The University of Alabama in Huntsville
Electrical and Computer Engineering Department
CPE 221 01
Test 1 Solution
Spring 2016

This test is closed book, closed notes. You may not 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) The instruction __format__ specifies the size and meaning of fields within the instruction.
2. (1 point) _Data transfer_ instructions move data from a memory location or register to another memory location without changing its form.
3. (1 point) _Branch_ instructions can alter the normal flow of control from executing the next instruction in sequence.
4. (1 point) A 0-address instruction uses a _stack_ to hold both operands and the result.
5. (1 point) _Direct, indirect, indexed, …_ is an example of an addressing mode found in processors.

6. (10 points) Represent 208 and -132 as signed (2s complement) 16-bit numbers

```
2     0    1              2     0    1
2     1    1              2     1    0
2     3    0              2     2    0
2     6    1              2     4    0
2    13    0              2     8    0
2    26    0              2    16    1
2    52    0              2    33    0
2   104    0              2    66    0
2   208                   2   132

208 = 0000 0000 1101 0000
132 = 0000 0000 1000 0100
-132 = 1111 1111 0111 1100
```

7. (25 points) Consider the following SRC program. Trace the values of the registers shown as they change during program execution. Also, trace the writes to memory by the `st` instruction. There may be unused columns or rows in the tables. If you need to add columns or rows, you may do so. `.dc 1` reserves one word of storage and sets it equal to 1. `.dw 3` reserves 3 words but does not give those words a value.

```
                .org 200
200     size:   .dc 6
204     a:      .dc 5
208     x:      .dc 5, 3, -1, 2, 4, 37
232     y:      .dw 10
        orig:   .org 1000
1000             la      r29, loop
1004             la      r10, x
1008             la      r11, y
1012             ld      r1, size
1016             ld      r2, a
1020    loop:    ld      r3, 0(r10)
1024             add     r3, r3, r2
1028             st      r3, 0(r11)
1032             addi    r10, r10, 4
1036             addi    r11, r11, 4
1040             addi    r1, r1, -1
1044             brnz    r29, r1
1048             stop
```

r1	**6**	**5**	**4**	**3**	**2**	**1**	**0**						
r2	**5**												
r3	**5**	**10**	**3**	**8**	**-1**	**4**	**2**	**7**	**4**	**9**	**37**	**42**	
r10	**208**	**212**		**216**		**220**		**224**		**228**		**232**	
r11	**232**	**236**		**240**		**244**		**248**		**252**		**256**	
r29	**1020**												

Results of the `st` instruction.

Memory Address	Contents
232	**10**
236	**8**
240	**4**
244	**7**
248	**9**
252	**42**

8. (15 points) Translate the selected statements from the SRC program shown below into machine code (a series of ones and zeros), then express the result in hexadecimal.

```
        .org 200
count:  .dw 1
bits:   .dw 1
num:    .dc 0
mask:   .dc 1
      orig:  .org 1000
1000             lar   r30, done
1004             lar   r29, shift
1008             lar   r28, next
1012             ld    r1, num
1016             ld    r3, mask
1020             sub   r2, r2, r2
1024             sub   r5, r5, r5
1028             brzr  r30, r1
1032  next:      and   r4, r1, r3
1036             addi  r5, r5, 1
1040             brzr  r29, r4
1044             addi  r2, r2, 1
1048  shift:     shr   r1, r1, 1
1052             brnz  r28, r1
1056  done:      st    r2, count
1060             st    r5, bits
1064             stop
```

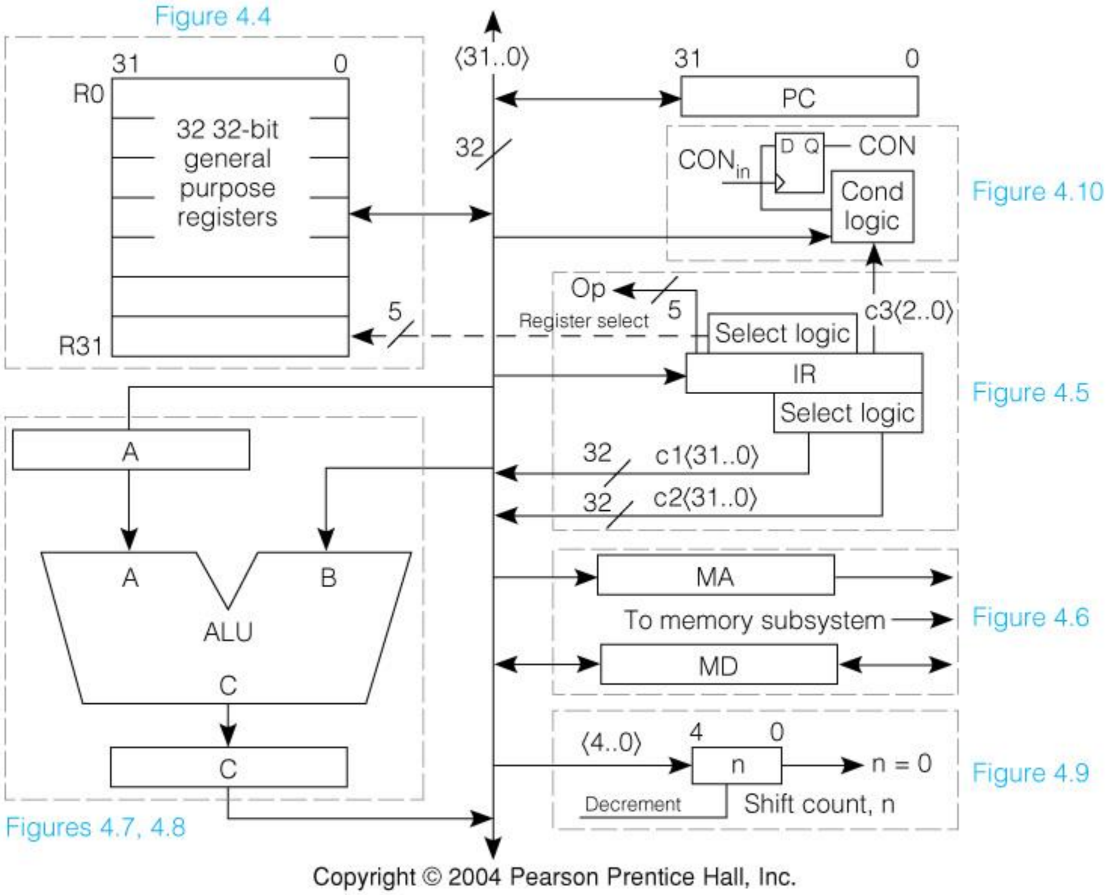
Address	Instruction	op	ra	rb	rc	c1	c2	c3	Hexadecimal
1032	and r4, r1, r3	20	4	1	3				0xA102_3000
1048	shr r1, r1, 1	26	1	1				1	0xD042_0001
1040	brzr r29, r4	8		29	4			2	0x403A_4002

```
and r4, r1, r3   10100_00100_00001_00011_000000000000 = 0xA102_3000
shr r1, r1, 1    11010_00001_00001_00000000000000001 = 0xD042_0001
brzr r29, r4     01000_00000_11101_00100_000000000010 = 0x403A_4002
```

9. (25 points) Write the code to implement the expression A = ((B/F) + (C × D)) × E on 3-, 2-, 1-, and 0-address machines. Do not rearrange the expression. In accordance with programming language practice, computing the expression should not change the values of its operands. When using a 0-address machine, the order used is SOS op TOS, where SOS is second on stack and TOS is top of stack.

3-Address	2-Address	1-Address	0-Address
div A, B, F	load A, B	load B	push B
mul T, C, D	div A, F	div F	push F
add A, A, T	load T, D	store A	div
mul A, A, e	mul T, C	load C	push D
	add A, T	mul D	push C
	mul A, E	add A	mul
		mul E	add
		store A	push E
			mul
			pop A

10. (20 points) Extend the SRC instruction set by adding the instruction `ldrr ra, rb, rc`, that is described by the abstract RTN R[ra] ← M[R[rb] + R[rc]]. (a) Write concrete RTN steps for this new instruction using the 1-bus SRC microarchitecture shown. (b) Which format would be a good one to use for `ldrr`?



Copyright © 2004 Pearson Prentice Hall, Inc.

T0	MA ← PC : C ← PC + 4
T1	MD ← M[MA] : PC ← C
T2	**IR ← MD**
T3	**A ← R[rb]**
T4	**C ← A + R[rc]**
T5	**MA ← C**
T6	**MD ← M[MA]**
T7	**R[ra] ← MD**

b. Instruction Format 6 is the closest match