The University of Alabama in Huntsville
Electrical & Computer Engineering Department
CPE 431 01
Test 1 Solution
Fall 2003

1. (10 points) Use the following code fragment:

```
loop: lw    $1, 0($2)
      addi  $1, $1, 4
      sw    $1, 0($2)
      addi  $2, $2, 8
      bne   $2, $3, loop
```

Assume that the initial value of $2 is $3 — 192 and that this code fragment is run on a machine with a 500-MHz clock that requires the following number of cycles for each instruction:

Instruction	Cycles
addi	1
sw, bne	2
lw	3

In the worst case, how many seconds will it take to execute this code?

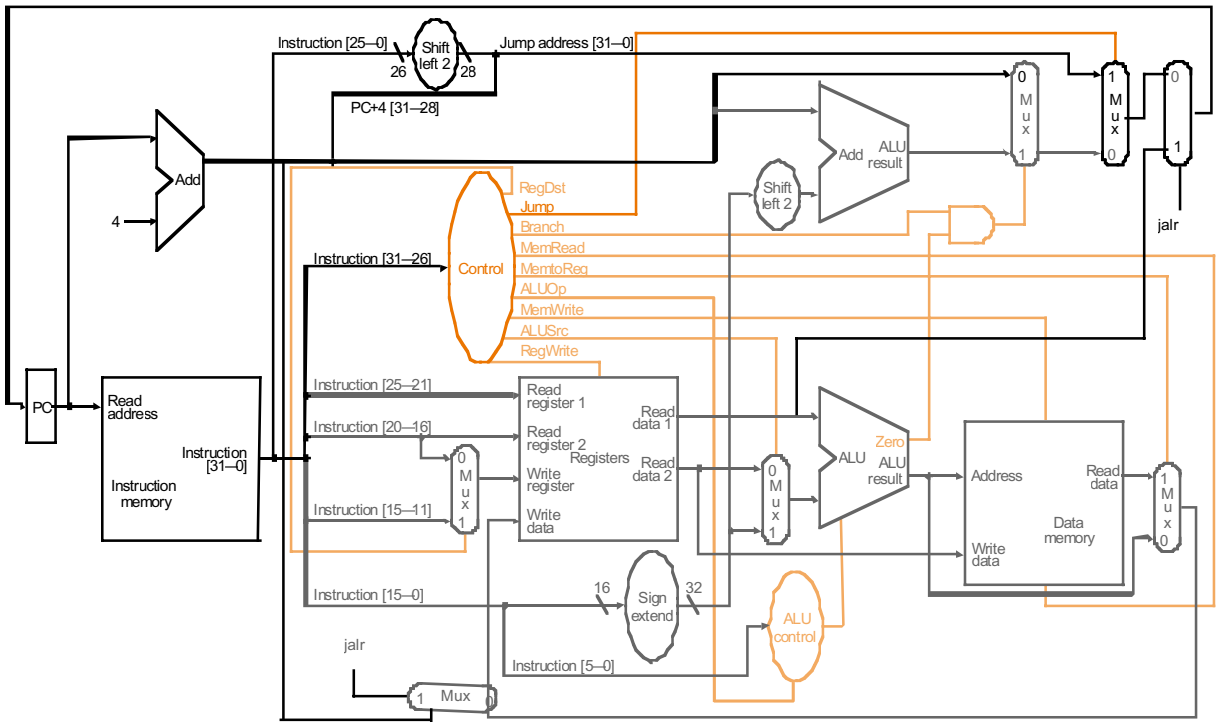
ET = #iterations*cycles/iteration*s/cycle
#iterations = (184 — 0)/8 + 1 = 23 + 1 = 24
cycles/iteration = 3(lw) + 1(addi} + 2(sw} + 1(addi) + 2(bne) = 9
s/cycle = 1/500*10^6 cycles/s = 2 ns
ET = 24*9*2 ns = 432 ns

2. (1 point) In the 1960s and 1970s, a primary constraint on computer performance was the size of the computer s ____memory____.
3. (1 point) The manufacture of an integrated circuit begins with __silicon__, a substance found in sand.
4. (1 point) The field of a floating point number which represents its fractional part is called the ___significand/mantissa____.
5. (15 points) Add the instruction `jalr` (jump and link register) to the single-cycle datapath shown in the figure below. The `jalr` instruction is defined below. Add any necessary datapaths and control signals and show the necessary additions to the table of control signals given.

`jalr rd, rs`

PC ← $rs
$rd ← PC + 4

0	rs	0	rd	0	9



Instruction	RegDst	ALUSrc	Memto Reg	Re Write	Mem Read	Mem Write	Branch	ALUOp1	ALUOp0	jalr
R-format	1	0	0	1	0	0	0	1	0	0
lw	0	1	1	1	1	0	0	0	0	0
sw	d	1	d	0	0	1	0	0	0	0
beq	d	0	d	0	0	0	1	0	1	0
jalr	1	d	d	1	d	0	d	d	d	1

d-don t care

6. (1 point) The set of programs run to evaluate a new computer form a ___workload___.
7. (1 point) MIPS instructions are _____3_____ operand instructions.
8. (15 points) When designing memory systems, it becomes useful to know the frequency of memory reads versus writes and also accesses for instructions versus data. Using the following average instruction-mix information, find
    a. (5 points) the percentage of all memory accesses for instructions
    b. (5 points) the percentage of memory accesses that are writes
    c. (5 points) the percentage of all data accesses that are reads.

Instruction	Percentage
`lw`	29
`sw`	15
`add`	18
`sub`	3
`lui`	7
`beq, bne`	6
`jump`	3
`and, or`	16
`mult`	3

a. $\frac{\#\,instructions}{\#\,memory\_accesses} = \frac{1.0IC}{1.0IC + lw + sw} = \frac{1.0IC}{1.0IC + 0.29IC + 0.15IC} = \frac{1.0IC}{1.44IC} = 69\%$

b. $\frac{\#\,writes}{\#\,memory\_accesses} = \frac{sw}{1.0IC + lw + sw} = \frac{0.15IC}{1.0IC + 0.29IC + 0.15IC} = \frac{0.15IC}{1.44IC} = 10\%$

c. $\frac{\#\,read\_data\_accesses}{\#\,data\_accesses} = \frac{lw}{lw + sw} = \frac{0.29IC}{0.29IC + 0.15IC} = \frac{0.29IC}{0.44IC} = 66\%$

9. (5 points) Show the single MIPS instruction or minimal sequence of instructions for this C statement:

```
x[ 10]  = x[ 11]  + c;
```

Assume that `c` corresponds to register `$t0` and the array `x` has a base address of $4{,}000{,}000_{10}$ which is stored in register `$t1`.

```
lw   $t2, 44($t1)
add  $t2, $t2, $t0
sw   $t2, 40($t1)
```

10. (5 points) What number does the two s complement binary number represent:

`1101 1110 0010 1100`$_2$?

$-2^{15} + 2^{14} + 2^{12} + 2^{11} + 2^{10} + 2^9 + 2^5 + 2^3 + 2^2 = -32768 + 16384 + 4096 + 2048 + 1024 + 512 + 32 + 8 + 4 = -8660_{10}$

11. (15 points) You are the lead designer of a new processor. The processor design and compiler are complete, and you must now decide whether to produce the current design as it stands or spend additional time to improve it. You discuss this problem with your hardware engineering team and arrive at the following options:

a. Leave the design as it stands. Call this base machine Mbase. It has a clock rate of 500 MHz, and the following measurements have been made using a simulator:

Instruction Class	CPI	Frequency
A	2	40%
B	3	25%
C	3	25%
D	5	10%

b. Optimize the hardware. The hardware team claims that it can improve the processor design to give it a clock rate of 600 MHz. Call this machine Mopt. The following measurements were made using a simulator for Mopt:

Instruction Class	CPI	Frequency
A	2	40%
B	2	25%
C	3	25%
D	4	10%

(10 points)What is the CPI for each machine? (5 points) How much faster is Mopt than Mbase?

$CPI_{base} = 2 * 0.4 + 3 * 0.25 + 3 * 0.25 + 5 * 0.1 = 0.8 + 0.75 + 0.75 + 0.5 = 2.8$

$CPI_{opt} = 2 * 0.4 + 2 * 0.25 + 3 * 0.25 + 4 * 0.1 = 0.8 + 0.5 + 0.75 + 0.4 = 2.45$

$IC_{base} = IC_{opt} = IC$

$$\frac{P_{opt}}{P_{base}} = \frac{ET_{base}}{ET_{opt}} = \frac{IC_{base} * CPI_{base} * CC_{base}}{IC_{opt} * CPI_{opt} * CC_{opt}} = \frac{IC * 2.8 * \frac{1}{500 \times 10^6}}{IC * 2.45 * \frac{1}{600 \times 10^6}} = \frac{2.8 * 600}{2.45 * 500} = 1.37$$

12. (10 points) The table below shows the number of floating-point operations executed in two different programs and the runtime for those programs on three different machines:

Program	Floating-point operations	Execution time in seconds		
		Computer A	Computer B	Computer C
1	10,000,000	1	10	20
2	100,000,000	1000	100	20

One user has told you that the two programs above constitute the bulk of his workload, but he does not run them equally. The user wants to determine how the three machines compare when the workload consists of different mixes of these programs. Suppose the total number of FLOPS executed in the workload is divided two thirds for program one and one third for program 2. Find which machine is fastest for this workload and by how much.

$FLOPS_1 = 2/3\ FLOPS_{total}$, $FLOPS_2 = 1/3\ FLOPS_{total}$

For 1 run of program 2, 100,000,000 = 1/3FLOPStotal,so FLOPStotal = 300,000,000

So, FLOPS1 = 200,000,000 or for every time program 2 is run once, program 1 must be run 200,000,000/10,000,000 or 20 times

$$ET_A = \frac{20}{21} * 1 + \frac{1}{21} * 1000 = \frac{1020}{21}, ET_B = \frac{20}{21} * 10 + \frac{1}{21} * 100 = \frac{300}{21}, ET_C = \frac{20}{21} * 20 + \frac{1}{21} * 20 = \frac{420}{21}$$

$$\frac{P_B}{P_A} = \frac{ET_A}{ET_B} = \frac{\frac{1020}{21}}{\frac{300}{21}} = \frac{1020}{300} = 3.4 \qquad \frac{P_B}{P_C} = \frac{ET_C}{ET_B} = \frac{\frac{420}{21}}{\frac{300}{21}} = 4\frac{1020}{300} = 1.4$$

13. (5 points) Given the bit pattern:

`0000 00 00 000 0 1000 0101 0 011 11 00 0000`$_2$

what does it represent, assuming that it is a MIPS instruction?

op = `000000`, rs = `00000`, rt = `01000`, rd = `01010`, shamt = `01111`, funct = `000000`

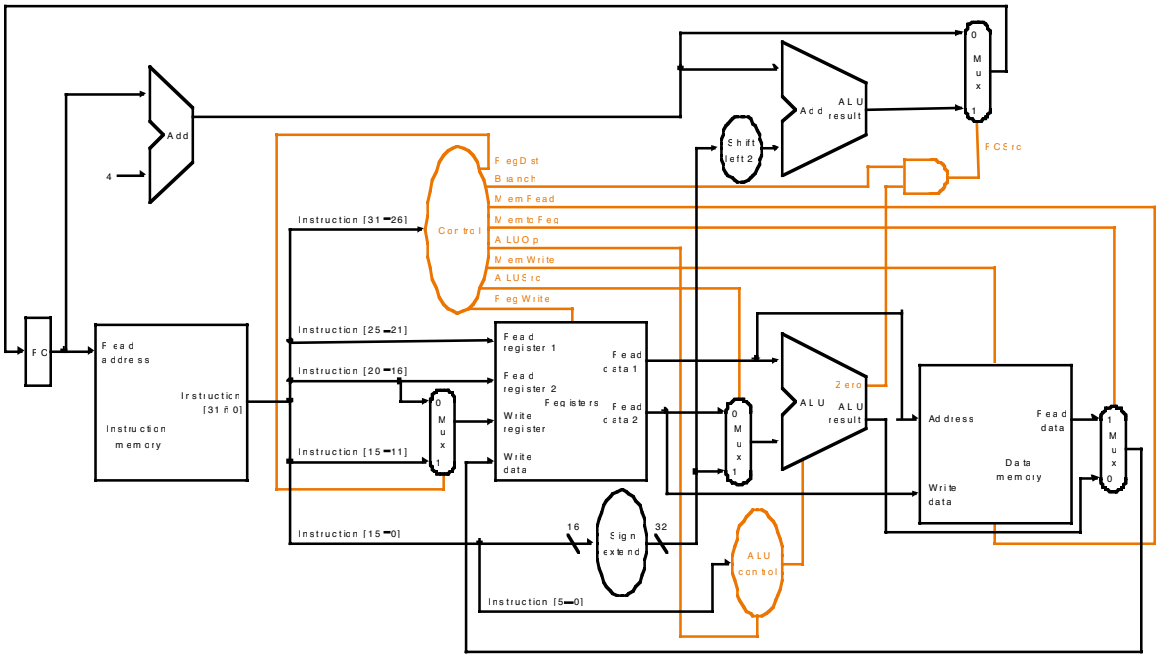
```
sll  $t2, $t0, 15
```

14. (15 points) Consider the following idea: Let s modify the instruction set architecture and remove the ability to specify an offset for memory access instructions. Specifically, all load-store instructions

with nonzero offsets would become pseudoinstructions and would be implemented using two instructions. For example:

```
addi $at, $t1, 104          # add the offset to a temporary
lw   $t0, $at               # new way of doing lw $t0, 104 ($t1)
```

(a) (5 points) What changes would you have to make to the single-cycle datapath and control if this simplified architecture were to be used?



(b) (10 points) If the delay for the instruction and data memory is 2 ns, the register file delay (read or write) is 1 ns, and the ALU delay is 2 ns, what are the clock cycles required for the original datapath and for the modified datapath? What is the highest percentage of load-store instructions with offsets that could be tolerated without total performance being degraded?

`lw` is the longest for both methods

CC_{orig} = 2 ns + 1 ns + 2 ns + 2 ns + 1 ns = 8 ns
CC_{mod} = 2 ns + 1 ns + 2 ns + 1 ns = 6 ns

For every load-store with a nonzero offset, an additional instruction must be executed. Thus, if the fraction of instructions which is a `lw` with a nonzero offset is x, the number of modified instructions is $IC_{orig}(1 + x)$. Remembering that $CPI_{mod} = CPI_{orig} = 1$

$$\frac{P_{mod}}{P_{orig}} = \frac{ET_{orig}}{ET_{mod}} = \frac{IC_{orig} * CPI_{orig} * CC_{orig}}{IC_{mod} * CPI_{mod} * CC_{mod}} = \frac{IC_{orig} * 1 * 8ns}{IC_{orig}(1+x) * 1 * 6ns} = 1$$

IC_{orig}*1*8 ns = IC_{orig}(1 + x)*1*6 ns
1.33 = 1 + x and x = 0.33

So, 33 % of the instructions could be `lw` with nonzero offsets and the total impact on performance would still be positive.