English Name: ____________________

Worksheet: Converting Values

1. Write the appropriate vocabulary word next to its definition.

a)	convert	to change from one form to another, for example from binary to denary
b)	complement	an operation that inverts (or "flips") the bits of a binary number (0 ↔ 1)
c)	binary	a name for the base-2 number system
d)	hexadecimal	a name for the base-16 number system
e)	byte	a set of 8 binary digits
f)	nibble	a set of 4 binary digits

2. Given the binary number 1010_0110, make the conversions specified below. It is easy to make small mistakes, so show your work for partial marks.

a) Convert the value to denary, assuming the byte represents a ***two's complement*** number.

place value in denary	128	64	32	16	8	4	2	1
original two's complement	1	0	1	0	0	1	1	0
complement the bits (flip)	0	1	0	1	1	0	0	1
add one (+1)	0	1	0	1	1	0	1	0
sum of denary bit values	64 + 16 + 8 + 2 = 90; negated = **-90**							

b) Convert the value to denary, assuming the byte represents a ***sign and magnitude***.

bit position		- / +	6	5	4	3	2	1	0
place value	exponential		2^6	2^5	2^4	2^3	2^2	2^1	2^0
	denary		64	32	16	8	4	2	1
binary digits (bits)		1	0	1	0	0	1	1	0
sum of denary bit values		- (32 + 4 + 2) = **-38**							

c) Convert the value to denary, assuming the byte represents an ***unsigned number***.

bit position		7	6	5	4	3	2	1	0
place value	exponential	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
	denary	128	64	32	16	8	4	2	1
binary digits (bits)		1	0	1	0	0	1	1	0
sum of denary bit values		128 + 32 + 4 + 2 = **166**							

d) Convert the value to *hexadecimal*.

1	0	1	0	0	1	1	0
A				6			

English Name: ______________________

Worksheet: Converting Values

3. Again working with the same binary number, 1010_0110, complete the tables below. It is easy to make small mistakes, so show your work for partial marks.

a) Perform an ***arithmetic right shift*** by <u>two bit positions</u>, then convert the result to denary, assuming the byte represents a ***two's complement*** number.

place value in denary	128	64	32	16	8	4	2	1
original unsigned number	1	0	1	0	0	1	1	0
arithmetically shifted signed number	1	1	1	0	1	0	0	1
one's complement	0	0	0	1	0	1	1	0
two's complement	0	0	0	1	0	1	1	1
sum of denary bit values	16 + 4 + 2 + 1 = 23; negated = **-23**							

b) Perform a ***logical left shift*** by <u>two bit positions</u>, then convert the result to denary, assuming the byte represents an ***unsigned number***.

place value in denary	128	64	32	16	8	4	2	1
original unsigned number	1	0	0	1	1	0	0	0
logically shifted unsigned number	1	0	0	1	1	0	0	0
sum of denary bit values	128 + 16 + 8 = **152**							

c) A ***logical left shift*** by <u>two bit positions</u> should be equivalent to multiplication by four (×4); however the shifted unsigned value calculated in part (3b) is obviously not even close to four times the original unsigned value we calculated in part (2c). What word describes what happened?

overflow

d) First perform a ***logical left shift*** by <u>two bit positions</u> as in part (b), then convert the result to *hexadecimal.*

1	0	0	1	1	0	0	0
9				8			

e) Perform a ***logical right shift*** by <u>two bit positions</u>, then convert the result to denary, assuming the byte represents an ***unsigned number***.

place value in denary	128	64	32	16	8	4	2	1
original unsigned number	1	0	1	0	0	1	1	0
logically shifted unsigned number	0	0	1	0	1	0	0	1
sum of denary bit values	32 + 8 + 1 = **41**							

f) What mathematical operation (operator and operand) is equivalent to a ***logical right shift*** by <u>two bit positions</u>, assuming the byte represents an ***unsigned number***.

operand (binary)	operator	operand (denary)
1110_1001	÷	4 (or: 2^2)

Note: the reason this mathematical expression isn't exactly true is due to the ***loss of precision*** as bits are shifted out of the number, and lost.