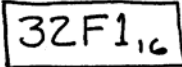Points missed: _____   Student's Name: _____

Total score: _____/100 points

East Tennessee State University
Department of Computer and Information Sciences
CSCI 4717 – Computer Architecture
TEST 3 for Fall Semester, 2005

Section 201

Read this before starting!

- The total possible score for this test is 100 points.

- This test is closed book and closed notes

- You may use one sheet of scrap paper that you will turn in with your test.

- **When possible, indicate final answers by drawing a box around them. This is to aid the grader (who might not be me!) Failure to do so might result in no credit for answer. Example:**

$$32F1_{16}$$

- **1 point will be deducted** per answer for missing or incorrect units when required. **No** assumptions will be made for hexadecimal versus decimal, so you should always include the base in your answer.

- If you perform work on the back of a page in this test, indicate that you have done so in case the need arises for partial credit to be determined.

"Fine print"

Memory Management

1. What does a "page fault" mean in terms of virtual memory? (3 points)

   When a process requests code or data from a page that has not yet been loaded in real memory requiring the operating system to load the page.

2. How does virtual memory allow programs that are larger than real memory to be run? (2 points)

   By only loading into memory the pages that a process is currently using. This means that unused pages can be left on the hard drive and therefore not take up as much memory as the full application.

3. What problem is typically caused by very small virtual memory page sizes? (2 points)

   An increase in the number of page faults because of the large number of pages that will be required to support a process.

4. What problem is typically caused by large virtual memory page sizes? (2 points)

   Large pages waste space because if only a small portion of the page is used, a large amount of code or data is loaded unnecessarily.

page number

5. Using the page table shown to the right representing a specific process, calculate the physical address from the logical address 235_{16}. Assume a page size of $2^8 = 256$. Be sure to show your work. (4 points)

   Start of page table

23	0
6A	1
F4	2
36	3
3B	4
24	5
BB	6

   Since the size of a page is 256 bytes, then the last 8 bits of a physical address ($2^8 = 256$) is the offset into a page. This means that the last 8 bits of both the physical and logical addresses represent offset. For the logical address, the bits above the lower 8 bits represent the page number in the page table. The frame address contained in the page table should replace the page number to give us the physical address.

   LOGICAL ADDRESS

2	35

   page number   lower 8 bits

   PHYSICAL ADDRESS

F4	35

   frame number   lower 8 bits
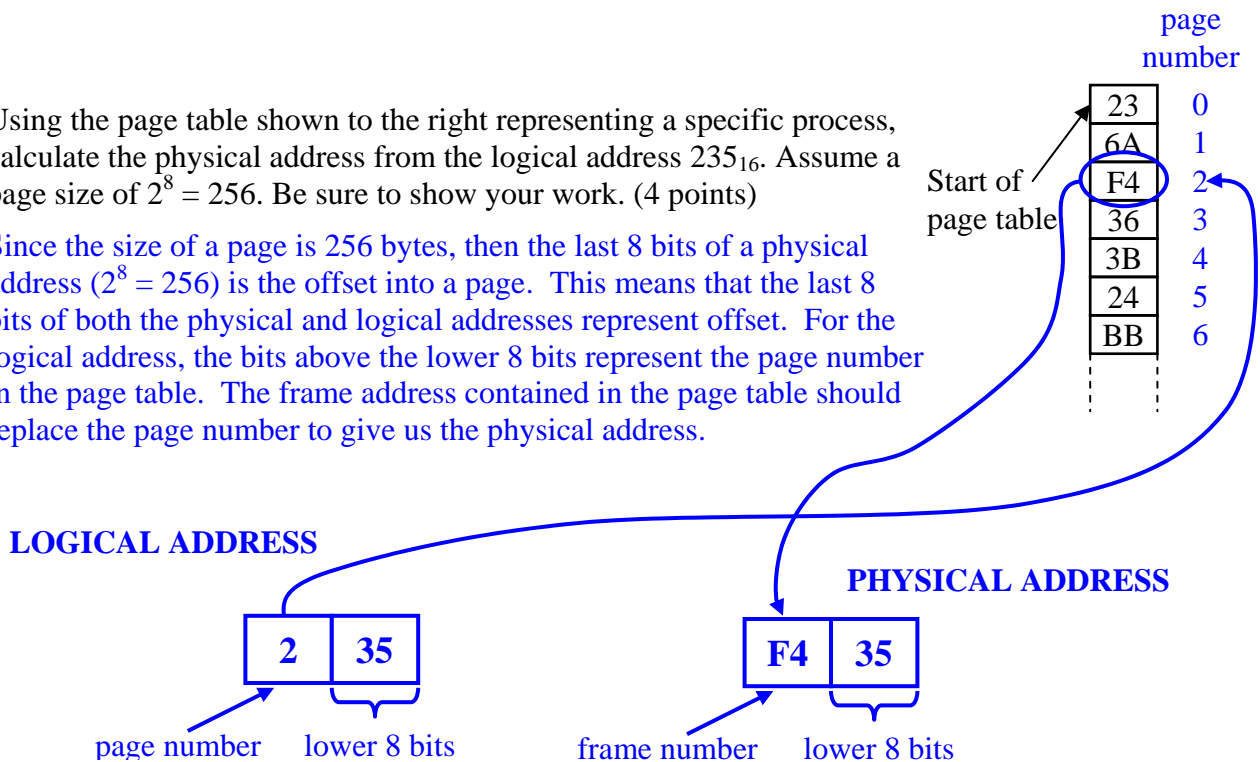
6. If 6 processes are running in paged memory with a page size of 512 bytes, what is the largest amount of memory that is being wasted? (2 points)

   A process that has been divided into pages fills all of the pages except for the last one. The last one takes the remaining bytes after dividing the process by the page size, i.e., process size modulo the page size. The smallest value this can be is *one word* which means that for this case $512 - 1 = 511$ bytes at most would be wasted per process. For six processes, this would be $511 \times 6 = 3066$.

7. (True) or false: The benefit of an inverted page table is that instead of maintaining an entry for every page in a process, it maintains an entry for every frame available in real memory. (2 points)

Assembly Language Details

8. Assume the address field of an instruction contains a decimal 105. The address field represents something different depending on the addressing mode used by the instruction. For each of the addressing modes below, describe where the processor is getting its data from. (2 points each)

    a.) Immediate: The data is the operand, i.e., 105. No fetching of operands is needed.

    b.) Indirect: The processor must go to memory address 105. At that location, it will find the data it needs to retrieve and use for the instruction.

9. Store the value $ABCDEF_{16}$ as individual bytes using big endian starting at address 100 in the figure to the right. Note that the figure will have blank locations after you are completed. (2 points)

$00FE_{16}$	
$00FF_{16}$	
0100_{16}	AB
0101_{16}	CD
0102_{16}	EF

10. The code below uses a three-operand instruction. In the space below, write three short programs that do exactly the same thing, one with two-operand instructions, one with one-operand instructions, and one with zero-operand instructions. If it fits the need of the instruction, feel free to use register names R1, R2, etc. (5 points)

```
ADD C,A,B          ;C = A + B
```

Two operand instructions	*One operand instructions*	*Zero operand instructions*
MOV C, A ADD C, B	LOAD A ADD B STORE C	PUSH A PUSH B ADD POP C

For the next 2 questions, use the following abbreviations for the stages of a 6-stage pipeline
Fetch instruction (FI)   Decode instruction (DI)   Calculate operands (CO)
Fetch operands (FO)   Execute Instruction (EI)   Write Operand (WO)

11. List all of the stages such that any 2 of them occurring at the same time could result in a bus resource conflict? (2 points)

    FI, FO, WO

12. List all of the stages that are ***always*** used in the execution of an instruction. (Hint: Think of the simplest instruction and decide which stages are needed to execute it.) (2 points)

    FI, DI, EI

Pipelines and Branch Prediction

13. In an ideal implementation, what is the speed up of a processor with a *k*-stage pipeline over a non-pipelined processor if the duration of each stage is τ? Don't consider pipeline flushes or delays incurred between stages. (2 points)

    a.) $k \cdot \tau$ b.) $k–2$ c.) $k–1$ (d.) k)   e.) $(k+1) \cdot \tau$ f.) $k+\tau–1$ g.) none of the above

14. How many bits are required for each conditional branch in a branch history table in order to remember the past 3 branch outcomes for a specific instruction? (2 points)

    a.) 2   (b.) 3)   c.) 4   d.) 5   e.) 6   f.) 7   g.) 8

15. In what level of the memory hierarchy are the branch history bits typically stored? (2 points)

    The cache

16. Identify the two problems with having two pipelines (one for the branch stream and one for the non-branch stream) in order to minimize the delay caused by an incorrectly predicted branch. (3 points)

    1.   If additional branches are encountered immediately after the first branch, we might need far more than two pipelines.
    2.   Resource contention – Additional active pipes would require multiple resources such as buses, ALUs, etc.

17. (True) or false: As far as the execution of a for-loop is concerned, the benefits of a loop buffer are only realized if the entire code executed for a for-loop can be contained in the buffer. (2 points)

18. How many cycles does prefetching the branch target save over not prefetching the branch target when a branch is incorrectly predicted? (2 points)

    a.) 0   (b.) 1)   c.) 2   d.) 3   e.) none of the above

For problems 20, 21, and 22, consider the following section of code.

```
for (i=0; i<10; i++)
{
    for (j=0; j<5; j++)
    {
        <code containing no conditional jumps>
    }
}
```

19. Once compiled, how many conditional jumps would be contained in the machine code resulting from the above section of code? (2 points)

    There would be one conditional jump to check if j reached 5 and a second to check if i reached 10. Therefore, there would be *2 conditional jumps*.

20. After fully executing the above section of code, how many conditional jumps would the CPU have encountered? (2 points)

    The processor would have encountered the conditional jump on the outer loop 10 times and on the inner loop 10 × 5 times for a total of *60 times*.

21. Using the static branch prediction algorithm "branch always," how many of the conditional jumps calculated in the previous problem would have been predicted ***incorrectly***? (2 points)

In a for-loop, branch always incorrectly predicts the branch once per full execution of the loop. For example, the conditional branch on the outer loop would have been incorrectly predicted only after i had been incremented to 10. Since the inner loop is executed 10 times and the outer loop is executed once, there will be ***11 incorrect predictions***.

22. What are the three items that should be stored in a branch history table? (3 points)

1. the address of the branch instruction itself
2. the address of the branch target, i.e., where we are going if a branch occurs
3. bits indicating the past branch history of this branch instruction

RISC Processors

23. (True) or false: The purpose behind the delayed branch is to avoid flushing the pipeline. (2 points)

24. Place a check mark next to each statement below that ***tends to be true*** for a RISC processor. (1 point each)

    ☐ RISC processors have more specialized registers rather than general purpose.

    ☑ RISC processors have a very limited number of addressing modes.

    ☑ RISC processors using 2-way pipelined timing (only two instructions can be in the pipe at any on time) do so in order to avoid two simultaneous memory accesses.

    ☑ RISC processors use a fixed instruction length.
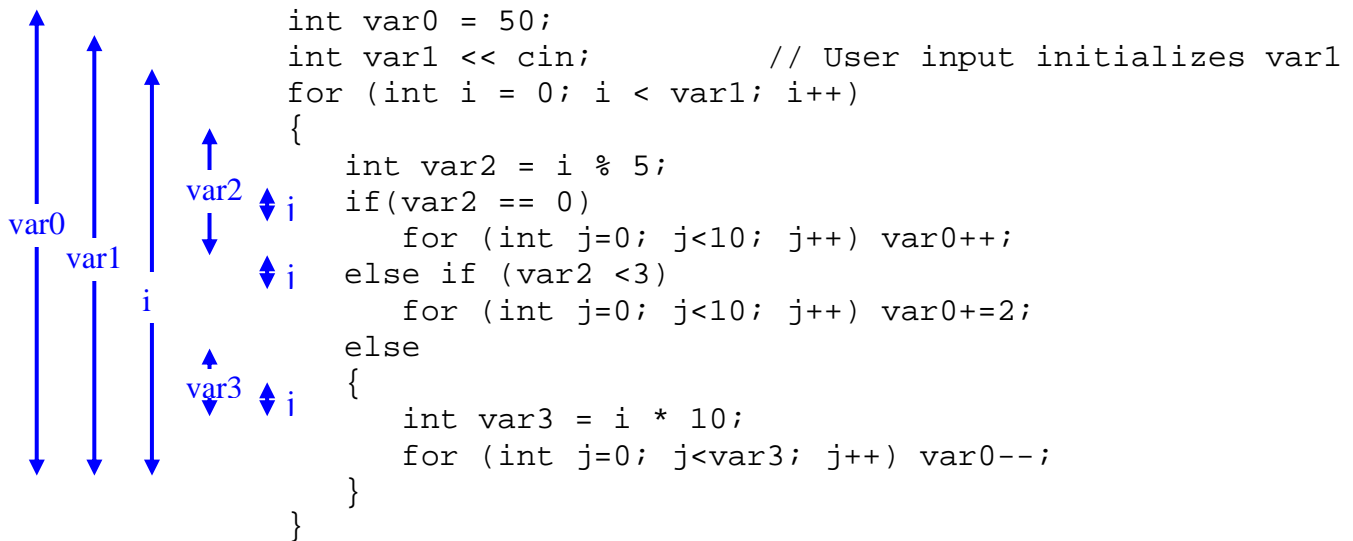
25. In order to avoid flushing the pipeline of a RISC processor, we need to modify the code to the right. If we wish to create an optimized delayed branch, which lines can be moved after L4? (2 points)

```
L1:   mov   al,bl
L2:   add   ch,23
L3:   cmp   bl,ch
L4:   jne   label_a
L5:   inc   bl
```

   a.) L1 only   b.) L2 only   c.) Either L1 or L2   d.) Neither

26. Find the absolute minimum number of registers required to execute the code below assuming that every variable in the code will need to be placed in a register. (In other words, don't try to optimize the code, just the use of registers.) (2 points)

Answer: __**5 registers**__

```
                    int var0 = 50;
                    int var1 << cin;           // User input initializes var1
                    for (int i = 0; i < var1; i++)
                    {
                        int var2 = i % 5;
                        if(var2 == 0)
                            for (int j=0; j<10; j++) var0++;
                        else if (var2 <3)
                            for (int j=0; j<10; j++) var0+=2;
                        else
                        {
                            int var3 = i * 10;
                            for (int j=0; j<var3; j++) var0--;
                        }
                    }
```

The arrows identify the periods during which each of the variables is active. By aligning arrows that do not overlap in time, we can see that a minimum of 5 columns of arrows can be created. This means that a minimum of 5 registers will be needed.

Superscalar

27. Which *two* of the following dependencies exhibit similar behavior? (2 points)

   a.) Data dependency   b.) Procedural dependency   c.) Resource conflict

28. Identify the write-read, write-write, and read-write dependencies in the instruction sequence below by entering each line pair with a dependency in the correct column of the table to the right. For example, if L1 and L4 had a write-write dependency (which they don't), you would enter L1-L4 in the column labeled "write-write". (4 points)

```
L1: R1 = R2 + R5
L2: R2 = R2 + 1
L3: R3 = R2 + R5
L4: R5 = R1 - R2
L5: R1 = R2 + 30
```

write-read	write-write	read-write
L1 – L4	**L1 – L5**	**L1 – L2**
L2 – L3		**L1 – L4**
L2 – L4		**L3 – L4**
L2 – L5		**L4 – L5**

29. In the code below, identify references to initial register values by adding the subscript 'a' to the register reference. Identify new allocations to registers with the next highest subscript and identify references to these new allocations using the same subscript. (3 points)

$$R1_b = R2_a + R5_a$$
$$R2_b = R2_a + 1$$
$$R3_b = R2_b + R5_a$$
$$R3_c = R3_b + R2_b$$
$$R5_b = R1_b \div R3_c$$
$$R1_c = R2_b \times R5_b$$

Questions 32 through 35 are based on the "in-order issue/in-order completion" execution sequence shown in the figure to the right.

Decode		Execute		Write		Cycle
I1	I2					1
I3	I4	I1		I2		2
I3	I4	I1				3
I5	I6		I3	I4	I1	I2 · 4
I5	I6	I3				5
	I6	I5		I3	I4	6
		I6				7
				I5	I6	8

30. Why do I3 and I4 need to stay in the decode stage of the pipeline for two cycles? (2 points)

Due to in-order issue, they must wait for I1 and I2 to fully execute before they can enter the execute stage. Because I1 takes two cycles to execute, they must wait in the decode stage until I1 and I2 **both** go to the write stage.

31. Why doesn't I5 get written during cycle 7 instead of cycle 8? (2 points)

Because of the in-order completion requirement, I6 must complete before both I5 and I6 can be written together. An out-of order completion would have allowed I5 to be written during cycle 7.

32. Why does I6 stay in the decode stage of the pipeline for one more cycle than I5? (2 points)

I6 uses the same resource as I5, so it must wait until I5 is finished executing.

33. Which instructions could have been written earlier if this had been an "out-of-order completion" machine. (2 points)

I2, I4, and I5

SMP and Clusters

34. Which cache coherence protocol uses distributed control, the directory protocol or snoopy protocol? (2 points)

The snoopy protocol

35. Assume a multiprocessor system uses the MESI protocol. If the current state of a line in processor A's cache is exclusive and processor B loads the same line into its cache, what does the state of that line in processor A's cache change to? (2 points)

a.) modified   b.) exclusive   c.) shared   d.) invalid   e.) cannot be determined

36. Assume a multiprocessor system uses the MESI protocol. If the current state of a line in processor A's cache is exclusive and processor A modifies that data, but ***does not*** write the new value to main memory, what does the state of that line in processor A's cache change to? (2 points)

a.) modified   b.) exclusive   c.) shared   d.) invalid   e.) cannot be determined

37. Assume a multiprocessor system uses the MESI protocol. If the current state of a line in processor A's cache is shared and processor B modifies that data without updating main memory, what does the state of that line in processor A's cache change to? (2 points)

a.) modified   b.) exclusive   c.) shared   d.) invalid   e.) cannot be determined

38. Which SMP bus configuration has the most complex printed circuit board? (2 points)

    a.) time-shared bus   **(b.) multiport memory)**   c.) central controller

39. The snoopy protocol is more suited to the _____ interconnection method for symmetric multiprocessors. (2 points)

    (a.) time-shared bus)   b.) multiport memory   c.) central controller

For the next 6 questions, identify whether the statement more closely identifies an SMP system or a cluster. (1 point each)

	SMP	Cluster	
40.	☐	☑	Processors are loosely coupled – communication usually at the file level
41.	☑	☐	Operation is closer to that of a single processor system
42.	☐	☑	The newer of the two architectures
43.	☐	☑	Easier to build, especially by end-user
44.	☑	☐	Takes less physical space
45.	☐	☑	Superior scalability