Homework Set 1 for Module 1

Due at the Beginning of Class (12:30 pm) on Friday, August 31

Name: _____ Class No: __ __ __ __ - __ Lab Div: ____

Signature: _____ Score: _____ / 100

1. [10 points] Interpreting the values loaded into (A) and (B) as binary fractions, determine the result produced in (A) after executing the following code sequence (express the result as a ***base 10*** fraction).

```
LDAA #$73
LDAB #$FD
MUL
ADCA #0
```

2. [20 points] Interpreting the values loaded into (D) and (Y) as *signed words*, determine the result stored in `product` after executing the following code sequence (express the result as a ***base 10*** number).

```
LDD #$274A
LDY #$FFE5
EMULS
STY  product
STD  product+2

product rmb 4
```

3. [20 points] Interpreting the values loaded into (D) and (X) as *signed words*, determine the values stored in `quotient` and `remainder` after executing the following code sequence (express the result as a ***base 10*** number).

```
LDD #$747E
LDX #$FA95
IDIVS
STX  quotient
STD  remainder

quotient rmb 2
remainder rmb 2
```

4. [4 points] Determine how the CC register bits will be affected (i.e., set, cleared, or unaffected) by execution of the following code sequence.

```
LDAA #$AA
BITA #$55
```

5. [4 points] Write a *single instruction* that *sets* bit 7 (the most significant bit) of memory location $900 and leaves the other bits unaffected.

6. [4 points] Write a *single instruction* that *clears* bit 0 (the least significant bit) of memory location $901 and leaves the other bits unaffected.

7. [4 points] Write a *single instruction* that *toggles* both bits 2 and 3 of the B register but leaves the other bits unaffected.

8. [4 points] Write a *single instruction* that *clears* the C, H, and V condition code register bits but leaves the others unaffected.

9. [4 points] Write a *single instruction* that *sets* the C and I condition code register bits but leaves the others unaffected.

10. [6 points] List all of the *conditional branch* instructions that utilize a *9-bit* offset and explain how the offset is encoded.

11. [10 points] Assemble the following code segment (by hand), and show the machine code that is generated:

```
        org   $8000
        lbra  jloop
jloop   bra   kloop
kloop   lbra  jloop
```

12. [10 points] Complete the missing entries in the "condition code generation chart," below, and *derive* the Boolean function used by the BGT function to determine if a branch should be taken, i.e. the function that is *true* ("1") when (A) > (B). ***Show all work.***

A_1	A_0	(A)	B_1	B_0	(B)	?	C	Z	N	V
0	0	0	0	0	0	(A) = (B)	0	1	0	0
0	0	0	0	1	+1	(A) < (B)				
0	0	0	1	0	-2	(A) > (B)	1	0	1	1
0	0	0	1	1	-1	(A) > (B)	1	0	0	0
0	1	+1	0	0	0	(A) > (B)	0	0	0	0
0	1	+1	0	1	+1	(A) = (B)	0	1	0	0
0	1	+1	1	0	-2	(A) > (B)	1	0	1	1
0	1	+1	1	1	-1	(A) > (B)	1	0	1	1
1	0	-2	0	0	0	(A) < (B)				
1	0	-2	0	1	+1	(A) < (B)				
1	0	-2	1	0	-2	(A) = (B)	0	1	0	0
1	0	-2	1	1	-1	(A) < (B)				
1	1	-1	0	0	0	(A) < (B)				
1	1	-1	0	1	+1	(A) < (B)				
1	1	-1	1	0	-2	(A) > (B)	0	0	0	0
1	1	-1	1	1	-1	(A) = (B)	0	1	0	0