Assignment 1: Introduction to C

In Hwan “Chris” Baek and William Kaiser

chris.inhwan.baek@ gmail.com ; kaiser@ee.ucla.edu

Introduction

In this assignment, you will learn:

1. To write simple programs in C using Vim editor,
2. To use GCC to compile your programs, and
3. To execute your programs.

Before beginning, it is important to note that this assignment is expected to be done on the Intel Edison, which runs an embedded Linux [0]. There are convenient IDEs (Integrated Development Environment) [1] such as Visual Studio and Eclipse. Please do **NOT** use an IDE for this assignment. Although IDEs are very useful, they abstract software development process. Hence, they may mislead the beginning learners about programming. Instead, you will write your C code using a text editor called “Vim” [2] and compile with GCC (GNU Compiler Collection) [3] for this assignment.

Step 1: Write, Compile, and Execute the “Hello World!” Program

First, you need to access the Intel Edison via serial connection or SSH. You can refer to IoT Tutorial 2 to make serial connection or IoT Tutorial 3 to connect via SSH. Once you log into the Intel Edison, enter the following commands to create a directory for this assignment and navigate into this:

```
$ mkdir c_program_assg1

$ cd c_program_assg1
```

You are now ready to write the “Hello World!” program. Create and open a file in Vim using the following command:

```
$ vi hello.c
```

Press the “**i**” key to enable the **insert** mode. Then, write the following C code:

```
/*
 * Hello World program
 */

#include <stdio.h>

int main() {
        printf("Hello World!\n");  //print string to stdout
        return 0;
}
```

Press the **ESC** key to go back to the normal mode. Then, type "**:wq**" and press the **ENTER** key to save and quit. Now you have written C code that is ready to be compiled. Enter the following command to compile the code:

```
$ gcc -o hello hello.c
```

You can execute the program by using the following command:

```
$ ./hello
```

You will see "**Hello World!**" message on the screen.

Step 2: Study the Code

Let us study the **hello.c** file. The first three lines show an example of comments [4]. A comment has no effect on the program. It is a good practice to add comments so that you can describe your code to others more effectively. A comment starts with **/*** and end with ***/** as shown in the first and the third lines. The second line begins with *****, which is optional but usually added to make it look nice. In the eighth line, you can find another form of comments. "**//print sting to stdout**" is a single-line comment. **//** and the following within the same line is a comment.

You can ignore the whitespaces [5] in the forth and the sixth lines. The whitespaces are only for human readability and does not have any effects on the program. The fifth line, **#include <stdio.h>,** tells the compiler to retrieve C code that is already written in a header file [6] called "**stdio.h**". The fifth line is then replaced with the retrieved code. The eighth line of the code has **printf**, which is defined in **stdio.h**. Thus, the content of **stdio.h** must be included before compiling. Otherwise, the compiler will not know what **printf** is.

The seventh line shows an example of C functions. The **main() function** [7] is the most important function in a C program. This is the entry point of the code. In other word, this is where your program starts when you execute it. The simplest form of the main function is **int main() { }**. In this code, the main() function starts at the seventh line and ends at the tenth line.

The eighth line is the interesting part. **printf() function** [8] prints a formatted string to the standard output (stdout), which is displayed on the screen by default. In this code, this function prints "**Hello World!\n**" to the screen. **\n** is the newline character, which is not displayed as "\n" itself. If you remove **\n** from the string, the output will not have any line separation between itself and the next command prompt.

What follows the **printf("Hello World\n")** is "**;**". This function call is a **statement** [5] and each statement in C is separated by semicolons. You can see that the ninth line also ends with a semicolon because it is a statement. It is a special statement called the **return statement**, which will be covered in another assignment. In this code, "**return 0**" indicates that the program executed with no error.

Step 3: Modify the Code

Now you have learned how to write a simple C program. You are asked to write a modified version of the "Hello World!" program shown above. Create a file named "**assg1_<your last name>_<your first name>.c**" (e.g. assg1_baek_chris.c). Your code will do the followings:

1. It has a comment section in the beginning that includes your name and UID,
2. It prints the following message to the screen:

   ```
   Hello
   World!

   Programming in C is fun.



   Done!
   ```
3. It has a single-line comment on each printf statement that describes what it does.

Please submit a screen capture of the displayed message along with your C code.

References

[0] https://en.wikipedia.org/wiki/Linux_on_embedded_systems

[1] https://en.wikipedia.org/wiki/Integrated_development_environment

[2] http://www.vim.org/about.php

[3] https://gcc.gnu.org/

[4] https://en.wikibooks.org/wiki/C_Programming/Structure_and_style#Comments

[5] https://en.wikibooks.org/wiki/C_Programming/Preliminaries#Block_Structure.2C_Statements.2C_Whitespace.2C_and_Scope

[6] https://gcc.gnu.org/onlinedocs/cpp/Header-Files.html#Header-Files

[7] http://en.cppreference.com/w/c/language/main_function

[8] http://en.cppreference.com/w/cpp/io/c/fprintf