MIPS32 Disassembler

(Required for MS Students, Optional for Undergraduate Students with a Bonus)

For this project, you will create a disassembler for MIPS32. Your program will be capable of loading a provided binary file (or text file) and displaying the MIPS assembly code equivalent along with the binary code. Your program will accept an input file *inputfilename* and output to *outputfilename* the MIPS disassembly of the input file.

Developing Environment

- Operating System: IOS or Linux (encouraged to use) or Windows
- Programming Language: C/C++, Java or other languages

Grading

Each student will have 1015 minutes to demonstrate his/her project at my office. You also need to briefly introduce your source code. Therefore, adding some comments into your source code will be helpful.

Instructions

For reference, please use the MIPS Instruction Set Architecture PDF to see the format for each instruction.
Your disassembler/simulator will need to support the following MIPS instructions:

- J, JR, BEQ, BNE, BGEZ, BGTZ, BLEZ, BLTZ
- ADDI, ADDIU
- BREAK
- SLT
- SW, LW
- SLL, SRL, SRA
- SUB, SUBU, ADD, ADDU
- AND, OR, XOR, NOR
- SLTI
- NOP

Floating point instructions (only required for Part III):

- DIV.S, MUL.S, SUB.S, ADD.S
- MOV.S
- SWC1, LWC1

Input

Your program will be given a text input file. This file will contain a sequence of 32-bit instruction words which begin at address "496". The final instruction in the sequence of instructions is always BREAK. The data section follows the BREAK instruction and begins at address "700". Following that is a sequence of 32-bit 2's complement signed integers for the program data up to the end of file. Note that the instruction words always start at address "496". If the instructions occupy beyond address "700", the data region will be followed the BREAK instruction immediately.

Your MIPS simulator (*MIPSsim)* should provide the following options to users:

MIPSsim *inputfilename outputfilename*

- *Inputfilename* - The file name of the binary input file.
- *Outputfilename* - The file name to which to print the output.

The disassembler output file will contain 4 columns of data with each column separated by one tab character ('\t' or chr(9)):
-The binary (e.g., 0's and 1's) string representing the 32-bit data word at that location. For instructions you should split this into six groups of digits to represent different parts of the MIPS instruction word: a group of 6 bits, 4 groups of 5 bits, and a final group of 6 bits.
-The address (in decimal) of that location
-The disassembled instruction opcode, or signed decimal integer value, depending on whether the current location is after the BREAK instruction.
-If you are displaying an instruction, the fourth column should contain the remaining part of the instruction, with each argument separated by a comma and then a space. (',')

The instructions and instruction arguments should be in capital letters. Display all integer values in decimal. Immediate values should be preceded by a "#" symbol. Be careful - some instructions take signed immediate values while others take unsigned immediate values. You will have to make sure you properly display a signed or unsigned value, depending on the context.

Sample Data

Here is a sample program to test your disassembler with.
-fibonacci_c.txt : This contains the C source code for the test program. This is for your reference only.
-fibonacci_mips.txt : This is the compiled version of the C code in MIPS assembly. This is for your reference only.
-fibonacci_out.txt : This is what your program should output.
-fibonacci_bin_txt.txt : This is the input txt file.

Remember that I will also test your program with other data that you will not know of in advance. It is recommended that you construct your own sample input files with which to test your disassembler further.