ELE 538 Problem Set 1 (Fall 1999)

Problems

1 Name the three (3) major functional blocks of any computer system. Name and briefly describe the busses that interconnect them.

2 Fill in the blanks in the following table: (Note that the first line gives the initial values of all the registers.)

Address	Contents	Assembler	A	B	CCR NZVC	X	Y	S	PC	cycles
			00	00	0000	0000	0000	0000	6100	
6100		`ldaa #2A`								
		`oraa 0x6100`								

Table 1: Table to fill in

3 Suppose a "new improved" version of the 6811—the 6811Z—is designed that is completely compatible with the old version (same bus sizes and electrical characteristics, all old instructions work in the identical way) but adds a new register: the Z register which is 24 bits wide. Suppose that that two new instructions `LDZ` and `STZ` are also added. The `LDZ` instruction loads the Z register with the operand which may be specified either as an immediate operand or using the extended addressing mode. Only extended addressing mode is supported for the `STZ` instruction.

Determine the minimum number of bytes to encode the new instructions and number of clock cycles required to execute them.

4 Write assembly language code to modify Accumulator A so that the most significant bit is set to a '1', the least significant bit is cleared to '0' and all the other bits are inverted. (For example, if the initial contents were 11110111, they would become 10001000.)

How many bytes does your machine language code occupy and how many clock cycles are required to execute it?

5 Disassemble the following memory dump:

```
6100 86 12 B7 61 01
```

6 Given:

```
ldx #0x8765
pshx
pulb
pula
```


a) What is the contents of Accumulator D following the execution of the `pula` instruction?

b) What are the values of the NZVC bits of the Condition Code register? (Assume that initially N = 0, Z = 1, V = 1, C = 0.)

7 Given:

```
        .area  data (ABS)
        .org   0x6000
foo:    .db   3, 1, 4, 1, 5, 9
bar:    .db   2, 7, 1, 8, 2, 8
res:    .ds   6
        ldx #foo
        ldy #bar
        ldab #5
loop:   ldaa 0,x
        adda 0,y
        staa 6,y
        inx
        iny
        decb
        bne loop
        swi
```

The program begins execution with the `ldx` instruction.

a) What are the contents of the X and Y index registers following the execution of the `ldy #bar` instruction.

b) What memory location will be written to after the `staa 6,y` instruction is executed the first time. What value will be written.

c) How many times is the loop executed?

d) What is the address of the last memory location modified and what are its final contents?

8 Write an assembly-language program that calculates

$$2 \times \textit{contents of memory location 0x1000} + \textit{contents of 0x1001} - 7 \qquad (1)$$

and place the result in memory location 0x1002.

9 Write an assembly-language program implements the following C code fragment:

```
unsigned char i, j, k, m;  /* Global variables  */

     i = j + k - m;
     if ( i >= 128)
         j++;
     k++
```

10 Write an assembly-language program implements the following C code fragment:

```
typdef struct _list {
     stuct _list * next;
     char * name;
} List_Element_t;

List_Element_t e[10];
List_Element_t *p;
char * cp;
unsigned char count;

p = &e[3];
count = 0;
cp = p->next->next->name;
while(*cp) {
   count++;
   cp++;
}
```

Use the X index register for the pointer `p` and the Y index register for the pointer `cp`.

11 Find a sequence of one or more 68hc11 instructions that execute in exactly 6 clock cycles and do not change the contents of any registers (except the PC) or memory locations (except possibly the stack area). The fewer number of bytes occupied by the machine-language instructions, the better.

12 Find a sequence of one or more 68hc11 instructions that execute in exactly 7 clock cycles and do not change the contents of any registers (except the PC) or memory locations (except possibly the stack area). The fewer number of bytes occupied by the machine-language instructions, the better.

Solutions

1 The main components are:

1. The CPU (Central Processing Unit);
2. The memory system;
3. The I/O (Input/Output) system.

The three components are interconnected by the buses:

Data The Data bus contains the actual data copied from one block to another.

Address The Address Bus contains the address in memory or I/O space that the data comes from or is destined to.

Control The Control Bus determines the direction of data flow and the transfer timing. (It also has auxiliary function signals such as system Reset and interrupts.)

2

Address	Contents	Assembler	A	B	CCR NZVC	X	Y	S	PC	cycles
			00	00	0000	0000	0000	0000	6100	
6100	86 C7	`ldaa #2A`	2A	00	0000	0000	0000	0000	6102	2
6102	8A 61 00	`oraa 0x6100`	AE	00	1000	0000	0000	0000	6105	4

Table 2: Table filled in

3

Instruction	Number bytes	Clock cycles
`ldz #IMMEDIATE`	4	4
`ldz Extended`	3	6
`stz Extended`	3	6

4 The most straight forward solution is:

```
ora #0x80  ;set MSbit to 1 (2 bytes, 2 cycles)
anda #0xFE ;clear LSB (2 bytes, 2 cycles)
eora #07E  ;Invert other bits (2 bytes, 2 cycles)
 ;; TOTAL 6 bytes, 6 cycles
```

Another possible solution (more compact and equal speed) is:

```
coma       ;invert all bits (1 byte, 2 cycles)
ora #0x80  ;set MSbit to 1 (2 bytes, 2 cycles)
anda #0xFE ;clear LSB (2 bytes, 2 cycles)
 ;; TOTAL 5 bytes, 6 cycles
```

A more compact and faster solution (but that is somewhat obscure) is:

```
ora #0x81  ;set MSbit and LSB to 1 (2 bytes, 2 cycles)
eora #07F  ;Invert other bits and invert LSB to 0 (2 bytes, 2 cycles)
 ;; TOTAL 4 bytes, 4 cycles
```

5

```
ldaa #0x12
staa 0x6101
```

6

a) 0x6587

b) N = 1, Z = 0, V = 0, C = 0.

7

a) X is 0x6000, Y is 0x6006.

b) 5 will be written to location 0x600C.

c) 5 times

8

```
ldaa 0x1000
adda 0x1000
adda 0x1001
suba #7
staa 0x1002
```

9

```
     i:  .ds   1
     j:  .ds   1
     k:  .ds   1
     m:  .ds   1

           ldaa  j
           adda  k
           suba  m
           staa  i
           bpl  endIf_1
           inc j
endIf_1:   inc k
```


10

```
ListElementSize = 4   ;Number of bytes in data structure
next            = 0   ;Offset to "next" field
;   Data Declaration Section
        .area  data (ABS)
        .org   0x6000
count:      .ds 1
e:          .ds ListElementSize*10  ; space for 10 data structures

;   Program Section
        .area  main (ABS)
        .org   0x6100
        ldx #e + 3*ListElementSize
        clr count
          ;cp = p->next->next->name;
        ldy next,x
        ldy next,y
        ldy name,y

          ;while (*cp)
loop:   ldaa 0,y
        beq done
        inc count
        inc y
        bra loop
done:   swi
```

11 A simple (but not optimal) solution is:

```
nop   ;1 byte, 2 cycles
nop   ;1 byte, 2 cycles
nop   ;1 byte, 2 cycles ==> TOTAL: 3 bytes, 6 cycles
```

A better solution is:

```
xgdx  ;1 byte, 3 cycles
xgdx  ;1 byte, 3 cycles ==> TOTAL: 2 bytes, 6 cycles
```

12 One solution is:

```
nop   ;1 byte, 2 cycles
nop   ;1 byte, 2 cycles
brn 0 ;2 byte, 3 cycles ==> TOTAL: 4 bytes, 7 cycles
```

A better solution is:

```
psha  ;1 byte, 3 cycles
pula  ;1 byte, 4 cycles ==> TOTAL: 2 bytes, 7 cycles
```