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 1 for Fall Semester, 2008

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.***
- You may ***NOT*** use a calculator. Leave all numeric answers in the form of a formula.
- You may use one sheet of scrap paper that you must turn in with your test.
- Please draw a box around your answers. This is to aid the grader. Failure to do so might result in no credit for answer. Example:

$$\boxed{32F1_{16}}$$

- **1 point will be deducted** per answer for missing or incorrect units when required. **No** assumptions will be made for hexadecimal versus decimal versus binary, 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.
- Statement regarding academic misconduct from Section 5.7 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."

Basic Rules of Boolean Algebra

	OR	AND	XOR
Combined w/0	$A+0=A$	$A\cdot 0=0$	$A\oplus 0=A$
Combined w/1	$A+1=1$	$A\cdot 1=A$	$A\oplus 1=\overline{A}$
Combined w/self	$A+A=A$	$A\cdot A=A$	$A\oplus A=0$
Combined w/inverse	$A+\overline{A}=1$	$A\cdot\overline{A}=0$	$A\oplus\overline{A}=1$
Other rules	$A+A\cdot B=A$	$A+\overline{A}\cdot B=A+B$	$(A+B)\cdot(A+C)=A+B\cdot C$
DeMorgan's Th.		$\overline{A\cdot B}=\overline{A}+\overline{B}$	$\overline{A+B}=\overline{A}\cdot\overline{B}$

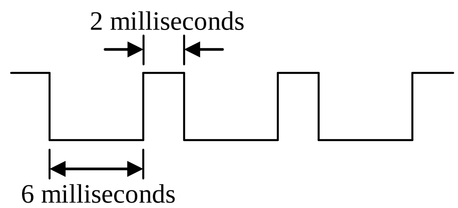
Short-ish Answer (2 points each unless otherwise noted)

1. For a system to reliably capture the frequency range from 250 Hz to 500 Hz, the system's sampling rate must be greater than __________.

   a.) 250 samples/sec. b.) 500 samples/sec. c.) 750 samples/sec. d.) 1000 samples/sec.

   The Nyquist rate is the minimum frequency required to sample a signal at in order to capture a desired frequency. The theorem states that you must sample at least twice as fast as the highest frequency you wish to capture. In this case, the highest frequency is 500 Hz which means that the sampling rate must be 2×500Hz = 1000 samples per second.

2. What is the frequency of the signal show to the right? (Note: 1 millisecond = 1×10^{-3} seconds)



   Frequency is the inverse of the period, so the first thing we need to do is determine the period. The period is equal to the time of a full cycle, which in the figure to the right is 6 milliseconds plus 2 milliseconds. This gives us a measurement for the period of 8 milliseconds or 0.008 seconds. Therefore, the answer is:

$$\text{frequency} = \frac{1}{\text{period}} = \frac{1}{0.008\text{ seconds}} = 125\text{ cycles/seconds} = 125\text{ Hz}$$

   You could have left your answer as 1/(0.008) Hz if you wanted to, but I wanted to see the units of Hz to be sure that you knew the units for frequency.

3. The duty cycle for the previous problem is:

   a.) 25% b.) 33% c.) 40% c.) 50% d.) 66% e.) 75% f.) 100%

   Before doing this mathematically, let's look at this by beginning with the definition of duty cycle. ***The duty cycle is the percentage of time that a signal is high, i.e., a logic 1.*** Looking at the diagram for this problem we see that the signal is high only a quarter of the time or 25%. Therefore, A should be the answer.

   We can also look at it using the equation that represents duty cycle. Officially, the expression used to determine the duty cycle is:

$$[\text{ time high } (t_h) / \text{ period } (T)\ ] \times 100\%$$

   Since during a single period, the signal from the problem is high 0.002 seconds and low 0.006 seconds, then the duty cycle is (0.002 seconds ÷ (0.002 + 0.006 seconds) × 100% = 25%.

4. True or False The frequency of a periodic signal can be calculated using only the duty cycle.

   Duty cycle is only a percentage of time that the signal is high. It doesn't refer to the rate at which the signal is changing back and forth between high and low. Therefore, the answer is FALSE.

5. What is the most negative value that can be stored using 10-bit 2's complement representation?

a.) -2^{10} b.) $-(2^{10} - 1)$ c.) -2^{9} d.) $-(2^{9} - 1)$ e.) -2^{8} f.) $-(2^{8} - 1)$

This problem could also have been done a couple of different ways. First, you could have memorized the formula that the most negative twos-complement value is -2^{n-1} where n is the number of bits. That would have given us -2^9. You could have also remembered that the most negative 2's complement value in binary is 1000…000 (all bits are zero except for the most significant bit which is a 1.) For 10 bits this would have been 1000000000_2. Converting 1000000000_2 to a positive value would be the same pattern, i.e., 1000000000_2. In unsigned binary, 1000000000_2 is 29 or 512. Therefore, the original negative value is -2^9.

If you gave the answer for the most negative signed magnitude value, i.e., $-(2^9 - 1)$, I gave you a point.

6. What is the minimum number of bits needed to represent 999_{10} in unsigned binary representation?

a.) 8 b.) 9 c.) 10 d.) 11 e.)12 f.) 13

This is yet another problem that could have been done a couple of different ways. First, you could have memorized the formula for the largest possible unsigned binary value is $2^n - 1$ where n is the number of bits. Trying a couple of values for n might have resulted in the following table:

n	$2^n - 1$
8	255
9	511
10	1023
11	2047

The table shows us that 10 bits is the first level that has a high enough maximum value.

Another way to do this is to realize that the largest unsigned binary value is all ones. We can create a table similar to the previous one for every unsigned binary value with all ones.

Binary Value	Decimal Value
11111111_2	255
111111111_2	511
1111111111_2	1023
11111111111_2	2047

Once again, we see that 10 bits is the threshold.

7. Which signed binary representation works for addition of both positive and negative values?

a.) signed magnitude   b.) 2's complement   c.) neither

Signed magnitude would require both a subtraction unit and an addition unit and logic to decide whether negative numbers were being added to determine which unit to use. Twos complement, however, can add both negative and positive numbers with same addition unit.

8. How many patterns of ones and zeros does a 6-bit ***Gray code sequence*** consists of?

Just like unsigned binary, Gray code uses all the patterns of ones and zeros available for the number of bits being used. Therefore, the answer is $2^6 = 64$.

9. The IEEE-754 32-bit floating-point value 01000000100101000000000000000000 is _________. (3 points)

a.) less than zero   b.) between zero and one   c.) greater than one

This was a little trickier a problem than I've given in the past, therefore, everyone got at least some partial credit. Basically, the first bit is a zero. This is a sign bit which means that the number is positive. That meant that answer A is not possible. Second, the value of E is 10000001_2. In decimal, this is 128 plus 1 which equals 129. Therefore, when insert E into the IEEE-754 expression $\pm 1.F \times 2^{(E-127)}$, we see that the exponent of the 2 is going to be 129 – 127 = 2 which is a positive number. Therefore, the decimal point is going to be shifted to the right making this a value greater than one.

I gave everybody at least a point, i.e., the most I took off for this problem was 2. You got an extra point if you identified it as a positive value, and you received the last point if you knew it was greater than 1.

10. Write the complete truth table for a 2-input XOR gate. (3 points)

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

Remember that the output of a XOR gate is one if the number of ones at the inputs is an odd number. For two inputs, the only times this happens is for A=0, B=1 and A=1, B=0.

The following two questions are based on the 8-bit binary addition shown below.

```
    0   01110010
       +00010101
        10000111
```

No carry out

11. True or False: If the addition above is considered 8-bit 2's complement, an overflow has occurred.

True. If two positive numbers are added together to result in a negative number, an overflow has occurred. Remember that the only way a 2's complement can have an overflow is if the sign bits of the numbers that are being added are the same as each other, but different from the sign bit of the result.

12. True or False: If the addition above is considered 8-bit unsigned, an overflow has occurred.

False. A non-zero carry out of an unsigned binary addition is the indicator of an overflow.

13. In the boolean expression below, circle the ***single*** operation that would be performed first.

$$A + \overline{(B \oplus C)} \cdot D$$

It is the OR inside of the parenthesis that is executed first. It is executed even before the inversion, so if you selected the inversion too, you actually identified ***two*** operations and lost a point.

14. Multiply the 16-bit value 0000111011101000_2 by 8. ***Leave your answer in 16-bit binary.*** (Hint: Remember the shortcut!)

Since 8 is a power of two, i.e., $8 = 2^3$, then the multiplication can be performed by simply shifting the binary number left 3 positions. This is the same thing as adding three zeros to the right hand side of the number. The additional bits are shown below in red. Note that the processor has a fixed size for storing the value, so three zeros were discarded from the left side.

$$0111011101000000_2$$

If you want to do things the hard way, you could have converted the numbers. 0000111011101000_2 equals $2^{11} + 2^{10} + 2^9 + 2^7 + 2^6 + 2^5 + 2^3 = 2048 + 1024 + 512 + 128 + 64 + 32 + 8 = 3816_{10}$. Multiplying 3816_{10} by 8 gives us 61056_{10}. Wow! Okay, reversing the conversion process gives us $61056 = 2^{14} + 2^{13} + 2^{12} + 2^{10} + 2^9 + 2^8 + 2^6$ which in binary is 0111011101000000_2.

15. Convert $1101000011011111011101111_2$ to hexadecimal.

Create the conversion table between binary and hex. (That table is shown to the right.) Next, partition the number to be converted into nibbles. You must do this starting from the ***right side*** with the least significant bits. Starting from the left might leave you with a partial nibble on the right side. The correct result is shown below:

$$0001\ 1010\ 0001\ 1011\ 1110\ 1110\ 1111_2$$

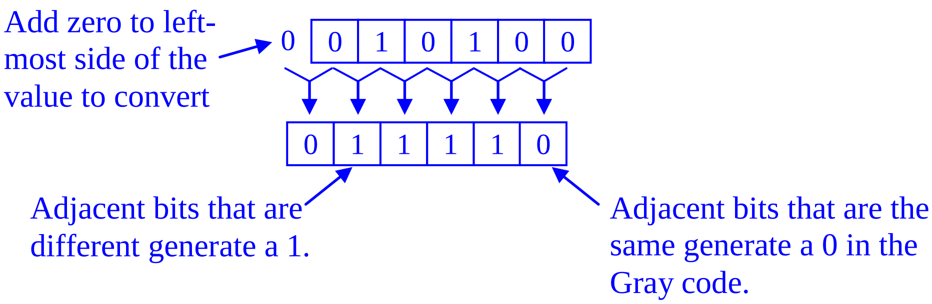
Notice that three leading zeros (in red) needed to be added to the left side. Each of these nibbles corresponds to a pattern from the table to the right. Now it just becomes a straight conversion.

$1A1BEEF_{16}$

Binary				Hexadecimal
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	A
1	0	1	1	B
1	1	0	0	C
1	1	0	1	D
1	1	1	0	E
1	1	1	1	F

16. Convert the unsigned binary value 010100_2 to its corresponding 6-bit binary Gray code. (3 points)

From page 39 of the textbook, we have the conversion sequence which says to begin by adding a leading zero to the number to be converted. For each boundary between bits, place a 1 if the bits on either side of the boundary are different and place a 0 if the bits on either side of the boundary are the same.



Therefore, the answer is 011110.

Medium-ish Answer (4 points each unless otherwise noted)

17. Convert the 32-bit IEEE 754 floating-point number 00111110110101100000000000000000 to its binary exponential format, e.g., -1.001010×2^{-12}, (which, by the way, is not the answer).

Begin by dividing up the floating-point number into its components.

S	E	F
0	01111101 = 64 + 32 + 16 + 8 + 4 + 1 = 125	10101100000000000000000

Substituting into the expression $\pm 1.F \times 2^{(E-127)}$ gives us our answer.

$$\pm 1.F \times 2^{(E-127)} = +1.101011 \times 2^{(125-127)} = 1.101011 \times 2^{-2} = 0.01101011 = 0.41796875$$

The last step, converting to 0.01101011 = 0.41796875 was not necessary.

18. Convert 1011.101_2 to decimal. (You may leave your answer in expanded form if you wish.)

Remember that binary digits to the right of the point continue in descending integer powers relative to the 2^0 position. Therefore, the powers of two are in order to the right of the point $2^{-1} = 0.5$, $2^{-2} = 0.25$, $2^{-3} = 0.125$, and $2^{-4} = 0.0625$. (Note that 2^{-4} is not needed for this problem.)

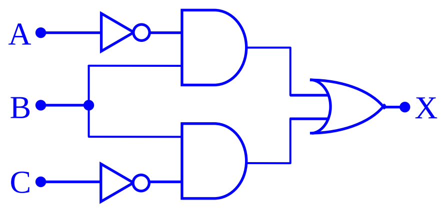
2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
0	1	0	1	1	1	0	1	0

Therefore, the answer is:

$$2^3 + 2^1 + 2^0 + 2^{-1} + 2^{-3} = 8 + 2 + 1 + 1/2 + 1/8 = 11\ 5/8 = 11.625$$

You could have left your answer in any of these forms in order to receive full credit.

19. Draw the circuit ***exactly*** as it is represented by the Boolean expression $\overline{A}\cdot B+B\cdot\overline{C}$.

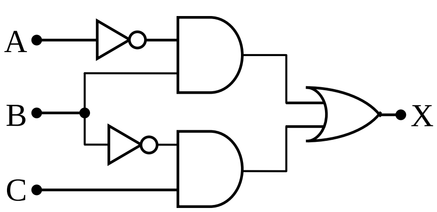


20. Prove that $A\oplus\overline{A}=1$. (Remember that $\oplus$ is the XOR or exclusive-OR) ***For full credit, please show all steps in detail.***

The table below is all that is needed to prove the expression is true. The important part is that both the columns for A⊕~A ***and*** ~A (~A represents "not A") are both shown so that the data sources are obvious. Basically, anything exclusive-OR'ed with its inverse results in 1.

A	~A	A⊕~A
0	1	0 ⊕ 1 = 1
1	0	1 ⊕ 0 = 1

21. In the space to the right, create the truth table for the circuit shown below. (6 points)



A	B	C	$\overline{A}$	$\overline{A}\cdot B$	$\overline{B}$	$\overline{B}\cdot C$	$\overline{A}\cdot B+\overline{B}\cdot C$
0	0	0	1	0	1	0	0
0	0	1	1	0	1	1	1
0	1	0	1	1	0	0	1
0	1	1	1	1	0	0	1
1	0	0	0	0	1	0	0
1	0	1	0	0	1	1	1
1	1	0	0	0	0	0	0
1	1	1	0	0	0	0	0

22. Write the Boolean expression for the circuit shown in the previous problem. ***Do not simplify!***

The answer below is simply copied from the truth table above where the Boolean expression was derived.

$$\boxed{\overline{A}\cdot B+\overline{B}\cdot C}$$

23. Use DeMorgan's Theorem to distribute the inverse of the expression $\overline{A \cdot B + \overline{C}}$ all of the way to the individual input terms. Be careful of the AND and OR precedence. It might help to see the expression as a logic circuit. ***Do not simplify!***

$\overline{A\cdot B}\cdot C$	Start by distributing the inverse across the A·B term and the C term. This will flip the + (OR) to a · (AND). The double bar over C cancels itself
$\overline{A\cdot B}\cdot C$	Next, distribute the bar across A·B. Note that the bar acts as parenthesis making it so that ~(A·B) and subsequently ~A + ~B must be processed first.
$(\overline{A} + \overline{B})\cdot C$	This gives us our final answer. If you were not able to keep A and B together with parenthesis, I took off a point.

24. Assume that an 8-bit binary number is used to represent an analog value in the range from -8 to 119. Convert all four of the following binary values to their analog equivalent, i.e., what analog value does each of these binary values represent? You may leave your answer in the form of a fraction in some cases if you wish. (5 points)

Remember that the range of bit patterns for an 8-bit binary value is distributed evenly across the analog range where all zeros represents the low end of the range and all ones represents the high end. That should immediately answer parts a and d, i.e., 00000000_2 = the lowest value, i.e., –8, and 11111111_2 = highest value, i.e., 119.

For b and c, you need to calculate how much a single increment changes the analog value. For 8 bits, there are $2^8 - 1 = 255$ increments over a range from –8 to 119 = 127. That means that a single increment represents a difference in the analog value of 127/255. This immediately answers part b because 00000001_2 is exactly one increment above –8 and hence represents –8 + 127/255. Part c is the hard one. In unsigned notation, c represents $00001010_2 = 8 + 2 = 10_{10}$ increments above –8. Therefore, the analog value it corresponds to is –8 + (10 × 127/255).

a.) 00000000_2 = –8

b.) 00000001_2 = –8 + (127/255)

c.) 00001010_2 = –8 + 10×(127/255)

d.) 11111111_2 = –8 + 255×(127/255) = –8 + 127 = 119

Longer Answers (Points vary per problem)

25. Mark each Boolean expression as ***true*** or ***false*** depending on whether the right and left sides of the equal sign are equivalent. Show all of your work to receive partial credit for incorrect answers. (3 points each)

a.) $A \cdot B \cdot (\overline{A} + \overline{B}) = 0$ Answer: **True**

$A\cdot B\cdot\overline{A} + A\cdot B\cdot\overline{B}$	Distribute A·B
$B\cdot A\cdot\overline{A} + A\cdot B\cdot\overline{B}$	Rearrange with commutative law
$B\cdot 0 + A\cdot 0$	Anything AND'ed w/inverse = 0
$0 + 0$	Anything AND'ed w/0 equals 0
0	Zero or zero is always zero

b.) $A + \overline{(A + \overline{B})} + \overline{A} \cdot B \cdot C = A + \overline{B}$ Answer: **False**

$A + \overline{A}\cdot B + \overline{A}\cdot B\cdot C$	Apply DeMorgan to ~(A + ~B)
$A + \overline{A}\cdot B(1 + C)$	Pull out ~A·B from last 2 terms
$A + \overline{A}\cdot B\cdot 1$	Anything OR'ed w/1 is 1
$A + \overline{A}\cdot B$	Anything AND'ed w/1 is itself
$A + B$	Rule from the "other" rules

c.) $\overline{\overline{(A + B\cdot C)} + \overline{A\cdot\overline{B}}} = A\cdot\overline{B}$ Answer: **True**

$(A + B\cdot C)\cdot(A\cdot\overline{B})$	Apply DeMorgan to ~(A + B·C) and ~(A·B). This cancels double inverses and turns OR to AND.
$A\cdot\overline{B}\cdot A + A\cdot\overline{B}\cdot B\cdot C$	Multiply A·(~B) through
$\overline{B}\cdot A\cdot A + A\cdot\overline{B}\cdot B\cdot C$	Rearrange using commutative law
$\overline{B}\cdot A\cdot A + A\cdot 0\cdot C$	Anything AND'ed w/inverse = 0
$\overline{B}\cdot A + A\cdot 0\cdot C$	Anything AND'ed w/self = self
$\overline{B}\cdot A + 0$	Anything AND'ed w/0 equals 0
$\overline{B}\cdot A$	Anything OR'ed w/0 equals self

26. Fill in the blank cells of the table below with the correct numeric format. ***For cells representing binary values, only 8-bit values are allowed!*** If a value for a cell is invalid or cannot be represented in that format, write "X". (7 points per row)

Decimal	2's complement binary	Signed magnitude binary	Unsigned binary	Unsigned BCD
–112	**10010000**	**11110000**	**X**	**X**
–58	**11000110**	**10111010**	**X**	**X**
198	**X**	**X**	**11000110**	**0001 1001 1000 or X**

Negative values: Let's take care of the negative values first. Note that the first row is clearly a negative value, i.e., –112. Therefore, there should be X's in the unsigned binary and unsigned BDC columns for this row. Second, the 2's complement value in the second row, 11000110, begins with a 1. This is the sign bit meaning that this row also is negative and X's should be placed in the unsigned binary and unsigned BCD columns for this row. Wow, that's 6 points off the bat.

Out of range values: The last column has an unsigned binary value of 11000110. This value uses the MSB for magnitude and therefore cannot be used as a 0 (positive) sign bit for either of the signed representations, i.e., twos complement and signed magnitude. Therefore, X's must go in these two columns for this row.

Unsigned binary values: The third row contains a simple unsigned binary value. To convert this to decimal, simply add up the powers of two represented by each of the 1's in the binary value. These are 2^7, 2^6, 2^2, and 2^1. Adding these up gives us $2^7 + 2^6 + 2^2 + 2^1 = 128 + 64 + 4 + 2 = 198_{10}$.

For this same column, we could convert to unsigned BCD. There were two possible answers for this column. First, if you noted that the limit was 8 bits for this column and since 198 would take three nibbles in BCD, the value is outside of the range, i.e., put an X in this column. If, however, you wanted to make sure that I knew you could convert to BCD, you could have used the hexadecimal to binary conversion table you developed earlier in this test to convert 198 to 0001 1001 1000. I accepted either answer.

Converting from negative decimal: Conversion for any negative value must begin first with the positive representation. Therefore, begin converting –112 to binary by starting with 112. Breaking 112 into its powers of 2 components gives us $112 = 64 + 32 + 16 = 2^6 + 2^5 + 2^4$. Therefore, the binary value we get is the one shown in the table below.

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
0	1	1	1	0	0	0	0

Since the MSB is 0, the magnitude of 112 is small enough to be converted to 8-bit twos complement and signed magnitude.

For the 2's complement representation, we take the positive magnitude, and do the bit-flippy thing. This is shown in the table below where the blue numbers are just copied down and the red numbers are the inverted bits.

112:	0	1	1	1	0	0	0	0
–112:	1	0	0	1	0	0	0	0

So the final 2's complement value is 10010000_2.

To convert 01110000_2 to signed magnitude, simply flip the MSB (sign bit) to get 11110000_2.

Converting from negative 2's complement: Once again, converting any negative value must begin first with the positive representation. Therefore, we must take the negative 2's complement value and turn it into a positive (unsigned binary) using the bit-flippy thing. This is shown in the table below where the blue numbers are just copied down and the red numbers are the inverted bits.

1	1	0	0	0	1	1	0
0	0	1	1	1	0	1	0

This means that the unsigned binary value (absolute value) of the twos complement value 11000110_2 is 00111010_2.

To convert 00111010_2 to negative signed magnitude, simply flip the MSB (sign bit) to get 10111010_2.

To convert twos complement value 11000110_2 to decimal, first figure out what the unsigned value of 00111010_2 is, then add a negative sign. 00111010_2 has ones in the 2^5, 2^4, 2^3, and 2^1 positions. Therefore, its decimal value is $2^5 + 2^4 + 2^3 + 2^1 = 32 + 16 + 8 + 2 = 58$. Therefore, the twos complement value 11000110_2 is equal to $–58_{10}$.