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 Spring Semester, 2009

Read this before starting!

- The total possible score for this test is 100 points.
- All problems are worth 2 points unless otherwise noted.
- This test is ***closed book and closed notes.***
- ***Please turn off all cell phones & pagers during the test.***
- **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}	1 kilo (K)
2^{20}	1 mega (M)
2^{30}	1 giga (G)
2^{40}	1 tera (T)
2^{50}	1 peta (P)

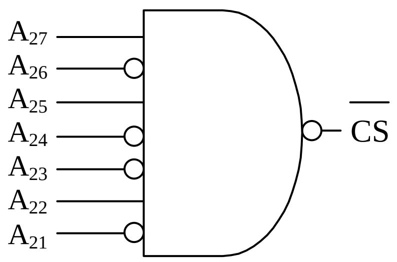
1. True or [✓ false:] All memory devices connected to a processor's system bus must be the same size due to chip select circuitry constraints. (1 point)

   The only requirements for creating a chip select are that you must select an address range for which (1) there is a clear division between the chip select bits and the memory address lines and (2) that the address range cannot overlap any other address range. As long as these two requirements are met, (and that the size of the address range is less than the processor's address space) the size of the memory chip doesn't matter.

2. How many 1-bit storage cells, e.g., D-latches, are contained in a memory that has 16 address lines and 8 data lines?

   An SRAM with 16 address lines has $2^{16} = 2^{6} \times 2^{10}$ = 64K memory locations. Since we know that there are 8 data bits per location because of the 8 data lines, then this 64K SRAM contains $2^{16} \times 8 = 2^{16} \times 2^{3} = 2^{19}$ = 512K latches. (Note: you are not responsible for any of the mathematical calculations. Simply putting $2^{16} \times 8$ would have been sufficient.)

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 28 address lines coming from the processor. We know this because the highest address line has a subscript of 27, and since we begin numbering the address lines from 0, we know that there are 28 address lines. Address lines 21 to 27 go to the chip select circuitry while the remaining twenty-one address lines, 0 through 20, go to the address inputs of the memory device.

   Looking at the inputs to the NAND gate, we see that to set ^CS to zero, their values must be: A_{27}=1, A_{26}=0, A_{25}=1, A_{24}=0, A_{23}=0, A_{22}=1, and A_{21}=0. (Inverted inputs need a zero input in order to send a 1 into the NAND gate.) Therefore, the address lines have the following values for the high and low address. (Note that 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:  1010 0100 0000 0000 0000 0000 0000₂ = A400000₁₆
High address: 1010 0101 1111 1111 1111 1111 1111₂ = A5FFFFF₁₆
```

   Low address: ____________________ High address: ________________________

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

   There are 21 address lines that go to the address inputs of the memory chip. Therefore, there are 2^{21} possible addresses. This means that the memory chip has $2^{21} = 2^{1} \times 2^{20}$ = ***2 Meg 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)

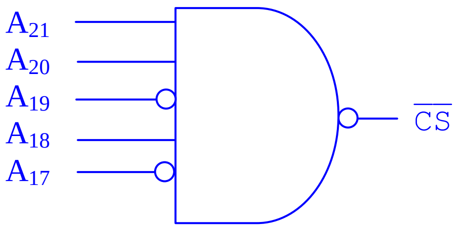
   There are 28 address lines coming out of the processor. Therefore, there are 2^{28} possible addresses that the processor can address, i.e., the memory space is $2^{28} = 2^{8} \times 2^{20}$ = ***256 Meg.***

6. Using logic gates, design an active low chip select for a memory device placed in a ***4 Meg*** memory space with a low address of $3C0000_{16}$ and a high address of $3DFFFF_{16}$. ***Label all address lines used for chip select.*** (5 points)

   Since 4 Meg = $2^2 \times 2^{20} = 2^{22}$, the processor must have 22 address lines coming out of it in order to support a 4 Meg memory space. (A_0 through A_{21}). Therefore, our addresses must represent all twenty-two bits. Converting the high and low addresses 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.

```
3C0000₁₆  =   0011 1100 0000 0000 0000 0000₂
3DFFFF₁₆  =   0011 1101 1111 1111 1111 1111₂
```

   Note that the conversion resulted in 24 bits, so the most significant two bits (in fuchsia) must be discarded because the processor doesn't actually have these bits. The processor's bits only go up to A_{21}.



   The low and high conversions show that the lower 17 address lines (A_0 through A_{16}) go to the memory chip, and the upper 5 address lines (A_{17} through A_{21}) go to the chip select. Also from this diagram, we see that $A_{21} = 1$, $A_{20} = 1$, $A_{19} = 1$, $A_{18} = 1$, and $A_{17} = 0$. By inverting A_{17}, the only input that is to be recognized as zero, we get the NAND circuit for the chip select shown to the right.

7. A 16K memory can have a starting (lowest) address of $93C000_{16}$ in a processor's memory space.

   [✓ a.) True] b.) False c.) Not enough information given

   To begin with, 16K = $2^4 \times 2^{10} = 2^{14}$. This means that a 16K memory requires 14 address lines. This means that the 14 least significant bits of the starting address must be zeros.

   $93C000_{16} = 1001\ 0011\ 1100\ 0000\ 0000\ 0000_2$

   Exactly 14 of the least significant bits in this case equal zero which means that a 16K memory is actually the largest possible memory that can start at this address.

8. A chip select circuit for a single memory device can be designed for the address range $5D000_{16}$ to $5EFFF_{16}$.

   a.) True [✓ b.) False] c.) Not enough information given

   Remember that to design a chip select, you need to know three things:

   1.) the total number of address lines from the processor;
   2.) the number of address lines going to the memory, which in turn tells us how many address lines are left to go to the chip select; and
   3.) the fixed, binary values of the bits that define the chip select lines.

   So let's see if we can determine all three of these requirements. Let's start by trying to design the chip select. Begin by converting the starting and ending addresses to binary.

```
5D000₁₆ = 0101 1101 0000 0000 0000₂
5EFFF₁₆ = 0101 1110 1111 1111 1111₂
```

Now try to draw the vertical line that separates the chip select bits (the ones that remain constant) from the memory bits (the ones that go from all zeros to all ones). This cannot be done. If address bit A_{13} had been zero and address bit A_{12} had been a one in the high address, this might have been possible. At that point, we still would need to have known the total number of processor address lines which we aren't given. Unfortunately for this case, the answer is **FALSE.**

9. A chip select circuit for a 256K memory device placed inside of a 1 Meg processor memory space can be designed knowing only that one of the addresses inside of its range is 43564_{16}.

   [✓ a.) True] b.) False c.) Not enough information given

   Once again, can we find the three things needed to make this circuit? Let's begin with trying to determine the number of address lines coming out of the processor. We know the processor's memory space, and therefore can determine the number of address lines.

   1 Meg = 2^{20} → the processor has 20 address lines

   Next, can we determine the number of address lines to the memory? We know the size of the memory, and therefore we can determine the number of address lines it needs.

   $256K = 2^8 \times 2^{10} = 2^{18}$ → the memory needs 18 address lines

   This means that the two most significant bits go to the chip select. Since we know one of the addresses that goes to this memory, we can determine the two most significant bits.

   $43564_{16} = 0100\ 0011\ 0101\ 0110\ 0100_2$ → $A_{19} = 0$ and $A_{18} = 1$

   We can, in fact, design this chip select.

10. Name a characteristic of memory that ***improves*** as you move down in the memory hierarchy, i.e., father from the processor.

    The primary characteristics that improve as you move away from the processor are capacity and cost per bit of storage.

11. Identify ***all*** of the following statements that are true regarding a hard drive's read/write head? (3 points)

    [✓ a.)] The farther a hard drive's write head is from the platters, the larger the write head gap must be in order to reliably write to the disks.

    [✓ b.)] For a specific encoding method, the minimum space reserved for a single bit on a hard drive is directly proportional to the gap in the write head. (But not necessarily equal to.)

    [✓ c.)] By modifying the hard drive controller to use a different encoding method, the data density on the platters can be increased with no modification to the write head or the platters.

    [✓ d.)] A Winchester head was developed to have the head float on a cushion of air so that it keeps a more consistent distance from the hard drive's platters.

12. Which one of the following measurements used to calculate hard drive data access times is most accurately predictable?

    a.) queuing time   b.) rotational latency   [✓ c.)] transfer time   d.) seek time

13. The time to read a fragmented file from a hard drive is greatly increased over a non-fragmented file because of the numerous delays due to ________________. Circle all that apply.

    a.) queuing time   [✓ b.)] rotational latency   c.) transfer time   [✓ d.)] seek time

14. The speed at which the data is read from the platters of a ***multiple zone recording*** hard drive ______________ as you go toward the outside tracks of a hard disk.

    [✓ a.)] increases   b.) decreases   c.) stays the same   d.) cannot be predicted

15. [✓ True] or false: One drawback of ***multiple zone recording*** hard drives is that they require more complex controllers capable of much higher data rates. (1 point)

16. If the word ID used by the cache of a processor with a 1 Gig memory space uses 3 bits, then the number of words in a block equals ____________?

    a.) 2^2 [✓ b.)] 2^3 c.) 2^4 d.) 2^{27} e.) 2^{30} f.) cannot be determined

    Remember that the size of a block is equal to the number of patterns of 1's and 0's that are possible using the word ID bits, i.e., the bits not contained within the block ID. Since there are 3 word ID bits, then there are 2^3 possible patterns of 1's and 0's, and hence a block contains 8 words.

The table below represents a small section of a cache using fully associative mapping. Each tag and word ID is in binary while the data is in hexadecimal. Refer to it to answer questions 16 through 19.

Tags	Word within the block															
(binary)	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
1100011111010000	00	61	C2	13	84	E5	46	A7	12	34	56	78	9A	BC	DE	F0
0101100110001110	60	71	D2	33	94	F5	36	B7	23	45	67	89	AB	CD	EF	01
1101001101000011	20	81	E2	83	A4	05	66	C7	88	99	AA	BB	CC	DD	EE	FF
0101100111100110	30	91	F2	53	B4	15	A6	D7	FE	DC	BA	98	76	54	32	10
1001011011010110	40	A1	02	63	C4	25	86	E7	ED	CB	**A9**	87	65	43	21	0F
1101100011011001	50	B1	22	73	D4	35	96	F7	11	44	55	77	0F	1F	2F	3F
Column →	a	b	c	d	e	f	g	h	i	j	k	l	M	n	o	p

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

    The tag is 16 bits, and in fully associative mapping, the tag equals the block ID. The word ID is 4 bits. Since the full address is made from the combined block ID and word ID, then the number of address lines the processor has is 16 + 4 = 20 bits.

18. How many word ID bits does this cache use to define a block size?

    The number of words contained within a block of this cache is 16. This comes from the 16 patterns of 1's and 0's possible with the 4 word ID bits. The answer is 4.

19. From what address in main memory did the value $A9_{16}$ (the value in the fifth row, column k) come from? Leave your answer in binary.

Fully associative mapping divides the physical address into two pieces, the block ID (which in fully associative mapping is used as the tag) and the word ID. In this case, the word id is the last 4 bits of the address. Since the $A9_{16}$ has a tag of 1001011011010110_2 and a word id of 1010_2, then the physical address is 10010110110101101010_2 or $96D6A_{16}$.

20. A copy of the data from main memory address $D3438_{16}$ is contained in the portion of the cache shown above. What is the value stored at that address? (3 points)

Dividing the physical address $D3438_{16} = 11010011010000111000_2$ into its block ID and 4-bit word ID gives us a tag of 1101001101000011_2 and a word id of 1000_2. Searching through the visible lines shows us that the third line has the same tag, i.e., it contains the block which contains the data from the physical address $D3438_{16}$. A word id of 1000_2 points us to the data in column i. This means that the value stored at $D3438_{16}$ and copied into the cache is $\mathbf{88_{16}}$.

21. Which of the following statements are true regarding the relationship between a processor's L1 and L2 caches? Circle all that apply. (4 points)

a.) One of them contains code while the other contains data.
[✓ b.)] L1 is smaller and faster than L2
[✓ c.)] L1 is always inside the processor chip while L2 may or may not be inside the processor chip.
[✓ d.)] Blocks contained in L1 must also be contained in L2.

22. When trying to load a new block into a fully associative cache that is full, the FIFO cache replacement algorithm replaces the block in the cache that has been in the cache the longest. What is one of the drawbacks of this type of replacement algorithm? (3 points)

The first items to be loaded are probably the operating system or driver software. Just because something is loaded first doesn't mean that it won't be used continuously.

There is a second problem too. A FIFO algorithm requires that a timer or counter of some sort be added to each line of the cache to determine which one has been in the longest. This creates slightly more complicated circuitry than some of the other methods.

23. What is the most common reason for a processor's instruction pipeline to be flushed?

A conditional branch, i.e., any part of the code where one of two different paths might be taken depending on the outcome of a condition, could flush the pipe. These include if-statements, while- or for-loops, switch-case blocks, etc. Interrupts, since they are unexpected changes in the stream of executable instructions, also flush the pipe.

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

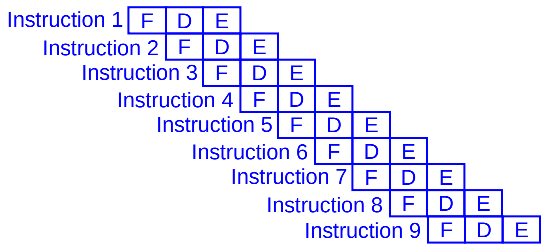
a. How many cycles would a ***non-pipelined*** processor take to execute 9 instructions? (2 points)

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:

$$\boxed{\text{number of cycles} = 3 \times 9 = 27 \text{ cycles}}$$

b. How many cycles would a ***pipelined*** processor take to execute 9 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 with instructions, then one cycle per instruction to execute the 9 instructions.

$$\boxed{\text{number of cycles} = 2 + 9 = 11 \text{ cycles}}$$

You could also just count the cycles shown in the diagram above.

25. What are the settings of the zero flag, sign flag, carry flag, overflow flag, and parity flag after a processor performs the addition shown to the right? Remember to treat the parity flag as the Intel x86 processors would. (5 points)

```
1  11111
   10110101
+  11111100
   --------
   10110001
```

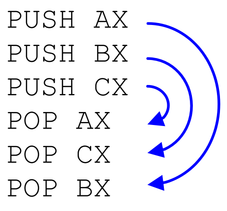
ZF = ___**0**___ SF = ___**1**___ CF = ___**1**___ OF = ___**0**___ PF = ___**0**___

The zero flag (ZF) is set to a 1 only if the result equals 0, which in this case it does not. The sign flag (SF) is set to a one for negative results and follows the most significant bit of the result, which in this case is a 1. The carry flag (CF) contains the carry out of the most significant bit of the addition. This addition did result in a carry, so CF=1. The overflow flag (OF) is set to one when there is a two's complement overflow, i.e., two positive numbers are added together resulting in a negative number or two negative numbers are added together resulting in a positive number. There is not a 2's complement overflow in this addition (we added a negative to a negative and got a negative) so OF=0. Lastly, the parity flag (PF) is set to a 1 if the number of ones in the resulting binary value is odd. There are four ones in the result for this case, so the parity flag contains a 0.

26. Which of the following four components of the CPU ***does not*** have a direct connection to the CPU's internal data bus. (Select only one.)

[✓ a.)] Control Unit b.) Registers c.) Instruction Decoder d.) ALU

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 AX
POP CX
POP BX
```

Place your answers in space below:

AX = $\mathbf{3000_{16}}$

BX = $\mathbf{1000_{16}}$

CX = $\mathbf{2000_{16}}$

28. Three uses were discussed in class for the stack. Describe one.

The stack is used to:

- temporarily store register values so that the registers may be used for another purpose
- pass parameters to a method or function
- store the address to return to after completing a method or function

29. Assume we want to take the negative of an 8-bit signed magnitude value by inverting the most significant bit (the sign bit). Which bitwise operation could be used to do this?

a.) AND b.) OR [✓ c.) XOR] d.) This function is not possible with a bitwise operation

30. What would the mask look like for the previous problem?

[✓ a.) 10000000_2] b.) 01111111_2 c.) As I said before, this function isn't possible

31. Using an original value of 11000011_2 and a mask of 11110000_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
11000011_2	AND	11110000_2	**11000000_2**
11000011_2	OR	11110000_2	**11110011_2**
11000011_2	XOR	11110000_2	**00110011_2**

32. There were two reasons given in class for using a "borrow-less" subtraction to simulate a long division when generating a CRC. Name one.

- If a standard long division is used, the whole value being divided must be contained in a single register in order to handle the possible borrows from higher order bits. This is impossible with values (packets) containing thousands of bits.
- A long division performed using a borrowless subtract is much faster.

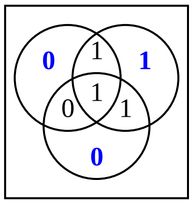
33. The borrow-less subtraction is performed using which bitwise operation?

a.) AND b.) OR [✓ c.) XOR] d.) This function is not performed with a bitwise operation

34. For each of the following statements, place a checkmark in the column(s) identifying which protocol(s) the statement describes. Some statements have more than one checkmark. (8 points)

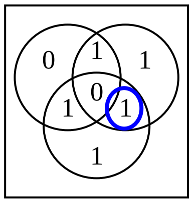
Ethernet	IP	TCP	
☒	☐	☐	Has a preamble to synchronize the receiving network interface cards (NICs)
☒	☐	☐	Uses 6-byte MAC addresses to uniquely identify NICs on the network
☒	☐	☐	Uses a CRC to check for errors across the whole frame
☐	☐	☒	Identifies the packet's purpose by using ports such as port 80 (http)
☐	☒	☐	Uses 4-byte logical addresses instead of physical addresses (Version 4)

35. Add the parity/check bits that are missing from the graphic shown to the right.



The sum of 1's inside of each circle must equal an even number. Therefore, just place a 1 or a 0 in the blank portion of each circle to make sure that that circle has an even number of 1's.

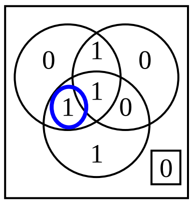
36. The graphic to the right depicts the digits of a 4 data-bit/3 check-bit Hamming code where a single bit error has occurred. Circle the bit that has flipped.



The upper left circle is the only one that contains an even number of 1's. Therefore, the error must be in the region that is contained in both the upper right and the lower circles, but not in the upper left circle. The only bit that satisfies this requirement is the one that is circled in the diagram. It needs to be flipped to a 0 in order to correct the data.

37. Which of the following does the graphic to the right depict?

a.) No error
b.) A single bit error
c.) Double bit errors



The first step toward solving this problem is to go ahead and attempt to correct the error by first checking the circles to see which ones have an incorrect parity. The top left and the bottom circles have incorrect parity while the top right one is okay. Therefore, if there is a single-bit error, it is contained in the left circle and the bottom circle, but not the right circle. This bit is circled in the diagram.

The next step is to verify that after correcting the bit, the parity in the whole diagram including the bit in the little box in the lower, right corner is correct. After flipping the bit we thought was incorrect from a 1 to a zero, the whole diagram now has three bits, which is an odd number. Therefore, there were two bit flips and we cannot correct them.