Solutions to Worksheet 3

1) a) $M[0x100 + 8] \Rightarrow M[0x108] \rightarrow \underline{0x13}$

b) $M[0x100 + 0xC] \Rightarrow M[0x10C] \rightarrow \underline{0x11}$

c) 260 (%ecx, %edx)

$M[260 + \underline{0x1} + \underline{0x3}] \Rightarrow M[260 + 1 + 3]$

$\Rightarrow M[264_{10}]$

		Remainder
16	264_{10}	
16	16	8
	1	0

$264_{10} \Rightarrow 0x108_{16}$

$\Rightarrow M[0x108] \Rightarrow \underline{0x13}$

d) 9 (%eax, %ecx, -1)

$M[9 + 0x100 + 0x1 * -1] \Rightarrow M[0x108]$

$\Rightarrow 0x13$

c) leal 4 (%ecx), %edx

Destination → %edx

Value → M[4 + 0x1]

→ M[0x5].

Value stored in edx ⇒ 0x5

3)

One advantage of using leal

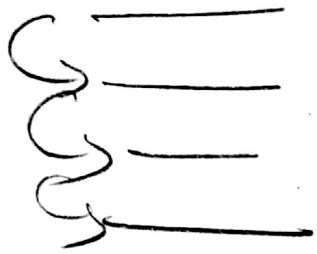
Suppose you want to subtract 3 from %ebx and store it in %eax.

```
leal -3(%ebx), %eax.
```

(or)

```
subl $3, %ebx
movl %ebx, eax.
```

CONTROL



So far → Sequential execution

Straight line is not ideal, we need to 'jump' from instruction to another based on the output of some other operation. //

Extra Registers!

← How to we know the output of a previous operation

Single bit

Condition code registers

Store some attribute about the outcome of the most recent arithmetic or logical operation.

CF → Carry Flag ⇒ Most recent operation generates a carry out of the most significant bit

2 n-bit numbers, and you get a (n+1) bit number, CF = 1

Eg 4-bit numbers

$$\begin{array}{r} 1000 \\ +\ 1000 \\ \hline (1)0000 \end{array}$$

ZF → Zero Flag ⇒ Most Recent op gives a zero.

SF → Sign Flag ⇒ Most Recent op gives a negative number //

(or) in other words,
When the MSB is 1,

SF = 1

OF → Overflow Flag ⇒ Most recent op causes a 2's complement overflow / (both positive and negative)

127 → 0111 1111

+1 → 0000 0001

1 000 0000 → -128.

OF = 1

2 cases when CF is set

When adding 2 n-bit numbers if the $n+1^{th}$ bit is set

```
 1111          CF=1
(+) 0001
-----
(1) 0000
-----
```

When subtraction of 2 numbers requires a borrow into the MSB

```
 ↱ 1 1 1 1
 - 0 0 0 0
(-) 0 0 0 1
-----------
   1 1 1 1
-----------
```

CF = 1

Otherwise,

```
 0 1 1 1
 0 0 0 1        CF = 0
---------
 1 0 0 0
---------
```

```
 1 0 0 0
-0 0 0 1
---------
 0 1 1 1   ⇒  CF = 0
---------
```

Overflow Flag => 2 cases where it is set

If the sum of 2 numbers with the sign bit off (MSB = 0) => gives a number with the sign bit on (MSB = 1)

Eg: 127 [0]111 1111
+ 1 [0]000 0001
[1]000 0000 → −128

OF = 1

If the sum of 2 numbers with the sign bit on (MSB = 1) => give a number with the sign bit off (MSB = 0),

-8 [1]000
-8 [1]000
[0]000 → CF = 1

And also

1000 -8
0001 +1
1001 -7 => OF = 0

CF = 1

$t = a + b$

CF = if (unsigned) $t <$ unsigned (a).

ZF = if $t = 0$

SF = if $t < 0$

OF = $(a < 0 == b < 0)$ && $(t < 0 != a < 0)$

$a < 0 == b < 0$: a and b have the sign

$t < 0 != a < 0$: t and a have opposite signs