The PIC Family: Speed

- Can use crystals, clock oscillators, or even an RC circuit.
- Some PICs have a built in 4MHz RC clock, Not very accurate, but requires no external components!
- Instruction speed = 1/4 clock speed (Tcyc = 4 * Tclk)
- All PICs can be run from DC to their maximum specified speed:

12C50x	4MHz
12C67x	10MHz
16Cxxx	20MHz
17C4x / 17C7xxx	33MHz
18Cxxx	40MHz

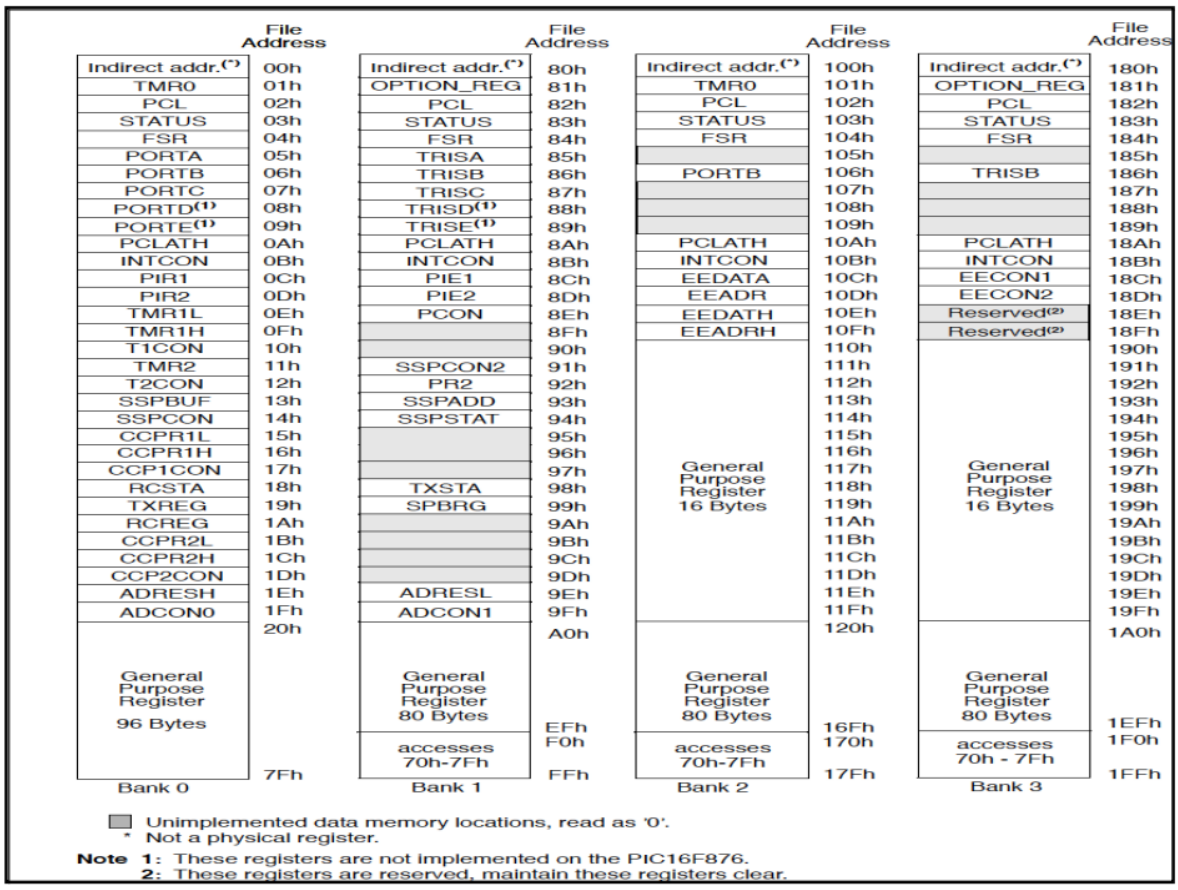
PIC 16F877 Pin IN/Out

- The chip can be obtained in different packages, such as conventional 40-pin DIP (Dual In-Line Package), square surface mount or socket format.
- Most of the pins are for input and output, and arranged as 5 ports: port A (5 pins), port B(8), C(8), D(8) and E(3), giving a total of 32 I/O pins.
  - These can all operate as simple digital I/O pins, but most have more than one function.
  - The mode of operation of each is selected by initializing various control registers within the chip.
  - Note, in particular, that Ports A and E become ANALOGUE INPUTS by default (on power up or reset), so they have to set up for digital I/O if required.

PIC 16F877 Pin IN/Out – cont.

- Port B is used for downloading the program to the chip flash ROM (RB6 and RB7), and RB0 and RB4–RB7 can generate an interrupt.
- Port C gives access to timers and serial ports.
- Port D can be used as a slave port, with Port E providing the control pins for this function.

- There are 512 RAM File Register addresses (0–1FFh), which are organized in 4 banks (0–3), each bank containing 128 addresses.
- The default (selected on power up) is Bank 0 which is numbered from 0 to 7Fh, Bank 1 from 80h to FFh and so on.
- These contain both Special Function Registers (SFRs), which have a dedicated purpose, and the General Purpose Registers (GPRs).



Bank 0	File Address	Bank 1	File Address	Bank 2	File Address	Bank 3	File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD(1)	08h	TRISD(1)	88h		108h		188h
PORTE(1)	09h	TRISE(1)	89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(2)	18Eh
TMR1H	0Fh		8Fh	EEADRH	10Fh	Reserved(2)	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h	General Purpose Register 16 Bytes	117h	General Purpose Register 16 Bytes	197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch		9Ch		11Ch		19Ch
CCP2CON	1Dh		9Dh		11Dh		19Dh
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
General Purpose Register 96 Bytes	20h	General Purpose Register 80 Bytes	A0h	General Purpose Register 80 Bytes	120h	General Purpose Register 80 Bytes	1A0h
accesses 70h-7Fh	7Fh	accesses 70h-7Fh	EFh F0h FFh	accesses 70h-7Fh	16Fh 170h 17Fh	accesses 70h - 7Fh	1EFh 1F0h 1FFh

Unimplemented data memory locations, read as '0'.
* Not a physical register.

Note 1: These registers are not implemented on the PIC16F876.
2: These registers are reserved, maintain these registers clear.

Indirect Addressing:

- Full 8 bit register address is written the special function register FSR
- INDF is used to get the content of the address pointed by FSR
- Exp : A sample program to clear RAM locations H'20' – H'2F:

```
        MOVLW 0x20 ;initialize pointer
        MOVWF FSR ;to RAM
NEXT    CLRF INDF ;clear INDF register
        INCF FSR,F  ;inc pointer
        BTFSS FSR,4 ;all done?
        GOTO NEXT ;no clear next
CONTINUE
: ;
```

b:4

0010 0000 -> 0011 0000

PIC Family Control Registers

- Uses a series of “Special Function Registers” for controlling peripherals and PIC behaviors.
  - **STATUS** ➔ Bank select bits, ALU bits (zero, borrow, carry)
  - **INTCON** ➔ Interrupt control: interrupt enables, flags, etc.
  - **OPTION_REG** ➔ contains various control bits to configure the TMR0 prescaler/WDT postscaler ,the External INT Interrupt, TMR0 and the weak pull-ups on PORTB

Special Function Register
"STATUS Register"

R/W-0	R/W-0	R/W-0	R-1	R-1	R/W-x	R/W-x	R/W-x
IRP	RP1	RP0	$\overline{TO}$	$\overline{PD}$	Z	DC	C
bit 7							bit 0

bit 7 **IRP:** Register Bank Select bit (used for indirect addressing)
1 = Bank 2, 3 (100h - 1FFh)
0 = Bank 0, 1 (00h - FFh)

bit 6-5 **RP1:RP0**: Register Bank Select bits (used for direct addressing)
1 1 = Bank 3 (180h - 1FFh)
1 0 = Bank 2 (100h - 17Fh)
0 1 = Bank 1 (80h - FFh)
0 0 = Bank 0 (00h - 7Fh)
Each bank is 128 bytes

bit 4 **$\overline{TO}$**: Time-out bit
1 = After power-up, CLRWDT instruction, or SLEEP instruction
0 = A WDT time-out occurred

bit 3 **$\overline{PD}$**: Power-down bit
1 = After power-up or by the CLRWDT instruction
0 = By execution of the SLEEP instruction

bit 2 **Z**: Zero bit
1 = The result of an arithmetic or logic operation is zero
0 = The result of an arithmetic or logic operation is not zero

bit 1 **DC**: Digit carry/$\overline{borrow}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions)
(for $\overline{borrow}$, the polarity is reversed)
1 = A carry-out from the 4th low order bit of the result occurred
0 = No carry-out from the 4th low order bit of the result

bit 0 **C**: Carry/$\overline{borrow}$ bit (ADDWF, ADDLW, SUBLW, SUBWF instructions)
1 = A carry-out from the Most Significant bit of the result occurred
0 = No carry-out from the Most Significant bit of the result occurred

Note: For $\overline{borrow}$, the polarity is reversed. A subtraction is executed by adding the two's complement of the second operand. For rotate (RRF, RLF) instructions, this bit is loaded with either the high, or low order bit of the source register.

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Special Function Register

"INTCON Register"

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-x
GIE	PEIE	TMR0IE	INTE	RBIE	TMR0IF	INTF	RBIF
bit 7							bit 0

bit 7 **GIE:** Global Interrupt Enable bit
1 = Enables all unmasked interrupts
0 = Disables all interrupts

bit 6 **PEIE**: Peripheral Interrupt Enable bit
1 = Enables all unmasked peripheral interrupts
0 = Disables all peripheral interrupts

bit 5 **TMR0IE**: TMR0 Overflow Interrupt Enable bit
1 = Enables the TMR0 interrupt
0 = Disables the TMR0 interrupt

bit 4 **INTE**: RB0/INT External Interrupt Enable bit
1 = Enables the RB0/INT external interrupt
0 = Disables the RB0/INT external interrupt

bit 3 **RBIE**: RB Port Change Interrupt Enable bit
1 = Enables the RB port change interrupt
0 = Disables the RB port change interrupt

bit 2 **TMR0IF**: TMR0 Overflow Interrupt Flag bit
1 = TMR0 register has overflowed (must be cleared in software)
0 = TMR0 register did not overflow

bit 1 **INTF**: RB0/INT External Interrupt Flag bit
1 = The RB0/INT external interrupt occurred (must be cleared in software)
0 = The RB0/INT external interrupt did not occur

bit 0 **RBIF**: RB Port Change Interrupt Flag bit
1 = At least one of the RB7:RB4 pins changed state; a mismatch condition will continue to set the bit. Reading PORTB will end the mismatch condition and allow the bit to be cleared (must be cleared in software).
0 = None of the RB7:RB4 pins have changed state

We should write this

MOVF PORTB, F

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z	1,2
CLRF	f	Clear f	1	00 0001 1fff ffff	Z	2
CLRW	-	Clear W	1	00 0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z	1,2
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff		1,2,3
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00 1111 dfff ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1,2
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z	1,2
MOVWF	f	Move W to f	1	00 0000 1fff ffff		
NOP	-	No Operation	1	00 0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF	f, b	Bit Clear f	1	01 00bb bfff ffff		1,2
BSF	f, b	Bit Set f	1	01 01bb bfff ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01 10bb bfff ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01 11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW	k	Add literal and W	1	11 111x kkkk kkkk	C,DC,Z	
ANDLW	k	AND literal with W	1	11 1001 kkkk kkkk	Z	
CALL	k	Call subroutine	2	10 0kkk kkkk kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00 0000 0110 0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to address	2	10 1kkk kkkk kkkk		
IORLW	k	Inclusive OR literal with W	1	11 1000 kkkk kkkk	Z	
MOVLW	k	Move literal to W	1	11 00xx kkkk kkkk		
RETFIE	-	Return from interrupt	2	00 0000 0000 1001		
RETLW	k	Return with literal in W	2	11 01xx kkkk kkkk		
RETURN	-	Return from Subroutine	2	00 0000 0000 1000		
SLEEP	-	Go into standby mode	1	00 0000 0110 0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from literal	1	11 110x kkkk kkkk	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	11 1010 kkkk kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF PORTB, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.
2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 module.
3: If Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is

- BTFSC (Bit Test and Skip if Clear) and BTFSS (Bit Test and Skip if Set) are used to test the bit and skip the next instruction, or not, according to the state of the bit tested.
- DECFSZ and INCFSZ embody a commonly used test – decrement or increment a register and jump depending on the effect of the result on the zero flag (Z is set if result 0).

- NOP simply does nothing for one instruction cycle (four clock cycles).
  – very useful for putting short delays in the program
- SLEEP stops the program, such that it can be restarted with an external interrupt.

OPTIONAL INSTRUCTIONS

- TRIS was an instruction originally provided to make port initialization simpler.
  - It selects register bank 1 so that the TRIS data direction registers (TRISA, TRISB, etc.) can be loaded with a data direction code (e.g. 0→output).
- The manufacturer no longer recommends use of this instruction, although it is still supported
- The assembler directive BANKSEL can be used
  - It gives more flexible access to the registers in banks 1, 2, 3.
- The other option is to change the bank select bits in the STATUS register directly, using BSF and BCF.
- OPTION, providing special access to the OPTION register, is the other instruction, which is no longer recommended.
  - It can be replaced by BANKSEL to select bank 1 which contains the OPTION register, which can then be accessed directly.

-CONFIG

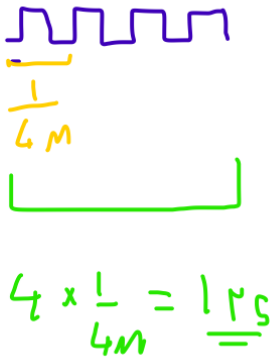
Chip Configuration Word

Bit	Label	Function	Default	Enabled	Typical
15	–	None	0	x	0
14	–	None	0	x	0
13	CP1	Code protection	1	0	1
12	CP0	(4 levels)	1	0	1
11	DEBUG	In-circuit debugging (ICD)	1	0	0
10	–	None	1	x	1
9	WRT	Program memory write enable	1	1	1
8	CPD	EEPROM data memory write protect	1	0	1
7	LVP	Low-voltage programming enable	1	1	0
6	BODEN	Brown-out reset (BoR) enable	1	1	0
5	CP1	Code protection (CP)	1	0	1
4	CP0	(repeats)	1	0	1
3	PWRTE	Power-up timer (PuT) enable	1	0	0
2	WDTE	Watchdog timer (WdT) enable	1	1	0
1	FOSC1	Oscillator type select	1	x	0
0	FOSC0	RC = 11, HS = 10, XT = 01, LP = 00	1	x	1

Default = 3FFF (RC clock, PuT disabled, WdT enabled).
Typical RC clock = 3FF3 (RC clock, ICD disabled, PuT enabled, WdT disabled).
Typical XT clock = 3731 (XT clock, ICD enabled, PuT enabled, WdT disabled).

CRYSTAL (XTAL) OSCILLATOR

- Used for greater precision
  - uses the hardware timers to make accurate measurements
  - generate precise output signals
- Normally, it is connected across the clock pins with a pair of small capacitors (15 pF) to stabilize the frequency.
- The crystal acts as a self-contained resonant circuit, where the quartz or ceramic crystal vibrates at a precise frequency when subject to electrical stimulation.
- A convenient value (used in our examples later) is 4 MHz; this gives an instruction cycle time of 1 µs
  - This is the maximum frequency allowed for the XT configuration setting.
- Operating at higher frequency requires the selection of the HS configuration option.
- Each instruction takes four clock cycles



Input/output ports

- There are five parallel ports in the PIC 16F877, labelled A–E.
- All pins can be used as bit- or byte-oriented digital input or output with
  - Some having alternate functions depending on the initialization of the relevant control registers.
- The TRIS (data direction) register bits in bank 1, default to 1, setting the ports B, C and D as inputs.
- Ports A and E are set to ANALOGUE INPUT by default, because the analogue control register ADCON1 in bank 1 defaults to 0 - - - 0000.
- To set up these ports for digital I/O, this register must be loaded with the code x - - - 011x (x don't care),e.g. 06h.
- ADCON1 can be initialized with bit codes that give a mixture of analogue and digital I/O on Ports A and E.
- ADCON1 is in bank 1 so BANKSEL is needed to access it.

if we put value in portA register 8-bit more than 5bit it will remove the upper bits values and just save the lower 5 bits

ADC
0-4: A
5-7: E

As you are only using the port in digital IO mode, set ADCON1 to a value of either 0x06 or 0x07 which makes all the pins digital. The LSB is ignored when 0x06 or 0x07 are used so both values do exactly the same thing.

Port functions

	Bits	Pins	Alternate function/s	Bit	Default
Port A	6	RA0–RA5	Analogue inputs	0,1,2,3,5	Analogue
			Timer0 clock input	4	Input
			Serial port slave select input	5	
Port B	8	RB0–RB7	External interrupt	0	Digital
			Low-voltage programming input	3	I/O
			Serial programming	6,7	
			In-circuit debugging	6,7	
Port C	8	RC0–RC7	Timer1 clock input/output	0,1	Digital
			Capture/Compare/PWM	1,2	I/O
			SPI, I^2C synchronous clock/data	3,4,5	
			USART asynchronous clock/data	6,7	
Port D	8	RD0–RD7	Parallel slave port data I/O	0–7	Digital I/O
Port E	3	RE0–RE2	Analogue inputs	0,1,2	Analogue
			Parallel slave port control bits	0,1,2	Input

Interrupt Control Registers

- The registers involved in interrupt handling are INTCON, PIR1, PIR2, PIE1, PIE2 and PCON.
- Interrupts are external hardware signals which force the MCU to suspend its current process, and carry out an Interrupt Service Routine (ISR).
- In PIC when an interrupt occurs the program execution jumps to address 004.
- By default, interrupts are disabled.
- If interrupts are to be used
  - the main program start address needs to be 0005, or higher, and a 'GOTO start' (or similar label) placed at address 0000.
  - A 'GOTO ISR' instruction can then be placed at 004, using the ORG directive, which sets the address at which the instruction will be placed by the assembler.
  - The Global Interrupt Enable bit (INTCON, GIE) must be set to enable the interrupt system.
  - The individual interrupt source is then enabled.
  - For example, the bit INTCON, **T0IE** is set to enable the Timer0 overflow to trigger the interrupt sequence.
  - When the timer overflows, INTCON, **T0IF** (Timer0 Interrupt Flag) is set to indicate the interrupt source, and the ISR called.
- The flags can be checked by the ISR to establish the source of the interrupt, if more than one is enabled.

Interrupt sources and control bits

Source	Enable Bit Set	Flag Bit Set	Interrupt Trigger Event
TMR0	INTCON,5	INTCON,2	Timer0 count overflowed
RB0	INTCON,4	INTCON,1	RB0 input changed (also uses INTEDG)
RB4-7	INTCON,3	INTCON,0	Port B high nibble input changed
Peripherals	INTCON,6		
TMR1	PIE1,0	PIR1,0	Timer1 count overflowed
TMR2	PIE1,1	PIR1,1	Timer2 count matched period register PR2
CCP1	PIE1,2	PIR1,2	Timer1 count captured in or matched CCPR1
SSP	PIE1,3	PIR1,3	Data transmitted or received in Synchronous Serial Port
TX	PIE1,4	PIR1,4	Transmit buffer empty in Asynchronous Serial Port
RC	PIE1,5	PIR1,5	Receive buffer full in Asynchronous Serial Port
AD	PIE1,6	PIR1,6	Analogue to Digital Conversion completed
PSP	PIE1,7	PIR1,7	A read or write has occurred in the Parallel Slave Port
CCP2	PIE2,0	PIR2,0 CCPR2	Timer2 count captured in or matched
BCL	PIE2,3	PIR2,3	Bus collision detected in SSP (I^2C mode)
EE	PIE2,4	PIR2,4	Write to EEPROM memory completed

Macros, Special Instructions, Assembler Directives
Another structured

- Supplementary instructions

S. Instruction		Meaning	Assembler Code	
BZ	addlab	Branch to destination (address label) if result of previous operation zero	BTFSC GOTO	STATUS,Z addlab
BNZ	addlab	Branch to destination (address label) if result of previous operation not zero	BTFSS GOTO	STATUS,Z addlab
BC	addlab	Branch to destination (address label) if carry set	BTFSC GOTO	STATUS,C addlab
BNC	addlab	Branch to destination (address label) if carry not set	BTFSS GOTO	STATUS,C addlab
NEG	num1	Negate (2s complement) a file register (labelled num1)	COMF INCF	num1 num1
TSTF	num1	Test a file register (labelled num1) to modify status bits	MOVF	num1

- Timer0 uses an 8-bit register
  - TMR0, file register address 01.
  - The register counts from 0 to 255, and then rolls over to 00 again.
  - When the register goes from FF to 00, an overflow flag, T0IF, bit 2 in the Interrupt Control Register INTCON, address 0B, is set.

Timers – cont.

- A timer can work as a counter.
  - Counts external pulses
  - timers can also be used as counters.
- Timer0, can be controlled by a pre-scaler, see next slide.
- The pre-scaler is a divide by N register, where N 2, 4, 8, 16, 32, 64, 128 or 256, meaning that the output count rate is reduced by this factor.
- This extends the count period or total count by the same ratio, giving a greater range to the measurement.
- The watchdog timer interval can also be extended, if this is selected as the clock source.
- The pre-scale select bits, and other control bits for Timer0 are found in OPTION_REG.

N=2 -> MAX_TIME = MAX_TIME * 2

4MHz -> I MHz clk inst.
1u time
N = 2
Then 1u -> 2u

OPTION_REG REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
$\overline{\text{RBPU}}$	INTEDG	T0CS	T0SE	PSA	PS2	PS1	PS0
bit 7							bit 0

bit 7 **$\overline{\text{RBPU}}$**

bit 6 **INTEDG**

bit 5 **T0CS**: TMR0 Clock Source Select bit
1 = Transition on T0CKI pin
0 = Internal instruction cycle clock (CLKOUT)

bit 4 **T0SE**: TMR0 Source Edge Select bit
1 = Increment on high-to-low transition on T0CKI pin
0 = Increment on low-to-high transition on T0CKI pin

bit 3 **PSA**: Prescaler Assignment bit
1 = Prescaler is assigned to the WDT
0 = Prescaler is assigned to the Timer0 module

bit 2-0 **PS2:PS0**: Prescaler Rate Select bits

Bit Value	TMR0 Rate	WDT Rate
000	1 : 2	1 : 1
001	1 : 4	1 : 2
010	1 : 8	1 : 4
011	1 : 16	1 : 8
100	1 : 32	1 : 16
101	1 : 64	1 : 32
110	1 : 128	1 : 64
111	1 : 256	1 : 128

Each of the PORTB pins has a weak internal pull-up. A single control bit can turn on all the pull-ups. This is performed by clearing bit **RBPU** (OPTION_REG<7>). The weak pull-up is automatically turned off when the port pin is configured as an output. The pull-ups are disabled on a Power-on Reset.

RB0/INT is an external interrupt input pin and is configured using the **INTEDG** bit (OPTION_REG<6>).

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Assuming a clock frequency of 4 MHz and a prescaler value of 256, which is a common configuration for Timer0 in the PIC16F877A, each Timer0 count would take approximately 64 microseconds to complete. This can be calculated as follows:

The clock period at 4 MHz is 0.25 microseconds (1/4 MHz).
The prescaler value of 256 divides the clock frequency by 256, resulting in a Timer0 clock frequency of 15.625 kHz (4 MHz / 256).
Each Timer0 count therefore takes 1 / 15.625 kHz, or approximately 64 microseconds (1/15625 seconds).
To count from 0 to 255, Timer0 would need to complete 256 counts, so the total time it would take for Timer0 to count from 0 to 255 would be 256 x 64 microseconds, or approximately 16.384 milliseconds.

Timers – cont.

- Timer1 is a 16-bit counter, consisting of TMR1H and TMR1L (addresses 0E AND 0F).
  - When the low byte rolls over from FF to 00, the high byte is incremented.
  - The maximum count is therefore 65535, which allows a higher count without sacrificing accuracy.
- Timer2 is an 8-bit counter (TMR2) with a 4-bit pre-scaler, 4-bit post-scaler and a comparator.
  - It can be used to generate Pulse Width Modulated (PWM) output which is useful for driving DC motors and servos, among other things.
- These timers can also be used in capture and compare modes, which allow external signals to be more easily measured.

T1CON: TIMER1 CONTROL REGISTER (ADDRESS 10h)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	T1CKPS1	T1CKPS0	T1OSCEN	$\overline{\text{T1SYNC}}$	TMR1CS	TMR1ON
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **T1CKPS1:T1CKPS0:** Timer1 Input Clock Prescale Select bits
11 = 1:8 prescale value
10 = 1:4 prescale value
01 = 1:2 prescale value
00 = 1:1 prescale value

bit 3 **T1OSCEN**: Timer1 Oscillator Enable Control bit
1 = Oscillator is enabled
0 = Oscillator is shut-off (the oscillator inverter is turned off to eliminate power drain)

bit 2 **$\overline{\text{T1SYNC}}$**: Timer1 External Clock Input Synchronization Control bit
When TMR1CS = 1:
1 = Do not synchronize external clock input
0 = Synchronize external clock input
When TMR1CS = 0:
This bit is ignored. Timer1 uses the internal clock when TMR1CS = 0.

bit 1 **TMR1CS**: Timer1 Clock Source Select bit
1 = External clock from pin RC0/T1OSO/T1CKI (on the rising edge)
0 = Internal clock (FOSC/4)

bit 0 **TMR1ON**: Timer1 On bit
1 = Enables Timer1
0 = Stops Timer1

ADC

- Registers used
  - ADCON0
  - ADCON1
  - The output from the converter is stored in ADRESH (analogue to digital conversion result, high byte) and ADRESL (low byte).

8-bit Conversion

- The 16F877 MCU has eight analogue inputs available, at RA0, RA1, RA2, RA3, RA5, RE0, RE1 and RE2.
- RA2 and RA3 may be used as reference voltage inputs, setting the minimum and maximum values for the measured voltage range.
- These inputs default to analogue operation, so the register ADCON1 has to be initialized explicitly to use these pins for digital input or output.
- The ADC converts an analogue input voltage (e.g. 0 – 2.56V) to 10-bit binary, but only the upper 8 bits of the result are used, giving a resolution of 10 mV per bit ((1/256) X 2.56 V).

at least we need 1.6 Vc

from which bit we want to read the analog signal

ADCON0 REGISTER (ADDRESS 1Fh)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0
ADCS1	ADCS0	CHS2	CHS1	CHS0	GO/$\overline{\text{DONE}}$	—	ADON
bit 7							bit 0

bit 7-6 **ADCS1:ADCS0:** A/D Conversion Clock Select bits (ADCON0 bits in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	Frc (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	Frc (clock derived from the internal A/D RC oscillator)

bit 5-3 **CHS2:CHS0:** Analog Channel Select bits
000 = Channel 0 (AN0)
001 = Channel 1 (AN1)
010 = Channel 2 (AN2)
011 = Channel 3 (AN3)
100 = Channel 4 (AN4)
101 = Channel 5 (AN5)
110 = Channel 6 (AN6)
111 = Channel 7 (AN7)

Note: The PIC16F873A/876A devices only implement A/D channels 0 through 4; the unimplemented selections are reserved. Do not select any unimplemented channels with these devices.

bit 2 **GO/$\overline{\text{DONE}}$:** A/D Conversion Status bit
When ADON = 1:
1 = A/D conversion in progress (setting this bit starts the A/D conversion which is automatically cleared by hardware when the A/D conversion is complete)
0 = A/D conversion not in progress

bit 1 **Unimplemented:** Read as '0'

bit 0 **ADON:** A/D On bit
1 = A/D converter module is powered up
0 = A/D converter module is shut-off and consumes no operating current

1: when start convert then we will wait it to set to 0 when finish

ADCON1 REGISTER (ADDRESS 9Fh)

R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	ADCS2	—	—	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit
1 = Right justified. Six (6) Most Significant bits of ADRESH are read as '0'.
0 = Left justified. Six (6) Least Significant bits of ADRESL are read as '0'.

bit 6 **ADCS2:** A/D Conversion Clock Select bit (ADCON1 bits in shaded area and in **bold**)

ADCON1 <ADCS2>	ADCON0 <ADCS1:ADCS0>	Clock Conversion
0	00	Fosc/2
0	01	Fosc/8
0	10	Fosc/32
0	11	FRC (clock derived from the internal A/D RC oscillator)
1	00	Fosc/4
1	01	Fosc/16
1	10	Fosc/64
1	11	FRC (clock derived from the internal A/D RC oscillator)

bit 5-4 **Unimplemented:** Read as '0'

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits

PCFG <3:0>	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0	VREF+	VREF-	C/R
0000	A	A	A	A	A	A	A	A	VDD	VSS	8/0
0001	A	A	A	A	VREF+	A	A	A	AN3	VSS	7/1
0010	D	D	D	A	A	A	A	A	VDD	VSS	5/0
0011	D	D	D	A	VREF+	A	A	A	AN3	VSS	4/1
0100	D	D	D	D	A	D	A	A	VDD	VSS	3/0
0101	D	D	D	D	VREF+	D	A	A	AN3	VSS	2/1
011x	D	D	D	D	D	D	D	D	—	—	0/0
1000	A	A	A	A	VREF+	VREF-	A	A	AN3	AN2	6/2
1001	D	D	A	A	A	A	A	A	VDD	VSS	6/0
1010	D	D	A	A	VREF+	A	A	A	AN3	VSS	5/1
1011	D	D	A	A	VREF+	VREF-	A	A	AN3	AN2	4/2
1100	D	D	D	A	VREF+	VREF-	A	A	AN3	AN2	3/2
1101	D	D	D	D	VREF+	VREF-	A	A	AN3	AN2	2/2
1110	D	D	D	D	D	D	D	A	VDD	VSS	1/0
1111	D	D	D	D	VREF+	VREF-	D	A	AN3	AN2	1/2

A = Analog input D = Digital I/O
C/R = # of analog input channels/# of A/D voltage references

A → Vref

will read the analog value as voltage in range

[Vref-, Vref+]

Vref+ = 5, Vref- = 0

then

(5 - 0) / 2 ^ 10 = 0.005

then if the result is
1000000000 = 512

then the voltage was
512 * 0.005 = 2.56 V

	ADRESH	ADRESL
ADFM = 1 Right justified	0000 00RR	RRRR RRRR
ADFM = 0 Left justified	RRRR RRRR	RR00 0000

R = Result bits

We can ignore ADRESL

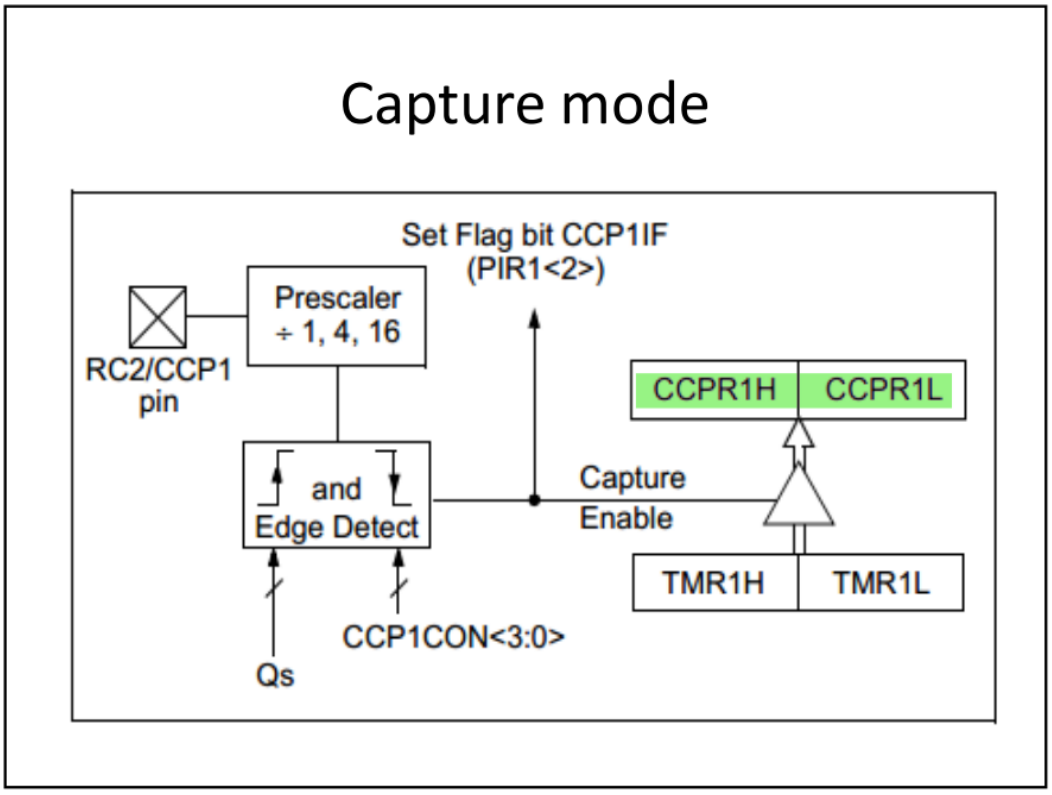
Capture/compare/ PWM

Each Capture/Compare/PWM (CCP) module contains a 16-bit register which can operate as:

- 16-bit Capture register
- 16-bit Compare register
- PWM Master/Slave Duty Cycle register

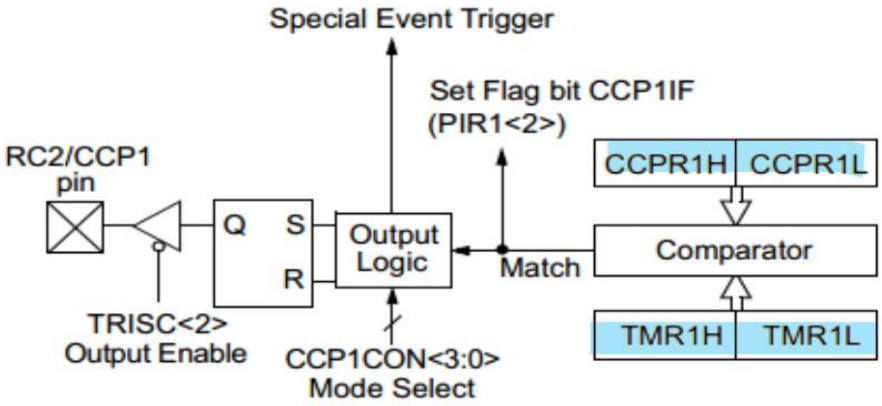
- Both the CCP1 and CCP2 modules are identical in operation

for find the period of the signal that enter on portc CCP1 or CCP2 from rising to rising edge or from falling to falling edge and will save the value in CCPR1H, CCPR1L



Compare mode

Special event trigger will:
reset Timer1, but not set interrupt flag bit TMR1IF (PIR1<0>)
and set bit GO/DONE (ADCON0<2>).



Will compare the value in timer1 if it become as the value in
CCPR1H and CCPR1L
then will set/clear the portc CCP1 or CCP2

CCP1CON REGISTER/CCP2CON REGISTER (ADDRESS 17h/1Dh)

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CCPxX	CCPxY	CCPxM3	CCPxM2	CCPxM1	CCPxM0
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **CCPxX:CCPxY**: PWM Least Significant bits

Capture mode:
Unused.

Compare mode:
Unused.

PWM mode:
These bits are the two LSbs of the PWM duty cycle. The eight MSbs are found in CCPRxL.

bit 3-0 **CCPxM3:CCPxM0**: CCPx Mode Select bits

0000 = Capture/Compare/PWM disabled (resets CCPx module)
0100 = Capture mode, every falling edge
0101 = Capture mode, every rising edge
0110 = Capture mode, every 4th rising edge
0111 = Capture mode, every 16th rising edge
1000 = Compare mode, set output on match (CCPxIF bit is set)
1001 = Compare mode, clear output on match (CCPxIF bit is set)
1010 = Compare mode, generate software interrupt on match (CCPxIF bit is set, CCPx pin is unaffected)
1011 = Compare mode, trigger special event (CCPxIF bit is set, CCPx pin is unaffected); CCP1 resets TMR1; CCP2 resets TMR1 and starts an A/D conversion (if A/D module is enabled)
11xx = PWM mode

PWM value we save it as 10 bit in:
CCPR1L<7:0>:CCP1CON<5:4>

PWM Mode (PWM)

- In Pulse Width Modulation mode, the CCPx pin produces up to a 10-bit resolution PWM output. Since the CCP1 pin is multiplexed with the PORTC data latch, the TRISC<2> bit must be cleared to make the CCP1 pin an output

8-bit register

- PWM Period =
  [(PR2) + 1] • 4 • TOSC • (TMR2 Prescale Value)

- PWM Duty Cycle =
  (CCPR1L:CCP1CON<5:4>) •TOSC • ($TMR2_{Prescale}$)

$$\text{Resolution} = \frac{\log\left(\frac{FOSC}{FPWM}\right)}{\log(2)} \text{ bits}$$

CMCON REGISTER

R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
C2OUT	C1OUT	C2INV	C1INV	CIS	CM2	CM1	CM0
bit 7							bit 0

bit 7 **C2OUT**: Comparator 2 Output bit
When C2INV = 0:
1 = C2 VIN+ > C2 VIN-
0 = C2 VIN+ < C2 VIN-
When C2INV = 1:
1 = C2 VIN+ < C2 VIN-
0 = C2 VIN+ > C2 VIN-

bit 6 **C1OUT**: Comparator 1 Output bit
When C1INV = 0:
1 = C1 VIN+ > C1 VIN-
0 = C1 VIN+ < C1 VIN-
When C1INV = 1:
1 = C1 VIN+ < C1 VIN-
0 = C1 VIN+ > C1 VIN-

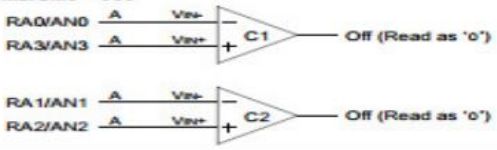
bit 5 **C2INV**: Comparator 2 Output Inversion bit
1 = C2 output inverted
0 = C2 output not inverted

bit 4 **C1INV**: Comparator 1 Output Inversion bit
1 = C1 output inverted
0 = C1 output not inverted

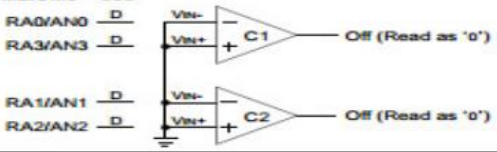
bit 3 **CIS**: Comparator Input Switch bit
When CM2:CM0 = 110:
1 = C1 VIN- connects to RA3/AN3
C2 VIN- connects to RA2/AN2
0 = C1 VIN- connects to RA0/AN0
C2 VIN- connects to RA1/AN1

bit 2 **CM2:CM0**: Comparator Mode bits
Figure 12-1 shows the Comparator modes and CM2:CM0 bit settings.

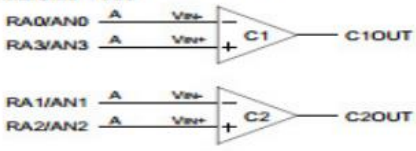
Comparators Reset
CM2:CM0 = 000



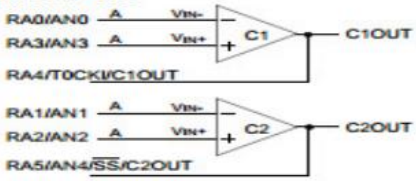
Comparators Off (POR Default Value)
CM2:CM0 = 111



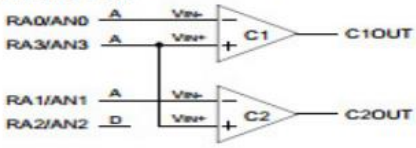
Two Independent Comparators
CM2:CM0 = 010



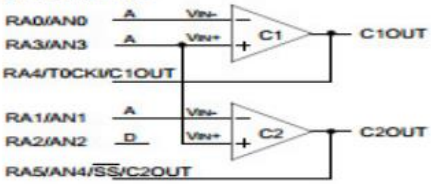
Two Independent Comparators with Outputs
CM2:CM0 = 011



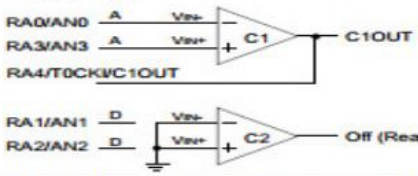
Two Common Reference Comparators
CM2:CM0 = 100



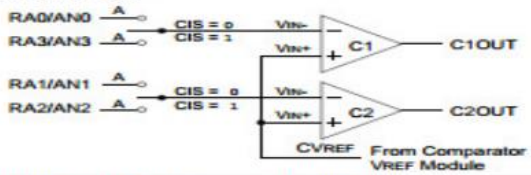
Two Common Reference Comparators with Outputs
CM2:CM0 = 101



One Independent Comparator with Output
CM2:CM0 = 001



Four Inputs Multiplexed to Two Comparators
CM2:CM0 = 110



A = Analog Input, port reads zeros always. D = Digital Input. CIS (CMCON<3>) is the Comparator Input Switch.

Instruction	Instruction Code										Description	Execution time (fosc=270Khz)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0		
Clear Display	0	0	0	0	0	0	0	0	0	1	Write “00H” to DDRAM and set DDRAM address to “00H” from AC	1.53ms
Return Home	0	0	0	0	0	0	0	0	1	—	Set DDRAM address to “00H” from AC and return cursor to its original position if shifted. The contents of DDRAM are not changed.	1.53ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	SH	Assign cursor moving direction and enable the shift of entire display.	39 μ s
Display ON/OFF Control	0	0	0	0	0	0	1	D	C	B	Set display (D), cursor (C), and blinking of cursor (B) on/off control bit.	39 μ s
Cursor or Display Shift	0	0	0	0	0	1	S/C	R/L	—	—	Set cursor moving and display shift control bit, and the direction, without changing of DDRAM data.	39 μ s
Function Set	0	0	0	0	1	DL	N	F	—	—	Set interface data length (DL:8-bit/4-bit), numbers of display line (N:2-line/1-line)and, display font type (F:5×11 dots/5×8 dots)	39 μ s
Set CGRAM Address	0	0	0	1	AC5	AC4	AC3	AC2	AC1	AC0	Set CGRAM address in address counter.	39 μ s
Set DDRAM Address	0	0	1	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Set DDRAM address in address counter.	39 μ s
Read Busy Flag and Address	0	1	BF	AC6	AC5	AC4	AC3	AC2	AC1	AC0	Whether during internal operation or not can be known by reading BF. The contents of address counter can also be read.	0 μ s
Write Data to RAM	1	0	D7	D6	D5	D4	D3	D2	D1	D0	Write data into internal RAM (DDRAM/CGRAM).	43 μ s
Read Data from RAM	1	1	D7	D6	D5	D4	D3	D2	D1	D0	Read data from internal RAM (DDRAM/CGRAM).	43 μ s

lcd016n002bcfhet.pdf (vishay.com)

```
        BSF     STATUS,C    ; set carry for subtract
        MOVLW   D'100'      ; load 100
sub1    SUBWF   ADbin,1     ; and subtract from result
        INCF    huns        ; count number of loops
        BTFSC   STATUS,C    ; and check if done
        GOTO    sub1        ; no, carry on

        ADDWF   ADbin,1     ; yes, add 100 back on
        DECF    huns        ; and correct loop count

; Calculate tens digit.....................................

        BSF     STATUS,C    ; repeat process for tens
        MOVLW   D'10'       ; load 10
sub2    SUBWF   ADbin,1     ; and subtract from result
        INCF    tens        ; count number of loops
        BTFSC   STATUS,C    ; and check if done
        GOTO    sub2        ; no, carry on

        ADDWF   ADbin       ; yes, add 100 back on
        DECF    tens        ; and correct loop count
        MOVF    ADbin,W     ; load remainder
        MOVWF   ones        ; and store as ones digit

        RETURN              ; done
```