

*The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL*

**Comp 411 Computer Organization**  
Spring 2014

**Problem Set #6**

*Issued Monday, 4/14/14; Due Tuesday, 4/22/14*  
*(hand in your work at start of class)*

Note: You may use additional sheets of paper, but please enter your answers in the space provided in this document.

# COMP 411

## miniMIPS

# COMP 411

## miniMIPS

The diagram illustrates the miniMIPS architecture, a simplified MIPS processor. Key components and their interactions are as follows:

- PCSEL (Program Counter Select):** A 7-bit register (bits 6-0) that selects between different PC update sources: 0x80000000, 0x80000040, 0x80000080, JT, BT, and PC+4.
- PC (Program Counter):** A 32-bit register that holds the current instruction address. It is updated by PCSEL and incremented by 4.
- Instruction Memory:** Provides instructions to the ALU and Register File based on the PC value.
- Register File:** Contains 32 registers (RA1, RA2, RD1, RD2). It receives write data (Wd) and write enable (WE) signals. It provides read data (Rd) and read enable (RE) signals to the ALU.
- Control Logic:** Receives control signals (RESET, IRQ, Z, N, Y, C) and provides control signals (PCSEL, WASEL, SEXT, BSEL, WDESEL, ALUFN, Wr, WERF, ASEL) to other components.
- ALU (Arithmetic Logic Unit):** Performs operations on register values (RA1, RA2) and immediate values (Imm). It supports operations like addition, subtraction, multiplication, and division. It outputs results to the Register File and provides status flags (NV, C, Z).
- Data Memory:** Provides data to the ALU based on the address (Adr) and read data (RD). It also receives write data (Wd) and write enable (WE) signals.
- PC+4:** A 32-bit register that holds the next sequential instruction address (PC + 4).

Fill in the missing entries of the Control Logic in the table below, based on the data path shown above. *Hint:* All of the information you need to answer this question can be found in the slides from the lecture on building a computer).

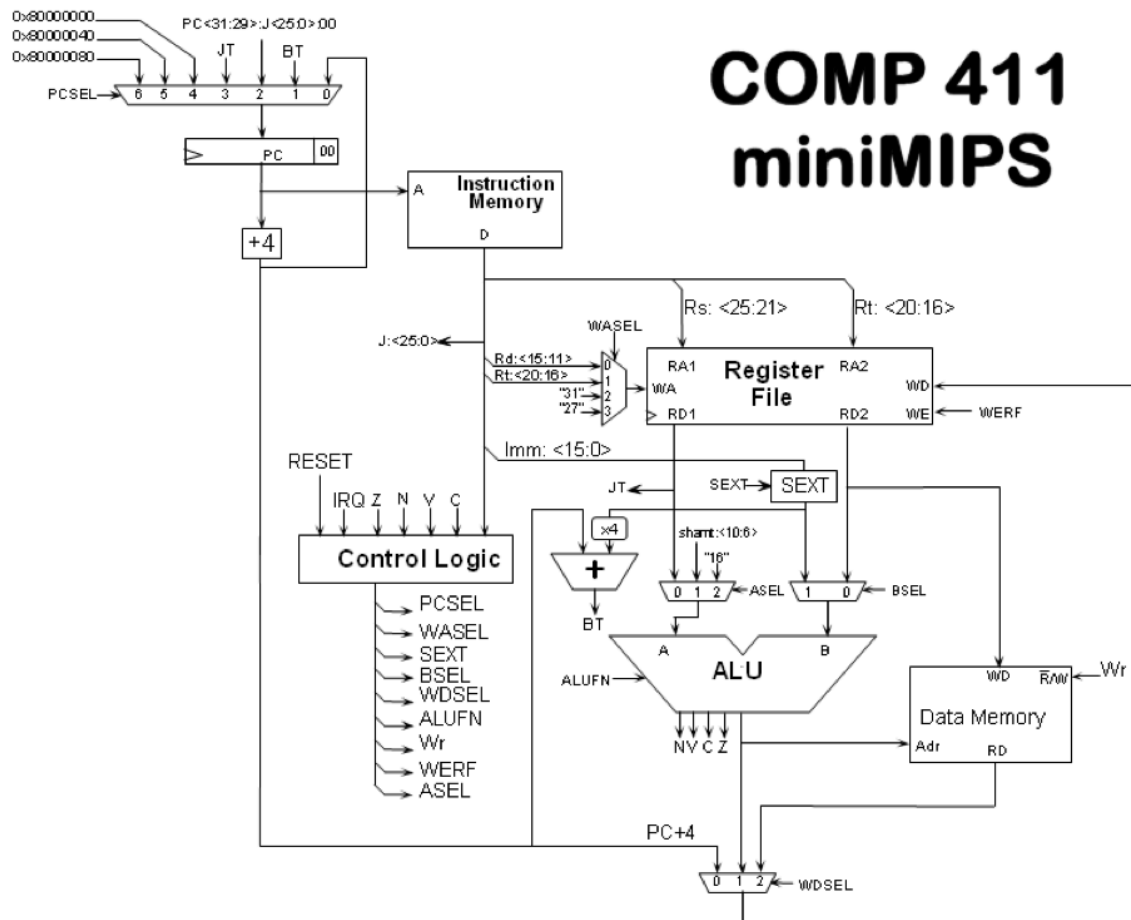
Opcode	PCSEL	WASEL	SEXT	BSEL	ASEL	WDSEL	ALUFN				Wr	WERF
							Sub	Bool	Shift	Math		
sub	0	0	X	0	0	1	1	XX	0	1	0	1
xor	0	0	X	0	0	1	X	10	0	0	0	1
addi	0	1	1	1	0	1	0	XX	0?	1	0	1
sll	0	0	X	0	1	1	X	00	1	0	0	1
andi	0	1	1	1	0	1	X	00	0	0	0	1
lw	0	1	1	1	0	2	0	XX	0	1	0	1
sw	0	X	1	1	0	X	X	XX	X?	X?	1	0
j	2?	X	X	X?	X?	X	X	XX	X	X	0	0
jal	3?	2	X	X?	X?	0?	X	XX	X	X	0	1
lui	0	1	1?	1	2	1	X	00	1	0	0	1

## Problem 2. “Expanding our Mini MIPS” (20 points)

For this problem, you are to expand the mini MIPS circuitry that we built in class to accommodate one more instruction. This instruction will be an auto-increment for store word (*swi*). For *swi* our mini MIPS will perform a typical store word, but then the register used to compute the address in main memory will be incremented by four. As an example, for the instruction

`swi $t1, 0x4444($t2)`

the contents of register `$t1` will be stored at address `0x4444 + $t2`. However, now the register `$t2` will also be incremented by four. This could be useful when iterating through an array, for example. Please **neatly and carefully** draw your added circuitry directly onto our mini MIPS implementation below. If you need to perform some trial and error first, please use a different sheet of paper.



**Problem 3. “Cache Memory” (20 points)**

For a direct-mapped cache design with a 32-bit address, the following bits of the address are used to access the cache.

Tag	Index	Offset
31-10	9-5	4-0

- a) What is the cache block size (in words)?

$$22 = 32 - (n + m + 2) \rightarrow \text{block size} = 2^3 = 8$$

- b) How many entries does the cache have?

$$2^5 = 32$$

- c) What is the ratio between total bits required for such a cache implementation over the data storage bits?

$$\text{total / storage bits} = ((32 * 22) * (2^{12} * 8) + 32) / 2^{12} * 8 = 1.089$$

**Problem 4. “Cache Memory” (20 points)**

Starting from power on, the following byte-addressed cache references are recorded.

Address											
0	4	16	132	232	160	1024	30	140	3100	180	2180

- a) How many blocks are replaced?

4

- b) What is the hit ratio?

.25

- c) List the final state of the cache, with each valid entry represented as a record of <index, tag, data>.

Address	Index	Tag	Data
0	0	0	-
4	0	0	-
16	0	0	-
132	1111	0	-
232	111	0	exists
160	101	0	-
1024	0	1	-
30	0	0	-
140	100	0	-
3100	10	11	exists
180	101	0	exists
2180	00100	10	exists