Lab Assignment 4
DE1-SoC Linux Getting Started

Introduction

This lab is an introduction to the DE1-SoC, an embedded computing device built around the Altera System-on-Chip (SoC) FPGA and an ARM processor running a Linux operating system. We will be using this platform throughout this course to get a hands-on experience working with an embedded system. The platform is a very rich environment with abundant hardware features, though we will primarily focus on the ARM processor (CPU) and the Cyclone V FPGA, both of which are part of the Altera's System-on-Chip (SoC), the central chip on the DE1-SoC. The DE1-SoC also contains a wide set of input/output (I/O) devices, among which we will use LED lights, switches, buttons, and 7-segment displays.

The DE1-SoC is running a version of the Ubuntu distribution of the Linux operating system, called *DE1-SoC-UP Linux*. This is a slightly scaled-down version of Ubuntu but has a lot of the functionality of a full-fledged distribution of this operating system. The operating system runs on the ARM processor and supports the execution of custom C or C++ programs.

Pre-Lab Assignment:

Fully read and understand the following documents and websites:

1. How To SSH - Windows - DE1-SoC.pdf
2. https://www.electronics-tutorials.ws/binary/bin_3.html
3. https://www.omnisecu.com/tcpip/binary-decimal-hexadecimal-numbers-and-conversions.php
4. https://www.tutorialspoint.com/cplusplus/cpp_if_else_statement.htm
5. https://www.tutorialspoint.com/cplusplus/cpp_while_loop.htm
6. https://www.tutorialspoint.com/Relational-and-comparison-operators-in-Cplusplus
7. Nano Tutorial: https://www.howtogeek.com/howto/42980/the-beginners-guide-to-nano-the-linux-command-line-text-editor/
   [Required for those who do not know how to use VI or VIM]
8. M. Stonebank: "UNIX Tutorial for Beginners"
   Read the Introduction, Tutorial One and Tutorial Two
   http://www.ee.surrey.ac.uk/Teaching/Unix/
   [Required for those who have not used Linux]

Nothing to turn in this time.

Group:

This assignment is intended to be completed individually but you are allowed to form a group of two to complete this assignment.

Lab Assignments

Lab 4.0 Connecting to the DE1-SoC

Please follow the guidelines posted on Canvas to connect your PC/Laptop to your DE1-SoC.

Lab 4.1 Hello-world on the DE1-SoC

As a first lab assignment, we will run a simple *hello world* program on the ARM processor of the DE1-SoC board.

1. Create a directory named lab4 in your home folder (**mkdir lab4**). In the future, make sure that you have a separate directory for each lab session, where you will place the corresponding files.
2. With lab4 as your current working directory, use the **nano** editor to create a C++ source file named hello.cpp (**nano hello.cpp**). Write the *Hello World* program in this file. An example of a *Hello World* program is shown below:

```
// Example program: Hello World
#include <iostream>
#include <string>

int main()
{
  std::cout << "Hello World \n";
  return 0;
}
```

3. Use *g++* to compile the program. You should generate an executable file named runhello, also located in directory lab4. (**g++ -o runhello hello.cpp**)
4. Execute the program and verify that the output is as expected. (**./runhello**)
5. Please make sure to save your C++ files in safe place. (Jump drive, Google drive, Dropbox, ….)

Observation 1

a) Replace the term ***std::cout*** from the program above with just ***cout*** from the program above and try to recompile the file hello.cpp again. Does the program run? Why/Why not?

Lab 4.2 Hello-world on the DE1-SoC part 2

As a second lab assignment, we will run a simple *hello world part2* program on the ARM processor of the DE1-SoC board.

1. Use the nano editor to create a C++ source file named hello2.cpp. Write the *Hello World* part 2 program in this file like the one shown below:

```
// Example program: Hello World part 2
#include <iostream>
#include <string>

using namespace std;

int main()
{
  cout << "Hello World \n";
  return 0;
}
```

2. Use *g++* to compile the program. You should generate an executable file named runhello2, also located in directory lab4. (**g++ -o runhello2 hello2.cpp**)
3. Execute the program and verify that the output is as expected. (**./runhello2**)
4. Please make sure to save your C++ files in safe place. (Jump drive, Google drive, Dropbox, ….)

Observation 2

a) Delete the line ***using namespace std;*** from the program above and try to recompile the file hello2.cpp again. Does the program run? Why/Why not?

Lab 4.3 Binary to Decimal and Hexadecimal number

One quality that all engineers need to have is the ability to dissect a complex problem into multiple smaller pieces. In this exercise we want to write a C++ program that will convert an **unsigned** binary number to Decimal and Hexadecimal.

Follow each step carefully:

1. Use the nano editor to create a C++ source file named lab43.cpp, the template is shown below:

```
#include <iostream>
#include <string>
//You are not allowed to use any other libraries including math

using namespace std;
int main()
{
  //Write Your Code here
}
```

2. Using *while loop* or *for loop*, ask a user to enter 8 integer numbers to represent an 8-bit number and save them to an array (Create an integer array with 8 elements). You will need this array for section 4.5. Make sure that you clearly indicate the position of the bits when you ask the user.

   For example:

   Please enter bit[0] =
   Please enter bit[1] =
   …
   Please enter bit[7] =

   Compile your program and make sure you have no errors.

3. Create a system using *while loop* or *for loop* to calculate the Decimal and Hexadecimal value based on the given input.

   If a user entered: **1000111** then you need to display **71** as your Decimal value and **47** as your Hex value.

   Compile your program and make sure you have no errors.

   The information needed to print a number in decimal and hexadecimal can be found here.

Assignment 3

a) Save the file lab43.cpp to your PC/Laptop once you are done.

Lab 4.4 Decimal to binary number

In this exercise we want to write a C++ program that will display a 32-bit number representation of an **unsigned** Decimal number.

Follow each step carefully:

1. Use the nano editor to create a C++ source file named lab44.cpp, the template is shown below:

```
#include <iostream>
#include <string>
//You are not allowed to use any other libraries including math

using namespace std;
int main()
{
  //Write Code here
}
```

2. Ask a user to enter an unsigned decimal number and display its value in a 32-bit number.
   Create an integer array with 32 elements to store each value of the bit. Use a *while loop* or a *for loop* to display each element of the array. You will need this array for the next page.

   For example:

   Please enter an unsigned decimal number =
   //if a user enters 42, then
   Your 32-bit number is = 00000000000000000000000000101010

3. Create a system to do so by using *while loop* or *for loop*
   Compile your program and make sure you have no errors.

Assignment 4

a) Save the file lab44.cpp to your PC/Laptop once you are done.

Lab 4.5 Address

In C++ you can display the address where all variables reside. The purpose of the assignment is for you to display the address of each element of all the arrays used in your code.

You are only allowed to use the following **libraries for Lab 4.5 and Lab 4.6**:
#include <iostream>
#include <string>

1. Use Linux to copy the content of your lab43.cpp to lab45.cpp. (Use the cp command)

2. Modify lab45.cpp by
    a. Creating a main menu to allow a user to enter
        i. "1" to *convert an unsigned binary number to Decimal and Hexadecimal (Lab 4.3),*
        ii. "2" *to display a 32-bit number representation of an unsigned Decimal number (Lab 4.4),*
        iii. "3" to exit the main menu which ends the program
    b. If a user enters "1" or "2", your program will execute its task and display the address of each element in an array. Your program will then loop back the main menu. Pressing a number 3 will terminate the program.

    For example:

    Welcome to Lab 4, my name is … and I would like you to select one of the options below:

    1. Convert an unsigned binary number to Decimal and Hexadecimal

    2. To display a 32-bit number representation of an unsigned Decimal number

    Please enter your selection:

Assignment 5

a) Save the file lab45.cpp to your PC/Laptop once you are done.

Lab 4.6 Value vs Address

We are now ready to create a function to simplify your code in Lab 4.5. Here are the steps:

1. Using Linux, copy the content of lab45.cpp into lab46.cpp. (Use the cp command)

2. Your lab4.5 program is not efficient because you have to print the value of the array and the address in option "1" and "2". To simplify this process, we will do step #3 and #4 below.

3. You can start by writing a function named PrintArray (for Lab 4.3) with the following prototype:

```
void PrintArray (int v[], int size)
{
      ...
}
```

   This function takes an array v of integers as the first argument, and its size in number of elements as the second argument. The function traverses the array with a loop and prints all elements between 0 and size – 1.

4. Write another function named PrintArrayAddress (for Lab 4.3) with the following prototype:

```
void PrintArrayAddress (int v[], int size)
{
      ...
}
```

   This function takes an array v of integers as the first argument, and its size in number of elements as the second argument. The function traverses the array with a loop and prints the address of all elements between 0 and size – 1.

5. Modify your main() function so that you call the PrintArray and PrintArrayAddress function.

Assignment 6

a) Save the file lab46.cpp to your PC/Laptop once you are done.

Delete all your files/works from the Linux system.
Use *rm*, and/or *rm –r* as needed.

Laboratory Report

You need to follow the lab report outline provided on Canvas.
Submit your lab43.cpp, lab44.cpp, lab45.cpp, and lab46.cpp separately/individually into Canvas. Your grader will run your code.

For individual submission:
Submit a document with the following naming convention: *FirstNameLastName*-Lab4.docx (for example: JohnDoe-Lab4.docx) that contains the code found in lab43.cpp, lab44.cpp, lab45.cpp, and lab46.cpp. Turnitin will check the genuineness of your work.

For a group of two submission:
Submit a document with the following naming convention: *YourFirstNameLastName*-*YourLabPartnerFirstNameLastName*-Lab4.docx (for example: JohnDoe-JaneDoe-Lab4.docx) that contains the code found in lab43.cpp, lab44.cpp, lab45.cpp, and lab46.cpp. Turnitin will check the genuineness of your work. Each individual is responsible to submit their own file(s)/document(s).

For this lab assignment, you are only allowed to use the following libraries:
#include <iostream>
#include <string>

Using any other libraries is prohibited and a Zero will be assessed as your overall grade for this lab assignment.