The University of Alabama in Huntsville
Electrical and Computer Engineering Department
CPE 221 01
Final Exam Solution
Spring 2016

This test is closed book, closed notes. You may use a calculator. You should have the reference packet that includes Figure 2.10 and Appendix B. You must show your work to receive full credit.

1. (1 point).C ++ is an example of a _**high-level**_ language.
2. (1 point) _**True**_ (True or False) Pipelining is a technique for improving the throughput of instruction execution.
3. (1 point) _**False**_ (True or False) Static RAM requires refreshing.
4. (1 point) _**True**_ (True or False) Most instructions take fewer cycles to execute on a 2-bus SRC than they do on a 1-bus SRC.
5. (1 point) _**Signed**_ numbers may have either positive or negative values.
6. (4 points) In an SRC computer, r2 contains a value of --3621 in decimal. What is the binary value of r1 after this instruction is executed?
   not r1, r2

   r2 = 1111_1111_1111_1111_1111_0001_1101_1011
   r1 = 0000_0000_0000_0000_0000_1110_0010_0100

7. (4 points) In an SRC computer, r2 contains a value of 1698 in decimal. What is the binary value of r1 after this instruction is executed?
   neg r1, r2

   r2 = 0000_0000_0000_0000_0000_0110_1010_0010
   r1 = 1111_1111_1111_1111_1111_1001_0101_1110

8. (2 points) In an SRC computer, r2 contains a value of -3621 in decimal while r3 contains a value of 1698 in decimal. What is the binary value of r1 after this instruction is executed?
   and r1, r2, r3

   r2 = 1111_1111_1111_1111_1111_0001_1101_1011
   r3 = 0000_0000_0000_0000_0000_0110_1010_0010
   r1 = 0000_0000_0000_0000_0000_0000_1000_0010

9. (2 points) Using the SRC, if we want to examine the last bit of a binary number to see whether it was 0 or 1, we use a mask with a value of 1 and take the logical operation and of the two operands; mask and number. What mask value would we use if we wanted to examine bit 19?

   The mask value would be 2^{19}, or 0000_0000_0000_1000_0000_0000_0000_0000

10. (10 points) Encode 001_0100_1101 using the Hamming code and odd parity. What is the final Hamming code?

```
PPDP_DDDP_DDDD_DDD
           111_111
1234_5678_9012_345
  0  010  1001_101
```

P1 = 1 ⊕ (D3 ⊕ D5 ⊕ D7 ⊕ D9 ⊕ D11 ⊕ D13 ⊕ D15) = 1 ⊕ (0 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 1) = 0
P2 = 1 ⊕ (D3 ⊕ D6 ⊕ D7 ⊕ D10 ⊕ D11 ⊕ D14 ⊕ D15) = 1 ⊕ (0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1) = 1
P4 = 1 ⊕ (D5 ⊕ D6 ⊕ D7 ⊕ D12 ⊕ D13 ⊕ D14 ⊕ D15) = 1 ⊕ (0 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 1) = 1
P8= 1 ⊕ (D9 ⊕ D10 ⊕ D11 ⊕ D12 ⊕ D13 ⊕ D14 ⊕ D15) = 1 ⊕ (1 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 1) = 1
Final Hamming Code: 0101_0101_1001_101

11. (8 points) What are the values of the following registers when the program executes "`brnz r30, r2`" for the third time? Answer in decimal.

(a) (4points) r2: _**6**_ (b) (4 points) r3:_**24**_

```
data:   .org   200
num1:   .dc    8
num2:   .dc    9
result: .dw    1
code:   .org   1000
         la    r30, again
         la    r29, done
         ld    r1, num1
         ld    r2, num2
         sub   r3, r3, r3
         brzr  r29, r1
         brzr  r29, r2
again:   add   r3, r3, r1
         addi  r2, r2, -1
         brnz  r30, r2
done:    st    r3, result
stop
```

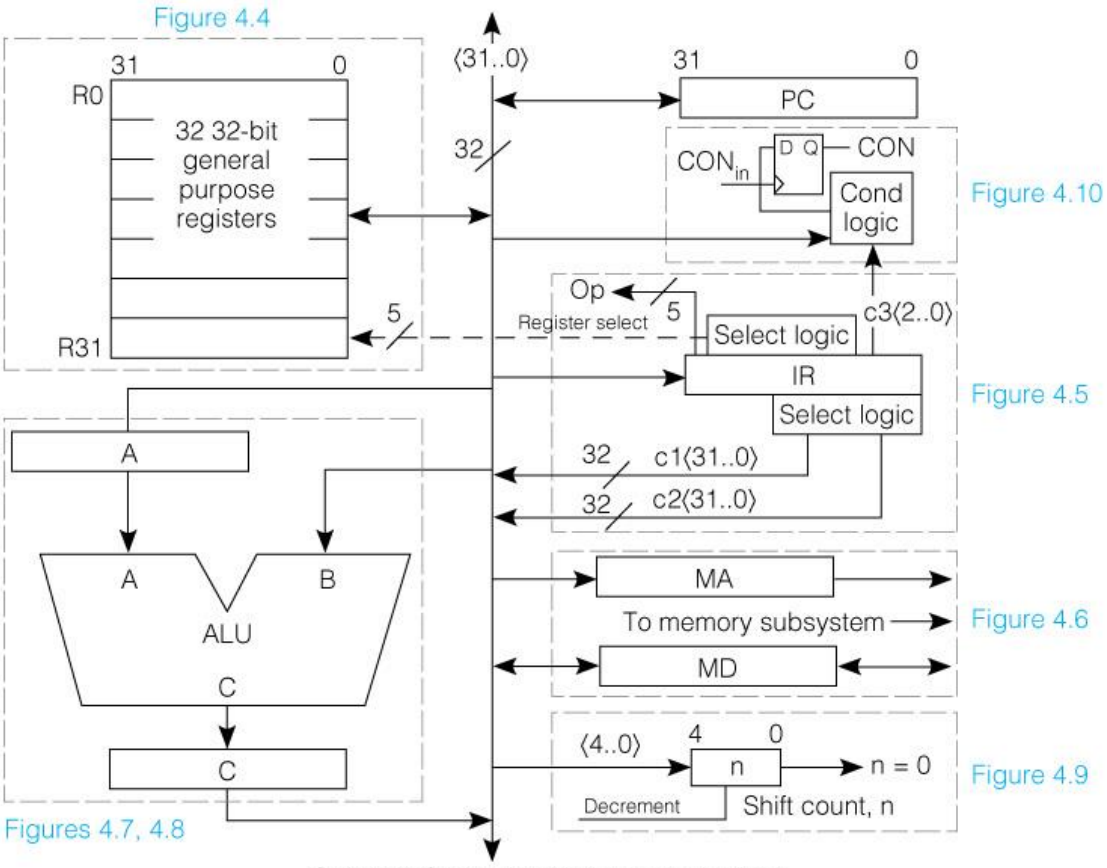
12. _**8**_ (2 points) The fields ra, rb, and rc in the SRC instruction format are 5 bits long. If the register file were enlarged to contain 256 registers, how many bits are required for each of these fields?
The number of bits required is $\log_2$ of the number of registers, $\log_2$ 256 = 8

13. (6 points) For the following pair of instructions, indicate how many bubbles must be placed between them in (a) (2 points) the absence of data forwarding, (b) (2 points) the presence of 3 to 3 forwarding only, and (c) (2 points) the presence of 4 to 3 forwarding only to resolve any dependence.

```
    ld   r2, (r4)
    st   r6, 0(r2)
```

(a) Without forwarding, the ID(2) stage of the st must come one cycle after the WB(5) stage of the ld, so 3 bubbles.
(b) With 3 to 3 forwarding, the result of ld is not available until after the ME(4) stage, so it's like there is no forwarding at all, 3 bubbles.
(c) With 4 to 3 forwarding, the result is forwarded from ME(4) to EX(3) after one bubble.

14. (15 points) Write concrete RTN steps for the SRC instruction `st` using the 1-bus SRC microarchitecture shown.



T0	MA ← PC : C ← PC + 4
T1	MD ← M[MA] : PC ← C
T2	IR ← MD
T3	**A ← (rb = 0 → 0: rb ≠ 0 → R[rb])**
T4	**C ← A + c2 {sign extend}**
T5	**MA ← C**
T6	**MD ← R[ra]**
T7	**M[MA] ← MD**

15. (10 points) A certain memory system has a 1024 MB main memory and a 64 MB cache. Blocks are 32 bytes in size. Show the fields in a memory address if the cache is 16-way set associative.

Main memory address = $\log_2$ 1024 MB = 30
Block offset = $\log_2$ 32 bytes = 5
$$64\ MB \times \frac{1\ block}{32\ bytes} \times \frac{1 set}{16\ blocks} = 2^{17}\ sets, bits\ of\ index = 17$$
bits of tag = address bits – index bits – block offset bits = 30 – 5 – 17 = 8

16. (6 points) If you want to build a 2^{48} word, 64-bits-per-word memory and the only parts you have available to you are static RAM chips that contain 2^{40} 8 bit words each. (a) (2 points) How many rows are required? (b) (2 points) How many columns are required? (c) (2 points) How many chips in all?
(a) $2^{48}/2^{40}$ = 28 or 256 rows
(b) 64/8 = 8 rows
(c) 256 x 8 = 2048 or 2^{11}

17. (6 points) Encode the lar r31, loop statement from the SRC program shown below in hexadecimal.

```
                .org 0
0       seq:    .dc 1
4       ans:    .dw cnt
           .org 1000
1000            lar   r31, loop
1004            la    r0, 8
1008            la    r1, seq
1012    loop:   ld    r2, 0(r1)
1016            ld    r3, 4(r1)
1020            add   r2, r2, r3
1024            st    r2, 8(r1)
1028            addi  r1, r1, 4
1032            addi  r0, r0, -1
1036            brnz  r31, r0
```

Instruction	op	ra	rb	rc	c1	c2	c3
lar r31, loop	6	31			8		

Instruction = 0011_0111_1100_0000_0000_0000_0000_1000 = 37C0_0008H

18. (20 points) Complete the SRC assembly language program below so that it implements the following C++ statements. You must store the calculated tax rate in the memory location pointed to by the label tax.

```
;       This program calculates a tax rate based on the value of income as
;       follows.
;       if (income < level1)
;         tax = 0;
;       else
;         if (income < level2)
;           tax = 5;
;         else
;           tax = 10;
;

          .org  200
income:   .dc   25000
level1:   .dc   15000
level2:   .dc   25000
tax:      .dw   1
orig:     .org  1000
          la    r31, done
          la    r30, five
          la    r6, tax
          ld    r7, income
          ld    r8, level1
          ld    r9, level2
          la    r10, 0
          sub   r11, r7, r8
          brmi  r31, r11
          sub   r11, r7, r9
          brmi  r30, r11
          la    r10, 10
          br    r31
five:     la    r10, 5
done:     st    r10, 0(r6)
          stop
```