

CPS 104

Homework 6

Due: November 11, 11:59pm

Please note!

This assignment is the first part of a two-part sequence. You should plan to use the code you write for this assignment in order to complete the next. Please start early and make sure that you have a complete working assignment to hand-in. Also note that these two homework assignments *have a larger weight* (~1.5X) than other assignments.

Description:

Write a **Java** program that disassembles (a subset of) MIPS instructions. Your program should read a “binary” file with hexadecimal list of MIPS instructions. The file has one instruction per line. (See an example below). Your program should translate the file back into a MIPS assembler file. All constants must be printed out in hexadecimal format. Register names should be the symbolic names and NOT the register number, except register \$0. Your program should generate **labels** for the “jump” and “branch” instructions. Your program should flag illegal or unimplemented instructions (instructions not in the list).

Instruction List:

Below is the complete list of MIPS instruction that your program must decode.

Instruction	Example
Store Word	sw \$fp,0x0008(\$sp)
Load word	lw \$t5,0x0000(\$t3)
Load upper immediate	lui \$t0,0x5555
Add	add \$t0,\$0,\$0
Subtract	sub \$t6,\$t0,\$t0
Subtract unsigned	subu \$sp,\$sp,\$a0
Add unsigned	addu \$fp,\$sp,\$a0
Add immediate	addi \$t0,\$0,0x0001
Add immediate unsigned	addiu \$t0,\$v0,0xffff1
AND	and \$t0,\$t3,\$t6
OR	or \$t5,\$t2,\$t5
NOR (not OR)	nor \$t3,\$t1,\$t4
XOR	xor \$s2,\$t6,\$t0
OR immediate	ori \$t4,\$t0,0x5555
AND immediate	andi \$s6,\$s6,0x3333
XOR immediate	xori \$t2,\$t2,0x5678
Branch equal	beq \$s5,\$0, label
Branch not equal	bne \$s1,\$t0, label
Branch greater-equal 0	bgez \$s3, label
Branch less than 0	bltz \$s4, label
Branch less equal 0	blez \$t1, label
Branch greater than 0	bgtz \$t1, label
Jump	j label
Jump register	jr \$ra
Jump and link	jal label
Jump and link register	jalr \$t1
Shift left logical	sll \$v1,\$a0,1
Shift left logical variable	sllv \$t1,\$t0,\$t4
Shift right logical	srl \$t6,\$a2,1
Shift right logical variable	srlv \$s6,\$s6,\$t1
Shift right arithmetic	sra \$t7,\$a1,1

Shift right arithmetic variable	sra \$s6,\$s6,\$t1
Set less than	slt \$t2,\$t0,\$t1
Set less than unsigned	sltu \$t3,\$t0,\$t1
Set less than immediate	slti \$t4,\$t0,0xffffb
Set less than immediate unsigned	sltiu \$t5,\$t0,0xffffb
System call	syscall

The instructions format and codes are in the textbook in **appendix B** (The SPIM manual). Please note that the MIPS `text` segment starts at `0x00400000`, that is the first instruction is at location `0x00400000`. You will need this information to figure out labels' locations. You can assume that the file you read has no more than 2000 instructions.

Generating test files:

A test assembler file `test.s` with a list of assembler instructions and the corresponding "binary" file `test.out` are provided on-line

See <http://kedem.cs.duke.edu/cps104/Homework.html>

To debug your program use the program `asmr2000` to generate "binary" test files. Use the web interface: <http://www.cs.duke.edu/~kedem/asmr2000.html> to submit a MIPS program with `.s` extension (i. e. `myMIPSprog.s`) containing a list of assembly instructions you want to convert to "binary". **Hint:** Start with very-short "programs" that have only one instruction.

Additional Requirements

In your **Java** program please pre-allocate the arrays that you need and then use that space to store the instructions.

Use bit-wise operations (Mask and Shift) to extract instructions' sub-fields.

Hints

- Start Early! We can provide extra UTA "office hours". Ask questions early. Don't wait for the last moment before the assignment is due.
- Deal with I/O first since Java I/O is a bit cumbersome. Please consult the code we provide on-line. You are free to "borrow" from it (or not). Java-5 supports `printf()` you might want to learn how to use it
- Don't write the whole program and then try a debug it all at once. Start with one simple instruction and build from there.
- Dealing with labels requires a bit of thinking, planning and experimenting. It might take longer then you think to get that part working, so **START EARLY!**
- Use the program `asmr2000` to generate test cases to test your code.
- *Remember that Pseudo-Instructions are translated into native MIPS instructions!*

What to submit

You must submit a program that gets a file-name interactively and opens the file. The file is a "binary" file with a list of assembler instructions in hexadecimal format. Alternatively, your program can read the file from `<standard_input>` (`system.in` in Java). Your program should translate the "binary" file to a list of assembler instructions and output a file to `<standard_out>` (`system.out` in Java).

Please Submit to the folder **HW6**

- Your source code,
- A **README** file that has a description of what the program is doing, and Instructions on how to run your program