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

Name: _____

Remember: CPU_{time}= Execution time
CPU_{time}=Instructioncount*CPI*Clockcycletime

1. (10 points) Consider an architecture that is similar to MIPS except that it supports update addressing for data transfer instructions. If we run gcc using this architecture, some percentage of the data transfer instructions will be able to make use of the new instructions, and for each instruction changes, one arithmetic instruction can be eliminated. If 25% of the data transfer instructions can be changed, which will be faster for gcc, the modified MIPS architecture or the unmodified architecture? Assume the CPI values shown and that the modified architecture has its cycle time increased by 15% in order to accommodate the new instructions.

Instruction Class	Average CPI	Frequency in gcc	Adjusted Frequency
Arithmetic	1.0	48 %	(0.48 – 0.0825)/0.9175 = 0.433
Data transfer	1.4	33%	0.33/0.9175 = 0.360
Conditional branch	1.7	17%	0.17/0.9175 = 0.185
Jump	1.2	2%	0.02/0.9175 = 0.022

$CPI_{unmodified}$ = 0.48 * 1.0 + 0.33 8 1.4 + 0.17 8* 1.7 + 0.02 *1.2 = 0.48 + 0.462 + 0.289 + 0.024
= 1.255

0.25 * 0.33 or 0.0825 of the data transfer have updates, so that number of arithmetic instructions can be eliminated.

CPImodified = 0.433 * 1 + 0.360 * 1.4 + 0.185 * 1.7 + 0.022 * 1.2 = 0.433 + 0.504 + 0.320 + 0.029 = 1.286

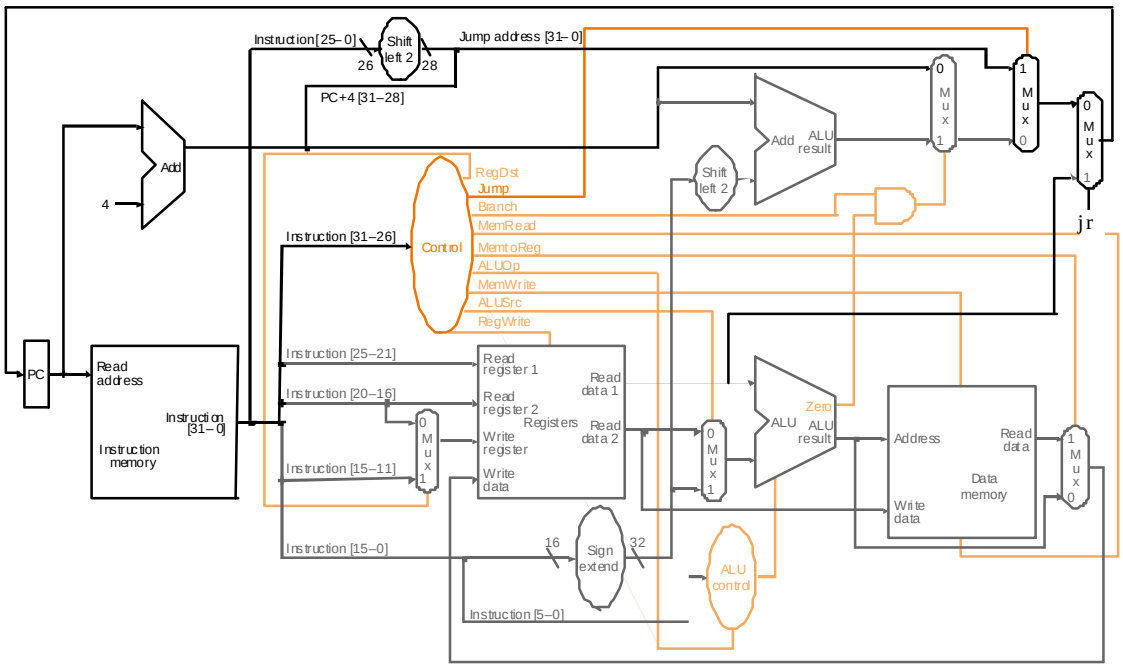
$$\frac{P_{\text{mod }ified}}{P_{un\,\text{mod }ified}} = \frac{ET_{un\,\text{mod }ified}}{ET_{\text{mod }ified}} = \frac{IC_{un\,\text{mod }ified} * CPI_{un\,\text{mod }ified} * CC_{un\,\text{mod }ified}}{IC_{\text{mod }ified} * CPI_{\text{mod }ified} * CC_{\text{mod }ified}} = \frac{IC_u * 1.255 * CC_u}{0.9175 IC_u * 1.286 * 1.15 * CC_u} = 0.925$$

Since the performance of the modified system is worse, do not proceed with the modification.

2. (1 point) A _____bit_____ is another name for binary digit.

3. (1 point) A _clocking methodology____ defines when signals can be read and when they can be written.

4. (10 points) Add the instruction `jr` (jump register) to the single-cycle datapath shown in the figure below. Add any necessary datapaths and control signals and show the necessary additions to the table of control signals given.



Instruction	RegDst	ALUSrc	Memto Reg	Reg Write	Mem Read	Mem Write	Branch	ALUOp1	ALUOp0	jr
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
jr	d	d	d	0	0	0	d	d	d	1

d-don't care

5. (1 point) A _____defect_____ is a microscopic flaw in a wafer.

6. (1 point) The layout of the instruction is called the _____instruction format_____.

7. (1 point) _____Benchmarks_____ are programs specifically chosen to measure performance.

8. (15 points) Assume that multiply instructions take 12 cycles and account for 10% of the instructions in a typical program and that the other 90% of the instructions require an average of 4 cycles for each instruction. Your hardware team has indicated that it would be possible to reduce the number of cycles required for multiplication to 6, but this will require a 20% increase in the cycle time? Nothing else will be affected. Should they proceed with the modification?

$CPI_{orig} = 0.9 * 4 + 0.1 * 12 = 3.6 + 1.2 = 4.8$

$CPI_{mod} = 0.9 * 4 + 0.1 * 6 = 3.6 + 0.6 = 4.2$

$Ic_{orig} = Ic_{mod} = IC, \qquad CC_{mod} = 1.2 * CC_{orig}$

$Et_{orig} = IC * CPI_{orig} * Cc_{orig} = IC * 4.8 * CC_{orig}$

$Et_{mod} = IC * CPI_{mod} * CC_{mod} = IC * 4.2 * 1.2 * CC_{orig}$

$$\frac{P_{modified}}{P_{orig}} = \frac{ET_{orig}}{ET_{modified}} = \frac{IC * CPI_{orig} * CC_{orig}}{IC * CPI_{modified} * CC_{modified}} = \frac{4.8 * CC_{orig}}{4.2 * 1.2 * CC_{orig}} = 0.952$$

Do not proceed because the modified performance is worse than the original performance.

9. (5 points) Write a minimal sequence of actual MIPS instructions to accomplish the same thing as the following pseusoinstruction:

```
clear $t5
```

`add $t5, $zero, $zero` or `sub $t5, $t5, $t5` or `addi $t5, $zero, 0`

10. (10 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 data accesses that are writes

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

a. $$\frac{instructions}{instructions + lw + sw} = \frac{IC}{IC(1 + 0.29 + 0.15)} = \frac{1}{1.44} = 69\%$$

b. $\frac{sw}{lw+sw} = \frac{0.15}{0.29+0.15} = \frac{0.15}{0.44} = 34\%$

11. (10 points) Given the bit pattern:

`1111 0110 0110 1100`$_2$?

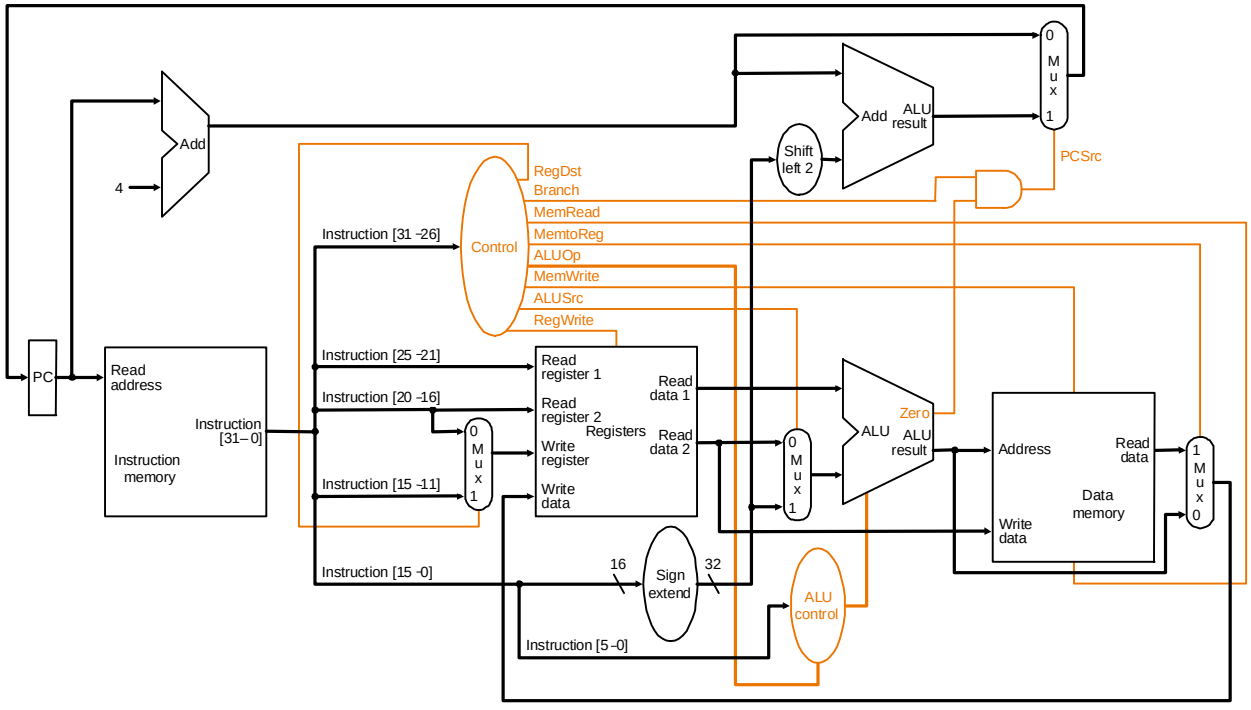
what does it represent, assuming it is

a. a two's complement integer?
   Take the twos complement and get `0000 1001 1001 0100`
   which is –(9 * 256 + 9 * 16 + 4) = -2452$_{10}$
b. an unsigned integer?
   This represents 15 * 4096 + 6 * 256 + 6 * 16 + 12 = 61440 + 1536 + 96 + 12 = 63084$_{10}$

12. (20 points) a. (5 points) Describe the effect that a single stuck-at-1 fault (i.e., regardless of what it should be, the signal is always 1) would have on the control signals in the single-cycle datapath shown. b. (15 points) Which instructions, if any would still work? Consider each of the following faults separately: RegDst = 1, ALUSrc = 1, MemtoReg = 1, RegWrite = 1.

b.

	RegDst = 1	ALUSrc = 1	MemtoReg = 1	RegWrite= 1
R-type	works	doesn't work	doesn't work	works
lw	doesn't work	works	works	works
sw	works	works	works	doesn't work
beq	works	doesn't work	works	doesn't work



a. RegDst s-a-l the register written always comes from the rd field of the instruction
ALUSrc s-a-l the second input of the ALU always comes from the sign extended immediate field of the instruction
MemtoReg s-a-l the data written to the register file always comes from memory
RegWrite s-a-l the register file is written every cycle

13. (15 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 that equal amounts of time will be spent running each program on some machine. Which machine is fastest assuming a weighting that generates equal execution time for each benchmark on machine A? How does this compare with the performance for a workload with equal number s of program executions?

First workload, equal amounts of time for programs 1 and 2 on machine A. To do this, program 1 must be run 1000 times for every time program 2 is run.

$ET_A = 1*1000 + 1000 * 1 = 2000s$
$ET_B = 10 * 1000 + 100 * 1 = 10000 + 100 = 10100s$
$ET_C = 20 * 1000 + 20 * 1 = 20000 + 20 = 20020s$

For this workload, machine A is the fastest.

Second workload, equal executions for programs 1 and 2.

$ET_A = 1000 + 1 = 1001s$
$ET_B = 10 + 100 = 110s$
$ET_C = 20 + 20 = 40s$

For the workload, machine C is the fastest.