Test 2 will be Thursday, November 16 in class. It will be closed book and notes, except for one 8.5" x 11" sheet of paper (front and back) with notes, green MARIE Assembly Language handout, and the ***front*** page of your MIPS Assembly Language Guide (available at: http://www.cs.uni.edu/~fienup/cs041f06/lectures).

Chapter 4.
Contribution of Computer Components: CPU, Bus, Memory, Clocks, I/O subsystems, Interrupts
CPU Basics: Fetch-Decode-Execute machine cycle, datapath including ALU and registers, control unit
Bus: types of lines (data, address, and control), types of buses (processor-memory, I/O buses), synchronous vs. asynchronous buses, bus arbitration schemes (daisy chain, centralized parallel, distributed arbitration using self-selection, distributed arbitration using collision detection)
Memory: (covered on the last test)
Clocks: frequency, relationship to program performance
I/O subsystems: I/O interface, memory-mapped vs. special I/O instructions
Interrupts: causes, alters normal flow of execution, maskable vs. nonmaskable
MARIE: Architecture, Instr. Set Architecture, machine language format, RTN/RTL for instructions, Fetch-Decode-Execute cycle, I/O
Assembly Process: two-passes, symbol table, assembler directives
Hardwired vs. Microprogrammed Control Units

MIPS Assembly Language
MIPS Processor Architecture: registers, register conventions, addressing modes, memory layout
Basic MIPS Instruction Set: loads/stores, arithmetic instructions, logical instructions, shift/rotate instructions, branch/jump instructions
SPIM Assembler Directives: .data, .text, .word, .globl

In addition to knowledge about the above concepts, the following assembly-language programming skills are to be tested too:
1) translate high-level language control statements (while, for, if, etc.) into MARIE and MIPS assembly language (be able to handle complex Boolean expressions involving ANDs, ORs, etc.)
2) translate high-level language code containing array accesses into MIPS assembly language

MATERIAL THAT WILL NOT BE ON TEST 2:
MIPS Assembly Language
MIPS Instruction Set: three ML instruction formats
Subprograms: MIPS Register conventions
MIPS Logical, Shift/Rotate Instructions
SPIM I/O and other System Calls:
SPIM Assembler Directives: .asciiz, .ascii, .align, .space
Arrays: element addressing 1-d, 2-d, 3-d, and higher
Walking pointer through an array

ASSEMBLY-LANGUAGE PROGRAMMING SKILLS THAT WILL NOT BE ON TEST 2:

3) use MIPS register conventions to decide which arguments/parameters and local variables should be stored in caller-saved ($a and $t-registers) or callee-saved ($s-registers)
4) translate high-level language subprograms into MIPS assembly language (passing parameters into the subprogram using the $a registers, building the call frame on the run-time stack if necessary, save $s and $ra registers if necessary, passing the value returned by a function in the $v0 register, restoring $s and $ra registers if necessary, jr back to the caller)