TEJ4MO

<u>PIC16F684 Microcontroller</u>

<u>Working Register</u>: where everything goes through when any operation is executed

<u>STATUS Register</u>:

1. Where all the flags are stored.
2. Indicates the important information about the previous function executed.
3. Indicates which bank is being accessed.

	Bit 7	*Bit 6*	*Bit 5*	*Bit 4*	*Bit 3*	*Bit 2*	*Bit 1*	*Bit 0*
STATUS	IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
	0	0	1 or 0	1 or 0	1 or 0	1 or 0	1 or 0	1 or 0

IRP and RP1 are reserved and should be 0 at all times.

RP0 bit: Indicates which bank is being registered, 0 for BANK0 and 1 for BANK1.

$\overline{\boldsymbol{TO}}$, $\overline{\boldsymbol{PD}}$ are unnecessary for this tutorial.

Z bit: indicates whether the **result of the logic or arithmetic function** is 0. It outputs 1 when it is 0, it outputs 0 when the result is anything but 0.

IT WILL NOT GIVE 1 TO movlw 0 BECAUSE THAT IS NOT AN ARITHATIC FUNCTION

DC bit: Indicates whether there was a **carry** or **borrow** from the lower byte from the second byte.

C bit: Indicates whether there is a **9th bit** required due to a carry or borrow.

Carry: When the result requires another digit.

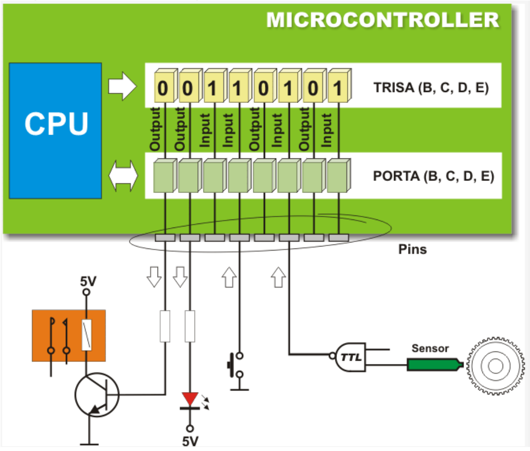
Borrow: When the result needs to borrow from its next digit.

<u>Input / Output Pin Configuration</u>

CMCON0: Comparator, it should be set to 00000111 for the tutorials.

ANSEL: Configures whether the ports are analog or digital. 0 = digital, 1 = analog
The PIC16F684 defaults all ports to analog!

Analog = Voltage can be anywhere between the maximum and 0.
Digital = Voltage is either maximum or 0.



TRISA / TRISC
These indicates whether the corresponding ports, the last letter of register name indicates the specific port are output or input.
0 for output, 1 for input

PORTA / PORTC
The respective ports and pins of the chip.
There are 5 pins on each port for PIC16F684

movlw x
moves literal value to working register, x must be an 8 bit number, under 256

movwf f
moves working register value to f, file.

movfw f
moves file value to the working register.

addwf f, d d takes: w = working register or f = file
adds working register value to f, file and locate it in d

subwf f, d d takes: w = working register or f = file
subtract working register from f, file, and put it in d

comf f, d d takes: w = working register or f = file
one’s complement f, file, and put it in d

call
call a subroutine

clrf f
clear the value of the file, setting it to 0,

clrw
clear the value of the WREG, setting it to 0

btfss f, b
bit test file, skip if set

btfsc f, b
bit test file, skip if clear

It skips the next instruction depending on the bit of the file.

nop = no operations

buttons give high value when it is not pressed, while it gives a low value when it is pressed.