

1. Given the binary bit string **1 0 1 1 1 0 1**

What decimal number does it represent if it is stored in:

- (1) Unsigned format?
- (1) Sign-Magnitude format?
- (1) Excess-64 format?
- (1) Two's complement format?
- (1) What **character** does it represent?

2. Given the following Base 4 number: **3 0 1 2**

- (1) What is the Binary representation of this number?
- (1) What is the Octal representation of this number?
- (1) What is the Decimal representation of this number?
- (1) What is the Hex representation of this number?

3. (6) Assuming a 12-bit floating point format where the most significant bit of the word is the sign bit for the mantissa, the next 6 bits are the mantissa itself, and the remaining 5 bits are the exponent stored in Excess-N format, write down the bit pattern for the following decimal number (Assume the format does use hidden one's):

$$-37.5 \times 10^{-2}$$

- 2 -

8. Suppose you have a machine with the following characteristics:

The left most 5 bits store the opcode

the next 6 bits tell how to find the Destination/Source1 operand

the last 6 next to it hold the Source2 operand

a.(1) Draw a picture of the instruction. How many bits does it have in total?

b (2). What is the maximum number of different opcodes (instructions) this machine could support?

c.(2) Assuming the machine only supports memory direct addressing, how many different memory locations can it access?

d.(3) Assuming the machine supports both memory direct and register direct addressing, and the instruction size does not change (still 5 bits in the opcode field, etc.), how many different registers can be accessed?

e.(2) Assuming the registers are the same size as the instructions, how many memory locations can be accessed if the machine supports register indirect addressing?

9. 9) Explain the effects of the following instruction sequence on the registers, flags, and memory locations that are indicated. Assume that all programs start at memory location \$000. If a value cannot be determined from the information in the instruction fragment, then indicate so by using ???

```
LDA#  $001  
CMA  $001  
JLT   $004  
ADA#  $302  
HLT
```

ACC =

LT =

PC =

10. (28) Given the following code sequence:

Program	Show your work area	Symbol Table
.PAGE		
.EQU	@,\$80	
.WORD	8	
BLOCK: .BLKW	5,\$001	
.EQU	VAR1,\$087	
START: LDX#	\$003	
LDA	BLOCK	
ADA	VAR2+1	
STA+	VAR1	
.EQU	@,\$8A	
LDA#	\$002	
CMA	VAR2	
JGE	DONE	
DEC	VAR2	
TAX		
DONE: HLT		
VAR2: .WORD \$040550		
.WORD \$00208F		
.WORD \$100015		
.END		

Fill out the symbol table. What memory locations and registers are **altered** by the *execution* of this program? Show the values of the registers and memory locations that have **changed** when the program terminates. Assume the PC is incremented before the current instruction is executed. (Fetch, Decode, Inc PC, Execute).

ACC =

XR =

PC =

MEM[] =

MEM[] =

MEM[] =

MEM[] =

MEM[] =

MEM[] =

MEM[] =

MEM[] =

MEM[] =

MEM[] =

11. (20) Your job, should you decide to accept it, is to assemble the following program and create a list file output. You should also fill out the symbol table as you go. (Just create the list file for the first 12 lines). You don't need to fill out all the source lines, just enough so I know which line you are on.

Assembly Language Program:

```

                .EQU    @,$100
                .EQU    ONE,2
                .EQU    TWO,127
THREE:         .WORD    $F02B36
FOUR:          .WORD    2
                LDA#    FOUR
                .EQU    FOUR,7
                STA+    ONE
                LDX     THREE
                HLT
                .END

```

Symbol Table	

LINE	ADDR	CONTENTS	SOURCE LINE
1			
2			
3			
4			
5			
6			
7			
8			
9			
10			
11			
12			

ASCII Table (MSD = Most Significant Digit)																
MSD (Hex)	Least Significant Digit (Hex)															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	space	!	"	#	\$	%	&	,	(	)	*	+	'	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	'	a	b	c	d	e	f	g	h	i	h	k	l	m	n	o
7	p	q	r	s	t	u	v	w	x	y	z	{		}		del

Instructions:

Instruction	Opcode (in Hex)
LDA	00
LDX	01
LDS	02
LDF	03
STA	04
STX	05
STS	06
STF	07
ADA	10
ADX	11
ADS	12
ADF	13
SBA	14
SBX	15
SBS	16
SBF	17
CMA	20
CMX	21
CMS	22
CMF	23
HLT	FFFFFF

Addressing Modes:

Mode	Opcode
Immediate	0
Direct	2
Indexed	4