1) Write a code segment to read data from a memory location in the ACCESS bank onto WREG in two alternate forms by using:

Let's show this for PORTA, which is in bank 15 and can also be accessed from the access bank.

a) The BSR register

MOVLB 0xF

MOVF PORTA, 0, 1 (here a = 1 which means BSR specifies the bank used by the instruction)

b) The "a" bit in the instruction

MOVF PORTA, 0, 0

c) Explain the main purpose of why the concept of the ACCESS bank exists in the PIC 18F architecture and its use through the "a" bit is more efficient.

Access bank allows one to access several general purpose registers (GPRs) and all of the special function registers (SFRs) without doing a bank switch. If the program does not need to use a large number of GPRs, then there is no need for bank switching at all. As can be seen from the examples above, bank switching requires an extra instruction and is therefore less efficient and can be more error-prone. The access bank significantly mitigates these problems.

2) A piece of PIC assembly code is given as follows:

IMAGE HERE

What are the contents of NUM1, NUM2, STATUS and Working Register (WREG), right after the execution of the instructions at line 4, 7, 9, 11 and 13?

Assume that

a) NUM1 and NUM2 are located at addresses 0x23 and 0x32, respectively,

b) STATUS = 0xBA , WREG = 0x23, NUM1 = 0x32 and NUM2= 0x23 before the execution of the instruction at line 1,

c) No interrupt occurs during the execution of this code segment.

Line number of last executed instruction	**NUM1**	**NUM2**	**STATUS**	**W**
1	0x23	0x23	0x1A	0x23
4	0x23	0x00	0x1E	0x23

7	0x23	0xFF	0x10	0xFE
9	0x23	0x00	0x07	0x01
11	0x91	0x00	0x03	0x31
13	0x92	0x00	0x10	0x62

3) On a PIC18, if a timer is loaded with 100 (decimal) and counts up to 200 and the timer clock frequency (Fosc/4) is 2MHz, what is the time calculated? Assume that both pre- and post-scalers are equal to 1. Show your work.

100 timer ticks take place. The period of each tick is 1 / 2Mhz = 500 ns. So after 100 ticks, we obtain 500 * 100 = 50000 ns = 50 micro seconds.

4) Calculate the total number of instruction cycles from label a to b in the given program and explain your calculation (just a result gets zero credit):

```
a:
    movlw 0x08
    movwf 0x20
here:
    decfsz     0x20,f
    goto here
    nop
b:
```

The initial movlw and movwf take 1 instr. cycle each (2 so far). decfsz will take 1 cycle and the goto will take 2 cycles for 7 times (3 * 7 = 21). In the 8th time, decfsz will take 3 cycles (because the following instruction is a 2 cycle instruction and the skip occurs). Then nop will take 1 cycle.

So in total: 2 + 21 + 3 + 1 = 27

5) Answer the following questions:
a) Write instructions to set up INT1 with low priority and INT2 high-priority and clear 0x01 to set up as an up counter and load 0x02 with 30H.

```
    clrf 0x01
    movlw 0x30
    movwf 0x02
```

```
movlw 10011000B    ; int2 high, int1 low, enable both, clear both
movwf INTCON3

clrf RCON
bsf RCON, RCON_IPEN_POSITION          ; enable priorities

movlw 0xC0                                    ; enable interrupts
movwf INTCON
```

b) Write ISR for INT1 to count up to 30 and then clear the counter.

```
btfss INTCON3, INTCON3_INT1IF_POSITION
retfie        ; only respond to INT1

bcf INTCON3, INTCON3_INT1IF_POSITION ; clear the interrupt flag

movwf 0x03            ; use this for saving the original value of wreg
movlw 0x30            ; this is the value to compare against

incf 0x01             ; increment
cpfslt 0x01           ; if the contents of 0x01 is less than 0x30, skip the next instruction
clrf 0x01
movf 0x03, 0          ; restore the wreg

retfie
```

c) Write ISR for INT2 to count from 30H to 0, and when the counter reaches zero, reset the counter to 30H. Assume all register are in Access Bank.

```
btfss INTCON3, INTCON3_INT2IF_POSITION
retfie 1      ; only respond to INT2

bcf INTCON3, INTCON3_INT2IF_POSITION ; clear the interrupt flag

decfsz 0x02    ; decrement, if the contents of 0x02 is zero skip the next instruction
retfie 1

movlw 0x30
movwf 0x02     ; reset the counter back to 0x30

retfie 1
```

6) What is the effect of executing the following instructions?

movlw b’11110000’
movwf TRISB

Pins 7:4 of Port B are set as input and the others are set as output

7) A snippet of PIC assembly is given as follows:

CODE HERE

What are the contents of VAR1, STATUS and Working Register (WREG), **right <u>after</u> the execution of the instructions** at lines 2, 3, 5?

0x96 0x5A 0x00
0xF0 0x5A 0x12
0xF0 0x5A 0x01
0xF8 0x00 0x10

8) In PIC18, <u>most</u> of the instructions are 16 bit(s) long and are executed in 1 instruction cycle(s).

9) What will be the decimal value of the data register at address 0x301 after the following code is executed. Assume all data registers initially have a value of zero.

10

10) Assuming that the current value of PC (program counter) is 56 (in decimal), what will be its new value (again in decimal) after the following code is executed:

This question can be quite confusing due to the assembler recomputing the argument to BC before computing the real hardware instruction. If you code it up, you will see that the answer is 18 (or 0x12). But if the hardware could actually use 0x12 as input to BC, then the answer would have been: 60 + 2 + 2*18 = 98.

11) Assume that you want to create a time delay without using timers. You are using a PIC device with 32-Mhz oscillator clock frequency. For this purpose you write the following code snippet:

CODE HERE

32Mhz oscillator clock frequency is equal to 8Mhz instruction frequency as each instruction is executed in 4 oscillator clock cycles. Therefore one instruction cycle takes 1/8 microseconds. The body of time_delay_loop takes 5 instruction cycles (1 + 1 + 1 + 2). Thus, the body of the loop takes 5 / 8 microseconds. We want to spend 10 microseconds there. As a result we must loop 10 / (5 / 8) = 16 times. This is the value of FIND_THIS_NUMBER

12) Imagine an 8-bit timer which is set to start at timer value of zero. The prescaler is set to 1:4 and postscaler is set to 1:2. Assuming that the timer is programmed to generate timer interrupts at overflow, how many times the interrupt will be generated when the timer's input clock makes 65536 pulses?

If both scalers were 1, it would generate 65536 / 256 = 256 interrupts. But the product of the scalers is 4 * 2 = 8. So it will generate 256 / 8 = 32 interrupts.

13) In PIC18, Data Memory is divided into banks of …… bytes long.

256

14) The majority of the access bank in PIC18F4620 (our PIC) contains special function registers.
This is false because it contains an equal number of GPRs and SFRs. See Figure 5-5 in the datasheet (page 62).

15) A program running on the PIC device cannot write data to the program memory during execution. It has to be done offline.

False. This is what table writes are for.