BLG351E Experiment 1 “Basic Assembly Coding” REPORT	CRN	12635
	Group	Group 11
	Name #1	Fatih Baskın
	Name #2	Mehmet Eymen Ünay
	Name #3	Nada Malek
	Name #4	Rojen Arda Şeşen

Q1) (40 pts.) How to use PORT1 and PORT2 for I/O operations? Summarize your findings from the experiment.

For setting ports as OUTPUT, individual bits in P1DIR or P2DIR should be set as 1.
Initially, we must set the port directions as OUTPUT. In order to do so, we must set &P1DIR and &P2DIR as 0xFFFF as follows:

```
mov.w  #FFFFh, &P1DIR
mov.w  #FFFFh, &P2DIR
```

Afterwards, we load P1OUT and P2OUT registers with the value we want to give as output.

For setting ports as INPUT, individual bits in P1DIR or P2DIR should be set as 0.
Initially, we must set the port directions as INPUT. In order to do so, we must set &P1DIR and &P2DIR as 0x0000 as follows:

```
mov.w  #0000h, &P1DIR
mov.w  #0000h, &P2DIR
```

Afterwards, we read the values from P1IN and P2IN registers.

Q2) (30 pts.) Assume that you used the following code to solve Part 2. How the bits of Status Register changes after each operation?

			V	N	Z	C
SetupP1	mov.b	#11111111b, &P1DIR	-	-	-	-
	mov.b	#11111111b, &P2DIR	-	-	-	-
start	mov.b	#00000001b, &P1OUT	-	-	-	-
	mov.b	#00000001b, &P2OUT	-	-	-	-
	mov.b	#0d, R6	-	-	-	-
MainLoop	mov.w	#050000, R15	-	-	-	-
L1	dec.w	R15	0	1	0	1
	jnz	L1	0	1	0	1
	rla.b	&P1OUT	0	0	0	0
	rla.b	&P2OUT	0	0	0	0
	inc.b	R6	0	0	0	0
	cmp	#04d, R6	0	1	0	0
	jeq	start	0	1	0	0
	jmp	MainLoop	0	1	0	0

Q3) (30 pts.) What is the difference between **mov.b #00001111b, &P1OUT**, **bis.b #00001111b, &P1OUT** and **bic.b #00001111b, &P1OUT** and **bit.b #00001111b, &P1OUT**? Let the initial value of P1OUT be 10101010. What will be the new value after each operation?

The reason why we use bic, bit or bis rather than mov is that when mov operation is applied, all the bits in the destination register changes but using other operations on only the necessary bits could be changed.

mov operation copies the sets the bits in the destination register as equal to the bits in the source register, i.e. assignment.

bis operation ORs the source and destination register and stores the result in destination register. => src OR dst -> dst

bic operation ANDs complement of source and destination register. => not src AND dest -> dest

bit operation ANDs the source and destination registers and stores the result in destination register.
=> src AND dest -> dest

mov.b => &P1OUT = #00001111b
bis.b => &P1OUT = #10101111b
bic.b => &P1OUT = #10100000b
bit.b => &P1OUT = #00001010b