Points missed: _____ Student's Name: _____

Total score: _____/100 points

East Tennessee State University
Department of Computer and Information Sciences
CSCI 2150 (Tarnoff) – Computer Organization
TEST 3 for Fall Semester, 2005

Read this before starting!

- The total possible score for this test is 100 points.
- This test is closed book and closed notes.
- **All** answers **must** be placed in space provided. Failure to do so may result in loss of points.
- **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.
- ***Calculators are not allowed.*** Use the tables below for any conversions you may need. Leaving an answer as a numeric expression is acceptable.

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
2^3	8
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, October 21, 2005:

"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."

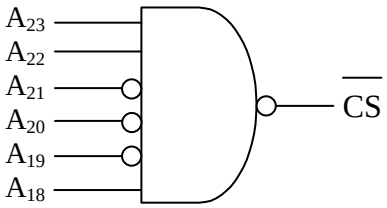
1. Match each of the settings of the bus control signals $\overline{R}$ and $\overline{W}$ on the left with the bus operation on the right. (3 points)

$\overline{R}$	$\overline{W}$	Operation of the bus
0	0	Processor reads from memory
0	1	Processor writes to memory
1	0	The bus is idle
1	1	Illegal setting

2. Circle ***all*** that apply. A storage cell in a DRAM: (4 points)

a.) is volatile
b.) is a capacitor
c.) is cheaper than cells in a SRAM
d.) is a latch
e.) must be refreshed regularly
f.) is smaller than cells in a SRAM
g.) is typically used for cache RAM
h.) is faster than an SRAM

3. What are the high and low addresses (in hexadecimal) of the memory range defined with the chip select shown to the right? (4 points)



There are 24 address lines. This is found by noting that the highest address line has a subscript of 23 and therefore, since we begin counting at 0, we know that there are 24 address lines. Looking at the inputs to the NAND gate, we see that to set ^CS to zero, their values must be: A_{23}=1, A_{22}=1, A_{21}=0, A_{20}=0, A_{19}=0, and A_{18}=1. (Inverted inputs need a zero input to put a 1 into the NAND gate.) Therefore, the address lines have the following values for the high and low address: (The shaded areas represent the bits that go into the memory device's address lines and range from all 0's for the low address to all 1's for the high address.)

Low address: $1100\ 0100\ 0000\ 0000\ 0000\ 0000_2 = C40000_{16}$
High address: $1100\ 0111\ 1111\ 1111\ 1111\ 1111_2 = C7FFFF_{16}$

4. For the chip select in problem 3, how big is the memory chip that uses this chip select? (3 points)

There are 18 address lines that go to the address inputs of the memory chip. Therefore, there are 2^{18} possible addresses. This means that the memory chip has $2^{18} = 2^8 \times 2^{10}$ = 256K memory locations.

5. For the chip select in problem 3, how big is the memory space of the processor whose address lines are used for the chip select? (3 points)

There are 24 address lines coming out of the processor. Therefore, there are 2^{24} possible addresses that the processor can address, i.e., the memory space is $2^{24} = 2^4 \times 2^{20}$ = 16Meg.

6. True or false: The address range $1AFFFF_{16}$ to $1AC000_{16}$ is a valid range for a single memory. (2 points)

   Begin by converting the low and the high addresses to binary.

   $$\text{Lo addr: } 1AC000_{16} = 0001\ 1010\ 1100\ 0000\ 0000\ 0000_2$$
   $$\text{Hi addr: } 1AFFFF_{16} = 0001\ 1010\ 1111\ 1111\ 1111\ 1111_2$$

   Since we can divide this address pair into the most significant bits defining the chip select (i.e., the bits that say constant) and the bits that go to the memory chip's address lines, (i.e., the bits that go from all zeros to all ones), then it should be possible to put a $2^{14} = 2^4 * 2^{10} = 16$ K memory between those addresses. Therefore, the answer is **TRUE**.

7. The NAND gate is used for chip selects because of its operation (it has exactly one combination of inputs that result in a low output) and its speed. What other logic gate might also work for a chip select when speed is not a factor? (2 points)

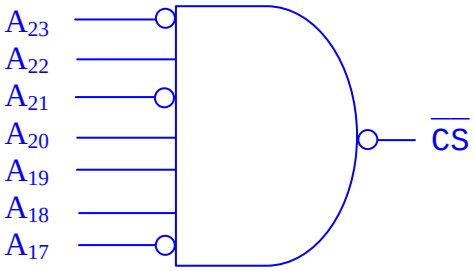
   The OR gate also has a single condition for which it had an output of 0. The difference between the circuits is that the inputs of the OR gate must be inverted from the way they would be set with the NAND.

8. Using logic gates, design an active low chip select for a RAM placed in a 16 Meg memory space with a low address of $5C0000_{16}$ and a high address of $5DFFFF_{16}$. ***Label all address lines used for chip select.*** (5 points)

   Since 16 Meg = $2^4 * 2^{20} = 2^{24}$, the processor must have 24 address lines coming out of it. (A_0 through A_{23}) Converting the high and low addresses to binary shows us where to draw the line separating the address lines that go to the chip select from the address lines that go to the memory chip.

   $$5C0000_{16} = 0101\ 1100\ 0000\ 0000\ 0000\ 0000_2$$
   $$5DFFFF_{16} = 0101\ 1101\ 1111\ 1111\ 1111\ 1111_2$$

   This shows that the lower 17 address lines (A_0 through A_{16}) go to the memory chip, and the upper 7 address lines (A_{17} through A_{23}) go to the chip select. Also from this diagram, we see that $A_{23} = 0$, $A_{22} = 1$, $A_{21} = 0$, $A_{20} = 1$, $A_{19} = 1$, $A_{18} = 1$, and $A_{17} = 0$. By inverting only the inputs that are to be recognized as zeros, we get the NAND circuit for the chip select shown to the right.

9. For each of the four following groups of information, put a check mark next to the ones for which there is enough information to correctly make the chip select logic for a memory device. (4 points)

   - [x] The high and low addresses for the memory device's address range.
   - [x] The starting (low) address and the size of the memory device.
   - [ ] The starting (low) address and the size of the processor's address space.
   - [x] The number of address lines going to the memory device, the number of address lines coming from the processor, and ***any*** valid address for that memory device.

10. What characteristic of storage devices ***improves*** as you move ***up*** through the memory hierarchy towards the processor? (2 points)

    speed

11. True or false: ***Multiple zone recording*** hard drives have more complex controllers than ***constant angular velocity*** hard drives. (2 points)

12. True or false: ***Multiple zone recording*** hard drives have better data density than ***constant angular velocity*** hard drives. (2 points)

13. By using different encoding methods, hard drive designers are able to increase _____ without changing the physical technology of the drive. (2 points)

    a.) reliability b.) data density c.) error detection d.) throughput (speed data is retrieved)
    e.) A and C b.) B and D c.) A, B, C, and D d.) none of the above

14. The number of sectors per track on a multiple zone recording hard drive _____ as you go closer to the outside edge of the disk. (2 points)

    a.) increases b.) decreases c.) stays the same

15. Describe how the LFU replacement algorithm for the fully associative mapping algorithm works. (3 points)

    When the cache needs to free up a line in order to store a new block, the least frequently used (LFU) replacement algorithm deletes/clears the line in the cache that has been accessed the least.

The table below represents a small section of a cache that uses fully associative mapping. Refer to it to answer questions 16 through 20.

Tags (binary)	Word within the block															
	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
110010110110	00_{16}	61_{16}	$C2_{16}$	13_{16}	84_{16}	$E5_{16}$	46_{16}	$A7_{16}$	12_{16}	34_{16}	56_{16}	78_{16}	$9A_{16}$	BC_{16}	DE_{16}	$F0_{16}$
010101011100	60_{16}	71_{16}	$D2_{16}$	33_{16}	94_{16}	$F5_{16}$	36_{16}	$B7_{16}$	23_{16}	45_{16}	67_{16}	89_{16}	AB_{16}	CD_{16}	EF_{16}	01_{16}
010111001011	20_{16}	81_{16}	$E2_{16}$	83_{16}	$A4_{16}$	05_{16}	66_{16}	$C7_{16}$	88_{16}	99_{16}	AA_{16}	BB_{16}	CC_{16}	DD_{16}	EE_{16}	FF_{16}
110010100110	30_{16}	91_{16}	$F2_{16}$	53_{16}	$B4_{16}$	15_{16}	$A6_{16}$	$D7_{16}$	FE_{16}	DC_{16}	BA_{16}	98_{16}	76_{16}	54_{16}	32_{16}	10_{16}
011011011001	40_{16}	**$A1_{16}$**	02_{16}	63_{16}	$C4_{16}$	25_{16}	86_{16}	$E7_{16}$	ED_{16}	CB_{16}	$A9_{16}$	87_{16}	65_{16}	43_{16}	21_{16}	$0F_{16}$
110010100101	50_{16}	$B1_{16}$	22_{16}	73_{16}	$D4_{16}$	35_{16}	96_{16}	$F7_{16}$	11_{16}	44_{16}	55_{16}	77_{16}	$0F_{16}$	$1F_{16}$	$2F_{16}$	$3F_{16}$

16. Assuming the tags shown above do ***not*** delete leading zeros, how many address lines does the processor that uses this cache have? (2 points)

    The processor address is broken into two parts when storing using fully associative mapping. The lower bits identify the word position within the block while the remaining upper bits become the tag. Therefore, since the tag has 12 bits and the word id has 4 bits, the processor's address has 16 bits.

17. What is the block size (in number of memory locations) for the cache shown above? (2 points)

The block size is determined by the number of bits in the word id, i.e., how many words are in a block. Since there are four bits used for the word id, the number of unique word id's for a block is $2^4 = 16$. This makes sense since there are 16 columns in the table in which to store a block.

18. From what address in main memory did the value $A1_{16}$ (the value in bold) come from? Leave your answer in binary. (3 points)

Fully associative mapping divides the physical address into two pieces, the tag and the word id. The word id is the last n-bits of the address where memory is divided into blocks of size 2^n. In case of the fully associative cache shown above, n=4. Since the value has a tag of 011011011001_2 and a word id of 0001_2, then the physical address is 0110110110010001_2 **= $\mathbf{6D91_{16}}$**.

19. A copy of the data from memory address $CA5B_{16}$ is contained in the portion of the cache shown above. What is the value stored at that address? (2 points)

Dividing the physical address $CA5B_{16} = 1100101001011011_2$ into its tag and 4-bit word id gives us a tag of 110010100101_2 and a word id of 1011_2. Searching through the visible lines shows us that the bottom line has the same tag, i.e., it contains the block which contains the data from the physical address $CA5B_{16}$. A word id of 1011_2 points us to the data in the twelfth column, i.e., the value of **$\mathbf{77_{16}}$**.

20. If the block containing memory address 4648_{16} were to be loaded into the cache described above, what would the tag be? (2 points)

Dividing the physical address $4648_{16} = 0100011001001000_2$ into its tag and 4-bit word id **gives us a tag of $\mathbf{010001100100_2}$** and a word id of 1000_2.

21. True or false: The method used to make a chip select for a memory device is the same as that used to identify a subnet ID in a TCP/IP network or to identify a block ID in a memory space. (2 points)

22. Assume a processor takes 3 cycles to execute any instruction (fetch, decode, execute)

a. How many cycles would a ***non-pipelined*** processor take to execute 7 instructions? (2 points)

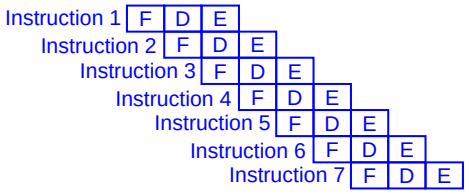
A non-pipelined processor simply executes the instructions one at a time with no overlap. Therefore, the number of cycles equals 3 cycles/instruction times the number of instructions:

number of cycles = $3 \times 7 = 21$

b. How many cycles would a ***pipelined*** processor take to execute 7 instructions? (2 points)

A 3-stage pipelined processor overlaps 2 cycles for each instruction as shown in the figure below.

Therefore, it will take 2 cycles to fill the pipe, then one cycle to execute each instruction. This means 2 cycles to fill the pipeline plus 7 cycles to execute the 7 instructions.



number of cycles = 2 + 7 = 9 cycles

23. What are the settings of the zero flag, the sign flag, the carry flag, and the overflow flag after a processor performs the addition shown to the right? (4 points)

```
1 111 1
  01010101
+ 10110110
  00001011
```

ZF = ___**0**__ SF = ___**0**__ CF = ___**1**__ OF = ___**0**__

24. What mathematical operation does a processor use to compare two values to see if they are equal or to see if one is greater than the other? (2 points)

Answer: __subtraction_____

25. What is the purpose of the ALU? (2 points)

The arithmetic logic unit (ALU) performs all of the mathematical and logical operations required of the central processing unit (CPU). It's kind of like the calculator for the processor.

26. Name the two benefits of the segment/pointer addressing system of the 80x86. (3 points)

- It allows us to use 16 bit registers (word-length) to access larger (20-bit) address spaces
- It allows for re-locatable code in memory

27. Assume AX=1000_{16}, BX=2000_{16}, and CX=3000_{16}. After the following code is executed, what would AX, BX, and CX contain? (3 points)

```
PUSH AX
PUSH BX
PUSH CX
POP BX
POP CX
POP AX
```

Place your answers in space below:

AX = **old AX = 1000_{16}**

BX = **old CX = 3000_{16}**

CX = **old BX = 2000_{16}**

28. What is the physical address pointed to by the 80x86 segment/pointer pair 3200:1234? Note that the values given are in hexadecimal. (2 points)

The first number represents the segment (3200_{16}) and the second number represents the pointer or offset (1234_{16}). To figure out the physical address, begin by converting the 16-bit segment to the 20-bit segment address by adding a hex 0 to the end of the segment value (4 binary 0's). This is the same as multiplying by 16.

3200_{16} → the segment address is $3200\mathbf{0}_{16}$ (notice the added zero)

The pointer or offset value can then be added as an offset to the segment address.

```
 32000
+ 1234
-------
 33234
```

Therefore, the physical address pointed to by 3200:1234 is **33234_{16}**.

29. Which 80x86 segment/pointer register pair points to the next instruction to be executed by the processor? (2 points)

a.) es:sp b.) es:di c.) ss:bp d.) ds:si e.) cs:ip f.) ds:ip g.) cs:di h.) ss:sp

30. Which 80x86 segment/pointer register pair points to where the arguments for a function or procedure are stored on the stack? (2 points)

a.) es:sp b.) es:di c.) ss:bp d.) ds:si e.) cs:ip f.) ds:ip g.) cs:di h.) ss:sp

31. Using an original value of 10011001_2 and a mask of 00001111_2, calculate the results of a bitwise AND, a bitwise OR, and a bitwise XOR for these values. (2 points each)

Original value	Bitwise operation	Mask	Result
10011001_2	AND	00001111_2	**00001001_2**
10011001_2	OR	00001111_2	**10011111_2**
10011001_2	XOR	00001111_2	**10010110_2**

32. If the datasum calculated from a sequence of values is a hexadecimal $5A_{16}$, what would the hexadecimal value of the 1's complement checksum be? (2 points)

The 1's complement checksum value is the 1's complement (bitwise inverse) of the datasum. Therefore, the 1's complement checksum would be the bitwise inverse of $5A_{16} = 01011010_2$ which equals $10100101_2 = A5_{16}$.

33. Describe the primary drawback discussed in class of a datasum-based checksum. (2 points)

The problem with a datasum-based checksum is that an error in the data only produces a small change in the checksum. This change can easily be canceled by a counteracting error in the data. Therefore, two errors could look like no errors.

34. Describe one of the two reasons discussed in class for using an XOR "borrow-less" subtraction in the calculation of a CRC. (2 points)

The goal of calculating a CRC is to "simulate" the mod or remainder function. We don't care that the value isn't numerically correct. This allows us to use the borrowless subtraction. Using a borrowless XOR subtraction is much faster than trying to do long division with a normal subtraction. Also, it does not require that the entire dividend (i.e., the block of data on which the CRC is being calculated) be contained in a single register. We only need to hold a small portion or a window of the data block equivalent to 1-bit less than the divisor.

35. True or false: When using a CRC for error checking, both the transmitting device and the receiving device must use the same polynomial (divisor) for calculation of the CRC. (2 points)

36. For each of the following statements, identify whether it describes an Ethernet frame (E), an IP packet (I), or a TCP packet (T) by placing the corresponding letter (E, I, or T) in the space provided. (6 points)

__I__ This protocol is used to get a message from one host to another across multiple interconnected networks.

__E__ This protocol uses a preamble of alternating 1's and 0's to synchronize all receivers.

__E__ This protocol uses a CRC instead of a datasum-based checksum.

__I__ This protocol includes a "time to live" field so that it can be removed from the network(s) in case it cannot find its destination.

__I__ Uses a logical address defined by a network administrator for its addressing.

__E__ Uses a physical address on the network interface hardware for its addressing.