C++ Programming - Basic - Assignment

Author: Peter Tse (mcreng)

This assignment should be attempted by **all** members, regardless of being a software/non-software members in internal competition period.

Introduction

Bytes and bits are common to see in embedded system, since they are controlled by digital signals and they can only be represented by high (1) and low (0). In this assignment, you will be handling binary and hexadecimal numbers which are different formats the embedded system can accept.

Copy the following header file and put it in `byte.h`. Your job is to finish the five function implementations.

Provided Header

```
/* byte.h */
/* DO NOT CHANGE ANYTHING */
#include <cstdint>
#ifndef __BYTE_H_ //#ifndef - #define - #endif will be introduced in Intermediate tutorial
#define __BYTE_H_

typedef uint8_t Byte; // defining a new type 'Byte' to be equivalent to uint8_t

Byte set_bit(Byte byte, uint8_t n);
Byte reset_bit(Byte byte, uint8_t n);
bool get_bit(Byte byte, uint8_t n);
void print_bin(Byte byte);
void print_hex(Byte byte);

#endif
```

Explanation

First, we define the digit number of the binary numbers.

```
0b11100111
  ^^^^^^^^

  76543210
```

Here provides the detailed explanations of functionalities of the five functions.

- `Byte set_bit(Byte byte, uint8_t n)`

  This returns a `Byte` with the n-th position of `Byte byte` changed to `1`.

  Expected output of `set_bit(0b00001110, 0)` is `15`.

- `Byte reset_bit(Byte byte, uint8_t n)`

  This returns a `Byte` with the n-th position of `Byte byte` changed to `0`.

  Expected output of `reset_bit(0b00001111, 0)` is `14`.

- `bool get_bit(Byte byte, uint8_t n)`

  This returns a `bool` equals to the n-th position of `Byte byte`.

  Expected output of `get_bit(0b00001110, 0)` is `0` or `false`.

- `void print_bin(Byte byte)`

  This prints the integer in binary in the format `0b<binary>`.

  Expected output of `print_bin(15)` is `0b00001111`.

- `void print_hex(Byte byte)`

  This prints the integer in hexadecimal in the format `0x<hex>`.

  Expected output of `print_hex(15)` is `0x0F`.

Submission

Another C++ program which includes `byte.h` is used to check whether your functions have been implemented correctly. Please, therefore, make sure you do not change the filename and function names when you submit your files. Finish your implementation in `byte.cpp` and submit it with a comment containing your name. Submission deadline is ????/??/??.

Appendix: Eclipse

It is **recommended** to use Eclipse to do your programming because in SmartCar team we also use Eclipse to code our cars. The instructions to install and use is as following:

- Install Eclipse Neon 1 version (Do not install any other version as it may not be compatible)
  - Windows 32-bit: http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/neon/1/eclipse-cpp-neon-1-win32.zip

  - Windows 64-bit: http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/neon/1/eclipse-cpp-neon-1-win32-x86_64.zip
  - Mac OS-X: http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/neon/1/eclipse-cpp-neon-1-macosx-cocoa-x86_64.tar.gz
- Install MinGW to compile your C++ programs
  - Windows: https://sourceforge.net/projects/mingw/files/Installer/mingw-get-setup.exe/download
- Add MinGW to your PATH
  - Windows Settings -> Edit the system environment variables
  - Advanced -> Environment Variables...
  - For 'User variables for ', press 'Edit' on variable 'Path'
  - 'New' -> Put in the path of MinGW\bin, usually in `C:\MinGW\bin`.
- Open Eclipse
  - File -> New -> C++ Project
  - Change project name/location
  - Toolchains: MinGW GCC
  - Right-click your project
    * Properties
    * C/C++ Build -> Settings -> Tool Settings
    * GCC C++ Compiler -> Miscellaneous
    * Append `-std=c++11` to Other flags
    * Apply and close
  - Pull `byte.h` to the project
  - You can create a `byte.cpp` to implement the five functions and also `main.cpp` with `int main()` to test your program
  - Press the 'Build All' button to compile the files (`Ctrl+B` works as well)
  - To execute the program, click the drop down button next to the 'bug' debug button
    * Debug configurations...
    * Left hand side: C/C++ Application

    - Create a new configuration
    - Right hand side: C/C++ Application
        - Search project...
        - Select the .exe file
- From now on you can just press the 'bug' debug button to execute the program, but remember to build it before executing