luciano trapa

I
MICROCONTROLLORI
(µC o MCU o microcomputer single chip)
ED IL
PIC
16 *F* 877 A

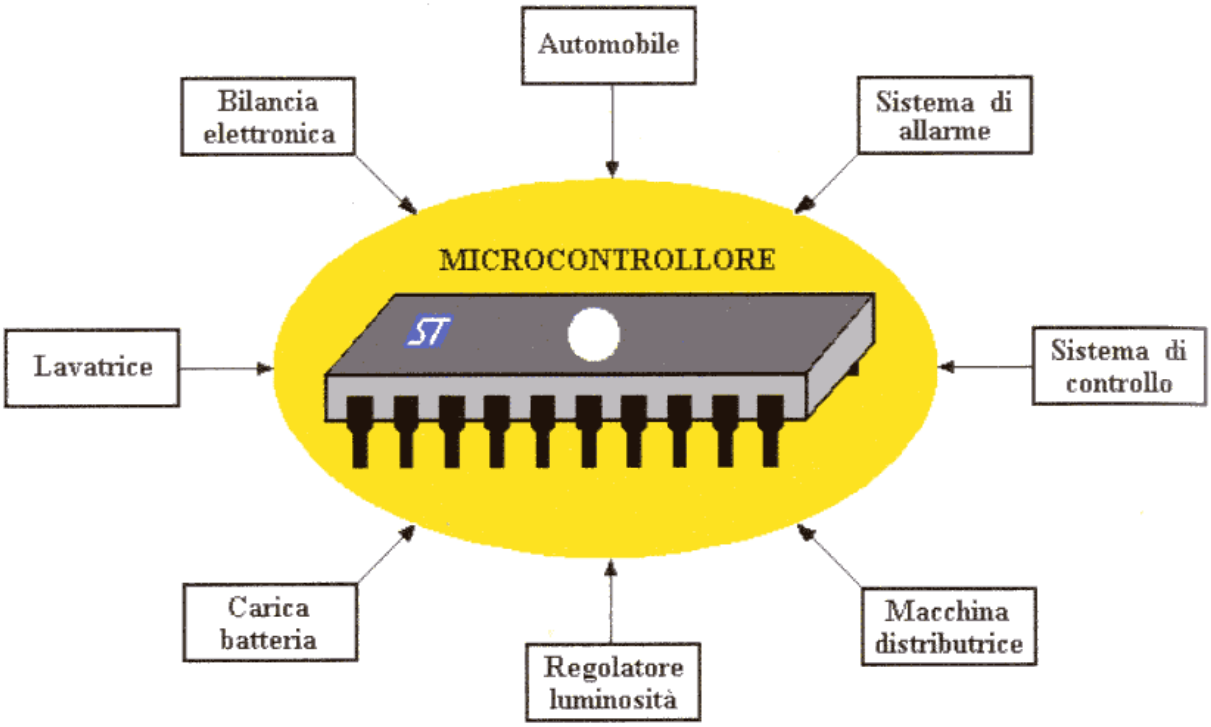
MICROCONTROLLORE

o

MCU (Micro Controller Unit)

o

MICROCOMPUTER SINGLE CHIP



PREMESSA

Dopo un prima parte di carattere generale, il capitolo, per conferire concretezza alla trattazione, farà riferimento a un microprocessore reale:
il PIC 16 F 877 A IP.

A proposito dei PIC, cominciamo col fare alcune puntualizzazioni:

- **L'acronimo "PIC":**
  L'acronimo "PIC" sta per **"Pheripheral Interface Controller"**

- **La lettera "F"**
  La lettera "F" nella sigla di un PIC sta per "Flash" e indica che il programma risiede in una ROM di tipo flash, che è una tipologia di memoria EEPROM.
  Una memoria di tipo EEPROM , oltre a essere elettricamente programmabile è anche **elettricamente cancellabile**, senza bisogno dell'esposizione ai raggi UV

- **CICLO ISTRUZIONE**
  E' l'intervallo di tempo nel quale viene eseguita una normale istruzione ed è equivalente a quattro periodi dell'oscillatore.
  L'esecuzione dell'istruzione può richiedere in casi particolari non uno, ma due cicli istruzione, come (per esempio) quando un test condizionale risulta vero o quando il valore del contatore di programma viene modificato dal risultato di un'operazione .

GENERALITA' SUL MICROCONTROLLORE

Il microcontrollore (che indicheremo a volte come "µC" oppure "mC" oppure, ancora, come "MCU") è un dispositivo **elettronico, digitale, integrato, programmabile,** in grado di eseguire autonomamente diverse funzioni in base alle **istruzioni** di un **programma** presente in una memoria non volatile (tipicamente EEPROM) contenuta **all'interno dell'integrato stesso**.

Essenzialmente il µC è un dispositivo che gestisce un certo numero di linee di ingresso e un certo numero di linee di uscita.

Nel corso dell'esecuzione del programma in esso memorizzato,il µC:

- **legge** lo stato delle linee di ingresso
- **elabora**, in base a quanto richiede il **programma**, i valori acquisiti
- **porta a livello alto o basso le linee di uscita** secondo i risultati dell'elaborazione imposta dal programma.

STRUTTURA INTERNA SEMPLIFICATA

In prima approssimazione, la struttura interna di un µC si può semplificare nei seguenti blocchi:

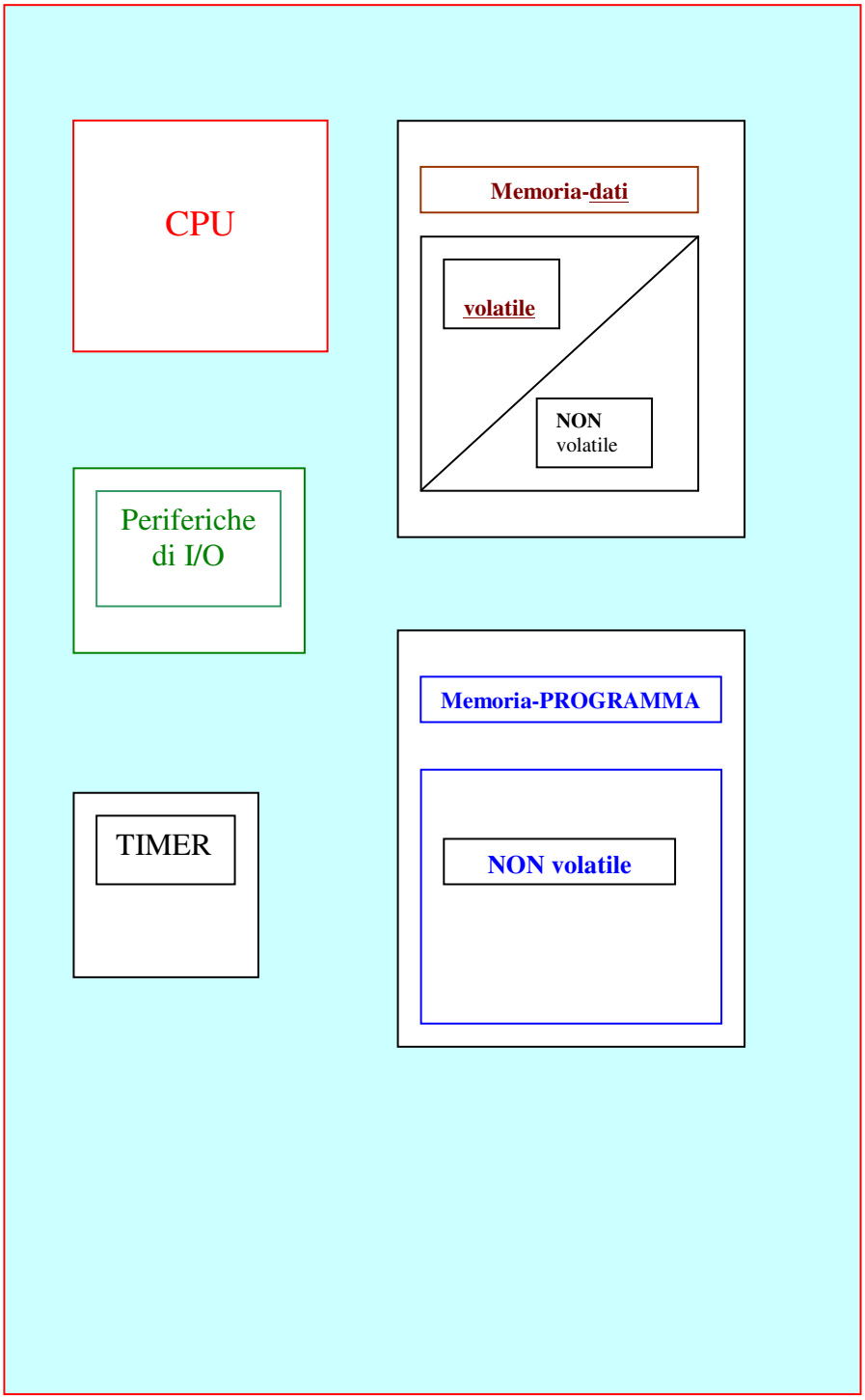
- **unità centrale** di elaborazione (CPU, Central Processing Unit);
  la CPU contiene:
  - l'unità logico-aritmetica (ALU)
  - circuiti di controllo
  - circuiti di indirizzamento
  - registro accumulatore (W)
  - registro di stato
  - registro contatore di programma (PC)

- memoria **NON** volatile per il **PROGRAMMA**

- memoria ***volatile*** (oppure parzialmente volatile e parzialmente non volatile) per i ***dati***

- **periferiche di Input/Output (I/O)** e in particolare:
  - PORTE, ossia gruppi di linee, di ingresso e PORTE di uscita
  - Convertitore analogico-digitale o convertitore A/D (ADC) in ingresso
  - Comparatori in ingresso
  - Moduli "CCP" per il Confronto dei dati, la Cattura dei dati e la generazione di segnali PWM
  - Dispositivi di I/O SERIALE per scambiare dati con dispositivi esterni usando un numero limitato di conduttori
  - TIMER (uno o più timer sincronizzano le azioni del µC).

µC: STRUTTURA INTERNA SEMPLIFICATA



PRECISAZIONI SULLA STRUTTURA INTERNA DEL MICROCONTROLLORE

PERIFERICHE DI I/O

Sono formate da PORTE DI I/O ossia da gruppi di linee, con relativi registri, che permettono:

- *l'emissione di dati digitali*
- *l'acquisizione di bit*
- *l'acquisizione di segnali analogici, grazie a un convertitore A/D integrato*

CPU
O
UNITA' CENTRALE DI ELABORAZIONE

Comprende:

- *l'Unità Aritmetico-Logica (ALU)*
- *un certo numero di REGISTRI, cioè memorie "temporanee" o "di transito" di tipo RAM di piccola capacità (decine di bit)*
- *i circuiti di CONTROLLO*

UTILIZZAZIONE E NUMERO DI BIT DEI µC

I microcontrollori sono usati come controllori in:

- sistemi di controllo di TEMPERATURA
- sistemi di controllo di VELOCITA' di MOTORI
- dispositivi e sistemi automobilistici (AUTRONICA)
- sistemi DOMOTICI
- LAVATRICI
- macchine DISTRIBUTRICI

NUMERO di BIT	UTILIZZAZIONE
4	**Sistemi di COMANDO di:** **- chiavi elettroniche** **- carte di credito** **- apparecchi elettromedicali** **- tastiere**
8	**c.s.**
16	**Sistemi di *CONTROLLO* di:** **- stampanti** **- sistemi di telecomunicazione a commutazione di pacchetto** **- sistemi ISDN**
32	**Formano l'UNITA' CENTRALE di sistemi di controllo complessi**

CARATTERISTICHE PRINCIPALI DI ALCUNI TIPI DI PIC

Key Features	PIC 16F873A	PIC 16F874A	PIC 16F876A	PIC 16F877A
Operating Frequency	DC - 20 MHz	DC - 20 MHz	DC - 20 MHz	DC - 20 MHz
Resets (and Delays)	*POR, BOR (PWRT, OST)*	*POR, BOR (PWRT, OST)*	*POR, BOR (PWRT, OST)*	*POR, BOR (PWRT, OST)*
Flash Program Memory (14-bit words)	4 K	4 K	8 K	8 K
Data memory (bytes)	192	192	368	368
EEPROM Data Memory (bytes)	128	128	256	256
Interrupts	14	15	14	15
I/O Ports	Ports A, B, C	Ports A, B, C, D, E	Ports A, B, C	Ports A, B, C, D, E
Timers	3	3	3	3
Capture/Compare/PWM modules	2	2	2	2
Serial Communications	MSSP, USART	MSSP, USART	MSSP, USART	MSSP, USART
Parallel Communications	–	PSP	–	PSP
10-bit Analog-to-Digital Module	5 input channels	8 input channels	5 input channels	8 input channels
Analog Comparators	2	2	2	2
Instruction Set	35 Instructions	35 Instructions	35 Instructions	35 Instructions

DISPOSITIVI INTERNI AL MICROCONTROLLORE (MCU)

PIEDINATURA PIC 16F877A IP

GIALLO → PORTA A

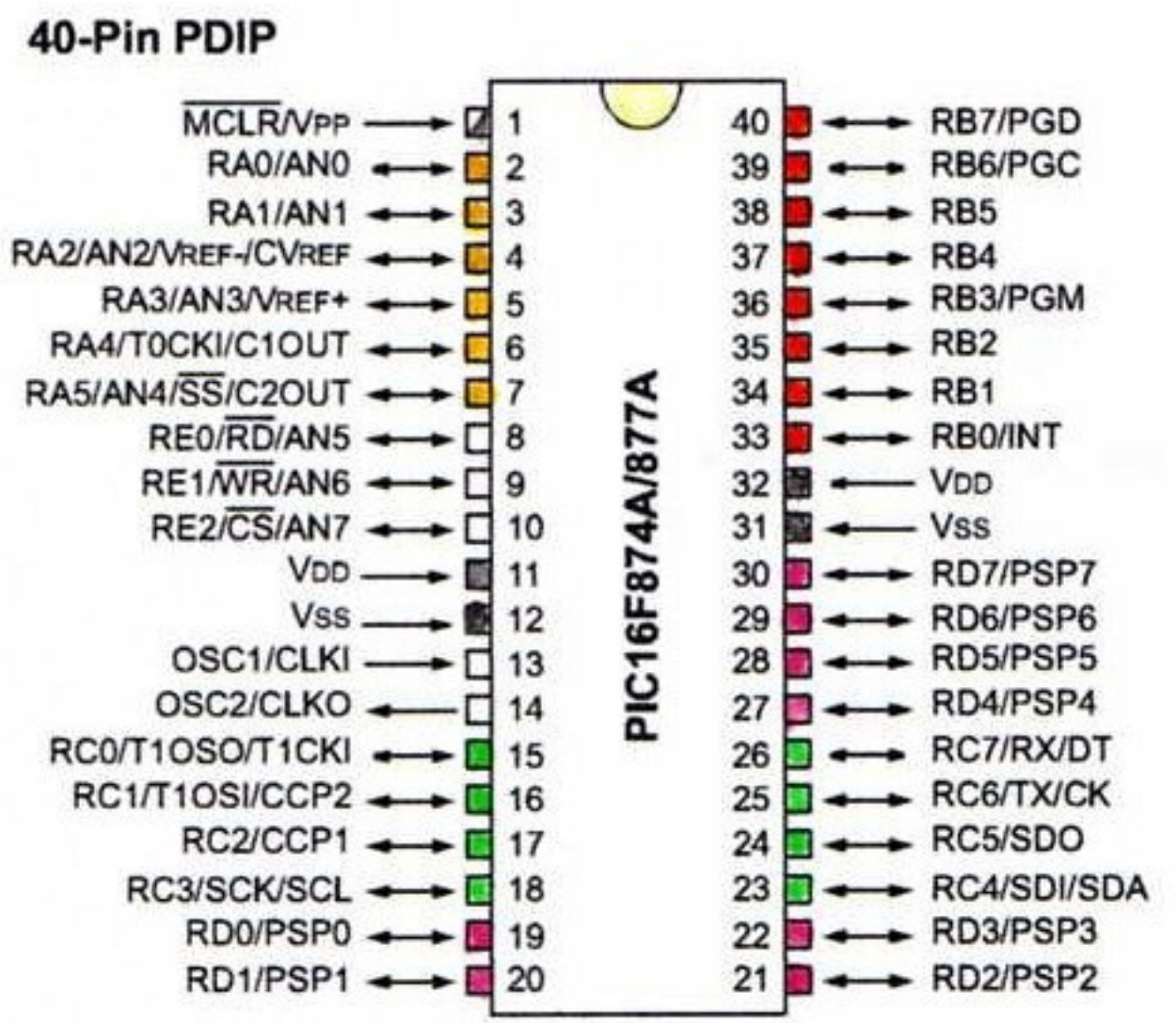
ROSSO → PORTA B

VERDE → PORTA C

VIOLA → PORTA D

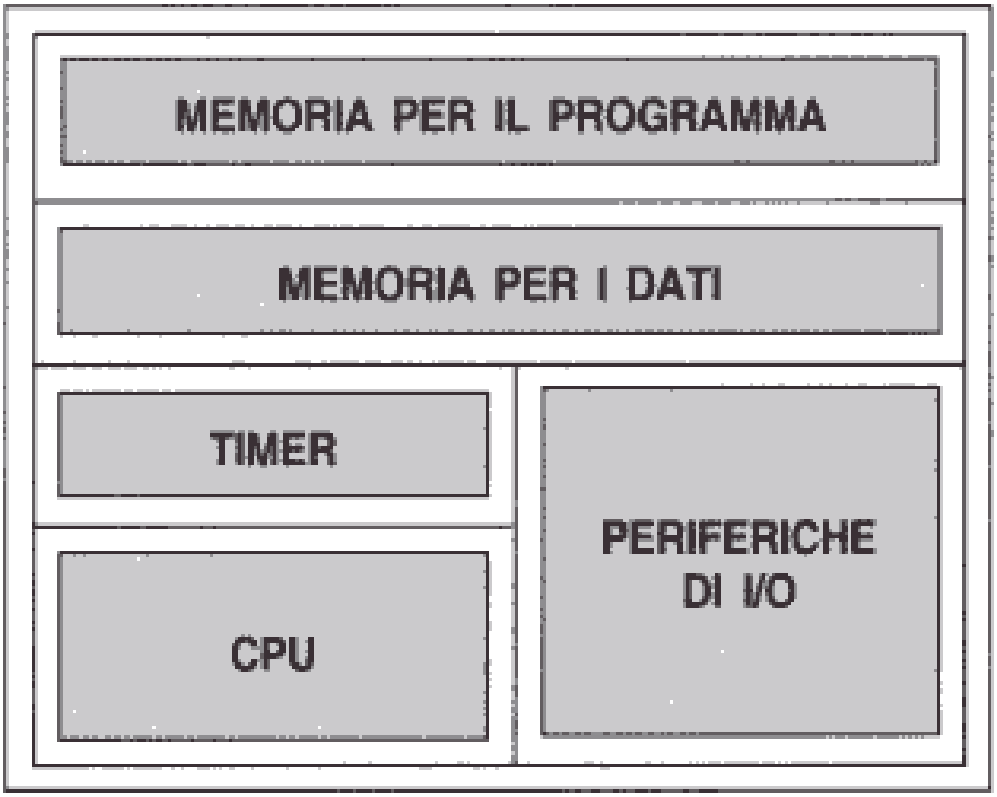
BIANCO → PORTA E, CLOCK/OSCILLATORI

NERO → ALIMENTAZIONI
(VDD per circuiti logici,
VPP per programmazione
VSS = GND = MASSA = pin 12)



STRUTTURA INTERNA COMPLESSIVA

(architettura di Harvard: programma e dati in memorie separate)



MEMORIA PER IL PROGRAMMA (EEPROM)
MEMORIA PER I DATI (EEPROM e RAM)

STRUTTURA INTERNA DI UN µC (in generale)

L'architettura di un microcontrollore può comprende i "blocchi" elencati nella tabella seguente:

BLOCCO	COMPONENTI DEL BLOCCO	EVENTUALE DESCRIZIONE
CORE o CPU	- **ALU** - **REGISTRO di FLAG o registro di STATO** - **Circuiti di CONTROLLO e TEMPORIZZAZIONE**	
MEMORIA-DATI		Può essere di tipo: - **RAM** e/o - **EEPROM**
MEMORIA-PROGRAMMA		**Può essere di tipo di tipo:** - **ROM** - **EPROM** (nel qual caso il µC può essere finestrato per permettere la cancellazione della memoria mediante raggi ultravioletti oppure **OTP** (***One Time Programmable*** non finestrati e ***programmabili una sola volta***) - **EEPROM**
TIMER (uno o più)		
WATCHDOG (WTD)		
INGRESSI DIGITALI		
PORTE di I/O		
CIRCUITI DI INTERFACCIA		
CONVERTITORE A/D		
COMPARATORI (CMP)		
MODULI CCP (Compare / capture/ PWM)		**Confronto di dati o di eventi/ Cattura di dati o di eventi/ generazione di un segnale PWM**

LE "PORTE"

La porta è una "periferica" di I/O.
Per **porta** si intende un **gruppo di linee** di INGRESSO/USCITA e quindi un **gruppo di piedini (pin)** dell'integrato MCU.

Mediante le linee di una porta il microcontrollore può inviare verso l'esterno **dati digitali** e **acquisire dati digitali e analogici**.

Alcune linee di alcune porte possono scambiare anche segnali di temporizzazione (clock), di controllo e ricevere la tensione continua di alimentazione dell'integrato.

A ogni porta, cioè a ogni gruppo di linee di I/O, sono sempre **associati** un **registro di nome *PORT*** e un **registro di nome TRIS**.
Si tratta di una COPPIA di **FILE REGISTER**, cioè una coppia di registri della **memoria-dati RAM**.
Così alla porta A sono associati i file register PORT**A** e TRIS**A,**
alla porta B sono associati i file register PORT**B** e TRIS**B** ecc.

Il registro ***PORT*** è il registro di "***utilizzazione***" della porta, nel senso che **tutti i bit** che il µC vuole inviare verso l'esterno attraverso una porta ***devono essere scritti nelle celle binarie del registro PORT.***
Analogamente ogni bit che il mC acquisisce dal mondo esterno viene sempre temporaneamente memorizzato nelle celle binarie del registro PORT.
Possiamo quindi anche dire che ***PORT è il registro di "transito***" della porta.

Il registro **TRIS** (TRIgger di Schmitt) è invece il **registro** di "**pilotaggio**" o di "**gestione** " o di "**configurazione**" della porta, nel senso che **il valore (1/O) che facciamo assumere a un bit del registro TRIS configura la corrispondente linea della porta (associata a TRIS) come uscita o come ingresso**.

A questo proposito notiamo esplicitamente che ad ogni bit (per esempio b_2) del registro TRIS è associato il corrispondente bit (b_2) del registro PORT e quindi è associata la corrispondente linea (con relativo pin) della porta collegata a PORT.

L'assegnazione del valore **UNO** a un bit di **TRISA** determina il fatto che il bit presente sulla corrispondente linea della porta A, e che è in transito sulla corrispondente cella binaria di PORTA, sarà un bit di **INGRESSO** per il µC, cioè che questo bit "**entrerà"** nel µC.

L'assegnazione del valore **zero** a un bit di **TRISA** determina il fatto che il bit presente sulla corrispondente linea della porta A, e che è in transito sulla corrispondente cella binaria di PORTA, sarà un bit di **USCITA** per il µC, cioè che questo bit "**uscirà**" dal µC.

0 → Output = uscita
1 → Intput = ingresso

UNA PORTA,
IL SUO REGISTRO DI TRANSITO
E IL SUO REGISTRO DI CONFIGURAZIONE

PORTA:

GRUPPO DI LINEE DI I/O

PORT:

REGISTRO DI UTILIZZAZIONE O DI TRANSITO DELLA PORTA (tutti i bit che entrano o escono dalla porta devono transitare attraverso il registro PORT)

TRIS[1]:

REGISTRO DI PILOTAGGIO O CONFIGURAZIONE O GESTIONE DELLA PORTA (configura le linee della porta come INGRESSI o come USCITE)

← →	--------	bit	--------	*
← →	--------	bit	--------	*
← →	--------	bit	--------	*
← →	--------	bit	--------	*
← →	--------	bit	--------	*
← →	--------	bit	--------	*
← →	--------	bit	--------	*
← →	--------	bit	--------	*

[1] "TRIS" sta per **TRI**gger di **S**chmitt.

LA PORTA A

La porta A è formata da **6 linee** bidirezionali,
corrispondenti ai **pin RA0, RA1, RA2, RA3, RA4, RA5** dell'integrato µC.

Le linee RA6 e RA7 **NON** sono attive.

Tutti i pin possono essere configurati:

- come **linee digitali di ingresso/uscita**
- come **linee di (solo) ingresso analogico (eccetto RA4)**

Il pin RA2 può anche essere utilizzato:

- come tensione di riferimento negativa per l'ADC o per il comparatore
- come tensione di riferimento per il comparatore

Il pin RA3 può anche essere utilizzato:
come tensione di riferimento positiva per l'ADC

Il pin RA4 può essere anche utilizzato:

- come ingresso di clock esterno per il modulo TIMER 0
- come uscita del comparatore;
  da un punto di vista elettronico RA4, configurato come linea digitale, presenta:
- *un **ingresso** a **trigger di Schmitt***
- *un'**uscita open-drain***

Il pin RA5 può anche essere utilizzato

- come tensione di uscita per il comparatore
- come ingresso analogico
- come ingresso (slave) di selezione per la porta seriale sincrona.

Riportiamo de denominazioni complete dei pin:

		RA 5	RA 4	RA 3	RA 2	RA 1	RA 0
X	**X**	**RA5/AN4/SS/C2OUT**	**RA4/TOCKI/C1OUT**	**RA3/ AN3/ V^{+}_{REF}**	**RA2 /AN2/ V^{-}_{REF}/CV REF**	**RA1/AN1**	**RA0/AN0**

LA PORTA B

La particolarità di questa porta rispetto alle altre è che **su tutte le linee**, (contrariamente a quelle della porta A quando sono configurate come **ingressi**, è possibile **abilitare** il collegamento con **resistori di pull-up interni all'integrato microcontrollore**.

La porta B è formata da **8 linee** bidirezionali,
corrispondenti ai **pin RB0, RB1, RB2, RB3, RB4, RB5, RB6, RB7** dell'integrato µC.

I file register PORT B e TRIS B sono simili, ma non uguali ai registri, di tipo analogo,
PORT A e TRIS A

Le differenze fondamentali fra le due coppie di registri sono essenzialmente:

- PORT B e TRIS B implementano **8 bit** (mentre PORT A e TRIS A implementano 6 bit)

- i bit di PORT B (contrariamente a quelli di PORT A) offrono la possibilità di realizzare un **pull-up**

- i segnali multiplexati nei pin di PORT B (cioè le "seconde funzioni" eventualmente effettuabili dai singoli terminali) sono diversi dai segnali multiplexati nei pin di PORT A.

Notiamo che le differenze sono sostanzialmente fra i registri "PORT", mentre i registri "TRIS" sono, a parte il numero dei bit, sostanzialmente identici.

Tutte le linee sono attive.
Tutti i pin possono essere configurati:

- come **linee digitali di ingresso/uscita**
  (ma **NON** come **linee di ingresso analogico, contrariamente alla porta A)**

E, in particolare:

Il pin RB0 può anche essere utilizzato:

- come linea di ingresso per interruzioni esterne (perciò è anche chiamato RB0/INT)
- per la programmazione del pull-up

Il pin RB1 può anche essere utilizzato

- per la programmazione del pull-up

Il pin RB3 può anche essere utilizzato

- per la programmazione del pull-up
- come pin di programmazione in modalità LVP ossia Low Voltage Progamming (perciò è anche chiamato RB3/PGM)

Il pin RB4 può anche essere anche utilizzato

- per la programmazione del pull-up
- per le interruzioni

Il pin RB5 può anche essere anche utilizzato

- per la programmazione del pull-up
- per le interruzioni

Il pin RB6 può anche essere anche utilizzato

- per la programmazione del pull-up
- per le interruzioni
- come pin per il debugging "in circuit"
- per la programmazione del clock seriale o PCG (perciò è anche chiamato RB6/PGC)

Il pin RB7 può anche essere anche utilizzato

- per la programmazione del pull-up
- per le interruzioni
- come pin per il debugging "in circuit"
- per la programmazione del clock seriale (PCG)
- per la programmazione dei dati seriali (PGD) (perciò è anche chiamato RB3/PGD)

LA PORTA C

La particolarità di questa porta rispetto alle altre è che **tutte le linee**, (contrariamente a quelle delle porte A e B), quando sono configurate come input, presentano
ingressi a trigger di Schmitt.

La porta C è formata da **8 linee** bidirezionali,
corrispondenti ai **pin RC0, RC1, RC2, RC3, RC4, R5, RC6, RC7** dell'integrato µC.

I file register PORT C e TRIS C sono simili, ma non uguali ai registri, di tipo analogo, relativi alle porte A e B

Tutte le linee sono attive.
Tutti i pin possono essere configurati:

- come **linee digitali di ingresso/uscita**
  (ma **NON** come **linee di ingresso analogico, contrariamente alla porta A)**

E, in particolare:

Il pin RC0 può anche essere utilizzato:
- come uscita dell'oscillatore (interno) per il modulo Timer 1
- come ingresso del clock (esterno) per il modulo Timer 1
  (perciò è anche chiamato RC0/T1oso/T1CKI)

Il pin RC1 può anche essere utilizzato:
- come ingresso dell'oscillatore (esterno) per il Timer 1
- come ingresso del modulo CAPTURE 2
- come uscita del modulo COMPARE2/PWM2
  (perciò è anche chiamato RC1/T1osI/CCP2)

Il pin RC2 può anche essere utilizzato
- come ingresso del modulo CAPTURE 1
- come uscita del modulo CAPTURE1
- come uscita del modulo PWM1
  (perciò è anche chiamato RC2/CCP2)

Il pin RC3 può anche essere utilizzato
- per il clock seriale dei moduli SPI e I^2C
  (perciò è anche chiamato RC3/SCK/SCL)

Il pin RC4 può anche essere anche utilizzato
- per i dati SPI, nella modalità SPI
  per i dati, nella modalità I2C
  (perciò è anche chiamato RC0/SCD/SCA)

Il pin RC5 può anche essere anche utilizzato
- come uscita dei dati della della porta seriale sincrona
  (perciò è anche chiamato RC0/TSDO)

Il pin RC6 può anche essere anche utilizzato
- per la trasmissione asincrona relativa all'USART
- per il clock sincrono
  (perciò è anche chiamato RC6/TX/CK)

Il pin RC7 può anche essere anche utilizzato
- per la ricezione asincrona relativa all'USART
- per i dati sincrono
  (perciò è anche chiamato RC7/RX/CD)

LA PORTA D

Come per la porta C, la particolarità di questa porta rispetto alle altre è che **tutte le linee**, (contrariamente a quelle delle porte A e B), quando sono configurate come input, presentano ***ingressi a trigger di Schmitt***.

La porta D è formata da **8 linee** bidirezionali,
corrispondenti ai **pin RD0, RD1, RD2, RD3, RD4, RD5, RD6, RD7** dell'integrato µC.

I file register PORT C e TRIS C sono simili, ma non uguali ai registri, di tipo analogo, relativi alle porte A, B e C

Tutte le linee sono attive.

Tutti i pin possono essere configurati:

- come **linee digitali di ingresso/uscita**

(ma **NON** come **linee di ingresso analogico, contrariamente alla porta A)**

E, inoltre tutti pin possono essere utilizzati anche come linee (bit) della **porta parallela slave**

LA PORTA E

La porta E ha soltanto **tre linee** utilizzabili, che sono associate ai **tre bit di peso più basso (bit 0, bit 1, bit 2)** dei corrispondenti registri PORT E e TRIS E.

Come per la porta C e la porta D, queste linee della porta E, selezionate come input, presentano ingressi a **TRIGGER DI SCHMITT**

Tutti i pin possono essere configurati:

- come **linee digitali di ingresso/uscita**
- come **linee di (solo) ingresso analogico (come la porta A)**

E, in particolare:

Il pin RE0 può anche essere utilizzato:

- come ingresso di controllo della lettura nella modalità Parallel Slave Port (perciò è anche chiamato RE0/RD/AN5)

Il pin RE1 può anche essere utilizzato:

- come ingresso di controllo della scrittura nella modalità Parallel Slave Port (perciò è anche chiamato RE1/WR/AN6)

Il pin RE2 può anche essere utilizzato:

- come ingresso di controllo della selezione del chip nella modalità Parallel Slave Port (perciò è anche chiamato RE2/CS/AN7)

LE MEMORIE

MEMORIA DI PROGRAMMA

La memoria di programma del PIC 16F877A è formata 8000 locazioni di 14 bit ciascuna ed è suddivisa in pagine di 2048 celle o locazioni.

MEMORIA-DATI E BANCHI di MEMORIA

Il banco è una "porzione" di memoria-dati.
La memoria-dati EEPROM del microcontrollore è divisa in un certo numero di banchi (quattro per il PIC 16F877A).
Ogni banco contiene registri di 8 bit.

I differenti file register della memoria-dati possono trovarsi alcuni in un banco, altri in un altro banco.
Se un dato deve essere posto in un registro che si trova in un determinato banco, è il microcontrollore che provvede a inviare il dato nel giusto banco (quello che contiene il registro di destinazione del dato) e al giusto indirizzo (l'indirizzo del registro).

Il banco di memoria può essere selezionato impostando, con le istruzioni **BCF** e **BSF**, il valore dei bit RP1 (bit 6) ed RPO (bit 5) del file register **STATUS.**

RPI	RPO	Banco
1	1	3
1	0	2
0	1	1
0	0	0

I REGISTRI: PREMESSE

In generale il **registro** è un **raggruppamento** (una "**catena**")
di **celle di memoria a un bit ciascuna**.
E' quindi un dispositivo in grado di **memorizzare temporaneamente** un **certo numero di bit**.

Da un punto di vista circuitale, i registri possono essere realizzati con **flip-flop (FF)** o latch, cioè con memorie digitali elementari a un bit.

Normalmente i registri (essendo le celle di memoria collegate tra loro) consentono lo **spostamento di un bit da una cella a quella adiacente** e quindi lo **scorrimento** di tutta la stringa di bit memorizzata: si parla allora di registri a scorrimento o **shift-register**.

Il **trasferimento dei dati**, provenienti dall'esterno, **verso l'interno** del registro viene detto **SCRITTURA** o **CARICAMENTO**.

Il trasferimento verso l'esterno dei dati memorizzati dentro il registro viene detto LETTURA o PRELIEVO o PRELEVAMENTO.

L'ingresso o scrittura dei dati può avvenire:

- in serie (un singolo bit dopo l'altro attraverso un'unica linea di ingresso)
  oppure
- in parallelo: tutti i bit contemporaneamente su più linee (ogni bit su una propria linea)
  scorrimento

Lo stesso vale per l'uscita dei dati (lettura)

In base alla combinazione di modalità di ingresso/uscita implementate i registri a scorrimento sono classificati in:

- SISO: Serial In – Serial Out
- SIPO: Serial In – Parallel Out
- PISO: Parallel In – Serial Out
- SISO: Parallel In – Parallel Out

ALCUNI REGISTRI A SCORRIMENTO COMMERCIALI

NUMERO di BIT	**FUNZIONAMENTO**	**SIGLA**	**SOTTOFAMIGLIE LOGICHE DISPONIBILI**
4	**universale**	. . .194	LS, HCT, FAST
8	***SIPO***	*. . .164*	*LS, HCT, HC, FAST*
8	**PISO /PIPO**	. . .165	LS, HCT, HC
8	***SISO***	*. . .166*	*LS, HCT*
16	**SIPO**	. . .673	LS
16		*. . .674*	*LS*

I REGISTRI

Con il termine "**file register**" si fa riferimento sia all'**area di memoria**-dati contenente registri sia a **singoli registri** (file register collocati nell'area di memoria suddetta).
I file register possono anche costituire l'operando di una istruzione[7]

I **fili register** si suddividono in due tipologie:

- file register di tipo "**registri speciali**" (special function register)- file register di tipo "registri speciali" (special function register)
  come **TRIS, PORT....., PCL e PCLATH** (questi ultimi due file register **"realizzano" il contatore di programma PC –** [il quale, nel PIC 16F877A non è fisicamente un file register]

- file register di tipo "registri generali" (general purpose register)
  come il registro accumulatore W, chiamato anche "registro di lavoro" o "working register"

Sono quindi "file register" sia i registri speciali, sia i registri generali.

INDIRIZZAMENTO DIRETTO DEI REGISTRI, ISTRUZIONI DEL SET, TRASFERIMENTO DEL CONTENUTO DELL'ACCUMULATORE IN UN FILE REGISTER

I **registri** del PIC 16F877A sono **512**, quindi per poter individuare ciascuno di essi mediante un indirizzo binario, l'**indirizzo** deve essere formato da almeno **9 bit**
(infatti $2^9 = 512$, mentre $2^8 = 256$, per cui 8 bit risulterebbero insufficienti)

TRASFERIMENTO DEL CONTENUTO DELL'ACCUMULATORE W NEL FILE REGISTER di INDIRIZZO 15Bh (esempio)

Indirizzo del registro destinazione: **15Bh** = 11+80+256 = 347_{10} = 101011011
Registro sorgente: accumulatore W

Il trasferimento al registro 15Bh si realizza mediante tutte e tre le istruzioni seguenti
(le prime due agiscono sui bit numero 8 e numero 9 dell'indirizzo 15Bh, mentre la terza agisce sui sette bit meno significativi di 15Bh):

BCF STATUS, 5 (Bit Clear File: azzera il bit numero 5, cioè RP0, del file register STATUS)
BSF STATUS, 6 (Bit Set File: imposta ad UNO il bit numero 6, cioè RP1, del file register STATUS)
MOVWF 5Bh (Move W to f: sposta il contenuto di W all'indirizzo f=5Bh, i cui sette bit meno significativi sono individuati dalla costante f)

-

L'istruzione
MOVWF 5Bh
è del tipo
MOVWF f
con
"f"= **costante a 7 bit**, i quali rappresentano i 7 bit **meno significativi** dell'indirizzo a 9 bit del **registro destinazione**;
nell'esempio in esame è: f = 5Bh = 101 1011,
mentre l'indirizzo-destinazione è **1**5Bh = 1 0101 1011
(in 5Bh la cifra 5 può essere espressa con 3 soli bit, essendo la più significativa, mentre ha bisogno di 4 bit in 15Bh)

Notiamo che l'operando di MOVWF non è 15Bh (esprimibile con 9 bit),
ma 5Bh, esprimibile con 7 bit.

L'istruzione MOWVF 5Bh trasferisce nel file register di indirizzo 15Bh
il contenuto del registro accumulatore W

Indirizzo del registro sorgente: 15Bh = 11 + 80 + 256 = 347_{10}

Bit 9	***Bit 8***	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1
Peso 256	***128***	64	32	16	8	4	2	1
1	***0***	**1**	**0**	**1**	**1**	**0**	**1**	**1**
RP 1 STATUS	***RP 0 STATUS***	Costante f	Costante f	Costante f	Costante f	Costante f	Costante f	Costante f

INDIRIZZAMENTO INDIRETTO DEI REGISTRI

In questa modalità di indirizzamento, l'indirizzo della cella di memoria alla quale si vuole accedere è contenuto in un registro che ha la funzione di puntatore.
Nel PIC 16F877A il registro puntatore è il file register FSR (File Select Register)
L'FSR è un registro a otto bit che consente di indirizzare 256 registri contigui.

CLOCK – OSCILLATORE

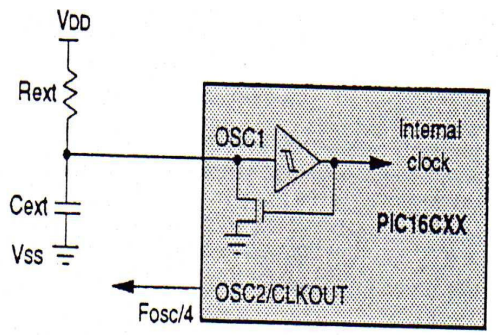
PREMESSA 1

Riguardo al PIC 16F877A ci interesseranno particolarmente questi pin:

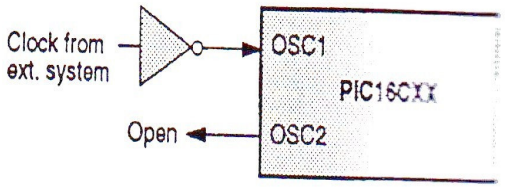
- pin 13 (OSC1/CLOCKI dove "I" sta per Input, cioè ingresso del clock esterno)
- pin 14 (OSC2/CLOCKO dove "O" sta per Output, cioè uscita del clock esterno)

PREMESSA 2

Non bisogna confondere il collegamento al PIC di **componenti passivi esterni** -quarzi, risonatori ceramici, condensatori o resistenze- (che vanno a completare **l'oscillatore interno**,[2] che devono essere collegati fra i pin 13 e 14 del PIC16F877A e che permettono le quattro modalità di funzionamento LP, XT, HS, RC) con il vero e proprio oscillatore esterno il quale può essere collegato al pin 13 lasciando aperto il pin 14 e che deve essere un oscillatore "completo", dotato quindi anche del componente attivo (p. es. porta logica).



Collegamento del PIC a un oscillatore INTERNO



Collegamento del PIC a un oscillatore ESTERNO

[2] basato, per esempio, su porta NOT triggerata

OSCILLATORE INTERNO E OSCILLATORE ESTERNO

La distinzione fra oscillatore esterno e interno è basata sul componente ATTIVO dell'oscillatore e non su quelli passivi (quarzi, resistenze, condensatori)
Ogni µC, per funzionare, ha sempre bisogno di componenti esterni all'integrato.

Se come componenti **attivi** dell'oscillatore vengono usati quelli **interni all'integrato**, allora è sufficiente applicare, come **rete esterna, soltanto un circuito passivo** basato o su quarzo[3] e condensatori oppure di tipo RC, circuito che viene collegato fra i pin 13 e 14: si parla in questo caso di **oscillatore interno**.

Se invece NON vengono usati, come componenti attivi dell'oscillatore, quelli interni all'integrato, allora si parla di oscillatore esterno.

OSCILLATORE INTERNO

L'oscillatore interno può funzionare in quattro differenti modalità.
Di queste modalità, tre richiedono un quarzo o un risonatore, mentre una quarta richiede un circuito RC.

Queste modalità:

- richiedono ciascuna gli opportuni componenti passivi applicati esternamente all'integrato
- possono essere selezionate via software mediante i bit dei registi OPTION, INCON, T1CON

oscillatore interno

MODALITA'	DESCRIZIONE	Range di FREQUENZA del QUARZO/RISUONATORE	CORRENTE massima assorbita	NOTE
LP	Low Power	solo quarzo con f < 200KHz	Decine di µA	
XT	XTal (crystal)	quarzo o risonatore ceramico con f < 4MHz	5mA	
HS	High Speed	quarzo o risonatore ceramico con f < 20MHz	10mA	
RC	Circuito con resistenze e condensatori	f < 10MHz, dipendente dai valori di R, di C e di VDD		3,3KΩ < R < 100 KΩ 20pF < C < 300 pF

OSCILLATORE ESTERNO

Per utilizzare un oscillatore esterno bisogna selezionare una delle tre modalità LP, XT, HS (quelle che, con oscillatore interno, prevedono la presenza di un quarzo o di un risonatore).
Non va invece selezionata la modalità RC

[3] o risonatore ceramico

MODULI TIMER

MODULO TIMER 0

E realizzato con un **registro** a **8 bit** a lettura e scrittura e può essere programmato per il funzionamento da contatore o da ***timer***.
Il registro è incrementato o dal ***clock interno di sistema*** o da un segnale esterno non necessariamente periodico applicato al pin RA4/TOCKI.

MODO TIMER

In assenza di prescaler (cioè senza divisore di frequenza) il registro viene ***incrementato ogni ciclo-macchina*** T_{CY} ,
con $T_{CY} = 4 \cdot T_{OSC}$

SELEZIONE:
si pone uguale a ***zero*** il bit ***TOCS*** del registro ***OPTION***

MODO CONTATORE

In assenza di prescaler (cioè senza divisore di frequenza) il registro viene incrementato a ogni fronte di salita del segnale esterno (non necessariamente periodico) , presente sul pin RA4/TOCKI.
Il fronte di comando è programmabile attraverso il bit TOSE del file register OPTION

In assenza di prescaler (cioè senza divisore di frequenza) il registro viene incrementato ogni ciclo-macchina

SELEZIONE:
si pone uguale a **uno** il bit **TOCS** del registro **OPTION**.

MODULO TIMER 1

E' realizzato con un **registro a 16 bit** a lettura e scrittura e può essere programmato per il funzionamento da contatore o da ***timer***.
Il **registro a 16 bit** è formato dai **due registri a 8 bit** (TMR1H e TMR1L).

La programmazione di TIMER 0 avviene attraverso il **bit TMR1CS**
del registro (file register) di controllo **T1CONl**

MODO TIMER

La sorgente di ***clock*** è ***interna*** (ricavata dal clock di sistema) con frequenza $\mathbf{f_{OSC}/4}$
In assenza di prescaler (cioè senza divisore di frequenza) il registro viene ***incrementato ogni ciclo-macchina*** T_{CY} ,
con $T_{CY} = 4 \cdot T_{OSC}$

MODO CONTATORE sincrono/asincrono

La sorgente di clock è esterna
Il registro TRMR1 è incremento sul fronte di salita del segnale di clock esterno a ogni fronte di salita del segnale esterno

MODULO TIMER 2

E' realizzato con un **timer a 8 bit** comandato dal ***clock interno*** di sistema con frequenza $\mathbf{f_{OSC}/4}$
La programmazione di TIMER 2 avviene attraverso il registro (file register) di controllo **T2CON**

E' anche utilizzato come base dei tempi per i **moduli CCP (Capture/Compare/PWM)**, dove:

- ***Capture = Cattura di dati o di eventi***
- ***Compare = Confronto di dati o di eventi***
- ***PWM = generazione di un segnale con modulazione di durata degli impulsi (Pulse Width Modulation), tipicamente ai fini del controllo***

RESET

Il **reset** è la **disattivazione del funzionamento del microcontrollore** e viene –fra l'altro- imposto al µC, per garantirne un corretto funzionamento, **quando i parametri elettrici dell'integrato non hanno ancora raggiunto i valori ottimali o si sono discostati da tali valori**.

I parametri elettrici che garantiscono il corretto funzionamento della MCU sono la **tensione di alimentazione**, la **frequenza del segnale di clock** e **la temperatura**.

Le condizioni di funzionamento nelle quali si ha abitualmente uno scostamento della tensione di alimentazione e/e della frequenza di clock dai valori di regime sono:

- la "accensione" del sistema
- imprevisti e indesiderati cali della tensione di alimentazione

SEQUENZA INIZIALE DI RESET

Sono effettuabili diverse tipologie di reset.
La "sequenza" di reset che viene realizzata tipicamente alla "accensione" del microcontrollore prevede, nell'ordine:

1. **POR (Power-On Reset o "reset di accensione")**
   E' un reset che **avviene sempre** e che **salvaguarda** il PIC dalla **tensione insufficiente**.
   Il POR viene imposto all'integrato quando si dà, inizialmente, tensione ad esso, cioè nella fase transitoria iniziale nella quale la tensione di alimentazione, **partendo da 0V,** non **raggiunge un valore compreso fra 1,2V e 1,7V**

2. **PWRT (Power-up Timer Reset o "reset del timer di power-up")**
   E' un reset che **NON avviene sempre** in quanto può essere attivato o disattivato via software.
   Il PWRT **salvaguarda** il funzionamento del PIC dagli effetti di una **tensione di alimentazione** quando essa, pur avendo superato la soglia minima garantita nel corso del primo reset (POR), **non ha ancora raggiunto il valore di regime**.
   IL PWRT genera un **ritardo di 72 mS** ed è prodotto da un oscillatore RC integrato nel PIC.

3. **OST (Oscillator Start-up Timer reset o "reset di start-up")**
   E' un reset che **avviene sempre** in fase di "accensione del PIC[4] e **salvaguard**a il funzionamento dell'unità centrale (CPU) del PIC dagli effetti di una **non ancora raggiunta stabilizzazione della FREQUENZA di CLOCK.**
   L'OST genera un **ritardo pari a 1024 periodi del segnale di clock presente sul pin** OSC1.

---

[4] L'attivazione è prevista però solo nelle modalità LP, XT e HS dell'oscillatore interno (modalità che prevedono la presenza di quarzi o oscillatori ceramici. L'OST viene sempre attuato anche in fase wake-up, cioè nel passaggio dalla modalità sleep alla modalità normale di funzionamento.

RESET DA INSUFFICIENTE TENSIONE DI ALIMENTAZIONE (BOR o RESET DA BROWN_OUT[5])

IL BOR viene imposto, da uno specifico circuito interno, al PIC quando
la **tensione di alimentazione scende al di sotto di 4V.**
Questo calo di tensione si può produrre, per esempio, nel corso della commutazione acceso/spento di carichi di potenza (sistemi in alternata e automotive)

ALTRE MODALITA' DI RESET

Il microcontrollore può essere anche posto in reset:

- Attraverso il pin $\overline{\text{MCLR}}$ (Master CLear Reset) durante il normale funzionamento
- Attraverso il pin $\overline{\text{MCLR}}$ (Master CLear Reset) durante il funzionamento SLEEP
- Da parte del watchdog durante il normale funzionamento (WTD reset)
- Da parte del watchdog durante il funzionamento SLEEP (WTD wake-up).

[5] In senso stretto "brown-out" significa "illuminazione insufficiente"

WATCHDOG[6]
O
WTD

Il **watchdog** è un **contatore**, *pilotato da un oscillatore RC (diverso dall'oscillatore principale e indipendente da esso)* che si trova **all'interno dell'integrato microcontrollore** e che **non** ha bisogno di nessun componente esterno per funzionare.

Il WTD **serve a evitare** che l'**esecuzione di un programma si blocchi per motivi accidentali**.

Siccome però il WTD determina[7] sistematicamente, dopo un certo intervallo di tempo[8], un RESET del microcontrollore (che è indesiderato durante la normale esecuzione di un programma e costringerebbe il sistema a ricominciare dall'inizio l'esecuzione del programma), risulta necessario introdurre, all'interno dei programmi stessi[9], un'istruzione di azzeramento del watchdog. Infatti, azzerando il conteggio del WTD, si evita il reset del sistema (dal momento che si impedisce al contatore WTD di andare in overflow e di di emettere l'impulso di time-out[10])

IL WTD può essere abilitato o disabilitato in fase di programmazione, il che significa che non tutte le applicazioni (non tutti i programmi) hanno bisogno di esso per lavorare.

Il ciclo del WTD è di[11] circa 18 ms. Ogni 18 ms il WTD genera un reset.
Mediante un PRESCALER (che è un dispositivo, anch'esso interno all'integrato MCU, che permette di dividere la frequenza secondo un determinato fattore) il ciclo può essere "allungato" fino a 2 o 3 secondi (corrispondenti rispettivamente a 0,50 Hz e a 0,33 Hz)

COSA FA IL MICROCONTROLLORE QUANDO IL WTD EMETTE L'IMPULSO DI FINE-CONTEGGIO

Quando il watchdog emette un impulso di fine conteggio (o impulso di "time-out), cioè quando va in overflow, il microcontrollore:

- se si trovava in funzionamento NORMALE va in RESET e ricomincia dall'inizio l'esecuzione del programma

- se si trovava in funzionamento SLEEP (cioè in condizione di "sonno", esce dalla modalità SLEEP di funzionamento e continua l'esecuzione del programma in modo normale.

---

6 "cane da guardia"

7 (emettendo un impulso di "time-out" o di "fine-conteggio")

8 (quando, dopo aver raggiunto il massimo valore di conteggio, va in overflow)

9 più esattamente, in quei programmi che prevedono l'uso del watchdog: non tutti i programmi infatti richiedono l'utilizzazione di questo dispositivo

10 "**time-out**" significa "tempo di attesa"
"**time-brown**" significa "tempo di oscuramento parziale" o "tempo di luce ridotta"

11 corrispondenti a una frequenza di circa 55 Hz

MODALITA' SLEEP
E WATCHDOG (WTD)

Precisiamo che per *sleep* (sonno, riposo) si intende una modalità di funzionamento caratterizzata essenzialmente:

- da un **bassissimo assorbimento di corrente**[12]
- dallo **spegnimento dell'oscillatore principale** (ma non di quello del WTD)
- dal **mantenimento dello stato precedente** da parte di tutte le **porte di I/O**

COME SI PORTA IL µC IN STATO DI SLEEP

Mediante l'istruzione SLEEP

COME SI FA USCIRE IL µC DALLO STATO DI SLEEP

- mediante un reset esterno applicato al piedino MCLR (Master Clock Reset)
- mediante un impulso di time-out emesso dal WTD
- mediante un'interruzione esterna.

[12] Circa 2 µA invece di 2 mA

MODULI CCP
(MODULI CAPTURE / COMPARE/ PWM

Sono tre moduli ciascuno dei quali permette alla MCU di funzionare in una particolare modalità, ossia di svolgere una delle tre funzioni capture, compare o PWM, dove:

- ***Capture = Cattura di dati o di eventi***
- ***Compare = Confronto di dati o di eventi***
- ***PWM = generazione di un segnale con modulazione di durata degli impulsi (Pulse Width Modulation), tipicamente ai fini del controllo***

USART (RiceTrasmettitore Universale Sincrono/Asincrono)
O
SCI (Interfaccia per le Comunicazioni Seriali)

Il modulo USART è uno dei due **moduli seriali di ingresso/uscita** della MCU

L'USART può essere configurato:

- o come sistema **A**sincrono **Full**-duplex[13], che può comunicare con dispositivi periferici come terminali con CRT (monitor) e personal computer

- o come sistema sincrono Half-duplex[14] che può comunicare con dispositivi periferici come convertitori integrati A/D e D/A, EEPROM seriali ecc.

---

[13] in grado cioè di comunicare in modo **bidirezionale e simultaneo**, ossia di trasmettere e ricevere contemporaneamente

[14] in grado cioè di comunicare in modo bidirezionale, ma non simultaneo, ossia di trasmettere e ricevere, ma non contemporaneamente

FILE REGISTER
del
PIC 16 F 877 A

FILE REGISTER
del
PIC 16F877A

PREMESSA:
LE "PORTE"

La porta è una "periferica" di I/O. Per PORTA si intende un GRUPPO di LINEE di INGRESSO/USCITA (e quindi un gruppo di piedini dell'integrato MCU).

A ogni PORTA, cioè a ogni gruppo di linee di I/O, è associata una COPPIA di FILE REGISTER, cioè una coppia di registri della memoria-dati RAM, i quali hanno il compito di configurare le linee come ingressi o come uscite e di gestire il trasferimento dei segnali attraverso le linee stesse.

Per esempio, al gruppo di linee che firmano la porta "A", sono associati i file register "PORT A" e "TRIS A".

I FILE REGISTER

PIC 16F877A

FILE REGISTER	NUMERO di BIT IMPLEMENTATI
PORT A	*6*
TRIS A	*6*
PORT B	8
TRIS B	8
PORT C	8
TRIS C	8

PORT A

File register PORT A

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	**X**	**RA5/AN4/SS/C2OUT**	**RA4/TOCKI/C1OUT**	**RA3/ AN3/ V^+_{REF}**	**RA2 /AN2/ V^-_{REF}/CV REF**	**RA1/AN1**	**RA0/AN0**

RA5 = bit di I/O o ingresso analogico o pin di uscita del comparatore
RA4 = bit di I/O o ingresso per il clock esterno
RA3 = bit di I/O o ingresso analogico o V^+_{REF}
RA2 = bit di I/O o ingresso analogico o V^-_{REF}
RA1 e RA0 = bit di I/O o ingresso analogico

Il file register PORT A è un registro, cioè un insieme di locazioni (sei bit utilizzabili per il PIC 16F877A) della memoria-dati RAM per il transito, ossia per la memorizzazione temporanea, dei dati che devono passare attraverso le linee che formano la "porta" indicata con "A".

Il dato presente su una certa linea della porta A è memorizzato nel corrispondente bit del registro "PORT A" e entra o esce dalla MCU a seconda che il corrispondente bit del file register "TRIS A" sia stato impostato al valore UNO (INGRESSO) o al valore ***zero*** (***uscita***).

Possiamo dire che il file register "PORT A" serve a:

- nelle operazioni di ***output*** (uscita dei dati dalla MCU), ***scrivere*** , cioè ***assegnare*** un determinato valore (1/0) alle singole linee che formano la porta associata al file register "PORT A" stesso
- nelle operazioni di INPUT (uscita nella MCU dei dati provenienti dall'esterno), LEGGERE un determinato valore (1/0) dalle singole linee che formano la porta associata al file register "PORT A" stesso.

TRIS A

Serve per configurare, singolarmente e indipendentemente, le linee del gruppo di I/O che formano la "porta A" (sei bit utilizzabili per il PIC 16F877A) come ingressi oppure come uscite:
valore UNO → (INGRESSO)
valore ***zero*** → (***uscita***).

File register TRIS A

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
X	**X**						

Tutti i bit di TRIS A sono dei "data direction bit" e servono per la configurazione come ingresso (1) o come uscita (0) dei corrispondenti bit di PORT A.

PORT B

I file register PORT B e TRIS B sono simili, ma non uguali ai registri, di tipo analogo, PORT A e TRIS A

Le differenze fondamentali fra le due coppie di registri sono essenzialmente:

- PORT B e TRIS B implementano **8 bit** (mentre PORT A e TRIS A implementano 6 bit)

- i bit di PORT B (contrariamente a quelli di PORT A) offrono la possibilità di realizzare un **pull-up**

- i segnali multiplexati nei pin di PORT B (cioè le "seconde funzioni" eventualmente effettuabili dai singoli terminali) sono diversi dai segnali multiplexati nei pin di PORT A

Notiamo che le differenze sono sostanzialmente fra i registri "PORT", mentre i registri "TRIS" sono, a parte il numero dei bit, sostanzialmente identici

File register PORT B

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RB7/PGD	**RB6/PGC**	**RB5**	**RB4**	**RB3/PGM**	**RB2**	**RB1**	**RB0/INT**

Tutti i bit (contrariamente a quelli di PORT A) offrono la possibilità di realizzare un **pull-up** internamente programmabile

RB7/**PGD** = bit di I/O o piedino per il **D**ebbugging "in circuit" e la **P**rogra**M**mazione
RB6/**PGC** = bit di I/O o piedino per la **P**rogra**M**mazione "in circuit" (del **C**lock Seriale)

RB5, RB4, RB2, RB1 = bit di I/O

RB3/**PGM** = bit di I/O o pin programmabile in modalità a bassa tensione
(LVP; **L**ow Voltage **P**rogra**M**min**G**)

RB0/**INT** = bit di I/O o pin per l'ingresso di un INTerrupt esterno

TRIS B

Esattamente come TRIS A, ma con **8 bit implementati** invece di 6 (per il PIC 16F877A), TRIS B serve per configurare, singolarmente e indipendentemente, le linee del gruppo di I/O che formano la "porta B" come ingressi oppure come uscite:

valore UNO → (INGRESSO)
velore ***zero*** → (***uscita***).

File register TRIS B

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0

Tutti i bit di TRIS B sono dei "data direction bit" e servono per la configurazione come ingresso (1) o come uscita (0) dei corrispondenti bit di PORT B.

PORT D e TRIS D

PORT D

PORT D è, nel PIC16877A, un file register a 8 bit (tutti implementati), ciascuno con buffer di ingresso a trigger di Schmitt.
Ogni bit è configurabile come ingresso o come uscita attraverso il file register TRIS D

File register PORT D

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
RD7/PSP7	**RD6/PSP6**	**RD5/PSP5**	**RD4/PSP4**	**RD3/PSP3**	**RD2/PSP2**	**RD1/PSP 1**	**RD0/PSP0**

(dove PSP = Parallel Slave Port)

OGNUNO degli otto PIN è configurabile come ingresso o come uscita

TRIS D

TRISD è, nel PIC16877A, un file register a 8 bit (di cui sette implementati), che ha la funzione di configurare come uscite o come ingressi i bit di PORT D e quindi le linee della porta D

File register PORT D

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IBF	**OBF**	**IBOV**	**PSP MODE**	**X**	*data*	*direction*	*bits*
Input BuFfer	**Output BuFfer**	**Input BuFfer**	**Modalità Parallel Slave Port**	**X**			

OGNUNO degli otto PIN è configurabile come ingresso o come uscita.

ADCON 0

REGISTER 11-1: 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 (ADCS 1) e Bit 6 (ADCS 0)

ADCS = A/D Clock Selection
CHS = CHannel Selection
Questi bit, insieme al bit 6 (ADCS 2) di ADCON 1, servono per la selezione della frequenza di clock in base alla tabella seguente:

ADCON 1 bit 6 (ADCS 2)	**ADCON 0 ADCS 1**	**ADCON 0 ADCS 0**	**Frequenza di clock**	**NOTE**
0	0	0	Fosc/2	Fosc/N = frequenza dell'oscillatore divisa per N FRC= frequenza dell'oscillatore RC (interno al modulo convertitore, integrato nel microcontrollore)
	0	1	Fosc/8	
	1	0	Fosc/32	
	1	1	FRC	
1	**0**	**0**	Fosc/4	
	0	**1**	Fosc/16	
	1	**0**	Fosc/64	
	1	**1**	FRC	

Bit 5 (CHS 2), Bit 4 (CHS 1) e Bit 3 (CHS 0)

CHS sta per "Channel Selection".
Sono i bit per la scelta del canale analogico[15].

Bit 2 (GO/DONE)

E' il bit di stato della conversione:
1 = conversione in corso
0 = conversione NON in corso.

Bit 1 (U-0)

Non è implementato ed è letto come zero.

Bit 0 (AD ON)

E' il bit di attivazione (ON) della conversione A/D (AD)
1 = il modulo A/D è alimentato
0 = il modulo A/D è NON alimentato

---

[15] Canale analogico della porta selezionata, per esempio PORT D
000 → canale 0 → AN. 0 [bit 0]
001 → canale 1 → AN. 1 [bit 1]
….....
111 → canale 7 → AN. 7 [bit 7]

ADCON 0

11.0 ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) Converter module has five inputs for the 28-pin devices and eight for the 40/44-pin devices.

The conversion of an analog input signal results in a corresponding 10-bit digital number. The A/D module has high and low-voltage reference input that is software selectable to some combination of VDD, VSS, RA2 or RA3.

The A/D converter has a unique feature of being able to operate while the device is in Sleep mode. To operate in Sleep, the A/D clock must be derived from the A/D's internal RC oscillator.

The A/D module has four registers. These registers are:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)

The ADCON0 register, shown in Register 11-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 11-2, configures the functions of the port pins. The port pins can be configured as analog inputs (RA3 can also be the voltage reference) or as digital I/O.

Additional information on using the A/D module can be found in the PICmicro® Mid-Range MCU Family Reference Manual (DS33023).

REGISTER 11-1: 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

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

© 2003 Microchip Technology Inc.

ADCON 1

ADCON1 è un file register a sei bit per il **con**trollo della conversione **A/D**

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

ADFM = bit per la selezione del **ForMato** del risultato della conversione **A/D**

ADCS 2 = bit per la **Selezione** del **Clock** per la conversione **A/D**

PCFG = bit per il controllo della **ConFigurazione** di **P**ORT A

ADCON 1

REGISTER 11-2: 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

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

Note: On any device Reset, the port pins that are multiplexed with analog functions (ANx) are forced to be an analog input.

© 2003 Microchip Technology Inc.

FILE REGISTER "STATUS"
O
REGISTRO DI STATO

Il registro di stato, a otto bit, contiene lo "stato aritmetico" della ALU, lo stato del RESET e I bit per la SELEZIONE dei banchi[16] della memoria-dati.

File register STATUS

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
IRP	**RP1**	**RPO**	**$\overline{TO}$**	**$\overline{PD}$**	**Z**	**DC**	**C**
INDIRECT ADDRESSING REGISTER	DIRECT ADDRESSING REGISTER	DIRECT ADDRESSING REGISTER	**Time Out bit**	**Power Down bit**	**Zero bit**	**Digit Carry/borrow bit**	**Carry/borrow bit**
selezione	***banchi***	***memoria***					

Il registro di stato può rappresentare la destinazione di alcune istruzioni così come possono esserlo altri registri.
.
Quando il registro di stato è la destinazione di una istruzione che interessi i bit Z, DC oppure C, allora la scrittura di questi 3 bit viene disabilitata.
Questi bit sono impostati a UNO (cioè "settati") oppure azzerati in accordo con la logica del dispositivo.

I bit **TO** (time out) e **PD** (power down) sono di **sola lettura**, ossia non si può scrivere in essi.

Si raccomanda di utilizzare, per modificare il registro di stato, solo le istruzioni.
BCF, BSF, SWAPP e MOVWF perché esse non interessano i bit Z, C e DC.

---

16 **BANCHI di MEMORIA**

Il banco è una "porzione" di memoria.
La memoria-dati EEPROM del microcontrollore è divisa in un certo numero di banchi (quattro per il PIC 16F877A).
Ogni banco contiene locazioni di memoria.
Ciascuna locazione può essere costituita, per esempio, da un byte di memoria.
I differenti file register della memoria-dati possono trovarsi alcuni in un banco, altri in un altro banco.
Se un dato deve essere posto in un registro che si trova in un determinato banco, è il microcontrollore che provvede a inviare il dato nel giusto banco (quello che contiene il registro di destinazione del dato) e al giusto indirizzo (l'indirizzo del registro).

I BIT DEL REGISTRO DI STATO

IRP = Indirect Addressing Register P=?

E' il bit di indirizzamento indiretto per la selezione del banco di memoria dati
IRP = 1 → banchi 2 e 3
IRP = 0 → banchi 0 e 1

RP =Direct Addressing Register P=?

E' il bit di indirizzamento DIRETTO per la selezione del banco di memoria.

RPI	RPO	Banco
1	1	3
1	0	2
0	1	1
0	0	0

$\overline{TO}$ = bit di Time Out

$\overline{TO}$ = 0 si è verificato un time-out del watchdog (WTD)

$\overline{TO}$ = 1 (valore assunto dal bit **$\overline{TO}$** dopo l' accensione (power- on) in seguito alle istruzioni **CLEAR WTD** e **SLEEP**).

$\overline{PD}$ = bit di Power Down

$\overline{PD}$ = 0 : (si ha in seguito all'esecuzione dell'istruzione di SLEEP)

$\overline{PD}$ = 1 (il livello alto di questo bit viene assunto, dopo l' accensione (power- up) con l'istruzione **CLEAR WTD)**

Z = bit di Zero

Z =0: il risultato di un'operazione aritmetica o logica è **zero**
Z = 1: il risultato di un'operazione aritmetica o logica **non** è zero

DC = bit di " Digit $\overline{\text{Carry}}$/Borrow "

DC= 0 : si è verificato un riporto dal quarto bit meno significativo del risultato
DC= 1 : **non** si è verificato un riporto dal quarto bit meno significativo del risultato

(Per **$\overline{\text{borrow}}$** si scambiano lo zero e l'uno)

C = bit di $\overline{\text{Carry}}$/Borrow

C= 0 : si è verificato un riporto sul bit più significativo del risultato
C= 1 : **non** si è verificato un riporto sul bit più significativo del risultato

(Per **$\overline{\text{borrow}}$** si scambiano lo zero e l'uno)

FILE REGISTER "OPTION"
O
OPTION_REG

E' utilizzato per effettuare varie **configurazioni del sistema**.
E' un registro a **lettura e scrittura**, a **otto** bit, che contiene **bit di controllo** i quali servono a configurare:

- il **PRESCALER**[17] del timer TMR0 (o il prescaler del watchdog)
- l'**interruzione** esterna INT
- il timer **TMRO**
- le **resistenze di pull-up** della porta **B**

File register OPTION

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$\overline{\text{RBPU}}$	**INTEDG**	**TOCS**	**TOSE**	**PSA**	***PS2***	***PS1***	***PS0***
PORT B PULL-UP ENABLE BIT	INTERRUPT EDGE SELECTION BIT	TIMER 0 CLOCK SELECTION BIT	TIMER 0 SOURCE EDGE SELECTION BIT	PRESCALER assignment BIT	*PRESCALER RATE SELECTION BIT*	*PRESC. RATE SELECTION BIT*	*PRESC. RATE SELECTION BIT*
Bit di **abilitazione** delle **resistenze di pull-up** della **porta B**	Bit di **selezione** del **fronte di salita/discesa** del segnale sul pin RBO/INT	Bit **di selezione** della **sorgente** di **clock**	Bit di **selezione** del **fronte di salita/discesa** del segnale della **sorgente di clock**	Bit di **assegnazione** del **prescaler**	*bit di selezione del FATTORE di DIVISIONE del PRESCALER*	*bit di selezione del FATTORE di DIVISIONE del PRESCALER*	*bit di selezione del FATTORE di DIVISIONE del PRESCALER*

Possibili configurazioni dei bit di selezione del fattore di divisione del prescaler per il timer TMR 0 e per il WTD

PS2 PS1 PS0	Fattore di divisione per TMR 0	Fattore di divisione per WTD
0 0 0	1/**2**	1/**1**
0 0 1	1/4	1/2
0 1 0	1/8	1/4
0 1 1	1/16	1/8
1 0 0	1/32	1/16
1 0 1	1/64	1/32
1 1 0	1/128	1/64
1 1 1	1/**256**	1/**128**

[17] Dispositivo **divisore di frequenza**, presente nella MCU, applicabile al timer o al WTD, secondo un fattore di divisione selezionabile via software

CONFIGURAZIONI DEI BIT
del File register OPTION

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
$\overline{\text{RBPU}}$	**INTEDG**	**TOCS**	**TOSE**	**PSA**	***PS2***	***PS1***	***PS0***
0 = pull-up abilitato **1 = pull-up disabilitato**	**0 = fronte di discesa** **1 = fronte di salita**	**0 = ciclo del clock interno all'istruzione** **1 = transizione del segnale presente sul pin RA4/TOCK1**	**0 = incremento sulla transizione basso → ALTO** **1 = incremento sulla transizione ALTO → basso**	**0 = PRESCALER assegnato al TIMER 0** **1 = PRESCALER assegnato al WTD**			

I FILE REGISTER
PCL E PCLATH
E IL CONTATORE DI PROGRAMMA PC

Possiamo dire che il **contatore di programma** è **un gruppo** di bit (costituito o realizzato da **uno o più registri**) che rappresenta **l'indirizzo della cella di memoria**[18] nella quale è contenuto **l'indirizzo della "prossima" istruzione** che il µC dovrà eseguire.

In maniera più sintetica si può dire che il contatore di programma è un **gruppo di bit che PUNTA alla cella contenente la prossima istruzione che il µC dovrà eseguire**.

A eccezione dei casi di salto o di chiamata di subroutine (sottoprogrammi), il µC nella fase di fetch dell'istruzione corrente, incrementa il contenuto del PC di una unità, in modo che punti alla cella di memoria che contiene la prossima istruzione

Nel PIC **16F877A** il **PC** è formato da **13 bit in totale**:

- gli otto bit meno significativi sono a lettura e scrittura e assumono lo stesso valore dei degli otto bit del file register PCL (Program Counter Low)

- i ***cinque*** bit ***più*** *significativi* prendono il nome di ***PCH*** (Program Counter High).

Il PCH però ***NON*** *è un file register*, ma solo l'insieme dei cinque bit più pesanti del PC, i quali *NON sono direttamente accessibili* alle operazioni di scrittura, ma *sono "scrivibili" attraverso il file register PCLATH e vengono azzerati ogni volta che si ha un reset*.

Nel PIC 16F877A il PC non è quindi fisicamente un registro, ma è un insieme di 13 bit.
Di questi bit, 8 sono in pratica coincidenti con il file register PCL e 5 non sono direttamente accessibili, ma modificabili attraverso un "*buffer di scrittura* costituito dal file register PCLATH.

---

[18] memoria-programma

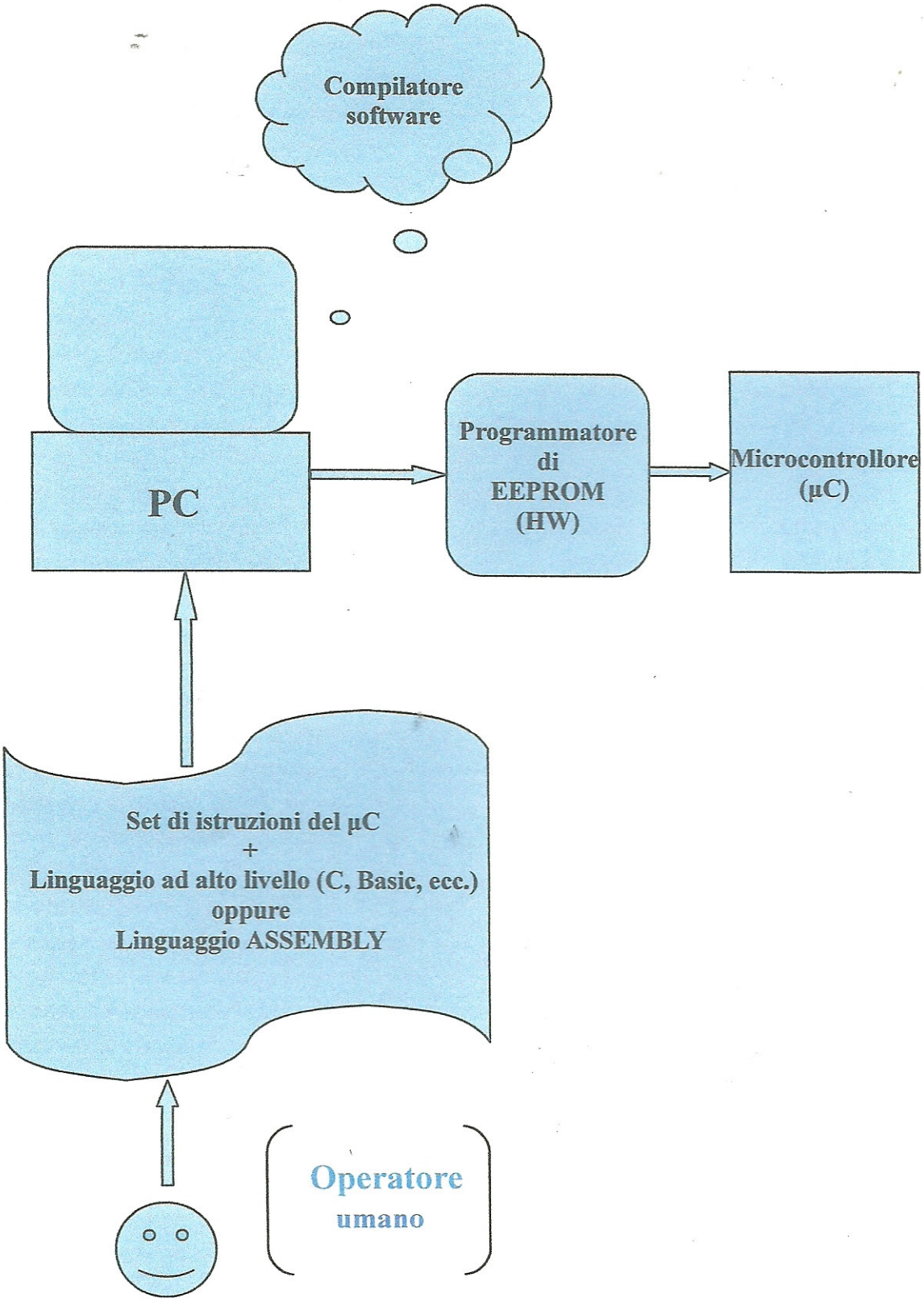
PROGRAMMAZIONE DEL PIC

COME SI PROGRAMMA, IN GENERALE, UN MICROCONTROLLORE

Il microcontrollore o MCU (MicroController Unit) o microcomputer single-chip è, come già accennato, un integrato digitale programmabile, cioè un integrato che può svolgere diverse funzioni in base al **programma** che gli è trasmesso, e che **viene immagazzinato** nella memoria **EEPROM** contenuta nell'integrato stesso.

Come tutti i dispositivi programmabili, il microcontrollore non è in grado di svolgere nessuna funzione se non ha ricevuto istruzioni di programma.

PROGRAMMAZIONE DEL PIC



COME SI COMUNICA IL PROGRAMMA ALLA MCU

Essendo un dispositivo digitale, la MCU è in grado di memorizzare ed elaborare solo programmi che siano scritti in "linguaggio macchina" (LM), che è un linguaggio costituito solo da bit, ossia da simboli "0" e "1".
Siccome però è molto difficoltoso, per l'uomo, programmare direttamente in linguaggio macchina, viene adottata una tecnica che permette di programmare il mC con un linguaggio più vicino al linguaggio umano, utilizzando codici non binari, ma mnemonici, e un certo numero di parole-chiave.
Le parole-chiave vengono poi tradotte in LM da un apposito software chiamato "compilatore".
Questo linguaggio è formato:

- da codici mnemonici[19] (come per esempio MOVW e ADDLW) che rappresentano, in caratteri alfabetici, le istruzioni del "set di istruzioni" del microcontrollore; il set di istruzioni è un insieme di comandi che il mC è in grado di riconoscere ed elaborare "da solo", in quanto ciascuno di essi non è altro che l'indirizzo di un microprogramma già memorizzato nella EEPROM dal costruttore del mC

- dalle parole-chiave del linguaggio "assembly[20]" oppure di un linguaggio ad alto livello come per esempio il Basic, il Pascal, il "C"; queste parole chiave (espresse dall'operatore umano in caratteri alfabetici) devono essere tradotte in linguaggio macchina da un programma compilatore

COME SI PROGRAMMA -IN PRATICA- UN MICROCONTROLLORE
La programmazione di un microcontrollore può essere fatta utilizzando un PC, un software"compilatore", installato sul PC, e un integrato programmatore di EEPROM.

---

[19] Esempi di codici mnemonici di istruzioni (per il microcontrollore PIC16F877A) sono:
MOVW k
che significa:
"carica il numero k nel registro di lavoro W"
e
ADDLW k
che significa:
"somma il numero k al contenuto del registro di lavoro W e inserisci il risultato in W"

[20] Detto anche "assembler"

BIBLIOGRAFIA

Data sheet del PIC 16F877A
fascicolo "demoboard" da Futura elettronica
Cuniberti, De Lucchi, Galluzzo TDP ed. Petrini
De Santis, Cacciaglia, Saggese Corso di Sistemi ed.Calderini edagricole
Manuale della demoboard "easypic3" Mikroelektronika
Manuale "mikroC IDE"
www.docenti.org