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, 2007

Read this before starting!

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

"Fine print"

Academic Misconduct:

Section 5.7 "Academic Misconduct" of the East Tennessee State University Faculty Handbook, October 21, 2005:

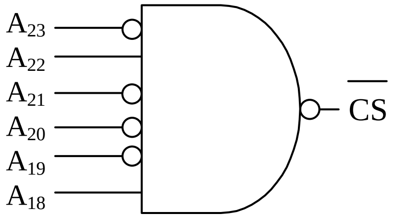
"Academic misconduct will be subject to disciplinary action. Any act of dishonesty in academic work constitutes academic misconduct. This includes plagiarism, the changing of falsifying of any academic documents or materials, cheating, and the giving or receiving of unauthorized aid in tests, examinations, 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. For each of the following statements, place a checkmark in the column identifying which memory technology, SRAM or DRAM, the statement best describes. (1 point each)

SRAM	DRAM	
☒	☐	is made from transistors much like a D-latch
☒	☐	is typically used as cache RAM
☒	☐	is the faster of the two technologies
☐	☒	is typically used as main memory
☐	☒	needs to be refreshed/rewritten when read

2. [✓ True] or false: The design of chip selects uses the same principles as that of subnet/host addressing in an IP address scheme. (2 points)

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 coming from the processor. This is determined 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. Address lines 18 to 23 go to the chip select circuitry while the remaining eighteen lines, 0 through 17, 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_{23}=0, A_{22}=1, A_{21}=0, A_{20}=0, A_{19}=0, and A_{18}=1. (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:  0100 0100 0000 0000 0000 0000`$_2$ = `440000`$_{16}$
`High address: 0100 0111 1111 1111 1111 1111`$_2$ = `47FFFF`$_{16}$

Low address: ____________________ High address: _________________________

4. For the chip select in problem 2, 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 2, 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.***

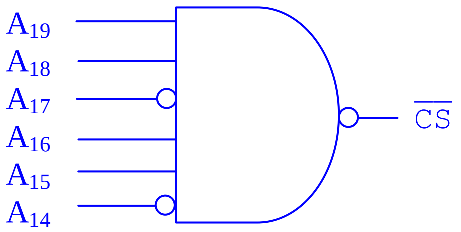
6. Using logic gates, design an active low chip select for a memory device placed in a 1 Meg memory space with a low address of $D8000_{16}$ and a high address of $DBFFF_{16}$. ***Label all address lines used for chip select.*** (5 points)

   Since 1 Meg = 2^{20}, the processor must have 20 address lines coming out of it in order to support a 1 Meg memory space. (A_0 through A_{19}). Therefore, our addresses must all have twenty 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.

   $D8000_{16} = 1101\ 1000\ 0000\ 0000\ 0000_2$

   $DBFFF_{16} = 1101\ 1011\ 1111\ 1111\ 1111_2$

   This shows that the lower 14 address lines (A_0 through A_{13}) go to the memory chip, and the upper 6 address lines (A_{14} through A_{19}) go to the chip select. Also from this diagram, we see that $A_{19} = 1$, $A_{18} = 1$, $A_{17} = 0$, $A_{16} = 1$, $A_{15} = 1$, and $A_{14} = 0$. By inverting the inputs that are to be recognized as zeros, we get the NAND circuit for the chip select shown to the right.



7. A chip select can have an address range from 9800_{16} to $9DFF_{16}$. (2 points)

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

   To determine this, we should simply try to design the chip select. Begin by converting the starting and ending addresses to binary.

   $9800_{16} = 1001\ 1000\ 0000\ 0000_2$

   $9DFF_{16} = 1001\ 1101\ 1111\ 1111_2$

   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_{10} had been zero or if address bit A_9 had been 1, this would have been possible. Unfortunately, this is not the case, so the answer is **FALSE**.

8. A chip select can be designed for a 1 Meg memory in a 16 Meg processor space where one of the memory device's addresses is $D4A1BD_{16}$. Don't try to design it; just say if it can be done. (2 points)

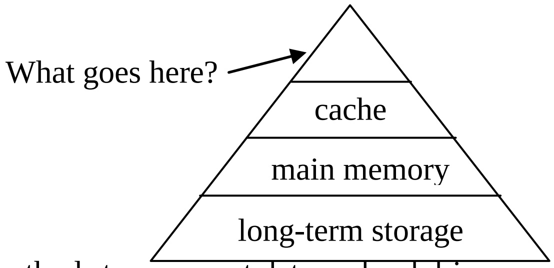
   [✓ a.) True] b.) False c.) Not enough information given

   Once again, to determine this we should simply try to design the chip select. Begin by converting the address $D4A1BD_{16}$ to binary.

   $D4A1BD_{16} = 1101\ 0100\ 1010\ 0001\ 1011\ 1101_2$

   The question now becomes where 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). We need to know where this occurs. We know that the memory chip is a 1 Meg chip, and therefore the twenty rightmost bits go to the memory chip. We also know that the processor has a memory space of 16 Meg. Therefore, there are 24 address bits coming out of the processor, 20 which go to the memory and 4 which go to the chip select. Since the chip select bits stay constant, then we need to design a chip select for $A_{23} = 1$, $A_{22} = 1$, $A_{21} = 0$, and $A_{20} = 1$. The answer is **TRUE**.

9. The figure to the right represents the memory hierarchy with one of the levels missing. Which level is missing? (2 points)



Registers are at the top of the memory hierarchy.

10. Name one of the benefits of using different encoding methods to represent data on hard drive platters. (2 points)

There are a number of benefits of using different encoding methods to store data on a hard drive.

- The controllers only detect changes in magnetic direction, not the direction of the field itself, and therefore only the boundaries between a 0 and a 1 or a 1 and a 0 would be detected.
- Large blocks of data that are all 1's or all 0's would be difficult to read because eventually the controller might lose track of where one bit ended and the next began.
- More advanced forms of encoding can be used to compress data allowing it to be stored more efficiently.

11. [✓ True] or false: It is possible to store data such that the width of 1 data bit is smaller than the width of the gap in the hard drive's write head, i.e., the minimum length of a polarity change. (2 points)

This is true. RLL for example can store a single bit in two-thirds of the gap in the hard drive's write head.

12. The average _______________ is computed as the time required for the platters to make half of a revolution. (2 points)

[✓ a.)] Rotational Latency
b.) Transfer time
c.) Seek Time

13. _______________ is the only period for which data is actually being read from the platters. (2 points)

a.) Rotational Latency
[✓ b.)] Transfer time
c.) Seek Time

14. The number of sectors per track on a ***constant angular velocity*** hard drive ________________ as you go closer to the center of the disk. (2 points)

a.) increases
b.) decreases
[✓ c.)] stays the same

15. True or [✓ false:] The rotational speed of the platter(s) measured in rotations per minute (RPM) of a ***multiple zone recording*** hard drive varies depending on the position of the head. (2 points)

16. Describe how the LFU replacement algorithm for the fully associative mapping algorithm works. (2 points)

The Least Frequently Used (LFU) replacement algorithm deletes the line from the cache that has been used (touched) the fewest times of any of the lines in the cache.

17. If system's memory is to be divided into blocks of size 32, how many bits are required for the word ID? (2 points)

a.) 1　　b.) 2　　c.) 3　　d.) 4　　[✓ e.) 5]　　f.) cannot be determined

Remember that the least significant bits are used to identify the word within a block. If 32 unique words need to be identified within a block, then it takes 5 bits to come up with 2^5 = 32 different IDs.

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 17 through 21.

Tag (21 bits)	Word position within block							
	000	001	010	011	100	101	110	111
110010110100110100110	00	61	C2	13	84	E5	46	A7
000110100000110010101	60	71	D2	33	94	F5	**36**	B7
010011110011011011111	20	81	E2	83	A4	05	66	C7
101110110011010100110	30	91	F2	53	B4	15	A6	D7
101001001110011010100	40	A1	02	63	C4	25	86	E7
100110100110100011001	12	34	56	78	9A	BC	DE	F0
011010110111001011001	23	45	67	89	AB	CD	EF	01
011011001101011011111	88	99	AA	BB	CC	DD	EE	FF
100010100000000010110	FE	DC	BA	98	76	54	32	10
101011001010110001101	ED	CB	A9	87	65	43	21	0F
111111010000110100110	11	44	55	77	0F	1F	2F	3F
Column →	a	b	c	d	e	f	g	h

18. 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 tag is 21 bits – therefore, the block ID is 21 bits. The word ID is three 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 21 + 3 = 24 bits.

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

The block size is the same thing as the number of words contained within a block. With three bits representing the word ID, then the block size is 2^3. Another way of doing this requires knowing that a line of the cache contains a block. Looking at the cache above shows that 8 data values are stored in a line, i.e., the block size is 8.

20. From what address in main memory did the value 36_{16} (the value in the second row, column g) come from? Leave your answer in binary. (2 points)

Fully associative mapping divides the physical address into two pieces, the block ID (which is used as the tag) and the word ID. In this case, the word id is the last 3 bits of the address. Since the 36_{16} has a tag of 000110100000110010101_2 and a word id of 110_2, then the physical address is $000110100000110010101110_2$ = **$1A0CAE_{16}$.**

21. A copy of the data from memory address $ACAC6B_{16}$ is contained in the portion of the cache shown above. What is the value stored at that address? (3 points)

Dividing the physical address $ACAC6B_{16}$ = $101011001010110001101011_2$ into its block ID and 3-bit word ID gives us a tag of 101011001010110001101_2 and a word id of 011_2. Searching through the visible lines shows us that the second to last line has the same tag, i.e., it contains the block which contains the data from the physical address $ACAC6B_{16}$. A word id of 011_2 points us to the data in column d. This means that the value stored at $ACAC6B_{16}$ and copied into the cache is **87_{16}.**

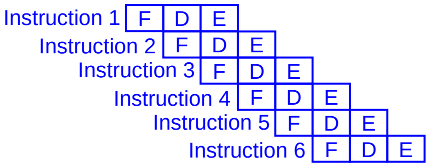
22. Name a type of instruction that would force a processor to "flush" its pipeline and begin filling it over again. (2 points)

Anything requiring a disruption in the sequential flow of code. This would include jumps due to loops, if-statements, switch/case statements, etc. In assembly language terms, these would be called conditional branches.

23. Assume a pipelined processor takes 3 cycles to execute any instruction (fetch, decode, execute). How many cycles would it take to execute 6 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 6 cycles to execute the 6 instructions.



$\boxed{\text{number of cycles} = 2 + 6 = 8 \text{ cycles}}$

24. [✓ True] or false: In theory, a processor's speed can be increased by increasing the number of stages in its pipeline. (2 points)

This is true. Basically, by dividing the execution of an instruction into more stages/cycles, you make much shorter cycles. Therefore, once you've filled the pipe, the execution of each instruction takes only a cycle, which in the case of more stages means it takes less time and is faster.

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? (5 points)

```
   1111
  00101010
+ 01111100
----------
  10100110
```

ZF = **0**   SF = **1**   CF = **0**   OF = **1**   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. We don't have a carry here. 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 a 2's complement overflow in this addition (the first case: positive + positive = negative). 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, therefore, the parity flag contains a 0.

26. The CPU has four main components. Which one performs the mathematical and logical functions, i.e., it acts as the CPU's calculator? (2 points)

This is the arithmetic logic unit or ALU.

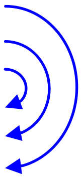
27. Name one of the three purposes presented in class for a stack. (2 points)

- Temporary storage of register values.
- Return address for a function call.
- Passing values to a function.

28. 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)

Place your answers in space below:

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



AX = **3000_{16}**

BX = **2000_{16}**

CX = **1000_{16}**

29. [✓ True] or false: High level languages such as C, C++, and Visual Basic have operators for performing bitwise operations. (2 points)

30. Remember that an IP address contains bits representing the subnet id and bits representing the host ID. Which bitwise operation could be used to isolate the subnet mask of an IP address, i.e., clear the bits for the host ID? (2 points)

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

31. Remember that a signed magnitude binary value represents a negative value by setting the MSB to a one. Which bitwise operation could be used to change the sign of a signed magnitude binary value, i.e., invert only the most significant bit? (2 points)

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

32. Using an original value of 10101010_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
10101010_2	AND	00001111_2	**00001010_2**
10101010_2	OR	00001111_2	**10101111_2**
10101010_2	XOR	00001111_2	**10100101_2**

33. A CRC is calculated using a simulated long division that using a bitwise XOR instead of subtraction to generate the result. Name one of the two reasons this is done. (2 points)

- 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.

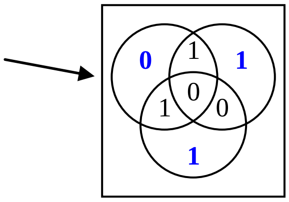
34. Name two of the benefits discussed in class of serial communications over parallel. (3 points)

- Faster communication rate due to lack of cross-talk
- Smaller connectors allowing for miniaturization
- Cheaper cabling due to fewer wires
- Fewer traces on circuit board allowing for miniaturization and cheaper circuit boards

35. 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	
☐	☒	☐	Used to direct packets from one device to another across multiple networks
☒	☐	☐	Uses a 7-byte preamble to synchronize all receiving devices
☐	☐	☒	Used to coordinate the breaking of a large message into smaller messages
☐	☒	☐	Uses logical addresses that are assigned to the device and can be changed
☒	☐	☐	Uses a 4-byte CRC for error checking
☐	☐	☒	Uses a 2-byte checksum for error checking that is based on the packet's header ***and*** a pseudo header created from the data and other information
☐	☒	☐	Includes a “time to live” field so that it can be removed from the network(s) in case it cannot find its destination.
☒	☐	☐	Limits the data field to a maximum of 1500 bytes

36. Add the parity/check bits that are missing from the graphic shown to the right. (2 points)



37. The graphic to the right depicts the digits of a 4-bit Hamming code where a single bit error has occurred. Circle the bit that has flipped. (2 points)