The University of Alabama in Huntsville
ECE Department
CPE 431 01
Test 1 Solution

You *must* show your work to receive full credit: You may use additional sheets of paper for your work, please put your name on each additional sheet.
Remember: ET = IC*CPI*CT

1. (1 point) A _**benchmark**_ is a program specifically chosen to measure performance.

2. (1 point) A program that manages the resources of a computer for the benefit of the programs that run on that machine is a(n) _**operating system**_.

3. (1 point) To specify a register choice from a register file with 256 registers requires _**8**_ bits.

4. (1 point) Performance is _**inversely**_ related to execution time.

5. (1 point) The address specified in a branch, which becomes the new program counter is the branch is taken, is known as the _**target**_ address.

6. (15 points) Translate the following C code to MIPS. Assume that the variables `f`, `g`, `h`, `i`, and `j` are given and are assigned to registers `$s0`, `$s1`, `$s2`, `$s3`, and `$s4`, respectively. Assume that the base address of the arrays A and B are in registers `$s6` and `$s7`, respectively. Assume that the elements of the arrays A and B are 4-byte words:

```
      B[i] = A[A[j]]
sll   $t0, s4, 2
add   $t0, $t0, $s6
lw    $t0, 0($t0)
sll   $t0, $t0, 2
add   $t0, $t0, $s6
lw    $t0, 0($t0)
sll   $t1, $s3, 2
add   $t1, $t1, $s7
sw    $t0, 0($t1)
```

6. (15 points) MIPS chooses to simplify the structure of its instructions. The way we implement complex instructions is to decompose such complex instructions into multiple simpler MIPS ones. The complex instruction `swap $rs, $rt` can be decomposed into three instructions. If the implementation of the swap instruction in hardware will increase the clock period of a single-cycle implementation by 10%, what percentage of `swap` operations in the instruction mix would recommend implementing it in hardware?

Let x represent the proportion of instructions that are swap instructions. Consider two cases: (1) before change in which swap instructions are pseudo-instructions which are replaced by three instructions, (2) after change in which swap instructions are implemented directly in hardware.

$CPI_{before} = CPI_{after} = 1$, $CT_{after} = 1.1CT_{before}$, $IC_{before} = (x*3 +(1-x)*1)IC_{after}$

$ET_{before} = ET_{after}$

$CPI_{before}*IC_{before}*CT_{before} = CPI_{after}*IC_{after}*CT_{after}$

Substituting for IC_{before} and CT_{after},,

$1*(x*3 +(1-x)*1)IC_{after}*CT_{before} = 1*IC_{after}*1.1CT_{before}$

$3x + 1 – x = 1.1$

$2x = 0.1$

$x = 0.05$ or 5 %

8. (10 points) Write down the hexadecimal representation of the decimal number 3967.4453125 in the IEEE 754 double precision format.

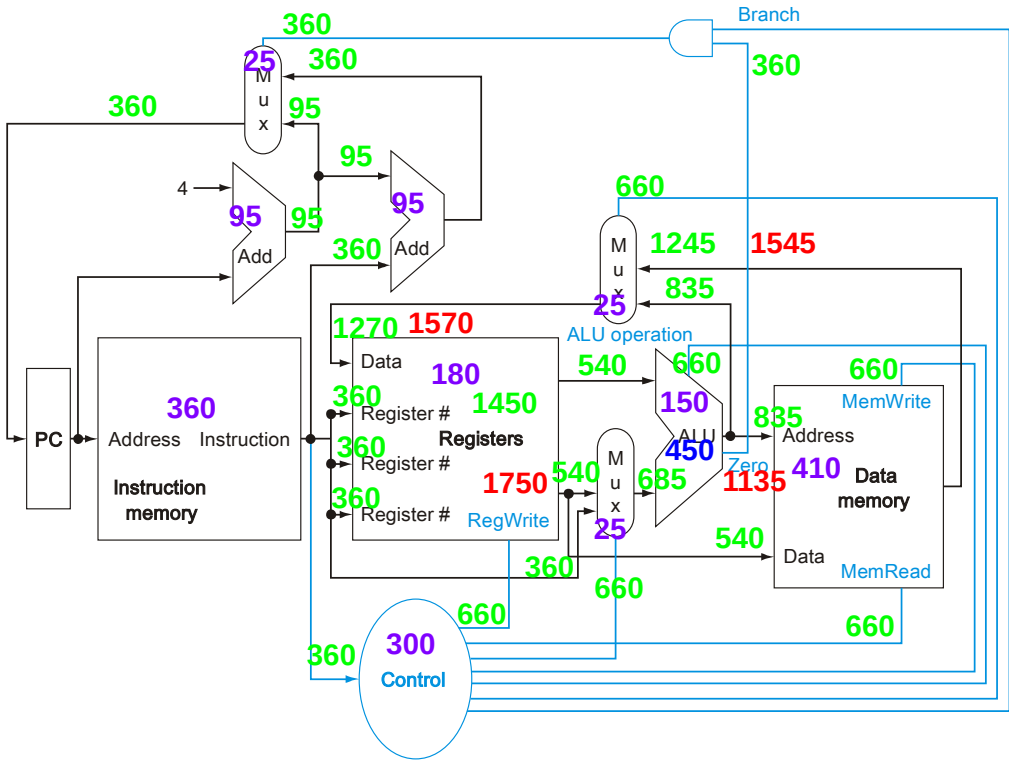
```
/2        0   1              x2    0.4453125
/2        1   1              x2    0.890625      0
/2        3   1              x2    1.78125       1
/2        7   1              x2    1.5625        1
/2       15   0              x2    1.125         1
/2       30   1              x2    0.25          0
/2       61   1              x2    0.5           0
/2      123   1              x2    1.0           1
/2      247   1
/2      495   1
/2      991   1
/2     1983   1
/2     3967
```

1111 0111 1111.0111 001 = 1.1110 1111 1110 1110 01 x 2^{11}
S = 0, EXP + BIAS = 11 + 1023 = 1034 = 100 0000 1010
Fraction = 1110 1111 1110 1110 01
0100 0000 1010 1110 1111 1110 1110 0100 = 0x40AE FEE4 0000 0000

9. (25 points) When processor designers consider a possible improvement to the processor datapath, the decision usually depends on the cost/performance trade-off. assume that we are starting with the datapath below, where I-Mem, Add, Mux, ALU, Regs, D-Mem, and Control blocks have latencies of 360 ps, 95 ps, 25 ps, 150 ps, 180 ps, 410 ps, and 300 ps, respectively, and costs of 1000, 30, 10, 100, 200, 2000, and 500, respectively.
Consider the addition of a multiplier to the ALU. This addition will add 300 ps to the latency of the ALU and will add a cost of 600 to the ALU. The result will be 5% fewer instructions executed since we will no longer need to emulate the MUL instruction.

(a) (15 points) What is the clock cycle time with and without this improvement?

CT_{before} = 1450 ps **CT_{after} = 1750 ps**



xxx -Original Element Delay
xxx -Original Path Delay
xxx -Modified Path Delay
xxx -Modified Element Delay

(b) (10 points) What is the ratio of cost/performance after the improvement to the cost/performance before the improvement?

$Cost_{before}$ = $Cost_{I\text{-}Mem}$ + 2*$Cost_{Add}$ + 3*$Cost_{MUX}$ + $Cost_{ALU}$ + $Cost_{Regs}$ + $Cost_{D\text{-}Mem}$ + $Cost_{Control}$ = 1000 + 2*30 + 3*10 + 100 + 200 + 1000 + 500 = 3890

$Cost_{after}$ = $Cost_{I\text{-}Mem}$ + 2*$Cost_{Add}$ + 3*$Cost_{MUX}$ + $Cost_{ALU}$ + $Cost_{Regs}$ + $Cost_{D\text{-}Mem}$ + $Cost_{Control}$ = 1000 + 2*30 + 3*10 + 700 + 200 + 1000 + 500 = 4490

$$\frac{\left(\frac{Cost}{Performance}\right)_{after}}{\left(\frac{Cost}{Performance}\right)_{before}} = \frac{\left(\frac{Cost}{\frac{1}{ET}}\right)_{after}}{\left(\frac{Cost}{\frac{1}{ET}}\right)_{before}} = \frac{(Cost * ET_{)after}}{(Cost * ET)_{before}}$$

$$= \frac{(Cost * IC * CPI * CT)_{after}}{(Cost * 0.95 * IC * CPI * CT)_{before}} = \frac{(Cost * 0.95 * IC * CPI * CT)_{after}}{(Cost * IC * CPI * CT)_{before}}$$

$$= \frac{4490 * 0.95 * 1750\ E-12}{3890 * 1450\ E-12} = 1.32$$

10. (5 points) For a pipeline with m stages, how many cycles does it take to execute n instructions?
m + n -1

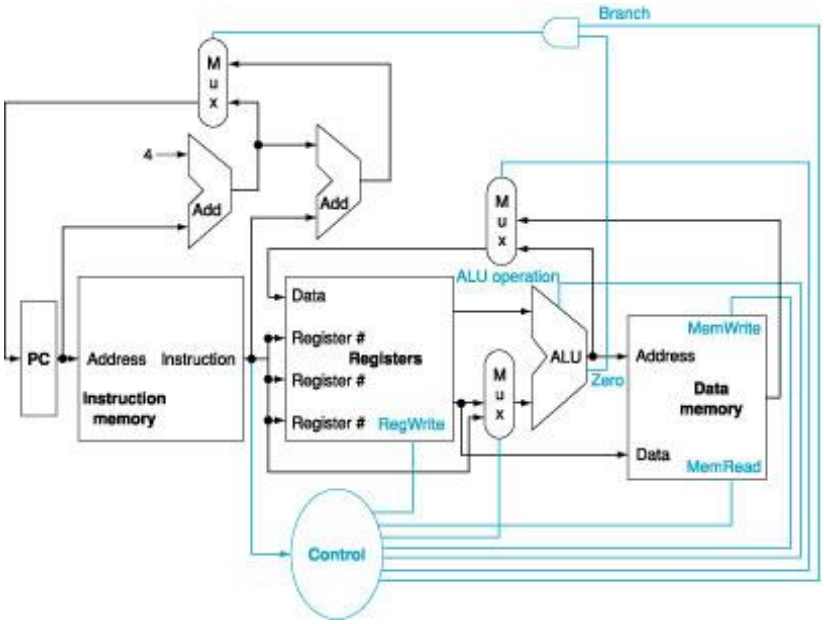
11. (10 points) What number does 0xF43B 0000 represent as an IEEE 754 floating point number?

0xF43B 0000 = 1111 0100 0011 1011 0000 0000 0000 0000
S = 1, number is negative, EXP + BIAS = 1110 1000 = 14 x 16^1 + 8x16^0 = 224 + 8 = 232, EXP = 232 -127 = 105
Fraction = 0.0111 0110 0000 0000 0000 000
Number is -(1.0111011)x2^{105}
= (1011 1011) x 2^{105} x 2^{-7}) =- 0xBB x 2^{98}
= -5.926 x 10^{31}

12. (15 points) We wish to add a variant of the `lw` (load word) instruction, which increments the index register after loading word from memory. This instruction (`l_inc`) corresponds to the following two instructions:

```
lw    $rt, L($rs)
addi  $rs, $rs, 1
```

Explain whether it is possible to implement `l_inc` without making any changes to the register file of the single-cycle datapath. If it is not possible, describe the changes that must be made to the register file.



It is not possible to implement this instruction without modifying the register file. This instruction requires that two registers be written at the same time, the one that gets data from the memory and the one that gets incremented. The current register file has two read ports and one write port, an additional write port is needed, the signals needed would be a register number, a 32 input that comes from the output of an incrementer and a control signal to control the writing on that port.