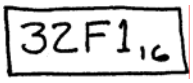
Points missed: _____ Student's Name: _____

Total score: _____ /100 points

East Tennessee State University
Department of Computer and Information Sciences
CSCI 4717 – Computer Architecture
TEST 1 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:**



- 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.

Binary	Hex
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7

Binary	Hex
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Power of 2	Equals
2^4	16
2^5	32
2^6	64
2^7	128
2^8	256
2^9	512
2^{10}	1K
2^{20}	1M
2^{30}	1G

"Fine print"

Academic Misconduct:

Section 5.7 "Academic Misconduct" of the East Tennessee State University Faculty Handbook, June 1, 2001:

"Academic misconduct will be subject to disciplinary action. Any act of dishonesty in academic work constitutes academic misconduct. This includes plagiarism, the changing of falsifying of any academic documents or materials, cheating, and the giving or receiving of unauthorized aid in tests, examinations, or other assigned school work. Penalties for academic misconduct will vary with the seriousness of the offense and may include, but are not limited to: a grade of 'F' on the work in question, a grade of 'F' of the course, reprimand, probation, suspension, and expulsion. For a second academic offense the penalty is permanent expulsion."

1. For each of the following traits of system component design, indicate which implementation method is the best, hardware (HW), software (SW), or firmware (FW), and which is the worst.

Best	Worst	Characteristic	
_SW_	_HW_	Easy to upgrade	(2 points)
_HW_	_SW_	Least likely to have a bug	(2 points)
_SW_	_HW_	Easiest to manufacture	(2 points)
_HW_	_SW_	Designing a high-speed system	(2 points)

2. Assume that you are part of a design group designing the next Palm Pilot ***hardware***.

   a.) Identify ***specific*** reasons why you might want to use a ***top-down*** design process. (2 points)
   Top down design makes it easier to attain hardware design requirements such as low power consumption, compact design (i.e., low chip count), and fast operation.

   b.) Identify ***specific*** reasons why you might want to use a ***bottom-up*** design process. (2 points)
   Bottom up design takes advantage of existing hardware technologies. This would help reduce costs and reduce design time (reach market quicker).

3. Assume that you are part of a design group designing the next Palm Pilot ***software***.

   a.) Identify ***specific*** reasons why you might want to use a ***top-down*** design process. (2 points)
   Top down design makes it easier to attain software design requirements fast operation with complex processes such as handwriting detection in addition to incorporating the latest handwriting algorithms that may not be available as shared libraries. In addition, since memory requirements are a concern, more compact code could be written.

   b.) Identify ***specific*** reasons why you might want to use a ***bottom-up*** design process. (2 points)
   Bottom up design might be necessary to make the Palm Pilot compatible with other PDA's by using existing software libraries. It will reduce design time too.

4. From the programmer's point of view, discuss one of the two effects of separating a processor's I/O space from its memory space. (3 points)

   A separate I/O space will free up memory address space for conventional memory. This does come at a cost, however. Many processors allow memory addresses to be used for all of the operand-based instructions. This might include mathematical operations along with logic operations. By moving I/O to a separate address space, these instructions will not be available to I/O, which in some cases may hamper performance.

5. What performance measurement would you suggest using to evaluate file servers? (3 points)

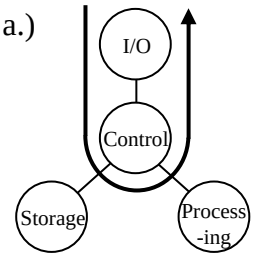
   Servers are interested in how much data they can transfer to the clients and how quickly it can be done. Therefore, any measurement identifying the ability to service I/O requests and to send data quickly would suffice. This includes throughput and I/O request rates.

6. What performance measurement would you suggest using to evaluate systems used to forecast weather? (3 points)

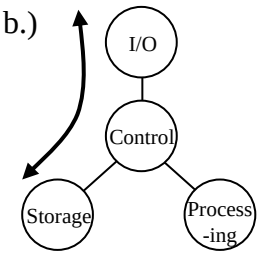
   A computationally demanding application such as this would benefit from high MFLOP rating.

For problems 7 through 10, identify which of the following system functional diagrams, a, b, c, or d, best describes the operation of the example system or application. (2 points each)

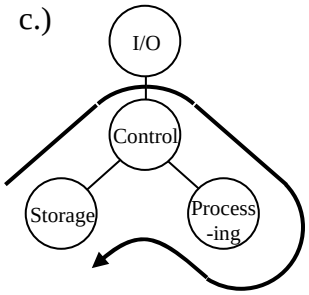
a.)



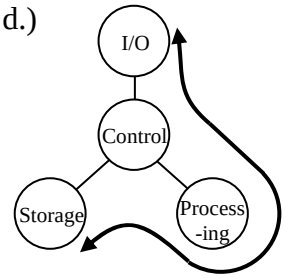
b.)



c.)



d.)



7. A network router: ___A_____

8. An MP3 player which must decompress data before playing it: ____D_____

9. A database being accessed through a web server: ____B or D____

10. Number crunching of huge matrices such as is done with weather predication: ____C_____

11. Name three of the five effects discussed in class that Moore's law has had on the contemporary application of computers. (5 points)

   – Costs have fallen dramatically since chip prices have not changed substantially since Moore made his prediction
   – Tighter packaging has allowed for shorter electrical paths and therefore faster execution
   – Smaller packaging has allowed for more applications in more environments
   – There has been a reduction in power and cooling requirements which also helps with portability
   – Solder connections are not reliable. Therefore, with more functions on a single chip, there are fewer unreliable solder connections

12. Circle ***all*** of the following components that have ***not kept pace*** with improvements in processor performance. (3 points)

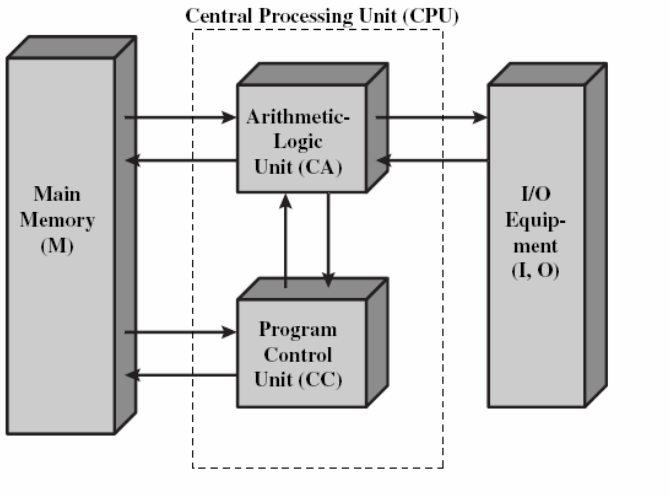
   a.) Memory speed b.) Memory capacity c.) Bus rates d.) I/O transfer rates

13. Circle ***all*** of the following traits that are identified as key concepts of the von Neumann architecture. (3 points)

   a.) A memory hierarchy to improve perceived performance
   b.) Sequential execution of code except in the case of "jumps"
   c.) Memory divided into blocks as a way to group related data or instructions
   d.) Data and instructions in a single read-write memory
   e.) Addressable memory contents without consideration to type of content
   f.) Memory mapped I/O, i.e., I/O devices use memory addresses for communication

14. In the space below, sketch the basic von Neumann architecture. Be sure to label each component and include each of the interconnections. (4 points)

Basically, this is Figure 2.1 from our text book.



15. Which is the fastest cache mapping function? (2 points)

a.) Direct mapping b.) Set associative mapping c.) Fully associative mapping

16. Which is the slowest cache mapping function? (2 points)

a.) Direct mapping b.) Set associative mapping c.) Fully associative mapping

17. Which cache mapping function is most likely to thrash, i.e., two blocks contending with each other to be stored in the same line? (2 points)

a.) Direct mapping b.) Set associative mapping c.) Fully associative mapping

18. True or False: A fully-associative cache does not require a replacement algorithm. (2 points)

Eventually, a fully-associative cache will become full. (Remember that a block can be stored anywhere in a fully-associative cache.) When it becomes full, the cache must then use a replacement algorithm to determine which lines to overwrite with a new block.

19. A split cache system uses two caches, one for _instructions_ and one for _data_. (2 points)
(fill in blank) (fill in blank)

20. Describe how the FIFO cache replacement algorithm works. (3 points)

The "First-In-First-Out" replacement algorithm replaces the block that was ***loaded*** before any of the other blocks within the same set. It doesn't make any difference how recently or how often it was used.

The table below represents a small section of a cache that uses fully associative mapping. Refer to it to answer questions 21, 22, and 23. Assume the processor's memory bus uses 20 bits for an address.

Tag (binary values)	Word within the block							
	000	001	010	011	100	101	110	111
01001001101010010	00_{16}	61_{16}	$C2_{16}$	23_{16}	84_{16}	$E5_{16}$	46_{16}	$A7_{16}$
11001100101110100	10_{16}	71_{16}	$D2_{16}$	33_{16}	94_{16}	$F5_{16}$	56_{16}	$B7_{16}$
00011101100110101	20_{16}	81_{16}	$E2_{16}$	43_{16}	$A4_{16}$	05_{16}	66_{16}	$C7_{16}$
10110011110011010	30_{16}	91_{16}	$F2_{16}$	**53_{16}**	$B4_{16}$	15_{16}	76_{16}	$D7_{16}$
01011001111001101	40_{16}	$A1_{16}$	02_{16}	63_{16}	$C4_{16}$	25_{16}	86_{16}	$E7_{16}$
01001010110010010	50_{16}	$B1_{16}$	12_{16}	73_{16}	$D4_{16}$	35_{16}	96_{16}	$F7_{16}$
	col 0	col 1	col 2	col 3	col 4	col 5	col 6	col 7

21. From what address in main memory did the value 53_{16} (the value in bold) come from? Leave your answer in binary. (3 points)

    A fully associative cache uses the bits not used to identify the word within a block as the tag to be stored on the line with the block. This means that the address breaks down like this:

20 – 3 = 17 bits	3 bits
Tag	Word id

    By taking the tag bits from the line with 53_{16} and combining it with the word position of 53_{16}, we come up with the following address.:

    "10110011110011010" + "011" = 10110011110011010011_2 = $B3CD3_{16}$

22. A copy of the data from memory address $CCBA2_{16}$ = 11001100101110100010_2 is contained in the portion of the cache shown above. What is the value stored at that address? (3 points)

    Breaking the address into its components as described in the answer for problem 21 gives us:

17 tag bits	3 word bits
11001100101110100	010

    This tag can be found in the second line. The word id of 010 identifies the value in column 2. Therefore, the value for this address is:

    $D2_{16}$

23. True or false: The number of lines contained in the fully associative cache described above can be calculated using the information given. (2 points)

    Since the number of lines in a fully associative cache has no link to the addressing, i.e., a block can be stored in any line regardless of its address, then the size of the cache cannot be computed based on the information presented above.

The table below represents a small section of a cache that uses direct mapping. Refer to it to answer questions 24 and 25. Assume the processor's memory bus uses 24 bits for an address.

Tag (binary)	Line number (binary)	Word within block				
		00	01	10	11	
1001110110	010111001001	08_{16}	19_{16}	$2A_{16}$	$3B_{16}$	Row a
1101100110	010111001010	$4C_{16}$	$5D_{16}$	$6E_{16}$	$7F_{16}$	Row b
1011010101	010111001011	50_{16}	61_{16}	72_{16}	83_{16}	Row c
0100101111	010111001100	94_{16}	$A5_{16}$	$B6_{16}$	$C7_{16}$	Row d
1110011100	010111001101	$D7_{16}$	$E9_{16}$	FA_{16}	$0B_{16}$	Row e
0110011110	010111001110	$1C_{16}$	$2D_{16}$	$3E_{16}$	$4F_{16}$	Row f
		Col 0	Col 1	Col 2	Col 3	

24. A block containing the address $27172B_{16} = 001001110001011100101011_2$ is not contained in the cache. When loaded, which row (a-f) and column (0-3) will its value be stored in? (3 points)

A direct mapping cache partitions the memory address in three parts: tag bits, line id bits, and word id bits. For a block size of 4, two word id bits are needed. From the line and tag numbers presented above, the cache uses 10 bits for a tag and 12 bits to identify the line number. This gives us a total of 24 bits, which is equal to the address size. Therefore, the figure below shows the allocation of the address bits to the different purposes of storing data within a cache.

10 bits	12 bits	2 bits
Tag	Line id	Word id

Breaking the given address into its components as described above gives us:

10 tag bits	12 line id bits	2 word id bits
0010011100	010111001010	11

This means that the data of the address $27172B_{16}$ would be stored in row b column 3.

25. What is the size of the cache in words (bytes)? (2 points)

Unlike the fully associative cache, the partitioning of the address identifies the number of lines in the cache. In this case, 12 bits means that this cache has 2^{12} = 4K lines. Combining this with the fact that a block has 4 words/bytes, the size of the cache is given to be:

$$4k \text{ lines} \times 4 \text{ words/line} = 16K \text{ words}$$

26. Assume a ***2-way set-associative cache*** is being used in a system with a 24-bit address using an 8-word block size. How many sets are there if the cache has 1024 lines, i.e., 8K words? (2 points)

Remember that a 2-way set associative cache has 2 lines/set. Therefore, if the cache has 1024 lines, there are:

$$1024 \text{ lines} \div 2 \text{ lines/set} = 512 \text{ sets}$$

Note that there's alot of information in the problem that isn't used for the answer. Sorry about that.

27. Repeat the previous problem for a ***4-way set-associative*** cache. (2 points)

A 4-way set associative cache has 4 lines/set. Therefore, if the cache has 1024 lines, there are:

$$1024 \text{ lines} \div 4 \text{ lines/set} = 256 \text{ sets}$$

Note that there's alot of information in the problem that isn't used for the answer. Sorry about that.

28. Assume a ***2-way set-associative cache*** with 1024 lines is being used in a system with a 24-bit address using an 8-word block size. If a block containing the address $7BE453_{16}$ is stored in the cache, what would the tag be? (2 points)

The address partitioning of a set associative cache is very much like that of a direct mapping cache. The difference is that the line id of a direct mapping cache is replaced with the set id for a set associative cache. Therefore, if we know the block size and the number of sets, we should be able to partition the memory address.

From problem 26, we know that this cache has 1024 lines ÷ 2 sets/line = 512 sets. Since $512 = 2^9$, then nine bits will be needed for the set id.

Since the system uses an 8-word block size, and since $8 = 2^3$, then three word id bits are needed. From these values, we can partition our 24-bit address.

12 bits	9 bits	3 bits
Tag	Set id	Word id

Breaking the address $7BE453_{16} = 011110111110010001010011_2$ into its components as described above gives us:

12 tag bits	9 set id bits	3 word id bits
011110111110	010001010	011

This means that the data of the address $7BE453_{16}$ would be stored with a tag of 011110111110_2.

29. Repeat the previous problem for a ***4-way set-associative*** cache. (2 points)

The only difference from the previous problem is that this cache has 1024 lines ÷ 4 sets/line = 256 sets. Since $256 = 2^8$, then eight bits will be needed for the set id. This leaves 13 bits for the tag. From these values, we can partition our 24-bit address.

13 bits	8 bits	3 bits
Tag	Set id	Word id

Breaking the address $7BE453_{16} = 011110111110010001010011_2$ into its components gives us:

13 tag bits	8 set id bits	3 word id bits
0111101111100	10001010	011

This means that the data of the address $7BE453_{16}$ would be stored with a tag of 0111101111100_2.

30. List one of the three implementations discussed in class for avoiding problems with writing to the cache when multiple cache-equipped CPUs share memory. Note that this is different than the two write ***policies*** we discussed. (2 points)

- Bus watching with write through – each cache watches the bus to see if data they contain is being written to the main memory by another processor. All processors must be using the write through policy.
- Hardware transparency – a "big brother" watches all caches, and upon seeing an update to any processor's cache, it updates main memory AND all of the caches.
- Noncacheable memory – Any shared memory (identified with a chip select) may not be cached.

31. Assume a cache uses a ***write through*** policy to update memory when a data element in the cache is altered. This cache updates memory: (2 points)

a.) immediately after the data element has been written to.
b.) just before the block in the cache is to be overwritten.

32. The problem with a cache that uses a ***write through*** policy is that: (circle one) (2 points)

a.) it runs the risk of leaving main memory invalid for a long period of time.
b.) it cannot be used with multiple CPUs even if the CPUs are watching main memory for an update.
c.) it slows down the write process.
d.) it requires additional overhead in the cache to keep track of which blocks have been modified.
e.) none of the above

33. Every time an address line is added to a DRAM, the ***number of addressable locations*** in that DRAM is multiplied by a factor of _____. (2 points)

Remember that a DRAM uses the incoming address lines twice for a single address: once to identify the row and a second time to identify the column of a particular memory location. This is done to save on pins. By adding a single address line, one bit has been added to the row address and one bit has been added to the column address. This in effect adds two bits to the memory address. Two bits added to a memory address quadruples the number of possible addresses since $2^2 = 4$.

a.) 1.5 b.) 2 c.) 4 d.) 8 e.) none of the above

34. There are two causes for a DRAM cell to require refreshing. List ***both*** of them. (3 points)

A cell needs to be refreshed on a regular basis due to ***leakage current*** and it must also be refreshed any time it is ***read*** from.

35. True or false: The processor may still read from the DRAM while the DRAM is being refreshed. (2 points)

False. The DRAM chip is disabled during refreshing.

36. A CDRAM uses a __cache__ on the memory stick to enhance performance. (2 points)
(fill in blank)

37. Circle ***all*** of the following statements that are true regarding SDRAM. (3 points)

a.) Access is synchronized with an external clock.
b.) The CPU is aided by the fact that it knows exactly when a data item will be available.
c.) The data element is available at the clock pulse immediately after the address is presented to SDRAM.
d.) Burst mode allows for streams of sequentially addressed words to be available without delays between them.
e.) Some forms of SDRAM allow for data elements to be sent on ***both*** the leading and trailing edges of a single clock pulse.
f.) The primary reason for the improved operation of SDRAM is its rigorously defined bus structure.