Homework 1

Write an assembly language emulator. The role of the emulator is to read in a SimpleRisc assembly language file, and execute the instructions in the assembly language file. You need to implement the following features:

1. Implement a register file with 16 registers. They are named : r0, to r15
   1. r14 can also be accessed as sp
   2. r15 can also be accessed as ra
   3. Implement the flags registers flags(E) and flags(GT)
2. Have support for specifying 16 bit immediates (in either decimal or hex (0x suffix))
   1. Immediates of the form, 2, -4, 0xABCD, 0x AB CD
   2. Note that there can be any number of spaces between 0x and the bytes. There can also be a set of spaces between the first byte and the second byte.
3. Support all 21 SimpleRisc Instructions
4. Support the modifiers – u and h
   1. For only the ALU instructions add, sub, mul, div, mod, or, and, cmp, not, mov
5. Assembly files can have any number of labels
   1. Every label needs to be of the form <name>:
   2. The label can either be on the same line as the instruction it points to, or can be anywhere after the previous statement.
   3. The program will start execution from the .main label
6. Ensure that your assembly language program has a free form.
   1. A statement has to be in one line. There can be any number of spaces between fields
   2. There can be any number of empty lines between statements.
7. Add a macro of the form: .print r1
   1. This will print the value of register r1 (in decimal) to the screen,
   2. We will use .print macros to test your program.
   3. Each call to the .print macro prints a value on a new line
8. Implement a memory
   1. It should be a 32 bit memory system
   2. However, we will use only the bottom 4096 entries
   3. Use a memory array that contains 4096 bytes
9. Initialize the stack pointer in your emulator to 0xFFF
   1. A program should not face the need to initialize the stack pointer.
10. No need to strictly implement error messages
    1. However, it is better that you at least print out some error message such that it is easier for you to debug.
11. Allowed languages – C, C++, Java, Perl, Python, Ruby, ML, OCaml, Erlang
12. Ensure that we can run a program by doing the following:
    1. All the source files should be in one directory
    2. Write a makefile. http://mrbook.org/tutorials/make/
    3. The command make should compile all the files
    4. There should be a file called run.sh (bash shell script) to run the binary, (http://www.gnu.org/software/bash/manual/bashref.html)
    5. Here, are the two commands that we will issue
       1. make
       2. ./run.sh <assembly file name>
    6. Your .print macros should print the values of the registers that we specify
13. Deadline: 15th September, 11:59 PM
14. Create a .tar.gz archive for all your files (no directories in the archive).
    1. Name the file <entry number>.tar.gz
    2. entry number starts with the year of entry (not user id)
15. You need to just emulate (execute) the instructions. There is no need to convert an instruction to a sequence of bits.
16. Submit the assignment to Sakai (details provided later)