1.1 Number representation

(To be done over 2 week)

(1) Give the *decimal* equivalent value of the 8-bit binary numbers using both the ***unsigned magnitude*** and ***2's complement*** number representations.

The binary numbers are:

(a) **0111 1111$_2$** (b) **1111 1111$_2$** (c) **0000 0000$_2$**
(d) **1000 0000$_2$** (e) **1111 1110$_2$**

(2) For each of the number representation above, state the **largest** and **smallest** decimal numbers you can represent using only eight binary digits.

(3) Give the numeric range of variables declared with the following ANSI C data types:

(a) `unsigned char` (b) `short int`
(c) `unsigned short int` (d) `long int`

(4) What would be the most appropriate ANSI C data type to assign to a variable in your program that represents the following:
(a) current temperature (°C) in whole number at any selected city in the world.
(b) total undergraduate population in NTU at any given moment.
(c) current total US national debt in US$ (check: http://www.usdebtclock.org/).
(d) whether a person is male or female.

1.2 Hexadecimal number representation

A list of hexadecimal numbers are given below :

(a) **0xFF**

(b) **0x0F**

(c) **0x4D**

(d) **0xC0**

(e) **0x30**

Note :
The **'0x'** prefix is used to signify that the number is in hexadecimal notation.

MS / LS	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	'	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	
F	SI	US	/	?	O	_	o	DEL

ASCII Character Set (7-Bit Code)

(1) Which of these numbers are negative in 2's complement representation?

(2) Which of these numbers are valid ASCII alpha-numeric characters?

(3) Give the decimal equivalent values of the 2's complement numbers in (a) and (b)?

(4) Would the 16-bit hexadecimal values of **0xFFFF** and **0x000F** have the same decimal value (*in 2's complement representation*) as the corresponding 8-bit values **0xFF** and **0x0F** given in (a) and (b) respectively?

Describe a simple technique to convert any 8-bit number to its equivalent 16-bit number without changing their signed value in 2's complement representation.

1.3 Data representation in memory

Various variables and constants of different ANSI C data types have been declared and stored in memory. Figure 1.3 shows the byte-sized contents in memory where they can be found. Based on these memory contents, answer the following questions:

Address	Contents
0x0000	0x08
0x0001	0x35
0x0002	0xFF
0x0003	0x01
0x0004	0xA8
0x0005	0x2A
0x0006	0x4C
0x0007	0x6F
0x0008	0x67
0x0009	0x69
0x000A	0x6E
0x000B	0x3A
0x000C	0x00
0x000D	0x61
0x000E	0x62
0x000F	0x63

Figure 1.3 – Contents in memory

(1) Find the start address of a 2-byte integer with the decimal value of 511. Is this integer in big or little endian format?

(2) Find a possible C string among the memory contents.

(3) A structure and variable declaration is given below:

```
struct rec {
   unsigned char i;
   long int j;
   char a[3];
};
struct rec r;
```

Assume the starting address of variable **r** is 0x0008 and the big endian format has been adopted, give (in hexadecimal) the values of following:

(a) **r.i** (b) **r.j** (c) **r.a[0]** (d) **r.a[2]**

1.4 Data and Address Busses

Figure 1.4 shows the pin out of the MC68000 microprocessor. The data pins are labeled Dn and the address pins are labeled An.

(1) Based on the address pin labeling, what is the maximum memory capacity addressable by this processor (in Mbytes)? Assume the missing address pin A0 can be derived from the UDS* and LDS* pins.

(2) Based on the data pin labeling, what is the maximum number of bytes can this processor transfer within one memory cycle?

(3) With reference to the data structure in Question 1.3 part (3), re-design the structure `rec` to make it more efficient for use within a computing system supported by a MC68000 processor? Give a reason for your re-design.

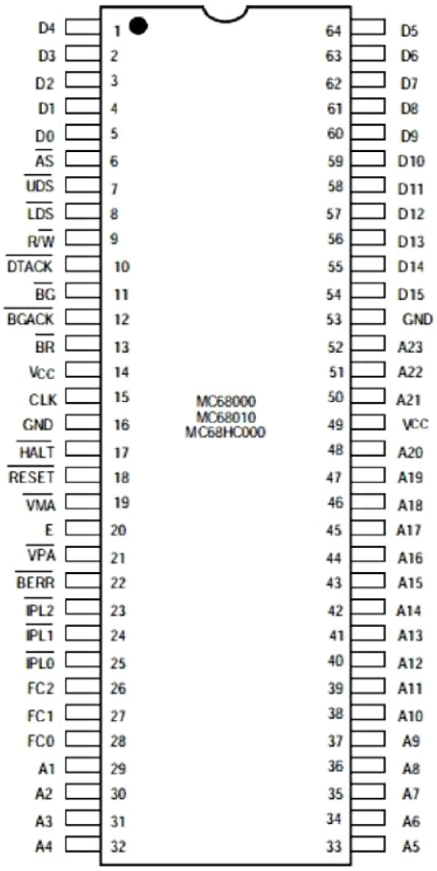

Figure 1.4 – MC68000 processor pin out

1.5 VIP instructions, its execution and representation in memory

Figures 1.5a and 1.5b show three VIP instructions in program memory and the current state of the registers in the VIP processor, respectively. All numeric values displayed are in hexadecimal notation except that displayed in the SR, which is in binary notation. Based on the information provided, answer the following questions:

Listing File

Source Line No.	Mnemonics	Program Address	Machine Code
1	MOV R1,#0xFFF	000	01C
1		001	FFF
2	MOV R2,#0x001	002	02C
2		003	001
3	ADD R1,R2	004	412

Figure 1.5a – Listing File view showing three VIP instructions in memory.

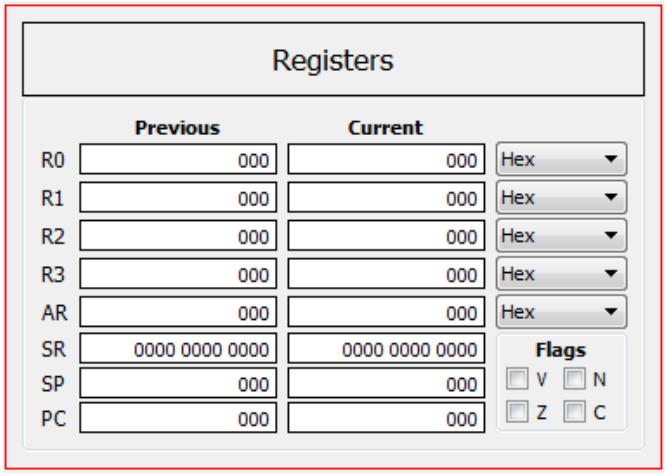

Figure 1.5b – Register View showing the various registers in the VIP processor.

(1) How many bits are stored at each program address in the VIP processor?

(2) What is the machine code for the mnemonic **MOV R2,#0x001** and how many word is required to encode this instruction?

(3) Briefly describe what each of the instructions will do when they are executed?

(4) What is the new value in the Program Counter (PC) after the first instruction **MOV R1,#0xFFF** is executed? What can you say about the value in the PC?

(5) Will executing **MOV R1,#0xFFF** affect any of the N, Z, V, C flags? If so, why?

(6) How many memory cycles will be required to completely execute the three instructions? Clearly explain how you derived this result.

(7) Does a program with fewer numbers of instructions always execute faster than one with more instructions?

(8) What are the values of the N, Z, V, C flags immediately after the execution of the last instruction **ADD R1,R2**?

(9) Using the information in the VIP Instruction Set Summary Chart, give the instruction encoding for the mnemonic **ADD R1,R2**.