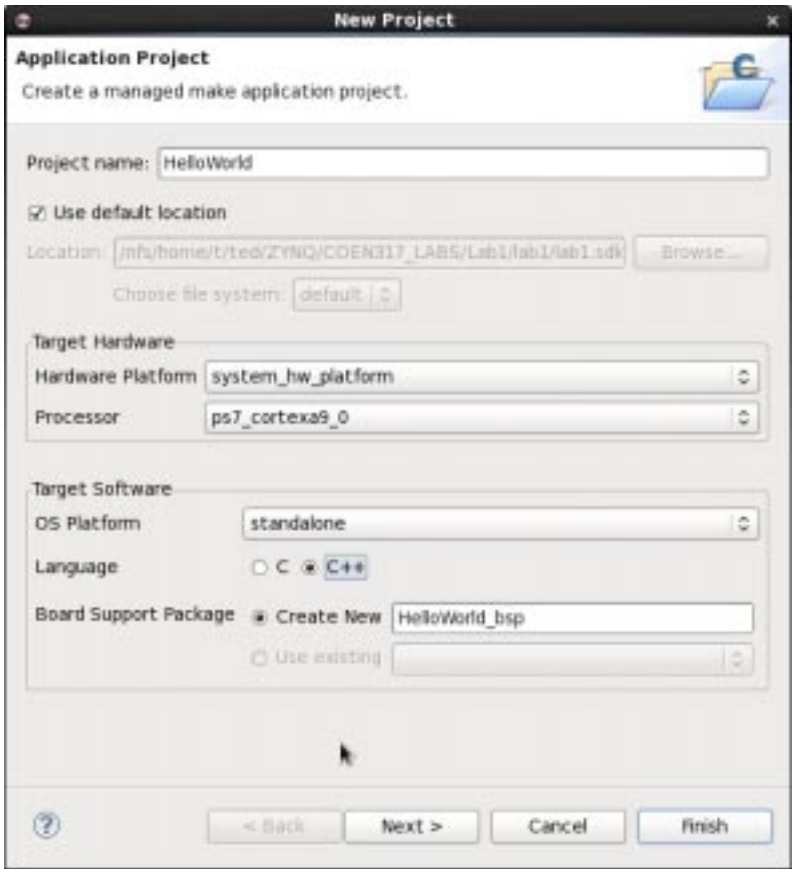

Figure 14: Specifying a new project.

7.3 To view the C++ source code (main.cc), choose the HelloWorld project in the top left hand pane of SDK and expand the source tab and select main.cc. An empty C++ program will appear in the middle pane of SDK as shown in Figure 15.

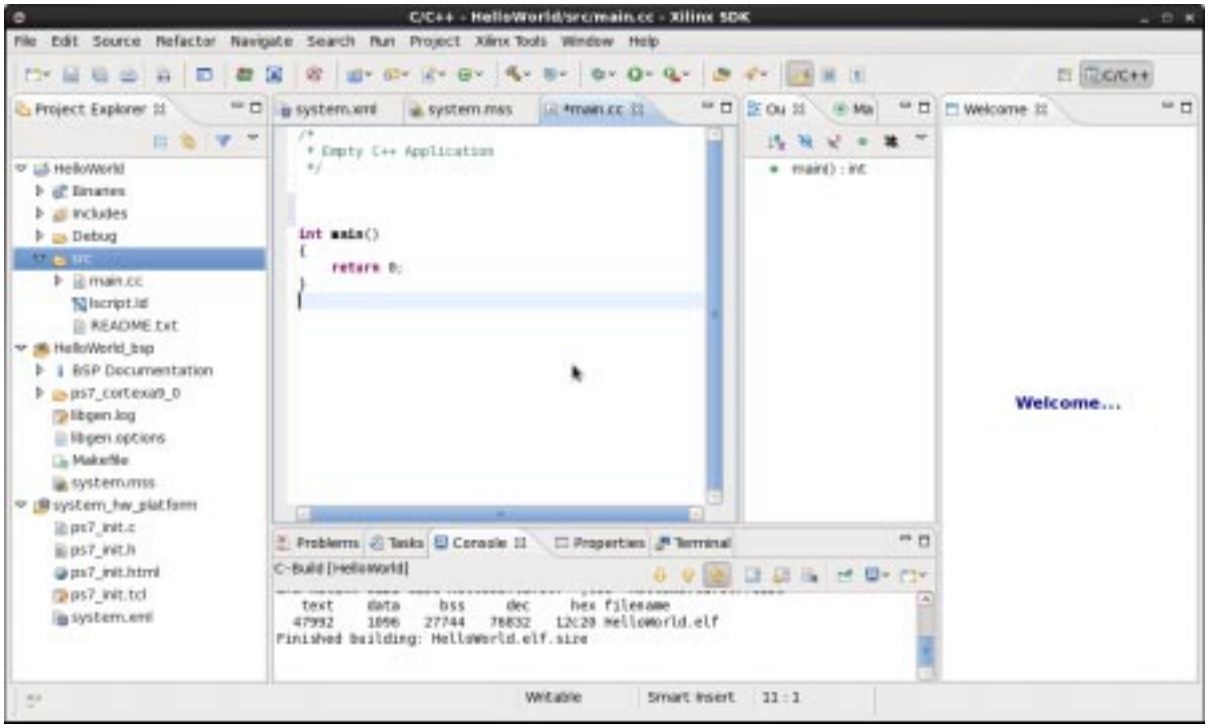

Figure 15: SDK window with main.cc in editor pane.

7.4 Add the necessary lines to the program so that it becomes a simple “Hello World” program:

```
#include <iostream>
using namespace std;

int main()
{
cout << "Hello World from Zynq PS" << endl;
return 0;
}
```

To save the changes you made to the source code, select File -> Save (or CTRL-S)

For those of you who prefer to use a Linux command line editor, the main.cc file is located in:

./lab1/lab1.sdk/SDK/SDK_Export/HelloWorld/src/main.cc

If the file is modified, saving it will automatically recompile it. Any warnings/errors will be shown the Console window located in the bottom portion of the SDK window.

7.5 Before downloading and running the program, establish a serial console window between the host PC and the ZC702 board by selecting Window -> Show View -> Terminal. The bottom part of the SDK window is now the Terminal window. One first has to establish the serial communication parameters by selecting the green Connect window (which looks like the letter N with two dots at either end). Refer to Figure 16 for the values of the settings.

Figure 16:Serial terminal settings.

If the terminal window is properly connected, the following will appear in the bottom portion of the SDK window:

Serial: (/dev/ttyUSB0, 115200, 8,1,None,None -CONNECTED)

7.6 To download and run the executable file (HelloWorld.elf) , right click the Hello World folder tab in the left hand part of SDK and select Run As -> Run Configuration.

7.7 In the Run Configuration pop-up window (Figure 17) , right click Xilinx C/C++ ELF and choose New (this needs only to be perfromed once, if you make changes to the program source code and recompile, it is not necessary to repeat this step).

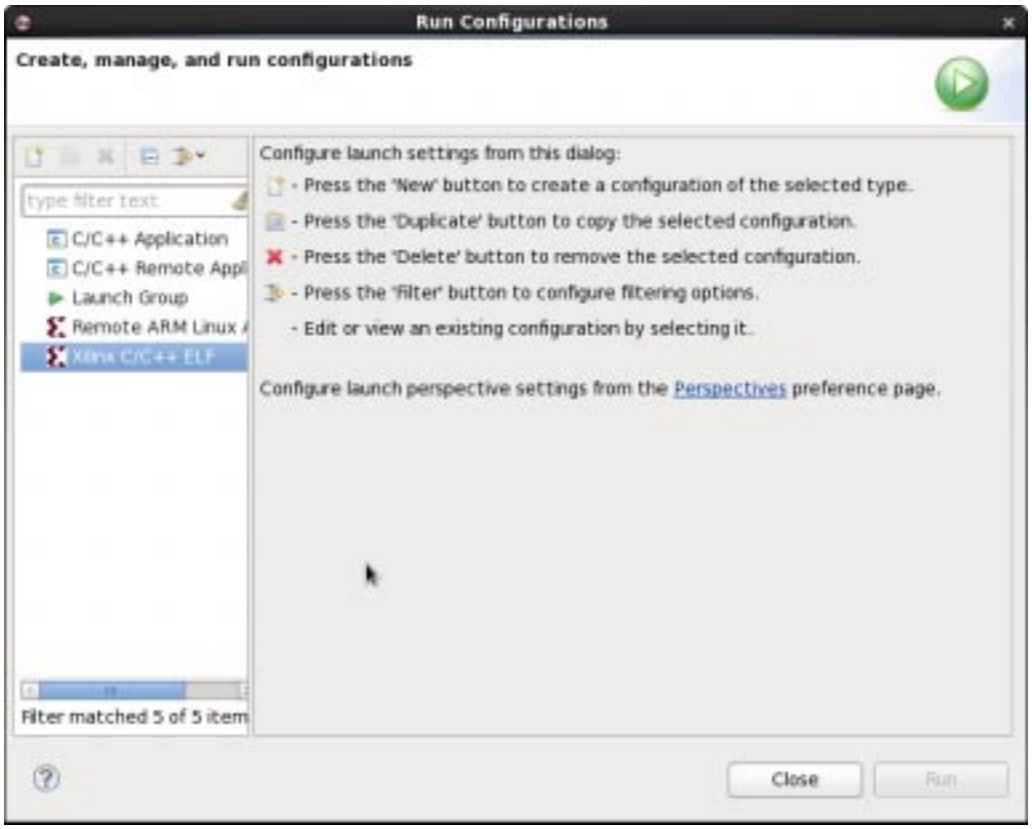

Figure 17: New Run Configuration.

7.8 The Run Configurations window will appear as shown in Figure 18. Leave the default settings and select Run to download and run the program. A scroll bar showing the progress of the download will be shown in the bottom right hand portion of the SDK window. Once the program has been downloaded to the board, return to the SDK terminal windo to see the output generated. as shown in Figure 19.

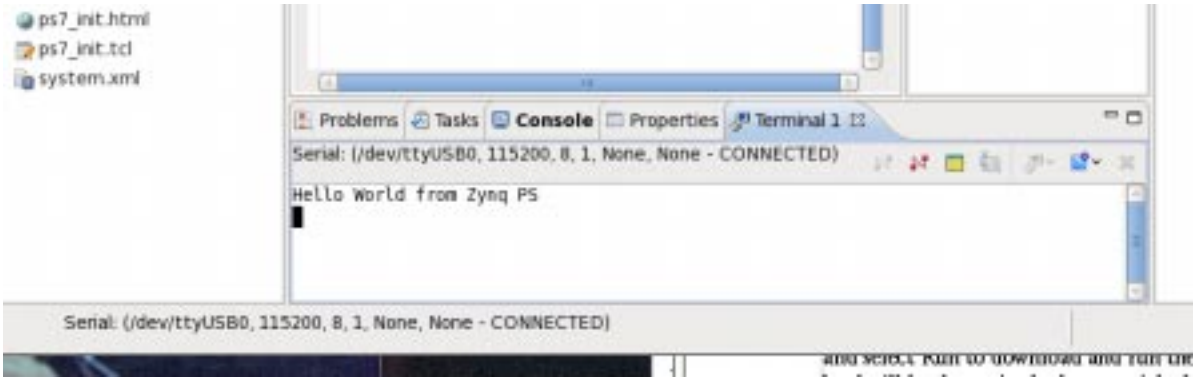

Figure 19. Terminal window with program output.

7.9 To exit from SDK, select File -> Close, followed by File -> Exit.

7,10. To exit from PlanAhead, select File -> Close Project, followed by File -> Exit from the main PlanAhead window.

References

1. Zynq-7000 All Programmable SoC: Concepts, Tools, and Techniques (CTT) A Hands-On Guide to Effective Embedded System Design, UG873 (v14.4), Xilinx Inc., Dec. 18, 2012, p9.

T. Obuchowicz, R. Lee, July 2014.

Revision history:

- Sept. 9, 2016: modified for sourcing /CMC/tools/xilinx_14.7/14.7/ISE_DS/settings64_CMC_central_license.csh