MICROCONTROLLER



MICROCONTROLLER

Computers: le macchine che governano il mondo.

Attualmente è impossibile concepire il mondo nel quale viviamo senza computer. Se questi smettessero di funzionare la Terra girerebbe comunque, però gran parte delle cose che esistono su di essa si fermerebbero. Si interromperebbero i collegamenti telefonici, fax, e Internet. Gli spostamenti su terra, mare e cielo sarebbero seriamente penalizzati. Le industrie fermerebbero i loro processi produttivi e la commercializzazione dei loro prodotti. Gli ospedali non potrebbero eseguire determinati interventi. Persino a casa nostra avremmo dei problemi con l'ascensore, l'antifurto, ecc. Il caos sarebbe totale.

Il computer è uno degli strumenti più importanti fabbricati dall'uomo ed è da questo strumento che dipende il suo progresso.

IL NOSTRO PERSONAL COMPUTER

Per poter seguire con profitto questi fascicoli, dedicati all'affascinante tecnica dello sviluppo, costruzione, programmazione ed applicazione dei Robots intelligenti, avrete bisogno di un PC. Il computer vi permetterà di eseguire interessanti lavori, ad esempio:

LAVORI CON IL PC

- Poter vedere Microrobots commerciali con il CD.
- Seguire lo sviluppo costruttivo del Microrobots.
- Constatare il comportamento del Microrobot eseguendo i programmi che riceverete.
- Produrre voi stessi i programmi per pilotare il vostro nuovo "animale domestico".
- Poter visitare siti Internet da noi consigliati per ottenere informazioni sulle caratteristiche, prezzi, punti d'incontro e corsi di Microrobots.
- Risolvere i dubbi ed i problemi che nasceranno durante lo sviluppo dell'opera attraverso il supporto elettronico.

STRUTTURA DEL COMPUTER

Tutti i computer, dal PC a quelli molto più potenti che controllano un insieme di celle robotizzate per l'assemblaggio d'automobili, o come quelli piccoli di un Microrobot, hanno la stessa architettura fondamentale, composta dai seguenti tre elementi:



Il personal computer o PC sarà uno strumento indispensabile per operare con i microrobot.

MEMORIA

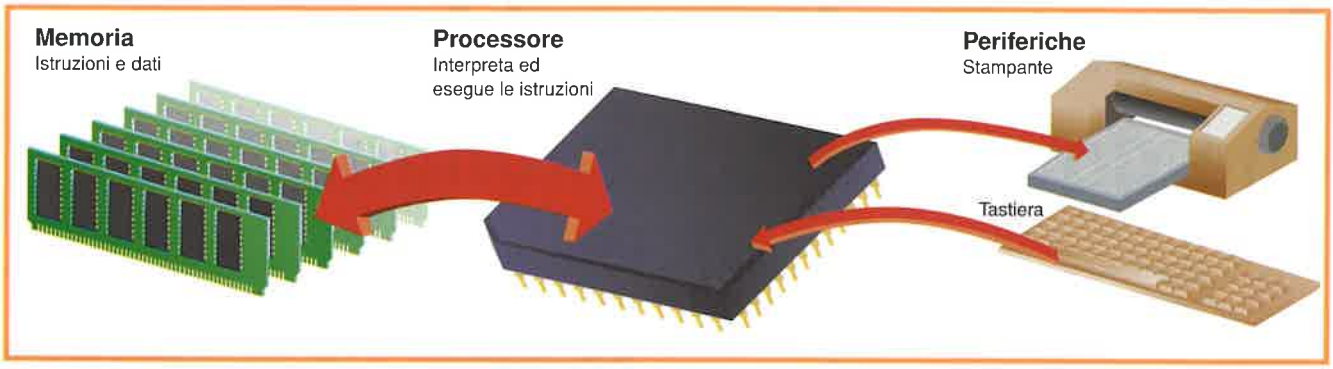
In questo elemento del computer vengono scritti i programmi che si vogliono utilizzare, assieme ai dati necessari ed ai risultati che vengono generati.

PROCESSORE

E' la parte più importante della macchina, esso deve interpretare le istruzioni del programma, ed eseguire le operazioni che competono ad ognuna. All'interno del processore, l'Unità di Controllo è quella che interpreta o traduce le istruzioni, quindi il

MICROCONTROLLER

Presentazione Generale



Un computer è composto da tre parti principali: memoria, processore e periferiche.

percorso dei dati esegue le operazioni impartite da ogni istruzione.

PERIFERICHE

Sono i dispositivi che utilizzano gli utenti, nei quali vengono caricate le informazioni e vengono raccolti i risultati ottenuti dall'analisi dei programmi e dei dati.

Le periferiche classiche sono la tastiera, il mouse, la stampante, il monitor, il modem, gli altoparlanti e la telecamera digitale, ma possono anche essere un PC motori, un display a sette segmenti, sensori di prossimità, ecc.

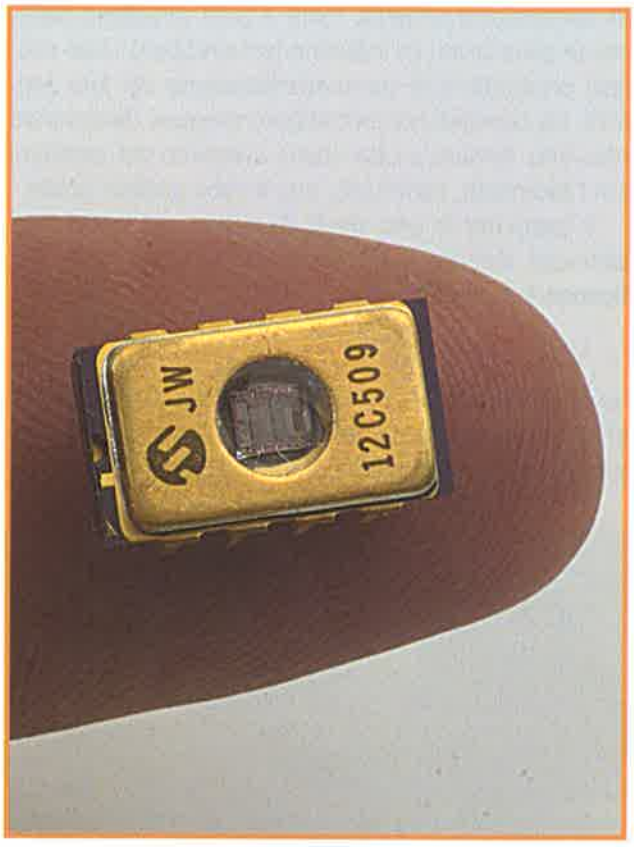
COS'È UN MICROCONTROLLER?

È un computer costruito all'interno di un circuito integrato. I progressi tecnologici dell'elettronica hanno permesso di implementare tutti i componenti di un computer dentro un chip di silicio in un solo circuito integrato.

Ad esempio, il Microcontroller della fotografia che viene evidenziato sul dito è un PIC12C509, ed è contenuto in un "case", o contenitore, di otto piedini, occupa un superficie di 12 mm. di lunghezza per 6 mm. di larghezza, pesa poco più di 3 gr., il prezzo può essere inferiore a 1 Euro, a seconda del numero di pezzi acquistati. Con le caratteristiche segnalate precedentemente è facile dedurre che un Microcontroller è un piccolo computer, con poca memoria, dove il processore è molto semplice, infatti può riconoscere soltanto poche istruzioni di base e le periferiche che possono essere collegate sono abbastanza limitate.

IL MICROCONTROLLER INTERNO, O EMBEDDED

I Microcontroller sono computer di dimensioni molto ridotte che possono essere situati nella stessa applicazione che gestiscono. La tastiera è un esempio di prodotto molto conosciuto, che ha al suo interno un Microcontroller, il quale individua il tasto scelto, genera i codici che lo identificano e lo invia al processore. Lo stesso succede con il mouse, al suo interno esiste un Microcontroller che conta, registra e trasmette al processore i segnali che vengono prodotti dai due encoders che controllano il movimento della pallina in X e Y; inoltre riconosce, codifica e trasmette i comandi dei tasti.



Un tipico Microcontroller sulla punta del dito.

MICROCONTROLLER

Presentazione Generale



La tastiera e il mouse sono due tipiche periferiche del PC, ognuna delle quali viene governata da un microcontroller che si trova al suo interno.

Un microcontrollore inserito nel prodotto che controlla, sa eseguire solamente un compito, motivo per il quale esegue sempre lo stesso programma: controlla dati che si assomigliano e controlla sempre le stesse periferiche. Un computer di utilizzo generale come un PC, possiamo utilizzarlo per disegnare un circuito elettronico, oppure per giocare.

Il microcontroller permette d'impiegare un computer per far funzionare tutti i prodotti, piccoli o economici che siano.

APPLICAZIONE DEI MICROCONTROLLER

Le possibilità d'utilizzo dei microcontroller sono limitate solo dall'ingegno dei suoi disegnatori. Chi avrebbe mai pensato, anni fa, che una radio avesse bisogno di un computer?

I ricevitori moderni, grazie al lavoro di uno o più controller, hanno cambiato immagine a questo classico prodotto. La sintonia delle emittenti migliora, si possono memorizzare le stazioni desiderate, programmarle in modo casuale, realizzare l'accensione in data ed ora prefissate e tantissime altre possibilità, con la capacità di aumentare la categoria e le prestazioni di questo apparecchio che prima era solamente una radio.

L'incorporazione dei microcontrollori a tutti i prodotti e sistemi che abbiamo, è in costante crescita, e



Il numero di Microcontroller che vengono incorporati nell'automobile è sempre maggiore.

durante il secolo XXI assisteremo ad un'invasione generalizzata. La distribuzione percentuale della applicazione dei microcontrollori nei diversi settori alla fine del secolo XIX è la seguente:

COMUNICAZIONE: 30%
Telefoni fissi e mobili. Sistemi di comunicazione. Navigazione.

PRODOTTI DI LARGO CONSUMO: 27%
Elettrodomestici. Macchine da gioco. Giocattoli. Sistemi di sicurezza ed allarmi. Climatizzazione.

AUTOMOBILI: 18%
Controllo del motore (consumo, gas di scarico, ecc.). Controllo di frenata (ABS). Airbag. Strumentazione. Controllo della velocità. Climatizzatore. Navigazione satellitare. Computer di bordo. Sistema antifurto.

PERIFERICHE INFORMATICHE: 15%
Tastiera. Mouse. Disco rigido. Gestione dell'immagine e del suono. Stampante. Multimedia.

INDUSTRIA: 10%
Strumentazione. Elettromedicina. Automazione. Robotica. Telecamere. Applicazioni militari. Domotica (automazione delle abitazioni civili).

MICROCONTROLLER

Presentazione Generale

MICROCONTROLLER IN CASA

CUCINA
Lavatrice 1
Forno 1
Frigorifero 1
Lavastoviglie 1

SALOTTO
TV 2
Radio 1
Stereo 1
Telefono 1
Telefono portatile 2

GARAGE
Accesso automatico 1
Controllo illuminazione 1
Antifurto – Antincendio 1

SCALA
Ascensore 2
Controllo illuminazione 1
Antincendio 1

GENERALE
Riscaldamento 1
Aria condizionata 1
Videocitofono 1
Antifurto, allarme antigas, antincendio 2

AUTOVETTURA
Freni ABS 1
Climatizzatore 1
Allarme 1
Airbag 1
Stereo 1
Computer di bordo 1
Check motore 1
Sicurezza 1

COMPUTER E PERIFERICHE
PC e periferiche associate 6

TOTALE: 37

QUANTI MICROCONTROLLER HO IN CASA?

Secondo la Dataquest, nell'anno 2000 erano 240 i microcontroller installati in una casa americana. A casa nostra, quanti ce ne sono? Facciamo un piccolo bilancio.

In breve aumenteranno le quantità a causa della imminente nascita del nostro microrobot Monty.

IL FUTURO

L'invasione dei microcontroller è appena iniziata. Ogni anno appaiono sul nostro mercato modelli più potenti, capaci di realizzare complessi lavori con l'obiettivo di automatizzare prodotti importanti. Però uno dei più appassionanti e con più futuro commerciale e sociale è quello relativo ai microrobots. Queste piccole macchine, diventeranno compagne inseparabili dell'uomo e presto eseguiranno con velocità ed efficienza molti lavori quotidiani che adesso svolgiamo noi.

Oltre ad aiutarci con i nostri lavori domestici, nel nostro lavoro e nei nostri viaggi, divideranno e parteciperanno ai nostri momenti di relax e lotteranno per diventare i "migliori amici dell'uomo".



Prototipo del sistema PALMA di aiuto alla mobilità per bambini con paralisi cerebrale. Istituto di Automazione Industriale (CSIC).

Classificazione, mercato e costruttori di microprocessori



Il microrobot della Husqvarna taglia continuamente l'erba del giardino alimentandosi con energia solare.

I MODELLI DEI MICROPROCESSORI

Il microprocessore è un computer in un chip, però le sue risorse e le sue capacità sono molto limitate. Si utilizza nel controllo di applicazioni concrete e di solito è inserito nel medesimo prodotto che controlla. A seconda delle applicazioni, le caratteristiche che deve possedere il microprocessore sono differenti. Quindi per governare un robot industriale, si deve possedere una memoria con una grande capacità, perché dovrà contenere tutti i programmi e i dati, i vari temporizzatori per controllare i tempi, i convertitori AD per il trattamento delle grandezze analogiche, i circuiti speciali di interfaccia per i motori e comunicazione seriale ad alta velocità per trasmettere informazioni e immagini delle telecamere che utilizza. Un microrobot della prestigiosa azienda HUSQVARNA taglia continuamente l'erba del giardino. Il microprocessore che lo governa deve disporre di circuiti per il controllo dei suoi motori, interfacce con i sensori che limitano la superficie di taglio e sistemi di carica delle batterie mediante energia solare.

Per pilotare un distributore automatico di lattine, il microprocessore deve solo contare le monete che vengono introdotte, gestire i pulsanti di selezione, il pannello di visualizzazione e i meccanismi che liberano il prodotto selezionato. A fronte di necessità così diverse, a seconda delle applicazioni i costruttori offrono una

MICROCONTROLLER

Modelli

enorme varietà di microprocessori: dai più semplici ed economici, destinati a prodotti comuni, sino a quelli più completi capaci di controllare sistemi avanzati.

CLASSIFICAZIONE GENERALE

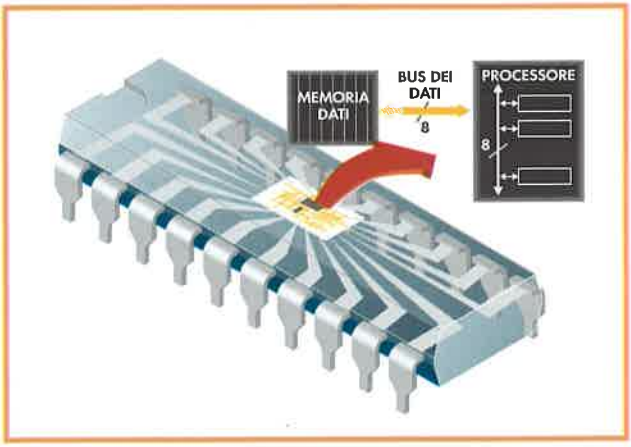
A seconda delle loro caratteristiche, i microprocessori si possono classificare in molti modi, il più significativo e utilizzato è quello riferito alla dimensione del bus dei dati. Il bus dei dati è l'insieme delle linee mediante le quali vengono trasferite le informazioni all'interno del processore. Ne esistono di quattro tipi:

- **Microprocessori a 4 bit**
- **Microprocessori a 8 bit**
- **Microprocessori a 16 bit**
- **Microprocessori a 32 bit**

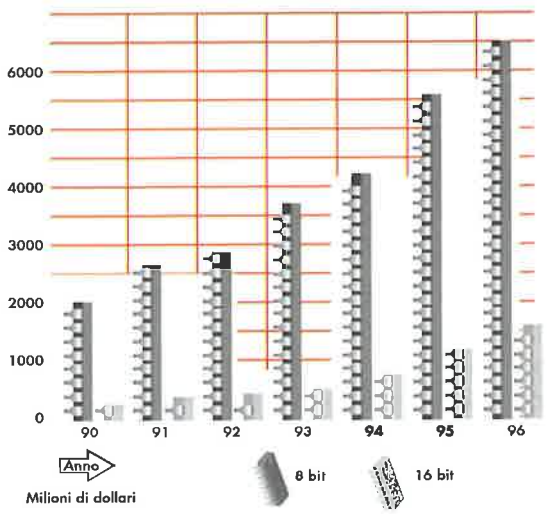
Quelli a 4 bit sono i più semplici e si utilizzano in piccoli automatismi e giocattoli. Vengono fabbricati in grandi quantità e risultano molto economici da installare nei prodotti di largo consumo. Quelli a 8 bit, sono i più utilizzati per la loro grande varietà e versatilità. Quelli a 16 bit sono impiegati in applicazioni più esigenti e, infine, quelli a 32 bit dominano il settore dei progetti più avanzati, come le applicazioni nell'intelligenza artificiale, oppure le applicazioni militari e dispositivi di immagazzinamento massivo di dati.

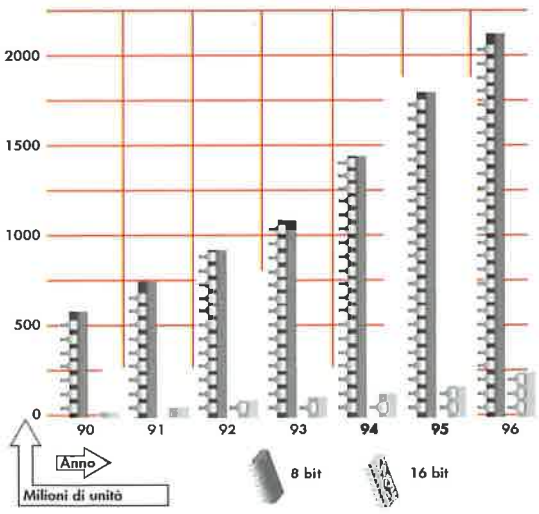
MICROPROCESSORI A 8 BIT

Sono i più usati e venduti, e dentro questa gamma si differenziano per le loro risorse e capacità. Un buon progettista deve scegliere il microprocessore che soddisfi tutte le specifiche del suo progetto, incluse le prevedibili espansioni future. I microprocessori a 8 bit sono caratterizzati dall'avere il bus dei dati da 8 linee, questo comporta che tutti i trasferimenti di informazioni all'interno della struttura del chip saranno realizzati a byte di 8 bit di dimensione, e la memoria dei dati avrà anch'es-



I microprocessori a 8 bit gestiscono dati di tale dimensione.





Evoluzione comparativa del mercato dei microprocessori a 8 e 16 bit.

sa un'architettura a 8 bit. La dimensione delle istruzioni può essere differente, e si adatterà al formato del microprocessore.

UN MICROPROCESSORE TIPICO

La struttura generale del microprocessore è composta da quattro parti:

Memoria di programma

Memoria dei dati

Processore

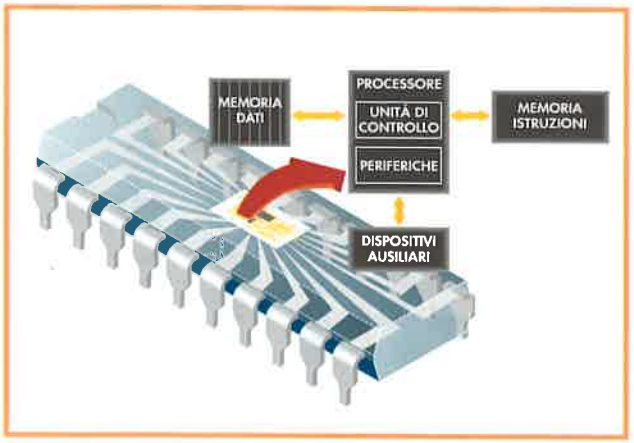
Dispositivi ausiliari

A seconda del modello, la capacità della memoria sarà diversa, così come il set di istruzioni che il processore può accettare, in sintonia alla dotazione di periferiche di cui è dotato.

MEMORIA DI PROGRAMMA

Contiene le istruzioni del programma che governa l'applicazione a cui è destinato il microprocessore. Dato che deve eseguire sempre lo stesso compito, deve essere una memoria non volatile, che non perda le informazioni memorizzate quando si scollega l'alimentazione. Esistono diversi tipi di memoria di programma:

• ROM mascherate: le istruzioni vengono scritte all'interno della memoria dal costruttore e non possono essere cancellate. Sono consigliabili per le grandi serie.



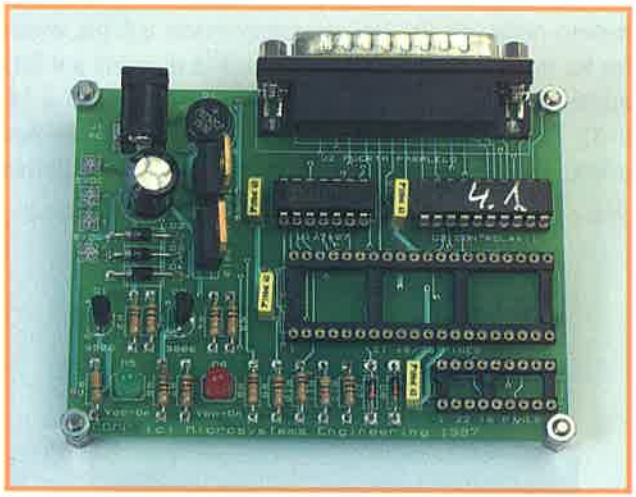
I quattro blocchi tipici di un microprocessore.

• OTP: la memoria viene scritta dall'utente finale tramite uno scrittore di memorie e un PC. Si possono solo programmare una volta e non si possono cancellare. Adatte per serie medio piccole, dato il loro basso prezzo.

• EPROM: mediante uno scrittore e un programma di comunicazione con il PC si memorizzano i programmi e si possono cancellare molte volte. Dispongono di una finestra di cristallo sulla parte superiore del contenitore per facilitare la cancellazione, che viene realizzata sottoponendo il chip a luce ultravioletta, per il periodo indicato dal costruttore.

• EEPROM: si scrivono e cancellano elettricamente senza necessità di raggi ultravioletti.

• FLASH: simili alle EEPROM. Le può scrivere e cancellare l'utente finale tramite un programmatore collegato a un PC.



Fotografia di un programmatore per microprocessori con memoria di programma OTP, EPROM, EEPROM e FLASH (Micro'PIC Programmer di Microsystems Engineering).

MEMORIA DEI DATI

Archivia le variabili di processo e i risultati temporanei. Deve permettere lettura e scrittura e, a seconda della tecnologia di fabbricazione, si divide in due categorie fondamentali:

• RAM: Memoria di lettura e scrittura molto rapida e volatile.

• EEPROM: Memoria di lettura e scrittura lenta, ma non volatile.

PROCESSORE

È la parte del computer incaricata di interpretare ed eseguire le istruzioni del programma. È formata da due

MICROCONTROLLER

Modelli

sezioni, l'Unità Centrale che traduce le istruzioni e la ALU (Unità Logico Aritmetica) che le esegue.

PERIFERICHE

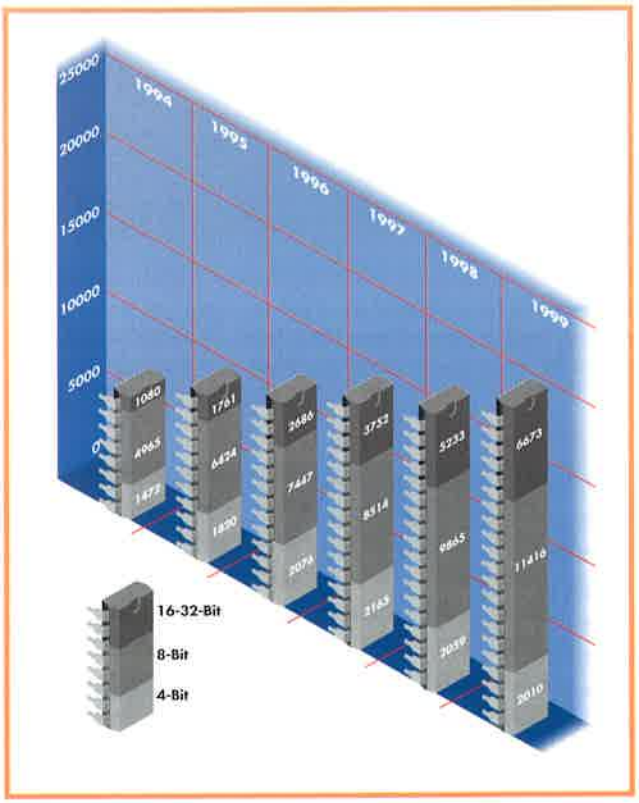
Questa parte del microprocessore contribuisce, insieme alla capacità della memoria, a determinare la potenza e la complessità del dispositivo. Alcune delle periferiche più importanti sono:

- Porte di ingresso e di uscita.
- Temporizzatori.
- Comparatori e campionatori di segnale.
- Convertitori AD e DA.
- Interrupt.
- Whatch dog.
- Sistema di stand-by.

MERCATO MONDIALE

La fine degli anni 90 ha consolidato un forte incremento nella vendita dei microprocessori a 8 bit, mentre ha mantenuto il volume di vendita di quelli a 4 bit, inoltre sono incominciati a crescere anche quelli a 16 e 32 bit. Se si suppone una media di 4 dollari per unità, il valore dei microprocessori venduti nell'ultimo anno del secolo scorso, arriva a molte centinaia di milioni. Il maggior consumo, circa la terza parte, si è verificato nel settore dei computer e relative periferiche.

La quarta parte è stata assorbita da prodotti di largo consumo, come elettrodomestici, giochi, TV, eccetera. Il settore delle telecomunicazioni, l'area industriale e l'automazione completano il resto.



La vendita in milioni di dollari dei microprocessori a 8 bit supera ampiamente la somma di tutti gli altri.

I COSTRUTTORI

I maggiori costruttori di circuiti integrati dispongono di una linea di microprocessori, date le enormi aspettative di mercato che si prevedono. INTEL è considerato come il padre dei microprocessori, perché fu il pioniere nella fabbricazione di questi componenti, con il modello 8048 negli anni 70 poco dopo aver commercializzato, nel 1971, il primo microprocessore del mondo. In seguito ottenne un grande successo con la produzione del microprocessore a 8 bit della famiglia 8051, alla quale seguì la 80C196 a 16 bit. Tuttavia è stata MOTOROLA che ha occupato il primo posto della classifica mondiale negli anni 90. Il modello a 8 bit 68HC11, costruito con tecnologia HCMOS, è capace di interpretare il repertorio completo di istruzioni del microprocessore m6800 più 91 nuove. L'azienda americana MICROCHIP, produttrice del microprocessore PIC, che utilizzeremo in questa opera per costruire il nostro robot, ha scalato la classifica in modo vertiginoso, passando dal 20° al 2° posto, in un arco di tempo che va dal 1990 al 2000.

Questa ascesa è basata sulle sue eccellenti prestazioni, che sono le stesse che ci hanno indotti a sceglierlo per la nostra macchina.

N°	1990 RANKING	1991 RANKING	1992 RANKING	1993 RANKING	1994 RANKING	1995 RANKING	1999 RANKING
1.	Motorola	Motorola	Motorola	Motorola	Motorola	Motorola	Motorola
2.	Mitsubishi	Mitsubishi	Mitsubishi	NEC	Mitsubishi	Mitsubishi	Microchip
3.	NEC	NEC	intel	Mitsubishi	NEC	NEC	
4.	Intel	Intel	NEC	Hitachi	Philips	Philips	
5.	Hitachi	Hitachi	Philips	Philips	Intel	Microchip	
6.	Philips	Philips	Hitachi	Intel	Microchip	SGS	
7.	Matsushita	Matsushita	Matsushita	SGS	SGS	Intel	
8.	National	SGS	SGS	Microchip	Matsushita	Hitachi	
9.	Siemens	Siemens	National	Matsushita	Hitachi	Toshiba	
10.	Ti	Ti	Ti	Toshiba	Toshiba	Fijitsu	
11.	Sharp	National	Zilog	National	National	Ricoh	
12.	Oki	Toshiba	Toshiba	Zilog	Ti	Matsushita	
13.	Toshiba	Sony	Siemens	Ti	Ricoh	Ti	
14.	SGS	Sharp	Microchip	Siemens	Fijitsu	National	
15.	Zilog	Oki	Sharp	Sharp	Zilog	Siemens	
16.	Matra MHS	Zilog	Sanyo	Oki	Siemens	Zilog	
17.	Sony	Microchip	Matra MHS	Sony	Sharp	Sony	
18.	Fujitsu	Matra MHS	Sony	Sanyo	Oki	Oki	
19.	AMD	Fujitsu	Oki	Fujitsu	Sony	Sanyo	
20.	Microchip	Sanyo	Fujitsu	AMD	Temic	Temic	

Spettacolare ascesa dell'azienda MICROCHIP, produttrice del microprocessore PIC.

Perché il PIC?



Presso i rivenditori che distribuiscono MICROCHIP in tutto il mondo, troviamo una straordinaria offerta di prodotti relativi ai PIC.

All'interno dell'ampio panorama di modelli e fabbricanti di microprocessori, qual è la regione che ci ha indotto a scegliere il PIC per la costruzione del nostro robot? Bisogna tenere conto che per ogni applicazione concreta esiste un microprocessore ottimale, però a seconda delle circostanze o delle prospettive per il futuro, il modello o il fabbricante possono cambiare. Questo significa che non esiste un microprocessore migliore in termini assoluti, dato che ogni caso risulta differente.

MICROCHIP ha scalato la classifica mondiale di vendita dei microprocessori a 8 bit, infatti dal 20° posto del 1990, è arrivata al 2° posto ai giorni nostri. Questa fantastica accoglienza da parte di industrie, professionisti e appassionati, è dovuta al fatto che i PIC hanno una dote molto particolare: questi microprocessori sono dotati di un complesso di caratteristiche medie che li avvantaggia notevolmente rispetto ai loro concorrenti. Inoltre l'immagine dell'impresa è aumentata di prestigio, e molte università e centri di formazione specializzati, hanno scelto questo fabbricante per dare supporto allo sviluppo dei loro programmi formativi. Nella tabella che segue elenchiamo gli aspetti più rilevanti del PIC:

L'ORGOGLIO DEI PIC
Semplicità di utilizzo
Eccellente documentazione e bibliografia
Prezzo molto conveniente
Magnifico insieme di parametri fondamentali
Strumenti di sviluppo economici e facili da utilizzare
Disegno e programmazione rapidi ed efficaci
Enorme varietà di modelli di microprocessori

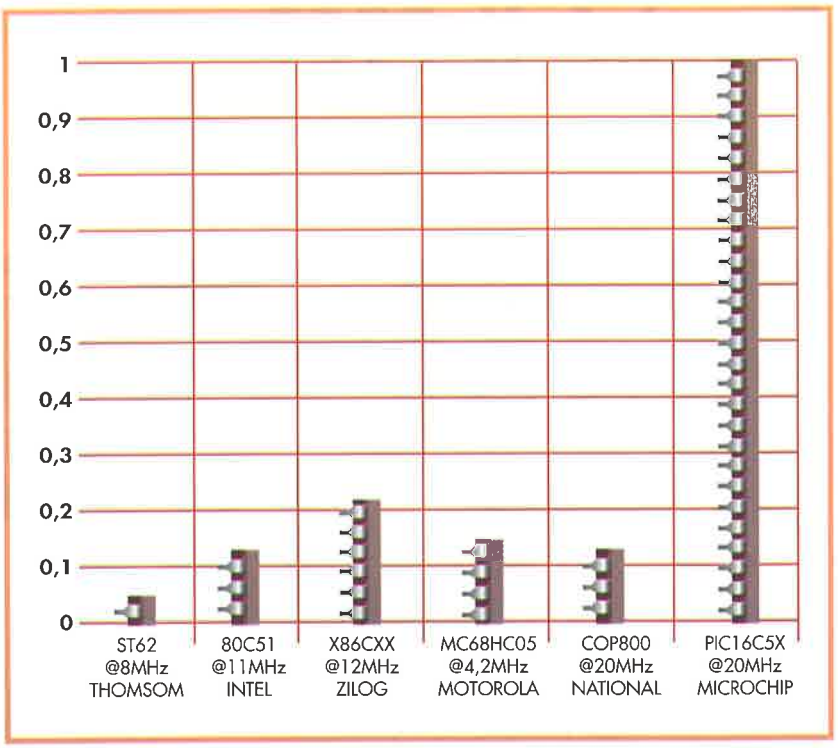
MICROCONTROLLER

La famiglia "PIC" di Microchip

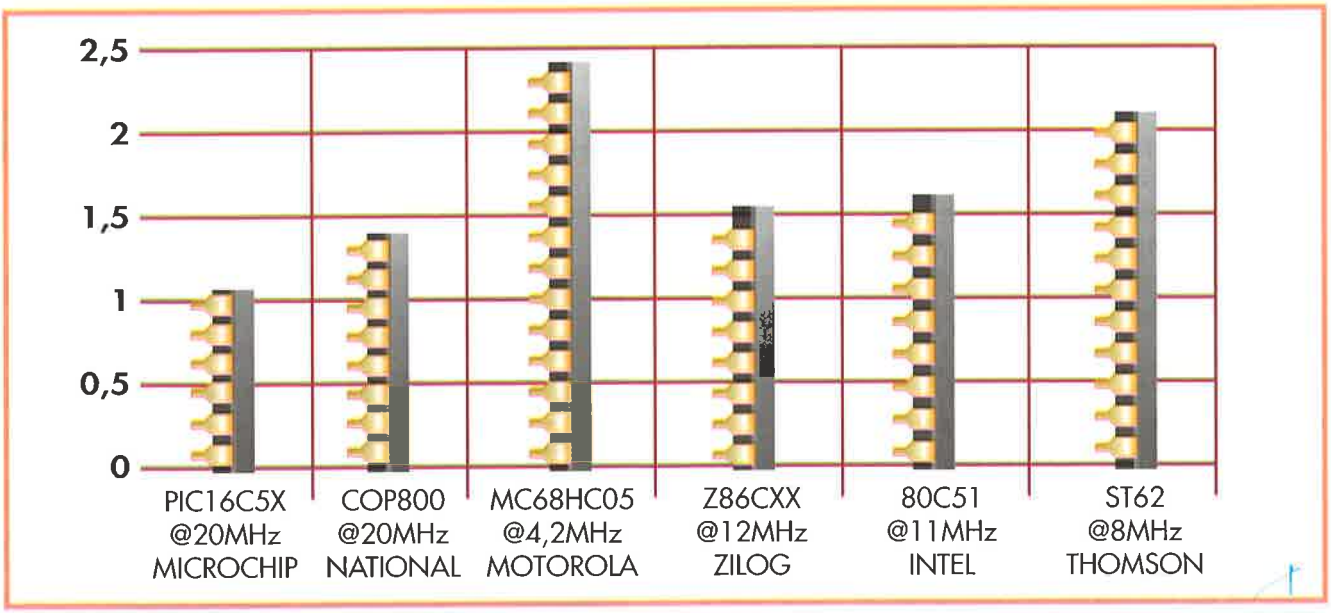
IL MEGLIO DEI PIC

Un dato significativo è costituito dal fatto che il fabbricante MICROCHIP, si è preoccupato più di vendere e far mercato con i suoi microprocessori, che con gli strumenti ausiliari di cui questo componente necessita. In questo modo molti strumenti software si trovano liberamente disponibili attraverso i loro distributori nel mondo, o possono essere trovati nei siti Internet. Le risorse hardware che sono utilizzate nella sua applicazione hanno fatto sì che molti fabbricanti s'interessassero alla loro costruzione, quindi attualmente si hanno modelli di tutti i tipi e prezzi.

Sui mercati esiste un'importantissima e varia offerta di prodotti relativi ai PIC. L'enorme varietà di modelli di PIC, più di un centinaio, permettono di scegliere il migliore a seconda delle applicazioni. Il costo dei microprocessori PIC dipende molto dalla quantità acquistata, però comparativamente è inferiore ai suoi competitori. Sono facili da utilizzare, perché nella gamma media - che è la più popolare - la gamma di istruzioni è di 35 che è quella più semplice ed efficiente. Il basso consumo e l'ampio range di voltaggio che permette la sua alimentazione è molto apprezzato. Come si può vedere dai grafici sia la dimensione dei codici che genera il suo programma, come la velocità di esecuzione lo avvantaggiano notevolmente rispetto ai suoi concorrenti.



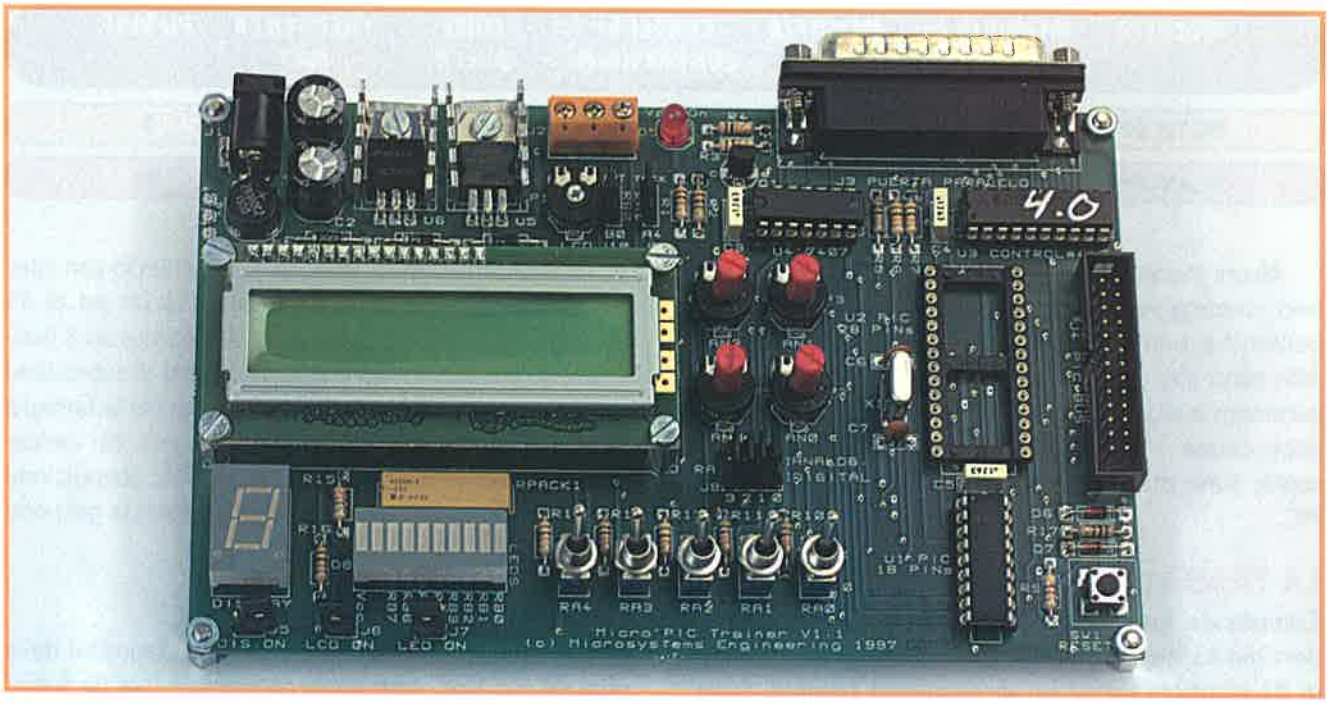
Comparazione della velocità di esecuzione di un insieme di programmi di prova del PIC16C5X a 20 MHz, rispetto ai principali concorrenti.



La dimensione del codice per l'insieme dei programmi di prova per il PIC16C5X è molto più piccola di quella dei suoi principali concorrenti.

MICROCONTROLLER

La famiglia "PIC" di Microchip



Dispositivo per la programmazione del microprocessore PIC, preparato per lavorare con chip da 18 a 28 pin. Permette la connessione a un PC attraverso la porta parallela.

BREVE STORIA DI MICROCHIP

Come succede di solito con le grandi imprese americane, tutto inizia da qualcosa di molto piccolo, che grazie allo sforzo, alla qualità tecnica e alla visione commerciale, assume rapidamente dimensioni enormi.

Il primo successo di questa giovane impresa, la cui sede centrale è a Chandler in Arizona, risale al 1989, quando mise in vendita un PIC16C5X a 8 bit in versione OTP, ad un costo molto basso. Due anni dopo, sviluppò il sistema di emulazione integrale per i suoi PIC PICMASTER sotto Windows. Nel 1992 introdusse sul mercato il primo microcontroller con convertitore AD integrato: si trattava del modello PIC16C71. Attorno al 1994 commercializzò i primi prodotti con memoria FLASH e il PIC16C84 con memoria di programma EEPROM, che fu presto sostituita con la memoria FLASH, tecnicamente più efficiente. Nel 1996 presentò i modelli PIC12C508/9, in contenitore con solo 8 pin, che ebbero un successo senza precedenti. Alla fine degli anni '90 presentò la gamma alta, migliorata con una poderosa architettura che risiede nella famiglia PIC18CXXX, riducendo i prezzi dei microprocessori, fino ad arrivare a 0,50 $ per unità. Nel 1999 aveva venduto un miliardo di microprocessori a 8 bit in tutto il mondo.

UN BREVE RIASSUNTO DELLA NUMEROSA FAMIGLIA DEI PIC

I modelli di PIC superano abbondantemente il centinaio, e si possono raggruppare in cinque categorie, che si presentano secondo le loro principali caratteristiche, nella seguente tabella:

PRODOTTO	FAMIGLIA	ESECUZIONE/ISTRUZIONI	SET DI ISTRUZIONI
PIC12CXXX	FASCIA BASSA (base e media a 8 pin)	1000 ns / 4 MHz; 10 MHz	33/35 istruzioni
PIC16C5X	FASCIA BASE	200 ns / 20 MHz	33 istr. da 12 bit
PIC16CXXX	FASCIA MEDIA	200 ns / 20 MHz	35 istr. da 14 bit
PIC17CXXX	FASCIA ALTA	120 ns / 33 MHz	58 istr. da 16 bit
PIC18CXXX	FASCIA MIGLIORATA	10 MIPS / 40 MHz	77 istr. da 16 bit

MICROCONTROLLER

La famiglia "PIC" di Microchip

MODELLO	MEMORIA DI PROGRAMMA	MEMORIA DATI	CANALI DI CONVERSIONE	LINEE DI I/O	FREQUENZA MAX	PIEDINI
PIC12C508	512 x 12	25 x 8	NESSUNO	6	4 MHz	8
PIC18C858	16 K x 16	1.536 x 8	16	68	40 MHz	80

Alcuni modelli di PIC sono molto semplici ed economici, dispongono delle risorse minime per controllare le periferiche principali, come nel caso del PIC12C508. Altri sono così potenti, come il PIC18C858, che i suoi parametri e le sue prestazioni assomigliano a sistemi di applicazione molto avanzati. Nella tabella riportata sopra, sono messi in comparazione questi due tipi di PIC.

LA FASCIA BASE

Comprende tutti i modelli inquadrati all'interno della denominazione PIC16C5X, i quali sono dotati di un set di 33 istruzioni da 12 bit di lunghezza ognuna. Sono talmente semplici che non sono dotati di interrupt, ed hanno solamente due livelli di stack. Hanno una dimensione molto ridotta, infatti sono commercializzati in contenitori da 18 a 28 pin, sono ideali per applicazioni semplici, di grande consumo, e che funzionano a batteria.

LA FASCIA MEDIA

Sotto questa denominazione si raggruppano i processori che si identificano con la nomenclatura PIC16CXXX e formano la gamma più ampia e diversificata, con ben 72 modelli differenti. Si trovano in commercio con contenitori da 18 a 68 pin e sono dotati di un set di 35 istruzioni da 14 bit ognuna. Il loro stack arriva ad 8 livelli, questo permette sino a 8 annidamenti di subroutine come massimo. In questa fascia è compresa la famiglia dei PIC14C000, specializzati per i progetti dei carica-batterie intelligenti, gruppi di continuità, acquisizione ed elaborazione dei segnali, che richiedono la gestione dell'energia di alimentazione.

LA FASCIA BASSA

In questa gamma sono compresi undici dispositivi della serie PIC12CXXX, tutti quelli con contenitore da 8 pin, che hanno avuto un successo commerciale senza precedenti, data la loro dimensione e il loro prezzo. Alcuni di questi modelli corrispondono alla fascia base, altri a quella media.

LA FASCIA ALTA

Hanno un'architettura aperta, che permette l'ampliamento esterno delle periferiche del microprocessore, tramite il bus dei dati, quello degli indirizzi e dei controlli. Il set di istruzioni arriva sino a 58 e la loro dimensione è di 16 bit. Dispone di un sistema molto potente di gestione vettorializzata degli interrupt, e vengono utilizzati in applicazioni di grande complessità.



Alcuni modelli di PIC della gamma media.

LA GAMMA MIGLIORATA

MICROCHIP ha prodotto nell'anno 2000 la serie PIC18CXXX, caratterizzata da uno straordinario rendimento, dell'ordine dei 10 MIPS ad una frequenza di 40 MHz. Sono dotati di un potentissimo set di 77 istruzioni da 16 bit ognuna, e vengono commercializzati in contenitori da 28 fino a 80 pin. Sono destinati a lavorare con compilatori molto efficienti in linguaggio C, e la loro memoria arriva a capacità di 16 KWord da 16 bit.

Selezione e presentazione del "PIC16F84"

Una volta deciso di utilizzare un microprocessore PIC di MICROCHIP, il passo seguente è scegliere il modello adeguato per il nostro progetto, all'interno dei 116 possibili. Per poter fare questa scelta, dovremo studiare molto attentamente le caratteristiche dell'applicazione che il chip dovrà gestire. Nel lavoro che abbiamo intrapreso vogliamo costruire un robot mobile che sia capace di eseguire diversi compiti.



Un robot come il famoso R2D2 di "Guerre stellari" necessita di un microprocessore che gestisca i suoi motori e sia capace di ricevere ed elaborare le informazioni inviategli dai sensori.

CARATTERISTICHE DELL'HARDWARE

Per governare un robot mobile, il microprocessore dovrà regolare due motori che muoveranno le ruote motrici, e dovrà ricevere ed elaborare le informazioni che gli invieranno alcuni tipi di sensori, come quelli di prossimità, di contatto, di temperatura, a ultrasuoni eccetera. Il corretto trattamento delle informazioni fornite dai sensori, determinerà il movimento delle ruote.

CARATTERISTICHE DEL SOFTWARE

I compiti che desideriamo installare nel nostro animaletto intelligente, saranno molto diversi, anche se non molto complicati. Per questo motivo, non serve che la memoria del programma e dei dati del PIC sia molto grande. Quello che risulta indispensabile è che la memoria del programma possa essere scritta e cancellata quante volte sia necessario, perché il programma dipenderà dal lavoro che dovrà eseguire il robot. D'altra parte il microprocessore scelto servirà per farci imparare la programmazione prima di costruire il robot; questo significa che lo programmeremo e lo proveremo eseguendo molti esercizi didattici e pratici.

LA FORMIDABILE GAMMA MEDIA DEI PIC

Una volta stabilite le caratteristiche più importanti che deve avere il modello del PIC che governerà il nostro

CARATTERISTICHE CHE DEVE AVERE IL PIC

- **Dimensioni e costi ridotti**
- **Deve disporre approssimativamente di una dozzina di linee di I/O per gestire i motori ed i sensori**
- **Memoria di programma con capacità di circa 1000 istruzioni**
- **Memoria di programma e area dei dati che permetta la lettura e la scrittura, ma non volatile**
- **Memoria dei dati di almeno due dozzine di indirizzi, e non volatile**
- **Risorse ausiliarie raccomandate: un temporizzatore, vari interrupt, watch-dog, modo stand-by.**

MICROCONTROLLER

Scelta del PIC

PICmicro® 8-BIT MICROCONTROLLER FAMILY PRODUCTS

Product	Program Memory Bytes	Program Memory OTP/FLASH Words	Program Memory ROM Words	E²PROM Data Memory Bytes	RAM Bytes	Analog 8-Bit ADC Channels	Analog Comparators	Analog Other	Digital I/O Ports	Digital Serial I/O	Digital PWM 10-Bits	Digital Timers	System Max. Speed MHz	System ICSP™	Other Features	Packages
PIC16CXXX — 4-12 Interrupts, 200ns Instruction Execution, 35 Instructions, 4/5 Oscillator Selections, Upwardly Compatible with PIC16C5X/PIC12CXXX (continued)																
PIC16C774*	7168	4096x14	—	—	256	10 (12-bit)	—	PBOD & VREF & PLVD	33	USART/I²C/SPI	2	1-16 bit, 2-8 bit, 1-WDT	20	Yes	25mA source/sink per I/O, PSP 2 CCP MSSP 9Bit	40P 40JW, 44L, 44PQ, 44PT
PIC16F83 PIC16CR83	896 (FLASH) 896	512x14 (FLASH) —	— 512x14	64 64	36 36	—	—	—	13 13	—	—	1-8 bit, 1-WDT 1-8 bit, 1-WDT	10 10	Yes —	20mA source and 25mA sink per I/O 20mA source and 25mA sink per I/O	18P 18SO 18P 18SO
PIC16F84 PIC16CR84	1792 (FLASH) 1792	1024x14 (FLASH) —	— 1024x14	64 64	68 68	—	—	—	13 13	—	—	1-8 bit, 1-WDT 1-8 bit, 1-WDT	10 10	Yes —	**20mA source and 25mA sink per I/O** **20mA source and 25mA sink per I/O**	**18P 18SO** **18P 18SO**
PIC16F84A*	1792 (FLASH)	1024x14 (FLASH)	—	64	68	—	—	—	13	—	—	1-8 bit, 1-WDT	20	Yes	25mA source/sink per I/O	18P 18SO, 20SS
PIC16F873*	7168 (FLASH)	4096 x 14 (FLASH)	—	128	192	5 (10-bit)	—	BOD	22	USART/I²C/SPI	2	1-16 bit, 2-8 bit, 1-WDT	20	Yes	25mA source/sink per I/O, CCP MSSP 9Bit	28SP 28SO
PIC16F874*	7168 (FLASH)	4096 x 14 (FLASH)	—	128	192	8 (10-bit)	—	BOD	33	USART/I²C/SPI	2	1-16 bit, 2-8 bit, 1-WDT	20	Yes	25mA source/sink per I/O, CCP PSP MSSP 9Bit	40P 44L, 44PQ, 44PT
PIC16F876*	14336 (FLASH)	8192 x 14 (FLASH)	—	256	368	5 (10-bit)	—	BOD	22	USART/I²C/SPI	2	1-16 bit, 2-8 bit, 1-WDT	20	Yes	25mA source/sink per I/O, CCP MSSP 9Bit	28SP 28SO
PIC16F877*	14336 (FLASH)	8192 x 14 (FLASH)	—	256	368	8 (10-bit)	—	BOD	33	USART/I²C/SPI	2	1-16 bit, 2-8 bit, 1-WDT	20	Yes	25mA source/sink per I/O, CCP PSP MSSP 9Bit	40P 44L, 44PQ, 44PT
PIC16C923	7168	4096x14	—	—	176	—	—	—	52	I²C/SPI	1	1-16 bit, 2-8 bit, 1-WDT	8	Yes	25mA source/sink per I/O, CCP LCD module, static, 1/2, 1/3, 1/4 multiplex	64SP 64PT
PIC16C924	7168	4096x14	—	—	176	5	—	—	52	I²C/SPI	1	1-16 bit, 2-8 bit, 1-WDT	8	Yes	25mA source/sink per I/O, CCP LCD module, static, 1/2, 1/3, 1/4 multiplex	64SP 68QL, 68L, 64PT

robot, le potremo riassumere nella tabella "Caratteristiche che deve avere il PIC".

Conoscendo le esigenze che deve avere il microprocessore, potremo trovare nel catalogo del fabbricante un modello della gamma media del PIC che soddisfi le nostre esigenze.

Il modello che meglio si adatta alle nostre richieste, è il PIC16F84, capace di lavorare fino a 10 MHz, però dispone di una versione PIC16F84A, capace di funzionare fino a 20 MHz. Sceglieremo il modello normale, perché tendiamo a minimizzare il costo, e la velocità di elaborazione non è critica.

IL FAVOLOSO PIC16F84

È piccolo perché il suo contenitore ha solo 18 piedini. La memoria di programma ha la capacità di 1 K word da 14 bits, e ognuna di queste può contenere un'istruzione.

Inoltre, è di tipo flash, il che permette di registrarla e cancellarla un migliaio di volte in tutta sicurezza. La memoria dei dati contiene una zona RAM volatile da 68 bytes, e un'altra EEPROM non volatile, da 64 bytes, adeguata alle nostre esigenze. Dispone di 13 linee di I/O digitali, un temporizzatore, un watch-dog, può funzionare in modo "riposo" con basso consumo. La velocità

BIGLIETTO DA VISITA DEL PIC16F84

- **Memoria di programma: 1 K x 14, tipo flash**
- **Memoria dei dati RAM: 68 Bytes**
- **Memoria dei dati Eprom: 64 Bytes**
- **Livelli di Stak: 8**
- **Set di istruzioni: 35 da14 Bits**
- **Tempo di esecuzione delle istruzioni normali: 4 X Tosc**
- **Tempo di esecuzione delle istruzioni di salto: 8 X Tosc**
- **Cause che provocano Interrupt: 4**
- **Frequenza massima di lavoro: 10 Mhz**
- **Linee di I/O digitali: 13**
- **Temporizzatori: uno per l'utilizzatore, uno per il watch-dog.**
- **Tensione di alimentazione: da 2 a 6 VDC**
- **Tensione di programmazione: da 12 a 14 VDC**
- **Contenitore: DIP a 18 PIN (oppure SOIC)**

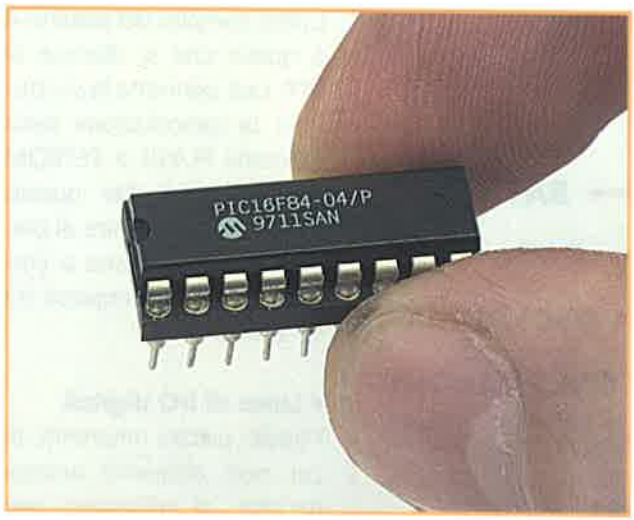


Immagine del PIC16F84 sopra uno zoccolo per proteggere i suoi piedini.

massima di funzionamento è di 10 Mhz, che risulta sovrabbondante, dato che le istruzioni si eseguono a 400 ns, tempo inferiore a quello di cui noi abbiamo bisogno nella nostra applicazione. Pensate che il programma più lungo è di 1.024 istruzioni e a questa frequenza ci sarà un ritardo nell'esecuzione di 0,5 ms, che vengono accettati come tempo massimo di risposta del robot.

Assieme al PIC16F84 esistono altri modelli con caratteristiche molto simili. Così il PIC16C84 è uguale, però con memoria di codice di tipo EEPROM. Il PIC16F83 ha come memoria di programma FLASH solo 512 parole. Ci sono modelli le cui lettere intermedie sono CR, il che significa che la memoria di programma è ROM; questi sono interessanti solo quando il programma è sempre lo stesso, è già stabilizzato, e viene montato in serie di molte migliaia di pezzi.

L'ABITO DA LAVORO: SCHEMA DEI COLLEGAMENTI

Il PIC16F84, che utilizzeremo nel nostro lavoro di apprendimento dei microprocessori e applicazioni alla robotica, è fabbricato con tecnologia CMOS e si trova incapsulato in un contenitore plastico con 18 piedini, come si può osservare nella figura. Secondo le loro funzioni, i piedini si classificano in 4 grandi gruppi:

• Alimentazione

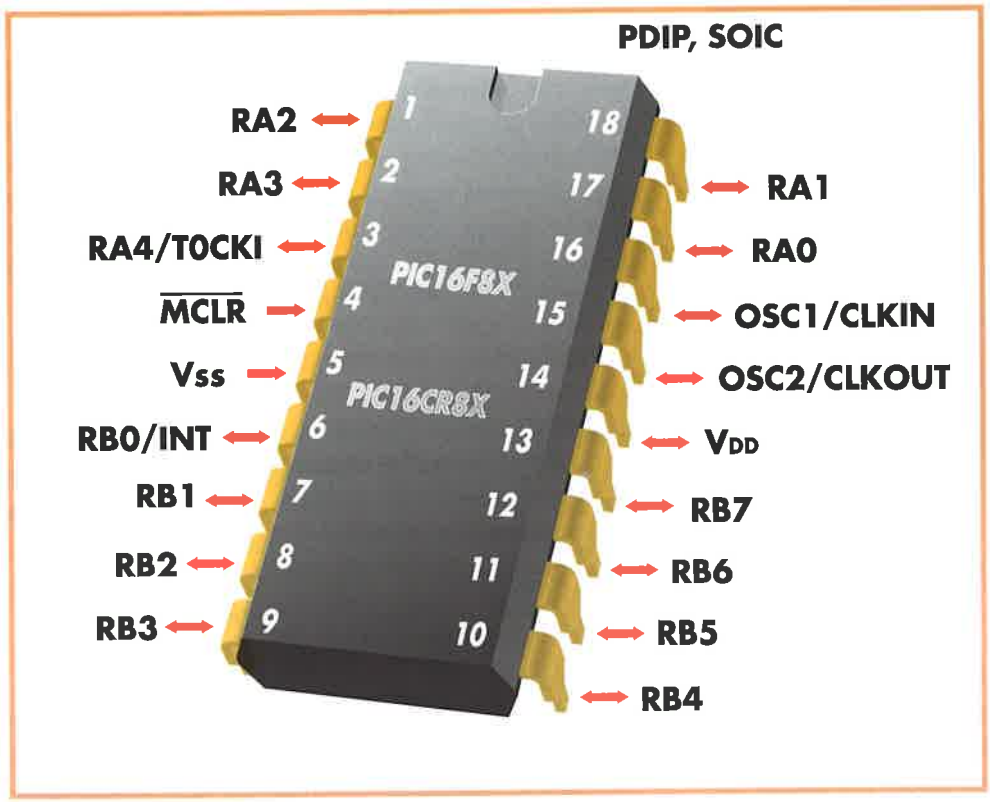
Perché tutti i componenti elettronici del microprocessore possano funzionare, è necessaria una tensione di alimentazione a corrente continua, che deve essere applicata nel seguente modo: il positivo al piedino VDD e la terra a VSS. Il voltaggio di alimentazione è compreso tra 2 e 6 VDC.

		PIC16C84	PIC16F84	PIC16CR84	PIC16F84	PIC16CR84
CLOCK	FREQUENZA MASSIMA	10	10			
MEMORIA	FLASH	-	1k			
	EPROM	1k	-			
	ROM	-	-			
	MEMORIA DEI DATI (bytes)	36	68			
PERIFERICHE	DATI EPROM (bytes)	64	64			
	TEMPORIZZATORI	TMR0+WDT	TMR0+WDT			
CARATTERISTICHE	FONTI DI INTERRUPT	4	4			
	PIEDINI DI I/O	13	13			
	RANGE DI ALIMENTAZIONE (V)	2.0-6.0	2.0-6.0			
	CONTENITORI	18 pin DIP, SOIC	18 pin DIP, SOIC			

Varianti del PIC16F84.

MICROCONTROLLER

Scelta del PIC



Contenitore del PIC16F84, così come di tutte le sue varianti.

• Frequenza di lavoro

La velocità di funzionamento del PIC16F84, è determinata dalla frequenza che si genera internamente, che deve essere stabilizzata dall'esterno mediante un cristallo di quarzo collegato ai piedini OSC1/CLKIN e OSC2/CLKOUT.

• Reset e programmazione

Il piedino 4 denominato MCLR#/VPP, supporta due funzioni che si multiplexano nel tempo. Il PIC lavora normalmente eseguendo il programma che ha registrato nella memoria FLASH, quando si attiva a un livello basso sul piedino 4, si reinizializza o resetta il microprocessore, questo significa che i suoi registri interni assumono un valore determinato e che il contatore di programma punta la direzione 0, da dove incominciare il programma per inizializzarlo. MCLR: Master Clear Reset.

NOTA IMPORTANTE

Il simbolo # che si antepone a certi segnali indica che i medesimi sono attivi a livello basso. Questo simbolo è identico alla riga che si trova sopra al nome del segnale.

L'altro compito del piedino 4 è quello che si riferisce al VPP, cioè permette la scrittura e la cancellazione della memoria FLASH e EEPROM del PIC16F84. Per questo lavoro si deve fornire al piedino 4 una tensione a corrente continua compresa tra 12 e 14 V.

• Linee di I/O digitali

I tredici piedini rimanenti, di cui non abbiamo ancora parlato, si utilizzano per introdurre e ottenere informazioni digitali, da o verso il processore. Questi piedini sono distribuiti su 2 porte. La porta A consta di 5 linee che corrispondono ai piedini denominati RA0-RA4. La porta B è disposta su 8 linee di I/O che corrispondono ai piedini RB0-RB7. Ci sono un paio di piedini che, oltre a realizzare le funzioni di I/O digitali, hanno un altro compito multiplexato, come si spiegherà più avanti.

RACCOMANDAZIONI DI UTILIZZO

Il PIC16F84 è un circuito integrato con doppia fila di piedini. Ha 9 piedini metallici su ogni lato del contenitore che sono abbastanza flessibili.

Questo circuito integrato, a volte, deve essere scritto col programma che supporta il lavoro, per poi essere inserito nella scheda di applicazione ed è abbastanza frequente doverlo inserire sullo zoccolo dello scrittore per poi toglierlo e reinserirlo nuovamente su quello della scheda di applicazione. Questa operazione è molto delicata perché se si forzano e si piegano i piedini ripetute volte, potrebbero staccarsi rendendo irrecuperabile il circuito integrato. Raccomandiamo di collocare il PIC16F84 sopra uno zoccolo normale, perché siano i piedini di quest'ultimo a subire la pressione, o meglio sostituire lo zoccolo dello scrittore o della scheda di applicazione, con uno zoccolo a forza di inserzione nulla; questo possiede una leva che apre i fori dove si introducono i piedini, ciò aiuta enormemente a preservare l'insieme dei piedini.

La frequenza di funzionamento del microprocessore

Il cervello del nostro robot sarà un microprocessore. Il modello scelto è il PIC16F84, che è un piccolo computer la cui missione è eseguire le istruzioni che noi scriveremo nella sua memoria di programma. Attraverso queste, decodificherà i dati che gli invieranno i sensori per riconoscere le caratteristiche dell'ambiente, stabilendo con esse la strategia di movimento dei suoi motori per realizzare efficientemente i compiti che gli abbiamo attribuito.

Il PIC sa solo eseguire istruzioni. Ne conosce 35, non una di più. Nel suo costante lavoro di esecuzione, converte le istruzioni in operazioni elementari. Ognuna di queste operazioni elementari necessita per essere eseguita sempre dello stesso tempo, che è stabilito da un clock molto preciso che indica l'inizio e la fine di ognuna.

Il nostro robot ha incorporato un PIC16F84 che può funzionare ad una frequenza di clock massima di 10 Mhz. Esiste una versione speciale chiamata PIC16F84A, che ammette una frequenza massima di 20 Mhz. Se lavora a 10 Mhz ogni ciclo di clock durerà 100 ns e in questo tempo si realizzerà un'operazione elementare di una istruzione. Pertanto, un'istruzione verrà eseguita in vari cicli di clock.

Tanto maggiore è la frequenza di lavoro del PIC tanto più rapidamente si eseguono le istruzioni e prima si realizzano i programmi, però ci sarà maggiore consumo di energia, cosa molto importante nei sistemi alimentati da batterie.

CICLO DI ISTRUZIONI

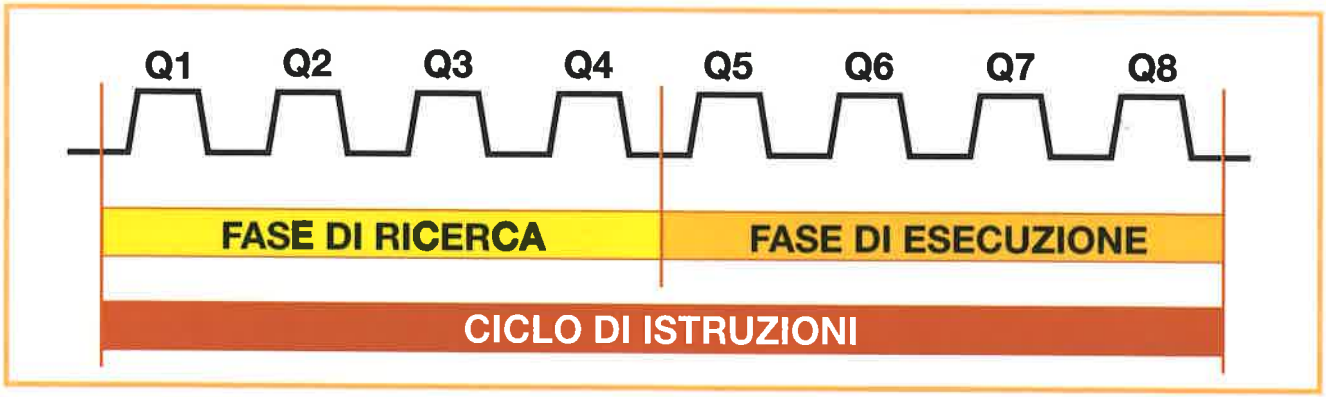
Il ciclo di istruzioni è il tempo impiegato ad eseguire un'istruzione. Non deve essere confuso con il ciclo di clock, che è il tempo con cui si realizza un'operazione elementare.

Un'istruzione si esegue in due fasi:

- **Fase di ricerca**: nella quale si cerca il codice binario dell'istruzione nella memoria del programma.
- **Fase di esecuzione**: nella quale si interpreta il codice, si cercano gli operandi e si eseguono le operazioni conseguenti.

Ognuna delle due fasi di istruzione si realizza in quattro cicli di clock, che si chiamano Q1, Q2, Q3 e Q4. Quindi se ogni istruzione fosse eseguita in sequenza si tarderebbero quattro cicli di clock nella fase di ricerca e altri quattro in quella di esecuzione, in totale otto cicli, come si vede nella figura.

I microprocessori PIC dispongono di un processore molto moderno, nel quale si utilizza la tecnica della segmentazione per eseguire le istruzioni. Questa tecnica si usa anche nella fabbricazione delle macchine, e nelle catene robotizzate delle grandi



Un'istruzione è composta da due fasi: ricerca ed esecuzione.
Ogni fase occupa quattro cicli di clock, e se il PIC avesse un processore normale il ciclo dell'istruzione occuperebbe otto cicli di clock.

MICROCONTROLLER

Velocità di lavoro e inizializzazione

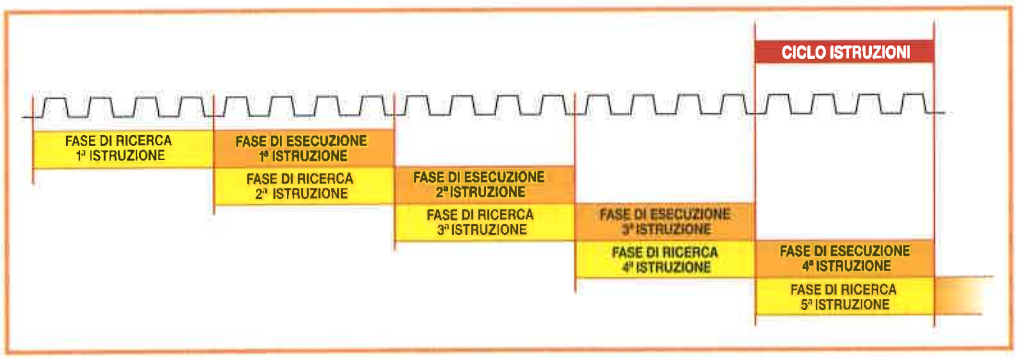


Il processore del PIC è diviso in due parti indipendenti.
Mentre una realizza la fase di ricerca di un'istruzione, l'altra si incarica della fase di esecuzione di quella precedente.

fabbriche. In pratica, invece di iniziare un'automobile, e fermare la produzione fino a che non sia stata terminata la prima, si divide il lavoro in stadi successivi, in modo che i veicoli passino da uno stadio all'altro. Con questa tecnica del lavoro parallelo, si riesce a dividere il tempo impiegato in tante parti quanti sono gli stadi utilizzati.

Il processore del PIC è costruito con stadi differenziati, o stadi che generano una segmentazione di secondo ordine. Uno stadio realizza la fase di ricerca e l'altro la fase di esecuzione. Gli stadi sono indipendenti fra di loro, quindi possono lavorare in parallelo; mentre il primo sta effettuando la fase di ricerca di un'istruzione, il secondo si dedica a realizzare la fase di esecuzione dell'istruzione che è stata precedentemente trovata. Con la segmentazione si ottiene l'esecuzione di ogni istruzione nel tempo di una fase, sarebbe a dire quattro cicli di clock.

Al sovrapporsi dei due stadi d'istruzione, il tempo che impiega un ciclo di istruzioni è di quattro cicli di clock.



Dato che la fase di ricerca e quella di esecuzione si realizza in parallelo, il ciclo di istruzione dura quattro cicli di clock.

ESEMPIO

Un PIC16F84, che funziona ad una velocità di 10 Mhz, esegue un programma di 100 istruzioni, delle quali 30 sono di salto. Calcolare il tempo che impiega ad eseguire il programma.

SOLUZIONE

TEMPO DI CLOCK = 1 s / 10 Mhz = 100 ns

TEMPO DI CICLO ISTRUZIONE NORMALE = 4 x T. CLOCK = 400 ns

TEMPO DI CICLO ISTRUZIONE SALTO = 8 x T. CLOCK = 800 ns

TEMPO DI PROGRAMMA = 70 x 400 + 30 x 800 = 52.000 ns

ESEMPIO

Calcolare il tempo di ritardo nell'esecuzione del programma anteriore in un PIC16F84, che funziona ad una frequenza di 4 Mhz.

SOLUZIONE

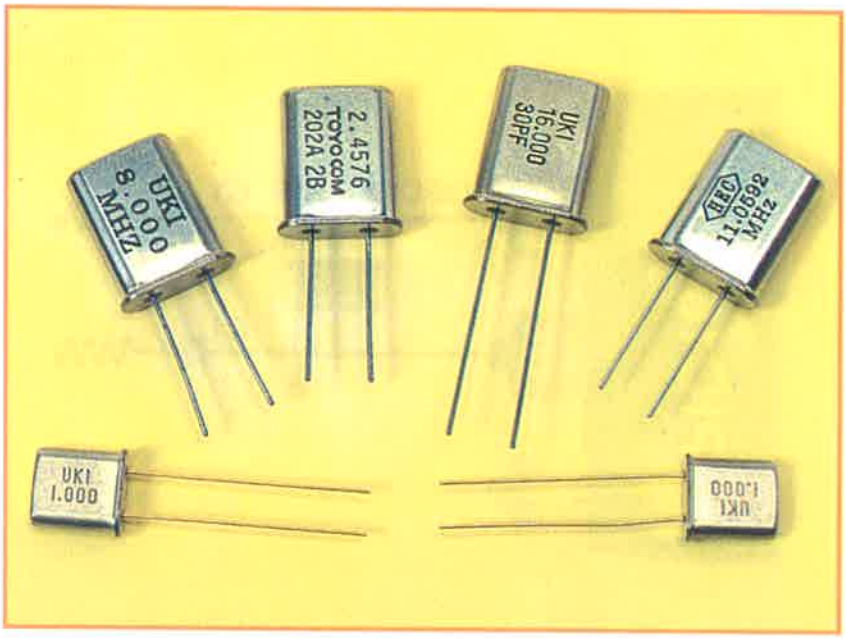
TEMPO DI CLOCK = 1s / 4 Mhz = 250 ns

TEMPO CICLO ISTRUZIONE NORMALE = 4 x T. CLOCK = 1.000 ns = 1 µs

TEMPO DI CICLO ISTRUZIONE DI SALTO = 8 x T. CLOCK = 2.000 ns = 2 µs

TEMPO DI PROGRAMMA = 70 x 1 + 30 x 2 = 130 µs

Questo è valido per tutte le istruzioni del repertorio, eccetto che per quelle di salto, che sono più lente e impiegano il doppio del tempo, cioè otto cicli di clock. La ragione di questo ritardo si deve al fatto che quando arriva al processore un'istruzione di salto non si può cominciare la fase di ricerca della seguente, perché non si sa quale sia. Le istruzioni normali sono ordinate in indirizzi successivi, però nel caso di un'istruzione di salto, fino a quando non si completa la fase di esecuzione non si può sapere quale sarà l'indirizzo seguente.

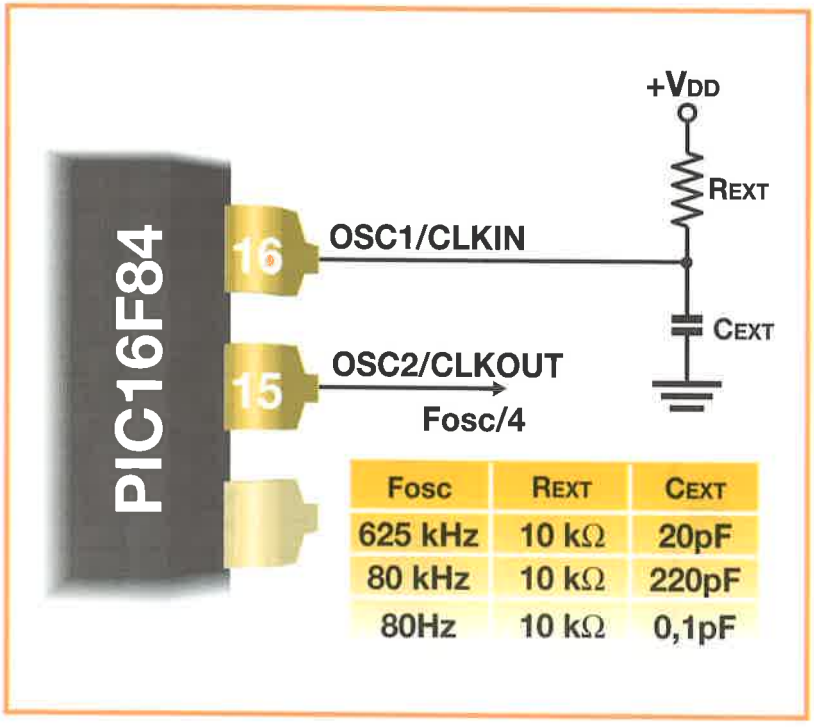


Cristalli al quarzo di uso corrente.

TIPI DI OSCILLATORI

I microprocessori PIC dispongono di un oscillatore interno che determina la frequenza degli impulsi di clock. Senza dubbio, per controllare il valore della frequenza e stabilizzarla, è necessario aggiungere alcuni elementi esterni ai piedini 15 e 16 (OSC1/CLKIN e OSC2/CLKOUT). Secondo il tipo di circuito esterno, gli oscillatori si dividono in quattro tipi:

- Oscillatore RC
- Oscillatore LP
- Oscillatore XT
- Oscillatore HS



Fosc	REXT	CEXT
625 kHz	10 kΩ	20pF
80 kHz	10 kΩ	220pF
80Hz	10 kΩ	0,1pF

I valori della resistenza R esterna e del condensatore C esterno determinano la frequenza nell'oscillatore RC. Questi dati sono forniti dal fabbricante.

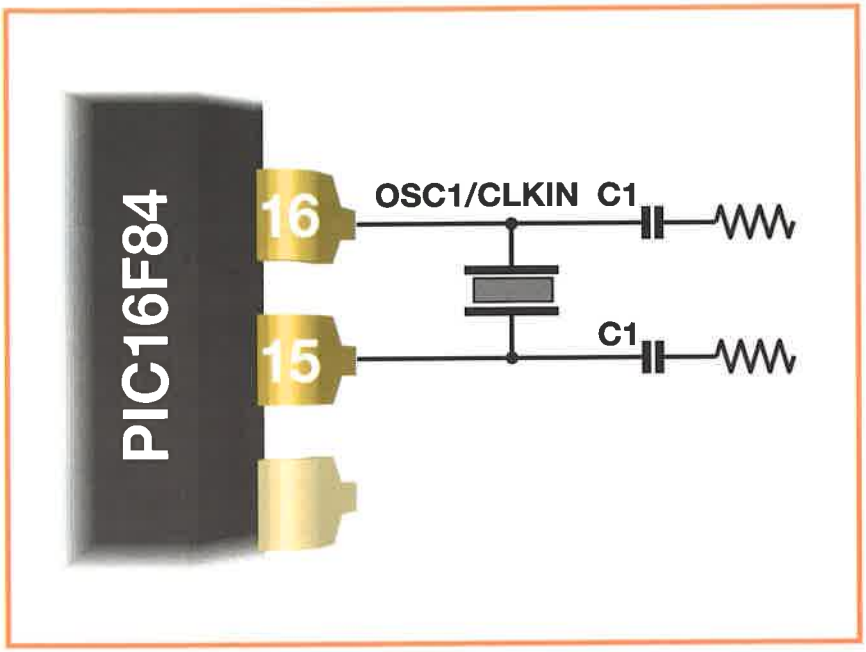
Uno dei dati di inizializzazione necessari al microprocessore riguarda il tipo di oscillatore che si utilizza. Per questo esistono due bits FOSC1 e FOSC2 in una posizione della memoria di programma, chiamata Parola di Configurazione, che dobbiamo scrivere con la combinazione corrispondente.

OSCILLATORE RC

In questo caso si collegano al piedino 16 del PIC16F84 (OSC1/CLKIN) una resistenza e un condensatore, i cui valori sono forniti dal costruttore nella scheda tecnica, e che sono funzione del valore della frequenza. Questo procedimento è molto economico, però la stabilità della frequenza è mediocre e sono adatti solo in sistemi in cui il tempo non sia un fattore critico, e il risparmio sia un obiettivo prioritario. Dal piedino 15 (OSC2/CLKOUT) esce la quarta parte della frequenza di clock (Fosc/4) che è il tempo di ciclo di istruzione normale.

MICROCONTROLLER

Velocità di lavoro e inizializzazione



Per le frequenze comprese tra 4 e 10 Mhz i condensatori C1 e C2 hanno una capacità compresa fra 15 e 33 pF, secondo le informazioni del fabbricante.

OSCILLATORE HS

Questo tipo di oscillatore si usa per lavorare con frequenze superiori a 4Mhz.

Utilizza un cristallo di quarzo e un risonatore ceramico tra i piedini 15 e 16 collegato ai condensatori di disaccoppiamento. Non tutti i modelli di PIC ammettono questo tipo di oscillatore. Data la sua alta frequenza, il consumo è molto elevato.

MESSA IN MARCIA, RESET O REINIZIALIZZAZIONE

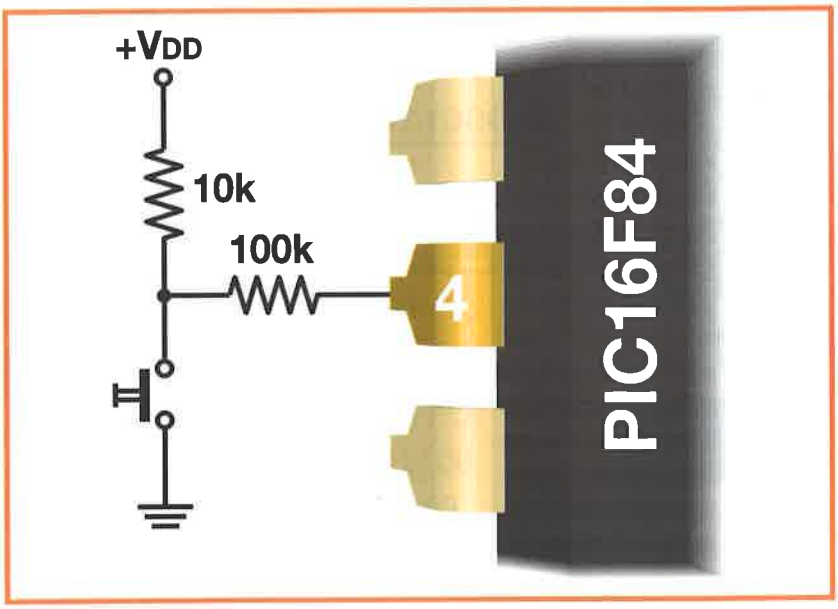
Quando il pin 4 (MCLR#) passa a un livello basso, il PIC16F84 si resetta e ricomincia a eseguire il programma dall'inizio ponendo a uno stato noto tutti i suoi registri interni. Questo processo avviene anche quando si collega per la prima volta la tensione di alimentazione al chip, e si chiama "Power On Reset". Le due azioni principali del processo di reinizializzazione o reset, sono mettere a zero il Program Counter con il quale si inizia l'esecuzione dell'istruzione situata nella prima posizione della memoria di programma, e porre ad uno stato conosciuto tutti i bits dei principali registri di controllo del PIC.

OSCILLATORE LP

Tra i piedini 15 e 16 del microprocessore si monta un cristallo di quarzo della frequenza corrispondente a quella con cui si vuole lavorare. È possibile installare al posto del cristallo di quarzo un risonatore ceramico. A ognuno di questi piedini si collega un condensatore di disaccoppiamento il cui valore è fornito dal fabbricante. Questo tipo di oscillatore è adatto per lavorare con basse frequenze, comprese tra 35 e 250 Khz. Il principale vantaggio dato dalla bassa frequenza è il consumo di energia ridotto che necessita al PIC per funzionare, dato molto importante nei sistemi che si alimentano a batterie.

OSCILLATORE XT

Questo è il tipo di oscillatore che useremo per controllare il PIC del nostro robot, perché è predisposto per lavorare tra 100 Khz e 4 Mhz, la frequenza di lavoro selezionata per il nostro Monty. Monteremo un cristallo di quarzo o un risonatore ceramico tra i piedini 15 e 16, collegato ai rispettivi condensatori di disaccoppiamento, i cui valori sono indicati nella documentazione tecnica e dipendono dalla frequenza selezionata.

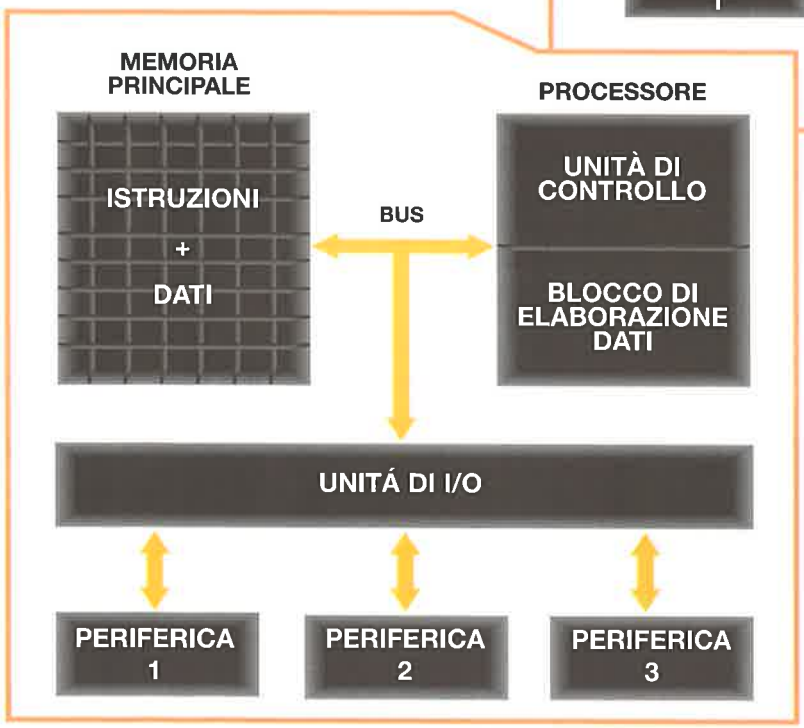


Quando si attiva il pulsante di RESET, si resetta il microprocessore, così come quando si applica la tensione di alimentazione per la prima volta.

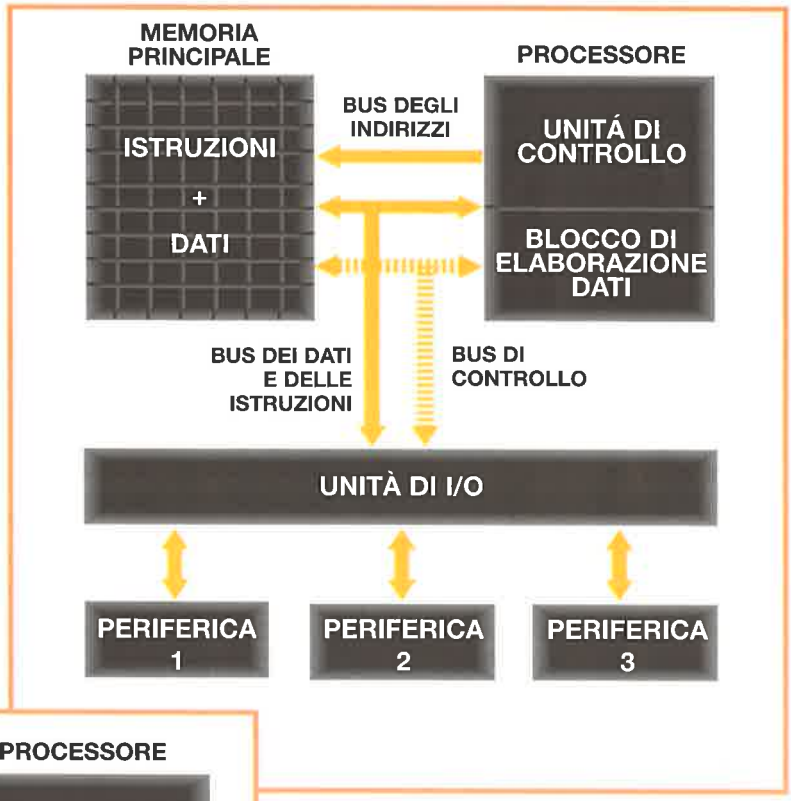
L'intuizione di un sognatore

Nel 1945 Von Neumann stabilì le basi per la costruzione dei computer che oggi hanno invaso il mondo. Disegnò una macchina in cui vi era un blocco - unità di controllo - capace di interpretare un insieme di istruzioni, generava i segnali necessari per la realizzazione delle operazioni che coinvolgevano le varie sezioni del blocco di elaborazione dei dati.

Un grande archivio serviva per contenere le istruzioni del programma insieme ai dati che esso gestiva, infine un ultimo blocco, chiamato unità di ingresso/uscita, permetteva il trasferimento delle informazioni da e per le periferiche esterne. Quest'insieme di elementi è quello che è conosciuto come computer.



Architettura base del computer di Von Neumann.



I bus sono un insieme di linee digitali, trasportano tutte le informazioni che gestisce il computer.

UNA MACCHINA DIGITALE COMUNICA TRAMITE I BUS

Il computer di Von Neumann è una macchina digitale. Tutte le informazioni vengono archiviate in formato binario, ed essendo una macchina elettronica, impiega per rappresentare i due bit di questo sistema, il livello logico alto e il livello logico basso, che generalmente corrispondono alla tensione di alimentazione +VDD e massa o VSS rispettivamente.

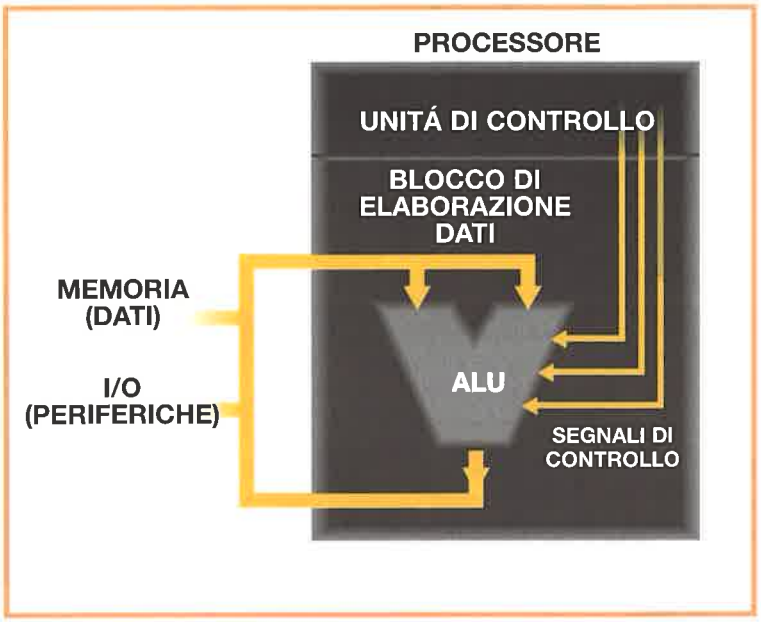
Un'istruzione del computer è formata dall'insieme di bit che la codificano, e la stessa cosa avviene per i dati. La memoria è composta dagli indirizzi necessari per

MICROCONTROLLER

Architettura generale del PIC16F84

immagazzinare le istruzioni e i dati binari. L'unità di controllo riceve le istruzioni che man mano deve eseguire, le decodifica e genera la sequenza di segnali necessari all'esecuzione. Il blocco di elaborazione dati si incarica, sotto la supervisione dell'unità di controllo, di realizzare le operazioni generate dalle istruzioni raccogliendo gli operandi dalla memoria e dagli I/O, inviando i risultati a questi blocchi secondo quanto specificato da ogni istruzione.

Però tutto il movimento delle informazioni all'interno del computer è realizzato da tre insiemi di linee digitali, denominate bus. Il bus dei dati e istruzioni ha il compito di trasferire queste informazioni fra i vari blocchi del computer. Il bus degli indirizzi è caricato dal processore con gli indirizzi delle istruzioni da eseguire o degli operandi da cercare. Infine, il bus di controllo supporta nelle sue linee differenti segnali, come quelli del clock che sincronizzano il lavoro della macchina, quelli di lettura/scrittura che indicano il tipo di accesso alla memoria eccetera.



Nella fase di esecuzione, la ALU del blocco di elaborazione dati esegue le operazioni indicate dal codice dell'istruzione.

ESECUZIONE DI UN'ISTRUZIONE

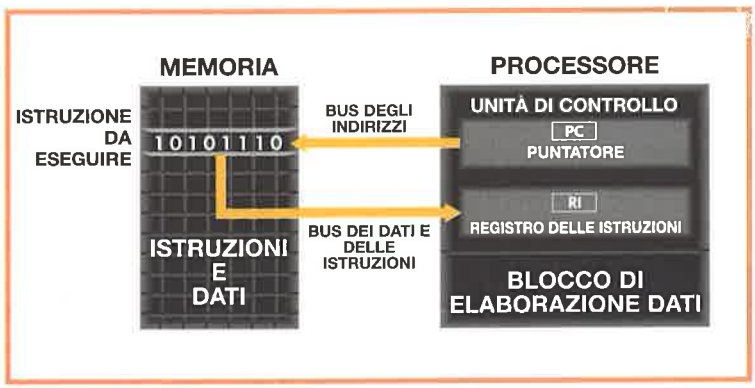
L'esecuzione di un'istruzione inizia in una parte del processore che si chiama Puntatore, o Program Counter (PC). È situato nell'unità di controllo e contiene l'indirizzo dell'istruzione da eseguire.

Di solito il Puntatore dispone di un contatore che aumenta automaticamente di un'unità il proprio valore ad ogni istruzione eseguita.

Questo perché la maggior parte delle istruzioni di un programma, si trovano ordinate in sequenza nella memoria, e all'indirizzo successivo all'istruzione in corso d'esecuzione, si trova quella da eseguire successivamente. Questa regola non vale per le istruzioni di salto, in cui si include l'indirizzo dell'istruzione che deve essere eseguita al passo successivo.

Il contenuto del Puntatore arriva tramite il bus degli indirizzi sino alla memoria, dove seleziona una delle posizioni nella quale risiede il codice binario dell'istruzione in corso. In questo istante attraverso il bus dei controlli arriva l'ordine di lettura della memoria, e il contenuto dell'indirizzo selezionato dal Puntatore viene caricato sul bus dei dati e istruzioni e portato al processore, dove passa su di un registro specifico chiamato registro delle istruzioni.

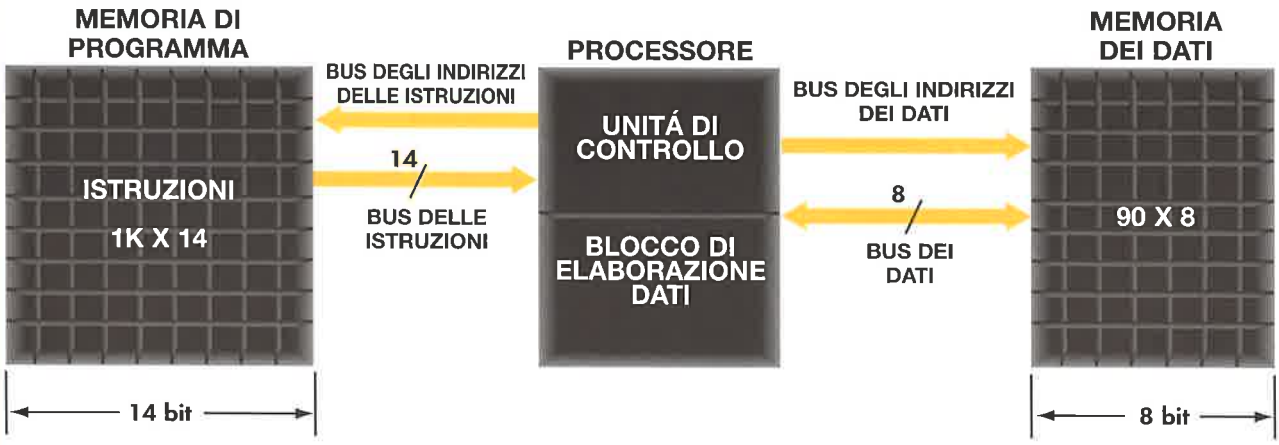
Tutto il processo di localizzazione delle istruzioni e il caricamento sul registro delle istruzioni stesse prende il nome di fase di ricerca e si ripete per tutte le istruzioni.



Fase di ricerca di un'istruzione.

Dopo che il codice binario delle istruzioni è stato caricato sull'apposito registro, inizia la fase di esecuzione. L'unità di controllo interpreta o traduce il codice delle istruzioni e genera una serie di segnali che invia al blocco di elaborazione dati, il quale è formato da una Unità Logico-Aritmetica o ALU, che sarà incaricata di effettuare le operazioni logiche o aritmetiche che comporta l'istruzione.

Sia gli operandi che partecipano all'operazione sia i risultati generati dalla ALU sono posti negli indirizzi di memoria specificati dall'istruzione stessa.



Architettura Harvard nel PIC16F84.

UN INCONVENIENTE NELL'ARCHITETTURA DI VON NEUMANN

Sino agli ultimi anni del secolo XX, la maggior parte dei processori erano costruiti seguendo l'architettura proposta da Von Neumann, come se si trattasse di un dogma di fede.

Le necessità degli utilizzatori richiedevano però sempre maggiori valori di rendimento, sarebbe a dire sempre maggior numero di istruzioni eseguite per unità di tempo (MIPS: Milioni di Istruzioni Per Secondo).

Per migliorare questo parametro furono implementate le tecniche moderne, come la segmentazione, la predizione del salto, e altre ancora più sofisticate; comunque i principi di Von Neumann seguitavano a rimanere inviolabili.

In seguito ci si rese conto di una grave limitazione: l'esistenza di una memoria in cui si scrivevano a caso sia i dati sia gli indirizzi delle istruzioni.

Era impossibile accedere nel medesimo tempo a entrambi i tipi di informazione.

Inoltre il memorizzare richiedeva una memoria dimensionata per entrambi i tipi di informazione: istruzioni e dati, mentre la dimensione o la lunghezza dei dati è molto differente da quella delle istruzioni.

Riassumendo, l'inconveniente dell'architettura di Von Neumann era localizzato nella memoria unica di cui disponeva il processore, che da un lato impediva l'accesso contemporaneo ad istruzioni e dati, e dall'altro la dimensione delle sue celle dovevano adattarsi ad entrambi i tipi di informazioni.

ARCHITETTURA HARVARD

I microprocessori PIC si basano su di una nuova architettura chiamata Harvard. Essa è basata su due memorie indipendenti, una dedicata alle istruzioni e l'altra ai dati. Questa indipendenza permette di realizzare accessi simultanei ad entrambe le memorie, e trovare un'istruzione mentre si sta scrivendo un dato. Inoltre la capacità di ogni memoria può essere adattata alla necessità specifica dell'informazione che si deve gestire, in questo modo la capacità della memoria delle istruzioni può essere molto più grande di quella dei dati, come conviene che sia nella maggior parte delle applicazioni con i microprocessori. Inoltre si può differenziare e adeguare alle specifiche esigenze la dimensione in bit delle celle di ogni memoria.

Nel caso del PIC 16F84 la memoria di programma che è quella dedicata alle istruzioni, ha una capacità di 1024 celle di 14 bit ognuna, è costruita con tecnologia FLASH, che permette di scriverla e cancellarla elettricamente. La memoria dei dati ha una capacità molto più piccola e le sue celle hanno una dimensione di 8 bit, che è la dimensione tipica dei dati. È volatile, di tipo RAM, ed a complemento ha una piccola memoria ausiliaria non volatile di tipo EEPROM, per contenere i dati di tipo permanente, necessari in alcune applicazioni.

L'indipendenza delle due memorie esige dei bus dif-

MICROCONTROLLER

Architettura generale del PIC16F84

ferenziati, per cui la memoria di programma si controlla con il bus degli indirizzi, con linee sufficienti per discriminare 1 K posizioni, e un bus per trasferire istruzioni da 14 bit. Il bus che gestisce gli indirizzi della memoria dei dati ha meno linee, perché questa memoria ha minore capacità. Il bus sul quale circolano i dati ha 8 linee per supportare i bit di ogni posizione.

ALTRE ARCHITETTURE DI COMPUTER

Anche se l'architettura classica di Von Neumann e la Harvard sono molto popolari, esistono molti modi per classificare queste macchine: uno dei più rilevanti è quello proposto da Flynn nel 1966. Per distinguere i processori, Flynn ne distinse il numero di flussi di istruzioni e il numero di flussi di dati che questi processori avevano.

Il processo computazionale consiste nell'esecuzione di una sequenza di istruzioni su un insieme di dati. Il flusso di istruzioni è la sequenza sopra la quale opera un processore, e il flusso di dati comprende la sequenza dei dati in ingresso e la sequenza dei dati parziali e totali che si generano.

Le architetture dei computer si caratterizzano per l'hardware che utilizzano per gestire il flusso dei dati. Flynn propose quattro categorie:

COMPUTER SISD

Corrispondono ai monoprocessori convenzionali, tipo Von Neumann, e dispongono di una unità di controllo e di un blocco di elaborazione dati. Le istruzioni in codice binario ricevute dalla memoria sono decodificate dall'unità di controllo ed eseguite nel blocco di elaborazione dati, con gli operandi che arrivano dalla memoria dei dati, archiviando poi in quest'ultima i risultati.

COMPUTER MISD

Sono composti da 'n' unità di controllo che ricevono ognuna un flusso di istruzioni differente.

Le istruzioni codificate sono inviate a 'n' unità di processo, ognuna delle quali riceve il medesimo flusso dei dati o operandi. Questo significa che differenti sequenze di istruzioni vengono eseguite sopra lo stesso dato. MISD: Multiplo flusso di istruzioni, unico flusso di dati. Non sono stati costruiti modelli commerciali di questi tipi di computer.

COMPUTER SIMD

Hanno un flusso di istruzioni unico e un flusso di dati multiplo. Esiste una sola unità di controllo, che governa differenti processori. Tutti i processori ricevono la stessa istruzione da eseguire, ma dato che ricevono differenti flussi di dati, questa istruzione ottiene differenti risultati. Per poter condividere i dati intermedi e i risultati che producono i processori, è necessario che esista una intercomunicazione fra i microprocessori. Ci sono due modi di ottenerla:

1ª) SIMD CON MEMORIA CONDIVISA

In questo caso tutti i processori utilizzano una memoria comune, e quando uno vuole inviare un dato ad un altro, prima lo scrive in una posizione che l'altro conosce, e successivamente l'altro legge questa posizione. Secondo l'accesso permesso alla memoria, abbiamo quattro categorie:

- **EREW:** Lettura Esclusiva - Scrittura Esclusiva. Non si può leggere né scrivere in una locazione con due processori contemporaneamente.
- **CREW:** Lettura Concomitante – Scrittura Esclusiva. Vari processori possono leggere una locazione di memoria, però solo uno può scriverla.
- **ERCW:** Lettura Esclusiva – Scrittura Concomitante. Solo un processore può leggere una locazione, però in molti la possono scrivere.
- **CRCW:** Lettura Concomitante – Scrittura Concomitante. Vari processori possono leggere e scrivere simultaneamente una locazione.

Negli ultimi due tipi si origina il problema dello scrivere nella stessa locazione da parte di vari processori. Questo si risolve applicando una di queste tre alternative:

1. Scrive la posizione solo il processore con priorità più alta.
2. Se tutti vogliono scrivere la stessa cosa, si può scrivere, in caso contrario, non scrive nessuno.
3. Si scrive la somma di tutti i valori che vogliono essere scritti dai processori.

2ª) SIMD CON RETE DI INTERCONNESSIONE

In questa topologia la memoria è ripartita fra tutti i processori, i quali sono uniti da linee bidirezionali di comunicazione. Ci sono diverse varianti, a seconda della rete di interconnessione che si utilizza.

COMPUTER MIMD

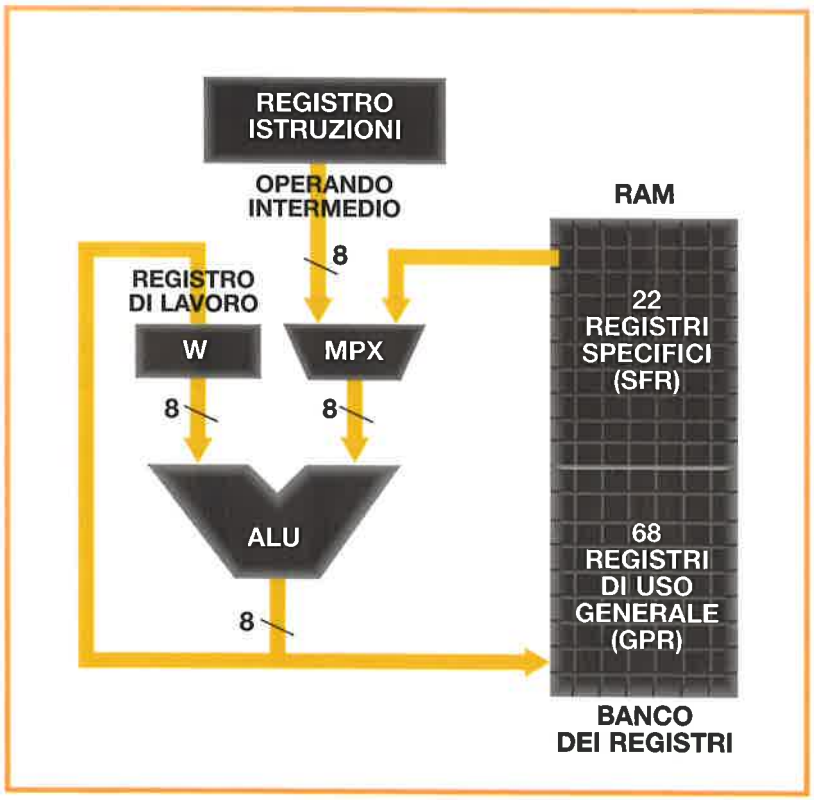
È un gruppo di monoprocessori convenzionali, ognuno con la sua unità di controllo e il suo blocco di elaborazione dati, associato ad una memoria locale. Lavorano in parallelo ognuno con il proprio flusso di istruzioni e il proprio flusso di dati. Inoltre se utilizzano la memoria condivisa, ricevono il nome di Sistemi a Multiprocessore, se utilizzano la rete di interconnessione danno luogo a Sistemi Multicomputers.

Il banco dei registri

I PIC fanno un uso intensivo dei registri per realizzare le operazioni necessarie alle istruzioni che si stanno eseguendo. Questo significa che normalmente la ALU del blocco di elaborazione dati, utilizza come operando il contenuto di un registro, e il risultato lo può archiviare in un altro registro. In realtà la ALU necessita di due operandi fonte per realizzare un'operazione. Uno di questi è sempre scritto nel registro di lavoro W (WORK) e l'altro proviene da un registro, o dall'istruzione stessa quando si tratta di un operando immediato.

Data la semplicità dell'architettura del PIC non sono stati implementati nel processore i banchi dei registri, utilizzando al loro posto le celle della memoria RAM come se si trattasse di registri. La memoria dei dati RAM del PIC16F84 consta di 90 posizioni da 8 bits ciascuna. Queste celle si comportano come se fossero un banco dei registri aventi la dimensione del byte. I 90 registri si dividono in due grandi gruppi:

- *Registri specifici (SFR):* sono 22, e i loro bits hanno il compito di controllare e programmare il lavoro delle diverse risorse del microprocessore. Svolgono il compito di registri di controllo del funzionamento del PIC.
- *Registri di utilizzo generale (GPR)*: sono le 68 celle restanti della RAM, le quali sono a disposizione dell'utilizzatore per archiviare i dati. Nella figura si mostra la ALU che riceve gli operandi tramite il registro di lavoro W, oppure dalla propria istruzione o dal banco dei registri, a seconda di come è selezionato l'ingresso del multiplexer. Il risultato della ALU si può depositare in W o in uno dei registri implementati nella memoria RAM.



Nelle operazioni della ALU intervengono i registri della RAM.

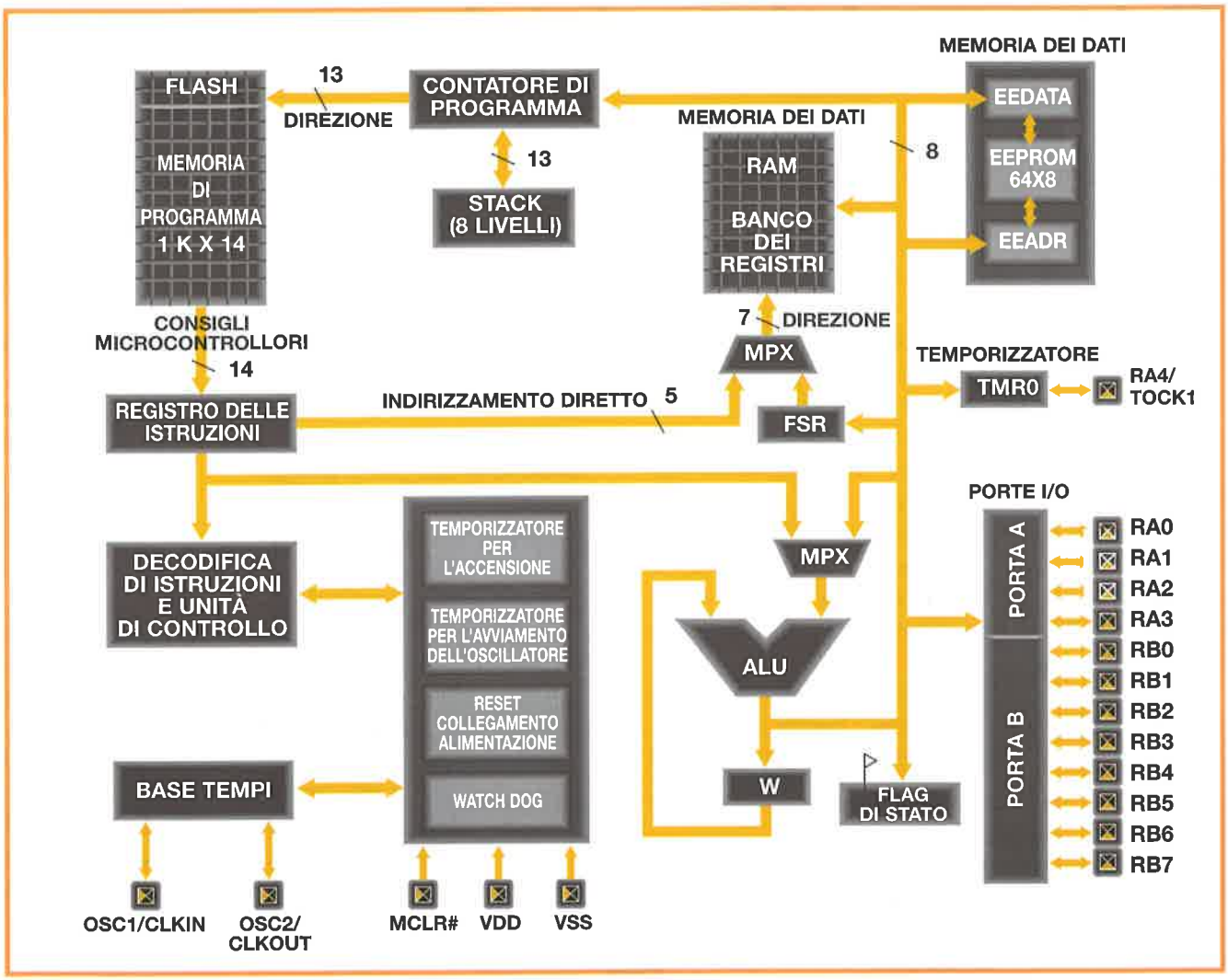
LA VERITÀ SUL PIC16F84

Tutto il contenuto del PIC16F84, sarebbe a dire, tutte le risorse di cui dispone insieme ai collegamenti delle medesime, è mostrato nella figura, che presenta l'architettura dettagliata di questo microprocessore. Si può riassumere tutta la sua potenza descrivendo i principali elementi di cui è costituito e le relative caratteristiche:

- Memoria di programma FLASH da 1 K parole da 14 bits ognuna.
- Memoria di dati RAM con 90 celle da 8 bits, funzionante come banco dei registri diviso in 22 specifici (SFR) e 68 di utilizzo generale (GPR).
- Memoria EEPROM di dati da 64 bytes.
- ALU da 8 bits per realizzare operazioni logiche e aritmetiche. Un operando provie-

MICROCONTROLLER

Il PIC16F84 dentro



Architettura base del PIC16F84, con i collegamenti fra i vari blocchi che lo costituiscono.

ne dal registro di lavoro W, e l'altro dalla propria istruzione o dal banco dei registri.

- Due porte di I/O digitali. La porta A con 5 linee e la B con 8.
- Un temporizzatore da 8 bits chiamato TMR0.
- Un temporizzatore speciale da 8 bits, chiamato Watch Dog (WDT).
- Un puntatore per indirizzare la memoria dei codici legato a uno stack a otto livelli, dove si registrano temporalmente gli indirizzi di ritorno delle chiamate di subroutine e degli interrupt.
- Un sistema per l'indirizzamento diretto e indiretto della memoria RAM.
- Circuiti di Reset e temporizzatore legato al medesimo.
- Possibilità di lavoro in condizioni di basso consumo.

ESECUZIONE DI UNA ISTRUZIONE

Quando un processore esegue un'istruzione lo fa in due fasi diverse. Il PIC16F84 procede in questo modo, e chi vorrà comprendere a fondo il funzionamento di una macchina programmata, deve conoscere queste due tappe fondamentali, pertanto insisteremo su questo tema. Delle due fasi la prima è comune a tutte le istruzioni e realizza sempre la medesima operazione, si chiama fase di ricerca. La seconda è specifica per ogni istruzione e si chiama fase di esecuzione.

FASE DI RICERCA

Il contenuto del puntatore viene caricato nel bus degli indirizzi della memoria di programma e contiene l'indirizzo in cui si trova il codice binario dell'istruzione che bisogna eseguire. Il PC contiene solo indirizzi di istru-

zioni, mai valori relativi a dati. Nel caso del PIC16F84 la memoria di programma ha una capacità di 1 K posizione di 14 bits.

Dato che la longitudine di ogni istruzione è sempre di 14 bits, questa memoria ha la possibilità di contenere programmi con un massimo di 1.024 istruzioni; inoltre è di tipo FLASH il che significa che si può scrivere e cancellare elettricamente con l'aiuto di un programmatore e un PC.

Il bus delle istruzioni sposta i codici binari dell'istruzione puntata dal PC fino all'unità di controllo del processore, dove si trova un registro speciale denominato Registro delle Istruzioni, che viene caricato con questi valori, terminando così la prima fase dell'istruzione. Questa fase di ricerca del codice dell'istruzione da eseguire, è uguale per tutte le istruzioni del repertorio.

FASE DI ESECUZIONE

Con il codice dell'istruzione collocato nel Registro delle Istruzioni, l'unità di controllo procede a decodificarlo, e a interpretare a che istruzione si riferisce.

Terminata l'interpretazione, viene generata una serie di segnali di controllo che si applicano al resto degli elementi, dando via all'esecuzione. La maggior parte delle istruzioni implica il realizzarsi di un'istruzione logica o aritmetica, tipo somme, sottrazioni, AND, OR, rotazione eccetera.

Per effettuare le operazioni esiste all'interno del blocco di elaborazione dati una unità denominata ALU. La ALU del PIC16F84 riceve due operandi, uno arriva dal registro di lavoro accumulatore W e l'altro può provenire dalla propria istruzione, se si tratta di operando immediato, oppure da uno dei registri del banco che è implementato nelle posizioni della memoria RAM dei dati.

A volte bisogna estrarre o introdurre informazioni nel microprocessore, per questo abbiamo due porte, la A e la B, che dispongono di linee collegate con i piedini di uscita dell'integrato, attraverso i quali entrano ed escono i bits.

Come aiuto alle operazioni del blocco elaborazione dati, il PIC16F84 dispone di un temporizzatore, TMR0, che è incaricato di contare il tempo, e un altro, chiamato Watch-Dog, che vigila sulla corretta elaborazione delle istruzioni del programma.

LA MEMORIA DI PROGRAMMA

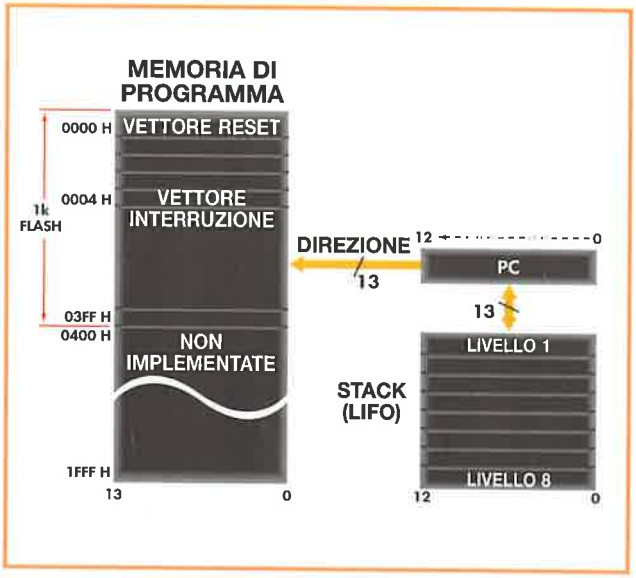
Il PIC16F84 dispone di una memoria di programma di tipo FLASH, con una capacità di 1 K parole da 14 bits ciascuna. La memoria FLASH ammette fino a 1.000 cicli di scrittura e cancellazione con la massima sicurezza.

Per leggere una memoria da 1 K posizioni abbiamo bisogno di indirizzi a 10 bits, dato che $2^{10} = 1.024$. Il program counter, PC, del nostro PIC, ha 13 bits con i quali può indirizzare $2^{13} = 8$ K posizioni diverse. I bits in più di cui dispone il program counter sono dovuti al fatto che l'architettura di tutti i modelli dei PIC della gamma media PIC16CXX è la stessa; però, in ogni modello, il costruttore ha implementato differenti capacità di memoria, per cui l'utilizzatore selezionerà la più conveniente.

Nel PIC16F84 sono state fisicamente implementate solo le prime 1.024 posizioni degli 8 K possibili della mappa di memoria. Il primo indirizzo è lo 0000H, e l'ultimo valido è lo 03FFH.

L'indirizzo 0000H è riservato per il Vettore di Reset, questo significa che quella è la posizione a cui va il puntatore ogni volta che si effettua un reset. Per questo motivo il programma di lavoro del microcontroller deve iniziare obbligatoriamente all'indirizzo 0000H, dato che l'istruzione che contiene è la prima che si esegue dopo il Reset.

Se vogliamo iniziare il programma all'indirizzo 0005H, dobbiamo mettere un'istruzione di salto in questo indirizzo nel Vettore di Reset, sarebbe a dire, go to 5.



Nella memoria di programma del PIC16F84 sono state implementate solo le prime 1.024 posizioni della mappa. Il program counter ha una longitudine di 13 bits e lo stack può contenere fino a 8 indirizzi di ritorno.

L'indirizzo 0004H è assegnato al Vettore di Interrupt, e funziona in forma simile a quello del Vettore di Reset. Quando si verifica un interrupt, il contatore di programma si carica con 0004H e in seguito esegue l'istruzione che si troverà memorizzata in questa posizione.

Di conseguenza la prima istruzione della parte di programma relativo ad un interrupt deve essere collocata in questo indirizzo della memoria di programma.

IL PUNTATORE E I SUOI REGISTRI ASSOCIATI

Il puntatore è un registro da 13 bits, dei quali, nel PIC16F84 se ne utilizzano solo i 10 meno significativi per indirizzare le 1.024 parole della memoria FLASH in cui si trova il programma. I tre bits più significativi di questo registro, non sono utilizzati in questo modello di microcontrollore.

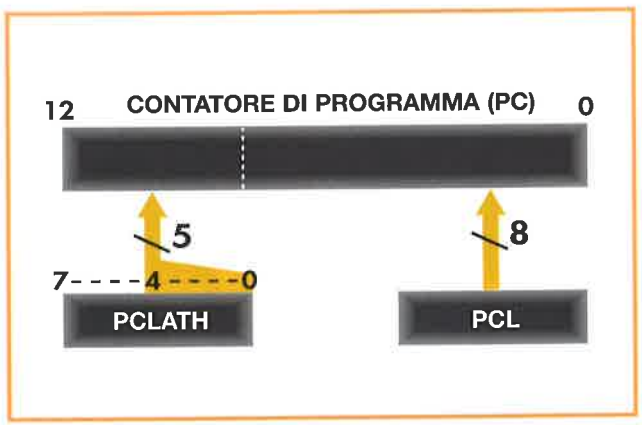
Per gestire e conoscere il contenuto del puntatore, si utilizzano i registri appositi della zona SFR della RAM. Questi registri sono il PCL e il PCLATH, in modo che gli 8 bits meno significativi del PC sono dedicati al PCL e i 5 più significativi al PCLATH, così come è illustrato nella figura.

Il registro PCL è situato all'indirizzo 02H del banco 0, e ripetuto nel banco 1 della mappa dei registri SFR della memoria RAM. Il registro PCLATH si trova anche duplicato all'indirizzo relativo 0AH dei due banchi.

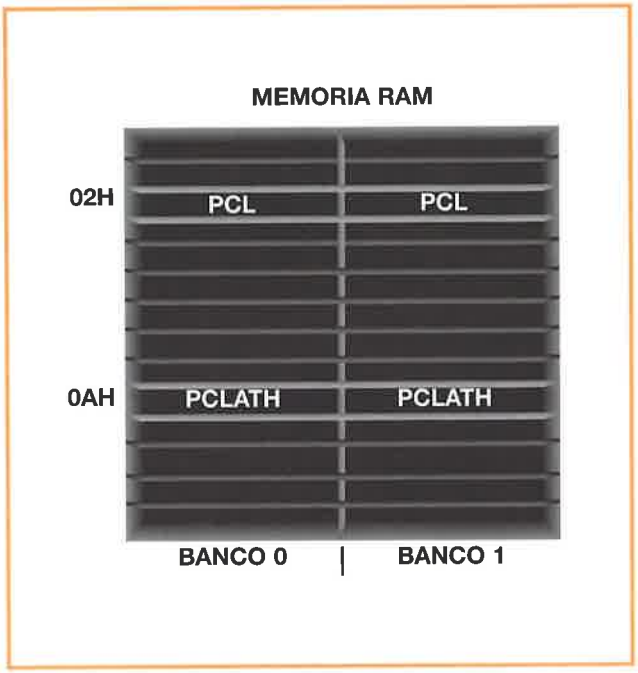
LO STACK

Associato al puntatore esiste una memoria tipo LIFO a 8 posizioni da 13 bits ognuna che si chiama stack.

Dato che è di tipo LIFO si carica e si scarica tramite il suo livello superiore, che è direttamente collegato al puntatore, in modo che durante la fase di carica il valore passa dal puntatore al livello 1, e gli altri valori scendono di un livello. Quando lo stack si scarica, il contenuto del livello 1 passa al PC, e tutti gli altri valori salgono di un livello; questo significa che l'ultimo valore che è stato caricato è il primo che si scarica (LIFO: Last Input First Output).



I bits del puntatore si scrivono nei registri PCL e PCLATH, secondo la distribuzione indicata.



I registri PCL e PCLATH sono duplicati rispettivamente nelle posizioni 02H e 0AH dei due banchi della RAM.

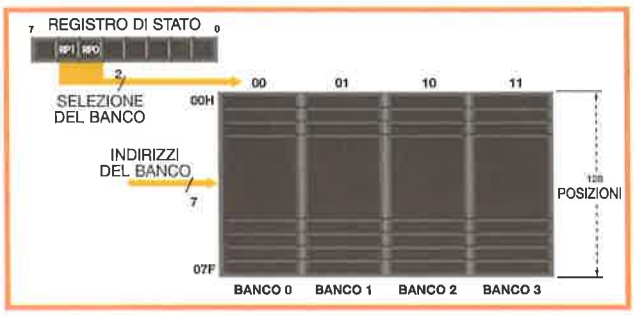
Lo stack serve per scrivere temporalmente gli indirizzi delle istruzioni, e si usa sempre automaticamente se si ha la chiamata di subroutine (CALL) o se avviene un interrupt. In entrambi i casi bisogna memorizzare gli indirizzi di ritorno al programma principale, infatti terminata la subroutine o il programma legato all'interrupt, si ritorna al programma principale.

A volte possono esserci situazioni di annidamenti: dall'interno di una subroutine ne viene richiamata un'altra. Ogni volta che questo succede, bisogna memorizzare nello stack l'indirizzo di ritorno. Visto che non esiste un segnalatore che segnala il riempimento dello stack, sta al programmatore tenere conto degli annidamenti, per evitare che superi gli 8 valori e debordi, perdendo informazioni.

A differenza degli altri processori, lo stack del PIC16F84 non è accessibile da parte dell'utilizzatore e non si può caricare o scaricare. Queste due operazioni si realizzano automaticamente quando si esegue l'operazione CALL o avviene un interrupt.

La memoria dei dati RAM

Con la finalità di semplificare l'indirizzamento della memoria RAM e diminuire i bits degli indirizzi, i progettisti della gamma PIC16CXX hanno organizzato in 4 banchi da 128 posizioni da 8 bits cadauna. L'indirizzamento implica la selezione del banco e la posizione relativa in esso. Per determinare il banco a cui si desidera accedere esistono due bits denominati RP0 e RP1, che si trovano ubicati nel registro specifico della stessa RAM, che si chiama Registro di Stato. Scelto il banco sono necessari altri 7 bits per indicare una delle 128 posizioni del medesimo, come si vede nella figura.

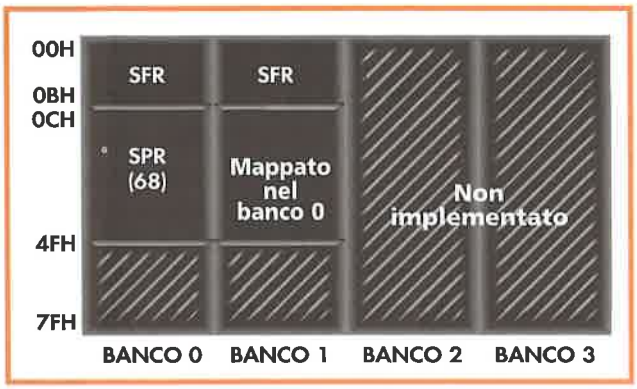


I PIC della gamma media hanno la memoria organizzata in 4 banchi da 128 posizioni di 8 bits ciascuna. Il banco è selezionato con i due bits del Registro di Stato e la posizione con un indirizzo a 7 bits.

ORGANIZZAZIONE DELLA RAM NEL PIC 16F84

Il microcontroller che useremo per il controllo del nostro robot ha una memoria RAM piccola. Non sono stati implementati fisicamente i quattro banchi possibili, che forniscono la capacità massima di 4 x 128 = 512 bytes. Alcuni modelli più potenti della gamma media, dispongono di questa capacità. Il PIC16F84 ha implementato solo i primi due banchi, anche se in modo incompleto. Infatti solo le prime 80 posizioni di questi banchi, cioè dalla posizione 00H alla 4FH, in totale 2 x 80 = 160 bytes, sono stati implementati. Le prime posizioni dei due banchi sono riservate per i registri specifici e le restanti per quelli di utilizzo generale.

Le prime 12 posizioni dei due banchi, i cui indirizzi vanno dalla 00H alla 0BH, sono riservarti ai valori dei registri specifici (SFR), i cui bits controllano il funzionamento delle risorse interne del microcontroller. Funzionano come registi di controllo programmabili. In questa zona SFR abbiamo due posizioni non implementate, che occupano gli indirizzi 07H di ogni banco, pertanto il numero totale dei registri specifici SFR del PIC16F84 è 22.



Il PIC 16F84 ha implementato fisicamente solo le prime 80 posizioni dei primi due banchi.

Le 68 posizioni finali di ogni banco con indirizzi compresi fra 0CH e 4FH, sono usate come registri di utilizzo generale, GPR, destinati a contenere dati e risultati parziali dei programmi. Le 68 posizioni GPR del banco 1 sono mappate nel banco 0. Questo significa che anche facendo riferimento ad una posizione GPR del banco 1, si accede alla posizione correlata del banco 0, i due banchi cioè non funzionano in modo indipendente. Riassumendo, il numero totale dei registri GPR è solo di 68, dato che i 68 del banco 1 funzionano su quelli del banco 0.

REGISTRI SPECIFICI SFR

Il nostro PIC dispone di 22 registri specifici SFR; attraverso i bits di questi registri si programma il funzionamento e l'utilizzo delle risorse del microcontroller. Ci sono 11 registri SFR nel banco 0 e altrettanti nel banco 1. Il registro che occupa la posizione 07H in ogni banco non è implementato. Nella figura si mostra la nomenclatura di questi registri, possiamo notare che alcuni di questi sono ripetuti in entrambi i banchi. Per esempio il Registro di Stato occupa l'indirizzo 03H del banco 0 e il corrispondente indirizzo del banco 1. Questo succede con tutti i registri fondamentali del microcontroller; per poter accedere ad essi indipen-

MICROCONTROLLER

I banchi della RAM

dentemente dal banco che è stato definito con i bits RP0 e RP1.

REGISTRI DI UTILIZZO GENERALE GPR

Integrano un insieme di 68 posizioni di ogni banco, compresi tra gli indirizzi 0CH e 4FH. Però non ci sono 68 x 2 = 136 registri GPR. Solo la metà sono validi, perché quelli del banco 1 sono mappati sopra quelli del banco 0. Come è stato precedentemente spiegato, questo significa che quando si accede a un registro GPR del banco 1, in realtà si realizza l'accesso al corrispondente del banco 0. Sono validi solo i 68 registri GPR del banco 0.

ACCESSO AI REGISTRI

Dato che il PIC16F84 ha operativi solo i primi due banchi della RAM, non sono necessari due bits per specificare il banco selezionato. Dei due bits RP0 e RP1 del Registro di Stato che servono per determinare il banco, al nostro PIC basterà programmare il bit RP0, che è quello che occupa la posizione 5 nel Registro di Stato. È indifferente il valore che ha il bit RP1, che occupa la posizione 6 nel Registro di Stato. Se il bit RP0 = 0, si accede al banco 0, se RP0 = 1, si accede al banco 1.

Quando vogliamo accedere ai registri GPR non dobbiamo tenere conto del valore del bit RP0, dato che, come abbiamo già detto, il banco 1 è mappato sopra il banco 0.

Quindi la gestione dell'accesso ai banchi, consiste nell'impostare il valore del bit RP0, che è il quinto del Registro di Stato, il quale occupa l'indirizzo 03H dei due banchi dell'area SFR. Ci sono due istruzioni nel repertorio del PIC che facilitano enormemente il controllo di RP0. La prima è l'istruzione bsf, che ha il compito di mettere a 1 il bit specificato del registro che si indica. Se troviamo in un programma l'istruzione "bsf STATO,5", la sua esecuzione metterà a 1 il bit 5, RP0, del Registro di Stato, in modo che, a partire da quel momento, l'indirizzamento della RAM si effettuerà dal banco 1 nel programma. La seconda istruzione bcf, mette a 0 il bit specificato del registro che si indica. L'istruzione "bcf STATO,5" pone RP0 = 0 e lascia puntato il banco 0 della RAM.



Distribuzione e nomenclatura dei registri SFR e GPR della memoria RAM del PIC 16F84.

ANCORA SUI REGISTRI SFR

I registri specifici SFR si classificano in due gruppi. Il primo include tutti i registri incaricati di controllare le funzioni vitali del nucleo del microcontroller, tra i quali citiamo STATO, OPTION, INTCON, eccetera. Nel secondo gruppo sono compresi i registri incaricati del controllo del funzionamento delle risorse ausiliarie e delle periferiche, tra le quali citiamo PORTAA, PORTAB, TRISA, TRISB e altre.

Nella figura è rappresentata l'organizzazione dell'area SFR della memoria RAM in modo dettagliato. In essa sono indicati i nomi di ogni registro specifico, l'indirizzo che occupano in ogni banco e il nome di ognuno dei bits. Infine si indica il valore che avrà ognuno di questi bits dopo aver effettuato un Reset normale o uno speciale.

BANCO 0

DIR.	NOME	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	VALORI DOPO IL RESET	VALORI ALTRI RESET
00 h	INDF	Uso del contenuto di FSR per indirizzare la memoria (non è un registro fisico)								---- ----	---- ----
01 h	TMR0	Clock / Contatore in tempo reale a 8 bits								xxxx xxxx	uuuu uuuu
02 h	PLC	Byte meno significativo del PC								0000 0000	0000 0000
03 h	STATO	IRP	IRP1	RP0	TO#	PD#	Z	DC	C	0001 1xxx	000q quuu
04 h	FSR	Indirizzamento indiretto con INDF								xxxx xxxx	uuuu uuuu
05 h	PORTA A	-	-	-	RA4/T0CK	RA3	RA2	RA1	RA0	---x xxxx	---u uuuu
06 h	PORTA B	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0/INT	xxxx xxxx	uuuu uuuu
07 h		Non implementato si legge come "0"								---- ----	---- ----
08 h	EEDATA	Registro dei dati EEPROM								xxxx xxxx	uuuu uuuu
09 h	EEADR	Registro degli indirizzi EEPROM								xxxx xxxx	uuuu uuuu
0A h	PCLATH	-	-	-	Si scrive nei 5 bits di PCH					---0 0000	---0 0000
0B h	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u

BANCO 1

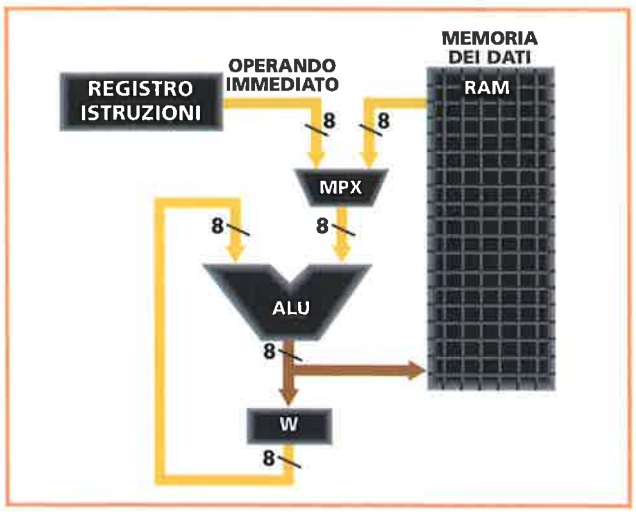
DIR.	NOME	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	VALORI DOPO IL RESET	VALORI ALTRI RESET
80 h	INDF	Uso e contenuto di FSR per indirizzare la memoria (non è un registro fisico)								---- ----	---- ----
81 h	OPTION	RBPU#	INTDG	T0CS	T0SE	PSA	PS2	PS1	PS0	1111 1111	1111 1111
82 h	PLC	Byte meno significativo del PC								0000 0000	0000 0000
83 h	STATO	IRP	IRP1	RP0	TO#	PD#	Z	DC	C	0001 1xxx	000q quuu
84 h	FSR	Indirizzamento diretto con INDF								xxxx xxxx	uuuu uuuu
85 h	TRISA	-	-	-	Configurazione porta A					---1 1111	---1 1111
86 h	TRISB	Configurazione porta B								1111 1111	1111 1111
87 h		Non implementato si legge come "0"								---- ----	---- ----
88 h	EECON1	-	-	-	EEIF	WRERR	WREN	WR	RD	---0 x000	---- q000
89 h	EECON2	Registro di controllo EEPROM (non è un registro fisico)								---- ----	---- ----
0A h	PCLATH	-	-	-	Configurazione porta A					---0 0000	---0 0000
0B h	INTCON	GIE	EEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u

Organizzazione dettagliata dei registri SFR del PIC 16F84. Sono indicati gli indirizzi, il banco, il nome del registro e di ognuno dei suoi bits e i valori che questi avranno dopo un Reset normale o speciale.

Indirizzamento della memoria dei dati RAM

I dati gestiti dalle istruzioni prima dell'esecuzione, vengono detti operandi fonte. Il dato prodotto dall'esecuzione dell'istruzione è il risultato. Nel PIC16F84 i dati, in generale, occupano indirizzi della memoria dei dati RAM, sarebbe a dire, registri. Esistono dati speciali il cui valore non deve essere perso quando si toglie tensione al sistema, per questo vengono scritti in una zona di memoria EEPROM, che studieremo successivamente. Il dato può anche essere contenuto nel codice binario dell'istruzione, nel qual caso si chiama operando immediato.

Normalmente tutte le istruzioni sono realizzate con un'operazione elementare logica o aritmetica nella ALU, la quale riceve uno dei due operandi fonte dal registro di lavoro W, che è stato precedentemente caricato in modo adeguato. L'altro operando, se non è immediato e quindi è già dentro l'istruzione, arriverà da una posizione o da un registro della RAM. Infine, come è indicato graficamente nella figura, il risultato della ALU è caricato in W e in uno dei registri della RAM.



Gli operandi fonte della ALU arrivano o dal registro W o dal multiplexer. Il risultato finisce o in W o in una posizione della RAM.

INDIRIZZAMENTO DELLA RAM

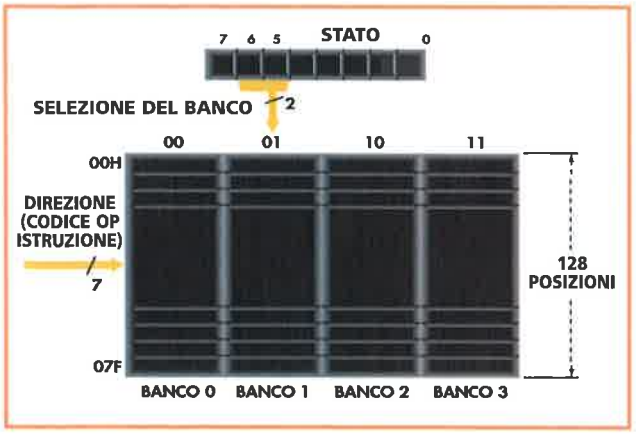
La memoria dei dati RAM del PIC della gamma media è organizzata in quattro banchi da 128 posizioni con 8 bits di dimensione cadauna. Nel caso del PIC16F84 sono state implementate solo le prime 80 posizioni (00H-4FH) dei banchi 0 e 1.

Per indirizzare una posizione della memoria RAM in un PIC bisogna specificare, in primo luogo, il banco in cui si trova, e dopo, l'indirizzo relativo al medesimo. Per selezionare il banco ci sono due bits nel Registro di Stato dell'area SFR della RAM, che si chiamano RP0 e RP1, e occupano le posizioni 5 e 6 del suddetto registro. Per determinare la posizione relativa in un banco, bisogna specificare 7 bits per poter formare 27 = 128 indirizzi possibili.

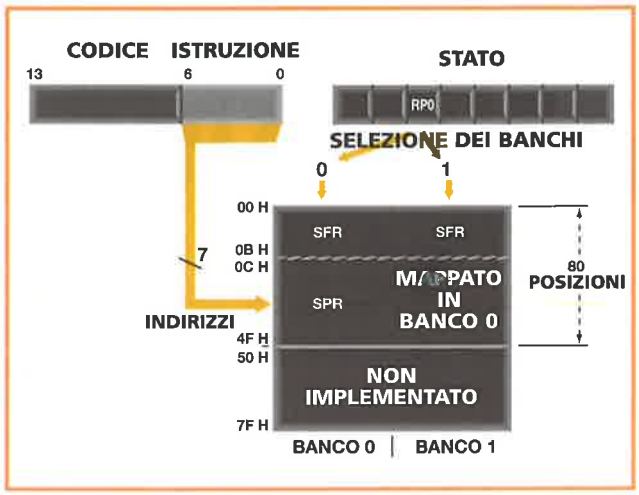
Esistono due modi per indirizzare la RAM: l'indirizzamento diretto e quello indiretto.

INDIRIZZAMENTO DIRETTO

Questo modo per definire la posizione nella memoria RAM, dove si trova il dato ricercato, inizia scrivendo nei due bits RP0 e RP1 del registro di STATO il codice corrispondente al banco. L'indirizzo all'interno del banco è incluso nel codice della propria istruzione, dato che dei



Nell'indirizzamento diretto i bits RP0 e RP1 del Registro di Stato selezionano il banco, e i 7 bits meno significativi dei codici dell'istruzione, la posizione relativa.



Per selezionare il banco nel PIC 16F84 basta scrivere nel Registro RP0 del Registro di Stato.

14 bits che la compongono, i 7 meno significativi sono riservati a questo compito.

Per caricare il codice del banco nei bits RP0 e RP1, esistono, come abbiamo già visto, un paio di istruzioni molto efficaci, capaci di mettere a 1 o a 0 un bit di qualunque registro. Per esempio, se vogliamo accedere al banco 2 dobbiamo caricare RP1 = 1 e RP0 = 0, visto che si usano le istruzioni "bsf STATO,6" e "bcf STATO,5".

Nel PIC16F84, l'indirizzamento diretto è più semplice dato che la memoria RAM è solo implementata nei primi due banchi, il banco 0 e il banco 1. Questo suppone che con un solo bit si sceglie il banco, in realtà è usato solo il bit RP0, essendo indifferente lo stato che ha il bit RP1. Se RP0 = 0, si accede al banco 0 e se RP0 = 1, si accede al banco 1. Rimangono gli altri 7 bits necessari per indirizzare una delle 80 possibili posizioni che ci sono all'interno di ciascun banco.

INDIRIZZAMENTO INDIRETTO

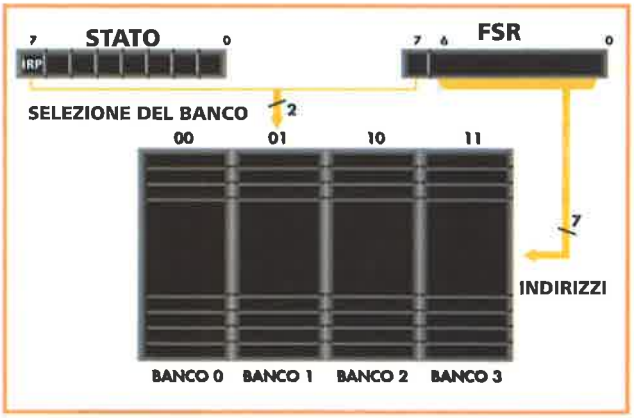
Quando si utilizzano tecniche avanzate di programmazione, o si lavora con linguaggi di alto livello come il C, così interessante nei sistemi con microcontrollori, è molto conveniente disporre di un modo speciale per l'indirizzamento dei dati nella memoria RAM. Si tratta del modo di indirizzamento indiretto, nel quale il codice dell'istruzione non contiene i 7 bits che formano la direzione del banco. Con questo metodo si fa riferimento ad un registro specifico attraverso il quale si ottengono questi 7 bits.

Nei microcontroller PIC le istruzioni che gestiscono l'indirizzamento indiretto utilizzano il registro INDF, che occupa l'indirizzo 00H dei due banchi dell'area SFR. In realtà il registro INDF non è implementato fisicamente, pertanto quando si fa riferimento a questo registro, si accede ad un altro registro specifico dell'area SFR, che si chiama FSR. Nei 7 bits meno significativi del FSR si trova il valore dell'indirizzo relativo al registro cercato nel banco selezionato.

L'istruzione con modo di indirizzamento indiretto, utilizza come registro operando INDF, però il PIC accede al contenuto del registro FSR, che occupa, per duplicato, la posizione 04H dei due banchi della RAM.

Non solo è diversa la maniera di ottenere i 7 bits di indirizzo, ma nel metodo indiretto cambia anche il modo di specificare il banco. In quello diretto si usavano i bits RP0 e RP1 per scegliere il banco, nel modo indiretto si usa il bit 7 - il più significativo - del registro di STATO, che si chiama IRP, e il bit 7, il più significativo, del registro FSR (ricordiamo che i 7 bits meno significativi contengono gli indirizzi relativi). Dato che nel PIC16F84 abbiamo solo disponibili i banchi 0 e 1, il bit IRP del registro di stato varrà sempre 0.

Se, per esempio, vorremo selezionare in forma indiretta la direzione 12H del banco 1 in un PIC16F84,



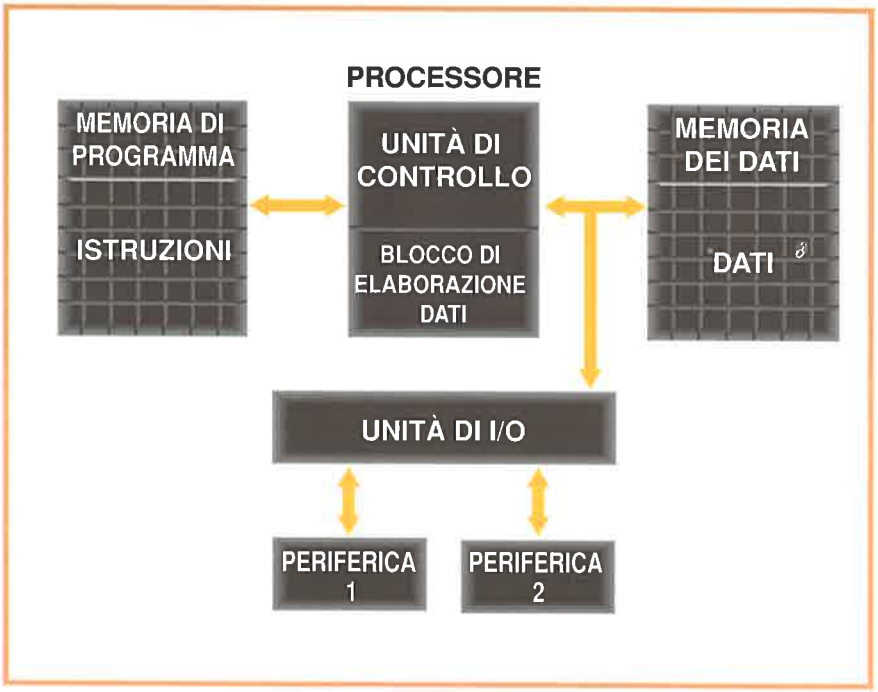
Nell' indirizzamento indiretto il banco si sceglie con il bit più significativo del Registro di Stato e quello più significativo di FSR. L'indirizzo relativo è contenuto nei 7 bits meno significativi di FSR.

cominceremo caricando in FSR il valore 12H (0x12), per i 7 bits di minor peso, e metteremo a 1 il bit 7 di maggior peso del registro per scegliere il banco 1.

```
movlw   0x12    ; Si carica W con 12 H
bsf     W,7     ; Si mette a 1 il bit 7 di W
movwf   FSR     ; Il valore di W si carica in FSR
```

Le porte di ingresso/uscita

Il processore che contiene il PIC16F84 indirizza e raccoglie i codici delle istruzioni residenti nella memoria di programma, le decodifica, cerca gli operandi, le esegue e produce dei risultati che memorizza. Questo è il tipico lavoro necessario per processare un'istruzione, però a che scopo? L'obiettivo finale è quello di controllare una serie di dispositivi esterni che configurano il sistema che dobbiamo governare. Questi dispositivi, che chiameremo periferiche, certe volte hanno il compito di introdurre dati nel microprocessore e quindi vengono chiamati di "ingresso". Altre volte ricevono i risultati dell'elaborazione delle istruzioni, sono dati che vengono conferiti alle periferiche per regolarne il funzionamento, questo tipo di periferiche viene chiamato di "uscita".

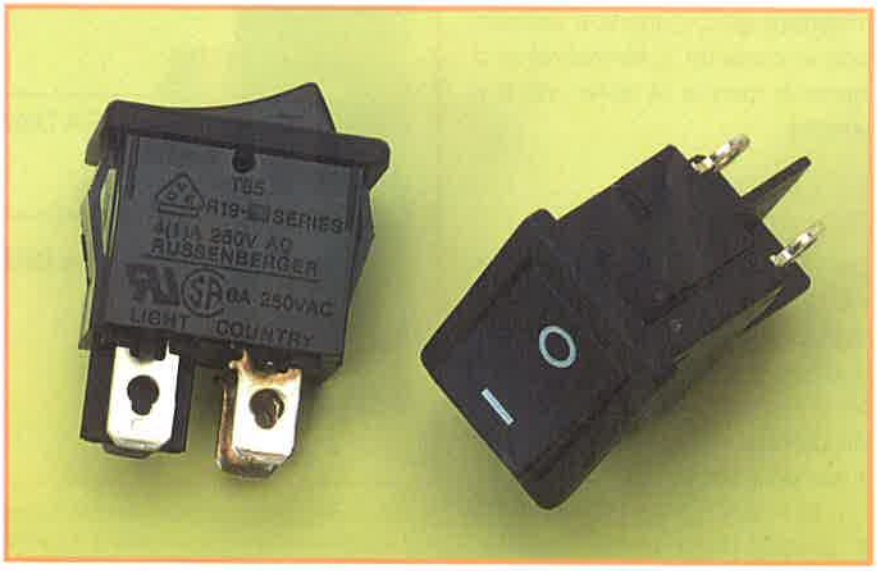


Le periferiche sono dispositivi esterni che introducono e ricevono informazioni da o per il mondo esterno.

TIPI DI PERIFERICHE

Le periferiche si classificano in due grandi gruppi: digitali e analogiche. Le prime ricevono e/o inviano informazioni digitali attraverso un insieme di linee digitali il cui numero è uguale a quello dei bits contenuti in ogni parola di informazione trasferita. Le periferiche analogiche inviano e/o ricevono informazioni analogiche, ossia segnali che variano il loro valore da un minimo ad un massimo, passando per infiniti valori intermedi.

All'interno delle periferiche digitali, le più semplici sono quelle che generano o si governano con una sola linea digitale, di modo che, se



La periferica di ingresso digitale più comune è l'interruttore, che può assumere diverse forme.

MICROCONTROLLER

Porte per il trasferimento d'informazioni con il mondo esterno



I diodi LED sono le periferiche digitali di uscita più semplici e utilizzate.

sono periferiche di ingresso introducono il valore di un bit su di una linea, mentre se si tratta di periferiche di uscita il loro funzionamento necessita solo di un bit. La periferica di ingresso digitale più comune e più utilizzata è l'interruttore, che assume diverse forme come pulsante, commutatore ecc. La periferica di uscita più semplice, ma non per questo la meno usata, è il diodo LED. Il primo tipo di questi interruttori, a seconda dello stato, introduce un 1 o uno 0. Il secondo tipo se riceve un 1, normalmente si illumina, mentre se riceve uno 0 si spegne.

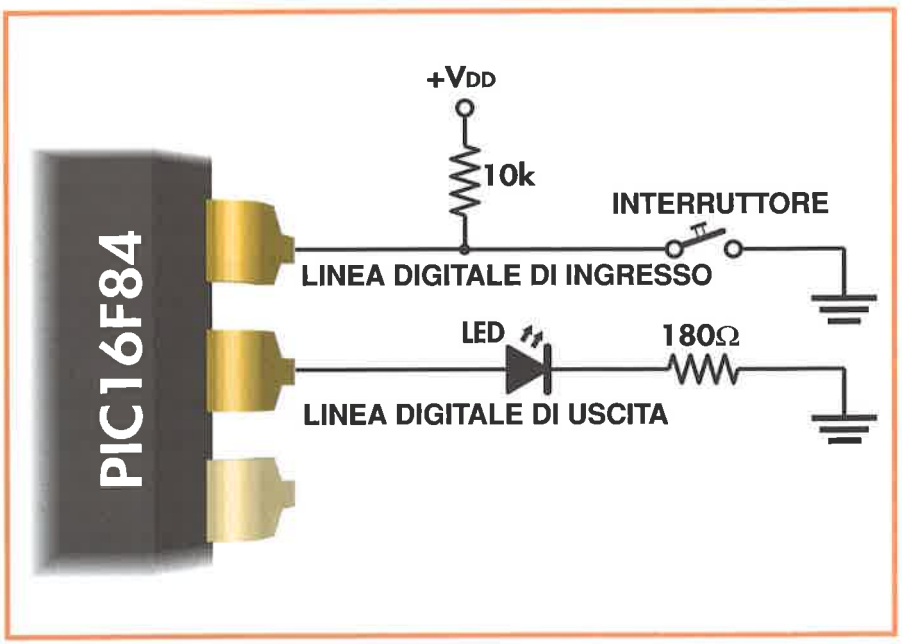
LE LINEE I/O DEL PIC16F84

Dei 18 piedini che dispone il nostro PIC, 13 sono destinati a linee di I/O digitali. Ognuna di esse può introdurre al processore il valore di un bit o essere incaricato di prendere dal processore il valore di un bit per il mondo esterno.

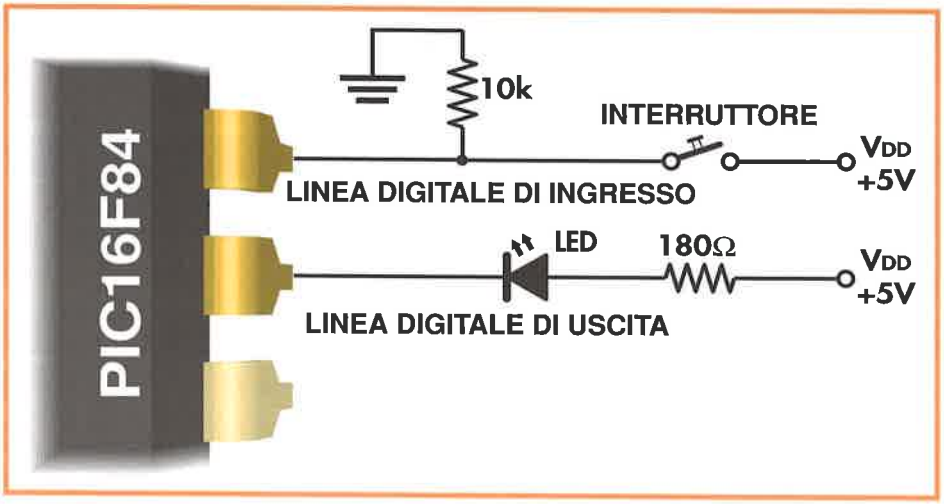
Se una di queste 13 linee di I/O è programmata come ingresso, sarà possibile collegare ad essa un dispositivo esterno che introduca il valore di un bit, in modo che quando il programma che sta eseguendo il microprocessore legge questa linea, riconosce se c'è un 1 o uno 0. Programmando una di queste linee di I/O come uscita, potremo collegare ad essa una periferica di uscita, come un LED e quando il programma va a scrivere il valore di un bit sopra questa linea, se è un 1 il LED si illuminerà, e se è uno 0 il LED si spegnerà. Il collegamento di una periferica d'ingresso, in questo caso un interruttore, e di una periferica di uscita, un LED, è illustrato nella figura.

Può capitare, in alcuni sistemi, che i livelli logici delle periferiche siano al contrario rispetto a quelli indicati nella figura; per esempio, se l'interruttore è aperto, viene introdotto uno 0 nella linea di I/O, e se è chiuso un 1.

Inoltre, può succedere che si desideri accendere il LED quando la linea di uscita digitale sia a 0 e spegnerla quando sia a 1. Nella figura si vedono le modifiche al



Il processore conosce lo stato dell'interruttore, se è aperto o chiuso, questo perché leggendo la linea di ingresso esso riceve un 1 se è aperto o uno 0 se è chiuso. Il LED della linea di uscita è controllato dal processore perché lo accende scrivendo 1 e lo spegne scrivendo 0.

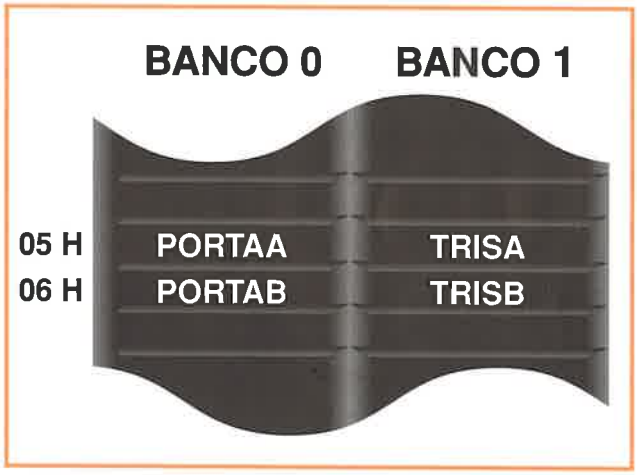


Se l'interruttore è aperto introduce un bit 0.
Per accendere il LED dobbiamo scrivere un bit 0 sulla linea digitale a cui è collegato.

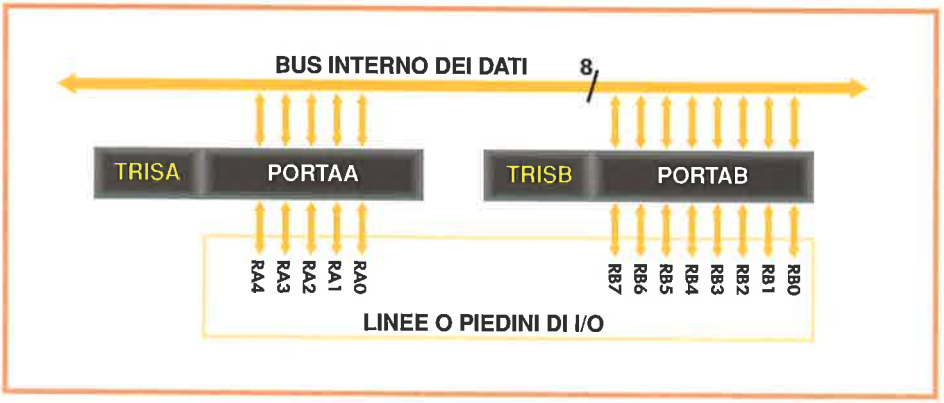
circuito di adattamento delle due periferiche per operare con livelli logici opposti.

LE PORTE DEL PIC16F84

Normalmente si dice che il PIC16F84 è povero perché ha solo 13 linee di I/O, scarse risorse e poca capacità di memoria. Le linee di I/O si raggruppano in due insiemi che sono chiamate "porte". La porta A raggruppa 5 linee di I/O, mentre la porta B raggruppa le 8 restanti. Le porte di I/O operano come registri che possono essere letti e scritti dall'esterno e dall'interno del processore. Questi registri si chiamano PORTA A e PORTAB. In realtà lo scambio di informazioni lo realizza il processore, scrivendo sopra una porta mentre la periferica esterna la legge, o al contrario, è la periferica esterna che colloca l'informazione nella porta e il processore la legge.
Ogni registro nel quale si collocano le informazioni da trasferire ha associato un altro registro che serve per configurare ogni linea come ingresso o uscita. La porta A ha associato il registro TRISA e la porta B il registro TRISB. Il procedimento per configurare le linee è molto semplice. Se desideriamo che la linea 2 del registro PORTAA (RA2) funzioni come una linea di ingresso dobbiamo scrivere nel bit 2 del registro TRISA un 1. Se al bit 3 di TRISA si memorizza uno 0, la linea RA3 del



Indirizzi delle porte
e dei relativi registri di configurazione dell'area SFR della RAM.



Ogni registro di scambio d'informazione, PORTA A, e PORTA B, ha associato un registro di configurazione TRIS A e TRIS B, rispettivamente.

registro PORTAA rimane configurato come linea di uscita. I registri PORTAA e PORTAB, insieme ai loro associati TRISA e TRISB, occupano gli indirizzi 5 e 6 di entrambi i banchi della memoria, come si vede nella figura.

UTILIZZI DELLE PERIFERICHE

Prima di leggere o scrivere le informazioni in una periferica, dobbiamo configurare adegua-

MICROCONTROLLER

Porte per il trasferimento d'informazioni con il mondo esterno

ESEMPIO

Si desidera configurare come ingressi le linee RA4, RA3, RA2, e RA1 della PORTAA, mentre la linea RA0 deve funzionare come uscita. Realizzare il programma corrispondente.

SOLUZIONE

Si deve caricare nel registro di configurazione TRISA il valore binario "xxx11110". I tre bits di maggior peso di TRISA non hanno significato né funzione, dato che la PORTA A dispone solo di 5 linee. Scriveremo in questi registri tre zeri, in modo che il valore da caricare sarà "00011110", che corrisponde al valore 1EH. Il programma richiesto dovrà scrivere 1EH nell'indirizzo 5 del banco 1, dove è indicato TRISA.
NOTA: in assembler i valori esadecimali si scrivono direttamente preceduti da 0x.

```
TRISA ..... EQU ..... 0x05 ..... ;L'etichetta TRISA corrisponde all'indirizzo 5H
STATO ..... EQU ..... 0x03 ..... ;L'etichetta STATO corrisponde all'indirizzo 3H
Bsf ..... STATO,5 ..... ;Bit 5 di STATO = 1 (RP0 = 1). Si accede al banco 1
Movlw ..... 0x1e ..... ;Si carica W con 1EH
Movwf ..... TRISA ..... ;Il valore W si carica in TRISA
```

Al momento dell'accensione del PIC, o in caso di reset, tutti i bits del registro di configurazione TRIS vengono posti a 1, il che significa che tutte le linee delle porte rimangono inizialmente configurate come ingressi.

ESEMPIO

Preparare un programma per configurare tutte le linee della porta A come ingressi e le linee della porta B come uscite.

SOLUZIONE

Dovremo scrivere nei 5 bits meno significativi di TRISA un 1 e in tutti i bits di TRISB uno 0.

```
TRIS A ..... EQU ..... 0x05
TRIS B ..... EQU ..... 0x06
STATO ..... EQU ..... 0x03
Bsf ..... STATO,5 ..... ; RPO = 1, per accedere al banco 1
Movlw ..... 0x1f ..... ; W si carica con 1FH
Movwf ..... TRISA ..... ; Il valore di W si scrive in TRISA
Movlw ..... 0x00 ..... ; W si carica con 0
Movwf ..... TRISB ..... ; W si carica in TRISB
```

tamente le linee mediante le quali si trasferiscono le informazioni digitali, definendo gli ingressi e le uscite. Realizzata la configurazione delle linee I/O scrivendo nei registri TRIS, per portare al processore l'informazione che arriva da una periferica d'ingresso collegata al pin di una porta, dobbiamo solo leggere il contenuto del registro di questa porta, ossia, PORTAA o PORTAB.

Al contrario, se si desidera portare dal processore il valore dell'informazione digitale ad una periferica di uscita collegata al pin di una porta, configurata come uscita, basta scrivere il suddetto registro con il valore da trasferire. Per leggere il contenuto di una porta d'ingresso che contiene il valore che la periferica corrispondente vi ha depositato, si utilizza l'istruzione "movf PORTAA,W", con la quale il dato presente nella porta passa al registro W.

Per spostare un dato verso una porta d'uscita, prima si carica quest'ultimo nel registro W e dopo si esegue l'istruzione "movwf PORTAA". Il valore contenuto in W viene spostato sulla PORTAA e le linee di uscita corrispondenti portano verso il mondo esterno i bits corrispondenti.

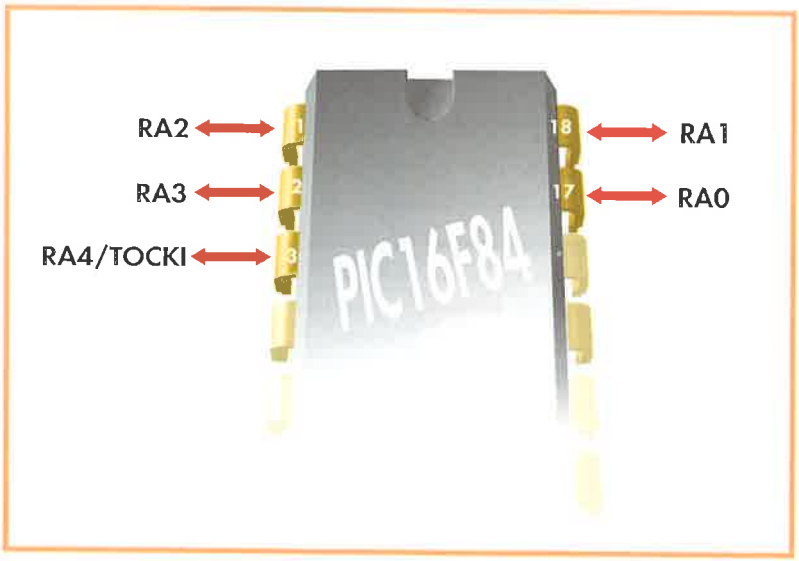
Caratteristiche delle porte A e B

LA PORTA A

Questa porta ha solo 5 linee digitali che si collegano con l'esterno mediante cinque piedini denominati RA0, RA1, RA2, RA3 e RA4. Essi occupano le posizioni superiori delle due file di cui è composto il contenitore del PIC16F84, come mostrato nella figura.

Le informazioni che entrano ed escono da questi piedini passano temporaneamente nel registro PORTAA, che occupa l'indirizzo 5 del banco 0 dell'area dei registri specifici SFR della RAM. Inoltre per configurare come ingresso o uscita ognuna delle linee di questa porta disponiamo di un registro chiamato TRISA, che occupa la posizione 5 del banco 1. Come ricorderete, se in un bit del registro TRISA si scrive un 1, la linea della porta A corrispondente funzionerà come ingresso, e se si scrive uno 0 funzionerà come uscita. Ogni linea della porta si configura in modo indipendente, per cui nella stessa porta potremo avere contemporaneamente linee di ingresso e di uscita.

Il piedino RA4 è speciale, dato che oltre a comportarsi come una tipica linea di I/O, si può programmare perché serva come ingresso di impulsi esterni per il controllo del timer TMR0, che studieremo più avanti.



Posizione dei piedini della porta A.

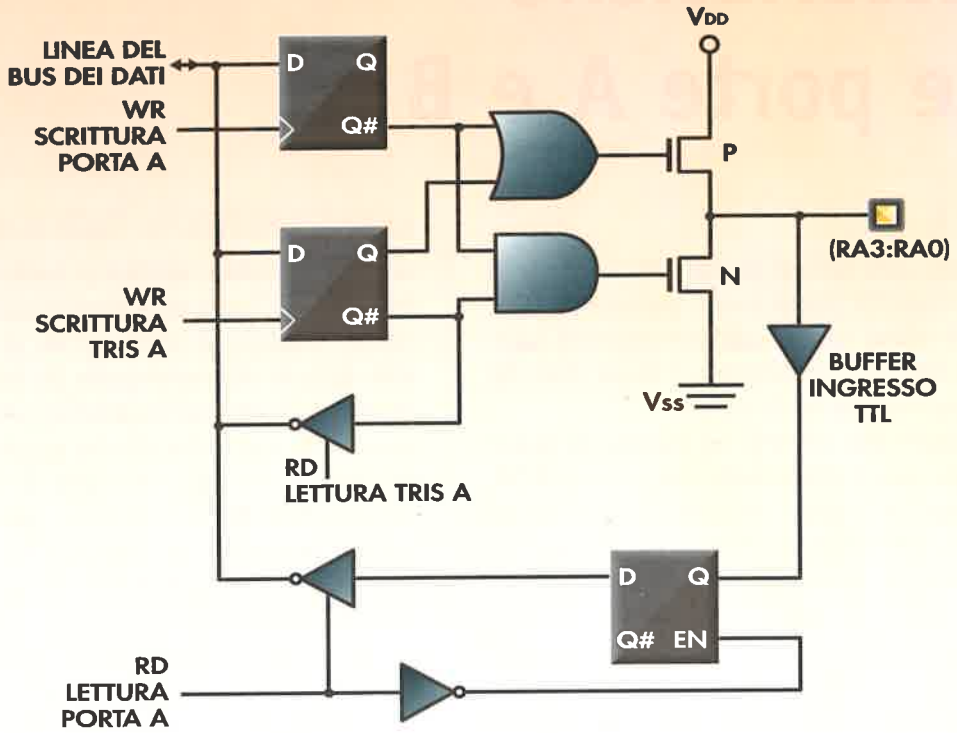
CARATTERISTICHE TECNICHE

Le linee RA0-RA3 accettano livelli di ingresso TTL e CMOS. Nella figura della pagina seguente è mostrato il circuito elettrico di adattamento di questi pin con le linee di controllo e con quelle del bus dei dati del processore. Leggere una linea della porta A significa catturare il livello logico che essa ha in quel momento, e questo viene fatto ogni volta che si esegue l'istruzione "movfw PORTAA". Al contrario, quando le linee della porta A lavorano come uscite, lo fanno tramite il latch interno, che mantiene stabile nel tempo l'ultimo valore caricato in PORTAA.

Come si può vedere dalla figura, per portare un livello logico su una linea della porta A, prima viene scritto nella linea corrispondente del bus interno dei dati e viene attivato il segnale di WRITE, il quale permette la scrittura di questo livello sul flip-flop dei dati. In questa situazione il flip-flop di configurazione dovrebbe contenere un livello 0, in modo che si comporti come un'uscita. Con questi valori la porta OR avrà uno 0 alla sua uscita e la stessa cosa succederà alla porta AND. Questi valori origineranno la conduzione del transistor PMOS superiore e l'interdizione di quello NMOS inferiore. In queste condizioni il pin di I/O resterà collegato a VDD e quindi avrà livello alto.

Dato che la linea di uscita è "lecciata" (cioè è comandata da un latch) conserverà il suo valore logico sino a che non scriveremo nuovamente sul flip-flop D.

Se una linea funziona come ingresso, quando si attiva il segnale READ il livello logico depositato in essa dall'esterno, passa alla linea corrispondente del bus interno dei dati, e il dispositivo ad alta impedenza che li unisce diventa conduttore. Dato che il pin è programmato come ingresso, i due transistor MOS rimangono bloccati, e la linea di uscita passa ad alta impedenza. Quando si legge un piedino, si rileva il valore che ha in quel momento, e non deve essere memoriz-



Circuiteria interna che collega i piedini RA0-RA3 con le linee di controllo e il bus dei dati interno al processore.

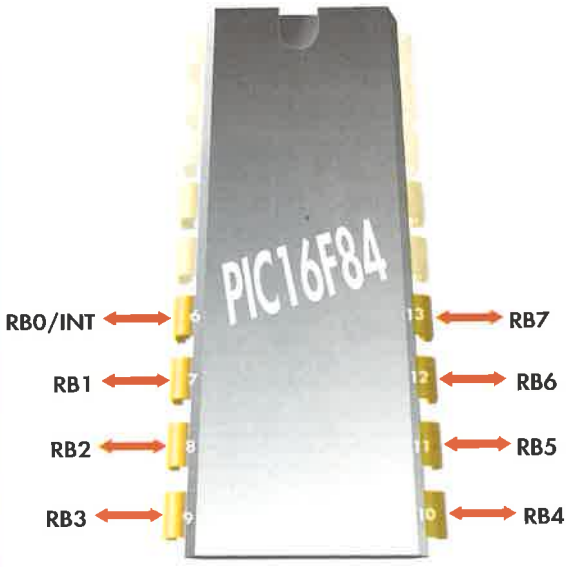
zato nel flip-flop dei dati. L'informazione presente nella linea d'ingresso, deve già essere presente al momento d'inizio del ciclo di istruzione, e deve mantenersi stabile durante tutta la sua esecuzione.

LA PORTA B

È formata da otto linee che comunicano con l'esterno mediante i piedini RB0-RB7.

Il registro di I/O contiene le informazioni che si scambiano con le periferiche, si chiama PORTAB, e occupa l'indirizzo 6 del banco 0. I bits del registro associato TRISB servono per configurare le linee di questa porta come ingressi o come uscite (1: ingressi e 0: uscite) ed è situato all'indirizzo 6 del banco 1 dell'area SFR della RAM.

Come succedeva con la porta A, anche la porta B ha una linea speciale che ha multiplexate nel tempo due funzioni possibili. Si tratta della linea RB0/INT che oltre a funzionare come linea di I/O normale, può funzionare come linea di ingresso di interrupt esterno per l'hardware. Se si programma questo piedino per funzionare come ingresso di interrupt, ogni volta che lo si



Distribuzione dei piedini della porta B.

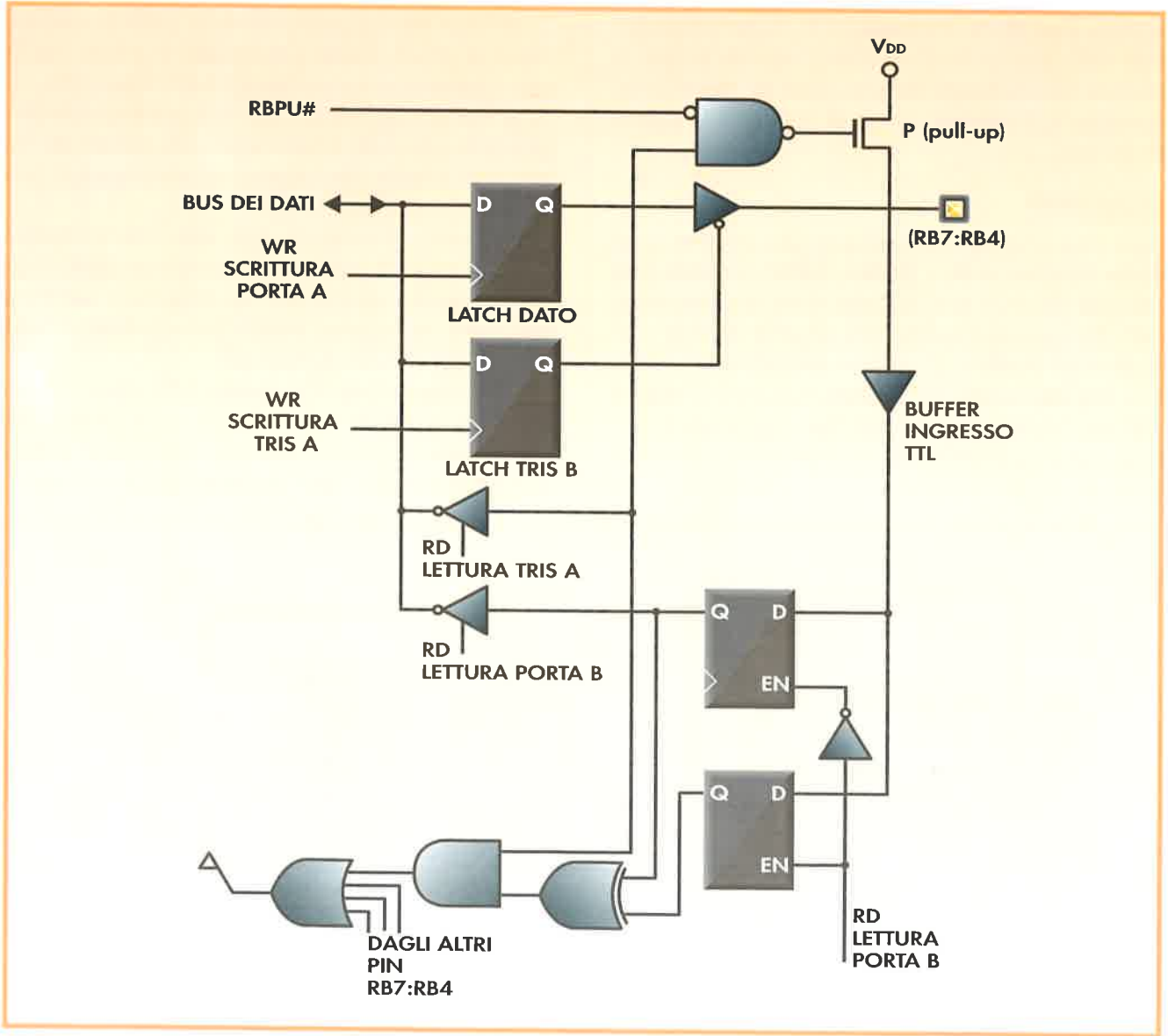
attiverà tramite un livello alto, verrà generato un interrupt.

LE DUE METÀ DELLA PORTA B

A tutte le linee di questa porta è possibile collegare delle resistenze di pull-up di valore elevato, collegate a loro volta al positivo della tensione di alimentazione. Se si vuole realizzare questa connessione dobbiamo mettere a zero il bit RBPU# che c'è nel registro specifico OPTION della RAM. Quando si produce un Reset vengono scollegate tutte le resistenze di pull-up, e tutti i piedini della porta B vengono configurati come ingressi, come succedeva con la porta A.

Le quattro linee più significative della porta B (RB4-RB7) insieme, possono svolgere un compito molto interessante e pratico, orientato alla gestione delle tastiere. Le tastiere matriciali hanno i loro pulsanti disposti nella configurazione di quattro file per quattro colonne. Per rilevare l'attivazione di un pulsante e individuare quale sia questo pulsante, si collegano le quattro file della tastiera ai piedini RB4-RB7.

Le quattro colonne si collegano agli altri quattro piedini della porta B (RB0-RB3), in modo da introdurre un livello logico alto in modo sequenziale, uno dietro l'altro. L'attivazione di un pulsante permette la comunicazione di una fila con una colonna, pertanto si avrà un



Collegamenti dei piedini RB4-RB7 con le linee di controllo e quelle del bus dei dati interno al processore.

livello alto, sia sulla fila che sulla colonna dove è situato il pulsante premuto.

Si possono programmare i quattro piedini più significativi della porta B, in modo che se uno di questi cambia il suo stato logico viene generato un interrupt. Quando si attiva un pulsante, cambia il livello logico di un piedino e si genera un interrupt, la cui routine individua il pulsante premuto conoscendo la fila e la colonna che sono passate a livello alto.

Il circuito elettrico di adattamento dei piedini RB4-RB7 è abbastanza complesso, ed è mostrato nella figura.

I piedini RB6 e RB7 inoltre hanno una funzione speciale, durante la funzione di scrittura del programma nella memoria FLASH. La scrittura in memoria si realizza inviando le informazioni in modo seriale tramite il PC, per cui sono necessari il segnale di clock e il valore binario di ogni impulso, ossia dei dati in serie. Il piedino RB6 riceve il segnale di clock e RB7 l'ingresso dei dati.

CONSUMO

Ogni linea d'ingresso può erogare una corrente il cui valore massimo arriva a 20 mA, mentre un'uscita può assorbire fino a 25 mA. Dato che esiste una limitazione nella dissipazione massima della potenza del chip, l'assorbimento massimo di corrente della porta B è limitato a 150 mA, mentre l'erogazione massima è limitata a 100 mA. La porta A, con meno linee, raggiunge la corrente massima di assorbimento di 80 mA e 150 di erogazione.

GESTIONE DI UNA TASTIERA CON LA PORTA B

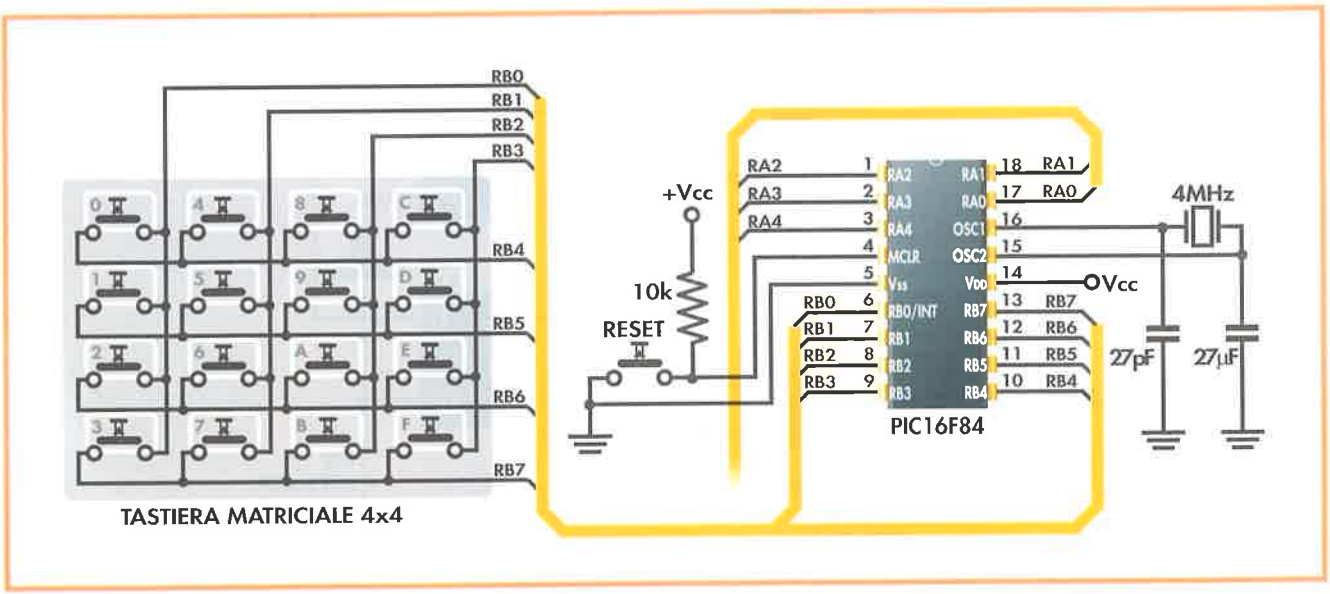
Le tastiere più comuni, impiegate nelle applicazioni con microcontrollori, dispongono di 16 tasti, numerati da 0 a F e disposti in matrice di quattro file per quattro colonne. Ogni volta che si attiva un pulsante il processore deve riconoscere immediatamente quale sia stato pigiato. La possibilità di generare un interrupt quando varia lo stato logico di una delle quattro linee più significative della porta B, semplifica la rivelazione della tastiera, per questo si collegano, come indica la figura, le quattro linee meno significative della porta B con le quattro colonne, e le quattro file con le linee RB4-RB7 della porta B.

Le linee meno significative della porta B vengono messe a uno in modo sequenziale, in modo che in ogni momento ci sia una linea a livello logico alto.

Se non ci sono pulsanti premuti le file sono scollegate dalle colonne e il livello logico alto inviato alle file non può passare alle colonne; di conseguenza non cambierà lo stato di RB4-RB7.

Premendo un pulsante, ogni volta che la colonna corrispondente a questo pulsante invia un livello logico alto, questo passerà alla fila associata e cambierà stato uno dei quattro piedini più significativi della porta B, generando un interrupt.

La routine dell'interrupt verifica la fila da cui arriva il livello logico alto e la colonna da cui è partito, per determinare il pulsante che è stato premuto, in pratica quello che si trova all'incrocio fra la fila e la colonna.



Circuito per gestire una tastiera con le linee della porta B.

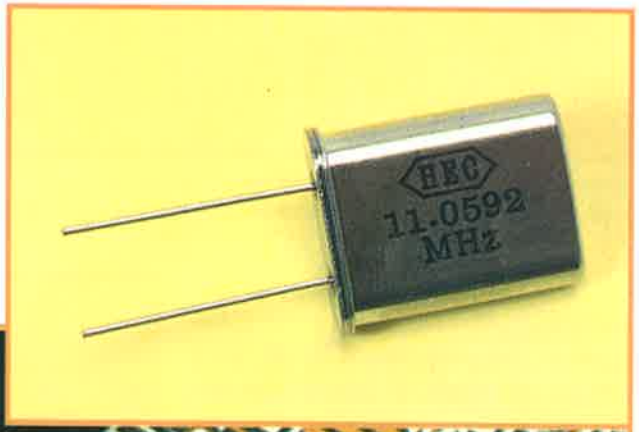
Parola di configurazione e identificazione

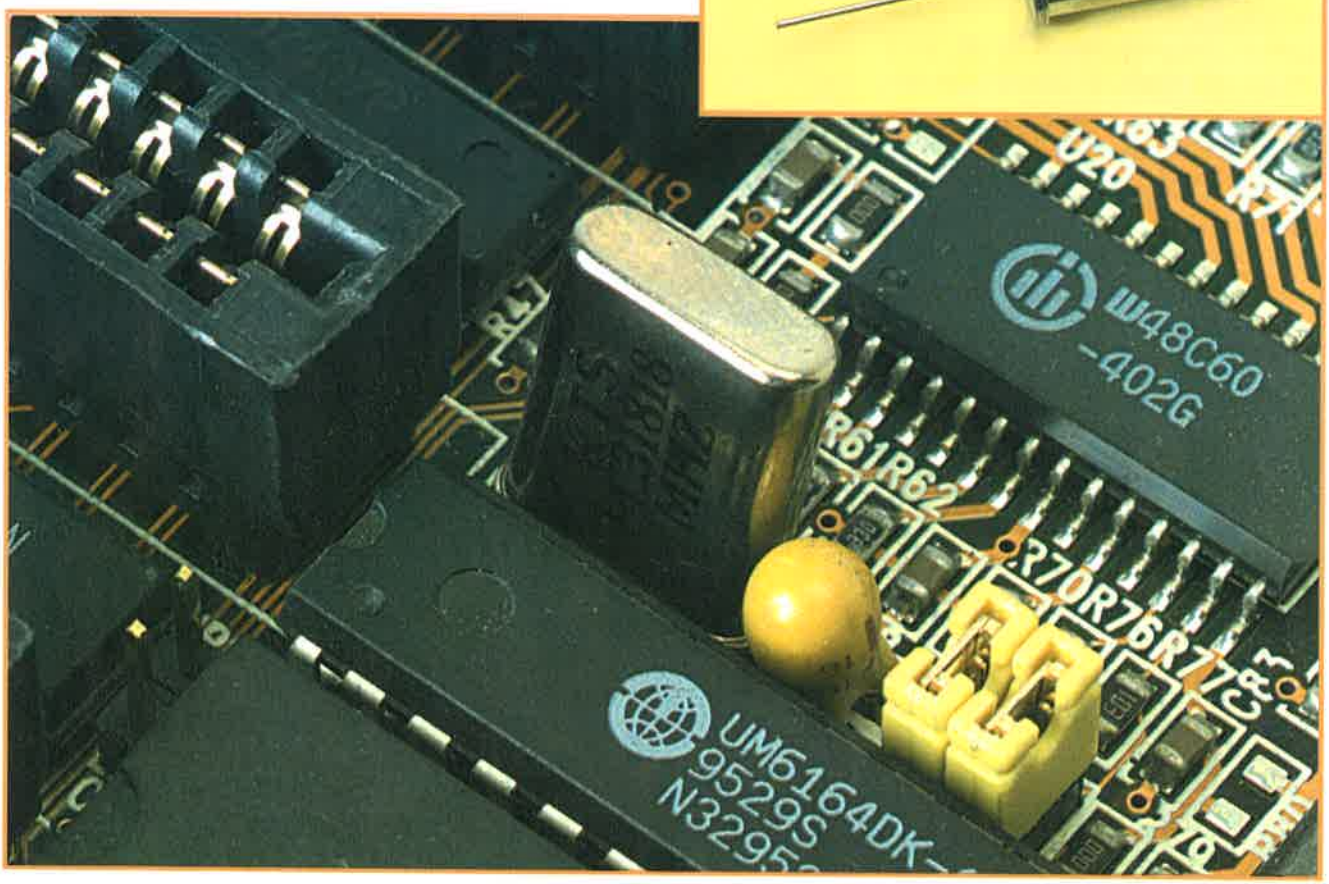
Per fare in modo che il PIC16F84 elabori le istruzioni – che è l'unica cosa che sa fare – è necessaria la presenza di un clock, il cui impulso determina la velocità di lavoro. L'oscillatore che genera questi impulsi è implementato dentro il circuito integrato, però, per regolare, selezionare e stabilizzare la frequenza di lavoro, dobbiamo montare esternamente alcuni componenti, fra i piedini OSC1/CLKIN e OSC2/CLKOUT, che corrispondono rispettivamente ai piedini 15 e 16 del contenitore.

Un'istruzione impiega normalmente 4 periodi di clock ad essere eseguita, e se si tratta di un'istruzione di salto, impiega il doppio.

Secondo il tipo di componenti esterni che monteremo, si configurano quattro tipi di oscillatori:

Tipo RC: è un oscillatore di basso costo poco stabile. Necessita solo di una resistenza e di un condensatore esterno.

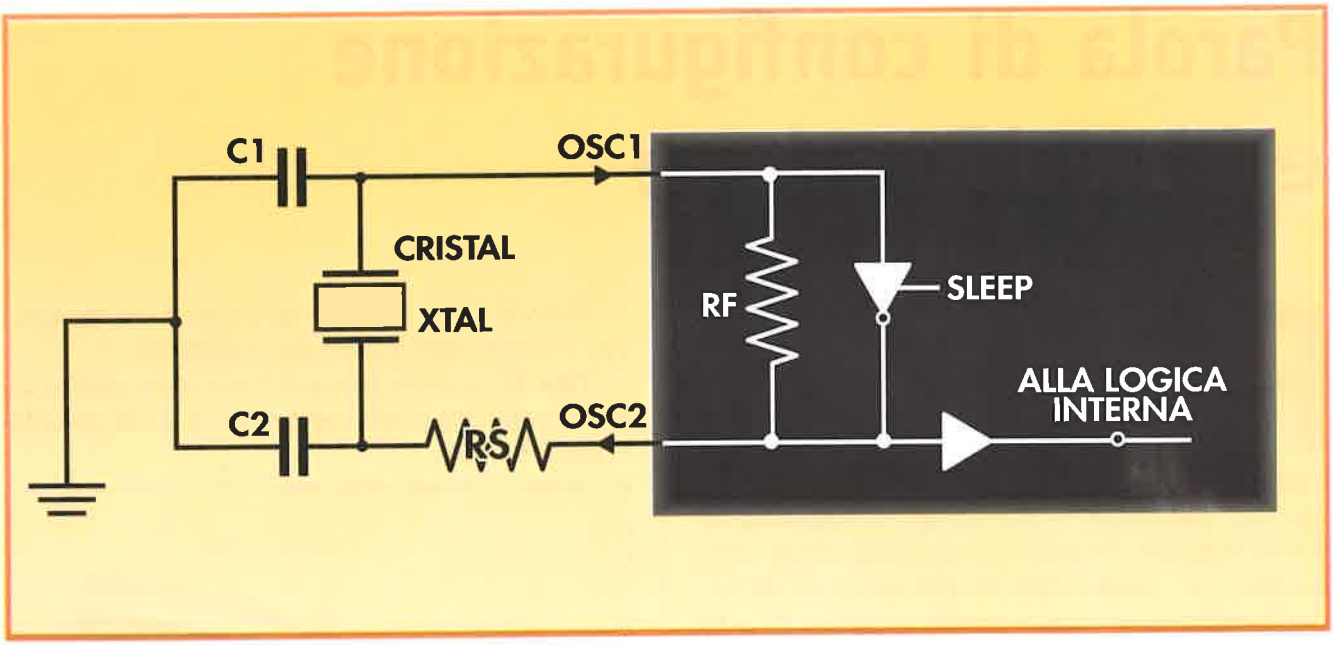




Fotografia di un cristallo di quarzo e parte di un circuito nel quale è montato.

MICROCONTROLLER

Ripasso dei concetti sulla frequenza di funzionamento



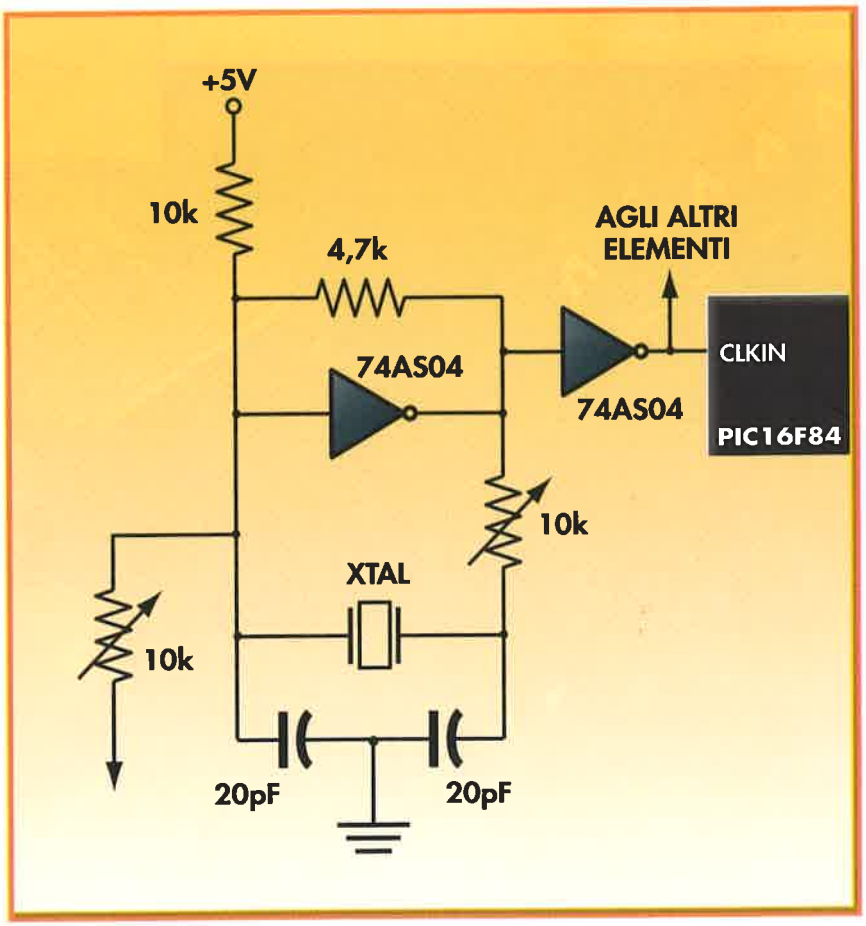
Schema generale dell'interfaccia del cristallo di quarzo e dei condensatori esterni. La resistenza RS si utilizza solo nella versione speciale ad alta frequenza.

TIPO HS: è un oscillatore ad alta velocità molto stabile. Funziona a frequenze comprese tra 4 e 20 Mhz. Utilizza un cristallo di quarzo o un risonatore ceramico, collegato a due condensatori di disaccoppiamento.

TIPO XT: impiega anch'esso un cristallo al quarzo o un risonatore ceramico, collegato ad un condensatore. Lavora in frequenze medie comprese tra 100 KHz e 4 MHz.

TIPO LP: Impiegato in applicazioni a basso consumo, quindi per basse frequenze – infatti a maggiore velocità si ha maggiore consumo –, usa cristalli o risonatori e le frequenze di lavoro oscillano tra 35 e 200 KHz.

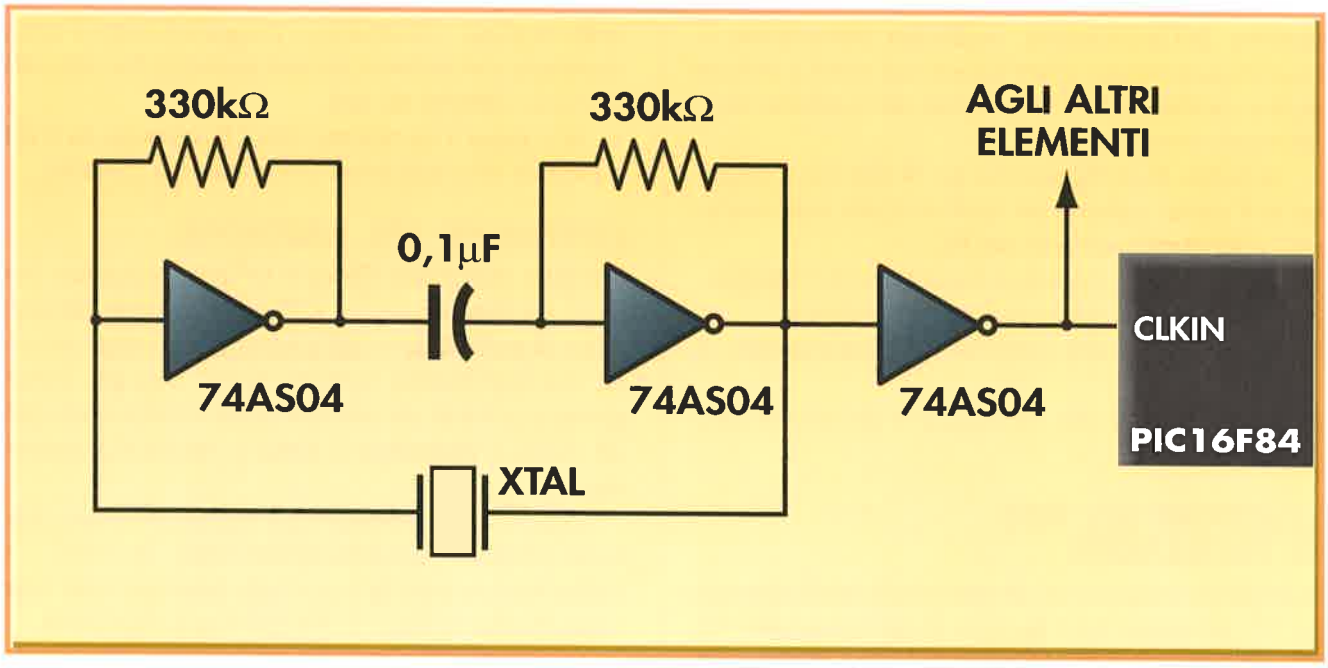
Nella figura si mostra lo schema degli oscillatori HS, XT e LP che usano un cristallo di quarzo e due condensatori di disaccoppiamento, il cui valore è specificato dal fabbricante in funzione della frequenza.



Schema elettrico dell'oscillatore di cristallo a risonanza parallela.

ALTRI CIRCUITI OSCILLATORI

Impiegando alcune porte logiche TTL molto semplici si possono costruire



Schema elettrico dell'oscillatore di cristallo a risonanza seriale.

oscillatori che hanno un ottimo comportamento e un'ampia gamma di frequenza. Ci sono due tipi di oscillatori al quarzo: a risonanza parallela e a risonanza in serie.

Il circuito oscillatore con risonanza parallela illustrato nella figura oscilla alla frequenza del cristallo che incorpora. La porta invertente 74AS04 realizza un'inversione di fase di 180°.

Il circuito oscillatore a risonanza in serie – il modello riportato nella figura – permette un'elevata precisione nella generazione della frequenza, che è determinata dal cristallo di quarzo.

LA PAROLA DI CONFIGURAZIONE

Risiede all'indirizzo 2007H della memoria FLASH, che è riservata per un compito speciale, infatti è accessibile solamente durante il processo di scrittura del programma nel PIC16F84.

Questa registro di 14 bits riceve la denominazione di "Parola di configurazione". Quando viene scritto il pro-



Distribuzione dei bits nella parola di configurazione con i quattro campi che la compongono.

gramma dell'applicazione, dobbiamo memorizzare in essa un determinato valore binario che andrà a definire alcune caratteristiche fondamentali del funzionamento del microcontrollore.

La parola di configurazione ha 14 bits che si dividono in 4 campi, ognuno dei quali controlla una caratteristica del comportamento del PIC:

1. Campo dei bits che selezionano il tipo di oscillatore.
2. Campo dei bits che controllano il Watchdog.
3. Campo dei bits che controllano il temporizzatore di "POWER-UP".
4. Campo dei bits che controllano la protezione della memoria.

SELEZIONE DEL TIPO DI OSCILLATORE

Quando si memorizza un programma di applicazione in un PIC, bisogna scrivere la parola di configurazione, e in uno dei suoi campi dobbiamo specificare il tipo di oscillatore fra i quattro possibili, con cui funzionerà il microcontrollore.

A questo scopo sono riservati i due bits meno significativi di questa parola, che si chiamano FOSC1 e FOSC0, nei quali bisogna scrivere i codici indicati nella tabella per specificare l'oscillatore utilizzato.

TIPO DI OSCILLATORE	FOSC1	FOSC0
RC	1	1
HS	1	0
XT	0	1
LP	0	0

PROTEZIONE DELLA MEMORIA

In molte situazioni, il fabbricante o i progettisti di un prodotto governato da un microcontrollore, non vogliono che si conosca il programma e i dati presenti nella memoria. In altri casi questo non è importante, quindi sia il programma che i dati possono essere letti.

I 10 bits di maggior peso della parola di configurazione sono denominati CP (Code Protection) ed hanno il compito di proteggere o meno l'accesso alla memoria dei codici.

Se si scrive un 1 nei bits CP, la memoria sarà sprotetta, e si potrà leggere liberamente. Nel caso che i bits CP siano a 0, il programma non si potrà leggere, evitando le copie. Ovviamente il programma non si potrà nemmeno sovrascrivere, né sarà permesso l'accesso alla memoria EEPROM dei dati.

Se si prova a modificare i bits CP passando da 0 a 1 si provoca la cancellazione completa della EEPROM.

CONTROLLO DEL WATCHDOG

Il WTD o Watchdog Timer, è un temporizzatore che serve per controllare che il corretto sviluppo del processo di elaborazione delle istruzioni avvenga secondo una determinata cadenza. Se per esempio, il programma entra in un ciclo chiuso da cui non riesce più ad uscire, il Watchdog si attiva e resetta il programma.

La parola di configurazione dispone di un bit chiamato WTDE che controlla il Watchdog. Se WTDE = 1 il Watchdog è attivato e controlla l'elaborazione delle istruzioni, Se WTDE = 0 il Watchdog è disattivato.

TEMPORIZZATORE "POWER-UP"

Il temporizzatore Power-Up controlla un tempo fisso di 72 ms e serve ad evitare problemi durante il collegamento iniziale della tensione di alimentazione al microcontrollore.

A volte la tensione di alimentazione, nel momento immediatamente successivo all'accensione, è instabile e conviene ritardare la sua applicazione alla circuiteria interna del PIC sino alla sua completa stabilizzazione.

Nella parola di configurazione il bit PWRTE# si usa per attivare o disattivare il temporizzatore Power-Up, che ritarda la connessione dell'alimentazione esterna alla circuiteria interna.

LA PAROLA DI IDENTIFICAZIONE

Similmente a quello che succede con la parola di configurazione, a cui è riservato l'indirizzo 2007H della memoria di programma, esistono altre quattro posizioni che sono riservate per le parole di identificazione e che occupano gli indirizzi dal 2000H al 2003H. Non sono accessibili durante il funzionamento normale del PIC, e possono solo essere letti o scritti durante il processo di memorizzazione del programma.

Nelle parole di identificazione sono solo validi i 4 bits meno significativi di ogni posizione da 14 bits. Esse sono utilizzate per immagazzinare codici di identificazione, numeri di serie, chiavi segrete, dati di fabbricazione, numeri sequenziali eccetera.

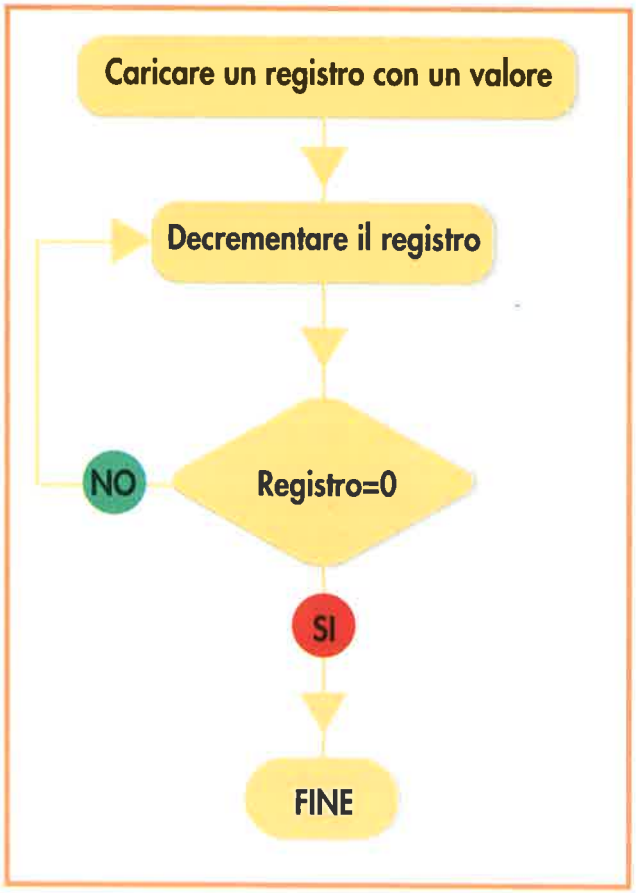
Il temporizzatore TMR0

Nella maggior parte delle applicazioni in cui intervengono dei microcontroller troviamo dei procedimenti e dei compiti in cui è necessario controllare il tempo. Per far fronte a questa esigenza si può istruire il proprio processore mediante un semplice programma, che inizi caricando un valore in un registro, e poi lo decrementi sino ad arrivare a zero. Se con un solo registro non si ottiene il tempo desiderato, se ne possono concatenare diversi, in modo che quando uno arriva a zero, decrementi di una unità il successivo e ripeta il ciclo. Questo procedimento è molto semplice e molto preciso, perché conoscendo il numero di istruzioni da eseguire, si deduce il tempo totale necessario ad eseguire il ciclo completo.

L'inconveniente principale nel dedicare il processore a compiti di cronometro, è che vengono ipotecate le sue risorse per realizzare altre operazioni, inoltre non potrebbero essere gestiti gli interrupt, perché significherebbe fermare il conteggio del tempo. I temporizzatori sono dispositivi hardware ausiliari integrati nel microprocessore, e sono dedicati alla misura del tempo, scaricando da questo lavoro il processore.

CHE COS'È UN TEMPORIZZATORE?

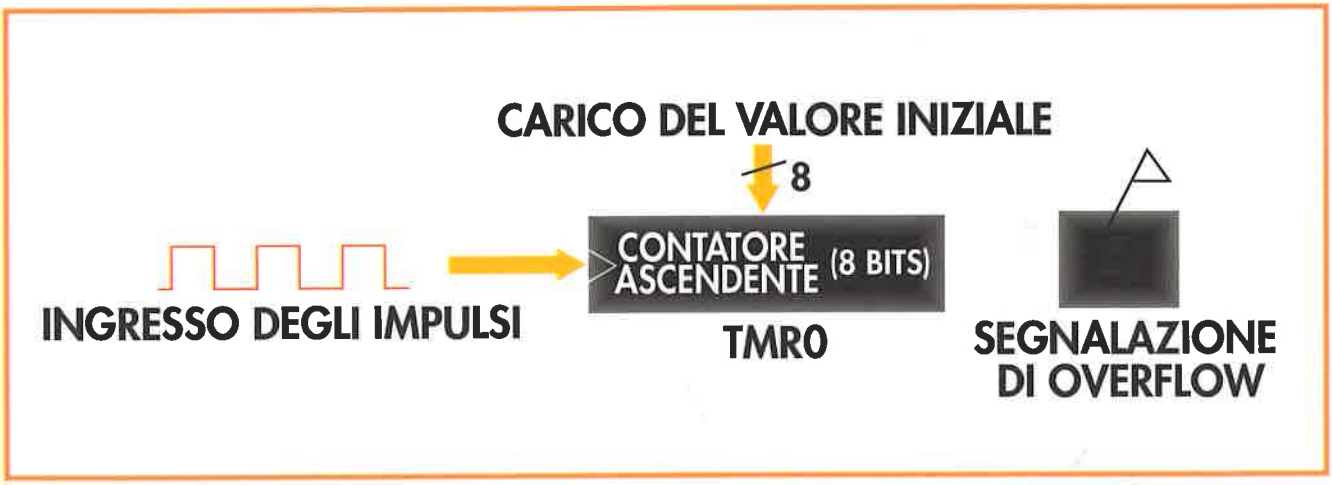
È un contatore di n bit che si incrementa o decrementa ogni volta che riceve un impulso di clock. In caso di



Organigramma di un progetto tipico per il controllo del tempo mediante il decremento di un registro.



I temporizzatori sono contatori che si incrementano o decrementano con gli impulsi di clock, dispongono di un bit di segnalazione che va a 1 al termine del conteggio.



Schema e funzionamento del temporizzatore TMR0.

decremento quando il valore del contatore arriva a zero, viene attivato un bit ausiliario che funziona come segnalatore, o "flag", e va a 1 avvisando dell'accaduto. In caso che il contatore incrementi il suo valore con gli impulsi del clock, quando supera il valore massimo e arriva l'impulso che fa passare i bits del valore di conteggio, da tutti uno a tutti zero, il segnalatore va a 1 come indicato nel disegno.

Il grande vantaggio di utilizzare temporizzatori, è scaricare il processore dai compiti di controllo del tempo. Se il processore vuole sapere quando il timer ha terminato il suo conteggio, deve vigilare sulla messa a uno del segnalatore o flag. Questa continua esplorazione del flag sottrae tempo di esecuzione al processore, per questo esiste un'altra possibilità più immediata e chiara, che consiste nel produrre un interrupt ogni volta che il contatore finisce il suo ciclo.

I TEMPORIZZATORI DEL PIC16F84

Questo modello di microcontroller dispone di vari temporizzatori integrati nel chip. Un paio di questi dispositivi sono utilizzati per ritardare di un tempo costante la messa in marcia del PIC, durante la prima accensione o in caso di reset. Quelli che ci interessano sono quelli programmabili, e che sono dedicati alla misura del tempo all'interno dei compiti a cui è dedicato il microcontroller. Per questa finalità il PIC16F84 dispone di due temporizzatori da 8 bits: il TMR0 che è di utilizzo generale e il WDT, Watchdog Timer che vigila sul corretto tempo di elaborazione delle istruzioni del programma, resettando il processore in caso di anomalia.

IL TEMPORIZZATORE TMR0

È formato da un contatore ascendente a 8 bits che può essere caricato inizialmente con qualsiasi valore e successivamente si incrementa ad ogni impulso di clock, sino a superare il valore massimo, cioè il passaggio dal maggior valore di conteggio (tutti uno) al minore (tutti zero). Quando il contatore passa dal valore FFH al valore 00H si attiva il flag.

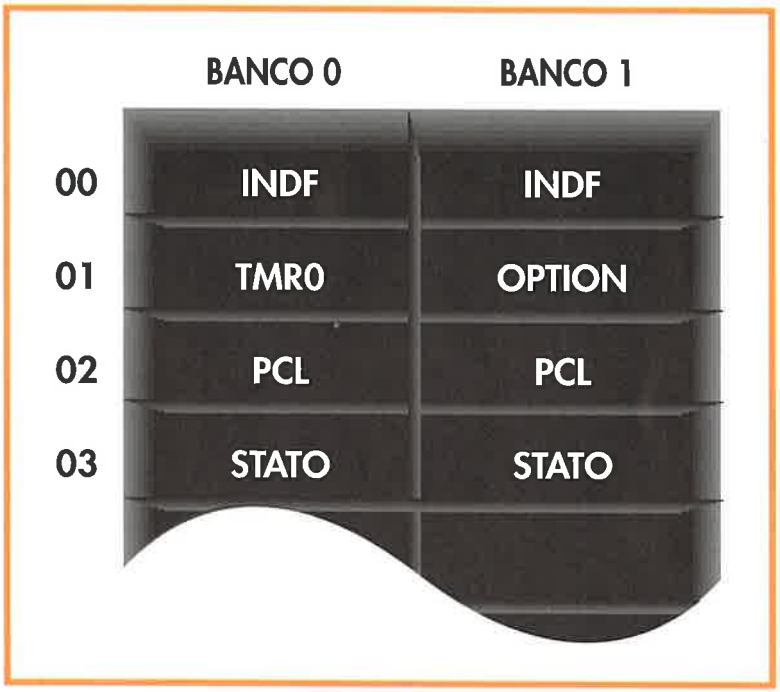
Il temporizzatore TMR0 può realizzare due funzioni principali:

1ª. Temporizzatore o controllore del tempo. Ha il compito di misurare il tempo fra il carico di un valore iniziale, e i successivi incrementi ad ogni ciclo di istruzione (Fosc/4) sino al superamento del valore massimo, o overflow, che può essere rivelato mediante il flag o producendo un interrupt.

2ª. Contatore di eventi. In questo caso ogni evento che deve essere contato, è rappresentato dall'impulso di clock che si applica tramite il pin RA4/T0CKI. Se va in overflow, il contatore mette il flag a 1 oppure produce un interrupt.

Per fare in modo che il TMR0 funzioni come contatore di eventi esterni, dobbiamo programmare il piedino RA4/T0CKI come ingresso, e caricare alcuni bits del registro specifico OPTION con determinati valori. Il registro con il valore del contatore TMR0 occupa l'indirizzo 1 del banco 0 della RAM, mentre il registro OPTION – di cui alcuni bits programmano il funzionamento del TMR0 – occupa l'indirizzo 1 del banco 1 dell'area SFR della RAM.

Come vedremo in seguito, il registro OPTION dispone di due bits che determinano il comportamento del temporizzatore come contatore di impulsi. Uno di que-



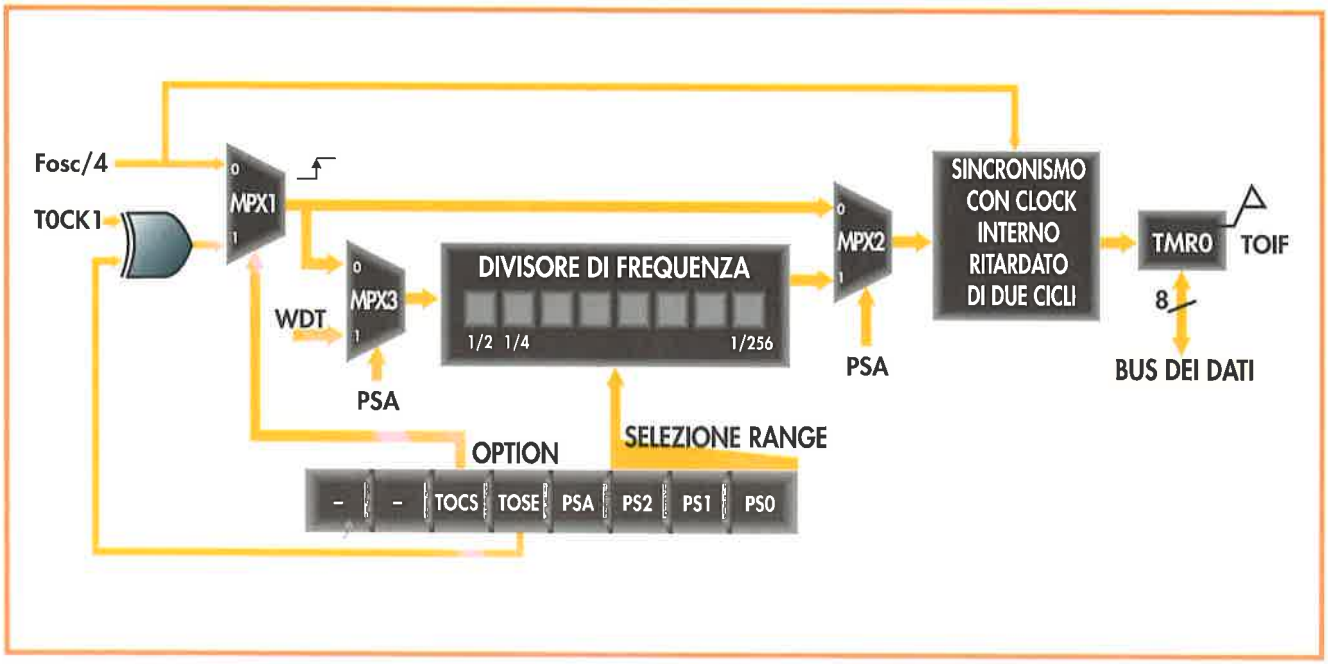
Il contatore TMR0 occupa l'indirizzo 1 del banco 0 e il registro OPTION, di cui alcuni bits programmano il funzionamento del TMR0, occupa l'indirizzo 1 del banco 1.

sti bits si chiama TOCS, e se vale 1 il TMR0 funziona come contatore di impulsi, se vale 0 come temporizzatore di utilizzo generale. L'altro bit chiamato TOSE, sceglie il fronte attivo per il conteggio del segnale applicato al pin RA4/T0CKI, con il quale si determina l'incremento del contatore. Se TOSE = 1 il fronte attivo sarà quello di discesa e se TOSE = 0 sarà quello di salita.

IL DIVISORE DI FREQUENZA

In molte occasioni, sia il TMR0 che il Watchdog devono gestire lunghi intervalli di tempo, e necessitano di aumentare il tempo che intercorre fra ogni impulso di clock, per ritardare l'incremento, e di conseguenza aumentare la durata dell'intervallo. Per far fronte a questa necessità, si dispone di un divisore di frequenza programmabile, che divide la frequenza di clock per i diversi valori possibili.

Abbiamo un solo divisore di frequenza, che deve essere assegnato al TMR0 o al Watchdog; per fare questa scelta disponiamo di un bit nel registro OPTION, chiamato PSA. Se PSA = 0 il divisore di frequenza è applicato al TMR0, con il quale funziona come Predivisore, o Prescaler. In questo modo gli impulsi del clock passano prima attraverso il divisore di frequenza, vengono divisi per il valore scelto e sono poi applicati al contatore TMR0 per ottenere l'incremento di valore. Gli impulsi del clock possono provenire dal proprio oscillatore interno, con una frequen-



Schema interno del TMR0 con i bits di programmazione del registro OPTION.

za di Fosc/4, o da impulsi esterni applicati al piedino RA4/T0CKI. Questi impulsi passano al Prescaler che li divide per un valore impostato dai tre bits meno significativi di OPTION (PS2, PS1, PS0).

Per calcolare il tempo che controlla il temporizzatore TMR0, dobbiamo tener presente il valore caricato inizialmente e il fattore di divisione del Prescaler, in accordo con la seguente formula:

Temporizzazione = 4 . Tosc . Valore caricato in TMR0 . Fattore di divisione del Prescaler

Valore da caricare in TMR0 = Temporizzazione / (4 . Tosc . Fattore di divisione del Prescaler)

In qualsiasi momento si può leggere il valore presente in TMR0 senza influenzare il conteggio, per cui si utilizza l'istruzione "movfTMR0, W" che carica nel registro di lavoro il valore del contatore nel momento in cui si esegue l'istruzione. Nella figura della pagina precedente è riportato lo schema interno del TMR0, e si può notare l'esistenza di un circuito che ritarda di due cicli il conteggio, per sincronizzare il momento dell'incremento prodotto dal segnale esterno T0CKI con il quale si producono gli impulsi del clock interno. Quando si scrive il TMR0 si ritarda di 2 cicli il suo reincremento e si pone a 0 il divisore di frequenza o Prescaler.

IL REGISTRO OPTION

Questo registro occupa la posizione 1 del banco 1 nell'area SFR della RAM, che ha il compito di controllare le risorse del microcontroller. I bits di OPTION, principalmente governano il comportamento dei temporizzatori TMR0 e WDT; inoltre il bit RBPU# serve per collegare o meno le resistenze di pull-up della porta B. Un altro bit, INTEDG, decide quale sarà il fronte attivo per l'interrupt esterno che potrà essere collegato alla porta RB0/INT. Nella figura è indicata la distribuzione e la nomenclatura dei bits del registro OPTION così come la funzione che ognuno di essi compie a seconda del valore, che può essere 1 o 0.

OPTION

RBPU#	INTDG	TOCS	TOSE	PSA	PS2	PS1	PS0

PS2:PS0 Valori con i quali funziona il divisore di frequenza

PS2	PS1	PS0	Divisione del TMR0	Divisione del WDT
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

PSA: **Assegnazione del divisore di frequenze**
1= Il divisore di frequenze si assegna a WDT
0= Il divisore di frequenze si assegna a TMR0

TOSE: **Tipo di fronte in T0CK1**
1= Incremento di TMR0 con fronte di discesa
0= Incremento di TMR0 con fronte di salita

TOCS: **Tipo di clock per il TMR0**
1= Impulso in ingresso da TOCK1 (contatore)
0= Impulso di clock interno Fosc/4 (temporizzatore)

INTEDG: **Fronte attivo per interrupt esterno**
1= Fronte di salita
0= Fronte di discesa

RBPU#: **Resistenze di pull-up Porta B**
1= Scollegate
0= Collegate

Distribuzione e funzionalità dei bits che configurano il registro OPTION.

Un cane che morde



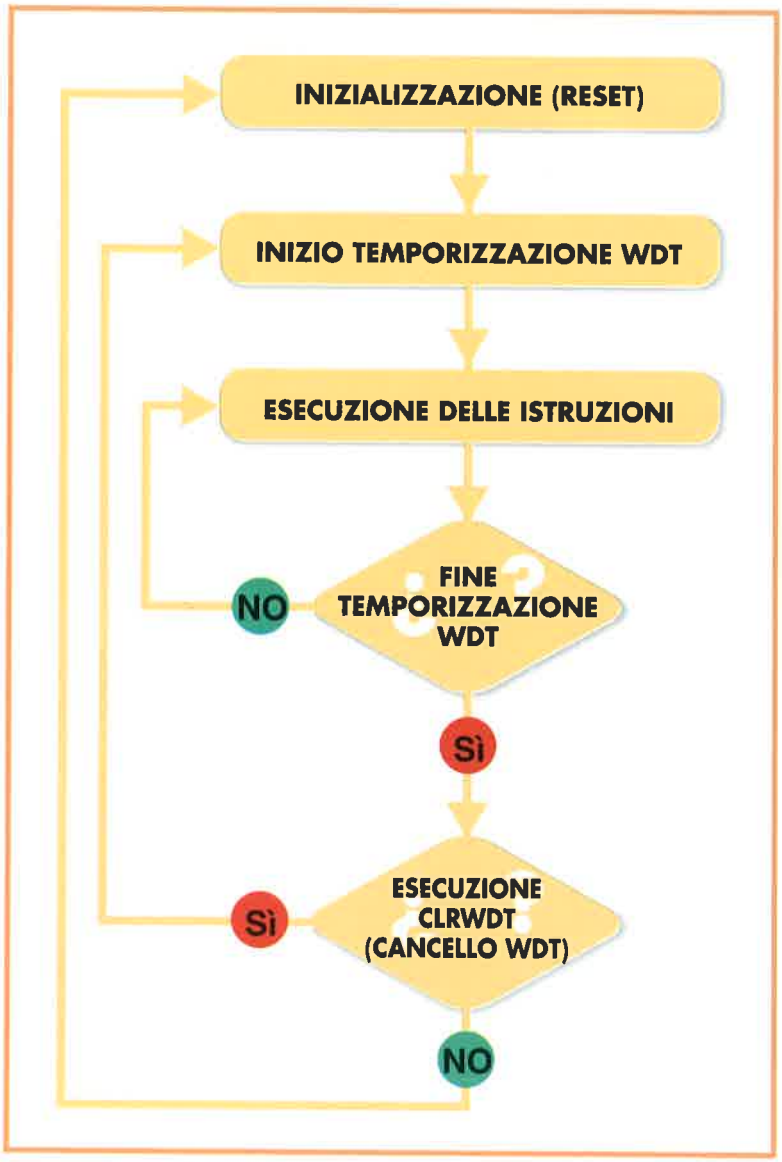
Questo è Aibo, un cane microrobot costruito dalla Sony, che sa anche mordere. È una mascotte disegnata per coloro che abitano in città, e non devono preoccuparsi di portare il cane a spasso, dargli da mangiare, evitare che abbai di notte ecc. Può capire 14 messaggi, o ordini del padrone, impara perché è sensibile alle carezze e ai castighi. Gioca con la palla, segue il padrone, abbaia, dice il suo nome, alza la zampa quando fa pipì ecc. Con la sua telecamera vigila e morde gli estranei, e non ha niente a che vedere con il Watchdog che contiene il PIC e che deve solo resettare il microcontroller, quando il programma di lavoro non è eseguito normalmente.

MICROCONTROLLER

Il Watchdog

Il Watchdog, chiamato brevemente WDT (Watchdog Timer) è un contatore ascendente a 8 bits che ha la proprietà di mandare il microcontroller in reset ogni volta che va in overflow. A differenza del TMR0, il Watchdog non dispone di segnalatori, né genera interrupt alla fine del ciclo, l'unica cosa che fa è provocare il reset del microprocessore, il quale ricomincia ad eseguire il programma principale dalla prima istruzione 0000H della memoria dei codici (Vector Reset).

L'obiettivo del Watchdog è verificare la corretta elaborazione delle istruzioni del programma. Se tutto va bene, prima che il Watchdog termini la sua temporizzazione, il programma gli fornisce un'istruzione che lo cancella e reinizia il conteggio, in altre parole effettua un "refresh". Se il programma si è "piantato", e non sono state eseguite le istruzioni secondo le previsioni, il programma non fornirà l'istruzione di refresh, di conseguenza il WTD andrà in overflow e verrà generato un reset. Con quest'azione si ottiene che il microcontroller inizi nuovamente il programma, ed eviti la causa del blocco. Un programma si può bloccare perché, ad esempio, entra in un ciclo infinito, o perché necessita di un segnale per uscirne che non arriva, ecc.



Organigramma sull'attuazione del Watchdog. Se il programma si esegue normalmente riceverà in tempo l'istruzione clrwdt e non deborderà. Se qualcosa va male, o questa istruzione non arriva in tempo, verrà generato un reset.

ORGANIGRAMMA

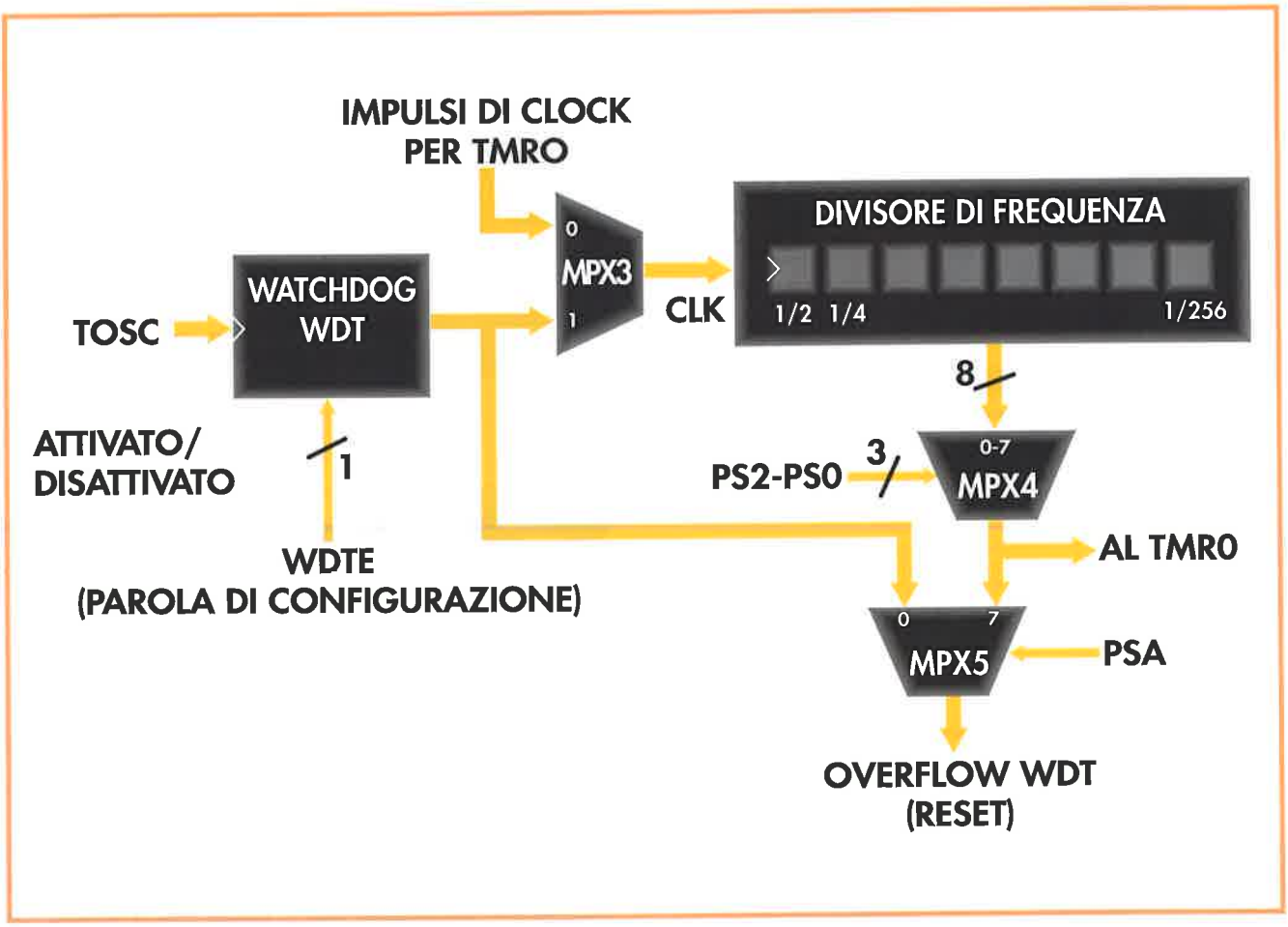
Nell'organigramma di attuazione del Watchdog si osserva che alla fine della temporizzazione del WDR si esegue l'istruzione "clrwdt", si cancella il temporizzatore e ricomincia il conteggio. Se non si esegue questa istruzione, qualcosa non ha funzionato, quindi si genera un reset. Per controllare il Watchdog si aggiunge l'istruzione "clrwdt" in certi punti del programma su cui si vuole vigilare. Con questa istruzione il contatore torna a 0 e inizia il conteggio, evitando l'overflow. Se il programma non è eseguito normalmente, l'istruzione "clrwdt" non arriverà in tempo, il WDT andrà in overflow provocando il reset del sistema, come rimedio per superare l'anomalia.

Il refresh o reinizio del conteggio del WDT, si può produrre anche quando si esegue l'istruzione "sleep", che serve per porre il PIC in uno stato di basso consumo, fermando l'oscillatore interno e sopprimendo l'elaborazione delle istruzioni. Anche se il WDT si cancella con l'istruzione "sleep", non si ferma, a meno che non venga fatto da programma.

SCHEMA DI LAVORO

Il Watchdog si attiva e si disattiva con il valore del bit WDTE della Parola di Configurazione, come abbiamo visto in precedenza.

La frequenza di lavoro di questo temporizzatore è sempre interna e indipendente, originata da un oscillatore RC proprio. Quando si assegna il Divisore di frequenza al Watchdog, questo lavora come postdivisore, al contrario di come lavora con il TMR0. In questo modo



Schema interno del funzionamento del Watchdog.

gli impulsi che genera il WDT passano al Divisore di frequenza, che li divide per il valore che selezioneremo con i tre bits meno significativi del registro OPTION.

Se il bit PSA nel registro OPTION è a 1, il Divisore di frequenza si applica al WDT. È curioso notare che il Watchdog non dispone di nessun registro nell'area SFR della RAM.

La ragione si deve al fatto che non si carica con alcun valore iniziale, iniziando sempre da zero sino all'overflow.

Ogni volta che clrwdt produce un refresh torna a zero e inizia a contare nuovamente. Nel registro STATO esiste un bit chiamato T0# che passa a valore 0 quando il WDT va in overflow, così discrimina il motivo per cui è avvenuto il reset.

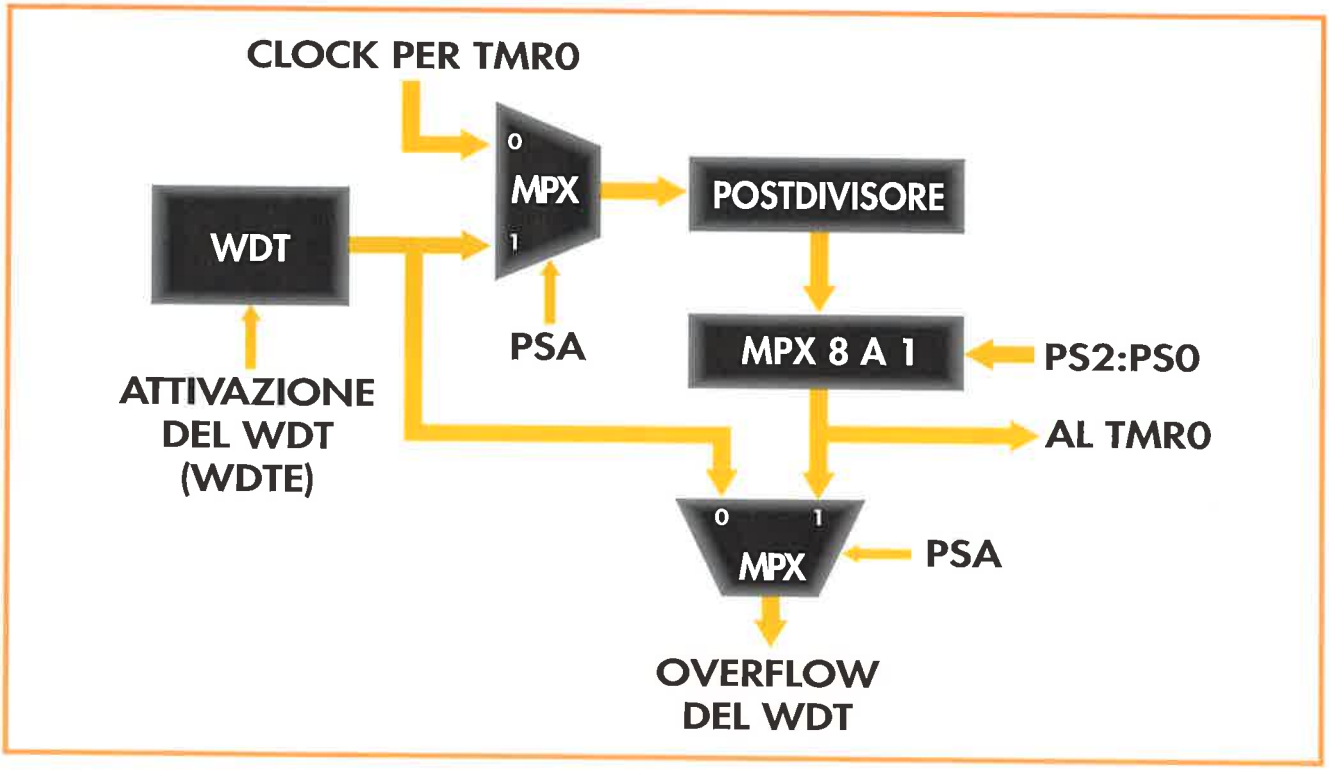
CONTROLLO DEL TEMPO DI REFRESH DEL WATCHDOG

Il temporizzatore WDT è preprogrammato per controllare un tempo di 18 ms, però regolando il Divisore di frequenza si può fare in modo che questo tempo si elevi fino ad un massimo di 2,3 secondi.

ESEMPIO

In un programma si desidera controllare che un determinato ciclo, composto da 200 istruzioni normali senza nessuna di salto, sia eseguito esattamente 100 volte. Si lavora con un PIC16F84 con un cristallo di quarzo a 4 Mhz e si utilizza il Watchdog per controllare questo tempo, che può essere critico nell'applicazione.

Il controllo del tempo del ciclo, richiede di posizionare davanti allo stesso un'istruzione "clrwdt", che resetta il Watchdog e inizializza la sua temporizzazione. Al termine del processo di 100 cicli, collocheremo nel programma principale un'altra istruzione di refresh del Watchdog. Con quale temporizzazione si deve programmare il Watchdog, per ottenere un reset nel caso sia superato il tempo che impiega il ciclo ad essere eseguito 100 volte?



Bits del registro OPTION che intervengono nel comportamento del Watchdog.

SOLUZIONE

Dato che il ciclo consta di 200 istruzioni normali, il tempo che dura ogni istruzione sarà:
Ciclo Istruzione = 4 . Tosc = 4 .250 ns = 1.000 ns = 1 µs
Tempo di Ciclo = 200 . Ciclo Istruzione = 200 . 1 µs = 200 µs
Tempo Totale = 100 . Tempo di Ciclo = 100. 200 µs = 20.000 µs = 20 ms

Il WDT deve essere programmato per andare in overflow 20 ms dopo il refresh. Gli impulsi per il WDT sono generati dall'oscillatore RC interno, proprio e indipendente dal principale.

CARATTERISTICHE DEL WATCHDOG

Con l'obiettivo di precisare tutte le caratteristiche principali di cui bisogna tenere conto per gestire correttamente il Watchdog, forniamo un riassunto di quelle più importanti e una figura che mostra graficamente i bits di controllo che intervengono nel suo comportamento.

RIASSUNTO DELLE SPECIFICHE

** Il WDT è un contatore che genera un reset ogni volta che va in overflow.*

** È controllato da un oscillatore RC interno ed indipendente dall'oscillatore principale del sistema.*

** La temporizzazione nominale è di 18 ms, anche se può essere aumentato con il valore che si sceglie nel Divisore di frequenza fra 1:2, 1:4, ...1:128, ottenendo con quest'ultimo caso una temporizzazione massima di 2,3 secondi.*

** Per stabilire il range di divisioni del Divisore di Frequenza si utilizzano i tre bits meno significativi di OPTION (PS2, PS1 e PS0).*

** Per prevenire l'overflow e il reset bisogna cancellare il WDT prima del termine della temporizzazione con l'istruzione "clrwdt" o "sleep".*

** Il WDT si attiva o disattiva con un bit della Parola di Configurazione (WDTE).*

** Se il bit PSA è a 1 nel registro OPTION, il Divisore di frequenza è assegnato al WDT.*

** Quando si esegue l'istruzione "sleep" si entra in modo basso consumo e si ferma il sistema e l'elaborazione delle istruzioni. Il WDT si cancella, però non si ferma e continua il conteggio.*

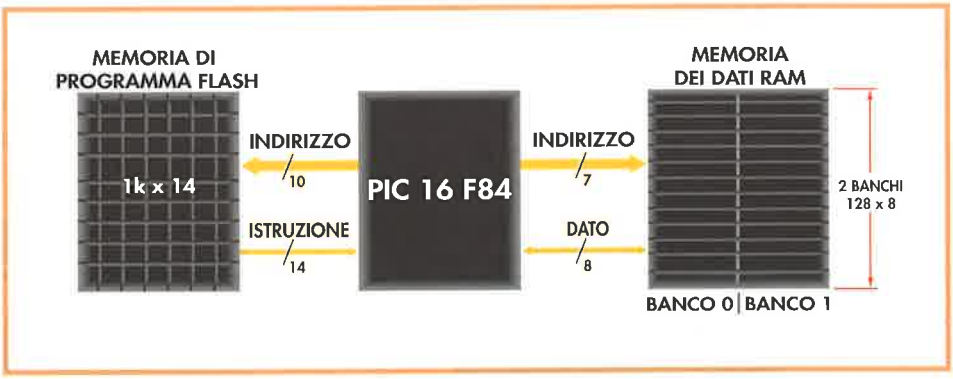
** Nel modo basso consumo si entra con lo sleep e termina con l'azione del reset esterno tramite MCLR#, o anche all'overflow del WDT.*

** Uscendo dal modo basso consumo si ricomincia eseguendo l'istruzione che segue quella di sleep.*

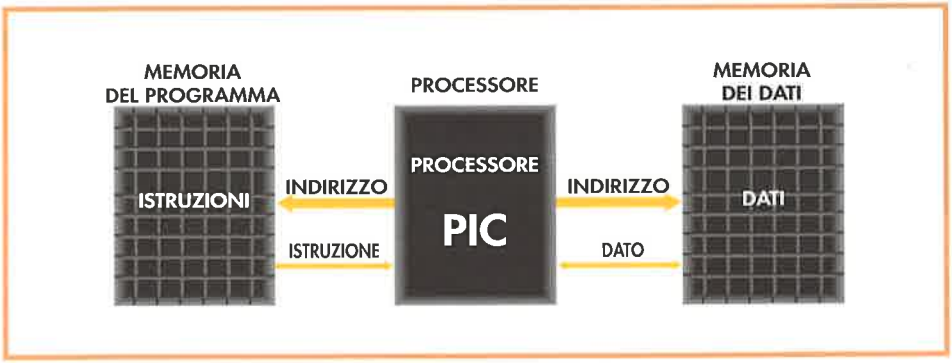
La memoria del programma

UN'ARCHITETTURA AVANZATA

L'architettura Harvard dei PIC, divide la memoria in due zone indipendenti, una destinata a contenere le istruzioni e l'altra i dati. Il primo vantaggio che deriva dall'architettura Harvard è che permette l'accesso simultaneo ad entrambe le memorie, in modo che il processore possa leggere un'istruzione di programma, e nel medesimo tempo scrivere un dato. Così, il processore attende a due istruzioni alla volta, conseguendo un parallelismo implicito chiamato segmentazione, che migliora notevolmente il rendimento del sistema; fatto che in pratica si traduce nella possibilità di eseguire un elevato numero di istruzioni al secondo. Se guardiamo il progetto costruttivo del PIC16F84, notiamo che la memoria di programma ha una capacità di 1 K byte da 14 bits ognuno, mentre la memoria RAM è costituita da due banchi da 128 byte ciascuno, lasciando gli altri banchi non implementati.



Un progetto esclusivo per il PIC16F84.

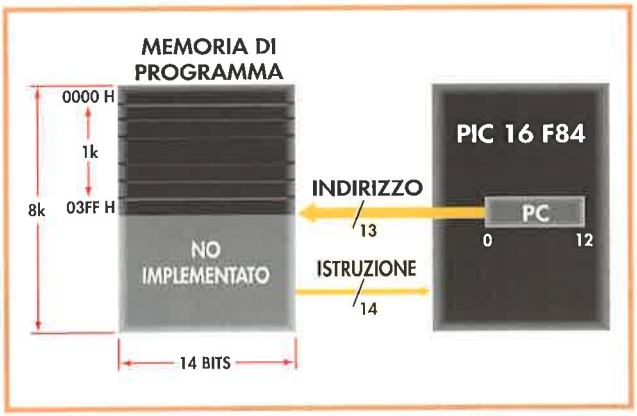


L'architettura Harvard del PIC rende indipendente la memoria del programma da quella dei dati.

diversi modelli della gamma media. Per questo il bus degli indirizzi, ovvero l'insieme di linee che trasportano il valore dell'indirizzo al quale si deve accedere nella memoria – che è generato dal Program Counter PC – non ha solo dieci linee, che sarebbero quelle sufficienti per gestire la memoria da 1 K, ma 13 con le quali è possibile indirizzare capacità

I BUS DEL PIC16F84

MICROCHIP non ha realizzato un progetto specifico per il PIC16F84, l'architettura utilizzata è la stessa per



Il contatore di programma e il bus degli indirizzi hanno 13 bits, con i quali si può indirizzare uno spazio di 8 K posizioni di memoria. Il PIC16F84 ha solo implementate le prime 1024 posizioni.

fino a 8 K posizioni di memoria del programma tipo FLASH. Nel caso del PIC16F84 è stato fisicamente implementato solo il primo K, sarebbe a dire, dall'indirizzo 0000 H al 03FF H.

INDIRIZZI SPECIALI

Nei PIC di gamma media abbiamo due indirizzi della memoria di programma che sono riservati per espletare due funzioni principali. Un indirizzo riservato è lo 0000 H, che è il primo della mappa. In esso è ubicato il Vettore di Reset o Reinizializzazione. Il suo compito è supportare la prima istruzione del programma principale, che è quella che esegue il processore dopo che è stato effettuato un reset.



I due indirizzi riservati nella memoria di programma. La 0000 H per il Vettore di Reset e la 0004 H per il Vettore di Interrupt.

Fra le altre cause, il reset si produce con l'accensione del microcontroller nel momento in cui si applica la tensione di alimentazione, il programma che esegue il microcontroller inizia all'indirizzo 0000 H.

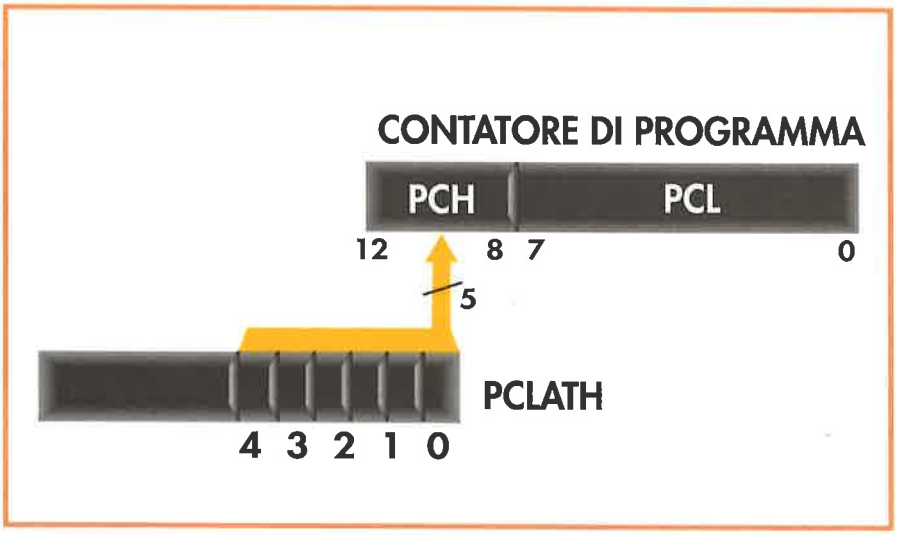
L'altro indirizzo riservato è lo 0004 H, in cui alloggia il Vettore di Interrupt. Se nel programma è previsto un interrupt, la routine che lo riguarda ha la sua prima istruzione all'indirizzo 0004 H.

Il fatto che il programma principale cominci all'indirizzo 0000 H e la routine di interrupt al 0004 H non significa che le restanti istruzioni continuino sequenzialmente, altrimenti il programma principale intaccherebbe la routine di interrupt che inizia quattro posizioni dopo. Normalmente all'indirizzo del vettore si colloca un'istruzione di salto incondizionato al vero indirizzo dove inizia il programma. Per esempio, se il programma principale – con l'obiettivo di salvare il Vettore di Interrupt – si decide di farlo iniziare all'indirizzo 0005 H, all'indirizzo 0000 H, viene collocato un GOTO alla direzione 0005 H, come si può vedere dal seguente programma:

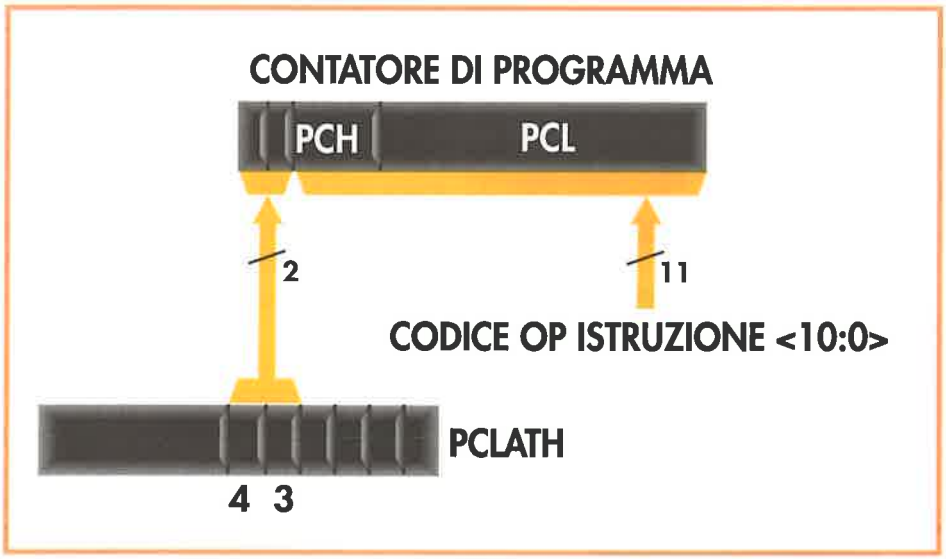
```
LIST    p=16f84   ;identifica il modello del PIC
RADIX   HEX       ;I valori si rappresentano in esadecimale
ORG     0         ;Il programma inizia all'indirizzo 0000 H
Goto    inizio    ;Salto all'istruzione con etichetta inizio
ORG     5         ; Istruzione successiva all'indirizzo 0005 H
Inizio  bsf       STATO, 5 ; Istruzione con etichetta inizio
```

IL CONTATORE DI PROGRAMMA

È un registro da 13 bits che contiene gli indirizzi dell'istruzione successiva che deve eseguire il processore. Dato che i registri della RAM sono di 8 bits, il valore del PC si implementa sopra due di essi. Il registro PCL dell'area SFR della RAM contiene gli 8 bits meno significativi del PC e occupa la posizione 2 dei due banchi. I 5



Gli 8 bits meno significativi del PIC risiedono nel registro PCL e i 5 più significativi nel PCLATH.



Nell'istruzione di salto il PC viene caricato con gli 11 bits meno significativi del codice OP e i bits <4:3> del registro PCLATH.

bits più significativi del PC risiedono nei 5 bits meno significativi del registro PCLATH, che occupa l'indirizzo 0A H nei due banchi della memoria dei dati. Il registro PCL si può leggere e scrivere direttamente, mentre il PCH è accessibile in scrittura attraverso i 5 bits meno significativi di PCLATH. In quelle istruzioni in cui la destinazione è il PC, gli 8 bits meno significativi si caricano dalle linee del bus dei dati interno al processore. I 5 bits più significativi, invece, si ricevono tramite i bits <4:0> del registro PCLATH.

Nell'istruzione di salto GOTO o CALL l'indirizzo del salto che si carica nel PC arriva dagli 11 bits meno significativi del codice OP dell'istruzione, e dai bits <4:3> del PCLATH.

STRUTTURA DELLO STACK

Lo stack è un insieme di registri che contengono le informazioni in una forma particolare, dato che, sia il carico che lo scarico è speciale. Nel caso del PIC16F84 lo stack è formato da 8 registri da 13 bits ognuno, ed è dotato di una struttura di carico e scarico tipo LIFO, acronimo inglese che significa "ultimo ad entrare primo ad uscire". Quando si carica un'informazione nello stack, si introduce sempre al livello 1.

Se precedentemente avevamo un'informazione al livello 1, questa viene spostata al livello 2 e quella del



Caricando lo stack al livello 1 l'informazione di ogni livello trasla al successivo.

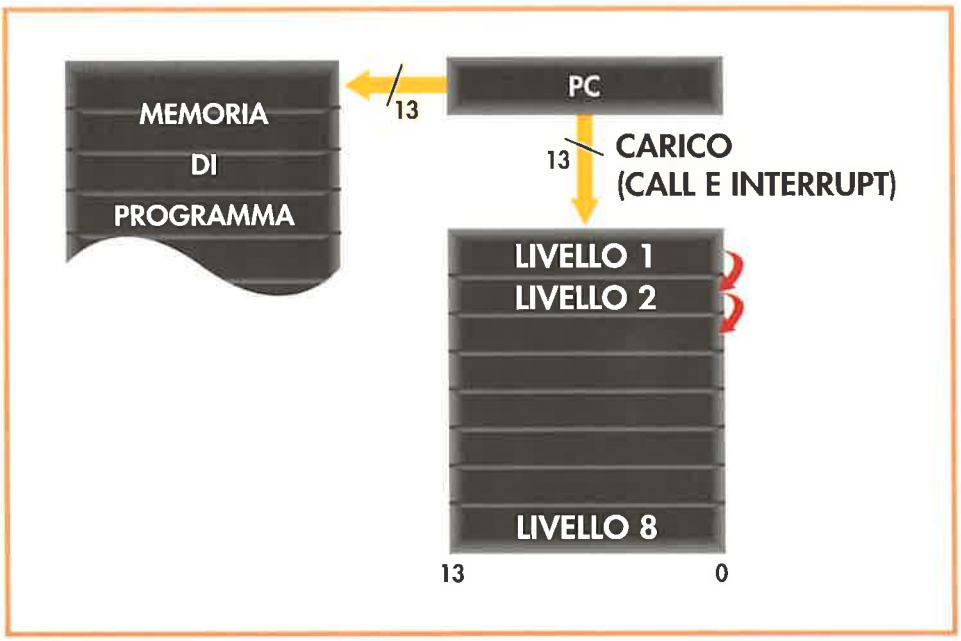


Anche lo scarico dello stack si realizza al livello 1, spostando l'informazione al livello precedente.

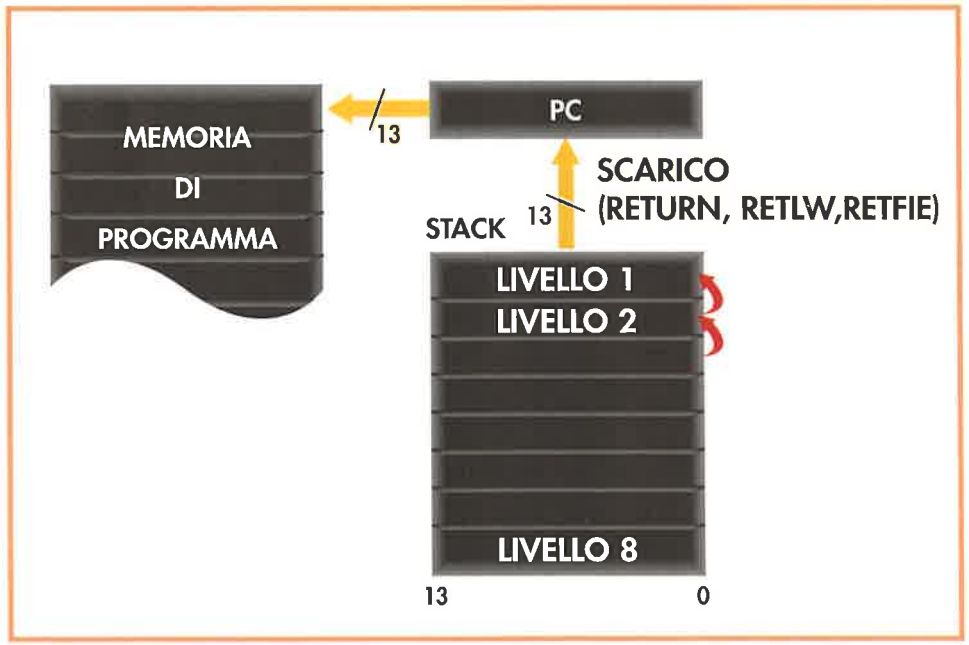
livello 2 al livello 3 e così di seguito; in altre parole, quando si carica l'informazione al livello 1, quello che c'era memorizzato in ogni livello si sposta al seguente livello.

Se lo stack è pieno e contiene informazioni in tutti gli otto livelli, il carico di un nuovo dato produce lo spostamento dell'informazione di ogni livello al seguente, quindi si perderà quella che esisteva al livello 8.

Per recuperare le informazioni dallo stack, lo spostamento delle informazioni si realizza al rovescio; esce quella che sta occupando il livello 1, e in esso viene caricata quella del livello 2. Le informazioni si spostano al livello precedente e il dato che esce dal livello 1 sarà l'ultimo che è stato inserito.



L'istruzione CALL o l'interrupt, provocano il caricamento automatico del contenuto del PIC nello stack.



Le istruzioni di RITORNO della subroutine o dell'interrupt scaricano immediatamente il contenuto del livello 1 dello stack sopra il PC.

LO STACK E IL PC

Lo stack del PIC funziona automaticamente, ed è direttamente relazionato con il contenuto del PC. Lo stack si carica sul livello 1 con il contenuto del PC. Quando lo stack scarica l'informazione del livello 1, la passa al PC. Ogni volta che si esegue l'istruzione CALL o che si produce un interrupt, è necessario salvare il contenuto del PC per sapere successivamente ritornare al punto dove avevamo lasciato il programma principale.

L'esecuzione di un'istruzione CALL o di un interrupt, produce lo spostamento automatico del contenuto del PC al livello 1 dello stack.

L'ultima istruzione di una subroutine o di una routine d'interrupt è di RITORNO, quando questa è eseguita produce lo scarico automatico del livello 1 dello stack sopra il PC, restituendo il flusso di controllo al programma principale, nel punto in cui era stato abbandonato.

Il programmatore deve tenere conto che non esistono segnalazioni che indicano quando lo stack è pieno, ed è sua responsabilità evitare il debordamento e la perdita di informazioni.

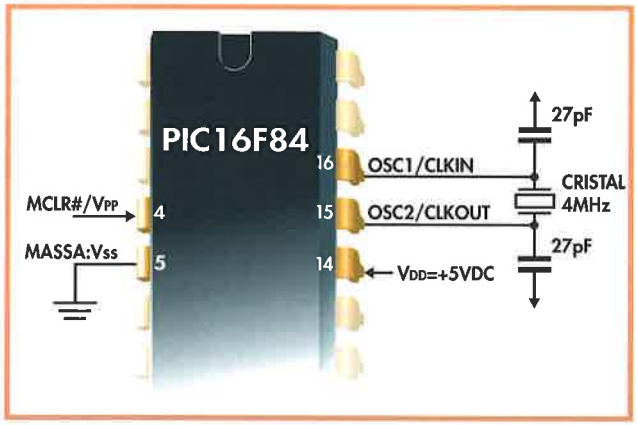
Reset e circuiteria fissa

IL MINIMO PER POTER LAVORARE

Per entrare in funzione il PIC16F84 ha bisogno di tre risorse:

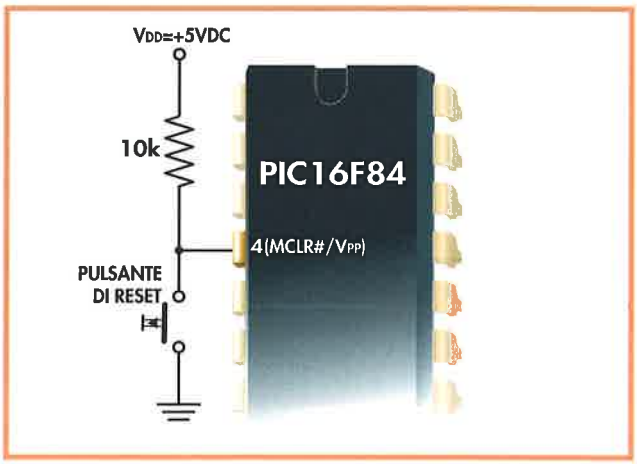
- Tensione di alimentazione.
- Frequenza di riferimento.
- Reinizializzazione o Reset.

La tensione di alimentazione è indispensabile per il funzionamento di tutti i componenti elettronici, e il suo valore può essere compreso fra +2 e +6 VDC, essendo



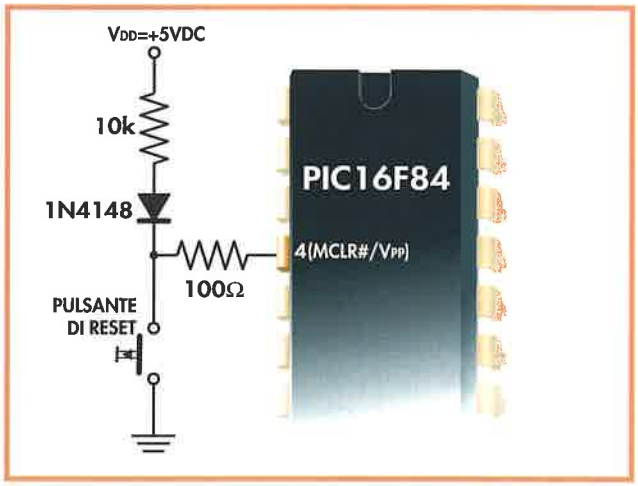
Questi sono i 5 pin del PIC16F84 che supportano le tre funzioni principali: alimentazione, frequenza di riferimento e Reset.

+5 V la tensione tipica, compatibile con il resto della circuiteria del sistema. Questa tensione si applica ai piedini 15 e 5, rispettivamente positivo e massa. Il consumo di corrente dipende dalla frequenza di funzionamento e dalla tensione di alimentazione, che non supera 10 mA. La frequenza di riferimento serve per stabilizzare e fissare la frequenza dell'oscillatore interno, che determina quella di lavoro del processore. Si ottiene da un cristallo di quarzo collegato ai piedini 16 (OSC1/CLKIN) e 15 (OSC2/CLKOUT). Nei nostri sviluppi utilizzeremo un cristallo di quarzo a 4 MHz, per formare un oscillatore tipo XT. In ognuno di questi piedini si monta un condensatore di pochi picofarad, per disaccoppiare le frequenze spurie.



Quando si preme il pulsante di Reset il piedino MCLR# passa a livello logico basso.

Infine, un microcontroller deve disporre di diverse forme di inizializzazione del programma di lavoro. Per provocare un Reset esterno si utilizza il piedino 4 (MCLR#/Vpp), che ha anche la funzione di introdurre la tensione Vpp da 12 a 14 VDC di cui il microcontroller necessita durante il processo di scrittura del programma. Nella figura si mostrano i 5 piedini del PIC16F84 che supportano le tre risorse principali.



Il diodo impedisce che circoli corrente dall'interno del microcontroller al positivo.

MICROCONTROLLER

Reinizializzazione o Reset

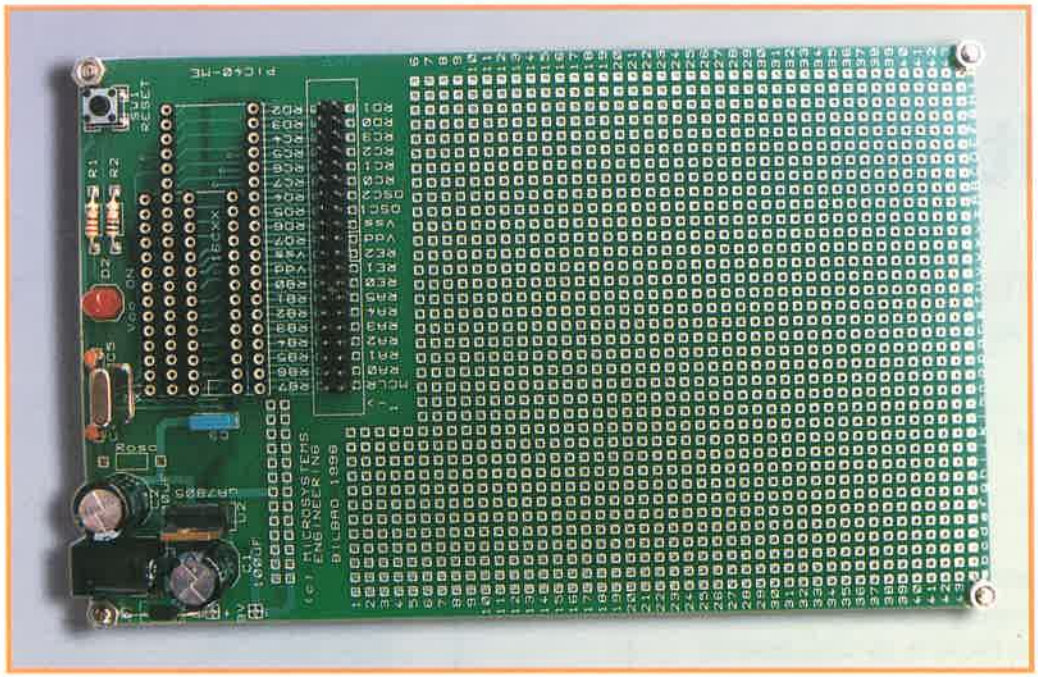


Immagine di una scheda di prototipi commercializzata da Microsystem Engineering..

LE SCHEDE PER I PROTOTIPI

Dato che quando si impiega un PIC16F84 nel progetto di un'applicazione, è sempre necessaria una circuiteria esterna, e varieranno solo i collegamenti delle linee di I/O con le periferiche che dovremo controllare esistono sul mercato schede di circuiti stampati che dispongono di uno zoccolo per il PIC, dei collegamenti per l'alimentazione, del cristallo di quarzo e del Reset, e una zona libera dove possono essere montate le periferiche che si collegheranno alle linee di I/O.

Nella figura della pagina successiva si può vedere lo schema elettrico della scheda per i prototipi che è molto utile per la presentazione e la messa a punto dei prototipi e dei progetti, data la facilità d'uso, la rapidità e l'economia che comporta.

IL CIRCUITO DI RESET

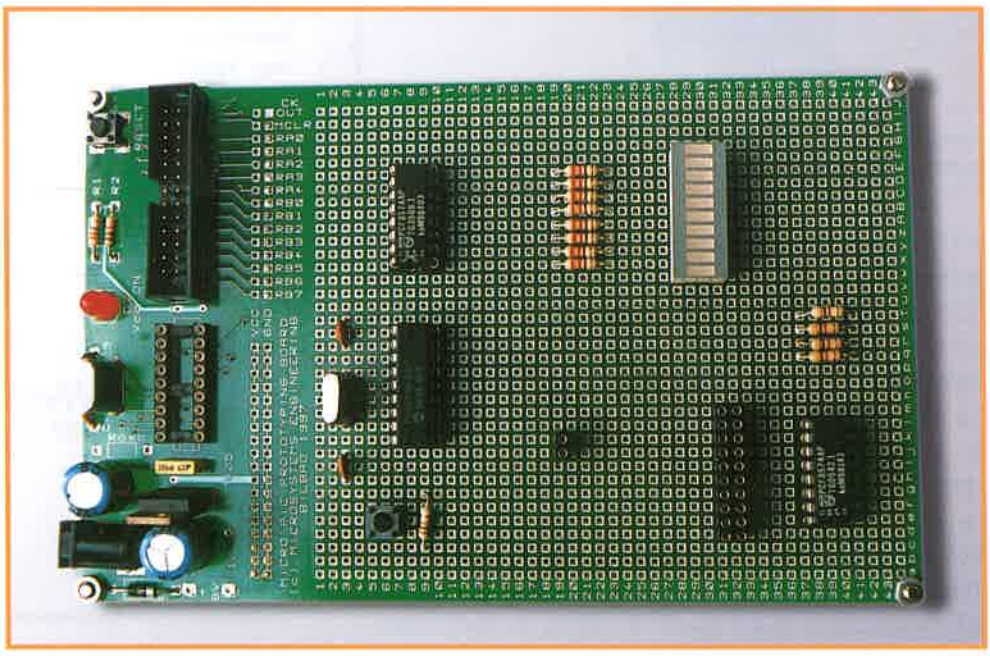
Quando si applica un livello basso per un tempo sufficiente al piedino MCLR#, si produce una reinizializzazione che comporta l'esecuzione di due azioni importanti:

1°. La messa a zero del Program Counter, che così passa ad indirizzare la prima istruzione del programma.

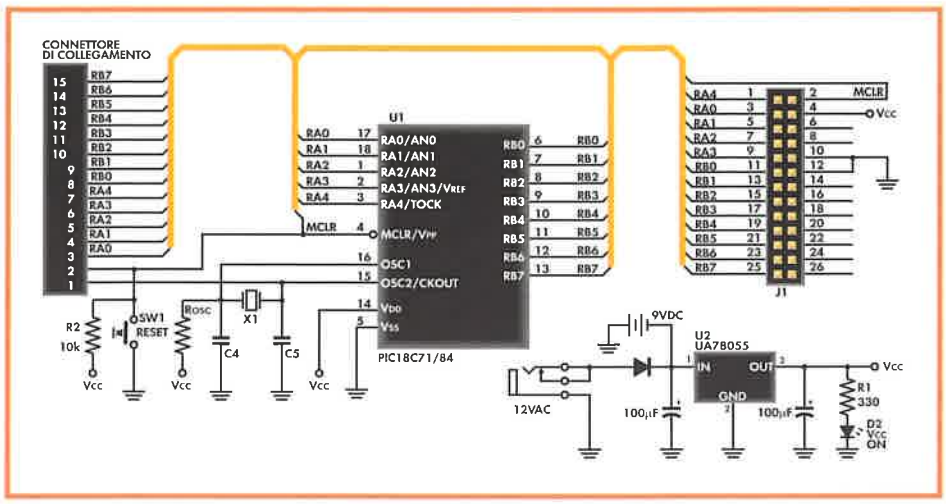
2°. Pone in uno stato predeterminato la maggior parte dei bit dei registri di controllo del processore.

Nella figura della pagina precedente si mostrava un semplice schema a base di un pulsante, che provoca il Reset esterno.

Un altro schema comune per il pulsante di Reset include una resistenza di assorbimento da 10 K, un diodo che impedisce la circolazione di corrente al piedino 4 e un'altra piccola resistenza da 100 Ω come si vede nella figura.



Scheda dei prototipi con un'applicazione che controlla diverse periferiche.



Schema elettrico della scheda dei prototipi

succedere sia nel modo funzionamento normale del PIC e sia nel suo stato di riposo.

In qualunque delle cinque possibili cause del Reset, il PC si carica con zero, eccetto quando il WDT va in overflow con il PIC in stato di riposo; in questo caso il PC si incrementa di una unità per eseguire l'istruzione seguente a quella che lo ha mandato nello stato di basso consumo. Nella tabella sono riportati i valori che prendono i bit del registro SFR dopo ognuna delle possibili cause di Reset.

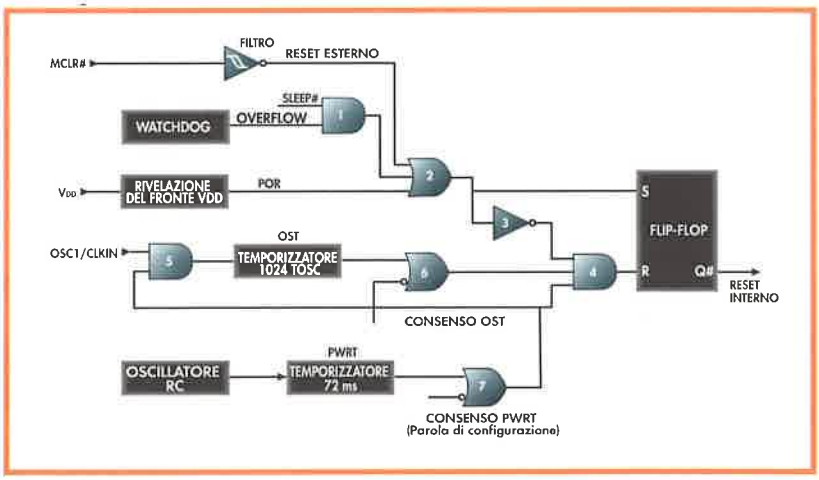
CAUSE CHE PROVOCANO IL RESET

Il Reset mette a zero il PC e colloca in uno stato predeterminato la maggior parte dei bit dei registri di controllo. Esistono tre cause principali che originano i Reset:

1°. Connessione dell'alimentazione (POR: Power On Reset). Si produce al passaggio della tensione di alimentazione da 1,2 a 1,7 V.

2°. Attivazione del piedino MCLR#. Questo può succedere sia mentre il PIC funziona normalmente, sia che si trovi in stato di riposo.

3°. Overflow del Watchdog (WDT). Può



Schema elettrico di controllo interno dell'attivazione del Reset.

REGISTRO	INDIRIZZI	CONNESSIONE DELLA ALIMENTAZIONE	OVERFLOW WATCHDOG MODO NORMALE	OVERFLOW WATCHDOG MODO RIPOSO	MCLR# MODO NORMALE	MCLR# MODO RIPOSO
W	....	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
INDIR	00 h	....	....	....	....	....
RTCC	01 h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PC	02 h	0000 h	0000 h	PC + 1	0000 h	0000 h
STATUS	03 h	0001 1xxx	0000 1uuu	uuu0 0uuu	000u uuuu	0001 0uuu
FSR	04 h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PORT A	05 h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
PORT B	06 h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TRIS A	85 h	---1 1111	---1 1111	---u uuuu	---1 1111	---1 1111
TRIS B	86 h	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
OPTION	81 h	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
EEDATA	08 h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
EEADR	09 h	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
EECON 1	88 h	---0 0000	---0 ?000	---u uuuu	---0 ?000	---0 ?000
EECON 2	89 h	....	....	....	....	....
PCLATH	0A h	---0 0000	---0 0000	---u uuuu	---0 0000	---0 0000
INTCON	0B h	0000 000x	0000 000u	uuuu uuuu	0000 000u	0000 0000

U = Non cambia x = Indeterminato .... = Non utilizzato ? = Dipende da altre condizioni.

Tabella con i valori che assumono i bit dei registri SFR durante i possibili Reset.

SCHEMA ELETTRICO DEL RESET

La figura mostra lo schema elettrico che controlla l'attivazione del Reset. Sono messi in evidenza i temporizzatori OST e PWRT.

Il temporizzatore PWRT (Power-up-Timer) aggiunge 72 millisecondi che mantengono il microcontroller nella situazione di Reset, per dare tempo alla tensione di alimentazione di stabilizzarsi. Per il suo funzionamento dobbiamo mettere a basso

livello il bit di accesso PWRTE#, che risiede nella parola di configurazione. L'altro temporizzatore, OST, fornisce un ritardo di 1.024 Tosc e ha il compito di assicurare che l'oscillatore principale del processore stabilizzi la sua frequenza. OST inizia a funzionare quando termina la temporizzazione di PWRT, dovuta al collegamento della porta OR7 con la AND5. L'ingresso Set del flip-flop di uscita si attiva:

- All'attivarsi del piedino MCLR#.
- Quando va in overflow il WDT.
- Se si rileva un fronte di salita nel piedino VDD (POR).

L'ingresso Reset del flip-flop si attiva quando, tramite la AND4, arriva un livello alto; questo richiede che agli ingressi della porta ci siano livelli alti, e quindi significa che:

- Non ci sono richieste di Reset e l'uscita di OR2 ha livello basso.
- Se PWRT è attivo è terminato il ritardo di 72 ms.
- Concluso il ritardo di PWRT è terminato il ritardo di OST.

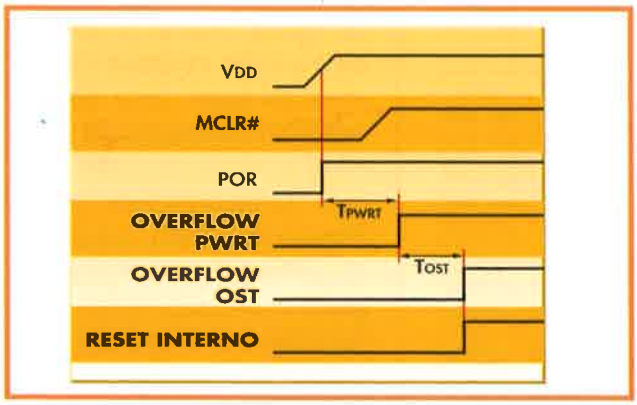
Nel cronogramma della figura possiamo vedere gli stati che assumono i diversi segnali che sono coinvolti nel Reset, quando questo viene prodotto alla rilevazione di un fronte di salita sull'ingresso della tensione di alimentazione.

DETERMINAZIONE DELL'ORIGINE DEL RESET

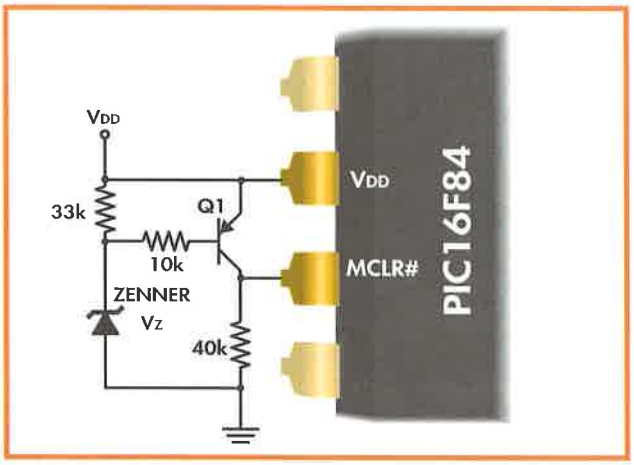
Dati i differenti effetti che provoca il Reset a seconda della causa che lo ha originato, è molto importante conoscere la causa stessa. Per questo si utilizzano due bit del registro di STATO o STATUS, che occupano l'indirizzo 3 dei due banchi della RAM. Si tratta del bit TO# (Timer Out), che si attiva con l'overflow del WDT, e del bit PD# (Power Down) che si attiva quando il PIC è in stato di riposo. La tabella in alto decodifica gli stati di questi due bit.

TO#	PD#	CONDIZIONI DI RESET
1	1	POR (RESET PER CONNESSIONE VDD)
0	1	OVERFLOW WDT IN FUNZIONAMENTO NORMALE
0	0	OVERFLOW WDT IN STATO DI RIPOSO
1	1	ATTIVAZIONE MCLR# IN FUNZIONAMENTO NORMALE
1	0	ATTIVAZIONE MCLR# IN RIPOSO

I bit TO# e PD# del Registro di Stato determinano la causa che ha originato il Reset.



Cronogramma dei segnali che partecipano al Reset all'arrivo di un fronte di salita nella tensione di alimentazione.



Quando la tensione di alimentazione scende al di sotto del valore di tensione del diodo zener, Vz = 0,7 V, si produce un Reset per l'attivazione del piedino MCLR#.

RESET PER BUCO DI TENSIONE

A volte è molto importante reinizializzare il processore quando si produce un buco di tensione e/o la tensione scende sotto un valore determinato, senza arrivare a zero, risalendo subito dopo.

Ci sono modelli del PIC che incorporano nel silicio i circuiti necessari per provocare il Reset a fronte di un buco di tensione. Questo non è il caso del PIC16F84, quindi per ottenere ciò bisogna costruire un circuito ausiliario come mostrato nella figura.

Necessità di memorizzare dati non volatili

Spesso, nelle applicazioni più comuni con i microcontroller, è necessario poter disporre di informazioni di tipo permanente. In determinati momenti vengono introdotti nel sistema alcuni dati che devono essere mantenuti anche dopo aver scollegato l'alimentazione.

Supponiamo un sistema di controllo di accesso ad un luogo, che controlla il relé la cui attivazione chiude i contatti che permettono l'apertura della porta. Le principali periferiche saranno una tastiera, un display di visualizzazione dei messaggi, e un relé di uscita. Ogni utilizzatore introduce la sua chiave segreta mediante la tastiera e i messaggi che appaiono nel video guidano le operazioni del medesimo.

Nell'applicazione descritta, ogni utilizzatore ha un suo codice, e l'insieme di questi deve essere memorizzato in modo permanente, per essere utilizzato quando è necessario. Inoltre sarà possibile cambiarli in qualunque momento.

Per memorizzare questi tipi di dati è necessaria una memoria non volatile, perchè il contenuto possa essere scritto e cancellato; però deve essere mantenuto anche quando viene scollegata l'alimentazione. Con l'obiettivo di supportare questo tipo di dati e applicazioni, il PIC16F84 dispone di una memoria EEPROM da 64 bytes.



Terminali programmabili della OMRON, che hanno funzione di interfaccia uomo-macchina.

CARATTERISTICHE TECNICHE DELLA MEMORIA EEPROM

Le memorie EEPROM nascono per superare gli inconvenienti delle EPROM, che si potevano scrivere e poi cancellare sottoponendole per alcuni minuti ai raggi ultravioletti, per cui disponevano di un cristallo sulla superficie. La scrittura della EPROM è di tipo elettrico, però per cancellarla, bisogna estrarla dallo zoccolo e posizionarla sul "cancellatore" a raggi ultravioletti. Questo processo è lungo e complicato.

Nelle memorie EEPROM sia la scrittura sia la cancellazione delle informazioni, sono realizzate mediante segnali elettrici, per cui non è necessario estrarre il circuito integrato dallo zoccolo.

Nella costruzione della cella MOSFET delle memorie EEPROM, si introduce una sottile lamina di ossido sopra il drain, che permette la cancellazione elettrica. Quando si applica una elevata tensione fra il gate e il source del MOSFET si introduce una carica nella barriera di potenziale, che rimane memorizzata anche se viene tolta l'alimentazione. Per distruggere questa carica basta applicare la stessa tensione inversa.

Nella EEPROM si possono scrivere e cancellare singole celle. Le operazioni di scrittura e cancellazione di solito durano pochi millisecondi. L'inconveniente è che la superficie che occupa questa cella è grande e per questo la densità di integrazione è bassa e il costo elevato.

La memoria FLASH utilizza la stessa tecnologia però riducendo lo spessore della lamina di ossido, il che permette un funzionamento simile alle EEPROM, con maggiore densità di integrazione, minor costo e maggior velocità di accesso.

Per questo motivo useremo la memoria di programma FLASH nel PIC16F84.

MICROCONTROLLER

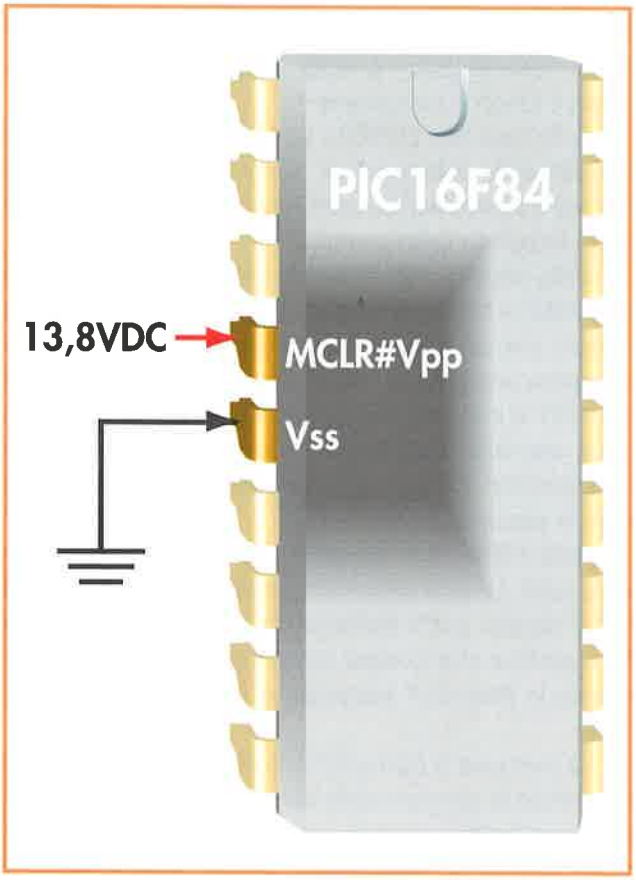
Memoria dei dati EEPROM [I]

Altre caratteristiche interessanti per la scrittura dei dati nella EEPROM del PIC16F84 sono:

- Supporta 1.000.000 di cicli di scrittura e cancellazione.
- Mantiene le informazioni più di 40 anni.

LA EEPROM DEL PIC16F84

Il microcontroller che usiamo come cervello di Monty, dispone di una zona di memoria dei dati implementata con tecnologia EEPROM, con una capacità di 64 posizioni da 8 bit cadauna. Il suo principale inconveniente è che la durata del ciclo di scrittura o cancellazione di una cella è molto lento, comparato con la velocità del processore, infatti sono necessari 10 millisecondi, tempo in cui un PIC16F84 lavorando a 10 Mhz, esegue più istruzioni delle 1024 che può contenere nella sua memoria di programma. Inoltre questo tempo è critico e bisogna aspettare che termini completamente ogni singola operazione per iniziarne una nuova.

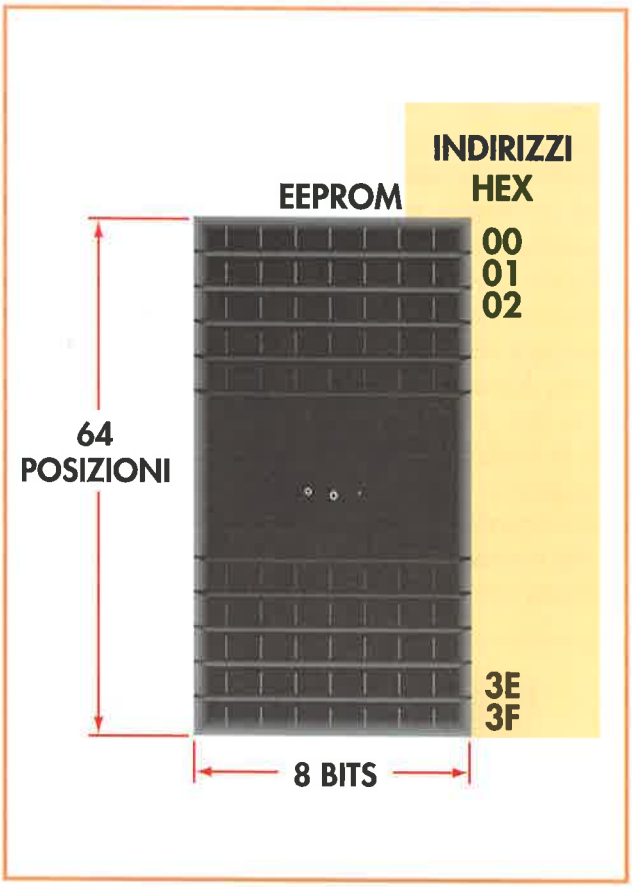


La tensione di 13,8 VDC, necessaria per la scrittura/cancellazione di un indirizzo della EEPROM, si applica tramite il pin 4 del PIC16F84.

Per le operazioni di scrittura e cancellazione, la memoria EEPROM necessita di una tensione Vpp alta, dell'ordine di 13,8 VDC, che viene introdotta tramite il pin MCLR#/Vpp, come si può vedere dalla figura.

LA MAPPA DELLA MEMORIA EEPROM

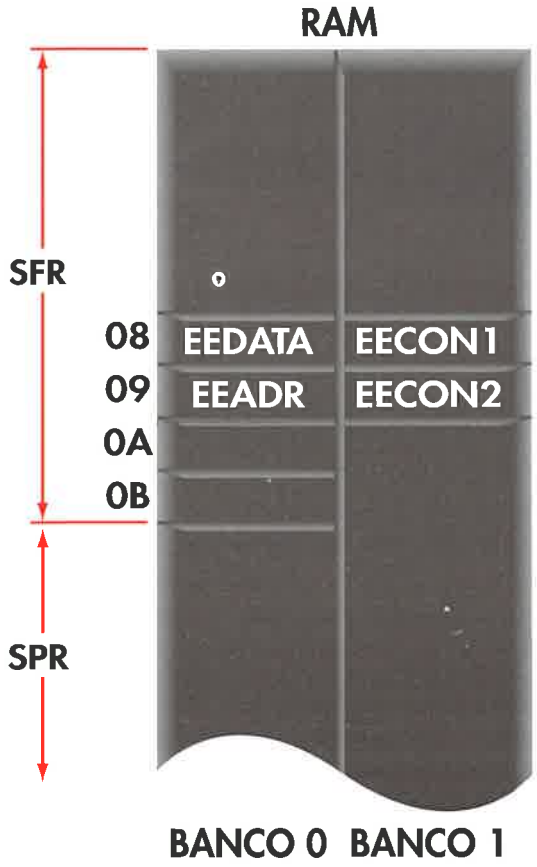
Anche se la EEPROM è una parte della memoria dei dati, i suoi indirizzi non sono mappati insieme ai registri SFR e GPR della RAM, che è volatile. Occupa uno spazio di memoria indipendente, così come mostrato nella figura.



Lo spazio nella memoria EEPROM è indipendente.

Per gestire lo spazio proprio della EEPROM bisogna utilizzare due registri di controllo dell'area SFR. Si tratta dei registri EEADR ed EEDATA. Il registro EEADR occupa la posizione 9 del banco 0 della RAM, mentre quello EEDATA si trova ubicato all'indirizzo 8 del banco 0, come riportato graficamente.

Nella posizione uguale a EEDATA, nel banco 1, si trova il registro di controllo della EEPROM, EECON1. In modo simile il registro di controllo EECON2 occupa una posizione correlata a EEADR, nel banco 1.



Ubicazione dei registri EEADR ed EEDATA, che si utilizzano nella gestione dell'area di memoria EEPROM; sono riportati anche i registri di controllo EECON1 ed EECON2.

GESTIONE BASE DELLA EEPROM

I cicli di lettura e scrittura della EEPROM sono realizzati sugli indirizzi di un byte.

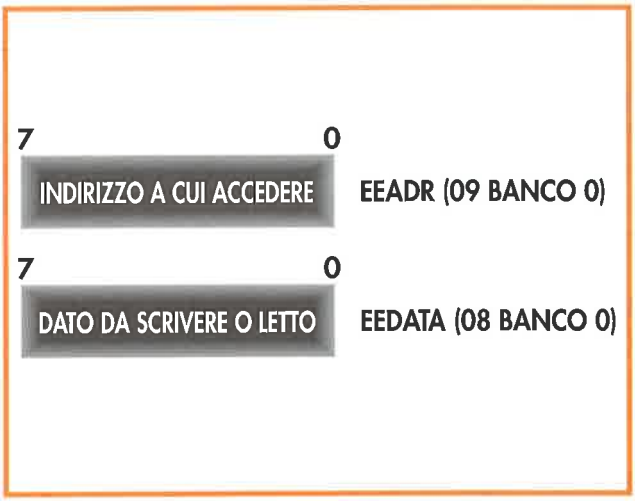
Il ciclo di scrittura comporta la precedente cancellazione in automatico delle informazioni che conteneva la posizione a cui si è fatto accesso. Il tempo di ciclo di scrittura è controllato da un temporizzatore integrato nel chip.

Il problema maggiore è che questo tempo non solo è lungo, 10 millisecondi come massimo, ma è variabile a seconda della temperatura e della tensione applicata.

Dato che questo tempo è molto critico, la fine delle operazioni è controllata da un flag che può essere complementato con un interrupt alla fine della scrittura di un indirizzo.

Per accedere ad una posizione della EEPROM dobbiamo caricare nel registro EEADR l'indirizzo della medesima, mentre in EEDATA si carica il dato da scrivere, o nel caso della lettura, si deposita il dato letto.

Dato che i registri EEADR ed EEDATA sono di 8 bit, i due bit più significativi del registro EEADR valgono sempre zero, perché il valore maggiore che si può caricare è 3 FH.



Nel registro EEADR si carica l'indirizzo a cui accedere, e in EEDATA, il dato da scrivere o quello letto.

IL REGISTRO DI CONTROLLO EECON1

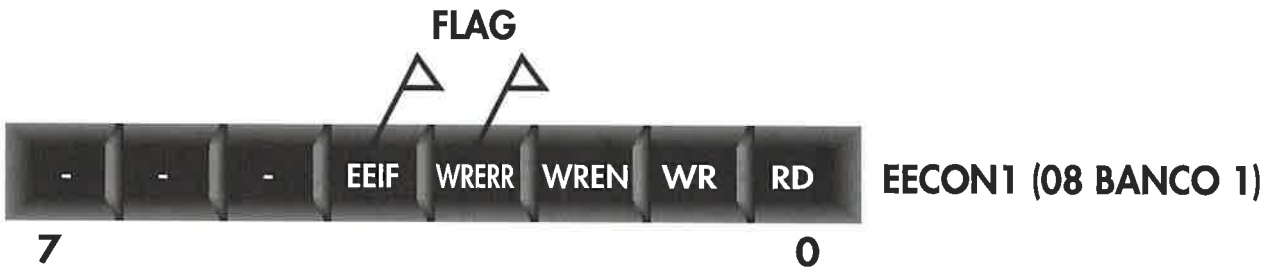
All'indirizzo 08 relativo al banco 1 della zona SFR della RAM è situato il registro EECON1 i cui bit hanno il compito di controllare le operazione della EEPROM.

I tre bit meno significativi di questo registro non hanno alcuna funzione.

Di seguito descriveremo il compito di ognuno dei bit del registro EECON1:

RD: Lettura

Quando questo bit è messo a 1, si realizza un'operazione di lettura all'indirizzo contenuto nel registro EEADR. Il contenuto di questo indirizzo viene caricato nel registro EEDATA. Terminata l'operazione di lettura RD torna automaticamente a 0.



Struttura e nomenclatura dei bit del registro EECON1

WR: Scrittura
Mettendo a 1 questo bit si realizza un'operazione di scrittura dell'indirizzo indicato da EEADR, con il dato caricato in EEDATA. Completato il ciclo di scrittura il WR torna a zero automaticamente.

WREN: Consenso alla scrittura
Questo bit realizza una funzione di sicurezza, se è a 1 permette la scrittura, se è a 0 no, indipendentemente dal valore del bit WR.

WRERR: Flag di errore di scrittura
Viene posto automaticamente a 1 quando un'operazione di scrittura termina prematuramente.

Segnala che si è verificato un errore di scrittura, attivandosi quando si interrompe l'operazione di scrittura, a causa di un reset prodotto da MCLR# o per overflow del WDT. Dopo il Reset l'utilizzatore deve verificare questo bit, e se vale 1, ripetere l'operazione di scrittura.

EEIF: Flag di termine dell'operazione di scrittura
Data l'importanza dell'operazione di scrittura, e il rischio di compromettere il sistema se non termina correttamente, questo bit è dedicato a segnalare il termine dell'operazione, prendendo il valore 1.

In seguito dovremo porre questo bit a zero tramite software, altrimenti resterà a 1.

IL REGISTRO DI CONTROLLO EECON2

In realtà questo registro non è fisicamente implementato nella RAM.

È utilizzato come un dispositivo di sicurezza, per convalidare l'operazione di scrittura della EEPROM, ed evitare disturbi che potrebbero prodursi durante il lungo intervallo di tempo prodotto dalla durata di questa operazione.

Se viene letto tutti i suoi bit valgono 0.



Il registro EECON2 funziona solo come dispositivo di sicurezza nell'operazione di scrittura.

Registri dedicati alla EEPROM

Per gestire in lettura e scrittura le 64 posizioni della EEPROM del PIC16F84, si utilizzano quattro registri dell'area SFR:

- **EEADR:** Contiene l'indirizzo della EEPROM a cui accedere.
- **EEDATA:** Contiene il dato da scrivere o il dato letto.
- **EECON1:** Controlla il funzionamento dell'operazione da compiere.
- **EECON2:** Si utilizza come dispositivo di sicurezza durante la scrittura.



Registri incaricati di realizzare le operazioni con la memoria EEPROM.

SVILUPPO DI UN'OPERAZIONE DI LETTURA

La lettura di una posizione della EEPROM inizia caricando in EEADR l'indirizzo a cui accedere. Successivamente si pone a 1 il bit RD del registro EECON1. Nel ciclo seguente il dato letto è disponibile in EEDATA. Questo dato permane nel registro sino al realizzo di un'altra operazione di lettura o scrittura nella EEPROM.

Creare un programma per leggere l'indirizzo 0C H della EEPROM e depositarlo nel registro W.

```
status    equ    0x03
eeadr     equ    0x09
eedata    equ    0x08
eecon1    equ    0x08

bcf       status,rp0    ;rp0 = 0 per accedere al banco 0
movlw     0x0c          ;w = 0c
movwf     eeadr         ;w = eeadr
bsf       status,rp0    ;rp0 = 1 per accedere al banco 1
bsf       eecon1,rd     ;rd = 1 per leggere
bcf       status,rp0    ;rp0 = 0 per accedere al banco 0
movf      eedata,w      ;eedata = w
```

SVILUPPO DI UN'OPERAZIONE DI SCRITTURA

Per scrivere un dato in un indirizzo della EEPROM, dobbiamo seguire una sequenza di istruzioni in cui partecipa in modo speciale il registro EECON2. Questo registro, che in realtà non è implementato fisicamente, assume solo una funzione di sicurezza, caricandosi con i valori 55 H e AA H in sequenza. Questo evita interferenze durante la lunga operazione di scrittura, che a volte dura fino a 10 ms.

Il ciclo inizia caricando in EEADR l'indirizzo della cella e in EEDATA il byte da scrivere. Se decidiamo che durante il processo di scrittura non si possono accettare interrupt, si deve mettere il bit di enable global GIE = 0. In questo modo si evita che durante l'operazione di scrittura, la CPU venga sviata verso un'altra routine al verificarsi dell'interrupt. In seguito dobbiamo autorizzare la scrittura, mettendo il bit WREN = 1 nel registro EECON1; poi dobbiamo includere nel programma una sequenza di istruzioni che caricano in primo luogo il valore 55 H in EECON2, in seguito AA H nel medesimo registro. Terminata questa sequenza, si pone il bit WR = 1 in EECON1 e si inizia la lunga operazione di scrittura.

MICROCONTROLLER

Memoria dei dati EEPROM [II]

Quando termina la scrittura, nel registro EECON1 il bit WR sarà uguale a zero, e il flag EEIF a 1. Questo flag si pone a 1 automaticamente, al termine del ciclo di scrittura e, leggendolo, possiamo quindi sapere se il processo è terminato. Il flag EEIF dovrà essere poi resettato dal programma.

Dopo ogni ciclo di scrittura è raccomandabile verificare l'indirizzo della EEPROM, per vedere se è stata scritta correttamente.

Creare un programma per scrivere il valore 0xff all'indirizzo della EEPROM 0x32.
Si suppone che prima del programma si siano specificate le direttive che assegnano ai registri di controllo impiegati gli indirizzi corrispondenti (eeadr, eedata, status, intcon, eecon1, eecon2).

```
bcf     status,rp0      ;rp0 = 0 per accedere al banco 0
movlw   0x32            ;w = 0x32
movwf   eeadr           ;w = eeadr
movlw   0xff            ;w = 0xff
movwf   eedata          ;w = eedata
bsf     status,rp0      ;rp0 = 1 per accedere al banco 1
bcf     intcon,gie      ;gie = 0 per proibire interrupt
bsf     eecon1,wren     ;wren = 1 per permettere la scrittura
                                nella EEPROM
```

Sequenza di scrittura

```
movlw   0x55            ;w  = 0x55
movwf   eecon2          ;w  = eecon2
movlw   0xa             ;w  = 0xaa
movwf   eecon2          ;w  = eecon2
bsf     eecon1,wr       ;wr = 1 inizia il ciclo di scrittura
```

I REGISTRI DELLA EEPROM DOPO IL RESET

I bit dei quattro registri che partecipano alle operazioni della EEPROM, prendono dei valori predeterminati dopo il Reset. Questi valori possono essere diversi se il Reset è dovuto al collegamento dell'alimentazione (POR) o ad altra causa.

Dopo un Reset provocato da POR (Power On Reset) i bit dei registri di EEDATA e EEADR prendono un valore casuale (x) però se il Reset è dovuto ad un'altra causa, i bit non cambiano valore (u). Il registro EECON2 non è implementato fisicamente, e i suoi bit hanno sempre valore zero in caso di lettura. Infine i bit RD, WR, WREN, ed EEIF del registro EECON1, prendono un valore casuale dopo un POR oppure un valore che dipende dalla condizione precedente (q) se il Reset dipende da altra causa. Nella tabella riportata qui sopra si riassumono i valori di questi registri dopo i vari tipi di Reset.

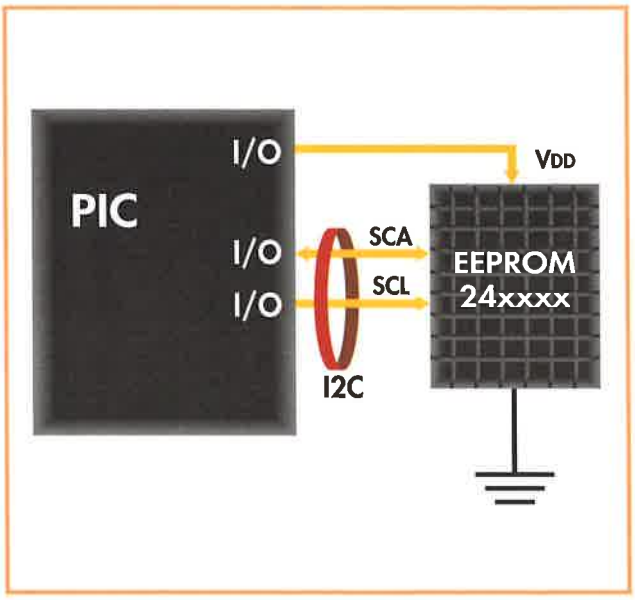
	Indirizzo	Registro	Valore dopo Reset POR	Valore dopo altri Reset
BANCO 0	08	EEDATA	xxx xxx	uuuu uuuu
	09	EEADR	xxx xxx	uuuu uuuu
BANCO 1	08	EECON1	---0 x000	---0 q000
	09	EECON2	---- ----	---- ----

Tabella con i valori che assumono i registri che gestiscono la EEPROM dopo un Reset.
(x: sconosciuto, u: non cambia, q: dipende dalla condizione).

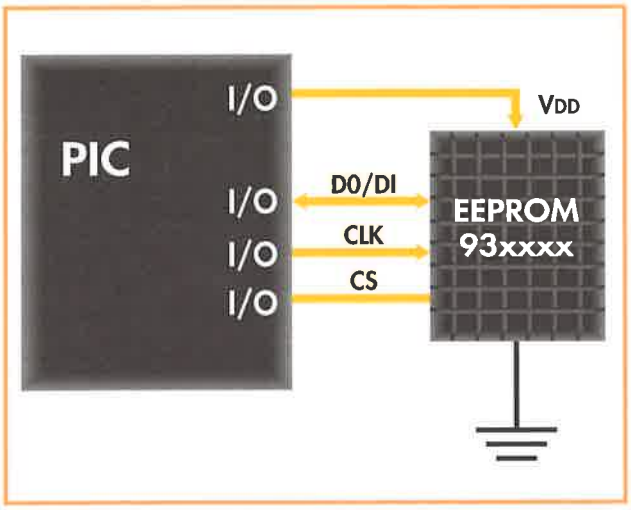
LE MEMORIE EEPROM COMMERCIALI

I principali fabbricanti di circuiti integrati e di memorie, offrono EEPROM con differenti capacità. Microchip dispone di tre grandi famiglie di memorie EEPROM:

- Famiglia 93xxxx: Memorie EEPROM di lettura, sequenza serie, a tre linee.
- Famiglia 24xxxx: Memorie EEPROM per bus I2C a due linee.
- Famiglia 25xxxx: Memorie EEPROM per bus serie SPI a tre linee.



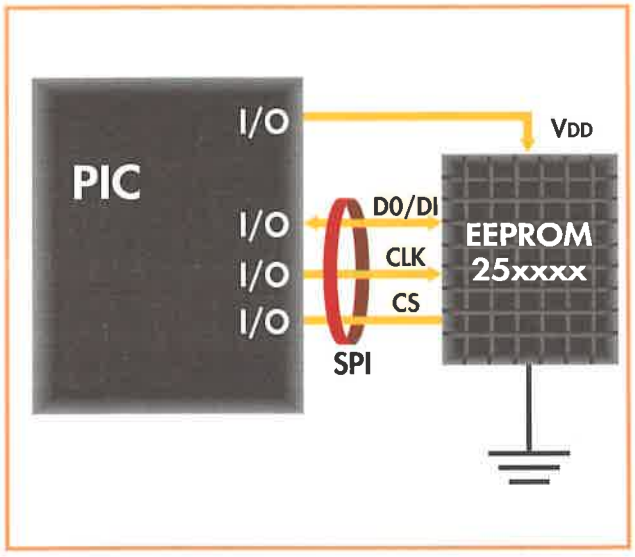
Adattamento delle memorie EEPROM 24xxxx per bus I2C alle linee di I/O di un microcontroller.



Collegamenti delle memorie EEPROM 93xxxx alle linee di I/O del microcontroller.

In queste famiglie è sufficiente riservare due o tre fili per realizzare l'interscambio d'informazioni con la memoria EEPROM. Questo aspetto è molto importante quando si collega uno di questi dispositivi ad un microcontroller con un numero limitato di linee di I/O, per ampliare la sua memoria. Il trasferimento si realizza in serie, seguendo uno dei protocolli abituali dell'industria, come illustrato nella figura.

Nella famiglia 24xxxx predisposta per il bus I2C esistono dispositivi come la 24C00, che ha una capacità di soli 128 bits, raggruppati in bytes, con una velocità di scrittura di 4 ms e una frequenza di 400 KHz, ed

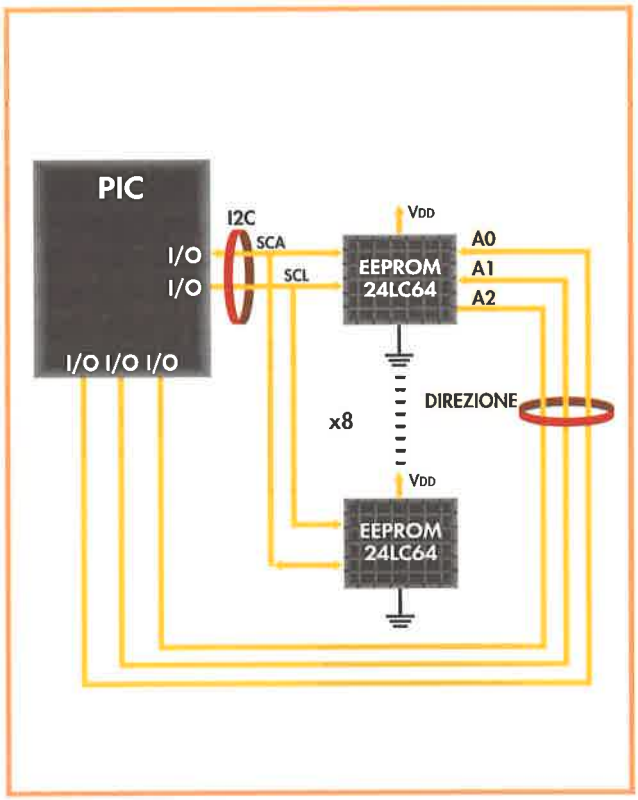


Collegamenti della memoria EEPROM 25xxxx per il bus SPI, ad un microcontroller PIC.

altre come la 24LC64SC che dispongono di 64 K bits con un tempo di scrittura di 5 ms e 400 KHz. Entrambi i modelli supportano 1.000.000 di cicli di scrittura/cancellatura.

Della famiglia 93xxxx ricordiamo il modello 93AA86, con una capacità di 16 Kbits, un tempo di scrittura di 5 ms e una frequenza di 2 MHz. La tensione di alimentazione può essere compresa fra 1,8 e 5,5 VDC, e il numero di cicli di scrittura/cancellazione garantito è di 1.000.000.

Della famiglia 25xxxx, progettata per supportare il protocollo SPI, spicca il modello 25LC640, con 64 Kbits di capacità, una velocità di scrittura di 5 ms e 2 MHz di



Collegamenti di otto memorie EEPROM 24LC64 per ottenere una capacità di 512 K bits.

frequenza, tensione di lavoro fra 2,5 e 5,5 VDC con 1.000.000 di cicli di scrittura/cancellazione.

In alcuni casi sono necessari grandi quantità di EEPROM, e per queste situazioni si possono configurare insiemi di dispositivi controllati in modo indipendente da diverse linee di indirizzamento, che utilizzano gli stessi I/O del microcontroller. Nella figura è mostrato un circuito formato da otto memorie da 64 Kbits ognuna, per ottenere un totale di 512 Kbits.

MICROCONTROLLER

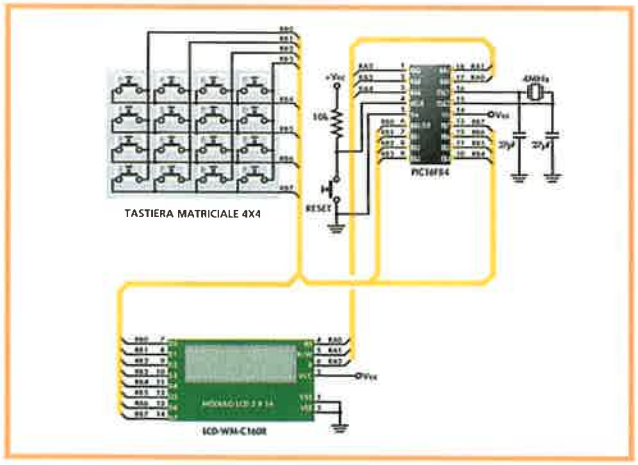
Memoria dei dati EEPROM [II]

UN'APPLICAZIONE DELLA EEPROM IN UN GIOCO DI INTRATTENIMENTO: LINGO NUMERICO

Per poter dare un'idea delle immense possibilità di utilizzo che hanno le memorie EEPROM, presentiamo un caso pratico. Abbiamo scelto un gioco sviluppato con un PIC16F84 che controlla una tastiera e un video LCD.

Il Lingo Numerico è un gioco nel quale bisogna indovinare un numero di 5 cifre generato in modo casuale dal gioco stesso. Con questo scopo, il microcontroller comunica al giocatore, tramite l'LCD i digit che sono stati indovinati, e che occupano la posizione corretta, e quelli che sono stati indovinati, ma che non occupano la posizione corretta.

Esistono due livelli di difficoltà che si differenziano per il tempo totale di cui si dispone per indovinare il numero giusto. Se si indovina il numero prima che il tempo sia scaduto, sul display LCD appare la scritta "HAI VINTO", seguita dal punteggio, che equivale al numero dei secondi che mancano allo scadere del tempo massimo.



Schema elettrico dell'applicazione.

In questa applicazione si utilizza la EEPROM molto frequentemente, ad esempio serve per memorizzare sia il nome di ciascun giocatore sia il nome del giocatore che ha preso più punti, però noi abbiamo pensato di utilizzarla per scrivere in ogni partita il numero casuale generato dal gioco.

Lo schema elettrico è molto semplice. Le 8 linee della Porta B supportano la tastiera matriciale 4 x 4, RB0-RB3 le file e RB4-RB7 le colonne; inoltre queste linee servono, in forma multiplexata, per trasferire informazioni al display LCD tramite i suoi ingressi D0-D7. RA0, RA1 e RA2 gestiscono i segnali di controllo del LCD.

Routine per calcolare il numero casuale.

Ogni volta che inizia una partita, il Lingo Numerico genera un numero casuale di cinque cifre che deve essere temporaneamente memorizzato negli indirizzi della EEPROM che vanno da 10H a 14H. Ogni cifra viene memorizzata con il suo codice ASCII, e la routine fa uso del Timer 0, situato all'indirizzo 00H della EEPROM, e delle variabili DATO_A e DATO_B della memoria RAM.

La formula per calcolare il numero casuale utilizza il valore del Timer 0, che cambia continuamente, inoltre utilizza un "numero base" che dipende dal valore precedentemente calcolato. Questa formula è la seguente:

Numero casuale = (Numero base * 173 + 49 + Timer 0) %10

Di seguito riportiamo la routine corrispondente in cui sono presenti molti commenti.

```
RAND    clrf    EEADR     ;Puntatore all'indirizzo del "numero base" nella EEPROM
        movlw   h'10'
        movwf   FSR
        bsf     STATUS,5  ;Selezioniamo il Banco 1
        bsf     EEDATA,RD ;Comando di lettura
        bcf     STATUS,5  ;Selezioniamo il banco 0
        movf    EEDATA,W
        movwf   DATO_A    ;Portiamo il numero base in DATO_A
R_BUC   movlw   h'15'
        subwf   FSR,W     ;Verifichiamo se è stato terminato il numero
        btfsc   STATUS,FZ ;E' terminato?
        goto    R-FIN     ;Si
        movlw   d'173'
        movwf   DATO_B    ;Carichiamo DATO_B con il valore 173
R_MUL   addwf   DATO_A,W  ;Somma di DATO_A con W
        decfsz  DATO_B,F  ;E' stato sommato 173 volte DATO_A?
        goto    R_MUL     ;No
        addlw   d'49'     ;Si; Somma 149 al numero
        addwf   TMR0,W    ;Somma il valore del Timer 0
        movwf   DATO_A    ;Si stabilisce il nuovo numero base
R_DIG   addlw*  d'10'     ;Si somma 10 a W sino a che non genera riporto
        btfss   STATUS,FC ;Questo fa si che il numero sia da 0 a 9
        goto    R_DIG
        addlw   h'30'     ;Si converte in ASCII il numero
        movwf   INDF      ;Si controlla la posizione all'interno del numero
        incf*   FSR,F     ;Si incrementa la posizione del numero
        goto    R_BUC     ;Si ripete sino a 5 volte
                          ;Scrittura del nuovo numero base nella EEPROM
R_FIN   movf    DATO_A,W  ;Si carica il nuovo numero base in W
        Movwf   EEDATA    ;Portiamo il nuovo numero nella EEPROM
        Call    GRAB_E    ;Routine che scrive l'indirizzo 00H della EEPROM
        Return
```

Impariamo ad utilizzare le EEPROM

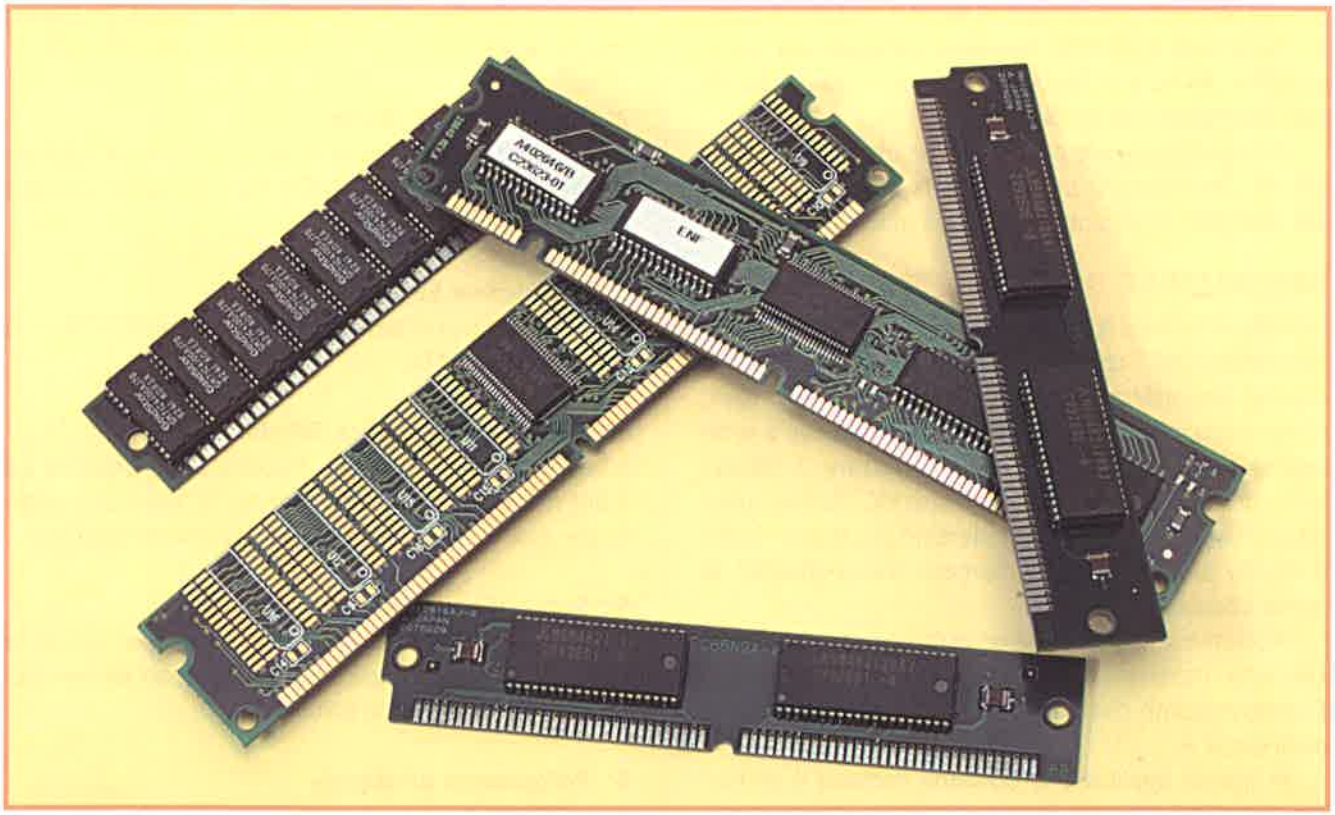
I computer e i sistemi informatici attualmente utilizzano le memorie come elemento base per supportare le loro applicazioni. L'importanza di questi dispositivi è enorme, e imparare il loro utilizzo è fondamentale a tutti i livelli. Fino ad ora abbiamo studiato la struttura e i registri che controllano la memoria EEPROM del PIC16F84. Si tratta di fare un passo in più, e imparare ad utilizzarla mediante i programmi adeguati.

Anche se la programmazione del PIC è ancora in fase di apprendimento, dato che abbiamo solo iniziato ad utilizzare le istruzioni principali del microcontroller, crediamo che sia di notevole aiuto, una volta studiato a fondo un dispositivo, provare ad assimilare le routines che lo devono governare. Non importa se non capiremo tutte le parti del progetto, quello che intendiamo fare è mostrare pezzi dedicati a leggere e scrivere la EEPROM. Concentriamoci su queste parti.

IL SISTEMA DI SVILUPPO MICRO'PIC TRAINER

Illustreremo ora un progetto che ha una vasta applicazione negli stabilimenti, per gestire gli ingressi del personale. Lo svilupperemo sopra un potente ed economico sistema di sviluppo chiamato "Micro'PIC Trainer", che è stato progettato e commercializzato dalla ditta di Ingegneria di Microsistemas Programados S.L.

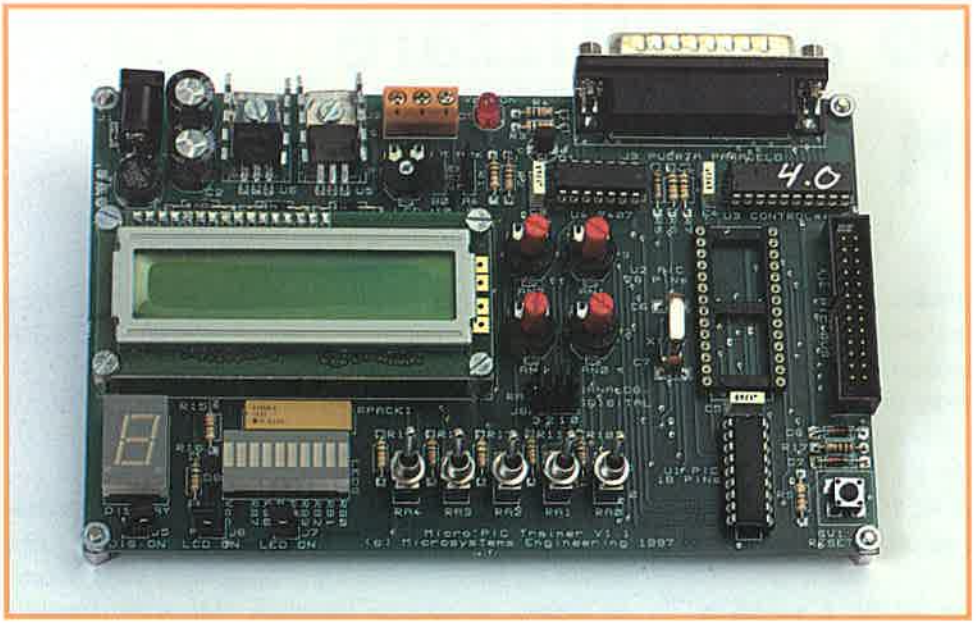
Un sistema di sviluppo è uno strumento fondamentale per la realizzazione di progetti, che in questo caso è stato dedicato ai microcontroller PIC della gamma



Le memorie sono un chiaro esempio di circuiti ad alta scala di integrazione.

MICROCONTROLLER

Utilizzo e programmazione della EEPROM



Fotografia del sistema di sviluppo Micro'PIC Trainer.

media. Sostanzialmente è formato da tre elementi: 1) uno scrittore/cancellatore di programma del PIC; 2) un insieme di periferiche utilizzate nelle applicazioni con microcontroller, 3) circuiteria ausiliaria che comprende alimentatore, circuito di Reset e cristallo di quarzo. La scheda Micro'PIC Trainer è collegata alla porta parallela del PC e si controlla facilmente dal computer con il software che la correda.

Data l'efficacia e le possibilità che offre questo strumento, dedicheremo più avanti una sezione completa per descriverne le caratteristiche e l'utilizzo.

DESCRIZIONE DEL PROGETTO

Si vuole emulare il funzionamento delle note macchine "ELIMINACODE" che sono usate in molti esercizi commerciali per ordinare le file dei clienti. Nel nostro caso, come video informativo utilizzeremo un display a sette segmenti. Sarebbe stato meglio utilizzare il display LCD, di cui dispone anche il Micro'PIC Trainer, però data la sua complessità, ne parleremo più avanti. Sopra il display sarà mostrato il numero corrispondente al turno attuale.

Abbiamo a disposizione un pulsante che, una volta premuto, incrementa il turno e prepara al seguente. Questo pulsante chiamato RA0 è collegato alla linea 0 della porta A.

In questa applicazione dovremo risolvere il problema che sorge nel caso di una mancanza momentanea di energia elettrica, o allo scollegamento involontario della macchina. Con questa finalità si impiega la memoria EEPROM del PIC16F84, che memorizza l'ultimo numero visualizzato, in modo che in caso di mancanza di alimentazione, permette di riprendere correttamente il conteggio con il ritorno dell'energia. Come punto di partenza consideriamo che, quando si inizia la prima volta, la macchina inizia il conteggio da 0.

STRUTTURA DEL PROGRAMMA

Nella sequenza delle istruzioni, prima di iniziare il programma principale, esistono una serie di blocchi e di soubroutines che descriviamo di seguito:

1°. Intestazione
Include le direttive per l'assembler, la libreria dei registri interni del PIC e le relazioni delle etichette utilizzate.

2°. Subroutine EE_Write
È una ruotine incaricata di scrivere un byte caricato inizialmente nel registro EEDATA nella posizione puntata da EEADR.

3°. Subroutine EE_Read
Legge il dato puntato dal contenuto di EEADR e lo deposita in EEDATA.

4°. Tabella di decodifica Tabella
È una ruotine che converte il codice BCD, presente sui 4 bits meno significativi del registro W, nel codice adatto per essere visualizzato sul display a sette segmenti.

5°. Subroutine di ritardo Delay_20_ms
Realizza una temporizzazione di 20 ms per eliminare l'effetto di rimbalzo tipico dei componenti elettromeccanici, come il pulsante RA0.

6°. Programma principale
In questo programma si gestiscono i blocchi precedenti per risolvere l'applicazione.

IL PROGRAMMA CON I COMMENTI

```
                List        p=16F84         ;Tipo di processore
                Include     "P16F84.INC"    ;Libreria con definizioni dei registri interni

Contatore       equ         0x0c            ;Definisce la variabile contatore

                org         0x00            ;Vector Reset
                goto        inizio
                org         0x05            Salto per salvare il vettore di interrupt

;EE_WRITE: È una routine che scrive un byte nella memoria EEPROM. L'indirizzo sarà contenuto nel registro
;EEADR e il dato è stato precedentemente introdotto in EEDATA

EE_Write        bsf         STATUS,RP0      ;Seleziona il banco1
                bsf         EECON1,WREN     ;Permesso di scrittura nella EEPROM
                movlw       b'01010101'
                movwf       EECON2
                movlw       b'101010101'
                movwf       EECON2          ;Sequenza di sicurezza, si carica 55 e aa in EECON2
                bsf         EECON1,WR       ;Ordine di inizio della scrittura
                bcf         EECON1,WREN     ;Impedisce nuove scritture
Wait            btfss       EECON1,EEIF     ;Testa il flag per vedere se la scrittura è terminata
                goto        Wait            ;La scrittura non è terminata
                bcf         EECON1,EEIF     ;La scrittura è terminata e si resetta il flag
                bcf         STATUS,RP0      ;selezione del banco 0
                return

;EE_Read: Routine che legge un byte nella EEPROM. Nel registro EEADR si carica l'indirizzo dove andare a
;leggere, e in EEDATA si carica il dato letto.

EE_Read         bsf         STATUS,RP0      ;Seleziona il banco1
                bsf         EECON1,RD       ;Ordine di lettura
                bcf         STATUS,RP0      ;Selezione del banco 0
                return

;Tabella: questa routine converte il codice BCD presente nei 4 bit meno significativi del registro W nel formato
;adatto ad essere rappresentato nel display a 7 segmenti. Il codice di 7 segmenti, ritorna anche nel registro W
;con l'istruzione retlw.

Tabella         addwf       PCL,F           ;Disposizione sulla tabella in base al PCL e W
                retlw       b'00111111'     ;Digit 0
                retlw       b'00000110'     ;Digit 1
                retlw       b'01011011'     ;Digit 2
                retlw       b'01001111'     ;Digit 3
                retlw       b'01100110'     ;Digit 4
                retlw       b'01101101'     ;Digit 5
                retlw       b'01111101'     ;Digit 6
                retlw       b'00000111'     ;Digit 7
                retlw       b'01111111'     ;Digit 8
                retlw       b'01100111'     ;Digit 9
```

Delay_20_ms: Si tratta di una routine che temporizza 20 ms per eliminare l'effetto rimbalzo dai dispositivi elettromeccanici. Se il PIC lavora a 20 MHz, il TMR0 si incrementa ogni microsecondo, quindi contando 20.000 microsecondi otterremo un tempo equivalente a 20 ms. Se scegliamo il valore di 128 per il divisore di frequenza, il TMR0 dovrà contare 156 impulsi che equivalgono a 0x9c in esadecimale, e dato che il TMR0 è ascendente, dovrà essere caricato con il suo complemento a 1, cioè 0x63 in esadecimale.

```
Delay_20_ms     bcf     INTCON,T0IF     ;Mette a zero il flag di overflow del TMR0
                movlw   0x63            ;Complemento a 1 di 156 (0x9c)
                movwf   TMR0            ;Carica il TMR0
Delay_20_ms_1   clrwdt                  ;Refresh del Watchdog
                btfss   INTCON,T0IF     ;Verifica se il TMR0 ha finito
                goto    Delay_20_ms_1   ;Non ha finito
                bcf     INTCON,T0IF     ;Il TMR0 ha finito, si resetta il flag
                return                  ;Inizio del programma principale
Inizio          clrf    PORTB           ;Cancella il latch di uscita della Porta B
                bsf     STATUS,RP0      ;Accesso al banco 1
                clrf    TRISB           ;Configura la Porta B come uscita
                movlw   b'00011111'     ;
                movwf   TRISA           ;Configura la Porta A come ingresso
                movlw   b'00000110'     ;
                movwf   OPTION_REG      ;Imposta il divisore di frequenza del TMR0 a 128
                bcf     STATUS,RP0      ;Seleziona il banco 0
                clrf    EEADR           ;Carica l'indirizzo 00 in EEADR
                call    EE_Read         ;Legge il byte della EEPROM
                movlw   0x09            ;
                subwf   EEDATA,W        ;
                btfsc   STATUS,C        ;È maggiore di 9?
                goto    Ini_0           ;È maggiore di 9 e si pone a 0 il contatore
                goto    Ini_1           ;Non è maggiore di 9
Ini_0           clrf    Contatore       ;Mette a zero il contatore
                goto    Loop            ;
Ini_1           movf    EEDATA,W        ;
                movwf   Contatore       ;Inizializza il contatore
Loop            movf    Contatore,W     ;
                call    Tabella         ;Converte il contatore in codice a 7 segmenti
                movwf   PORTB           ;Visualizza sul display
Wait_0          clrwdt                  ;Refresh del Watchdog
                btfss   PORTA,0         ;RA0 = 1?
                goto    Wait_0          ;Non è a 1, attendere
                call    Dealy_20_ms     ;Temporizzazione per eliminare il rimbalzo
Wait_1          clrwdt                  ;Refresh del Watchdog
                btfss   PORTA,0         ;RA0 = 0?
                goto    Wait_1          ;Non è a 0, attendere
                call    Dealy_20_ms     ;Eliminare il rimbalzo. C'è stato un impulso

                incf    Contatore,F     ;Incrementa il contatore
                movlw   .10             ;
                subwf   Contatore,W     ;
                btfsc   STATUS,Z        ;Il contatore è maggiore di 9?
                clrf    Contatore       ;Si è maggiore di 9. Si pone a 00
                movf    Contatore,W     ;
                movwf   EEDATA          ;
                call    EE_Write        ;Memorizza il nuovo valore del contatore nella EEPROM
                goto    Loop            ;
                end                     ;Fine del programma
```

Le risorse ausiliarie

Il PIC16F84 ha il soprannome di "fratello povero" della famiglia. Ha poco di tutto, ma nonostante questo, è molto più utilizzato dei suoi fratelli, che hanno più risorse. Ha avuto grande successo e la spiegazione arriva dalla sua semplicità, economia e media delle caratteristiche.

Prima di passare a studiare gli interrupt, così vitali nelle applicazioni del mondo industriale, abbiamo considerato molto interessante fare un piccolo ripasso delle risorse ausiliarie e complementari che circondano il nostro microcontroller che funziona come cervello di Monty. Queste risorse sono:

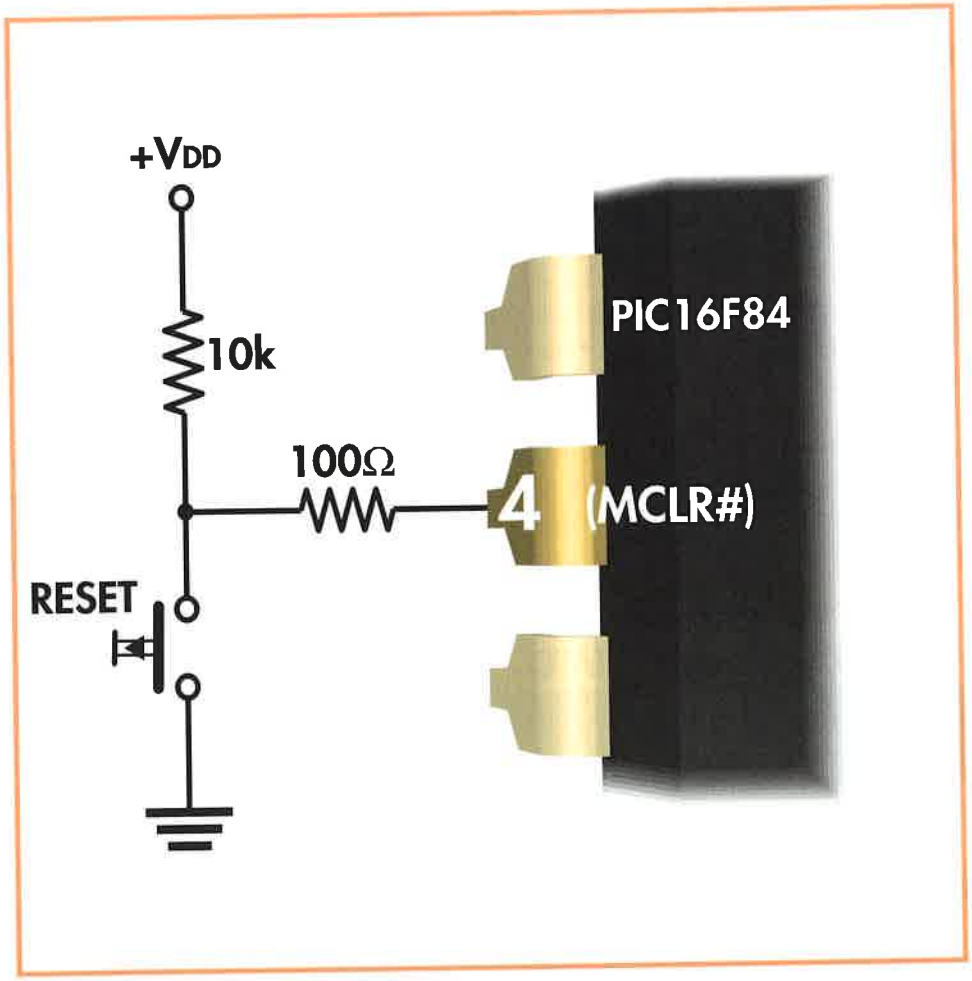
1B Reset. Che può prodursi per connessione dell'alimentazione.

(POR), per attivazione del piedino MCLR # e per overflow del WDT.

2B Il modo riposo di basso consumo.

3B La parola di configurazione.

4B Le parole di identificazione.



Circuito base per generare un Reset premendo un pulsante.

L'IMPORTANZA DEL RESET

Il Reset inizializza il sistema, e il processore inizia ad eseguire il programma dalla prima istruzione, partendo da uno stato noto di tutti i registri. È un avvenimento vitale a cui si arriva nelle seguenti situazioni:

1°. Quando si desidera provocarlo esternamente.

2°. Quando si collega l'alimentazione per la prima volta.

3°. Quando il programma si è "piantato" e vogliamo che ricominci dal principio.

4°. Quando trascorre un certo tempo in stato di riposo.

IL RESET ESTERNO

L'applicazione del livello logico basso al piedino 4 del PIC genera un Reset. Il circuito più semplice che permette la sua attuazione è mostrato nella figura.



Quando si produce un Reset per l'attivazione del pin MCLR#, il bit PD# del registro di Stato si pone a 0.



Valore che prendono i bit del registro di Stato quando si attiva MCLR# durante il normale funzionamento del microcontroller.

Premendo il pulsante e originando il Reset, può succedere che in quel momento il processore si trovi in stato di "riposo", oppure stia funzionando in modo normale, quindi eseguendo delle istruzioni. Nel primo caso il bit PD# del registro di Stato si pone a 0 e il bit TO# a 1.

Quando l'attivazione del l'MCLR# si origina col processore funzionante in modo normale i tre bit meno significativi del registro di Stato passano a 0 e i rimanenti non cambiano, così come è rappresentato nella figura.

RESET PER COLLEGAMENTO DELL'ALIMENTAZIONE (POR)

In questo caso la reinizializzazione del processore si produce quando si collega per la prima volta l'alimentazione. È chiamato Reset per POR (Power On Reset). Questo Reset è generato con un leggero ritardo rispetto al momento dell'applicazione dell'alimentazione, con il fine di permettere che la tensione dell'alimentazione stessa si stabilizzi al suo valore nominale. È un ritardo di 72 ms generato dal temporizzatore



Valori che prendono i bit del registro di Stato dopo un Reset per collegamento dell'alimentazione.

PWRT (Power-up Timer). Inoltre con l'obiettivo di ottenere la stabilizzazione della frequenza del funzionamento del sistema, si aggiunge un altro ritardo di 1.024 impulsi del segnale applicato in OSC1. Lo produce il temporizzatore OST (Oscillator Start-up Timer). Così quando si inizia l'esecuzione delle istruzioni, la frequenza di lavoro è stabile. Come si vede nella figura, dopo che si è prodotto questo tipo di Reset, i bit PD# e TO# restano disattivati, sarebbe a dire a 1, nel registro di Stato.

RESET PER OVERFLOW DEL WDT

Come ricorderemo il Watchdog (WDT) è un contatore che si incrementa al ritmo della frequenza dell'oscillatore principale, e quando va in overflow origina un Reset. Il WDT viene caricato con un valore, ed entra in funzione all'inizio del programma. Prima che vada in overflow, lo si deve resettare mettendolo a 0. Se il flusso di controllo entra in un ciclo infinito, o rimane in attesa di un segnale che non riceve, il Reset del WDT non arriva in tempo e l'overflow reinizzializza il processore.

L'overflow del Watchdog si può originare quando il processore funziona normalmente o quando si trova in "standby", in stato di riposo.

Il bit TO# del registro di Stato va a 0 quando il Reset è stato provocato dall'overflow del WDT. Nelle figure annesse possiamo apprezzare la differenza dei valori che prendono i bit del registro di Stato, in accordo con lo stato del processore quando il WDT va in overflow.

IL SONNO DEL PIC

Molte volte i sistemi basati su microcontroller devono rimanere inattivi per molto tempo, aspettando che qualcuno li faccia lavorare. Questo è il caso delle macchine distributrici di prodotti vari come caffè, bibite, sigarette, ecc. Durante lunghi periodi di tempo queste macchine non fanno nulla, fino a quando un cliente si avvicina, introduce una moneta e chiede qualcosa. Con lo scopo di ridurre al minimo il consumo di energia, si induce il microcontroller ad un sonno profondo, nel quale la sua attività è ridotta notevolmente. Il PIC che è in questo stato di riposo o di basso consumo, si suole dire che sta dormendo.



Così rimane il registro di Stato quando WDT va in overflow, mentre il processore funziona normalmente.



Registro di Stato dopo il debordamento del Watchdog quando il processore è in stato di "riposo".



Struttura interna della parola di configurazione.

Per produrre questo stato di relax, basta eseguire l'istruzione SLEEP. Pertanto si ottiene che l'oscillatore principale e le linee di I/O si congelano mantenendo il suo stato precedente.

Per togliere il processore dallo stato di riposo esistono due alternative: attivare esternamente il piedino MCLR#, o fare in modo che il Watchdog vada in overflow, se è rimasto attivo quando si entra nello stato di riposo, visto che non si eseguono istruzioni e non riceve refresh. Un'altra possibilità per togliere il processore dallo stato di riposo, consiste nell'annullare l'alimentazione per ricollegarla nuovamente, producendo un Reset POR.

LA PAROLA DI CONFIGURAZIONE

La parola di configurazione occupa una posizione speciale della memoria di programma FLASH, la direzione 2007 H, i cui 14 bit regolano le funzioni vitali del processore. L'organizzazione di questo indirizzo della memoria è presentato nella figura.

I 10 bit meno significativi, denominati tutti CP (Codice di Protezione), sono incaricati di proteggere l'accesso alla memoria dei codici. Se CP è 0 le istruzioni del programma non possono essere lette, al contrario, se CP è 1 il programma memorizzato nella FLASH si può leggere. Con i due bit meno significativi, FOSC0 e FOSC1, si determina il tipo di oscillatore che utilizza il processore, secondo la seguente codifica:

FOSC1	FOSC0	TIPO DI OSCILLATORE
1	1	RC
1	0	HS
0	1	XT
0	0	LP

Il bit WDTE abilita o disabilita il funzionamento del Watchdog. Se il WDTE è a 1, il Watchdog entra in funzione. In ultimo il bit PWRTE# abilita il temporizzatore PWRT, che ritarda di 72 ms il Reset per POR, per dare il tempo alla tensione di alimentazione di stabilizzarsi. Se PWRET# è a 0, lavora come temporizzatore.



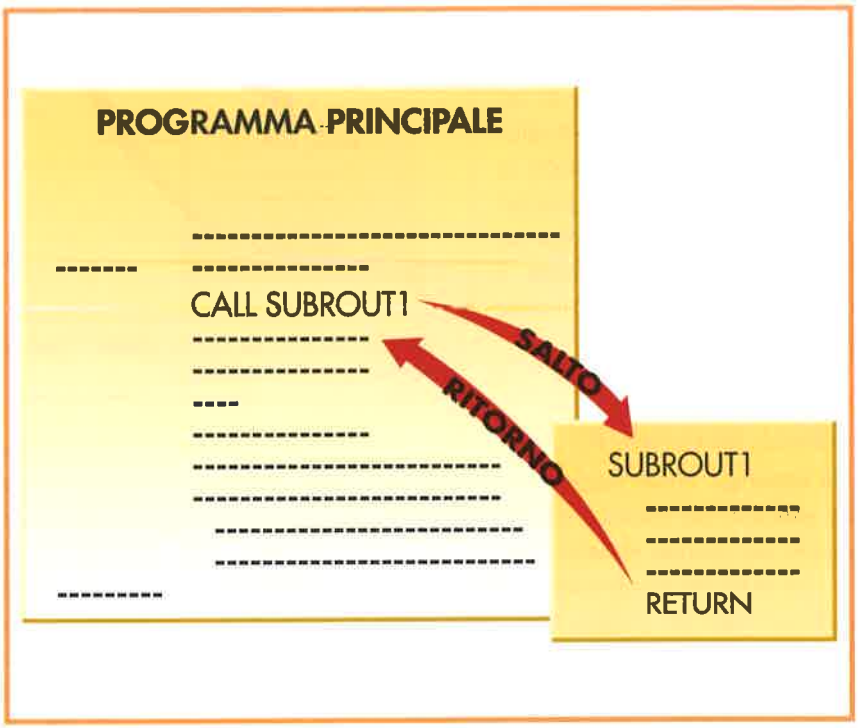
Nelle parole di identificazione sono validi solo i 4 bit meno significativi.

PAROLE DI IDENTIFICAZIONE

Sono quattro posizioni della memoria FLASH che occupano indirizzi compresi fra la 2000 H e la 2003 H. Sono solo accessibili in lettura e scrittura durante il processo di memorizzazione del programma. Dei 14 bit della parola di identificazione, sono validi solo i 4 meno significativi, e in essi, il programmatore può scrivere codici di identificazione, numeri di serie, dati sulla fabbricazione, modelli, ecc. Nella figura si possono notare i bit validi per ogni posizione.

Che cos'è un interrupt?

È una fermata del programma in fase di esecuzione, per saltare a realizzare una routine speciale che riguarda la causa che ha prodotto l'interrupt. Una volta completata la routine dedicata all'interrupt si ritorna al punto di partenza del programma principale. Si può dire che l'interrupt ha le stesse conseguenze di una "chiamata a subroutine". Quando in un programma troviamo un'istruzione con CALL SUBROUTINE 1, iniziano una serie di azioni, che in forma riassunta si riducono ad un salto all'inizio della subroutine e quando sarà stata eseguita la sua ultima istruzione, che è quella di RETURN o ritorno, si torna all'istruzione che segue a CALL SUBROUTINE 1. Nella figura si può vedere graficamente il comportamento della chiamata a subroutine.



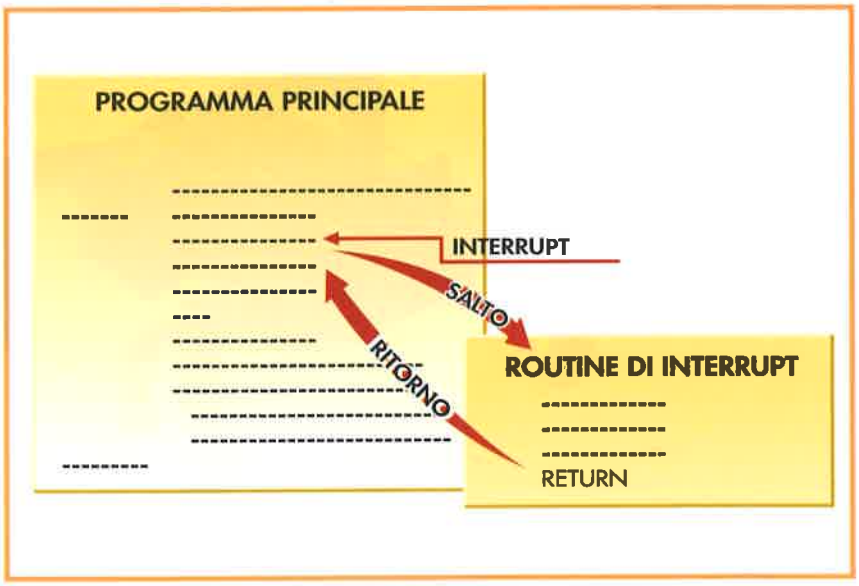
Operatività di una chiamata a subroutine.

L'impiego delle subroutine evita di ripetere nei programmi alcuni procedimenti di uso frequente. Invece di ripeterli tante volte quanto serve, si scrivono una volta sola in forma di subroutine, a cui si accede attraverso una chiamata (CALL).

DIFFERENZA TRA INTERRUPT E CHIAMATA A SUBROUTINE

Una chiamata a subroutine è una rottura del flusso del controllo delle istruzioni del programma principale, di forma sincrona. Si sa quando viene prodotta perché esplicitamente invocata con l'istruzione CALL.

Un interrupt è una rottura "asincrona" del flusso di controllo. Non si sa quando avviene perché le cause dell'origine sono dovute a circostanze non



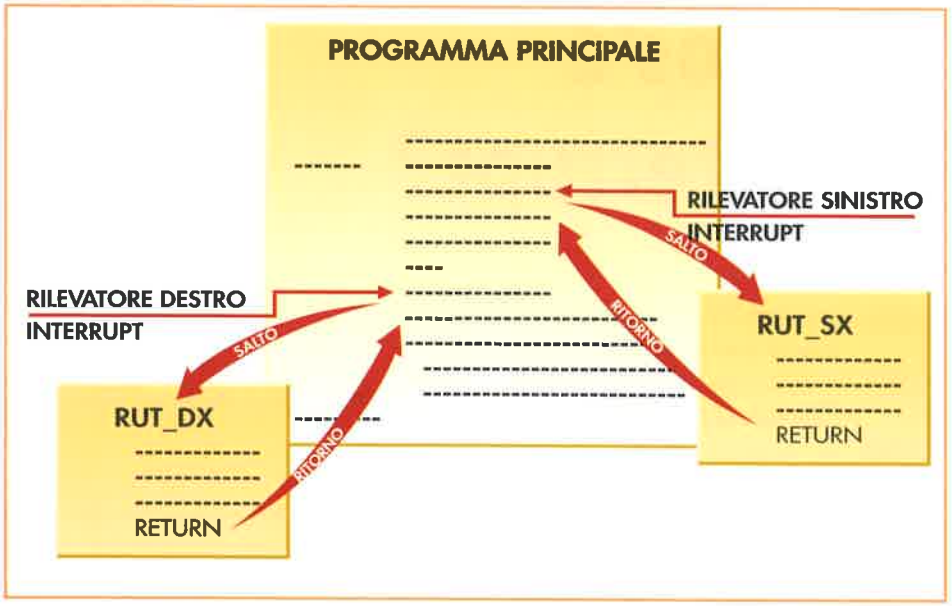
Un interrupt è un avvenimento asincrono che si produce in modo imprevedibile.

MICROCONTROLLER

Concetti e operatività degli interrupt

previste nelle istruzioni del programma. Ad esempio, una causa che provoca un interrupt al processore, è l'attivazione mediante un livello logico alto, di un piedino del microcontroller.

Questo livello alto potrebbe essere prodotto da un comparatore, che riceve come ingresso il segnale analogico di un sensore di temperatura; il valore del segnale ha superato la tensione di riferimento, perché la temperatura a sua volta ha superato la soglia prefissata. Il valore della temperatura varia in modo imprevedibile.



A seconda dell'antenna che ha rilevato l'ostacolo si produce un interrupt differente, con le corrispondenti strategie.

IMPORTANZA DEGLI INTERRUPT

Gli interrupt costituiscono il procedimento più rapido ed efficace del microcontroller per il trattamento degli eventi del mondo esterno.

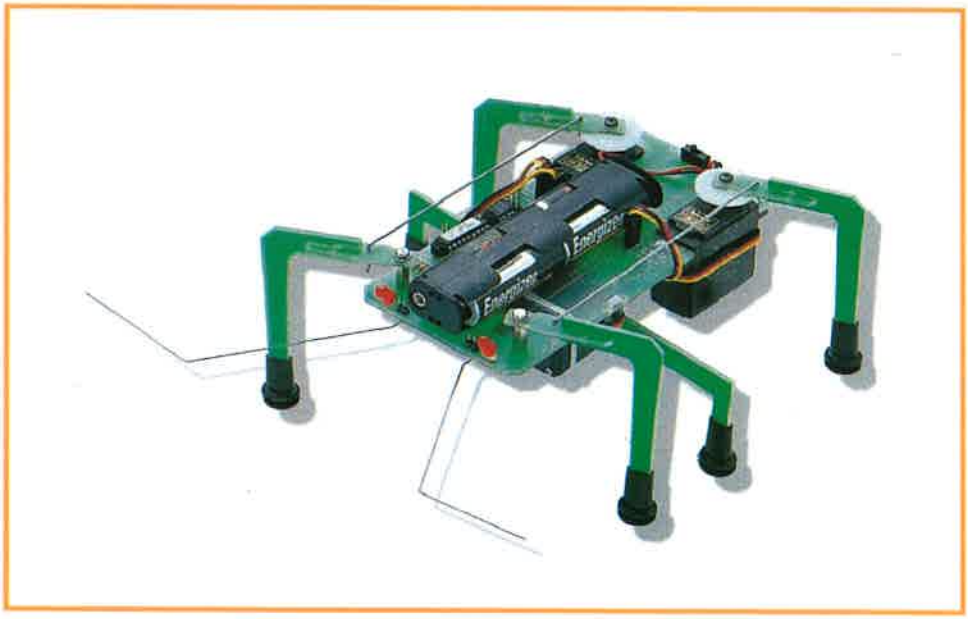
Se qualche parametro o circostanza ha influenza sullo sviluppo del programma, l'interrupt è il mezzo con cui il microcontroller lo gestisce immediatamente. La parte di programma dedicata alla gestione di questo evento si chiama "routine di interrupt".

Prendiamo ad esempio il microrobot a sei gambe che cammina, mostrato nella figura, con una coppia di antenne frontali che funzionano come sensori e che rilevano gli ostacoli sulla traiettoria.

Il microrobot a sei gambe è controllato da un microcontroller che esegue un programma principale, incaricato del movimento degli arti meccanici, per seguire una traiettoria.

Se ad un certo punto del cammino, l'antenna di sinistra tocca un ostacolo, viene applicato un livello logico alto ad uno dei piedini del microcontroller, e viene generato un interrupt.

A quel punto viene abbandonato il programma principale, e si salta ad una routine che si chiama RUT_SX, che contiene una procedura per realizzare la strategia di retromarcia e aggiramento, per evitare l'ostacolo. Nel caso il contatto fosse avvenuto sull'antenna destra, il livello logico sarebbe stato applicato ad un altro piedino, e sarebbe stata lanciata la routine RUT_DX con un'altra strategia per aggirare l'osta-



Le antenne, o fili, frontali del microrobot, funzionano come sensori per la rilevazione degli ostacoli.

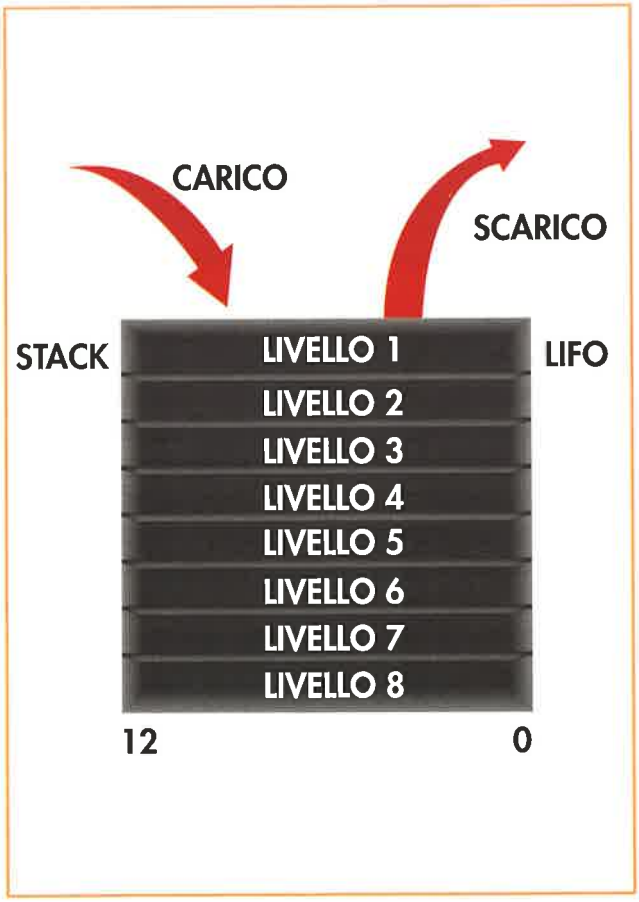
colo sul lato destro, come si può vedere nella figura. In questo esempio si producono due interrupt differenti, perché ogni antenna del microrobot fa riferimento ad un piedino diverso del microcontroller, questo permette i salti a due routine distinte.

IL RUOLO DELLO STACK

Per rendere possibile il corretto funzionamento degli interrupt, è imprescindibile l'utilizzo dello Stack. Come ricorderete, è un insieme di 8 registri, dal livello 1 al livello 8, che funzionano con una struttura LIFO (ultimo ad entrare, primo ad uscire) associata al PC.

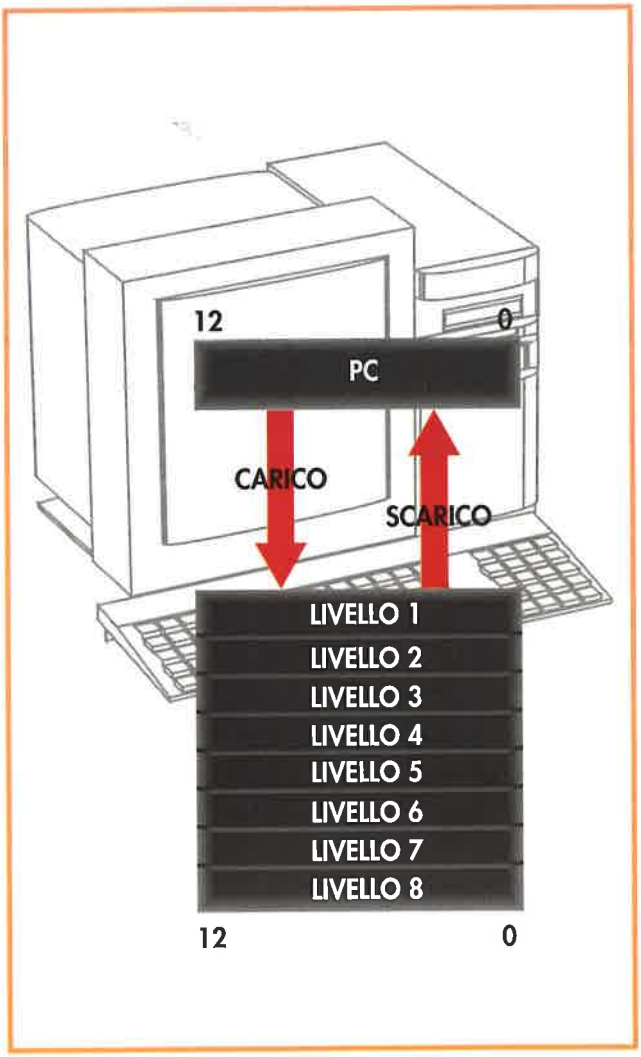
I dati in arrivo dal PC sono caricati nel livello 1, e quelli scaricati in uscita verso il PC sono ancora quelli del livello 1, come si vede nella figura.

Il carico del valore del PC nello Stack avviene in modo automatico, quando si esegue un'istruzione CALL, o quando si origina un interrupt. Anche lo scarico del livello 1 dello Stack sul PC avviene in modo automatico, quando si esegue l'ultima istruzione di ritorno dalla subroutine, o dalla routine di interrupt, come espresso graficamente nella figura.



Lo Stack ha una struttura LIFO.
L'ultimo dato caricato nel livello 1 sarà il primo che uscirà.

Lo Stack è il luogo dove si depositano temporalmente gli indirizzi di ritorno al programma principale.



Il carico dello Stack si realizza in modo automatico con l'istruzione CALL o con un interrupt. Anche lo scarico avviene in modo automatico, con l'istruzione di ritorno dalla subroutine o dall'interrupt.

Dato che dispone di otto livelli, dobbiamo fare attenzione affinché non debordi quando si producono annidamenti, ed è necessario memorizzare più indirizzi di ritorno.

Non si possono scrivere più di otto indirizzi di ritorno.

Molti Stack con un numero di livelli limitato, dispongono di un flag che ci avvisa quando sono pieni, però

questa risorsa non è disponibile nel PIC16F84, resta quindi responsabilità del programmatore vigilare su questo fatto.

OPERATIVITÀ DELL'INTERRUPT

LE FASI DI SVILUPPO DI UN INTERRUPT SONO:

1ª. Si origina l'interrupt e viene acquisito dal processore.

2ª. Si salva il valore del PC nello Stack. Si carica nel livello 1.

3ª. Il PC si carica con il valore del Vector di Interrupt, che punta la prima istruzione della routine dedicata all'interrupt.

4ª. Si eseguono le istruzioni della routine di interrupt.

5ª. Quando si esegue l'ultima istruzione della routine, che è RETFIE (RITORNO), il valore del livello 1 dello Stack viene caricato nel PC, si ritorna al programma principale nel punto dove lo si è abbandonato.

IL PROBLEMA DEL VECTOR DI INTERRUPT

Nel PIC 16F84 il Vector di interrupt contiene l'indirizzo 0004H della mappa di memoria del programma, invece il Vector di reset contiene l'indirizzo 0000H.

Questo implica che l'inizio del programma principale avvenga all'indirizzo 0000H, e che quattro indirizzi dopo, inizi la routine dedicata all'interrupt.

È chiaro che il programma principale "coprirebbe" le istruzioni della routine di interrupt, per evitare questo si colloca in entrambi i vettori un'istruzione di salto incondizionato GOTO, che trasla l'inizio del programma principale e della routine di interrupt ai punti corretti.

In generale, all'indirizzo 0000H, si colloca un salto all'indirizzo 0005H, salvando il Vector di interrupt. Normalmente l'inizio di un programma per il PIC è fatto come il seguente:

```
ORG 0;          Il PC viene caricato con l'indirizzo 0000
goto   inizio;  All'indirizzo 0 c'è un'istruzione di salto a "inizio"
ORG 5;          Il PC si carica con l'indirizzo 0005
inizio  bsf     STATUS,5;  Prima istruzione del programma
                           principale
```

GLI INTERRUPT NEGLI ALTRI PROCESSORI

Negli altri modelli di processori, ad esempio quelli dalla ditta INTEL, gli interrupt dispongono di diversi Vector di interrupt, non solo di uno come nel PIC.

Essi gestiscono una tabella di 256 Vector di Interrupt, come mostrato dalla figura.

Il vantaggio di disporre di tanti vettori, sta nel fatto



Tabella con 256 vettori differenti di interrupt, propria di alcuni processori di INTEL.

che si possono trattare in forma differenziata le distinte cause che generano gli interrupt.

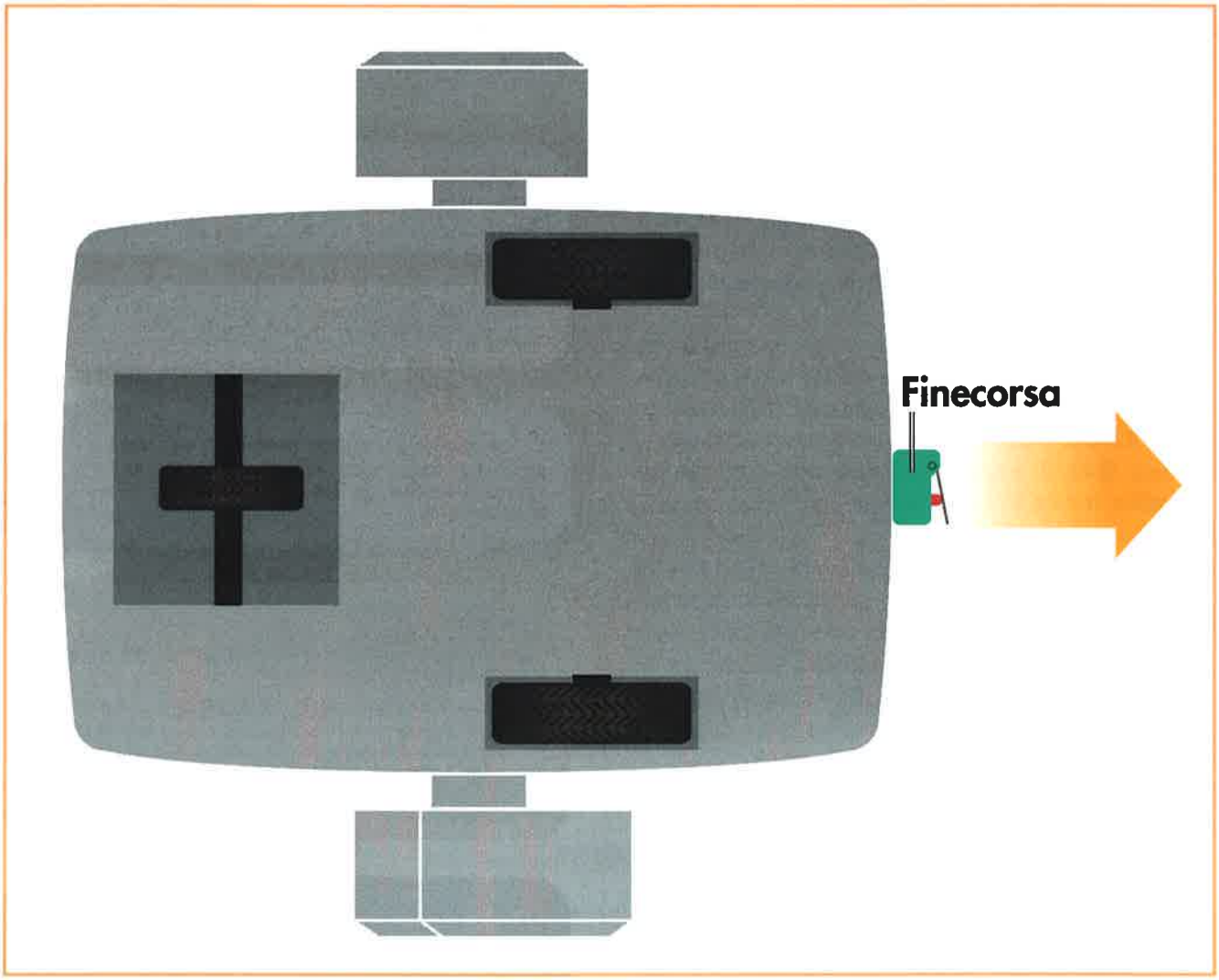
Nel PIC16F84, esiste un solo Vector di interrupt per le diverse cause che li provocano, quindi le routine dedicate ad essi iniziano in modo uguale per tutti.

La prima cosa da fare sarà verificare quale sia stata la causa dell'interrupt, per deviare il flusso di programma alla routine dedicata a quella determinata causa; per questo ogni diversa causa di interrupt dispone di un flag dedicato.

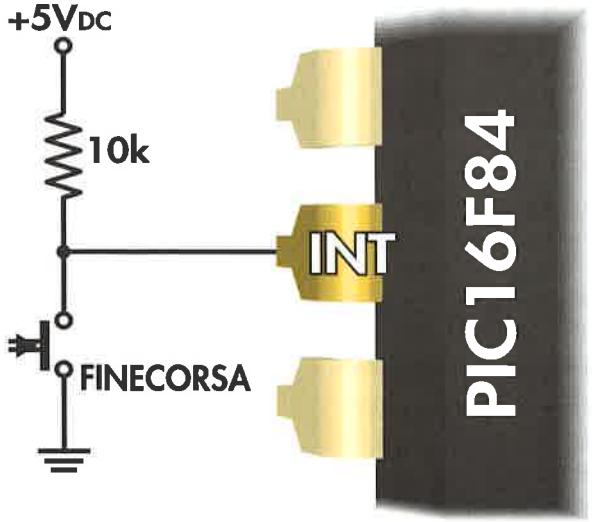
Utilità dell'interrupt

Possiamo montare su Monty un finecorsa sulla parte anteriore in modo che rilevi gli ostacoli sul suo cammino, come si può vedere dalla figura. L'ideale sarebbe che quando il sensore chiude i suoi contatti, il processore abbandoni il programma principale in esecuzione per entrare in una routine specializzata, che ad esempio lo faccia retrocedere di un centimetro, girare di 90°, avanzare di qualche centimetro, girare nuovamente di 90° e continuare il suo percorso.

Il tempo che deve trascorrere fra l'attivazione del sensore e l'esecuzione del programma specializzato nella manovra anti-ostacolo, deve essere minimo. La cosa migliore è utilizzare un interrupt, per questo è necessario progettare un circuito per fare in modo che subito dopo che il contatto di finecorsa si è chiuso, arri-



Il finecorsa anteriore serve per rilevare gli ostacoli.



Quando si chiude l'interruttore di finecorsa, si genera un fronte di discesa che si applica al piedino INT e causa un interrupt.

vi un livello logico basso al piedino RB0/INT del PIC16F84 che è quello dedicato alla richiesta esterna di interrupt. Il fronte di discesa che si produce collegando a massa il piedino quando si chiudono i contatti del sensore, produce l'interrupt, come si può vedere nella figura.

Utilizzando un interrupt, nello stesso momento che il finecorsa rileva un ostacolo, si attiva l'interrupt con la routine associata ad esso. L'altro modo consiste nel collegare il livello logico prodotto dai contatti del sensore, ad una linea di ingresso normale che il processore dovrebbe testare in continuazione, per conoscerne lo stato.

Questo suppone, oltre alla perdita di tempo dovuta all'esecuzione del test durante il programma principale, un'attenzione non immediata alla situazione visto che la CPU nel programma principale realizza anche altri compiti oltre all'ispezione della linea di ingresso che fa riferimento al sensore.

L'interrupt è il metodo più sicuro, rapido ed efficace, per reagire agli eventi esterni che influenzano l'elaborazione del programma principale.

OPERATIVITÀ DEGLI INTERRUPT NEL PIC16F84

Dopo aver completato l'istruzione in corso di esecuzione, se il PIC accetta un interrupt, sviluppa i seguenti passi operativi:

1°) Salva nel livello logico 1 dello Stack il contenuto del Program Counter o PC, che contiene l'indirizzo dell'istruzione successiva del programma principale.

2°) Nel PC viene caricato il valore del Vector di interrupt che è all'indirizzo 0004 H, dove inizia la routine dedicata all'interrupt.

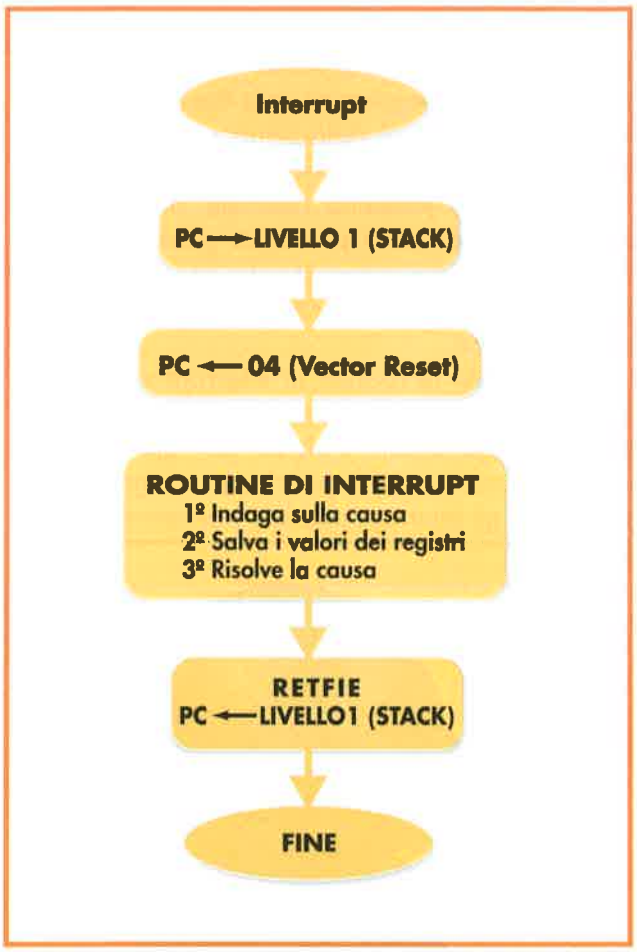
3°) Si esegue l'istruzione scritta all'indirizzo 4 della memoria di programma e poi quelle seguenti. La prima istruzione della routine di interrupt generalmente è un salto (GOTO) all'indirizzo del vero inizio di questa routine.

4°) La routine dedicata all'interrupt inizia realizzando due compiti importanti:

a) Indaga sulla causa che ha generato l'interrupt, e per fare questo controlla i flags

b) Scrive nella memoria dei dati RAM, area GPR i valori di alcuni registri del processore, con i valori relativi al programma principale. Nel caso in cui questi registri fossero utilizzati dalla routine di interrupt, i loro contenuti subirebbero delle modifiche, e al momento di tornare al programma principale non si disporrebbe più delle stesse informazioni di quando lo si era lasciato.

5°) Quando si giunge all'ultima istruzione della routine dedicata all'interrupt, che è quella di ritorno (RETFIE), si carica nel PC il contenuto del livello 1 dello Stack, ritornando al programma principale nel punto in cui lo si è abbandonato.



Organigramma del trattamento di un interrupt nel PIC16F84.

CAUSE DI INTERRUPT

Ci sono quattro cause che possono generare un interrupt nel PIC16F84:

1ª. Attivazione da un piedino. Si genera un fronte di tensione sul pin 6 (RB0/INT). Questa causa si chiama "interrupt esterno".

2ª. Quando va in overflow il temporizzatore TMR0. Il suo contenuto passa da FF H a 00 H.

3ª. Quando cambia lo stato logico di uno qualsiasi dei quattro ingressi più significativi della porta B (RB7:RB4).

4ª. Quando termina il ciclo di scrittura di un byte in un indirizzo della memoria EEPROM dei dati.

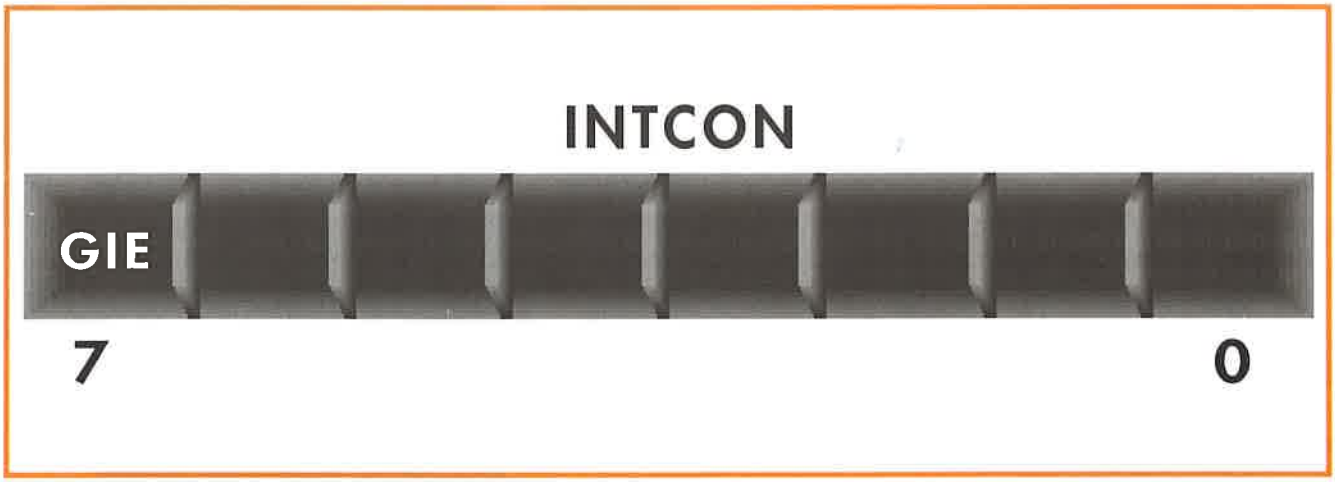
Una delle difficoltà iniziali che presentano queste funzioni, è che qualunque sia la causa che ha originato l'interrupt, la routine dedicata inizia sempre all'indirizzo del Vector di interrupt che è lo 0004 H. Altri processori dispongono di un Vector di interrupt per ogni causa che li produce, con il PIC 16F84 non c'è altra soluzione che iniziare indagando quale sia stata la causa dell'interrupt, testando i flags corrispondenti, e indirizzando il processore alla routine corretta.

IL CONTROLLO DELL'INTERRUPT

A volte molti programmi non necessitano di interrupt, e quindi non li usano.

Per questo motivo deve esistere una chiave generale che permetta o proibisca gli interrupt nel loro insieme.

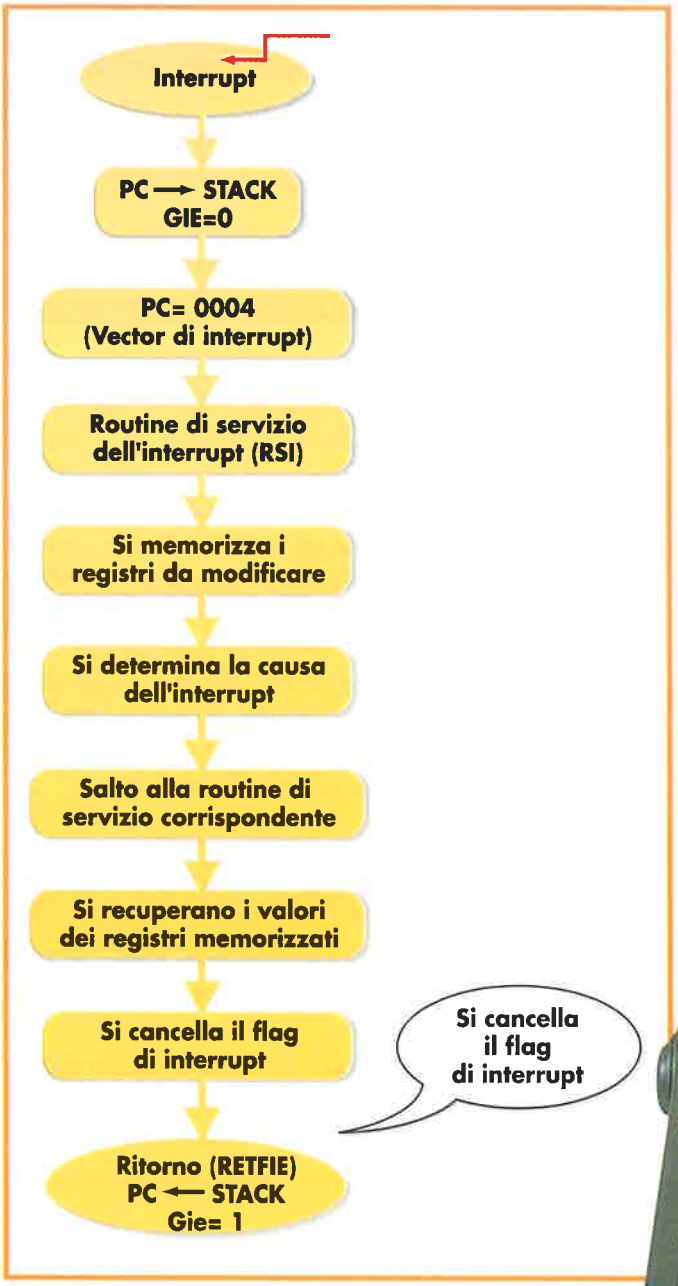
Nel PIC16F84, esiste un registro di controllo dell'area SFR della RAM, chiamato INTCON, che regola il funzionamento degli interrupt. Il bit più significativo di



Il bit più significativo del registro INTCON è quello che permette o proibisce globalmente le quattro cause di interrupt.

MICROCONTROLLER

Gli interrupt del PIC16F84



Organigramma dello sviluppo di un interrupt tenendo conto della funzione di GIE.

INTCON è chiamato GIE (Permesso Globale di Interrupt).

Se si scrive questo bit e si pone GIE = 0 tutti gli interrupt saranno disabilitati, e il processore non ne accetterà nessuno.

Al contrario se scriviamo GIE = 1 il processore accetterà qualsiasi interrupt che si genererà, e il programmatore potrà lavorare con essi.

Ogni interrupt dispone di un bit particolare di abilitazione, che deve essere attivato se si desidera che al prodursi dell'interrupt, il processore lanci le fasi del suo sviluppo. Nell'organigramma della figura, si approfondiscono le operazioni dello sviluppo di un interrupt, tenendo conto della funzione del bit GIE.

Nella figura possiamo osservare che quando si accetta un interrupt, il processore pone automaticamente GIE = 0, per evitare che durante l'elaborazione della routine dedicata a questo interrupt, se ne producano altri nuovi.

Completata la routine di interrupt ed eseguita l'ultima istruzione RETFIE, che è quella di ritorno al programma principale, il bit GIE riprende il valore 1, dato che ritornando al principale, bisogna fare in modo che gli interrupt possano nuovamente essere accettati.



L'interrupt esterno

Delle quattro possibili cause di interrupt che ammette il PIC16F84 ce n'è una che gli permette di restare in contatto permanente con gli avvenimenti che succedono nel mondo esterno.

Questo interrupt è supportato dal piedino 6 del microcontroller, e si chiama RB0/INT. Si tratta di un piedino multifunzione, che può essere utilizzato in due modi differenti. In primo luogo può funzionare come linea di ingresso/uscita digitale (RB0), nel qual caso si comporterà come ingresso se nel bit 0 del registro di configurazione TRISB avremo caricato un 1, mentre sarà un'uscita se nel suddetto bit avremo scritto uno 0. Inoltre può funzionare come linea di richiesta di interrupt esterno (INT).

Per fare in modo che questo piedino possa ricevere la richiesta esterna di interrupt deve essere abilitato per questo lavoro, per questo scopo è necessario mettere a 1 il bit di Permesso Globale di Interrupt, GIE. Inoltre bisogna abilitare anche il permesso particolare di questo interrupt, mettendo a 1 il bit INTE.

Sia GIE che INTE si trovano nel registro INTCON che occupa l'indirizzo 0B H del banco 0 e il correlativo del banco 1, come illustrato nella figura.



Per fare in modo che il piedino RB0/INT supporti richieste di interrupt esterni, i bit GIE e INTE devono valere 1.

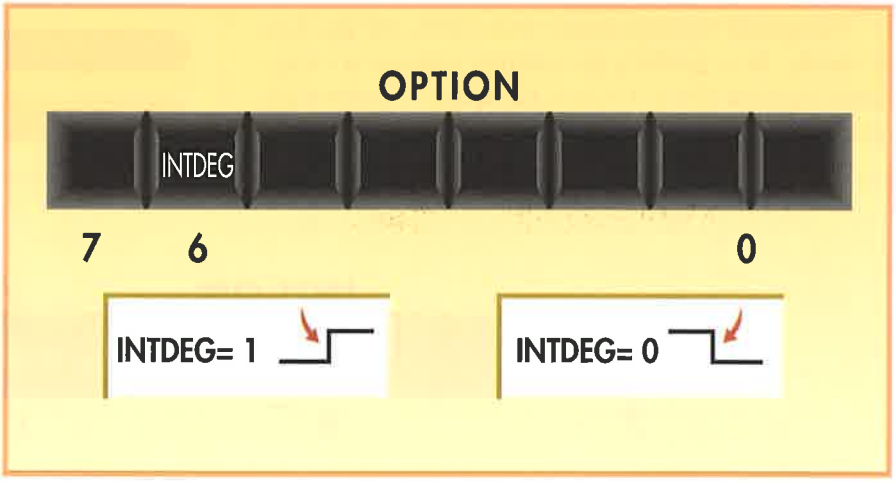
CONTROLLO DEL FRONTE DI ATTIVAZIONE

Il piedino RB0/INT può generare un interrupt quando ad esso viene applicato un fronte di salita o di discesa. Un fronte di salita consiste nella transazione di tensione dal basso verso l'alto, mentre un fronte di discesa è la transazione inversa.

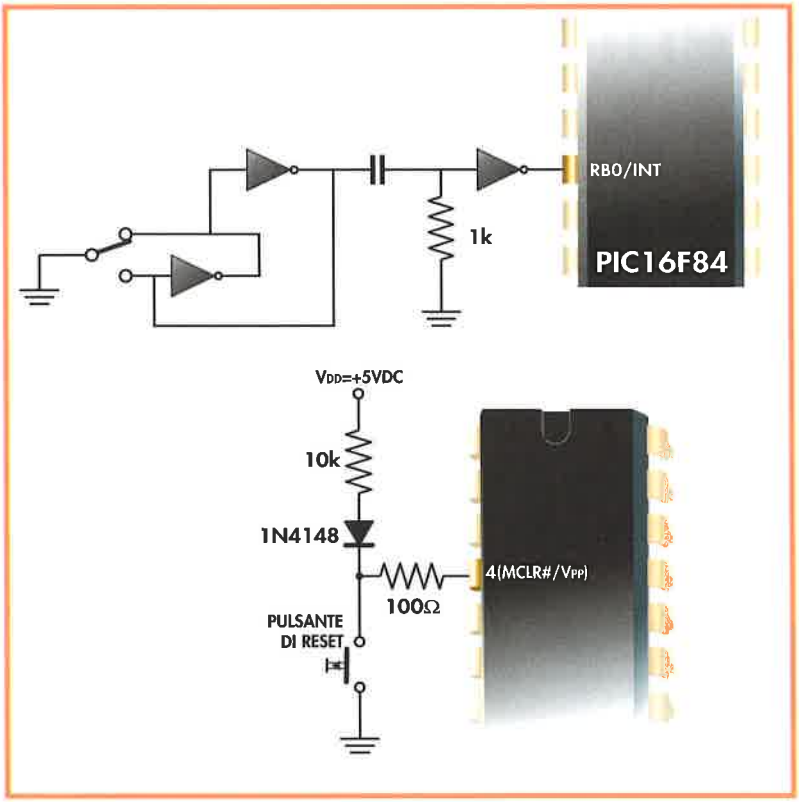
Per selezionare il tipo di fronte, il PIC16F84 dispone del bit INTDEG, situato nel registro OPTION, che occupa l'indirizzo 81 h del banco 1. Quando INTDEG = 1 il fronte attivo sarà quello di salita, mentre quando INTDEG = 0 il fronte attivo sarà quello di discesa, come indicato nella figura.

Lo schema elettrico riportato nella figura della pagina seguente, corrisponde ad un generatore di fronti per attivare il piedino RB0/INT.

Utilizza due porte invertenti 74LS04 che, grazie al particolare collegamento, costituiscono un filtro antirimbalzo per il pulsante del circuito. La rete R-C controlla la durata dell'impulso che contiene i due tipi di fronti.



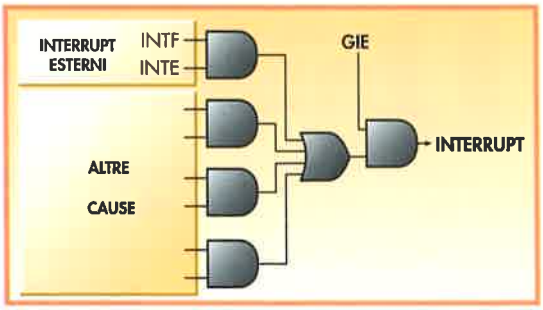
Il bit INTDEG del registro OPTION determina il fronte attivo sul pin RB0/INT.



Circuito che genera l'impulso di una durata determinata, per provocare interrupt in modo ciclico.

bit INTF = 1. Nella figura è mostrata la logica di controllo che è dedicata alla generazione di questo tipo di interrupt nel PIC16F84.

In seguito descriviamo con alcuni commenti un piccolo programma per gestire l'interrupt esterno impostando il fronte di discesa come fronte attivo.



Logica di controllo corrispondente al circuito di generazione degli interrupt esterni.

Dentro la ruotine di interrupt e prima di ritornare al programma principale, il programmatore deve necessariamente riportare il flag INTF = 0, altrimenti subito dopo l'istruzione di ritorno RETFIE, torneremo a

IL FLAG DELL'INTERRUPT ESTERNO

Ogni volta che si genera un fronte attivo sul pin RB0/INT programmato per ricevere interrupt, e questo viene generato, un flag incaricato di segnalare questa situazione si pone automaticamente a 1.

Si tratta del flag INTF del registro INTCON come mostrato nella figura.

Da quanto esposto sinora deduciamo che per fare in modo che si produca un interrupt esterno, è necessario che il bit GIE = 1, che il bit INTE = 1 e che il pin RB0/INT riceva un fronte attivo. Tutto questo fa sì che il

```
bsf     STATUS, rp0     ;Selezione del banco 1
bcf     OPTION, intdeg  ;INTDEG = 0, fronte di discesa
bcf     STATUS, rp0     ;Selezione del banco 0
bsf     INTCON, gie     ;GIE = 1, Permesso Globale di Interrupt
bsf     INTCON, inte    ;INTE = 1, Permesso di Interrupt Esterno
bcf     INTCON, intf    ;INTF = 0. Ci assicuriamo che il flag sia a 0
```



Il bit INTF è un flag che viene messo automaticamente a 1 quando si produce l'interrupt esterno.

ripetere il processo di interrupt, quindi il flag INTF sarà ancora a 1.

IMPORTANZA DEGLI INTERRUPT NEL CONTROLLO INDUSTRIALE

Il meccanismo di interrupt ricopre un aspetto vitale nei sistemi di controllo industriali. L'interrupt è il metodo più rapido ed efficace per porre in comunicazione il processore con il mondo esterno.

Ogni volta che si produce un evento fisico nell'ambiente reale da controllare, ad esempio l'innalzarsi della temperatura di un forno oltre ad una determinata soglia, un sensore trasduce il parametro temperatura in una grandezza elettrica proporzionale, convertendola in una tensione capace di attivare uno dei piedini di interrupt che ha il microprocessore, avvisandolo dell'evento che si sta verificando.

La risposta della CPU è immediata: sospende l'esecuzione delle istruzioni del programma in corso, salva in una zona della memoria chiamata STACK l'indirizzo di abbandono del programma e carica nel PC un indirizzo determinato a cui salta per eseguire una routine che sarà incaricata del trattamento e risoluzione del problema che ha generato l'interrupt. Nell'esempio che abbiamo commentato, la routine che fa riferimento all'interrupt, potrebbe ridurre l'ingresso del combustibile al forno in modo che questo si raffreddi e la temperatura torni a valori adeguati.

Nel momento in cui si abbandona questa routine, si ricarica l'indirizzo di ritorno dallo STACK, e si torna ad eseguire le istruzioni del programma principale dal punto in cui lo si è abbandonato.

La vera importanza degli interrupt risiede nella loro rapida comunicazione di qualsiasi anomalia fisica esterna che succede nel mondo reale al processore, in modo che questo passi al loro trattamento. La rilevazione di qualsiasi anomalia quando si usa un interrupt, provoca il trattamento immediato dell'anomalia stessa, mediante l'esecuzione di una routine preposta per risolvere la situazione.

GLI INTERRUPT NEI PROCESSORI INTEL

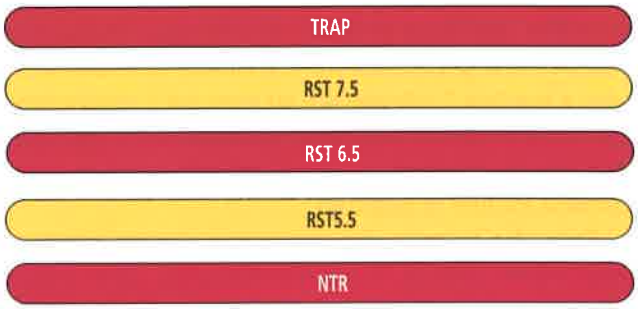
INTEL, il pioniere dei microprocessori e fabbricante dei Pentium che equipaggiano i nostri personal computer, ha implementato in essi un meccanismo di trattamento degli interrupt eccezionale, la cui flessibilità e varietà li rendono i più potenti del mercato.

INTEL fabbrica anche microprocessori che gestiscono dati a otto bit, così come il PIC. Si tratta del modello 8085, che è stato commercializzato dal 1976 e ancora oggi è uno standard industriale.



Il processore INTEL dispone di uno straordinario meccanismo di interrupt.

Dato che il fabbricante orientò l'8085 al settore dei sistemi industriali dedicò un insieme di cinque piedini per supportare le richieste di interrupt per l'hardware esterno. Questi piedini sono:



Alcuni dei piedini di interrupt dell'8085 si attivano mediante i fronti e altri con livelli. Inoltre ognuno ha un proprio grado di priorità. La priorità più alta è del TRAP a cui seguono, in ordine decrescente, la RST 7.5, RST 6.5, RST 5.5 e INTR.

Questi interrupt si classificano in due gruppi: mascherabili e non mascherabili. Quelli che sono mascherabili hanno la proprietà di essere ignorati dalla CPU quando vengono attivati, nel caso che un bit speciale situato in un registro di controllo sia stato messo a zero.

Questa caratteristica permette al programmatore, scrivendo e cancellando questo bit, di accettare o ignorare l'interrupt in parti vitali del programma.

Gli interrupt dell'8085 sono vettorializzati, il che suppone che quando sono accettati ognuno carichi sul PC un indirizzo specifico della memoria, originando un

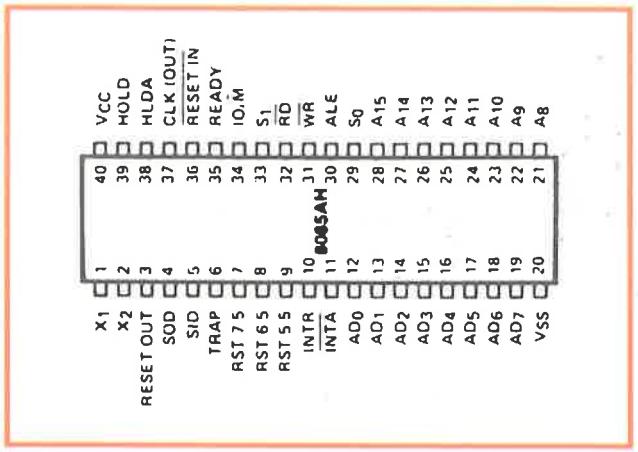


Diagramma dei piedini dell'8085.

salto del flusso di controllo a quella posizione dove si suppone inizi la routine che riguarda detto interrupt. Nell'8085 ogni piedino di interrupt dispone di un suo proprio indirizzo vettorializzato, così si differenziano le diverse cause che provocano gli interrupt e si assegna ad ognuna una routine indipendente.

Nel microcontroller PIC esiste solo un Vettore di interrupt, che ha assegnato l'indirizzo 4 della memoria di codice, per tutte le cause di interrupt. Qualunque interrupt che si produca nel PIC16F84, fa sì che il PC passi a puntare l'indirizzo 4 e ad eseguire il programma che inizia in questa posizione. Generalmente nell'indirizzo 4 si usa mettere un "goto" all'indirizzo vero di inizio della routine di interrupt.

Dato che le cause che provocano gli interrupt possono essere diverse, il loro trattamento deve essere fatto in modo individuale, per questo all'inizio della routine di interrupt in un PIC inizia la verifica di cosa ha originato l'evento in modo da poter scegliere la routine corrispondente.

DESCRIZIONE DEL FUNZIONAMENTO DEGLI INTERRUPT ESTERNI DELL'8085

TRAP

Si tratta di un interrupt non mascherabile, quindi la CPU lo accetta sempre indipendentemente dallo stato del bit di maschera. A questo corrisponde un vector che indirizza il flusso di controllo alla posizione di memoria 24 Hex. È sensibile sia sul fianco sia al livello.

È quello di maggior priorità, di conseguenza se arrivano diverse richieste di interrupt contemporaneamente, il TRAP è il primo che viene gestito. Generalmente è impiegato per far fronte a situazioni estreme, come la mancanza di alimentazione, che permette l'entrata in servizio della fonte di alimentazione alternativa o delle batterie di sicurezza.

RST 7.5

Il piedino che attiva questo interrupt è sensibile al fronte e gli corrisponde un vector il cui indirizzo è 3C H. È un interrupt mascherabile, che può essere ignorato quando il bit di maschera situato in un registro di controllo è messo a zero. Ha una priorità inferiore a quella del TRAP.

RST 6.5 e RST 5.5

Entrambi i piedini si attivano quando viene applicato un livello logico alto, che deve essere mantenuto fino a che la CPU non lo accetta. La loro priorità è inferiore alla RST 7.5 e i vector che hanno assegnati sono al 34 H e 2C H, rispettivamente.

LA VERSATILITÀ DEL PIEDINO DI INTERRUPT INTR

INTEL, non soddisfatta dalle precedenti richieste di interrupt esterna, ne ha introdotta una speciale, il cui meccanismo viene solo impiegato da questo fabbricante, che permette di saltare fino ad otto routines differenti con il solo piedino INTR.

Inizialmente INTEL ha introdotto nel repertorio di istruzioni la chiamata RESTAR n, il cui mnemonico è RST n. Dato che n può valere otto valori da 0 a 7, dà origine a otto codici di istruzioni corrispondenti a RST0, 1, 2, 3, 4, 5, 6 e 7, ognuno dei quali è composto dai soli otto bit. Eseguendo queste istruzioni si produce un salto agli indirizzi n x 8, ossia, alla 0, 8, 16, 32, ecc. Dal programma, e in forma sincrona, si può saltare a questi indirizzi di memoria, in ognuno dei quali si può dare inizio ad una routine differente.

INTR è il piedino di richiesta di interrupt mascherabile di priorità minima. Quando la CPU genera due cicli del bus di lettura, i quali sono attivati indicando il piedino INTA (Riconoscimento di Interrupt). Nel secondo dei cicli, la CPU legge gli otto bit presenti nel bus dei dati (D0-D7) e li interpreta come il codice dell'istruzione seguente che deve eseguire.

Se il codice che introduce tramite il bus dei dati il richiedente corrisponde con uno degli otto dell'istruzione RST n, si produrrà un salto agli otto indirizzi possibili (n x 8).

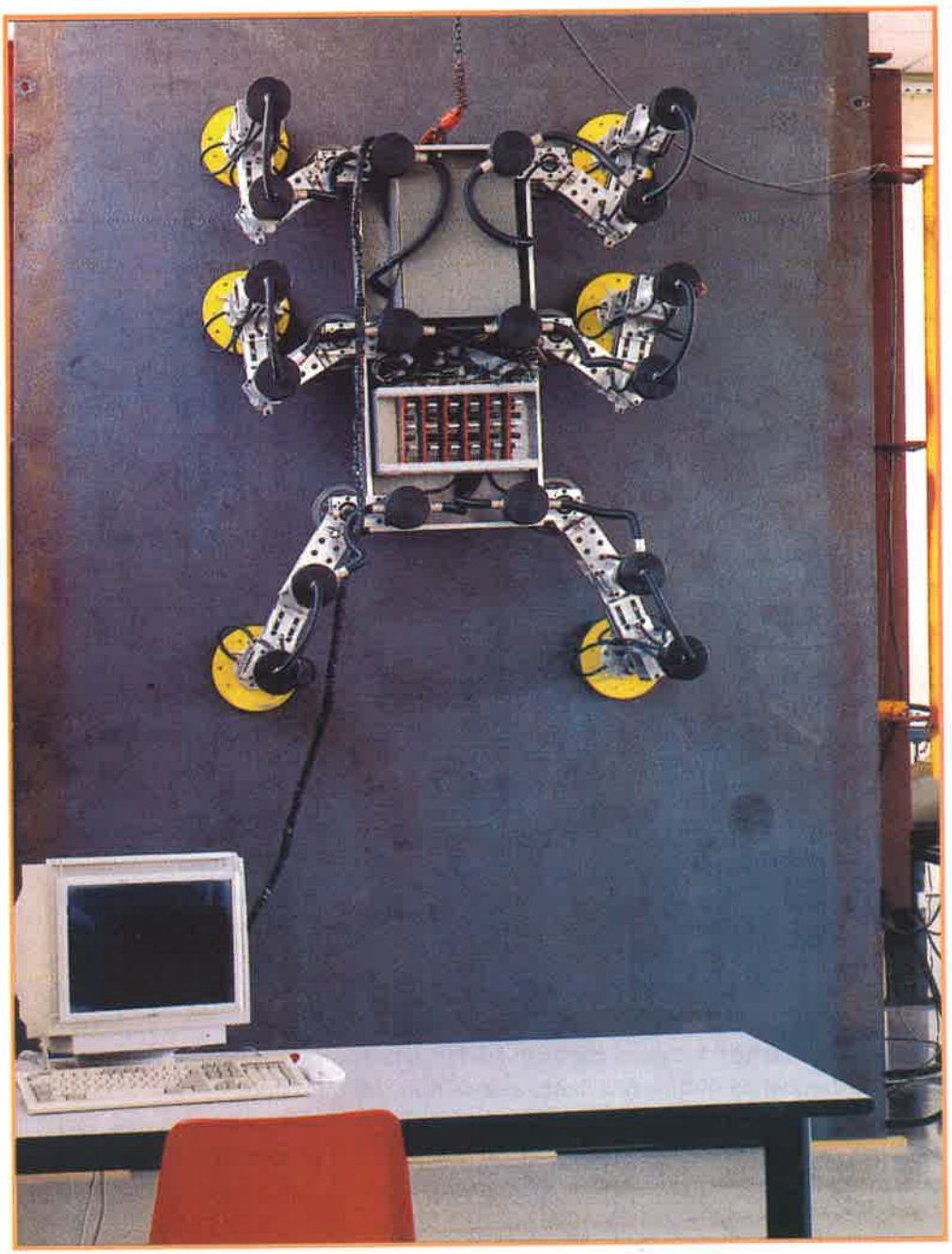
Questo fa sì che il PC si carichi con uno degli otto indirizzi e inizi l'esecuzione della routine applicata a partire da esso.

L'interrupt per overflow del TMR0

GLI SCOPI DEL TEMPORIZZATORE TMR0

La misura del tempo è uno dei lavori più comuni nelle applicazioni industriali. Nella fotografia è mostrato un robot scalatore a sei gambe, per la saldatura in verticale nelle costruzioni navali (REST) dell'Istituto delle Automazioni Industriali del CSIC, nel quale la temporizzazione nel movimento delle sue articolazioni è fondamentale per l'esecuzione dei suoi compiti.

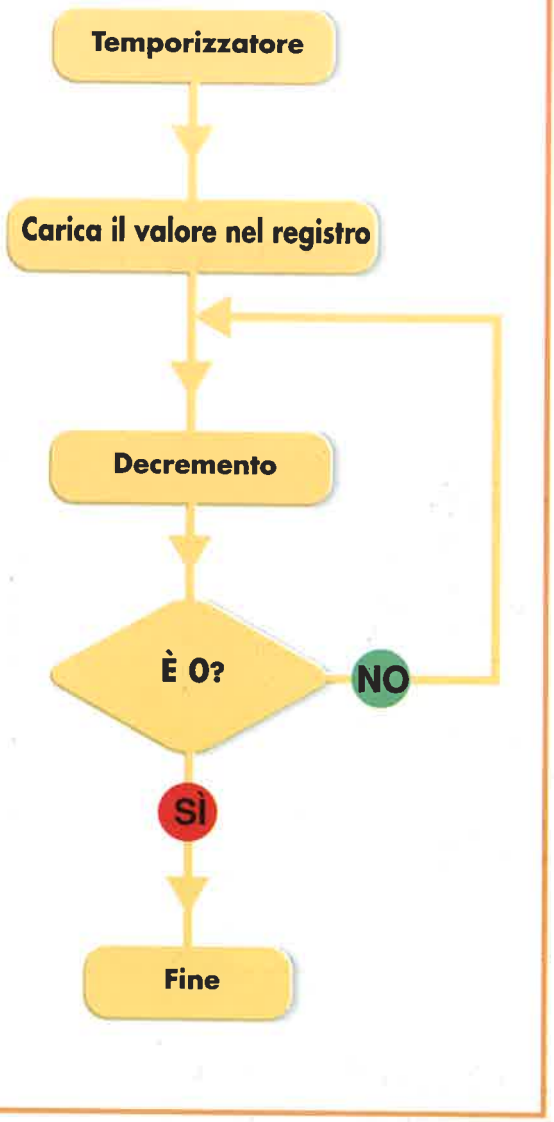
In una sequenza di tempi all'interno di un processo chimico si può apprezzare l'importanza e la necessità dei temporizzatori. Durante 3 ms si deve aprire una elettrovalvola per aggiungere un determinato prodotto ad una miscela in preparazione. Allo stesso tempo per un periodo di 2,6 ms, un'altra elettrovalvola dovrà essere attivata per inserire una certa quantità di un determinato liquido nella miscela. Dopo 20 ms di attesa, necessari per miscelare entrambi i prodotti, si procede allo scuotimento per 12 ms. Questa è una sequenza abituale



Robot scalatore a sei gambe per la saldatura in verticale nelle costruzioni navali (REST). (Istituto dell'Automazione Industriale del CSIC).

MICROCONTROLLER

La misura del tempo



Organigramma tipico che presenta la strategia che segue la CPU quando si dedica alla misura del tempo.



I temporizzatori sono contatori ascendenti o discendenti che funzionano in modo autonomo, e che dispongono di flag che avvisano quando arrivano al massimo o attraversano lo zero.

nello sviluppo di un processo industriale, e in essa i tempi sono un fattore fondamentale. Il processore può misurare facilmente il tempo; il controllo è effettuato in base alla frequenza di funzionamento e al tempo che impiega ogni istruzione ad essere eseguita. Così, ad esempio, un PIC16F84 lavorando con un cristallo di quarzo da 4 Mhz e un oscillatore XT, ha un periodo T = 250 ns. Per eseguire un'istruzione normale servono quattro periodi, ossia, 4 x 250 = 1.000 ns.

Le istruzioni di salto impiegano il doppio. Quando la CPU si dedica a contare il tempo generalmente lo fa caricando prima il registro con un determinato valore, dopodiché lo decrementa e verifica se è arrivato a zero, in caso contrario torna a decrementarlo e a ripetere la verifica.

Quando arriva a zero significa che è passato il tempo che doveva contare. Nell'organigramma della figura si può vedere la strategia impiegata dalla CPU nel conteggio del tempo.

Anche se la CPU può realizzare il compito di misurare il tempo senza alcuna difficoltà, mentre esegue questo lavoro non può fare altre cose.

Di conseguenza, destinare il processore alla misura del tempo significa privare di supervisione e di attenzione il programma principale.

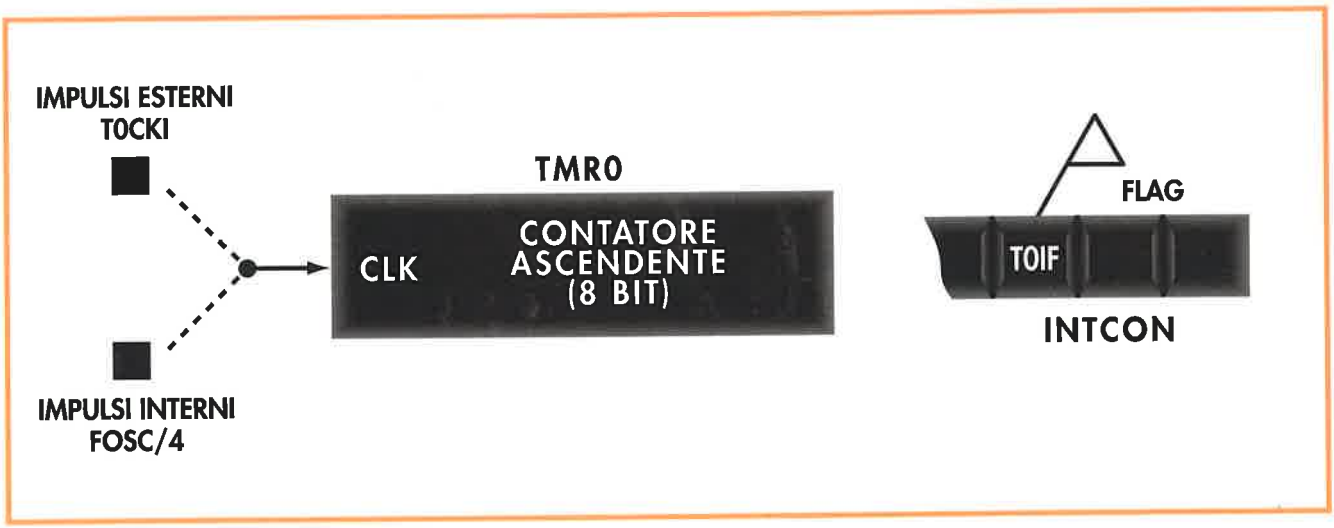
I temporizzatori sono contatori che si caricano con un determinato valore all'inizio del conteggio, e automaticamente si incrementano o decrementano fino ad arrivare al massimo valore o arrivare a zero, momento nel quale comunicano l'accaduto al processore, che nel frattempo in parallelo può eseguire altre istruzioni del programma principale.

L'OVERFLOW DEL TMR0

Il temporizzatore TMR0 è un contatore ascendente a 8 bit che passa a 00 H quando arriva al massimo valore di conteggio, FF H. Questa transizione è chiamata overflow.

Nel caso del TMR0 il suo valore si incrementa al ritmo degli impulsi di clock che sono applicati al piedino RA4/T0CKI o tramite l'oscillatore principale interno con la frequenza Fosc/4.

Quando il valore del TMR0 passa da FF H a 00 H il flag T0IF, che è il bit 2 di INTCON, si pone a 1 automaticamente.



Ingressi degli impulsi al TMR0 e situazione del flag al suo overflow.

L'INTERRUPT DEL TMR0

Per fare in modo che il processore sappia quando il TMR0 va in overflow, deve testare frequentemente lo stato del flag T0IF. Il check di questo bit suppone un'attenzione costante, quindi una perdita di rendimento. Per sapere immediatamente quando avviene l'overflow del TMR0 è molto più efficace generare un interrupt al momento in cui avviene.

Per fare questo è necessario abilitare l'interrupt del temporizzatore, il quale implica di porre il bit GIE = 1, che è il permesso Globale degli Interrupt, e dopo attivare il bit di permesso dell'interrupt del TMR0, ossia, scrivere un 1 in T0IE, così come si vede nella figura della pagina seguente.

MISURA DEL TEMPO

Per calcolare il tempo che misura il TMR0, bisogna conoscere il valore che si carica inizialmente in esso, dato che è un registro che occupa l'indirizzo 1 del banco 0 dell'area SFR della memoria dei dati RAM. Questo valore espresso in decimale, lo chiameremo N; bisogna anche tener conto della frequenza di funzionamento del processore così come del range programmato per il Divisore di Frequenza, in caso di un suo utilizzo. La formula che determina questo tempo è:

Tempo = 4 . Tosc . (256 – N) .
. Range del Divisore di Frequenza

Esercizio

Consideriamo un PIC16F84 funzionante con cristallo di quarzo da 4 Mhz. Supponiamo di caricare il TMR0 con il valore decimale 56, il Divisore di Frequenza sarà assegnato al TMR0 e sarà programmato con range di 1: 32. Calcoleremo il tempo di misura del TMR0 sino al suo overflow:

Tempo = 4 . 250 ns . (256 – 56) . 32 = 6,4 ms

GLI INTERRUPT ESTERNI NEI PENTIUM

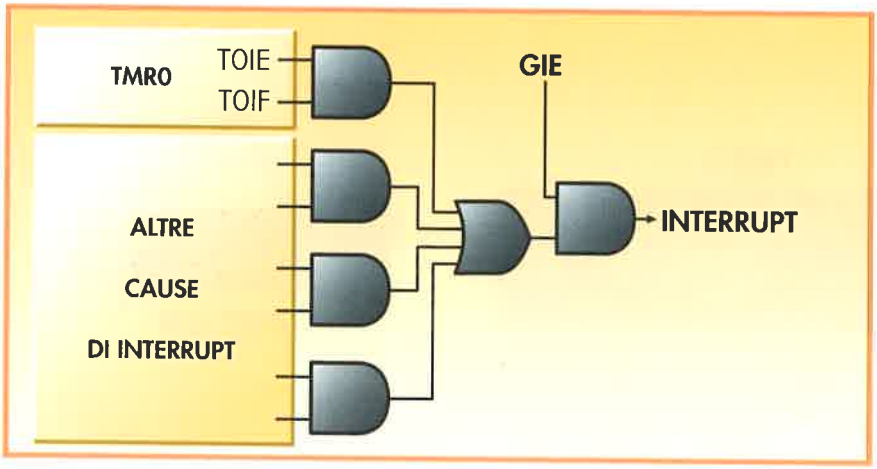
La famiglia x86 di INTEL, alla quale appartengono i moderni Pentium, dispone di un meccanismo di alta qualità e versatilità per supportare gli interrupt esterni. È interessante analizzare le alternative esistenti nei processori più utilizzati e verificarli con quelli dei PIC.

Per gli interrupt esterni la famiglia x86 utilizza due piedini: NMI e INTR. Ogni volta che uno di essi è attivato con livello logico appropriato, la CPU ferma il flusso di controllo, sospende l'esecuzione delle istruzioni nel programma principale in corso, salva nello Stack l'indirizzo di ritorno, e salta ad eseguire una routine incaricata del trattamento della causa che ha originato l'interrupt.

Esistono 256 vettori di interrupt che danno accesso a 256 routines differenti. Questi vettori sono indicati nella tabella degli ingressi che si chiama IDT (Tabella di Descrizione degli Interrupt).

Questa caratteristica determina una differenza fondamentale con i PIC che dispongono di un solo vettore di interrupt per tutte le cause che possono originare una richiesta di interrupt.

Nel PIC16F84 il vettore di interrupt occupa l'indirizzo 4 della memoria di programma e dato che è comune a tutte le cause, il primo lavoro che realizza



Quando il TMR0 va in overflow il segnalatore T0IF si pone a 1 automaticamente e si genera un interrupt sempre che i bit GIE e T0IE valgano 1.

L'INTERRUPT NMI

Si tratta di un interrupt esterno che si produce attivando il livello logico applicato al piedino NMI.

Si chiama interrupt "non mascherabile" perché è sempre accettato dalla CPU. È di priorità alta, e quando avviene ha accesso tramite il vector o ingresso 2 della tabella IDT.

È un interrupt asincrono perché non si conosce il momento in cui si origina, dato che dipende dalle condizioni e dai parametri del mondo esterno.

la routine è verificare quale di essa ha generato la richiesta.

Nelle 256 routines di interrupt possibili nel processore x86 a cui si fa riferimento mediante gli ingressi della tabella IDT, ce ne sono alcune che sono destinate a risolvere sempre la stessa causa di interrupt.

Ad esempio, l'ingresso 0 della IDT è riservato per risolvere una routine che riguarda gli errori che si producono nel compiere le operazioni di divisione con le istruzioni DIV e IDIV.

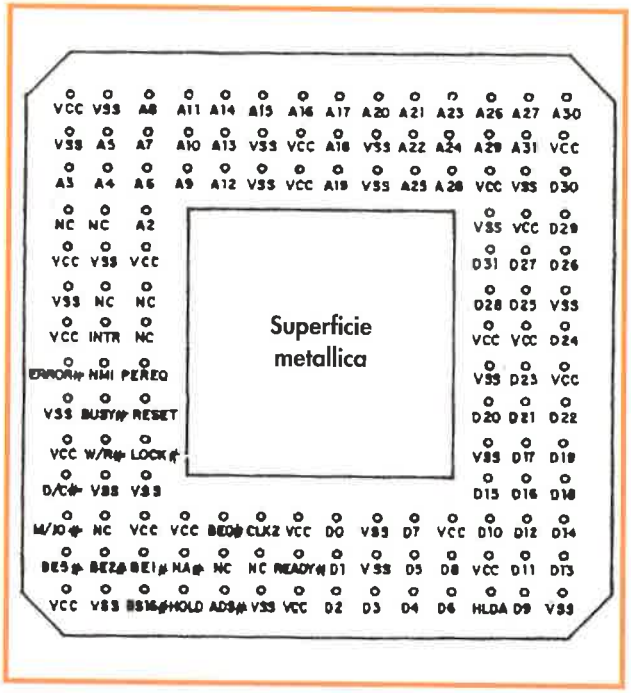


Diagramma dei piedini del 386 in cui si distinguono i due piedini riservati agli interrupt esterni NMI e INTR.

IL FAVOLOSO INTERRUPT INTR

In un capitolo precedente abbiamo analizzato il comportamento dell'interrupt INTR del microprocessore a otto bit 8085. Era capace di accedere fino a otto routines differenti. Nella famiglia x86 il piedino INTR può dare accesso a qualsiasi delle 256 routines che si selezionano agli ingressi della IDT. Quando la CPU accetta la richiesta INTR genera due cicli di bus di lettura, il secondo dei quali realizza la lettura del valore binario depositato negli otto bit meno significativi del bus dei dati (D0-D7). Questo valore è interpretato dalla CPU come ingresso della IDT con la quale risolvere la richiesta che arriva da INTR. In questo modo, con un solo piedino, si possono utilizzare tutte le 256 routines di interrupt possibili.

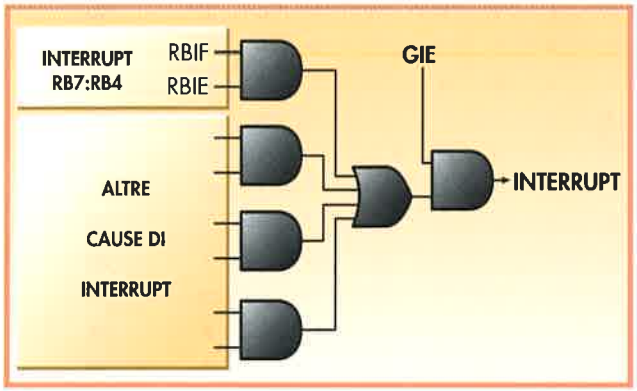
L'interrupt INTR è "mascherabile", questo significa che viene ignorata la richiesta quando il flag di maschera IF vale zero. Questo flag IF è indicato nel Registro di stato insieme ad altri flag del sistema.

Questa caratteristica ha la sua importanza nella costruzione dei programmi. Generalmente mediante il piedino INTR la CPU si dedica alle periferiche e alle risorse secondarie. Se in un pezzo di programma principale si risolvono azioni fondamentali per il buon funzionamento del sistema, non conviene che questo pezzo interrompa la sua esecuzione a causa di una richiesta a bassa priorità. Per evitare tali richieste si pone a zero il flag IF all'inizio di questa parte di istruzioni, proibendo gli interrupt in arrivo da INTR. Dopo aver completato la zona pericolosa del programma, si torna a mettere a 1 il flag IF con il quale si accettano nuovamente gli ingressi degli elementi secondari attraverso INTR.

Interrupt per cambio di stato in RB7:RB4

Una delle quattro cause che generano l'interrupt del PIC16F84, è il cambio di stato logico in uno dei quattro ingressi più significativi della Porta B (RB7:RB4).

Se ad un determinato momento in una di queste quattro linee, configurate come ingressi, cambia lo stato logico, il flag RBIF passa a 1, e se il bit di permesso di questo interrupt è attivo (RBIE =1) si genera un interrupt. Come per le altre cause di interrupt, per fare in modo che l'evento sia accettato, è necessario che il bit di Permesso Globale di Interrupt, GIE, sia a 1.



Quando cambia lo stato logico di una delle linee RB7:RB4 della Porta B, se i bit GIE e RBIE sono uguali a 1 si produce un interrupt.

Lo schema logico della figura esprime in modo grafico quanto descritto.

APPLICAZIONE DELL INTERRUPT

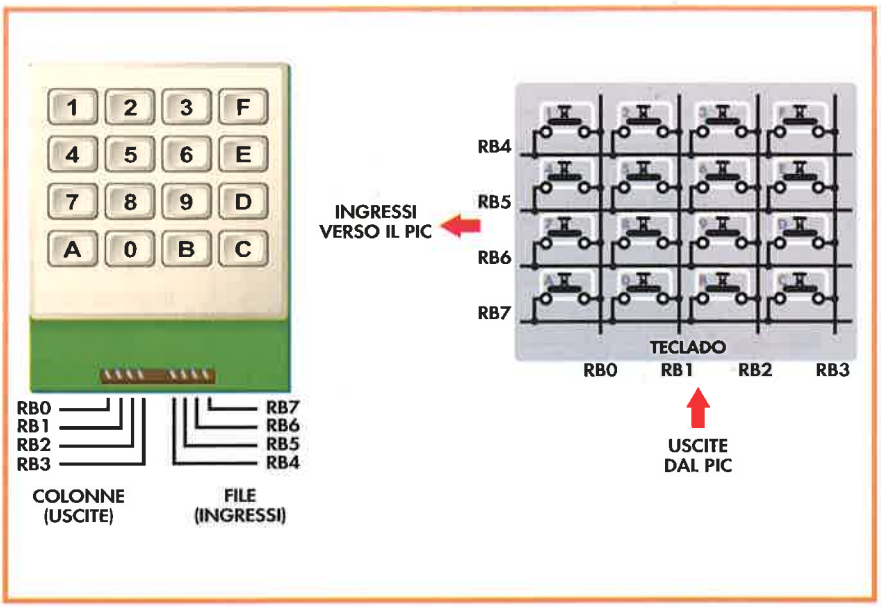
Questa strana causa di interrupt è stata sviluppata per gestire al meglio le tastiere matriciali, molto diffuse nelle applicazioni con i microcontroller. La tastiera è un dispositivo per inserire informazioni, facile da gestire, di ridotte dimensioni ed economico.

Una tastiera molto comune è quella a 16 pulsanti, configurata in una struttura matriciale composta da 4 linee per 4 colonne.

I pulsanti sono situati sull'incrocio delle linee con le colonne, quando si preme uno di essi si chiude il contatto fra la linea e la colonna che formano l'incrocio controllato dal suddetto pulsante. Quando non ci sono pulsanti premuti, le linee sono elettricamente isolate dalle colonne.

In pratica queste tastiere dispongono di 8 terminali di collegamento, 4 corrispondono alle file e si collegano con le linee RB7:RB4 della Porta B, che devono essere configurate come ingressi digitali.

Gli altri 4 terminali corrispondono alle colonne della tastiera, e si collegano ai piedini RB3:RB0 della porta B, che devono essere configurati come uscite digitali di informazioni dal microcontroller. Nella figura si mostrano i collegamenti delle file e delle colonne della tastiera con le linee della Porta B.



Struttura e collegamento della tastiera matriciale ai piedini della Porta B del PIC.

MICROCONTROLLER

Un interrupt molto particolare

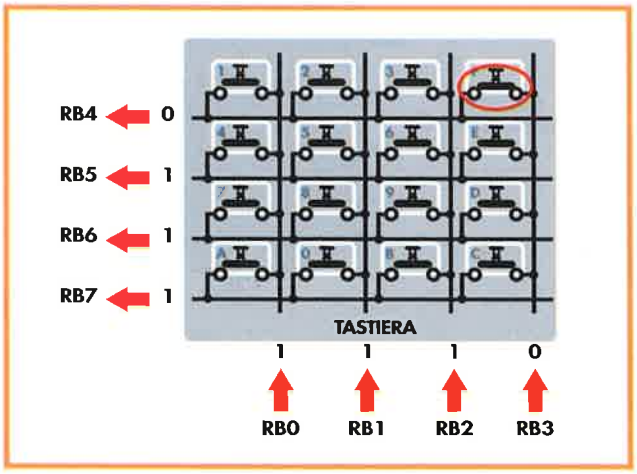
Per esplorare la tastiera e rilevare il pulsante premuto si applica il seguente metodo: il microcontroller deposita nelle 4 linee di uscita digitali (colonne) un codice formato da tre livelli logici alti (1) e un livello basso (0). Ad ogni scansione il codice sposta la posizione del livello logico basso in modo sequenziale, su ognuna delle 4 colonne collegate a RB3:RB0. Quando non c'è nessun pulsante premuto, le file essendo isolate dalle colonne, sono "in aria" e introducono un livello logico alto, dovuto ad una caratteristica costruttiva propria della tecnologia TTL. In questa situazione, quando si leggono gli ingressi digitali, che sono le file della tastiera, avremo quattro uno. Nel caso venga premuto un pulsante, quando il livello basso che esce dalle colonne, coincide con l'incrocio fila-colonna del pulsante premuto, per la fila corrispondente entrerà un livello logico basso, grazie al quale si potrà localizzare immediatamente il pulsante che è stato premuto, perché si produrrà un interrupt.

SCANSIONE DELLA TASTIERA

Il metodo usato per la rivelazione del pulsante premuto, utilizza il livello basso come livello attivo, mentre quello alto è considerato inattivo. Dalle quattro colonne collegate alle uscite RB3:RB0 escono quattro livelli logici, uno attivo, o basso, e tre inattivi, o alti. Se c'è un pulsante premuto nella colonna che ha il livello logico basso, la fila di incrocio riceve questo livello, che arriverà su una delle linee RB7:RB4. Se non ci sono pulsanti premuti, le file restano isolate dalle colonne, ed essendo sospese in aria, si comportano come se su tutte arrivasse un livello logico alto. Il livello basso in uscita da una delle colonne, viene spostato in modo sequenziale da una all'altra, e quando arriva alla fine il ciclo si ripete da capo. Dato che si conosce sempre da quale colonna si sta facendo uscire il livello basso, basterà conoscere la fila da cui detto livello viene ricevuto per riconoscere il pulsante di incrocio fra fila e colonna, come illustrato nella figura.

TASTIERA	FILE (INGRESSI)				COLONNE (USCITE)				CODICE HEX
	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	
0	0	1	1	1	1	1	0	1	7D
1	1	1	1	0	1	1	1	0	EE
2	1	1	1	0	1	1	0	1	ED
3	1	1	1	0	1	0	1	1	EB
4	1	1	0	1	1	1	1	0	DE
5	1	1	0	1	1	1	0	1	DD
6	1	1	0	1	1	0	1	1	DB
7	1	0	1	1	1	1	1	0	BE
8	1	0	1	1	1	1	0	1	BD
9	1	0	1	1	1	0	1	1	BB
A	0	1	1	1	1	1	1	0	7E
B	0	1	1	1	1	0	1	1	7B
C	0	1	1	1	0	1	1	1	77
D	1	0	1	1	0	1	1	1	B7
E	1	1	0	1	0	1	1	1	D7
F	1	1	1	0	0	1	1	1	E7

Tabella dei codici per verificare il pulsante premuto in funzione della presenza del livello logico basso nella fila e nella colonna.



Quando dalle colonne esce il codice 1110, se il pulsante premuto è F, sulle file si riceve il codice 0111.

Il programma di scansione metterà ciclicamente nelle colonne i quattro codici che possiedono un solo livello basso: 0111, 1011,1101,1110.

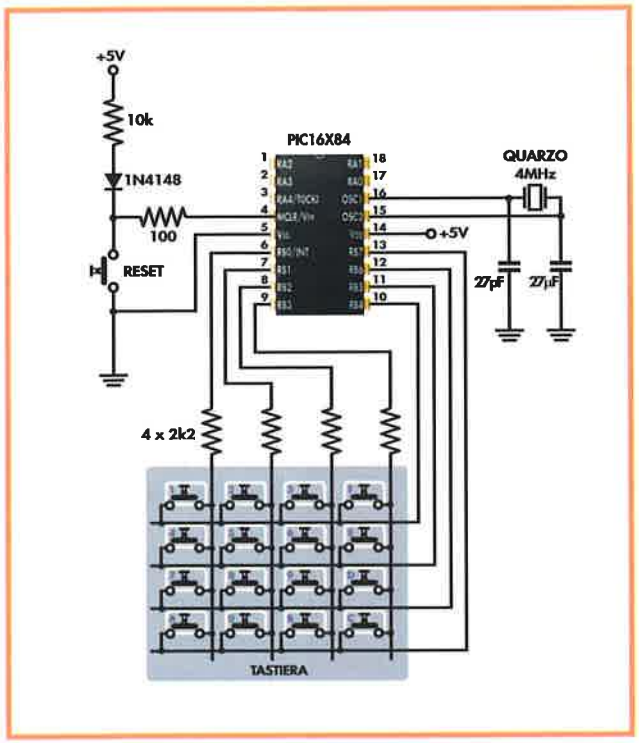
Per ognuno di questi codici in uscita viene verificato quello che si riceve dalle file che funzionano come ingressi digitali.

Se tutti gli ingressi sono a livello logico alto, significa che nessun pulsante della fila da cui sta uscendo lo zero è stato premuto. Nel momento in cui viene premuto un pulsante, nell'ingresso corrispondente all'incrocio fra la fila di questo pulsante e la colonna da cui sta uscendo lo zero, apparirà un livello logico basso.

Si può facilmente compilare una tabella che stabilisca il pulsante premuto in funzione della fila e della colonna che hanno il livello logico basso, come proposto dalla figura.

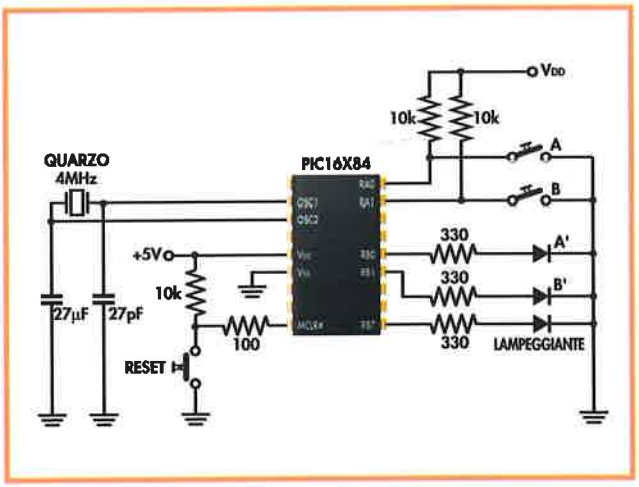
Per sfruttare i benefici dell'interrupt, si collegano le file della tastiera ai piedini RB7:RB4, che saranno configurati come ingressi. I terminali delle colonne si collegano ai piedini RB3:RB0, che saranno configurati come uscite. Si pone GIE = 1 per abilitare gli interrupt in generale, e anche RBIE =1 per abilitare l'interrupt per cambio di stato logico nelle linee RB7:RB4. Un semplice programma inserirà in modo sequenziale i codici con un livello basso sulle uscite RB3:RB0, che sono collegate alle colonne. Tutto qui.

Quando si premerà un pulsante cambierà lo stato logico su di una delle linee RB7:RB4 e si genererà un interrupt. Quindi si salterà ad una routine che inizia dall'indirizzo 004 della memoria di programma, e come prima cosa si verifica quale delle linee RB7:RB4 ha un livello basso. Con l'informazione della fila che ha ricevuto un livello basso, e della colonna che in quel momento aveva in uscita detto livello, consulteremo la tabella mostrata in precedenza, per dedurre il pulsante premuto.



Collegamento di un PIC16F84 ad una tastiera. Le linee RB3:RB0 sono uscite dalle quali esce il codice binario, e RB7:RB4 sono ingressi dai quali si riceverà un livello basso se c'è un pulsante premuto.

In commercio si può trovare una scheda commerciale per prototipi, chiamata PIC18-ME, progettata dalla ditta Ingegneria de Microsistemas Programmados S.L. per realizzare prototipi in modo sicuro, rapido e economico. Nella parte superiore contiene la circuiteria ausiliaria, insieme ad uno zoccolo a 18 pin, adatto a ricevere il PIC16F84 con il programma memorizzato. La parte inferiore è una zona libera con fori prestagnati ed a passo normalizzato, su cui è collegata una tastiera matriciale e un display a 7 segmenti. I collegamenti della tastiera seguono lo schema della tastiera mostrato in precedenza.



Schema del circuito descritto nell'enunciato del programma.

UN PROGRAMMA CHE GESTISCE L'INTERRUPT DEL TMR0

Per fare in modo che il lettore si abitui ad utilizzare nella pratica gli interrupt, proponiamo un programma che utilizza quello prodotto dal TMR0 quando va in overflow. In questo esempio, il TMR0 serve per controllare un tempo esatto di 1 secondo, momento in cui si genera l'interrupt e la CPU procederà a realizzare l'azione corrispondente.

Per ottenere la temporizzazione di 1 secondo, si carica il TMR0, con il valore iniziale di 12; in questo modo l'overflow si produrrà alla fine di (256-12) = 244 impulsi applicati al timer.

Questi impulsi arrivano dall'oscillatore interno del PIC, che funziona ad una frequenza di 4 MHz, però divisi per 256 grazie al divisore di frequenza, il che suppone una temporizzazione di:

Temporizzazione = 4 . Tosc . Valore del conteggio di TMR0 . Range del divisore di frequenza

Temporizzazione = 4 . 250 ns . 244 . 256 = 62,4 millisecondi

Un interrupt molto particolare

Listato del programma con commenti

```
                LIST    P=16F84                 ;Direttive per Assembler
                RADIX   HEX

                W       EQU     0               ;Etichette
                F       EQU     1
                TMR_OPT EQU     0x01
                STATO   EQU     0x03
                PORTAA  EQU     0x05
                PORTAB  EQU     0x06
                INTCON  EQU     0x0B
                CONTA   EQU     0x10            ;Contatore ausiliario

                ORG     0                       ;Vector reset
                goto    inizio                  ;Salto al vettore di interrupt
                ORG     4                       ;Vettore di interrupt
                goto    inter                   ;Salto all'inizio della routine di interrupt
                ORG     5

inizio          bsf     STATO, 5                ;Selezione del banco 1
                clrf "  PORTAB                  ;Porta B uscita
                movlw   b'00000011'             ;Configura RA0 e RA1 come ingressi
                movwf   PORTAA
                movlw   b'00000111'             ;Configura il registro OPTION
                movwf   TMR_OPT
                bcf     STATO, 5                ;Seleziona il banco 0
                movlw   b'10100000'             ;Abilitazione globale per gli interrupt e per il TMR0
                movwf   INTCON
                movlw   0x10
                movlw   CONTA                   ;Carico CONTA con il valore 16 in decimale
                movlw   0x0c
                movwf   TMR_OPT                 ;Carico TMR0 con 12 in decimale
ciclo           btfsc   PORTAA, 0               ;Se RA = 0 salto la prossima istruzione
                goto    ra0_1
                bcf     PORTAB, 0               ;Se RA0 = 0, RB0 = 0
                goto    ra1x                    ;Test su RA1
ra0_1           bsf     PORTAB, 0               ;Se RA0 = 1 si porta RB0 = 1
ra1x            btfsc   PORTAA, 1               ;Se RA1 = 0 salto
                goto    ra1_1                   ;Salta se RA1 = 1
                bcf     PORTAB, 1               ;Se RA1 = 0, RB = 1
                goto    ciclo2
ra1_1           bsf     PORTAB, 1               ;Se RA1 = 1,RB1 = 1
ciclo2          goto    ciclo                   ;Ciclo infinito dal quale si esce per interrupt

inter           decfsz  CONTA, 1                ;Decrementa CONTA e salta se vale 0
                goto    continua
conta_0         movlw   0x10                    ;Se CONTA = 0 si carica con 16
                movwf   CONTA
                btfsc   PORTAB, 7               ;Se RB7 = 0, salta
                goto    rb7_1
                bsf     PORTAB, 7               ;Se RB7 = 0 si inverte il valore
                goto    continua
rb7_1           bcf     PORTAB, 7               ;Se RB7 = 1 si inverte il valore
continua        movlw   b'10100000'             ;Si ripristina INTCON, interrupt disabilitati
                movwf   INTCON
                movlw   0x0C                    ;Si ricarica TMR0 con 12 in decimale
                movwf   TMR_OPT
                retfie                          ;Ritorno dall'interrupt
                end
```

Dato che non si arriva ancora ad 1 secondo, si impiega un contatore ausiliario CONTA, che deve essere inizializzato con valore 16 e si decrementa ad ogni interrupt del TMR0, cioè ogni 62,4 ms. Arriverà a zero in 16 x 62,4 ms = 1 secondo (approssimativamente).

ENUNCIATO

Comporre un programma in linguaggio Assembler per il PIC16F84, il cui oscillatore interno sia controllato da un cristallo di quarzo da 4 MHz.

Alle linee RA0 e RA1 della Porta A saranno collegati due interruttori A e B, i quali devono essere testati continuamente, e il loro stato (aperto o chiuso) riportato su due diodi LED A' e B' collegati alle linee RB0 e RB1 della Porta B.

Inoltre, un altro diodo LED sulla linea RB7 deve lampeggiare in continuazione con un tempo programmato di un secondo, cioè resterà acceso un secondo e spento il secondo successivo. Questo serve per controllare il tempo in cui avviene l'interrupt del TMR0 che è stato spiegato in precedenza.

La delicata operazione di scrittura nella EEPROM

La EEPROM dei dati del PIC16F84 ha una capacità di soli 64 byte, ma il suo pregio sta nella non volatilità dei dati. Scollegando l'alimentazione, l'informazione memorizzata nelle sue celle viene mantenuta; questa caratteristica rende questa memoria indispensabile in molte applicazioni commerciali.

Come già detto in precedenza, ogni indirizzo di questa memoria può essere scritto in modo indipendente, anche se la grande quantità di tempo (relativo) che richiede questa operazione, a volte rappresenta un problema. Scrivere un byte nella EEPROM può richiedere sino a 10 ms. Questo tempo è enorme, se comparato con la velocità di lavoro del microcontroller; per dare un'idea basti pensare che un PIC16F84 funzionante con quarzo da 4 MHz può eseguire 10.000 istruzioni normali nel tempo in cui si scrive un byte nella EEPROM.

Il tempo di scrittura di una cella della EEPROM non è costante, dipende dalla temperatura, dalla frequenza di campionamento, e da altri parametri; però questo tempo è critico, perché se non si compie non si riesce a scrivere in modo corretto il byte nella memoria, e il sistema resta degradato e inservibile. Pertanto è assolutamente necessario verificare con certezza che l'operazione di scrittura termini correttamente.

OPERATIVITÀ DELL'INTERRUPT

Così come per le altre tre cause di interrupt nel PIC16F84, l'interrupt per fine della scrittura nella EEPROM, esige la gestione di tre bit:

- GIE: Permesso Globale di Interrupt.
- EEIE: Abilitazione dell'interrupt per fine scrittura nella EEPROM.
- EEIF: Flag di fine scrittura della EEPROM.

Quando termina l'operazione di scrittura di un byte nella EEPROM, il flag EEIF viene messo a 1 automaticamente. Sino a che l'operazione di scrittura non è stata completata, non è possibile ripetere l'operazione stessa, inoltre conviene verificare che non siano stati prodotti errori durante la sua realizzazione, per questo test esiste un flag dedicato nel registro di controllo della EEPROM.



I bit di permesso dell'interrupt GIE e EEIE devono essere a 1 per permettere un interrupt quando termina la scrittura della EEPROM e il flag EEIF passa a 1.

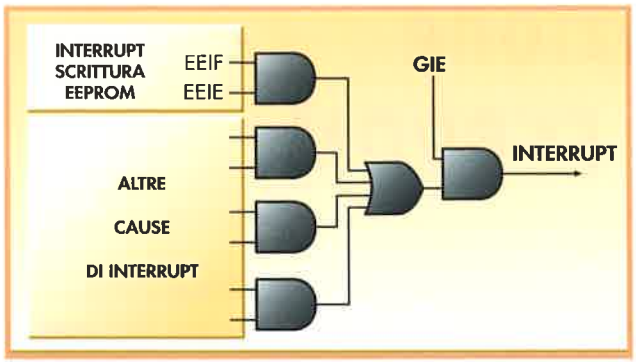
Per poter conoscere immediatamente la fine delle operazioni, è raccomandabile attivare un interrupt in cui il processore verifichi che le procedure siano terminate correttamente e possa scrivere gli eventuali byte successivi.

I bit GIE e EEIE sono ubicati nel registro INTCON, mentre il flag EEIF si trova nel bit 4 del registro EECON1; questo flag viene messo automaticamente a 1 quando termina la scrittura.

Dobbiamo considerare che quando si utilizza un interrupt per fine scrittura nella EEPROM, all'interno della routine dedicata, prima dell'istruzione di ritorno dobbiamo porre il bit EEIF = 0, perché in caso contrario appena ritornati al programma principale si ripeterebbe la chiamata alla routine di interrupt. Nella figura è riportata la situazione dei bit che gestiscono l'interrupt per fine scrittura.

Quando GIE = EEIE = 1, e termina la scrittura nella EEPROM, il bit EEIF si pone a 1 e si genera un interrupt. La routine dedicata deve esaminare se vi sono stati eventuali errori, testando il bit che lo segnala, e successivamente porre EEIF = 0 prima del ritorno al programma principale.

Una buona tecnica è verificare che sia stato scritto nell'indirizzo voluto il valore corretto. Nello schema logico della figura si mostra il modo in cui si genera un interrupt per fine scrittura nella EEPROM.



Schema logico che mostra il controllo della generazione di interrupt per fine scrittura nella EEPROM.

ESERCIZIO DI GESTIONE DELL'INTERRUPT

Il microcontroller è in continua esecuzione del programma a cui è stato dedicato, il quale controlla molti elementi esterni che fanno parte dell'ambiente di lavoro. Questi dispositivi sono influenzati dalle caratteristiche e dai parametri di molti fattori, sia interni che ambientali. Se ad esempio stiamo controllando un motore, la velocità di giro dell'asse è un fattore fondamentale per lo sviluppo delle sue operazioni, pertanto dobbiamo verificare continuamente che rimanga all'interno di margini adeguati, un altro fattore fondamentale potrebbe essere la temperatura. Sia la velocità di giro che la temperatura sono valori di cui deve tener conto il microcontroller, e reagire immediatamente in caso vengano superati i limiti stabiliti. Per poter rispondere in modo immediato alle condizioni esterne, il microcontroller dispone degli interrupt. Nel caso del PIC16F84 sono state descritte quattro cause che possono generare interrupt:

1ª. Che si attivi il piedino RB0/INT, ad esempio quando si rileva un valore limite di una grandezza.

2ª. Che vada in overflow il TMR0 al termine di un periodo di tempo.

3ª. Che cambi lo stato logico di uno dei quattro ingressi più significativi della Porta B.

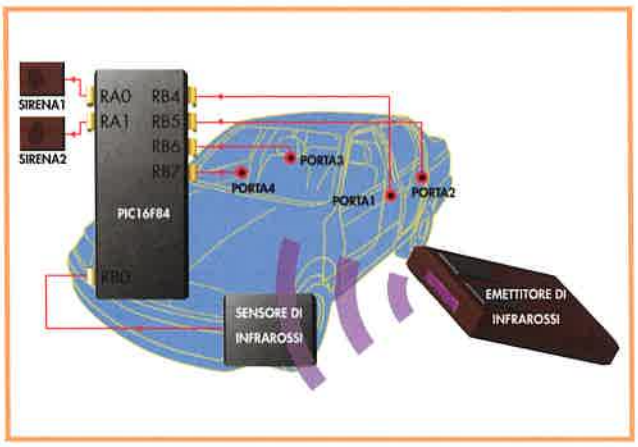
4ª. Che sia terminata la scrittura di un byte in una cella della EEPROM.

Quando si progetta un'applicazione si deve adattare una di queste possibili cause di interrupt alla grandezza esterna che si intende controllare, per intervenire su di essa immediatamente. Nel caso di un sensore digitale, che cambia il suo stato logico al variare della situazione che sta controllando, come ad esempio un sensore optoelettronico che rileva il passaggio di un oggetto, si può collegare direttamente l'uscita del sensore al piedino RB0 in modo che generi un interrupt. Quando il parametro è analogico, può essere necessaria la sua conversione in digitale, mediante un ADC (Analog to Digital Converter), oppure l'utilizzo di un comparatore che cambi lo stato della sua uscita se il valore analogico supera una soglia determinata. Per chiarire al lettore la gestione degli interrupt sia nell'aspetto fisico che in quello logico descriveremo un progetto basato sul PIC16F84.

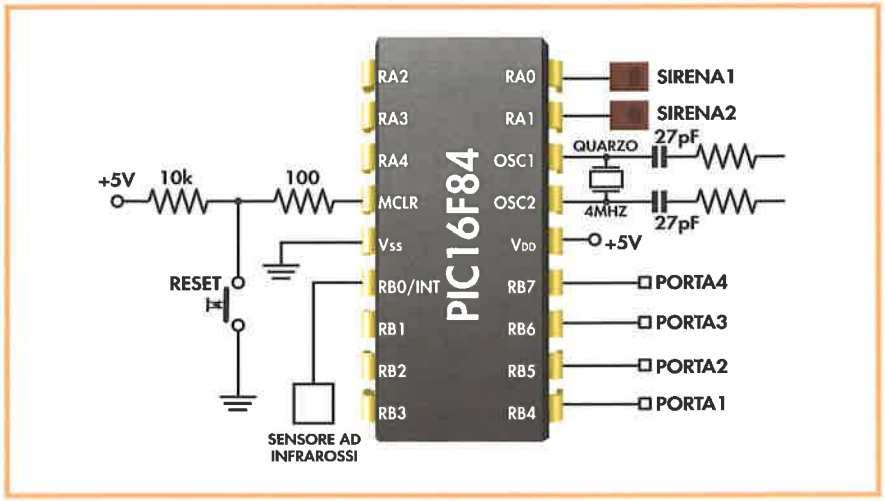
UN ALLARME PIENO DI INTERRUPT

Prendiamo in considerazione il progetto di un allarme antifurto per un'automobile. Per controllare l'apertura delle quattro porte principali della macchina si utilizzano quattro sensori a contatto che saranno collegati alle quattro linee più significative della Porta B (RB4-RB7). Quando si attiva l'allarme, l'apertura di una qualsiasi delle porte genera un cambio di stato in uno dei pin RB4-RB7, che saranno programmati in modo interrupt, originando il medesimo. La routine di interrupt ha il compito di attivare due sirene o altoparlanti di grande potenza, controllati dai piedini RA0 e RA1 della Porta A del PIC16F84. Questi elementi acustici resteranno in funzione sino a che un sensore ad infrarossi riceverà il comando adeguato originato da un telecomando a distanza. Questo sensore sarà collegato al piedino RB0/INT, come mostrato dalla figura.

Lo schema elettrico corrispondente all'allarme antifurto, è riportato nella figura alla pagina seguente, in cui sono stati aggiunti i circuiti ausiliari di alimentazione, clock e reset.



Schema generale del sistema di allarme antifurto.



Schema elettrico semplificato dell'allarme antifurto.

Nell'allarme per l'automobile abbiamo un programma principale che si occupa di configurare le porte per adattarle alle periferiche, dopo di che entra in un ciclo di attesa da cui non esce se non si attiva l'allarme per l'apertura di una porta, o si scollega l'allarme. Queste piccole operazioni, si riflettono nell'organigramma della figura.

L'implementazione, con le istruzioni del PIC, dei blocchi che compongono l'organigramma, si realizza con i seguenti passi:

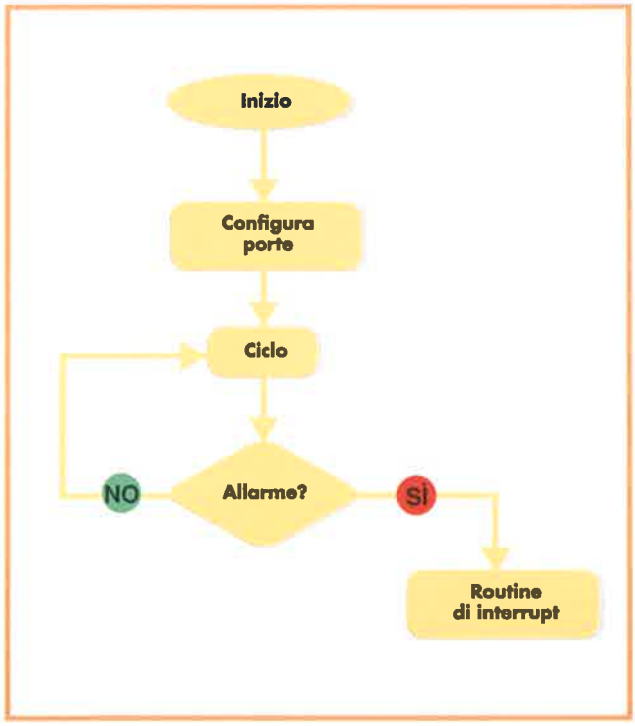
Per chiarire la programmazione del registro INTCON per l'abilitazione degli interrupt, riportiamo nella figura la struttura interna del medesimo.

Nella pagina a lato svilupperemo il programma di allarme, con i gruppi di istruzioni divisi nelle parti principali di cui è composto.

L'IMPORTANZA DEGLI ORGANIGRAMMI

Un buon programmatore ha la capacità di ordinare il lavoro di sviluppo in parti, o moduli, di piccole dimensioni. In questo modo al momento di scrivere le istruzioni farà riferimento a piccoli moduli, facili da eseguire e da capire, e rapidi da editare.

Per rappresentare comodamente la divisione del problema da risolvere è fortemente consigliabile utilizzare gli organigrammi.



Organigramma del programma principale.

1°. Definizione delle caratteristiche e registri del PIC.

2°. Definizione di inizio del programma principale e inizio del programma di interrupt.

3°. Configurare la porta A come uscita.

4°. Configurare la porta B come ingresso.

5°. Resettare le due porte.

6°. Attivare i bit di permesso dell'interrupt, il bit di permesso globale, quello di interrupt esterno tramite RB0 e quello per il cambio di stato nelle linee RB4-RB7.

7°. Ciclo di attesa che si generi un interrupt.



Struttura interna del registro di controllo degli interrupt INTCON.

PROGRAMMA PRINCIPALE DI ALLARME

Definizione delle caratteristiche e dei registri del PIC

```
        LIST        p=16F84
        RADIX       HEX
        W           EQU         0
        F           EQU         1
        STATO       EQU         0x03
        PORTAA      EQU         0x05
        PORTAB      EQU         0x06
        INTCON      EQU         0x0B
```

Definizione degli indirizzi di inizio del programma principale e della routine di interrupt

```
        ORG                     0
        goto                    inizio
        ORG                     4
        goto                    inter
        ORG                     5
```

Configurazione delle porte di ingresso e uscita

```
inizio  bsf                     STATO,5
        movlw                   b'0000 0000'
        movwf                   PORTAA
        movlw                   b'1111 1111'
        movwf                   PORTAB
        clrf                    PORTAA
        clrf                    PORTAB
```

Bit di abilitazione dell'interrupt

```
        movlw       b'10011000'
        movwf       INTCON
ciclo   goto        ciclo       ;Ciclo infinito in attesa di interrupt
```

Programma principale

```
inter   btfss       INTCON,0
        goto        fermare
        goto        allarme

allarme clrf        PORTAB
        movlw       b'10011000'
        movwf       INTCON

sirena  bsf         PORTAA,0
        NOP
        bcf         PORTAA,0
        bsf         PORTAA,1
        NOP
        bcf         PORTAA,1
        goto        sirena

fermare clrf        PORTAA
        bcf         PORTAB,0
        movlw       b'10011000'
        movwf       INTCON
        goto        ciclo

        end
```

L'importanza del consumo

Abbiamo un distributore di bibite in mezzo alla spiaggia. Ha avuto una buona accoglienza, e la sua gestione è stata un successo. Però lì non abbiamo una presa di corrente, e il distributore deve funzionare con una batteria, o delle pile ricaricabili.

Evidentemente è fondamentale che il consumo della macchina sia molto basso, per evitare le continue sostituzioni delle batterie, e periodi con mancanza di servizio. Inoltre funziona realmente da mezzogiorno fino a sera, fuori da questo periodo si avvicina solo qualche cliente occasionale.

Non c'è attività nemmeno nei giorni in cui manca il sole, quindi ci sono lunghi periodi in cui la macchina non funziona.

Per ridurre il consumo sarebbe auspicabile che la circuiteria elettronica consumasse energia solamente quando deve fornire il servizio. Il caso del cellulare, è simile a quello appena esposto, trascorre lunghi periodi di tempo in cui non è utilizzato, in attesa di ricevere qualche chiamata.

Nella maggioranza delle attrezzature portatili alimentate con pile, e che non dispongono di energia elettrica, conviene ridurre il consumo al minimo nei periodi in cui sono in attesa di realizzare un'operazione.

IL MODO RIPOSO O DI BASSO CONSUMO NEL PIC16F84

Il nostro microcontroller può rimanere molto tempo in uno stato in cui consuma una quantità insignificante di energia elettrica.

Questo modo si chiama di riposo o di basso consumo, e la corrente di alimentazione tipica, che è circa 2 milliampere, passa ad essere inferiore a 10 microampere; questo significa che è possibile alimentarlo con una piccola pila, che può durare sino a due anni.

Questa caratteristica lo rende molto adatto a quelle applicazioni in cui ci siano lunghi periodi di inattività in attesa che succeda qualche evento esterno, come l'attivazione di un pulsante. Durante questa condizione si dice che il microcontroller sta dormendo.



Il telefono cellulare resta molto tempo in stato di attesa.

MICROCONTROLLER

Il modo riposo

MODO RIPOSO	
STATO RISORSA	**CONSEGUENZA**
Oscillatore principale fermo	- Non si eseguono istruzioni - Non funziona il TMR0 (RAM/T0CKI collegato a VDD)
WDT	- Si resetta però continua a funzionare
Linee I/O congelate	- Mantengono lo stato precedente
Consumo minimo	- Passa da 2mA a meno di 10μA

Tabella che raccoglie le principali conseguenze degli ingressi nel modo riposo.

Per addormentarsi al microcontroller basta eseguire l'istruzione SLEEP, di conseguenza il primo elemento che si ferma è l'oscillatore principale, lasciando il compito di produrre gli impulsi di lavoro alla frequenza Fosc/4.

In questa condizione il processore non esegue alcuna istruzione, né tantomeno funziona il TMR0 dato che non riceve impulsi interni.

Per fare in modo che il TMR0 non funzioni con gli impulsi esterni, si porta il piedino RA4/T0CLKI al positivo. In questo stato dobbiamo evidenziare che il Watchdog che rimane in funzione, entrando in modo riposo viene resettato, o reinizializzato, però continua a funzionare, dato che dispone di un oscillatore proprio e indipendente. In questo caso i flag del registro STATUS resteranno TO# = 1 e PD# = 0.

Le linee di ingresso e uscita mantengono lo stato precedente all'ingresso in modo riposo, e quelle che non sono collegate a delle periferiche e che assumono lo stato di alta impedenza si raccomanda di collegarle al positivo o a terra, per evitare possibili fughe di corrente. Vedere la tabella della figura.

IL MOMENTO DEL RISVEGLIO

Il microcontrollore si addormenta eseguendo l'istruzione SLEEP nel programma e si sveglia, riprendendo il suo funzionamento normale, per le seguenti cause:

1°. Per attivazione del piedino MCLR#.

2°. Per l'overflow del Watchdog.

3°. Quando si genera un interrupt.

Quando si attiva il piedino MCLR# e il PIC16F84 è stand-by, si sveglia e inizia ad eseguire la prima istruzione del programma, che risiede nel Vector Reset. Lo stesso succede quando va in overflow il WDT: se il microcontroller stava dormendo, si sveglia e si reinizializza.

Per determinare quale delle due cause ha originato il Reset, si testano i flag TO# e PD# del registro STATUS, seguendo i valori indicati dalla tabella della figura.

Quando si sveglia il PIC si sviluppa la sequenza del

VALORE DEL FLAG	INDICAZIONE
PD#=0	- Esecuzione SLEEP. Modo riposo
PD#=1	- Si è svegliato
TO#=0	- È andato in overflow il WDT
TO#=1	- Non è andato in overflow il WDT

Stato e significato dei flag PD# e TO#.

temporizzatore OST, che ritarda di 1024.Tosc l'inizio del funzionamento normale, con l'obiettivo di dare il tempo alla frequenza di funzionamento di stabilizzarsi.

IL RISVEGLIO CON INTERRUPT

La terza causa che può fare risvegliare il PIC avviene quando si verifica un interrupt. Dei quattro possibili interrupt che ammette il PIC16F84, quella relativa al TMR0 non è possibile, dato che mancano gli impulsi, quindi sono valide solo le altre tre opzioni (vedi tavola in alto alla pagina seguente).

Quando venne eseguita l'istruzione SLEEP, il PC si fermò caricato con l'indirizzo dell'istruzione successiva (PC+1).

RISVEGLIO CON INTERRUPT

STATO DEI BIT	ISTRUZIONI SUCCESSIVE
GIE= 0 (Inibizione) RBIE= INTE= EEIE= 1	PC + 1 (Successiva a sleep) PC + 2 (Successiva)
GIE= 1 (Abilitazione) RBIE= INTE= EEIE= 1	PC + 1 (Successiva a sleep) 0004 (Vector di Interrupt)

Istruzioni che esegue il PIC16F84 quando si sveglia tramite un interrupt a seconda del valore del bit GIE.

Il modo riposo

1°. Attivazione del piedino RB0/INT.

2°. Cambio di stato in uno dei piedini RB7:RB4.

3°. Fine della scrittura nella EEPROM.

Per accettare uno dei tre interrupt è necessario che i relativi bit di abilitazione (INTE, RBIE, e EEIE) siano a 1. In questo caso quando si attiva uno dei flag di interrupt si testa il valore del bit GIE, che è il bit di abilitazione globale degli interrupt.

Se GIE = 1 e quindi gli interrupt sono abilitati, il PIC si sveglia ed esegue l'istruzione (PC+1) successiva a SLEEP, e infine si carica nel PC l'indirizzo 0004 H, saltando alla routine di interrupt.

Dato che a volte non si desidera eseguire nessuna istruzione prima di saltare alla routine di interrupt, dopo SLEEP si inserisce un'istruzione NOP.

Se si genera un interrupt abilitato a livello individuale ma non a quello globale, cioè con GIE = 0 (Inibizione Globale), il PIC si sveglia dallo stato di riposo, e continua l'esecuzione del programma a partire dall'istruzione successiva a SLEEP come mostrato nella tabella della figura.

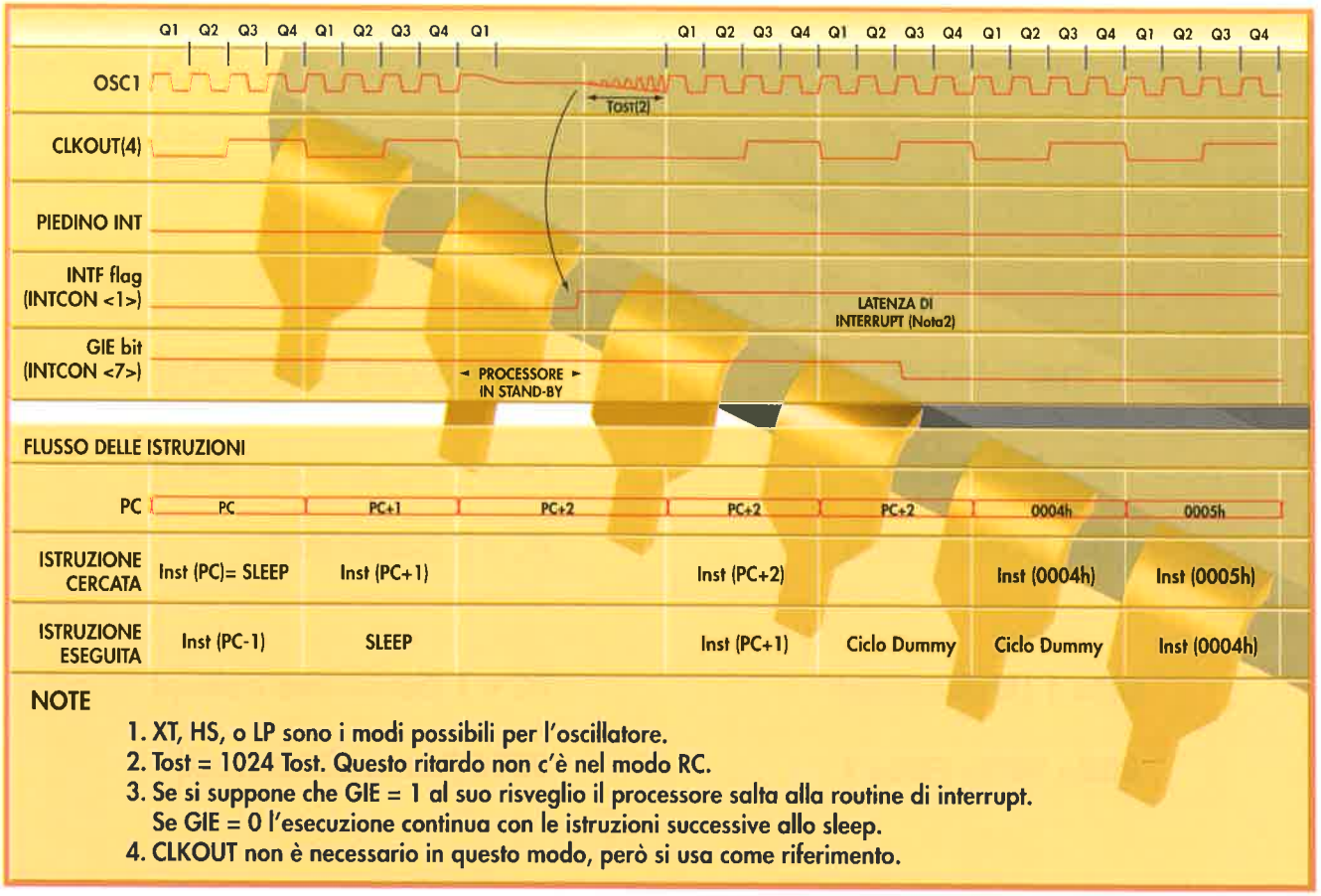
Quando GIE = 0 e si attiva il flag di un interrupt che ha 1 sul bit di abilitazione individuale, può verificarsi una delle seguenti condizioni:

1°) Se l'interrupt avviene prima dell'istruzione SLEEP, questa si completerà come se si trattasse di un'istruzione NOP. Non si cancella né il WDT né il suo Divisore di frequenza. Il bit TO# non viene messo a 1 e il bit PD# non si cancella.

2°) Se l'interrupt avviene durante o dopo l'istruzione SLEEP, il microcontroller si sveglierà immediatamente dopo aver eseguito l'istruzione SLEEP. Si cancella il WDT e il suo Divisore di frequenza, il bit TO# passa a 1 e il PD# passa a 0.

Per assicurare la cancellazione del WDT quando si esegue l'istruzione SLEEP, si raccomanda di mettere davanti ad essa l'istruzione CLRWDT.

Nella figura è presentato un cronogramma che illustra la forma di risveglio del PIC tramite interrupt.



Cronogramma del risveglio del PIC al prodursi di un interrupt.

Il Micro PIC Programmer: uno scrittore di PIC

LA SCRITTURA DEL PROGRAMMA NEL MICROCONTROLLER

Un microcontroller è un piccolo computer, che esegue il programma che si trova scritto nella sua memoria, per assolvere al compito a cui è stato destinato. La memoria dove risiede il programma non deve essere volatile, le istruzioni devono rimanere inalterate anche se viene tolta l'alimentazione.

Quando il programma risiede nella memoria ROM o in un modello OTP (One Time Programmable), non si può né cancellare né modificare; mentre se la memoria di programma è di tipo EPROM, EEPROM oppure FLASH, si può scrivere, cancellare, e riscrivere un numero di volte molto grande.

La scrittura del programma sorgente, e la sua successiva compilazione, sono fatti su di un personal computer, utilizzando i programmi necessari forniti dal fabbricante del microcontroller o da una ditta esterna. Una volta ottenuto il codice binario, o esadecimale, che deve essere memorizzato nel PIC, si passa alla fase di scrittura, tramite l'hardware esterno, collegato ad una porta del PC, da cui si scarica il programma in questione, con l'ausilio di un software di comunicazione fra il PC e lo scrittore.

CARATTERISTICHE DEL MICRO PIC PROGRAMMER

Per scrivere i PIC, oltre ai dispositivi forniti dal fabbricante, ve ne sono molti altri sviluppati da ditte esterne. Per questa spiegazione abbiamo scelto il programmato-



Il kit dello scrittore MICRO PIC PROGRAMMER
è composto da una scheda elettronica, un dischetto con il programma, il cavo per il collegamento al PC e il manuale d'uso in italiano.

MICROCONTROLLER

Uno scrittore di PIC



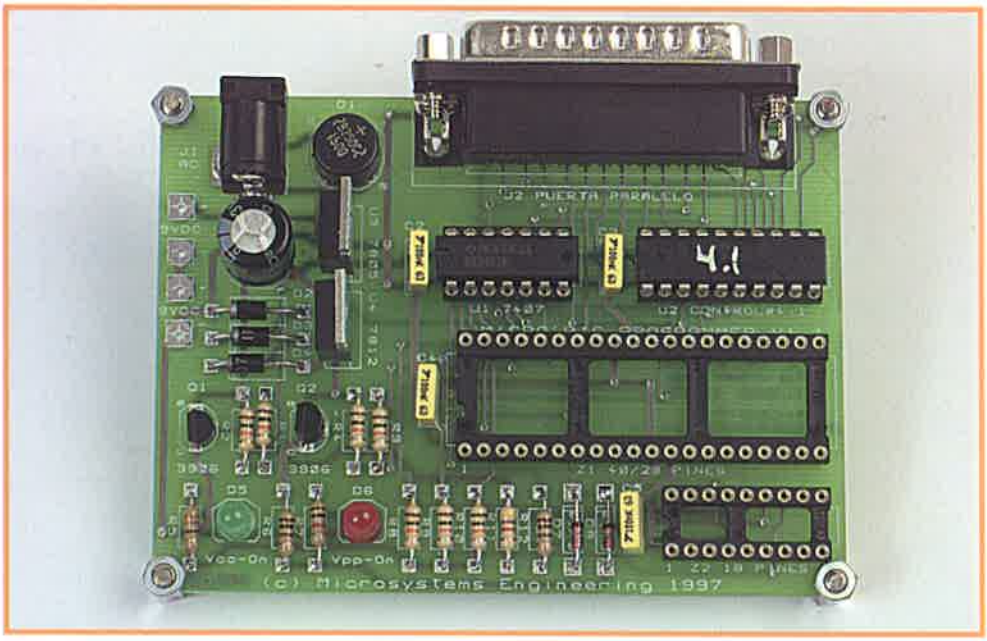
Fotografia del Micro PIC Programmer collegato alla porta parallela del PC.

re di PIC MICRO PIC PROGRAMMER della ditta "MSE Microsystem Engineering".

Si tratta di uno scrittore di PIC di gamma media, caratterizzato dalle sue ridotte dimensioni, dal basso prezzo e dalla semplicità di utilizzo. Fra i PIC che può scrivere troviamo il PIC16F84, e i moderni PIC16F87x.

È basato su una scheda in cui si trova tutta la circuiteria elettronica, compresi gli zoccoli su cui inserire il PIC che si desidera programmare.

Il sistema comprende anche un cavo a 25 fili da collegare alla porta parallela



Fotografia della scheda elettronica del Micro PIC Programmer. Anche se dispone solo di due zoccoli liberi, esistono molti adattatori per gli altri tipi di contenitori del PIC.

del computer. L'impiego di questa porta permette una maggiore velocità di scrittura e una maggiore protezione del PC, dato che la scheda dispone di alimentazione autonoma.

Inoltre è incluso anche un dischetto con il programma di comunicazione e controllo PICME-PR, e le informazioni tecniche sull'utilizzo e sul funzionamento.

Questo programmatore dispone di un circuito per la stabilizzazione delle tensioni di alimentazione, quindi può funzionare sia con due pile da 9 VDC che con un trasformatore da 220/15 VAC e una corrente uguale o superiore a 0,5 A.

ACCESSORI

Anche se sulla scheda del MICRO PIC PROGRAMMER ci sono solamente due zoccoli da 18 e da 40 piedini per inserire i PIC da programmare, esistono diversi adattatori che supportano i modelli a 28 piedini, con contenitori di differente larghezza, permettendo l'uso del programmatore con la maggior parte dei PIC della gamma media. Per proteggere i piedini del microcontroller durante la fase di scrittura, in cui bisogna inserire e togliere il circuito integrato dallo zoccolo del programmatore, si possono utilizzare zoccoli a "forza di inserzione nulla", come quello della fotografia.

INSTALLAZIONE E START-UP

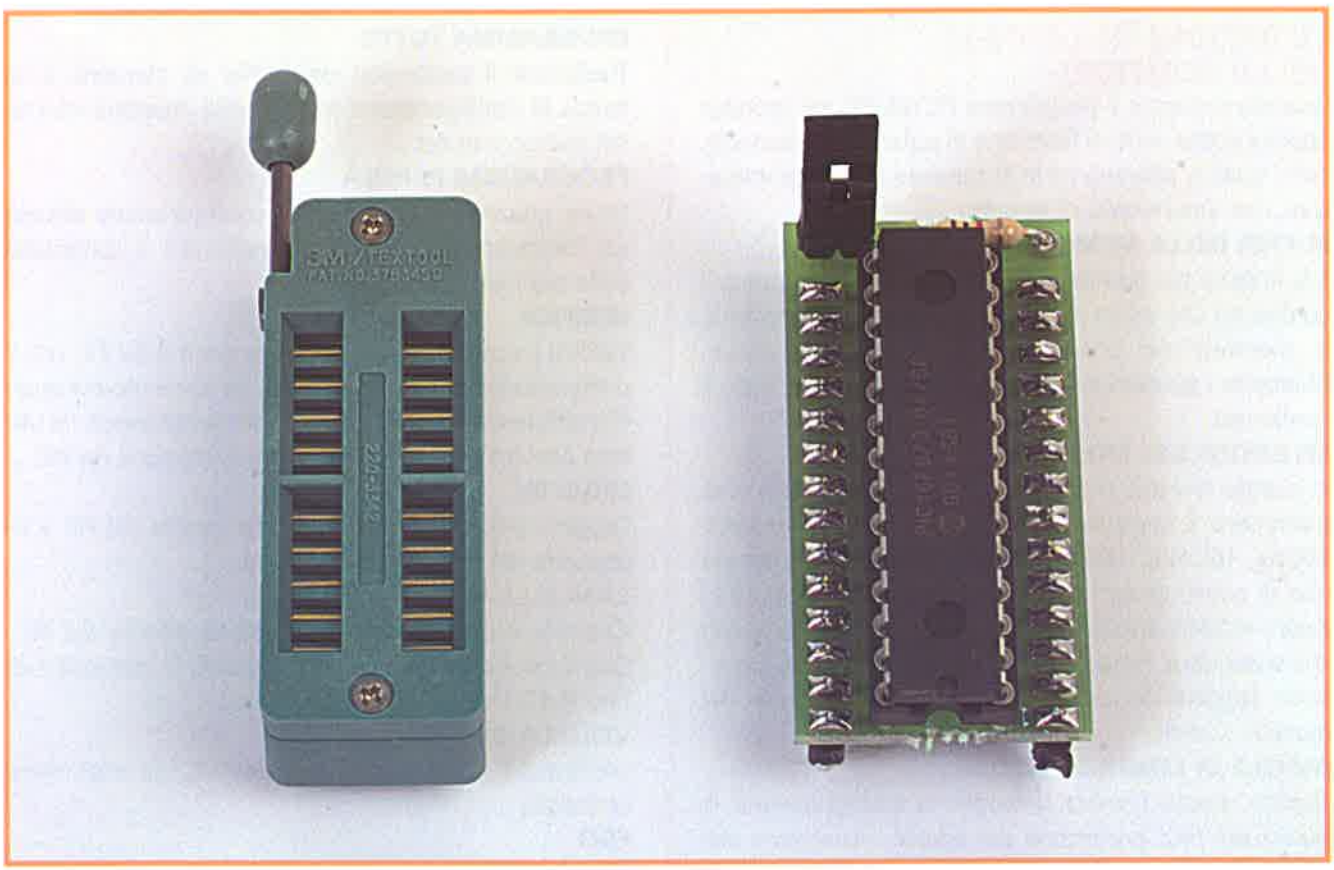
L'inizializzazione del funzionamento del MICRO PIC PROGRAMMER è molto semplice e rapida, e consta di quattro passi:

1° Passo: Copiare il software PICME-PR del dischetto, in una subdirectory sull'hard disk del PC.

2° Passo: Collegare il cavo da 25 fili fornito insieme al programmatore alla porta parallela del PC.

3° Passo: Fornire la tensione di alimentazione allo scrittore, o con due pile da 9 VDC o con un trasformatore da 220/15 VAC e una corrente superiore a 0,5 A.

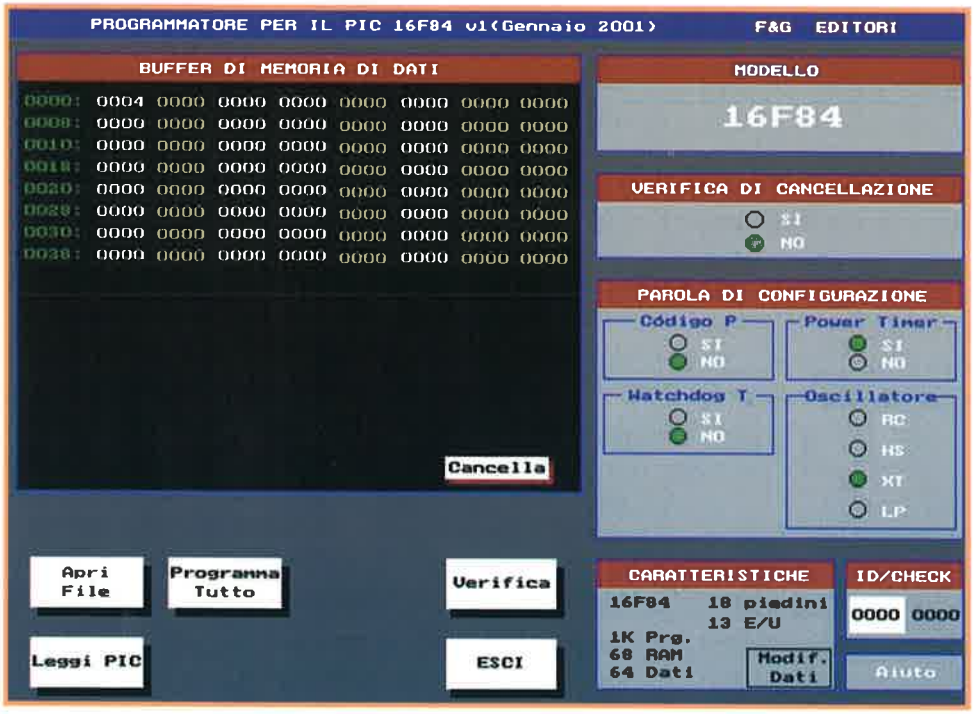
4° Passo: Eseguire dal PC il programma PICME-PR con il quale si possono realizzare le funzioni di lettura, scrittura, verifica, cancellazione, ecc. del microcontroller. Dovrà apparire nel monitor del PC la finestra di lavoro riportata in figura.



Fotografia dello zoccolo a forza di inserzione nulla, e un adattatore per un PIC con contenitore diverso.

MICROCONTROLLER

Uno scrittore di PIC



Videata di lavoro del programma PICME-PR.

LE OPZIONI DI LAVORO DELLO SCRITTORE

Quando si esegue il programma PICME-PR, sul monitor appaiono una serie di finestre e di pulsanti di comando, con i quali si attivano tutte le funzioni del programmatore, che descriviamo di seguito:

BUFFER DELLA MEMORIA DI PROGRAMMA

È la finestra più grande della videata, e in essa appare il contenuto che arriva da, o che deve essere trasferito a, la memoria del codice interno del microcontroller. Mediante i pulsanti adeguati si può visualizzare tutto il contenuto.

SELEZIONE DEL MODELLO

In questa finestra si sceglie il PIC che si desidera programmare. È organizzata in quattro gruppi di modelli: 16C6X, 16C62X, 16C7X, 16C84 (16F84). Selezionando uno di questi gruppi appare un menù a tendina, con i diversi modelli disponibili. Dopo aver selezionato quello che si desidera, nella parte inferiore della finestra si possono leggere le caratteristiche più importanti del modello scelto.

PAROLA DI CONFIGURAZIONE

Tramite questa finestra si sceglie la configurazione di lavoro del PIC: protezione del codice, attivazione del Watchdog, Brown-Out, Power Timer e tipo di clock (XT).

VERIFICA DI CANCELLAZIONE

Quando si attiva questa opzione, tutta la memoria del circuito integrato, prima della fase di scrittura, è sottoposta a una verifica di cancellazione.

ID/CHECK

Mediante questa finestra, si può scrivere un numero di identificazione che verrà scritto nella memoria del PIC (parola di identificazione). In egual modo verrà visualizzato il "Check-sum" del contenuto attuale della memoria.

APRI FILE

Questo pulsante permette di leggere il contenuto di un file eseguibile in formato INTEL.HEX e depositarlo nel buffer della memoria.

PROGRAMMA TUTTO

Trasferisce il contenuto del buffer di memoria e la Parola di Configurazione attuale nella memoria interna del microcontroller.

PROGRAMMA PAROLA

Scrive unicamente la parola di configurazione attuale sul microcontroller, lasciando inalterato il contenuto della sua memoria interna.

VERIFICA

Verifica il contenuto della memoria interna del PIC con il contenuto del buffer di memoria. Se viene rilevata qualche differenza appare il corrispondente messaggio. Si utilizza per verificare la corretta programmazione del PIC.

LEGGI PIC

Legge il contenuto della memoria interna del PIC e lo deposita nel buffer della memoria.

CANCELLA PIC

Cancella il contenuto della memoria interna del PIC. Questa icona risulta attiva solo quando la memoria è di tipo FLASH e può essere cancellata.

VERIFICA DI CANCELLAZIONE

Verifica che la memoria interna del PIC sia totalmente cancellata.

ESCI

Chiude il programma PICME-PR e restituisce il controllo del PC al sistema operativo.

Il MICRO PIC TRAINER: un sistema di sviluppo

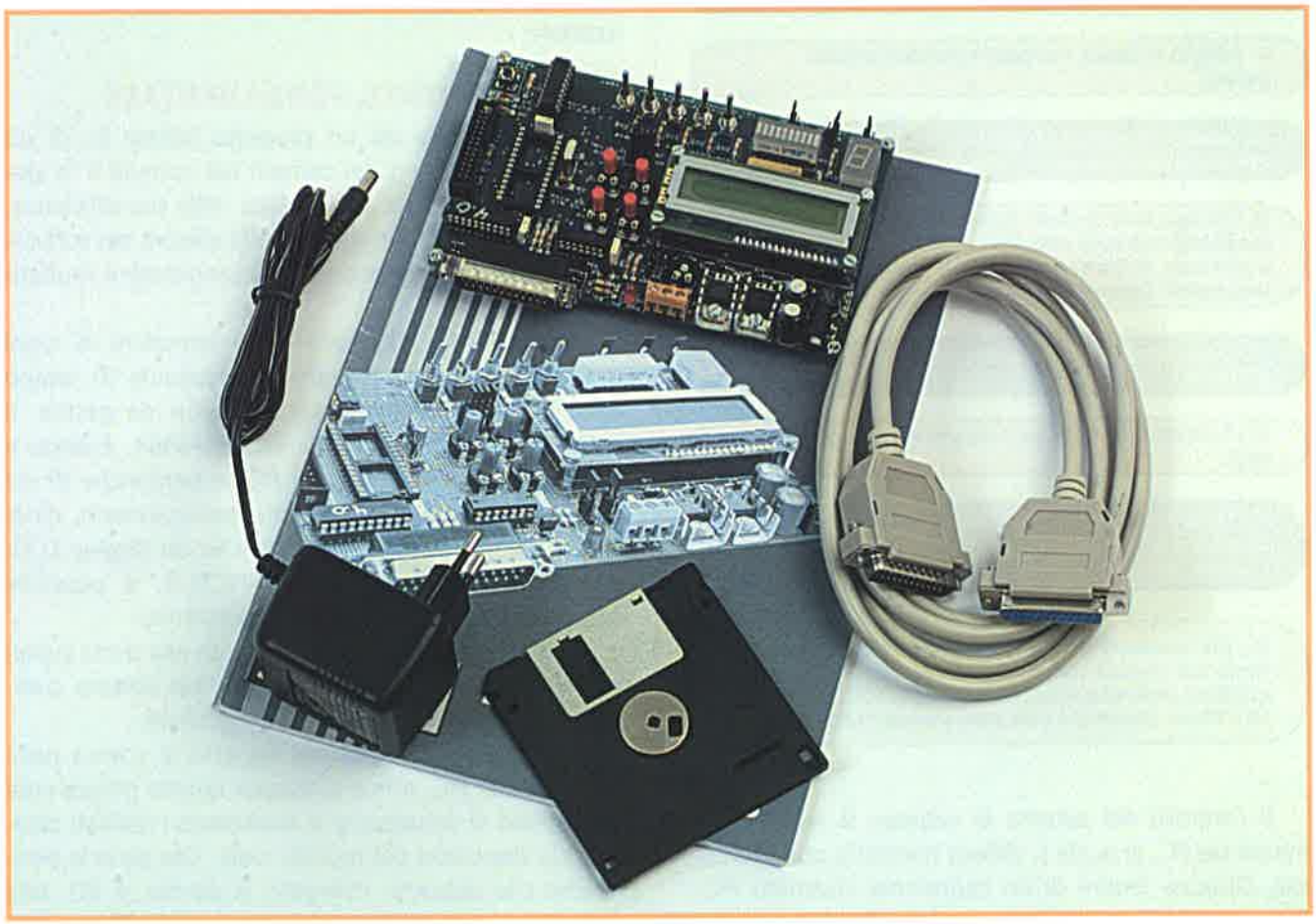
Un sistema di sviluppo è un insieme di strumenti hardware e software, che permettono di realizzare tutte le operazioni necessarie per porre in essere un progetto basato su di un microcontroller.

Mediante un sistema di sviluppo si può editare il programma, memorizzarlo, cancellarlo se la memoria lo permette, compilarlo, assemblarlo, eseguirlo in tempo reale e farne il "debug", ovvero la messa a punto.

Per poter eseguire queste operazioni è necessario uno scrittore/cancellatore, un insieme di periferiche tipiche, da collegare ai piedini di I/O del microcontroller, l'alimentazione e un pacchetto software contenente l'editor per scrivere il programma, il compilatore, l'assemblatore, il simulatore e il programma di comunicazione e controllo.

Per questa descrizione ci siamo basati su un sistema di sviluppo economicamente accessibile, efficace e semplice da utilizzare, denominato MICRO PIC TRAINER della ditta "MSE: Microsystem Engineering".

È destinato ai PIC della gamma media, con un massimo di 28 piedini, fra cui troviamo anche il PIC16F84. Nella figura si può vedere l'insieme dei materiali che compongono il MICRO PIC TRAINER.



Insieme dei materiali che formano il sistema di sviluppo MICRO PIC TRAINER.

MICROCONTROLLER

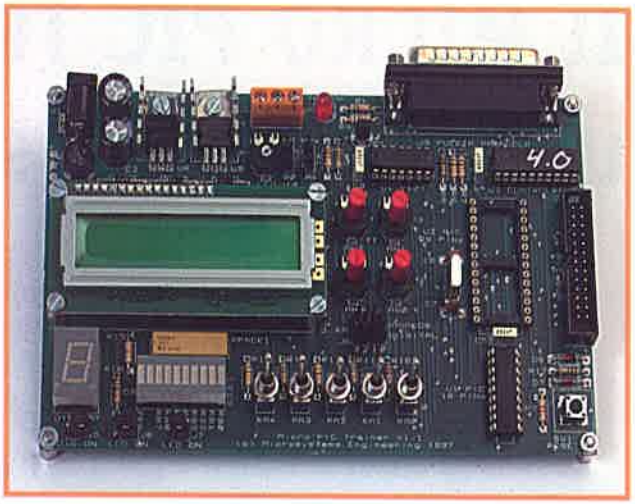
Concetti e utilità del sistema di sviluppo

CARATTERISTICHE GENERALI

Nella tabella seguente sono riportate le caratteristiche e i componenti principali di cui è composto il MICRO PIC TRAINER.

1. - Alimentazione tramite un trasformatore da 220/15 VAC, 0,5 – 1 A oppure due pile da 9 VDC.
2. - Frequenza di lavoro standard da 4 MHz, che può essere cambiata sostituendo il cristallo di quarzo.
3. - Generazione di Reset automatica (POR) e manuale tramite un pulsante.
4. - Cinque ingressi digitali implementati con cinque commutatori.
5. - Quattro ingressi analogici implementati con quattro potenziometri.
6. - Otto uscite digitali collegate a otto diodi LED o ad un display a 7 segmenti.
7. - Modulo LCD da 2 linee di 16 caratteri.
8. - Possibilità di collegare e scollegare le periferiche mediante jumpers.
9. - Connettore di espansione PICBUS da cui passano tutti i segnali di I/O per i circuiti ausiliari esterni.
10. - Circuito scrittore "in circuit". Permette di scrivere un PIC montato su uno qualsiasi degli zoccoli a disposizione. Il PIC16F84 si può scrivere, cancellare e riscrivere tutte le volte che si desidera, senza toglierlo dallo zoccolo.
11. - Diodo LED indicatore della presenza della tensione di scrittura, di 13,8 VDC.
12. - Connettore DB25 a 25 fili per il collegamento con la porta parallela del PC.
13. - Dischetto contenente il software in Italiano per realizzare tutte le operazioni: lettura, scrittura, verifica, cancellazione, editing dei test, ecc.
14. - Tutti i componenti che compongono il MICRO PIC TRAINER sono montati su di un circuito stampato professionale, a doppia faccia, con fori metallizzati, ampiamente serigrafato, di dimensioni EUROCARD da 100 x 160 mm. L'insieme può anche essere acquistato in kit, da montare.

Il controllo del sistema di sviluppo si realizza per mezzo del PC, al quale si collega tramite la porta parallela. Dispone inoltre di un connettore chiamato PICBUS, tramite il quale sono portati all'esterno tutti i segnali di I/O e l'alimentazione del microcontroller. Con questo connettore si può espandere verso l'esterno il MICRO PIC TRAINER, permettendo la connessione di numerose schede di espansione, sviluppate dal fabbricante stesso, oppure con le schede sviluppate dall'utilizzatore.



Scheda MICRO PIC TRAINER.

UNO STRUMENTO FONDAMENTALE

Quando si lavora ad un progetto basato su di un microcontroller, uno dei compiti più comuni è la stesura del programma, e la verifica della sua efficienza. Il progetto si divide in moduli, e la stesura del software che controlla ognuno di loro, propizierà il risultato finale.

Per verificare l'idoneità delle istruzioni di ogni modulo, dobbiamo provarle ed eseguirle in tempo reale sulle periferiche o sui dispositivi da gestire. Il sistema di sviluppo MICRO PIC TRAINER, collega e scollega ai piedini di I/O del PIC le periferiche di cui dispone: commutatori, pulsanti, potenziometri, diodi LED, display a 7 segmenti, e anche un display LCD. Inoltre tramite il connettore PICBUS, è possibile aggiungere qualsiasi componente esterno.

Dato che il dispositivo è collegato alla porta parallela del PC, quest'ultimo è utilizzato per editare, compilare, assemblare e simulare i programmi.

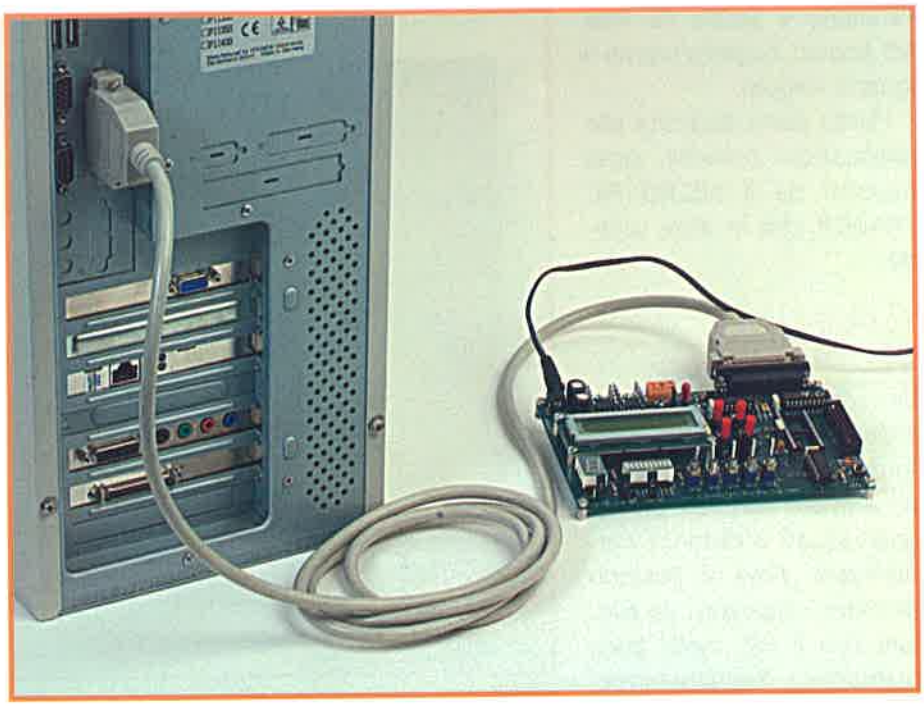
Dopodiché il programma oggetto si scarica nella memoria del PIC. Infine si esegue questo programma sul sistema di sviluppo, e si analizzano i risultati ottenuti sui dispositivi del mondo reale, che sono le periferiche che abbiamo collegato ai piedini di I/O. Alla vista di questi risultati si può valutare se il software che abbiamo scritto funziona o meno, passando alla suc-

cessiva fase di messa a punto e alla ripetizione del processo.

COMPLEMENTI E RISORSE AUSILIARIE

Anche se sulla scheda del MICRO PIC TRAINER ci sono solo due zoccoli da 18 e 28 piedini, esistono diversi adattatori che permettono di lavorare con i differenti modelli di PIC e di piedinature. Come riferimento ricordiamo che l'adattatore "ZOC0603" adatta i PIC a 28 piedini con dimensione 0,6" alla dimensione 0,3"; invece lo "ZOC87x" è un kit di adattamento per l'utilizzo dei PIC16F87x.

Raccomandiamo l'uso degli zoccoli "a forza di inserzione nulla", che dispongono di una leva che apre i fori e facilita l'inserzione e l'estrazione dei PIC. Grazie al connettore di espansione PICBUS, è possibile collegare al MICRO PIC TRAINER una serie di schede, alcune delle quali sono descritte di seguito, e che offrono molteplici possibilità di sviluppo.



Collegamento del sistema di sviluppo alla porta parallela del PC.

MICRO PIC TRAINER PLUS

Basata sul collegamento con il bus I2C permette l'impiego di nuove periferiche, come le tastiere matriciali, un visualizzatore da 4 display a 7 segmenti, convertitori AD, clock in tempo reale, espansione della memoria RAM non volatile, nuove porte di I/O, ecc.

MICRO PIC IO

Mette a disposizione dell'utilizzatore periferiche industriali come motori DC, microrelé, cicalini piezoelettrici, sensori ottici, controlli di illuminazione, ecc.

MSx84

Scheda controllata da un PIC16F84 e adattabile tramite il connettore PICBUS; orientata al controllo di motori DC e PAP e a diversi altri sensori. È la scheda principale del microrobot PICBOT-2.

BIBLIOGRAFIA RIFERITA AL MICRO PIC TRAINER

Oltre al manuale dell'utilizzatore, realizzato in italiano, per i nostri lettori, che accompagna il sistema di sviluppo e che contiene sezioni sul montaggio, messa in marcia, installazione del software, funzionamento, modulo LCD e tutorial per realizzare i primi progetti, si possono trovare su Internet gli aggiornamenti del software e le ultime novità consultando il sito: www.microcontroller.com.

Per fornire a professori e ad autodidatti un metodo pratico e chiaro per gestire il MICRO PIC TRAINER, la ditta costruttrice ha sviluppato un "corso di progetto pratico con il PIC", in cui la prima parte è dedicata al PIC16F84 con esercizi semplici, e la seconda parte propone una collezione di applicazioni più avanzate realizzate con il PIC16F873.

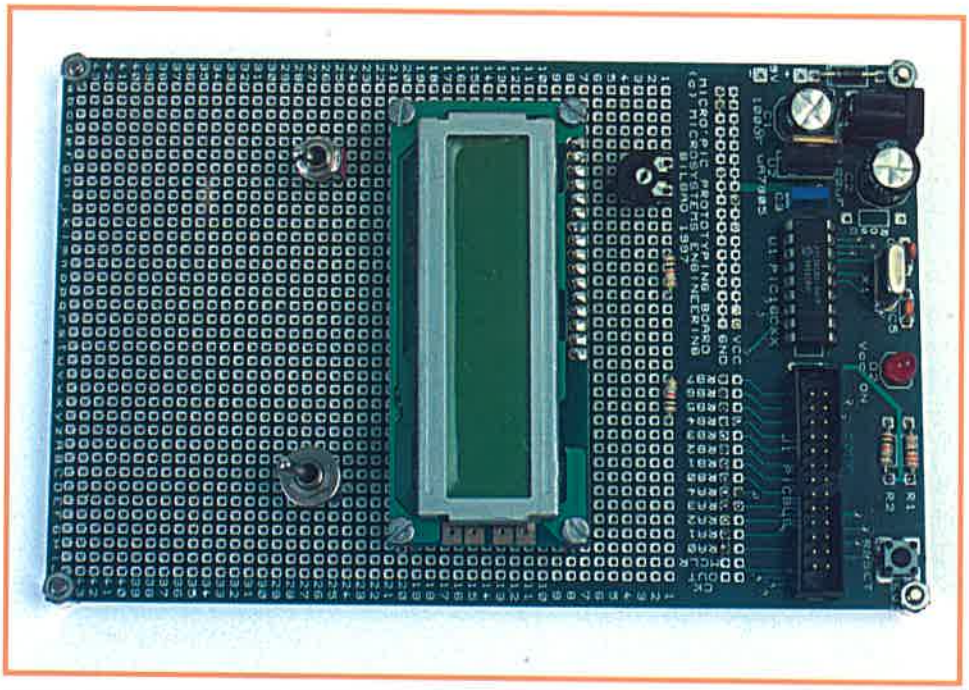
Anche l'opera edita da Mc Graw-Hill "Microcontroladores PIC: Diseño Práctico de Aplicaciones" e scritta da José Mª Angulo, Susana Romero e Ignacio Angulo, utilizza il MICRO PIC TRAINER come strumento base per l'esecuzione di tutti gli esercizi e gli esempi che contiene. La prima parte di quest'opera guida il lettore al primo approccio con il PIC16F84.

Il secondo libro è più avanzato, ed è dedicato alla presentazione teorica e pratica dei moderni microcontroller PIC16F87x e all'insieme di schede che ruotano attorno al MICRO PIC TRAINER.

Inoltre è anche consigliabile il libro "Microcontroladores PIC. La solución en un chip" edito da ITP

Paraninfo e scritto da José Mª Angulo, Eugenio Martín e Ignacio Angulo.

Nella parte dedicata alle applicazioni pratiche, sono descritti sia il MICRO PIC TRAINER che le altre schede.

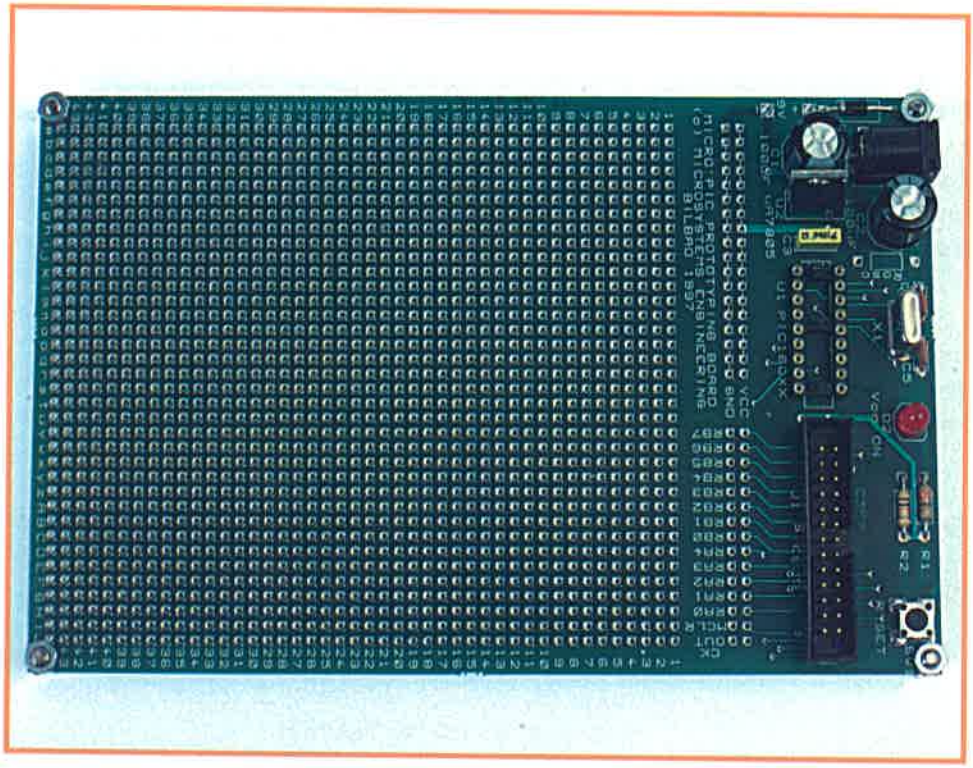


Scheda dei prototipi con alcuni componenti di un'applicazione.

SCHEDA DEI PROTOTIPI

La scheda dei prototipi, come si può osservare nella fotografia, dispone di una zona libera con fori prestagnati situati a distanza normalizzata, dove si possono montare i dispositivi da pilotare con il PIC. Nella zona premontata, è situato lo zoccolo che conterrà il PIC, e la sua circuiteria fissa, formata dal cristallo di quarzo con i suoi due condensatori, il circuito di Reset e lo stabilizzatore di tensione.

Esiste una scheda dei prototipi per i PIC da 18 piedini che si chiama MPIC-18M, e un'altra destinata ai PIC da 28 e 40 piedini, denominata MPIC-28/40M.



Scheda dei prototipi per progetti basati sui PIC della gamma media.

Dopo che il programma è stato messo a punto sul sistema di sviluppo MICRO PIC TRAINER ed è stato scritto nel PIC, quest'ultimo viene montato sulla scheda dei prototipi, dove, tramite i suoi piedini di I/O, piloterà i dispositivi periferici definitivi. In questo modo si disporrà di un prototipo del progetto con prestazioni professionali, basso prezzo e rapidità di montaggio.

In modo complementare, Ingeniería de Microsistemas Programados S.L. fornisce due kit di componenti e un manuale con esercizi per sviluppare una serie di progetti con il MICRO PIC TRAINER e le schede dei prototipi. Il kit di inizio si chiama "KIT DE PROYECTOS BÁSICOS" e quello avanzato "KIT DE PROYECTOS TÍPICOS".

Il proto PIC 2, scrittore di PIC

Anche se le basi fondamentali dell'elettronica attuale sono le stesse che la accompagnano dalla nascita, 50 anni fa, il suo studio e la sua applicazione hanno subito notevoli cambiamenti.

I semiconduttori e in particolare i transistors, che sono i più caratteristici tra i componenti microelettronici, hanno ridotto moltissimo le loro dimensioni, e ora ne possiamo disporre a milioni sulla piccola superficie di un chip di un circuito integrato. Il loro utilizzo è basato sulla programmazione, che ha semplificato la parte relativa all'hardware introducendo e sviluppando un nuovo componente: il software.

Comunque esistono numerosi prodotti elettronici che non sono programmabili, perché anche se i microcontroller lo sono, devono poi essere collegati a dispositivi semplici per sviluppare le applicazioni commerciali.

Per facilitare lo studio della moderna microelettronica in modo progressivo, pratico e modulare, descriveremo uno strumento didattico professionale che, oltre a supportare un programma proprio di autoinsegnamento sperimentale nelle principali aree dell'elettronica e dei microcontroller, serve come laboratorio e banco di prova per montare, modificare, mettere a punto e porre in marcia qualunque tipo di progetto commerciale o industriale in modo rapido, economico e sicuro.



Laboratorio di microelettronica e microcontroller UNIVERSAL TRAINER.

IL LABORATORIO DI MICROELETTRONICA E MICROCONTROLLER "UNIVERSAL TRAINER"

Sotto la denominazione UNIVERSAL TRAINER della MSE: Microsystems Engineering, si presenta un laboratorio di sperimentazione contenuto in una valigetta leggera e facilmente trasportabile che contiene tutti i dispositivi e i circuiti principali che richiedono il montaggio ed il funzionamento di qualsiasi pro-

MICROCONTROLLER

L'insegnamento dell'elettronica moderna

getto basato sulla microelettronica o sui microcontroller. Come si può vedere nella fotografia, dispone di una zona con una scheda protoboard per il montaggio dei circuiti senza saldatura e, attorno, vi sono raggruppate la maggior parte delle periferiche e dei circuiti funzionali comuni a queste applicazioni, accessibili anche loro mediante cablaggio a pressione.

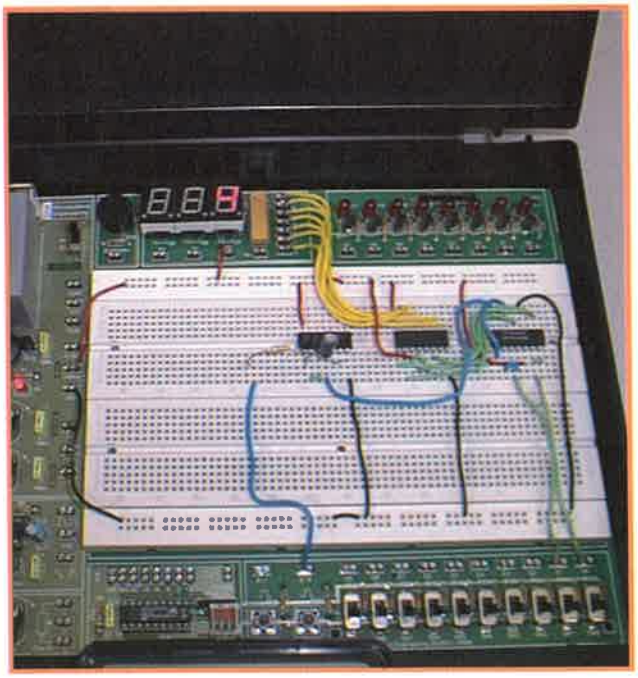
I sistemi e i blocchi più importanti dell'UNIVERSAL TRAINER sono descritti nella tabella seguente.

- Alimentatore da +5, +12 e -12 VDC
- **Alimentatore ad uscita variabile positiva e negativa**
- Trasformatore di alimentazione da 220 /12 +12 VAC
- **Generatore di funzioni**
- Generatore logico
- **Potenziometro regolabile**
- Sezione delle periferiche di ingresso formata da 10 interruttori
- **Sezione da 2 pulsanti**
- Sezione da 8 diodi LED
- **Cicalino piezoelettrico**
- Visualizzatore da 3 display a 7 segmenti
- **Valigetta trasportabile**
- Raccoglitore con schede a colori preforate ed estraibili, per il montaggio, la messa a punto, il funzionamento e l'utilizzo dell'UNIVERSAL TRAINER

Utilizzando la tensione che fornisce l'alimentatore del laboratorio, i segnali dei generatori di funzioni e del generatore logico, le periferiche disponibili e la zona per il montaggio senza saldatura, si possono costruire e provare rapidamente schemi elettronici complessi.

I MODULI FACOLTATIVI DI INSEGNAMENTO TEMATICO

L'UNIVERSAL TRAINER non serve solo da supporto per i montaggi e i progetti dell'utilizzatore, ma costituisce anche un eccellente banco di prova per iniziare a familiarizzare con le aree più rappresentative della micro-



Dettaglio del collegamento dei componenti utilizzati in un esperimento.

elettronica. Per raggiungere questo obiettivo formativo, "Ingegnieria di Microsistemi Programmabili S.L." ha creato sette moduli specializzati nell'insegnamento di altrettante sezioni che coprono l'elettronica moderna e che sono elencate nella tabella.

- MODULO 1: ELETTRONICA DIGITALE
- **MODULO 2: SEMICONDUTTORI**
- MODULO 3: ELETTRONICA ANALOGICA
- **MODULO 4: MICROCONTROLLER I (PIC16F84)**
- MODULO 5: MICROCONTROLLER II (PIC16F873)
- **MODULO 6: AMPLIFICATORI OPERAZIONALI**
- MODULO 7: CIRCUITI LOGICI PROGRAMMABILI

Ognuno di questi sette moduli è composto da:

- Una serie di schede estraibili a colori contenenti gli esercizi e i progetti proposti per l'insegnamento del corrispondente tema trattato nel modulo.
- Scatola di plastica contenente i componenti elettronici necessari per il montaggio degli esercizi dell'opera sull'UNIVERSAL TRAINER.



Serie di schede con gli esercizi e i materiali necessari per il montaggio degli esperimenti proposti sull'UNIVERSAL TRAINER, facenti parte del modulo 1, che tratta di elettronica digitale.

• Dischetti con programma per quei moduli che lo richiedono.

Il programma parte con l'insegnamento, nel Modulo 1, dell'elettronica digitale. Nel Modulo 2 si sperimentano e studiano i semiconduttori, diversi tipi di diodi, la rettificazione e gli alimentatori.

Il Modulo 3, focalizzato sull'elettronica analogica, fornisce numerosi esercizi e montaggi con i transistor. Nel Modulo 4 si inizia lo studio dei microcontroller, concentrandosi sulle applicazioni con il PIC16F84.

Il Modulo 5 approfondisce la gestione dei microcontroller, con esperimenti più avanzati, basati sul PIC16F873. Il Modulo 6 copre le funzioni e le applicazioni degli amplificatori operazionali e, infine, il Modulo 7 si dedica alla manipolazione e ai progetti con circuiti logici programmabili, come quelli del tipo PAL/GAL.

MODULI PER L'INSEGNAMENTO DEI MICROCONTROLLER

I Moduli 4 e 5 che completano l'UNIVERSAL TRAINER insegnano il progetto e la programmazione dei microcontroller PIC. Nel Modulo 4 si collega il PIC16F84 ai circuiti funzionali e alle periferiche presenti sul laboratorio, e si sviluppano 17 progetti di applicazioni. Nel Modulo 5, basato sul PIC16F873, sono contenuti dei progetti di difficoltà maggiore, compresa la gestione delle periferiche più comuni, come le tastiere matriciali e i display LCD. Come tutti gli altri moduli, anche questi contengono una serie di schede a colori dove sono presentati i progetti, una scatola con i materiali necessari per montarli sull'UNIVERSAL TRAINER e un dischetto contenente i programmi sorgente per tutti gli esercizi.

Sul sito Internet: www.microcontrolladores.com troviamo costantemente nuovi esercizi, modifiche a quelli proposti, aggiornamenti dei software e programmi, e



Montaggio di un progetto basato su di un PIC nell'UNIVERSAL TRAINER, utilizzando il PROTO PIC 2.

novità che possono essere liberamente visionate. All'interno della scatola con il materiale per il montaggio degli esercizi dei Moduli 4 e 5, deve esserci uno scrittore di PIC. Per coprire questa esigenza di scrittura dei programmi nella memoria dei codici del PIC16F84 e del PIC16F873, sono disponibili due opzioni.

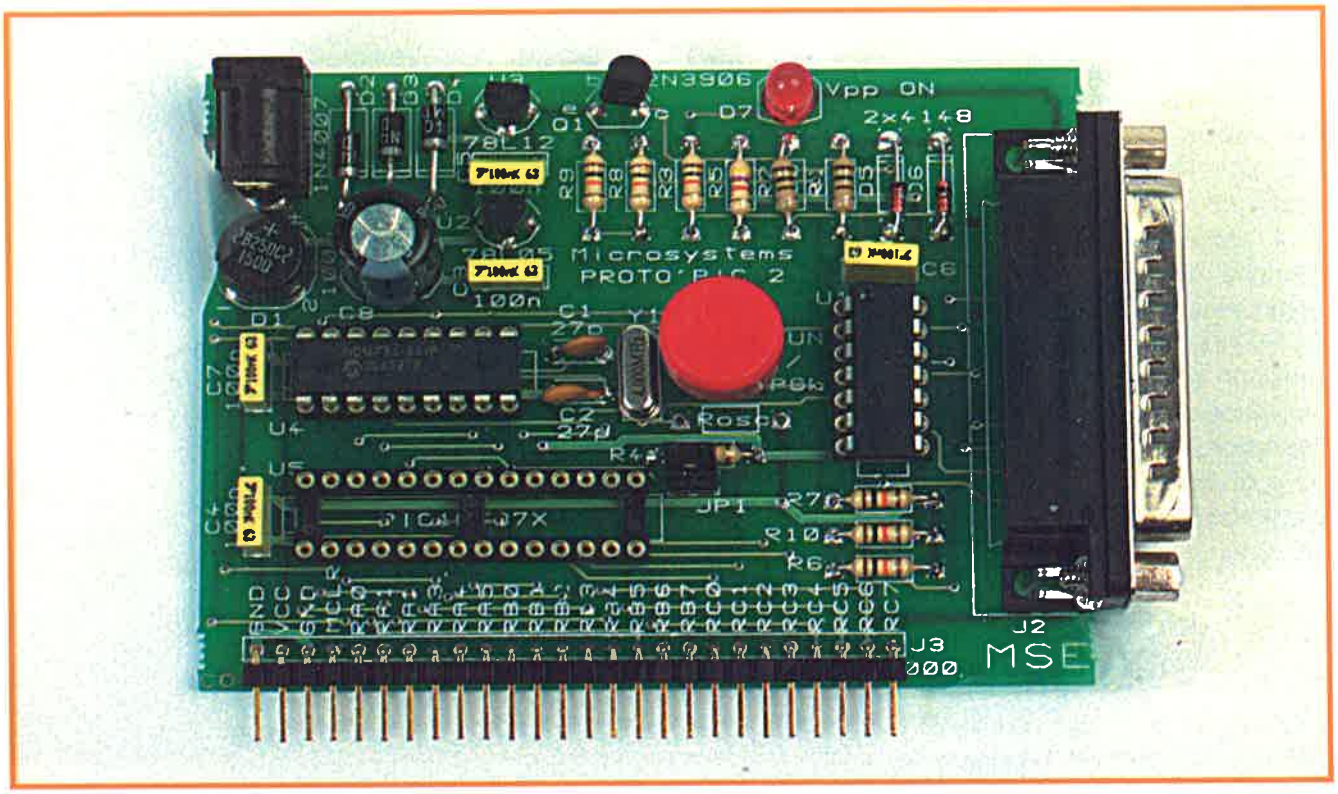
Per gli utenti che dispongono dello scrittore del PIC esistente sul MICRO'PIC PROGRAMMER o il MICRO' PIC TRAINER, descritto in precedenza, si offre una scheda di adattamento a questi tramite il connettore PICBUS.

Questa scheda, che si adatta ai menzionati scrittori esterni e si inserisce nella scheda protoboard del laboratorio, viene chiamata PROTO'PIC. Per gli utilizzatori che non dispongono di scrittore di PIC viene fornita un'altra versione che comprende la scheda PROTO' PIC2, la quale contiene su di essa uno scrittore di PIC.

Questa scheda si collega alla porta parallela del PC, il quale controlla il suo funzionamento e il processo di scrittura e, dall'altra parte, viene inserita sui fori della scheda protoboard, dal quale collega i dispositivi presenti sull'UNIVERSAL TRAINER a quelli esterni che si possono aggiungere.

DESCRIZIONE DELLA SCHEDA PROTO PIC 2

Tramite un connettore DB25 la scheda PROTO PIC 2 si collega alla porta parallela del PC, che ne gestisce il funzionamento. Con questa scheda si possono scrivere iprogrammi che sono stati editati, compilati e assemblati sul computer. Per questa operazione esiste un programma di comunicazione tra il PC e il PROTO'PIC 2 con il quale si scarica il programma oggetto residente nel PC nella memoria di programma del PIC che si trova sulla scheda. I piedini del PROTO'PIC 2 si inseriscono nei fori della scheda protoboard e tramite questi, mediante fili inseriti a pressione, si collegano con le periferiche dell'UNIVERSAL TRAINER o con quelle montate sulla propria scheda. Senza bisogno di modificare l'hardware, né muovere il PIC dallo zoccolo del PROTO'PIC 2, si può riscrivere un nuovo programma e verificarne il funzionamento immediatamente e sui dispositivi reali che possiamo gestire. La scheda PROTO'PIC2 serve come scrittore di PIC della gamma media, da 18 a 28 piedini, e come elemento di interfaccia rapido per collegare componenti montati su una scheda protoboard, e collegati mediante fili inseriti a pressione nei fori della protoboard stessa.



Scheda PROTO PIC 2, che contiene il proprio scrittore di PIC per realizzare i progetti dei Moduli 4 e 5.

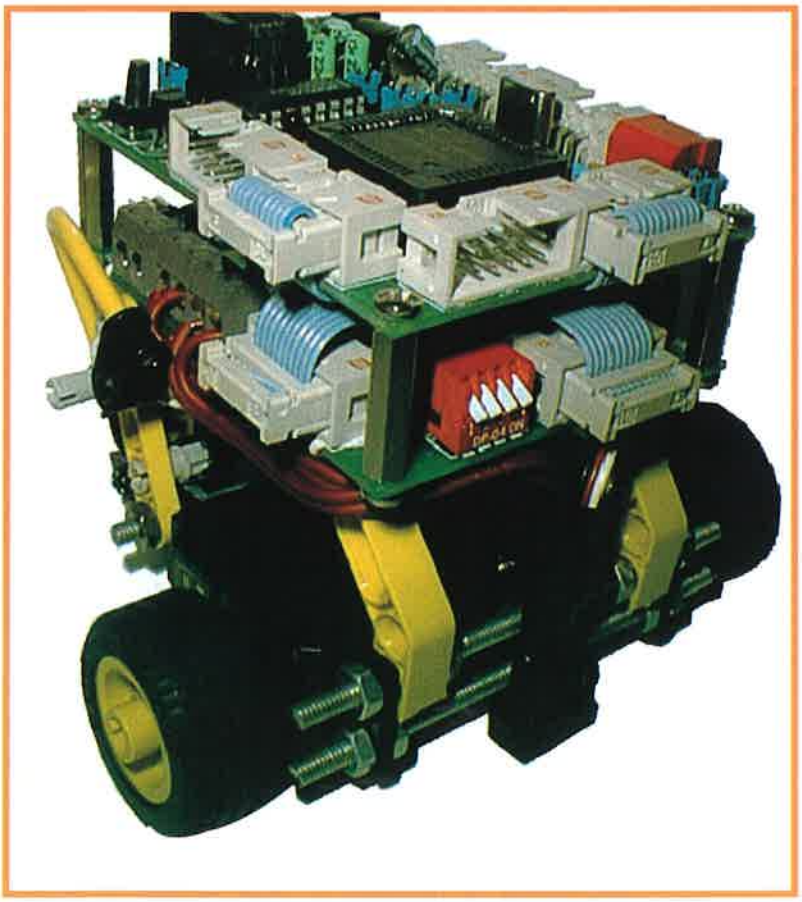
Altri microcontroller per microrobot: il 68HC11 di Motorola

Il microcontroller MC68HC11 è stato uno dei preferiti dai progettisti di microrobot. Le sue prestazioni e le sue eccellenti caratteristiche tecniche lo rendono molto adatto per questa tecnologia. Un esempio è il microrobot TRITT, mostrato nella fotografia in basso, progettato e commercializzato a partire dal 1997 dalla ditta MICRO-ROBOTICA S.L.

La struttura del TRITT è modulare, la sua parte elettronica è configurata sulla base di varie schede sovrapponibili, e la scheda che occupa la posizione superiore è quella che contiene il 68HC11 e la circuiteria ausiliaria. La scheda inferiore della fotografia, contiene il circuito incaricato della gestione dei motori e dei restanti sensori e attuatori.

PRESENTAZIONE GENERALE DEL 68HC11

La struttura interna di questo microcontroller è di 16 bit, però il bus dei dati esterno, utilizzato per collegarlo al resto dei componenti del sistema, è di sole 8 linee, per potersi adattare ai dispositivi che utilizzano questa dimensione del byte, così comune ed economica. L'architettura è di tipo Princeton, nella quale la memoria dei dati e quella del programma sono unite. Nella tabella sottostante sono



Microrobot TRITT costruito attorno al microcontroller 68HC11 di MOTOROLA.

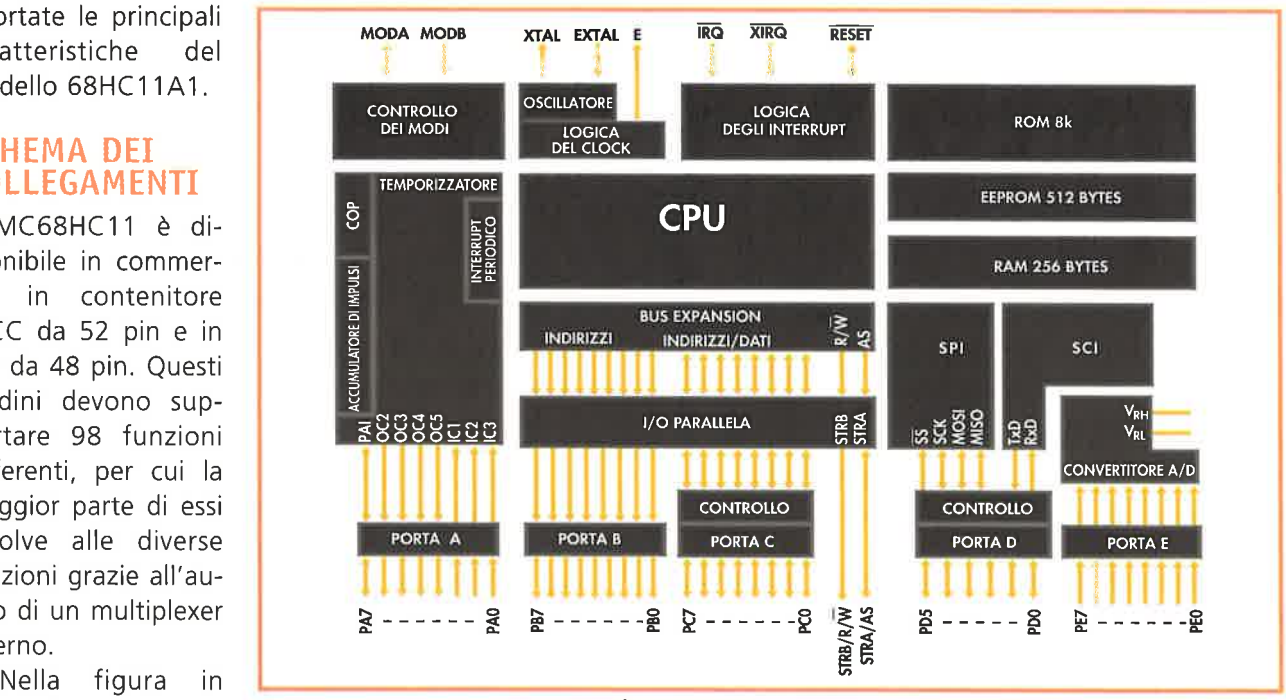
CARATTERISTICHE DEL 68HC11A1
Memoria di programma: 512 x 8 bit EEPROM
Memoria dei dati: 256 x 8 RAM
Alimentazione: da 3 a 6 V, nominale 5 V
5 porte di I/O
Temporizzatore da 8 e da 16 bit
Porta di comunicazione seriale asincrona UART
Porta di comunicazione sincrona SPI
Convertitore AD da 8 bit con 8 canali e 3 Imput Capture
Interrupt in tempo reale
4 modi di funzionamento
Software in ROM per caricare nella RAM un programma esterno.

riportate le principali caratteristiche del modello 68HC11A1.

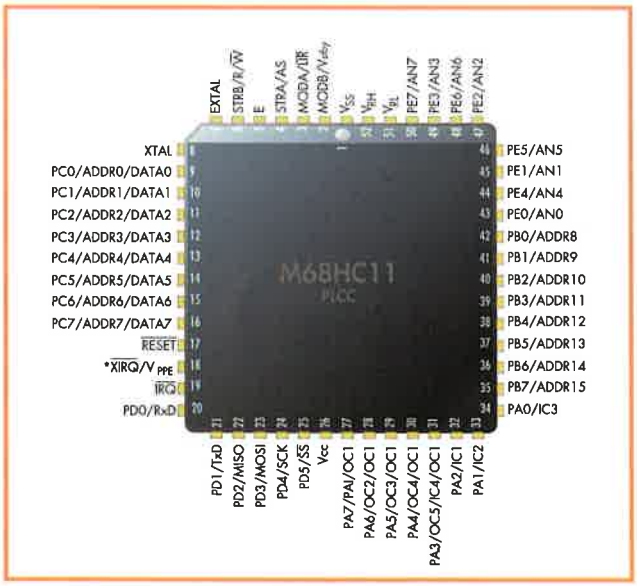
SCHEMA DEI COLLEGAMENTI

Il MC68HC11 è disponibile in commercio in contenitore PLCC da 52 pin e in DIP da 48 pin. Questi piedini devono supportare 98 funzioni differenti, per cui la maggior parte di essi assolve alle diverse funzioni grazie all'ausilio di un multiplexer interno.

Nella figura in basso è riportata la piedinatura per il contenitore PLCC montato sul TRITT. Il fatto che ogni piedino svolga diverse funzioni, obbliga a fare molta attenzione al progetto dell'hardware, perché qualunque errore potrebbe causare la rottura del chip. Molti dei segnali di uscita sono a "collettore aperto", e in questi casi si raccomanda di mettere una resistenza di "pull-up" da 4K7.



Struttura interna del 68HC11 con i blocchi che lo configurano e i relativi piedini.



Piedinatura del 68HC11 nella versione con il contenitore PLCC da 52 pin.

I pin del microcontroller sono raggruppati in insiemi che realizzano le stesse funzioni, come si può vedere dalla tabella.

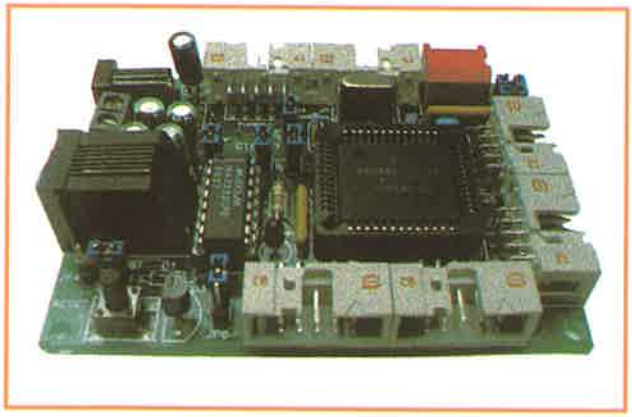
GRUPPI FUNZIONALI DI PIEDINI

1	Alimentazione: VDD e VSS
2	Clock: EXTAL, XTAL e E
3	Reset: RESET
4	Comunicazione seriale asincrona (SCI): TxD e RxD
5	Interrupt hardware: IRQ, XIRQ, ICQ-3, PAI e STRA
6	Selezione del modo di lavoro: MODA e MODB
7	Comparatori: OCI-5
8	Imput Capture: IC1-3, PAI
9	Comunicazione seriale sincrona (SPI): SCI, MISO, MOSI e SS
10	Porte di I/O: PA0-7, PB0-7, PC0-7, PD0-3 e PE0-3
11	Canali del convertitore AD: AN0-7
12	Piedini dei bus esterni: AD0-7, A8-A15, AS e R/W

Nello schema della figura in alto è riportata l'architettura interna del 68HC11, con i diversi blocchi che lo compongono e i relativi pin.

Nella fotografia della pagina seguente, vediamo la scheda di controllo del TRITT, nella quale possiamo notare il microcontroller nella versione con contenitore PLCC, circondato dalla circuiteria ausiliaria e dai connettori di adattamento.

L'architettura del 68HC11 mostrata nello schema precedente, è utilizzata in molti modelli di questo microcontroller, i quali si differenziano fra di loro prin-



Scheda di controllo del TRITT in cui si può vedere il 68HC11.

DISPOSITIVO	RAM	ROM	EPROM	EEPROM
MC68HC11A0	256	0	0	0
MC68HC11A1	256	0	0	512
MC68HC11A7	256	8k	0	0
MC68HC11A8	256	8k	0	512
MC68HC11D0	192	0	0	0
MC68HC11D3	192	4k	0	0
MC68HC711D3	192	0	4k	0
MC68HC11ED0	512	0	0	0
MC68HC11E0	512	0	0	0
MC68HC11E1	512	0	0	512
MC68HC11E8	512	12k	0	0
MC68HC11E9	512	12k	0	512
MC68HC711E9	512	0	12k	512
MC68HC811E2	256	0	0	2k
MC68HC11E20	768	20k	0	512
MC68HC711E20	768	0	20k	512
MC68HC11F1	1k	0	0	512

Tabella con i diversi modelli del 68HC11, che si differenziano per le diverse capacità della loro memoria.

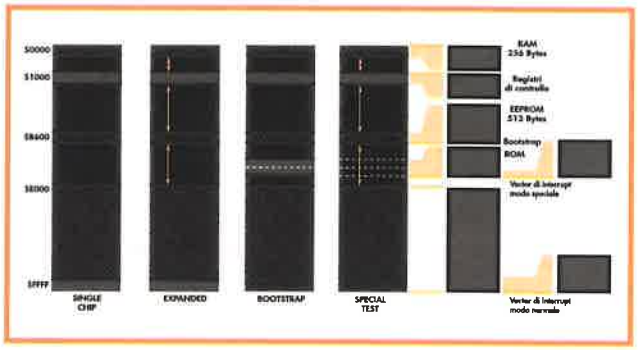
cipalmente dalla capacità della loro memoria, come si può vedere dalle relazioni della tabella in figura.

MODI DI FUNZIONAMENTO

A seconda dei livelli logici che sono applicati ai piedini MODA e MODB, il 68HC11 può lavorare in 4 modi differenti:

1°. Circuito unico o "single chip" (MODA = 0 e MODB = 1).

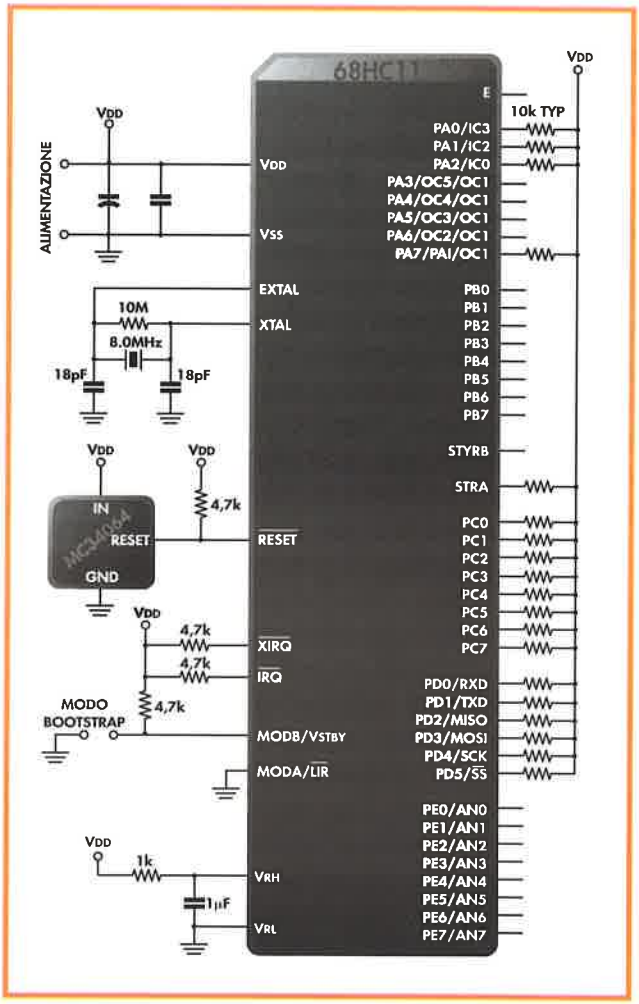
Come si può vedere dalla figura, la mappa della memoria è formata dalla RAM (256), dalla EEPROM



Configurazione della mappa di memoria per i differenti modi di funzionamento del 68 HC11.

(512), dai registri di controllo e da una ROM dove risiede un programma che si esegue alla partenza.

2°. Circuito espanso, o "expanded" (MODA = MODB = 1).



Schema semplificato dei collegamenti del 68HC11 per il funzionamento in modo circuito unico.

In questo modo oltre ad avere accesso alla mappa del circuito unico si può anche accedere agli indirizzi di un'eventuale memoria esterna. Le porte B e C supportano i bus di espansione. Si può disabilitare la memoria interna e lavorare solo con la memoria esterna, che dovrà contenere i vector di interrupt.

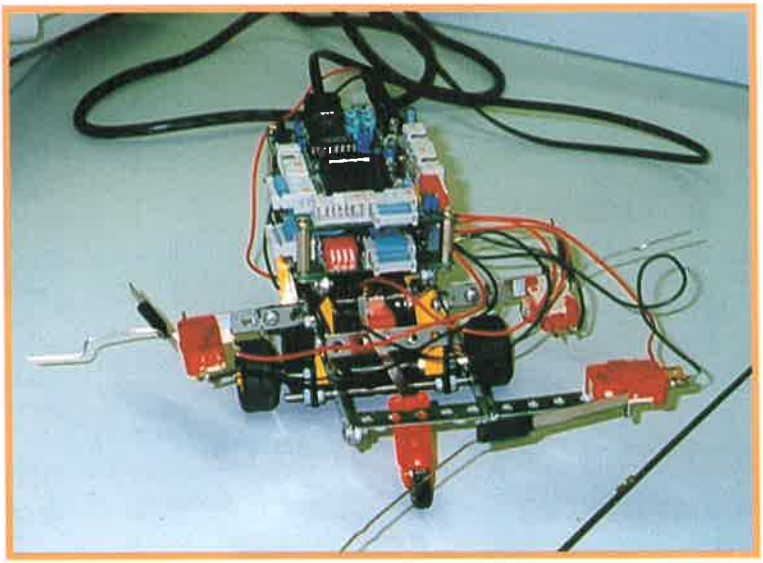
3°. Circuito speciale di Bootstrap (MODA = MODB = 0).

A differenza del circuito unico i vector di interrupt risiedono in una ROM speciale, chiamata "ROM di avvio". Alla partenza viene eseguito questo programma che rimane in attesa di ricevere informazioni tramite la porta seriale; questa procedura serve per scrivere un programma nella RAM dall'indirizzo 00 a FF.

4°. Circuito speciale di Test (MODA =1 e MODB = 0).

Funziona come nel modo bootstrap, in più si può accedere alla memoria esterna e anche ad alcuni registri protetti. È utilizzato nelle prove di fabbrica.

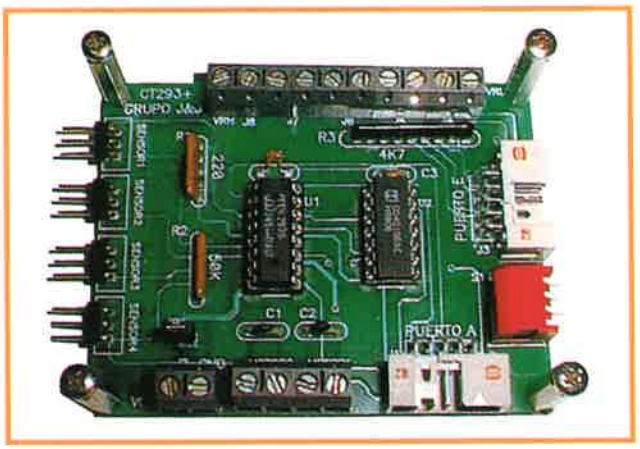
Nello schema della pagina precedente è riportato il collegamento semplificato dei piedini principali del 68HC11 quando funziona in modo circuito unico.



Questa versione espansa del microrobot TRITT può pilotare molti sensori e attuatori, grazie alle numerose linee di I/O del 68HC11.

LE PORTE DI I/O

Sia il funzionamento che la gestione e la programmazione del 68HC11 sono molto diversi del microcontroller PIC. La porta A del 68HC11 ha 8 linee, delle quali 3 funzionano come ingressi e 4 come uscite, la restante è configurabile sia come ingresso sia come uscita. Inoltre queste linee sono condivise da altre risorse, come i comparatori, gli Input Capture e gli accumulatori di impulsi.



Fotografia della scheda di controllo del motore e sensori del microrobot TRITT disegnati per MICROROBOTICA S.L.

La porta A occupa l'indirizzo esadecimale 1000 della zona dei registri; se si scrive in questo indirizzo il valore si rifletterà sulle corrispondenti linee di uscita. Leggendo questa posizione riceviamo l'informazione dai tre pin che funzionano come ingressi. Di conseguenza, durante questa funzione, i dispositivi interni associati a questi piedini sono scollegati. Nella fotografia si mostra una variante del microrobot TRITT che supporta moltissimi sensori e attuatori, i quali sono controllati perfettamente dal 68HC11, grazie alle sue numerose linee di I/O.

Un esempio della programmazione delle linee della porta A chiarirà al lettore la semplicità di questo processo.

Programma per accendere un diodo LED montato sulla linea PA6:

```
        ORG     $0000   ;Il simbolo $ rappresenta il valore esadecimale
        LDA     #40     ;Il registro è caricato con il valore 40
        STAA    $1000   ;Si scrive sull'indirizzo $1000 che corrisponde
                        ;alla Porta A, con PA6 = 1
CICLO   BRA     CICLO   ;Generiamo un "salto su se stesso" provocando
                        ;un ciclo infinito.
        END
```

Micro'PIC Trainer Plus

La scheda MICRO'PIC TRAINER PLUS è un modulo opzionale di espansione del sistema di sviluppo MICRO'PIC TRAINER descritto in precedenza. Entrambi i prodotti sono stati progettati dalla ditta Ingeniería de Microsistemas Programados S.L., che li commercializza in tutto il mondo.

Lo scopo di questa scheda è permettere l'espansione di diverse periferiche, molto comuni nelle applicazioni basate sui microcontroller, per poterli controllare tramite il MICRO'PIC TRAINER, con il quale si accoppia grazie al connettore PICBUS, che ha 26 linee e che si trova sulla scheda.

Per facilitare l'utilizzo congiunto di questi due moduli, è possibile acquistare il MICRO'PIC LAB PLUS, un dispositivo che su una scheda di metacrilato supporta il sistema di sviluppo MICRO'PIC TRAINER, la scheda di espansione delle periferiche MICRO'PIC TRAINER PLUS e un trasformatore professionale che alimenta tutto il sistema a 220 VAC, come si può vedere nella fotografia.

Il MICRO'PIC TRAINER PLUS, permette di realizzare e sviluppare progetti basati sui microcontroller PIC della gamma media, aggiungendo alle periferiche esistenti nel sistema di sviluppo MICRO'PIC TRAINER i seguenti dispositivi:

1. Canale di comunicazione seriale RS-232.
2. Tastiera matriciale 4x4.
3. Barra di diodi per visualizzare lo stato logico dei segnali più significativi.
4. Rete di periferiche comunicanti tramite il bus I2C.
5. Visualizzatore da 4 display a 7 segmenti pilotato dal SAA1064 (I2C).
6. Quattro canali di conversione A/D e uno D/A con il circuito I2C PCF8591.
7. Porte di I/O digitali supportate dal PCF8574 (I2C).
8. Clock/Calendario in tempo reale più RAM non volatile grazie al PCF8583 (I2C).

SCHEMA GENERALE E PRINCIPALI CARATTERISTICHE

La scheda MICRO'PIC TRAINER PLUS ha una dimensione normalizzata da 100 x 160 mm, Eurocard. È alimentata con +5 VDC che riceve tramite le linee del connettore PICBUS, le quali arrivano dal sistema di sviluppo su cui si trova il microcontroller.

Nella figura è riportato lo schema della scheda, che viene accompagnata da un Manuale di Utilizzo, il quale



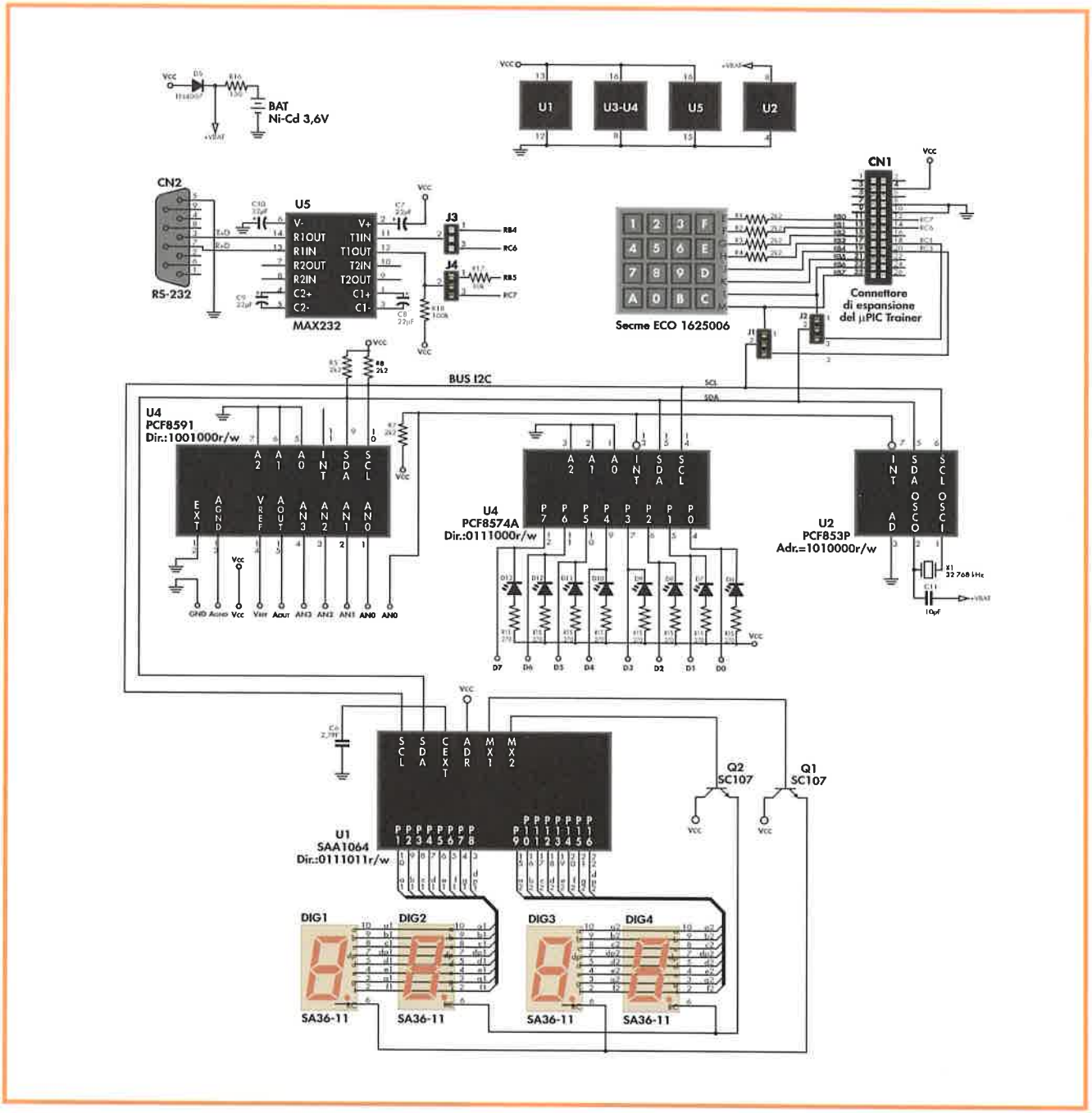
Sistema MICRO'PIC LAB PLUS.



Scheda MICRO'PIC TRAINER PLUS.

MICROCONTROLLER

Micro'PIC Trainer Plus



Schema generale della scheda MICRO'PIC TRAINER PLUS

descrive tutte le sue parti in modo approfondito, numerosi esercizi, programmi e routines tipiche. Tutti i programmi sono contenuti in un dischetto allegato. La maggior parte delle periferiche contenute in questa scheda sono controllate tramite il bus I2C, cui si collegano i seguenti circuiti integrati:

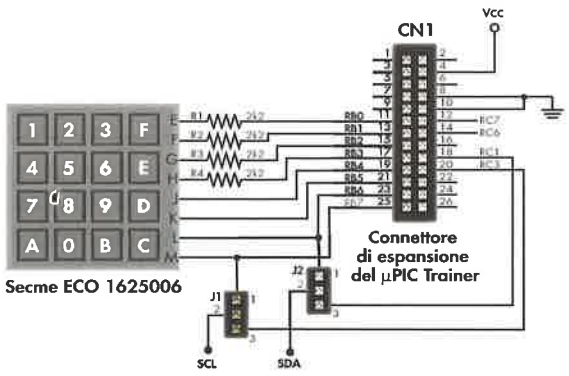
- SAA1064: Controller per 4 display a 7 segmenti.
- PCF8591: Convertitore D/A e A/D.
- PCF8574A: Porta da 8 linee di I/O digitali.
- PCF8583P: Clock/Calendario + RAM non volatile.

LA TASTIERA

Nella figura sono riportati i collegamenti della tastiera matriciale 4 x 4 con i piedini del microcontroller, tramite le linee che arrivano dal connettore PICBUS del sistema di sviluppo MICRO'PIC TRAINER. Osserviamo che la

tastiera è controllata tramite le 8 linee della porta B (RB7 - RB0) del microcontroller. Il software di applicazione configura le linee RB7 – RB4 come uscite, che sono attivate in modo sequenziale, applicando un livello attivo ad ognuna delle file della matrice. Le linee RB3 – RB0 sono configurate come ingressi che ricevono l'informazione delle colonne della matrice, e vengono lette per determinare se qualche pulsante è stato premuto.

Quando il microcontroller non dispone del bus I2C implementato sul chip, come succede nel caso del PIC16F84, il software incaricato di gestirlo utilizza i pin



Schema dei collegamenti della tastiera.

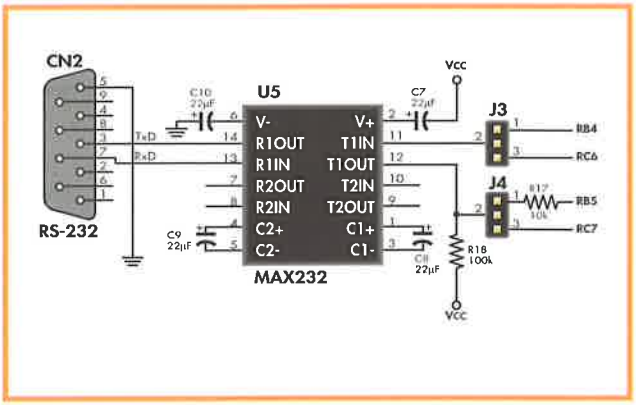
RB6 e RB7 per implementare le linee SDA e SCL. Se il PIC ha il bus I2C integrato all'interno, i due segnali sono supportati dai piedini RC4 e RC3. Per realizzare la selezione del tipo di implementazione del bus I2C via software o via hardware, si debbono utilizzare i jumpers J1 e J2, che commutano i segnali SCL e SDA.

IL CANALE SERIALE RS-232

Nello schema della figura possiamo vedere che è stato basato sul circuito MAX232 (U5), il quale ha il compito di convertire i livelli TTL, presenti sul pin 11 di trasmissione (Tx) a livelli logici RS-232, che si ottengono sul pin 14 (TxD) e si applicano al connettore CN2, per il loro utilizzo esterno.

CLOCK/CALENDARIO E RAM

Il dispositivo PCF8583P adattabile al bus I2C, contiene un clock/calendario in tempo reale, con funzioni orarie tipiche, più allarmi e temporizzazioni. Inoltre include una RAM statica da 256 byte dei quali i primi

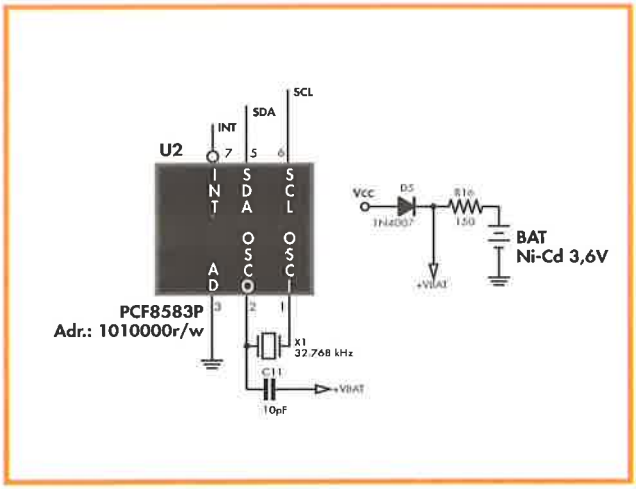


Schema del canale seriale RS-232 basato sul circuito integrato MAX232.

16 sono riservati, lasciando liberi per l'utente 240 byte.

Il PCF8583P risponde all'indirizzo di 7 bit del bus I2C: 1010000, dato che il pin 3 (A0) è collegato a massa. Il pin 1 (OSCI) è l'ingresso degli impulsi che si applicano come frequenza standard al clock, e che può essere di 50 Hz o di 32.768 Hz, generati da un cristallo di quarzo montato sui piedini 1 e 2 (OSCO). Il piedino 7 (INT) si attiva con un livello basso, nel caso si produca un allarme o una temporizzazione.

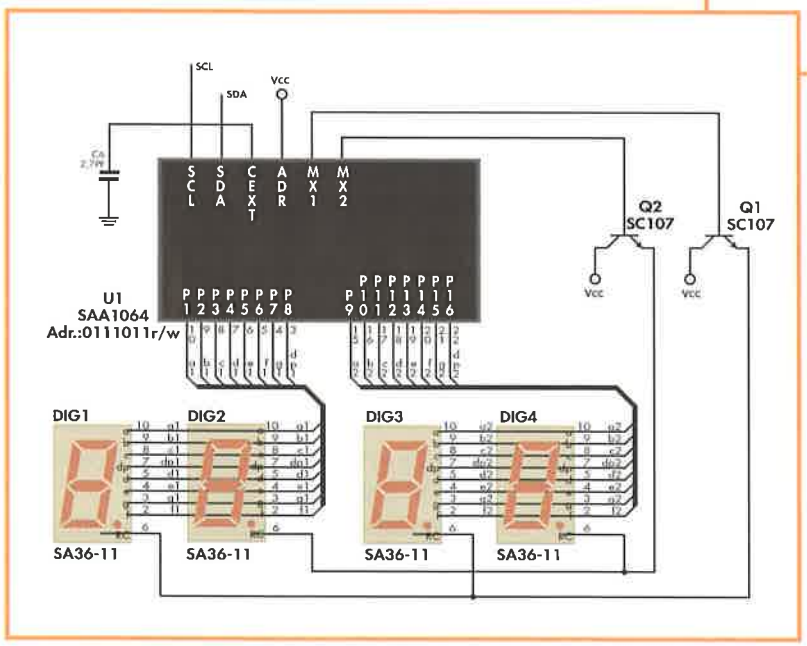
Questo dispositivo si può alimentare con una batteria ricaricabile al Ni/Cd, quando c'è una mancanza nell'alimentazione principale Vcc. In presenza della tensione Vcc, l'alimentazione arriva tramite il diodo D5, e nello stesso tempo si ricarica la batteria tramite lo stesso D5 e R16. Se manca Vcc la batteria mantiene l'ora nel clock/calendario e i dati nella RAM.



Schema dei collegamenti del clock/calendario + RAM contenuto nel PCF8583P.

MODULO DI VISUALIZZAZIONE

Basato sul circuito integrato I2C SAA1064, sulla scheda si controlla un visualizzatore formato da 4 display a 7 segmenti. L'indirizzo a 7 bit che gli corrisponde, ha i 5 bit più significativi programmati in fabbrica, mentre i 2 restanti prendono un valore che dipende dalla tensione applicata al piedino 1 (ADR). Se il pin è collegato a massa, i 2 bit assumono il valore 00; se si collega al positivo il valore sarà 11; se si collega ad una tensione paria a 3/8 Vcc il valore sarà 01 e se si collega ad una tensione di 5/8 Vcc sarà 10.

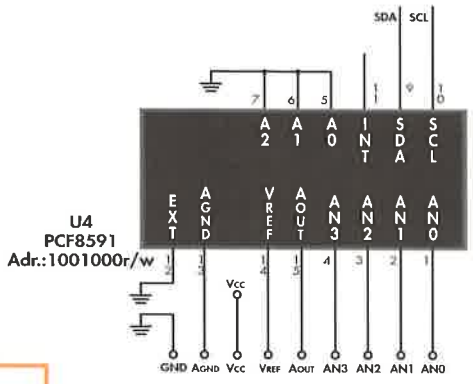


Schema dei collegamenti del modulo visualizzatore tramite il circuito integrato SAA1064.

L'informazione da visualizzare è multiplexata sui digit del visualizzatore. In primo luogo sulle linee P1-P8 appare il dato per il digit 1 e su P9 -P16 per il digit 3. In seguito su queste linee arriva il dato per i digit 2 e 4 rispettivamente. Il processo è automatico e sequenziale, e avviene ad una velocità tale che i 4 display sembrano attivi tutti insieme, mentre in realtà lo sono solamente due per volta.

IL CONVERTITORE A/D E D/A

Questo modulo è basato sul circuito integrato I2C PCF8591, che è già stato descritto in modo approfondito in precedenza, quindi riportiamo solo lo schema dei collegamenti sul MICRO'PIC TRAINER PLUS.

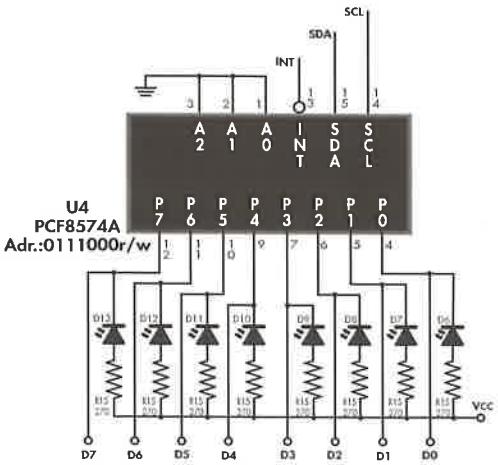


Schema dei collegamenti del PCF8591 che contiene un convertitore D/A e un altro A/D.

PORTE DI I/O

Il dispositivo I2C PCF8574A fornisce al sistema un'ulteriore porta da 8 linee di I/O digitali. Risponde all'indirizzo da 7 bit: 0 - 1- 1 - 1 - A2 - A1 - A0, che in questo caso, essendo collegati a massa i piedini A2, A1 e A0, si converte nell'indirizzo 0 - 1 - 1 - 1 - 0 - 0 - 0.

Le periferiche da controllare tramite le linee di I/O sono pilotate con dei diodi LED che visualizzano il loro stato, anche se sono attive con logica negativa.

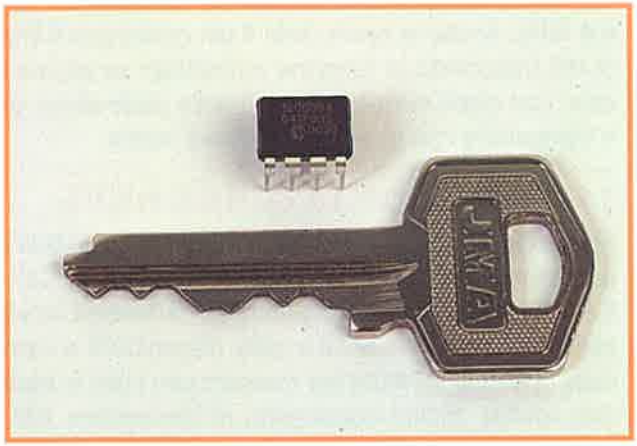


Schema dei collegamenti del dispositivo I2C PCF8574A che fornisce al sistema una porta di 8 linee di I/O digitali.

Il nano favoloso

IL MERAVIGLIOSO PIC NANO

All'interno della grande varietà di modelli di microcontroller PIC esiste una "casta" speciale che riceve il soprannome di "nano". Sono chip portentosi che contengono un computer completo, anche se misurano poco più di mezzo centimetro di lunghezza per ancor meno di larghezza, ed hanno solo 8 pin, quattro per lato. Hanno una dimensione minuscola, dettaglio che si può apprezzare nella fotografia, dove si compara ad una comune chiave. Questa caratteristica darà al lettore un'idea delle possibilità di questa famiglia; grazie alle sue ridotte dimensioni sarà possibile ottenere da questo semplice oggetto un potente controller, il quale, introducendo la chiave nella serratura, riceva l'alimentazione necessaria per eseguire il programma scritto in memoria, permettendo così di pilotare un insieme di periferiche quali relè, motori, luci, ecc.



Un PIC nano confrontato con una comune chiave.

Questo piccolo dispositivo dispone di tutte le risorse e riunisce tutte le caratteristiche tipiche dei PIC che abbiamo studiato sino ad ora. Come è logico, viste le piccole dimensioni, anche il numero delle risorse interne come la capacità di memoria è minimo. Le sue prestazioni, unite ad un giusto prezzo, rendono ideali i PIC nani per governare prodotti semplici. Ecco il motivo per cui è stato scelto il nano PIC12C508A per controllare la pinza che possiede Monty in uno dei suoi bracci, e con la quale potrà prendere, trasportare e depositare oggetti.



Monty dispone di una pinza su uno dei due bracci, che è controllata da un PIC nano.

CLASSIFICAZIONE DEI NANI

Per poter inquadrare i nani nella grande famiglia dei PIC ripassiamo la distribuzione della medesima. La famiglia

GAMMA	MODELLO	CARATTERISTICHE
BASE	PIC16C5X e PIC 12C5XX	33 istruzioni da 12 bit
MEDIA	PIC16CXX e PIC12CXXX	35 istruzioni da 14 bit
ALTA	PIC17CXX	58 istruzioni da 16 bit
MIGLIORATA	PIC18CXX	77 istruzioni da 16 bit

Tabella con i quattro grandi gruppi di microcontroller PIC da 8 bit.

MICROCONTROLLER

Microcontroller a otto pin

dei microcontroller PIC da 8 bit si divide in quattro grandi gruppi, che si diversificano per potenza e prestazioni, e che sono riportati nella figura a lato.

Analizzando la tabella della figura osserviamo che nella gamma base e nella media ci sono due modelli che rispondono alla nomenclatura PIC12C5XX e PIC12CXXX, rispettivamente. Questi modelli sono pertinenti alla casta dei nani e sono caratterizzati da un contenitore con soli 8 pin. Oltre a queste due serie, il costruttore MICROCHIP segue una politica di potenziamento di questi piccoli microcontroller, sviluppando nuovi modelli dotati di memoria FLASH ed EEPROM dei dati, come i PIC18F010 e PIC18F020.

MODELLO	MEMORIA PROGRAMMATA bytes	EPROM	ROM	EEPROM	RAM bytes	LINEE I/O	FREQUENZA MAX.	TEMPORIZZATORI
PIC12C508A	768	512x12			25	6	4MHz	1 da 8 bit
PIC12C509A	1536	1024x12			41	6	4MHz	1 da 8 bit
PIC12CR509A	1536	-	1024x12		41	6	4MHz	1 da 8 bit
PIC12CE518	768	512x12		16x8	25	6	4MHz	1 da 8 bit
PIC12CE519	1536	1024x12		16x8	41	6	4MHz	1 da 8 bit

Tabella con le principali caratteristiche dei PIC12C5XX nani della gamma base.

I PIC12C5XX DELLA GAMMA BASE

I microcontroller PIC da 8 bit sono caratterizzati dal possedere un repertorio di 33 istruzioni da 12 bit cadauna, con una frequenza massima di funzionamento di 4 MHz. Ci sono solo due livelli nello stack per memorizzare gli indirizzi di ritorno al programma principale, il contatore di programma può essere da 9 bit oppure da 11 bit e il numero delle risorse interne e delle linee di I/O è molto limitato.

Nella figura in basso a destra sono presentate le principali caratteristiche dei due modelli fondamentali (PIC12C508 e PIC12C509) e delle loro varianti PIC12C5XX. La lettera C indica che la memoria di programma è EPROM, le lettere CR che è ROM e le lettere CE significano che si dispone di memoria EEPROM dei dati non volatile.

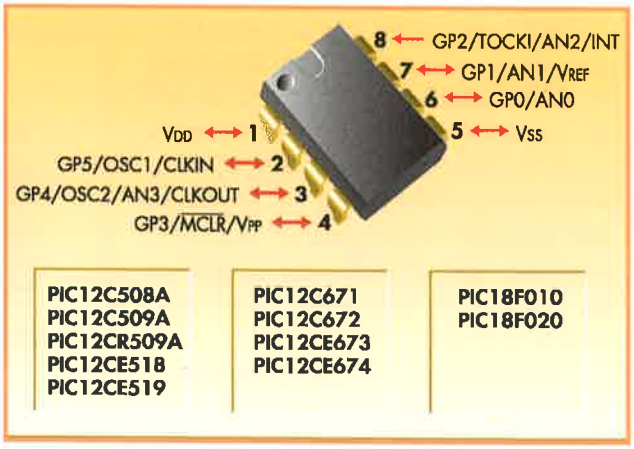
Questi PIC dispongono di un Watchdog o WDT, sono alimentati con una tensione compresa fra 2,5 e 5,5 VDC e consumano 2 mA quando lavorano a 5 VDC e 4 MHz. Anche se hanno solo 8 pin gestiscono 6 linee di I/O utilizzando la funzione multiplexer su alcune di esse, così come mostra il disegno della piedinatura, che è compatibile con quelli della gamma media.

I PIC12C67X DELLA GAMMA MEDIA

I PIC "nani" a 8 bit della gamma media sono divisi in quattro modelli fondamentali; si chiamano PIC12C671, 672, 673 e 674. La loro principale differenza risiede nella capacità di memoria e nella disponibilità o meno della memoria EEPROM per memorizzare i dati in modo non volatile. Inoltre dispongono di convertitore A/D a quattro canali e di interrupt: carenza importante nei nani della gamma base. I nani della gamma media possono funzionare ad una frequenza massima di 10 MHz,



Controllo della memoria di programma nella gamma base.



Piedinatura dei PIC nani, uguale per i modelli di tutte le gamme.

MODELLO	MEMORIA PROGRAMMATA bytes	EPROM	ROM	EEPROM	RAM bytes	LINEE I/O	FREQUENZA MAX.	TIMERS	CANALI A/D
PIC12C671	1792	1024x14			128	6	10MHz	1 da 8 bit	4
PIC12C672	3584	2048x14			128	6	10MHz	1 da 8 bit	4
PIC12C673	1792	1024x14		16x8	128	6	10MHz	1 da 8 bit	4
PIC12C674	3584	2048x14		16x8	128	6	10MHz	1 da 8 bit	4

Tabella con le principali caratteristiche dei PIC nani della gamma media.

nella tabella della figura sono riassunte le loro caratteristiche.

I PIC18F0XX CON MEMORIA FLASH

Nella politica di potenziamento dei prodotti dotati di memoria FLASH, MICROCHIP ha lanciato sul mercato una coppia di modelli dotati di questo tipo di memoria di programma, capaci di funzionare ad una frequenza di 40 MHz. Nella tabella in basso sono riportate le caratteristiche più importanti.

ARCHITETTURA DEI PIC12C5XX

Conoscendo già il PIC16F84, che è il cervello di Monty, è molto semplice analizzare e interpretare l'architettura dei PIC nani, dato che hanno meno risorse e sono dotati di memoria di capacità inferiore. Lo schema generale dell'architettura dei PIC12C5XX si può scomporre in cinque grandi blocchi:

1°. Memoria di programma EPROM.
2°. Memoria dei dati RAM.
3°. ALU.
4°. Clock e risorse associate.
5°. Linee di I/O e funzioni ausiliarie.

Come tutte le altre, anche la famiglia 12C5XX è costruita sull'architettura Harvard, in cui ci sono due memorie indipendenti, una per contenere le istruzioni, e l'altra per i dati. In questo modo è possibile accedere simultaneamente alle due memorie. La dimensione dei byte della memoria di programma è EPROM, coincide con quella delle istruzioni, vale a dire 12 bit. Nel modello 508 la memoria di programma è di 512 parole, da 12 bit cadauna e nel 509 è di 1024 indirizzi da 12 bit. La memoria RAM dei dati di questi due modelli è di 25 o 41 bytes, a seconda che si tratti del 508 o del 509. I registri specifici che formano l'area SFR della memoria dei dati, occupano le prime sette posizioni dalla 00 Hex alla 06 Hex, i registri di utilizzo generale partono dall'indirizzo 07 Hex. Lo stack ha solo due livelli e non dispone di flag di overflow, è quindi compito del programmatore controllare che non debordi quando si producono gli annidamenti nelle subroutines. Sono ammessi solo due annidamenti, dato che con il terzo lo stack perde uno degli indirizzi di ritorno e cade il sistema logico. L'indirizzamento delle istruzioni arriva sempre dal contatore di programma, mentre quello della memoria dei dati può essere "diretto" quando l'indirizzo è contenuto nel codice dell'istruzione, o "indiretto" quando l'indirizzo dell'operando è contenuto nel registro SFR.

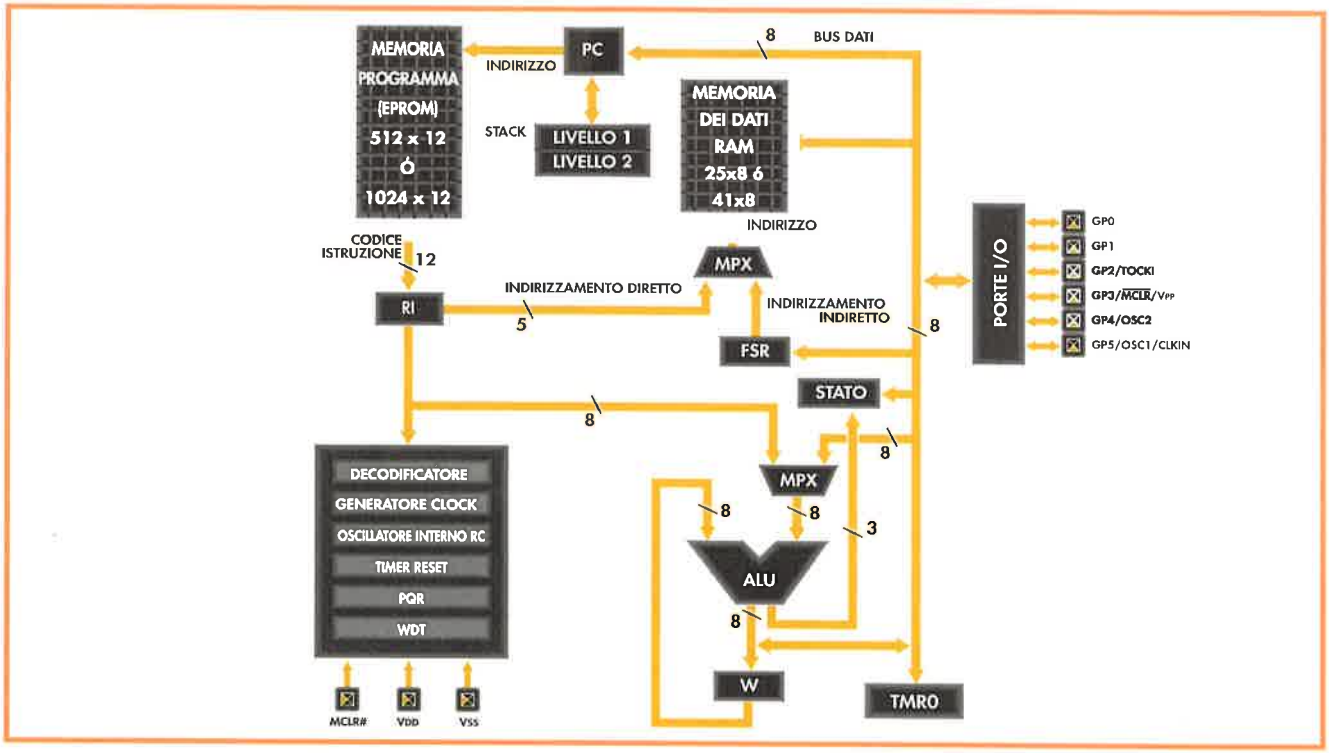
Per realizzare le operazioni logico-aritmetiche, la ALU riceve due operandi da 8 bit. Uno arriva dal registro W e l'altro è un letterale immediato, o arriva dal bus dei dati tramite qualche risorsa interna. Il risultato è caricato nel registro W, inoltre il registro di stato si carica con il valore dei flag che segnalano condizioni particolari del risultato.

Infine è interessante notare che questi PIC nani, con un contenitore da soli 8 pin, mettono a disposizione dell'utente sino a 6 linee di I/O, denominate GP0:GP5, utilizzando per alcune funzioni il multiplexaggio.

Così come succedeva nel PIC16F84, i nani dalla gamma base utilizzano la tecnica della segmentazione nell'esecuzione delle istruzioni, per cui dividono il compito del processore in due stadi. Nel primo si realizza la ricerca dell'istruzione suc-

MODELLO	MEMORIA PROGRAMMATA bytes	FLASH	EEPROM	RAM (bytes)	LINEE I/O	FREQUENZA MAX.	TIMERS
PIC18F010	2048	1024x14	64x8	256	6	40MHz	1 da 16 bit
PIC18F020	4096	2048x14	64x8	256	6	40MHz	1 da 16 bit

Tabella che raccoglie le caratteristiche più importanti dei PIC nani con memoria FLASH.



Architettura interna dei PIC12C5XX.

cessiva e nel secondo si esegue quella in corso. Questo parallelismo implicito permette che tutte le istruzioni, eccetto quelle di salto, si realizzino in un ciclo di istruzioni, che equivale a quattro cicli di clock. Quindi se il microcontroller funziona a 4 MHz, tutte le istruzioni, eccetto quelle di salto, impiegano un microsecondo ad essere eseguite. Le istruzioni di salto impiegano due microsecondi.

PIEDINATURA DEI PIC NANI DELLA GAMMA BASE

I cinque modelli di PIC nani della gamma base (508A, 509A, CR509A, CE518 e CE519) hanno la piedinatura riportata nella figura, nella quale si può vedere che solo quattro pin hanno una sola funzione, mentre i restanti quattro ne supportano diverse, multiplexate nel tempo. I pin 1 e 8 ricevono l'ingresso della tensione di alimentazione: VDD che è l'alimentazione positiva e VSS che è la massa. I restanti 6 pin si possono configurare come ingressi o uscite, ad eccezione di GP3 che funziona solo come ingresso.

Questo pin supporta altre due funzioni: la prima come MCLR# (richiesta esterna del Master Clear Reset) e la seconda come ingresso della tensione speciale per la programmazione, VPP. GP4 svolge anche la funzione di uscita dell'oscillatore e collegamento del cristallo di quarzo. GP5 funziona come uscita del processore e connessione del cristallo di quarzo. Inoltre GP0 e GP1 sono incaricati di ricevere i dati in serie e gli impulsi di clock per la sua sincronizzazione nel modo di programmazione seriale del dispositivo.

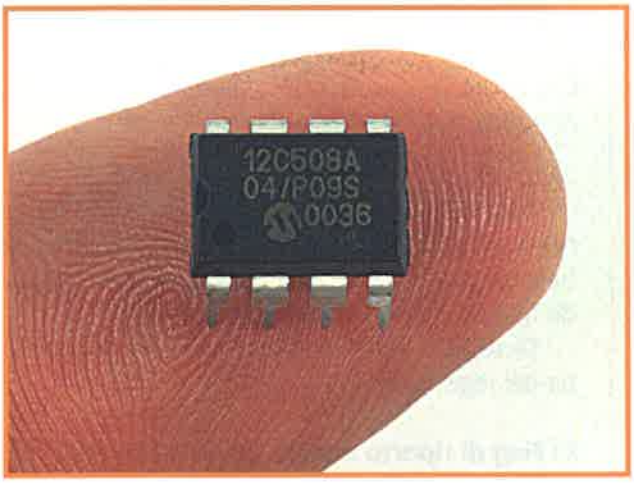


Piedinatura corrispondente a tutti i PIC12C5XX

Il PIC12C508A all'interno

SOMIGLIANZE E DIFFERENZE

I PIC nani da 8 pin hanno diverse cose in comune: fra queste, le principali sono il contenitore e la piedinatura. Le differenze sono dovute alla capacità della loro memoria, alla disponibilità o meno del convertitore A/D, alla possibilità di gestire interrupt e al repertorio di istruzioni a cui rispondono, secondo l'appartenenza alla gamma base o media. I nani della gamma base dispongono di un repertorio di 33 istruzioni, mentre quelli della gamma media supportano due istruzioni in più.



Fotografia del PIC12C508A.

Dato che conosciamo già il nostro vecchio amico PIC16F84 e vista la grande somiglianza, lo studio e la programmazione dei PIC nani risulterà un lavoro più semplice. Perché è stato scelto il nano PIC12C508A? Ci sono due ragioni molto logiche: la prima è che entrambi si assomigliano parecchio, oltre ad essere molto diffusi, la seconda perché è il microcontroller che si utilizza per pilotare la pinza che Monty possiede in uno dei suoi bracci.

SCHEDA TECNICA DEL PIC12C508A

Dalla nostra descrizione generale dei PIC nani, abbiamo visto che fra i nani della gamma base, il PIC12C508A è quello che ha meno capacità di memoria: la sua memoria di programma è di tipo EPROM e può contenere 512 istruzioni da 12 bit ognuna. La memoria dei dati RAM dispone solo di 25 byte per scrivere operandi di utilizzo generale e 7 indirizzi per i registri specifici di controllo. Funziona con una tensione di alimentazione compresa fra 2,5 e 6,25 VDC, supporta un temporizzatore da 8 bit chiamato TMR0, un Watchdog (WDT) e può lavorare ad una frequenza massima di 4 MHz. Non dispone di interrupt. Tutte le caratteristiche esposte appaiono sintetizzate nella tabella della figura.

PIC 12C508 A

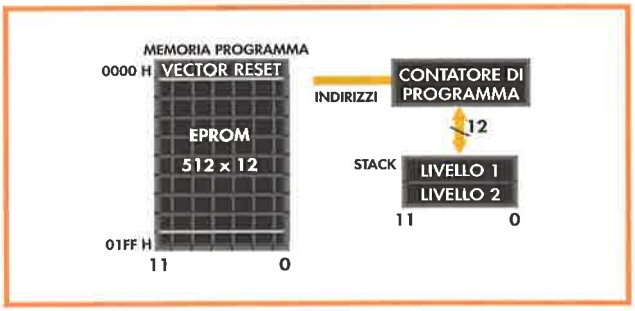
MEMORIA PROGRAMMATA EPROM	MEMORIA DATI RAM	TEMPORIZZATORI	LINEE I/O	ALIMENTAZIONE	FREQUENZA MASSIMA	CONTENITORE
512x12	25x8	1TMR0 + WDT	6	2,5 - 6,25 VDC	4 MHz	8 pin

Caratteristiche fondamentali del PIC12C508A.

L'architettura dei PIC nani è già stata descritta in modo generale, e in essa si è messa in rilievo l'indipendenza fra la memoria di programma e quella dei dati, tipica dell'architettura Harvard.

ORGANIZZAZIONE DELLA MEMORIA DI PROGRAMMA

Il PIC12C508A con cui si controlla la pinza del braccio di Monty, possiede una memoria di programma capace di contenere 512 istruzioni da 12 bit. La dimensione degli indirizzi della memoria coincide con quella delle



Struttura interna e indirizzamento della memoria di programma del PIC12C508A.

istruzioni, per cui il contatore di programma si incrementa normalmente, ogni volta che ne viene eseguita una.

Gli indirizzi di questa memoria si estendono da 0000 Hex sino a 01FF Hex, come si può osservare nella figura. L'altro nano della gamma base, il PIC12C509A, ha il doppio di capacità, per cui il suo range di indirizzi va da 0000 Hex a 03FF Hex.

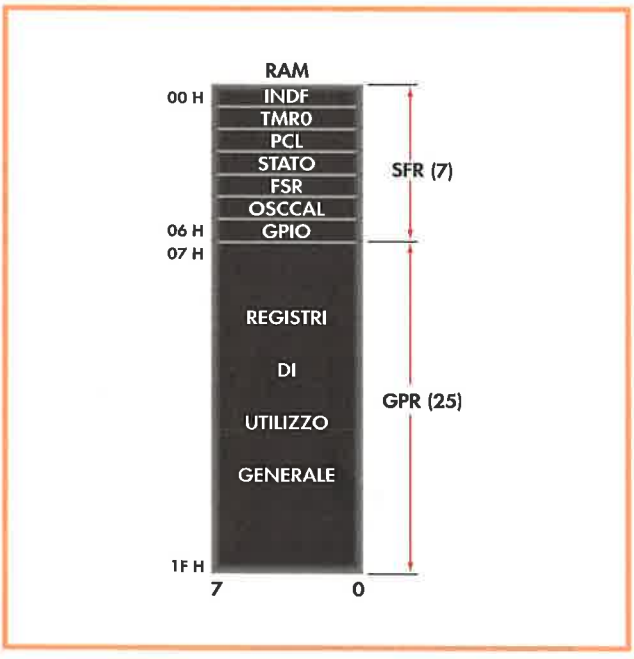
ORGANIZZAZIONE DELLA MEMORIA DEI DATI

Il PIC12C508A dispone di un solo banco di memoria RAM (Banco 0) per scrivere i dati. Questo banco ha 32 indirizzi da un byte di lunghezza ognuno. Il suo range di indirizzi parte dall'indirizzo 00 Hex sino a 1F Hex.

Dei 32 registri della RAM, i primi 7 sono dedicati ai registri di controllo del processore, questa zona viene definita zona dei Registri Specifici (SFR), mentre i rimanenti 25 sono di utilizzo generale (GPR). La distribuzione e l'assegnazione delle prime sette posizioni sono riportate nella figura in basso.

I REGISTRI SPECIFICI SFR

Le prime sette posizioni della RAM funzionano come registro di controllo del microcontroller, e i bit di ognuna governano le funzioni più importanti delle risorse del processore. Abbiamo già conosciuto la maggioranza di questi bit occupandoci del PIC16F84. Il registro INDF, che occupa l'indirizzo 00 Hex, non esiste in realtà, perché non è implementato fisicamente. Si fa riferimento a questo registro quando si punta ad un operando in una istruzione con indirizzamento indiretto, però in questo caso l'indirizzo dove si trova l'operando si ottiene dal contenuto del registro FSR, che occupa l'indirizzo 04 Hex.



Organizzazione della memoria dei dati RAM del PIC12C508A.

All'indirizzo 01 Hex c'è il TMR0, ovvero il valore che si carica nel TMR0 per iniziare la temporizzazione. Gli 8 bit meno significativi del contatore di programma sono contenuti nel registro PCL, all'indirizzo 02 Hex.

IL REGISTRO DI STATO

Uno dei registri più utilizzati nei programmi, è il registro di STATO, che si trova all'indirizzo 03 Hex.



Distribuzione dei bit del registro di STATO.

Degli otto bit di cui è composto, il penultimo non ha nessuna funzione. Dei rimanenti, 5 sono di lettura/scrittura e due si possono solo leggere (TO# e PD#), secondo quanto riportato nella figura in alto.

Di seguito analizzeremo le funzioni di ognuno dei bit del registro di STATO.

C: Flag di riporto e resto (carry e borrow)

Nell'istruzione di somma ADDWF, questo flag va a 1 quando c'è del riporto nell'ottavo bit, e nella sottrazione SUBWF va a zero quando c'è del resto nell'ottavo bit.

Nell'istruzione di rotazione RRF, si carica con il bit meno significativo dell'operando, e nella rotazione RRL si carica con il bit più significativo dell'operando.

DC: Flag di riporto e resto intermedio

Nell'istruzione di somma ADDWF va a 1 quando c'è riporto nel quarto bit (operazione con BCD), mentre nella sottrazione va a zero quando c'è resto nel quarto bit.

Z: Flag di risultato zero

Va a 1 automaticamente quando un'operazione logico-aritmetica dà come risultato zero.

PD#: Flag di "power-on"
Vale 1 quando si collega l'alimentazione VDD o si esegue l'istruzione CLRWDT. Si pone a zero quando si esegue l'istruzione SLEEP.

TO#: Flag di "time-out"
Collegando l'alimentazione, eseguendo l'istruzione CLRWDT o la SLEEP, si pone a 1, e quando vale zero significa che il WDT è andato in overflow.

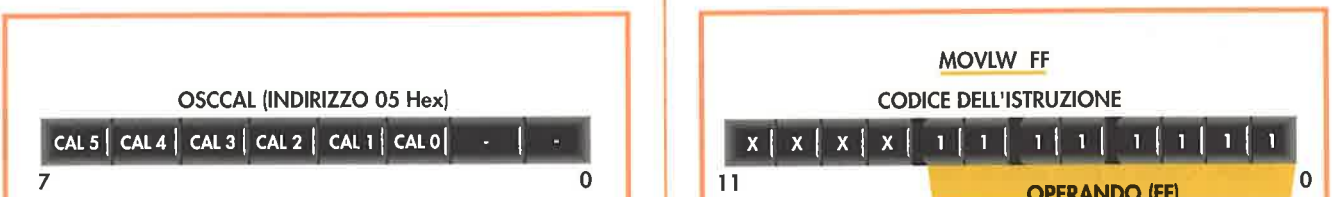
PA0: Selezione della pagina di memoria
La memoria di programma dei PIC 12CXXX si divide in pagine, ognuna delle quali è composta da 512 indirizzi da 12 bit. Il PIC12C508A dispone solo della pagina 0 dato che ha solo 512 indirizzi, invece il 509A ha 1024 indirizzi ed è composto dalla pagina 0 e dalla pagina 1. Il bit PA0 deve essere messo a 1 quando si desidera accedere agli indirizzi della pagina 1, compresi fra 200 e 3FF Hex. Va messo a 0 quando si accede agli indirizzi compresi fra 000 e 1FF Hex.

GPWUF: Bit di reset per i pin della porta GPIO
Quando il microcontroller entra in stato di riposo dopo aver eseguito l'istruzione SLEEP, esiste un modo di svegliarlo producendo un cambio di stato logico su alcune delle linee di I/O della porta GPIO. Questi pin risvegliano il PIC appositamente programmato, generando un Reset nel microprocessore. Se GPWUF vale 1 il Reset è stato provocato da un cambio di stato su uno dei piedini della porta GPIO; se questo bit vale 0, il Reset è stato provocato da altre cause.

I REGISTRI OSCCAL E GPIO

Negli indirizzi 05 e 06 Hex si trovano i registri OSCCAL e GPIO, rispettivamente. Come vedremo più avanti, il PIC12C508A ha la possibilità di funzionare con un oscillatore interno da 4 MHz, con il 5% di tolleranza, frequenza che viene calibrata dalla fabbrica.

Reinizializzando il microcontroller, il registro W si carica con il valore di calibrazione di questo oscillatore, che è stato determinato in fabbrica, e che occupa i 6 bit più significativi. In ogni caso, per adempiere il suo compito questo valore deve essere caricato nel registro OSCCAL, il che suppone di iniziare i programmi spostando il valore di calibrazione da W a OSCCAL, mediante l'istruzione MOVWF OSCCAL. In ultimo, il registro GPIO è quello incaricato di supportare le linee



Distribuzione dei 6 bit
che contengono il valore di calibrazione per l'oscillatore interno.



Distribuzione dei bit
del registro GPIO per gestire le 6 linee di I/O del PIC12C508A.

di I/O digitali con le quali si adattano le periferiche del mondo esterno. GPIO dispone di 6 linee di I/O chiamate GP0, GP1, GP2, GP3, GP4, e GP5; tutte queste linee possono lavorare come ingressi o uscite, ad eccezione di GP3 che può solo essere un ingresso. Come si esporrà più avanti, questo registro è utilizzato per la configurazione delle linee e per contenere l'informazione digitale da trasferire tramite le linee che lo compongono.

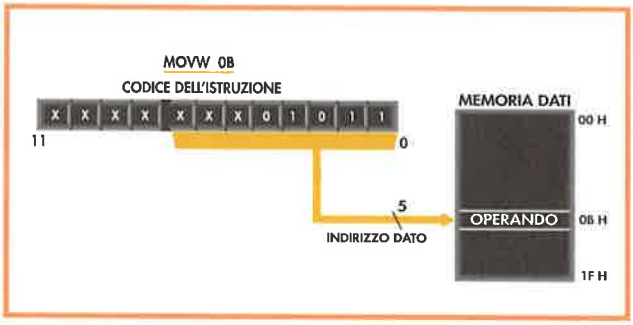
INDIRIZZAMENTO DEI DATI

La selezione di un dato o di un operando nel PIC12C508A si può realizzare in tre modi, che ricevono il nome di "modo di indirizzamento" e che sono i seguenti:

1°. Immediato
2°. Diretto
3°. Indiretto

Nel modo immediato l'operando è incluso nel formato della propria istruzione, generalmente occupa gli

Nell'istruzione "MOVLW FF", l'operando FF è un letterale,
o valore immediato, che è contenuto nell'istruzione stessa.

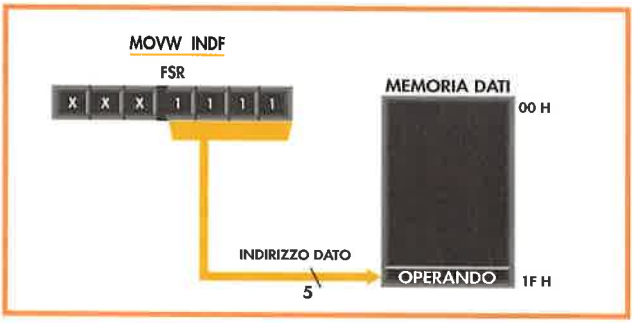


L'indirizzo dell'operando si trova nei 5 bit meno significativi del codice dell'istruzione.

8 bit meno significativi dei 12 che compongono il formato. Frequentemente questo valore viene caricato nel registro W.

Nell'indirizzamento diretto, l'operando occupa alcune delle posizioni della memoria dei dati RAM che si selezionano con i 5 bit di indirizzo compresi nella propria istruzione, occupando le ultime posizioni.

Nel PIC12C508A, dato che c'è un solo banco di memoria dei dati, è sufficiente indicare questi 5 bit, però negli altri nani che hanno più banchi dobbiamo



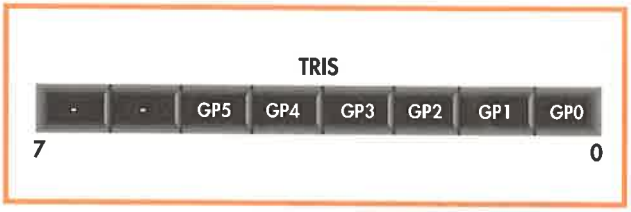
L'indirizzo dell'operando, nell'indirizzamento indiretto, risiede nei 5 bit meno significativi di FSR.

utilizzare i bit 5 e 6 del registro FSR per selezionare i banchi stessi, come rappresentato nella figura.

Nel caso del modo di indirizzamento indiretto, nello mnemonico dell'istruzione si usa come operando il registro INDF, che non è implementato fisicamente. Nella realtà si utilizzano i 5 bit meno significativi del registro FSR per selezionare l'indirizzo dell'operando nella RAM. Se il PIC avesse diversi banchi si utilizzerebbero i bit 5 e 6 di FSR, ma questo non è il caso del 508A, come si può vedere nella figura.

I REGISTRI TRIS E OPTION

Questi due registri non dispongono di indirizzi definiti nella zona SFR della memoria RAM. Si utilizzano automaticamente, quando si eseguono le istruzioni TRIS e OPTION. Il registro TRIS si utilizza per configurare come ingresso o come uscita le linee della porta GPIO; ricordate che la linea GP3 funziona solo come ingresso. Se quando si carica il registro TRIS uno dei suoi bit vale 1, la linea di I/O corrispondente rimane configurata come ingresso, se vale zero invece come uscita.



Distribuzione dei bit di configurazione delle linee di I/O nel registro TRIS.

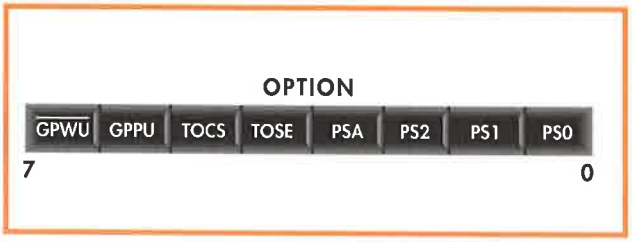
Ad esempio, se vogliamo configurare come uscita le linee GP2, GP1 e GP0 dobbiamo eseguire le seguenti sequenze di istruzioni:

```
MOVLW   b'00111000'
TRIS    GPIO
```

Le tre linee meno significative di GPIO sono configurate come uscite e le altre tre come ingressi. Se ora vogliamo portare sulle linee di uscita il valore binario GP2 = 1, GP1 = 0 e GP0 = 1, si eseguiranno le seguenti istruzioni:

```
MOVLW   b'00000101'
MOVWF   GPIO
```

Infine, il registro OPTION, anch'esso senza indirizzi assegnati nella RAM, e che si carica con il valore di W eseguendo l'istruzione OPTION, è dedicato principalmente al controllo del temporizzatore TMR0 e del Watchdog, che si studierà in modo approfondito più avanti. Nella figura è riportata solo la distribuzione dei bit di questo registro.



Distribuzione dei bit del registro OPTION, incaricato di governare il TMR0 e il WDT.

L'oscillatore, il TMRO e le linee di I/O del PIC12C508A

CLASSI DI OSCILLATORI

Il PIC12C508A ha quattro modi di lavoro, in base al tipo di oscillatore che si utilizza per generare la frequenza di funzionamento:

1. – EXTRC: Oscillatore esterno basato su RC.
2. – INTRC: Oscillatore interno a 4 MHz.
3. – XT: Oscillatore esterno con cristallo di quarzo da 100 KHz sino a 4 MHz.
4. – LP: Oscillatore esterno con cristallo a basso consumo da 32 KHz sino a 200 KHz.

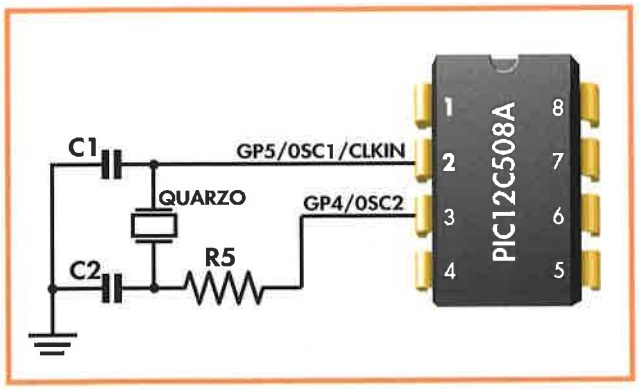
Quando si lavora con oscillatori esterni basati su cristalli di quarzo o risonatori ceramici, i piedini GP5/OSC1/CLKIN e GP4/OSC2, supportano i piedini del quarzo. I valori dei condensatori C1 e C2 mostrati nella figura, per i modi XT da 1 a 4 MHz e per il modo LP da 32 KHz, di solito sono da 15 pF.

Quando è molto importante ridurre al massimo il consumo di energia, si configura l'oscillatore esterno in modo LP. In questo caso, lavorando a 32 KHz, il consumo di corrente è di pochi microampere, invece dei milliampere necessari per lavorare in modo XT a 4 MHz. Nel modo di lavoro con l'oscillatore esterno, il generatore degli impulsi di clock può essere un circuito indipendente, in questo caso si collega al piedino GP5 e la frequenza può essere prelevata dal piedino GP4.

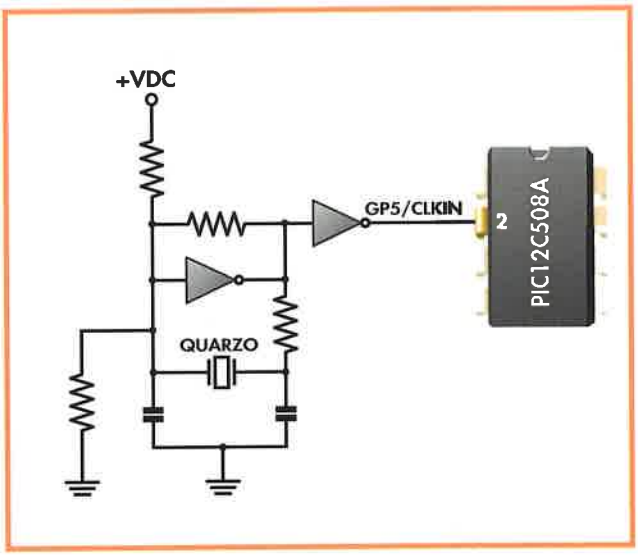
Nelle figure seguenti sono proposti gli schemi di un oscillatore esterno, basato su circuiti indipendenti con porte TTL e cristalli di quarzo in configurazione parallela e seriale. I valori dei componenti passivi sono forniti da MICROCHIP, nelle specifiche tecniche.

Infine nella figura è riportato lo schema di un oscillatore R-C.

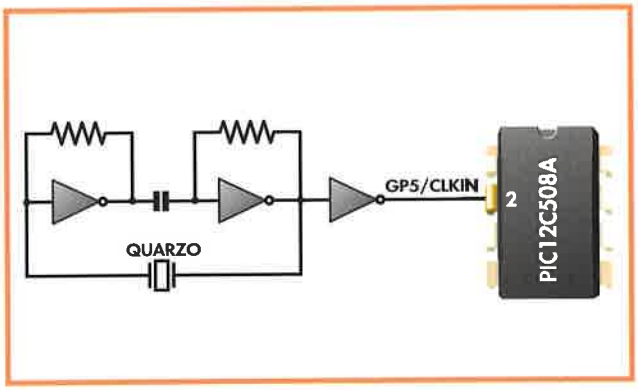
La modalità R-C dell'oscillatore è molto semplice ed economica, però non offre eccessiva stabilità nella frequenza.



Schema del collegamento del cristallo di quarzo negli oscillatori esterni.



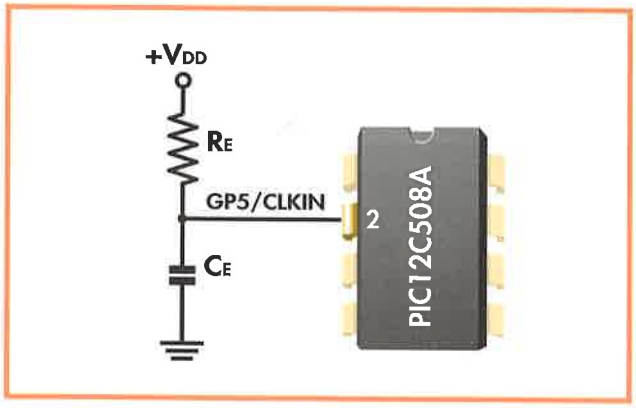
Generatore esterno con porte TTL e cristallo di quarzo in configurazione parallela.



Generatore esterno con porte TTL e cristallo di quarzo in configurazione seriale.

MICROCONTROLLER

Risorse di base del PIC12C508A



Disposizione di un oscillatore esterno di tipo R-C.

OSCILLATORE INTERNO

Con l'obiettivo di lasciare liberi la maggior parte dei pochi piedini del PIC12C508A per la gestione delle periferiche e le funzioni delle risorse interne, l'oscillatore che genera la frequenza di lavoro può essere interno; in questo modo non dovremo utilizzare dei pin per il collegamento di componenti esterni. Questa è la forma di lavoro in modo INTR.

Questo oscillatore interno viene calibrato in fabbrica per oscillare ad una frequenza di 4 MHz, con un 5% di tolleranza, quando è alimentato a 5 VDC, ad una temperatura di 25°C.

Per la corretta operatività di questo oscillatore, dobbiamo scrivere nel registro OSCCAL (Calibrazione dell'Oscillatore) un valore di calibrazione di 6 bit, che è stabilito dal costruttore del chip. Questo valore si carica automaticamente nel registro W quando si reinizializza il processore, per questo all'inizio del programma si utilizza l'istruzione MOVWF OSCCAL, per traslare il valore di calibrazione da W a OSCCAL. Nella figura pubblicata qui sotto è riportata la distribuzione dei bit del valore di calibrazione nel registro OSCCAL che occupa l'indirizzo 05 Hex.



Distribuzione dei bit che formano il valore della calibrazione per l'oscillatore interno.

I TEMPORIZZATORI

Il PIC12C508A dispone di due temporizzatori da 8 bit. Uno è di utilizzo generale e si impiega in tutte le applicazioni che richiedono un controllo preciso del tempo, l'altro è il Watchdog, il cui compito è resettare il microcontroller quando il programma si ferma in modo imprevisto, e cessa il suo periodico aggiornamento. Entrambi i temporizzatori funzionano come contatori ascendenti che si caricano con un determinato valore e si incrementano ad ogni impulso di clock. Quando vanno in overflow, cioè quando passano dal valore FF al valore 00 Hex, generano un avviso, ponendo a uno un flag, o resettando il microprocessore, come rappresentato nella figura successiva.

Dato che i temporizzatori da 8 bit hanno un range di conteggio ridotto, dispongono di un divisore di frequenza che riduce il numero degli impulsi, aumentando così il conteggio massimo a cui possono arrivare. Questo divisore divide la frequenza degli impulsi applicati per le successive potenze del 2. Nel caso del WDT il range di divisione arriva sino a 128, mentre se si tratta del TMR0 arriva sino a 256. Il controllo dei parametri fondamentali dei temporizzatori è fatto tramite il registro OPTION.

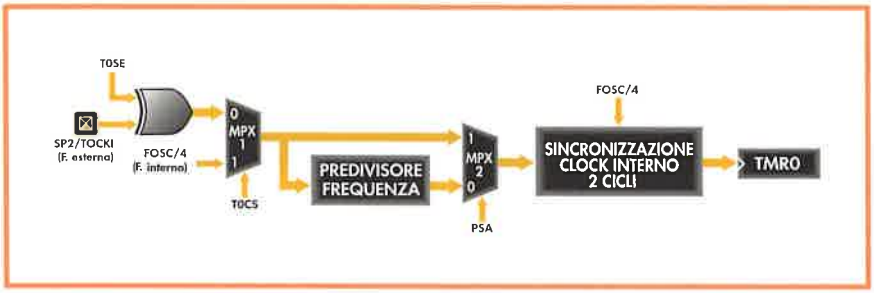


Struttura dei temporizzatori del PIC12C508A.

IL TMR0

È l'unico temporizzatore di utilizzo generale di cui dispone il PIC 12C508A, e può utilizzare il divisore di frequenza quando si scrive uno 0 nel bit PSA del registro OPTION. Quando il PSA vale 1 il divisore di frequenza lo utilizza il WDT.

Come frequenza di clock, il TMR0 può scegliere se utilizzarne una esterna indipendente, che si collega al piedino OCKI, oppure quella interna dell'oscillatore Fosc/4. Per funzionare nel primo caso dobbiamo impostare il bit T0CS a uno mentre per il secondo caso bisogna porre questo bit a zero. Osservate nello schema



Controllo della frequenza nel TMR0.

della figura il funzionamento della linea T0SC, controllata dal multiplexer.

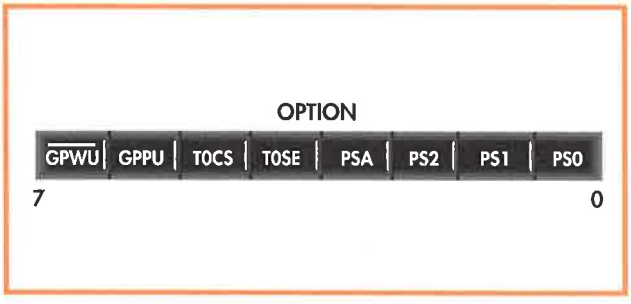
Il segnale T0SE seleziona il fronte attivo dell'impulso di clock che produce l'incremento di valore, quando questo proviene da un generatore esterno. Se vale zero sceglie il fronte di salita e se è uno quello di discesa.

Il valore del registro TMR0 che occupa l'indirizzo 01 Hex dell'area SFR si può leggere e scrivere tramite il bus dei dati interno. Quando si desidera contare un numero definito di impulsi, il valore con cui si deve inizializzare il TMR0 è il valore massimo FF, meno il valore desiderato più una unità, dato che l'overflow si produce passando dal valore FF a 00 Hex.

IL REGISTRO OPTION

Il suo compito principale è configurare il comportamento dei temporizzatori. Non occupa un indirizzo reale dell'area SFR, esiste un'istruzione speciale che carica in questo registro il valore che contiene il registro W. I bit di questo registro possono solo essere scritti, e quando avviene un reset, essi sono inizializzati al valore 1. Nella figura si mostra la struttura interna del registro OPTION.

I tre bit meno significativi di questo registro, PS0 – PS1 – PS2, stabiliscono il range di divisione degli impulsi di clock, da parte del divisore di frequenza, come si può vedere nella tabella della figura. Osservate che per il TMR0 il range è il doppio di quello del WDT.



Struttura interna del registro OPTION.

Se il bit PSA vale 0, si assegna il divisore al TMR0, in questo caso funziona all'ingresso degli impulsi, cioè come predivisore. Per contro, quando PSA vale 1 il predivisore si assegna al WDT, dove funziona all'uscita del medesimo, cioè come postdivisore. Il bit T0SE di OPTION sceglie il fronte attivo dell'impulso di clock: quando vale 1 è il fronte di discesa, quando vale 0 è quello di salita. T0SC seleziona il tipo di generatore di frequenza. Quando vale 0 si utilizza per il conteggio la frequenza interna del processore, Fosc/4. Se T0CS vale 1 il TMR0 si incrementa con gli impulsi di clock prodotti da un generatore esterno collegato al pin GP2/T0CKI.

PS2-PS1-PS0	PREDIVISORE TMR0	PREDIVISORE WDT
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

Tabella che indica i valori di funzionamento del divisore di frequenza.

Quando il bit GPPU si carica con uno 0, le linee di I/O GP0, GP1 e GP3 funzionano come ingressi, con una debole polarizzazione dovuta alle resistenze di pull-up; se GPPU = 1 queste resistenze di polarizzazione vengono disabilitate. Il bit più significativo di OPTION, GPWU#, ha una funzione particolare, nel repertorio delle istruzioni dei PIC della gamma base ne esiste una chiamata SLEEP, che manda il microcontroller nel modo di basso consumo, o di riposo. In questo stato non si eseguono le istruzioni, si congela lo stato precedente, e si spende il minimo di energia per mantenere il sistema "addormentato".

Per "risvegliarsi" da questa situazione e riprendere il normale lavoro dobbiamo cambiare lo stato logico dei piedini di I/O GP0, GP1 o GP3. Se GPWU# è a 0 i pin menzionati sono abilitati a risvegliare il PIC, quan-

do su di essi avviene un cambiamento di stato. Il risveglio si realizza producendo un reset, e se GPUW# = 1 si disabilita questa possibilità.

IL WATCHDOG

È un contatore ascendente, simile al TMR0, che ha la particolarità di andare in overflow, e al passaggio dal valore FF al valore 00 Hex genera un reset. Il suo compito è supervisionare il buon funzionamento del programma; si utilizza caricando nel WDT un valore iniziale, passando poi all'esecuzione sequenziale delle istruzioni del programma. Il programmatore dovrà disporre nel programma delle istruzioni di riaggiornamento del WDT, prima che questo vada in overflow, sempre che il programma si svolga normalmente. Se il programma si ferma o finisce in un ciclo infinito, l'aggiornamento del Watchdog non giunge in tempo, e allora quest'ultimo "morde", cioè genera un reset.

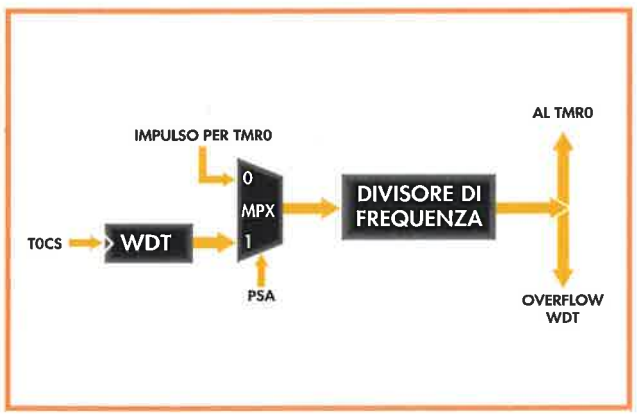
Se il divisore di frequenza è assegnato al WDT, funziona alla sua uscita, come si può vedere dalla figura.

CONFIGURAZIONE DELLE LINEE DI I/O

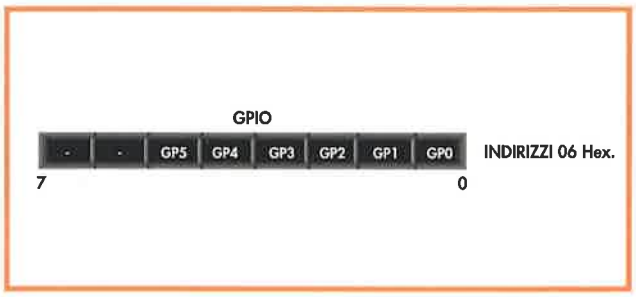
I PIC nani dedicano sei dei loro otto piedini alle linee di I/O multifunzionali, per rendere più alta possibile la connettività fra le periferiche e le risorse interne.

Queste sei linee si chiamano GP0-GP5 e tutte possono funzionare sia come ingressi sia come uscita, ad eccezione di GP3 che può solo funzionare come ingresso. Per controllare queste linee si utilizza il registro GPIO all'indirizzo 06 Hex della RAM dei dati, la cui struttura è riportata nella figura. Nello schema della figura vediamo il circuito elettronico associato ad ogni pin GPx.

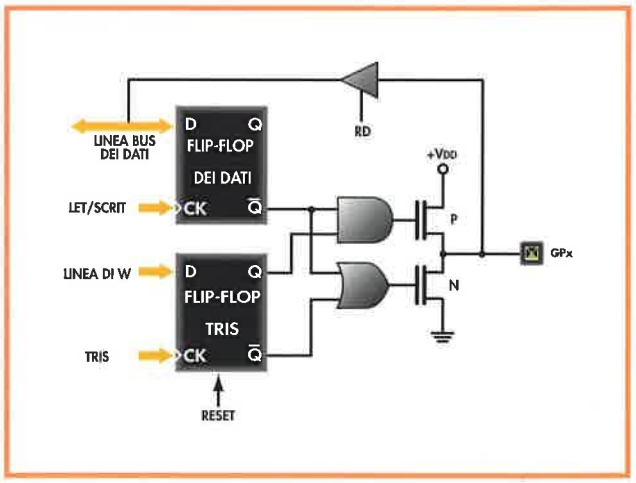
Quando si scrive un 1 nel flip-flop TRIS, il piedino GPx associato assume uno stato di alta impedenza, e il pin si configura come ingresso. Per leggere lo stato logico che si applica a questo piedino tramite l'esterno, basta leggere direttamente il flip-flop dei dati, che è uno dei bit del registro GPIO. Se nel flip-flop TRIS scriviamo uno 0 la porta si configura come uscita, e per portare su di essa un livello logico lo dobbiamo prima scrivere sul flip-flop dei dati che è il bit corrispondente del registro GPIO.



Il divisore di frequenza funziona all'uscita del WDT.



Struttura del registro GPIO.



Schema elettrico del collegamento dei pin di I/O del PIC12C508A.

L'istruzione TRIS GPIO carica il valore memorizzato in W nel flip-flop TRIS e lo configura come ingresso o come uscita. Ad esempio se si desidera configurare la linea GP0 come uscita e il resto come ingressi e in seguito portare su GP0 un livello logico alto, si dovranno eseguire le seguenti istruzioni:

```
MOVLW   b'11111110'   ; Carico W
TRIS    GPIO          ; Configuro GP0 come uscita
MOVLW   b'00000001'   ; Carico W
MOVWF   GPIO          ; Da GP0 esce uno 0
```

Il reset e le istruzioni del PIC12C508A

IL PRINCIPIO

Il PIC12C508A, carente di interrupt, dispone solo del Reset per reiniziare la sequenza che determina il flusso delle istruzioni del programma. Resettare il microcontroller significa assegnare un valore noto a tutti i suoi registri di controllo della zona SFR della RAM dei dati. Fra questi registri si trova il PCL, che dopo il Reset rimane caricato con il valore 1111 1111, il che significa che al successivo impulso di clock, passerà alla prima istruzione da eseguire, posta all'indirizzo 0000 0000, che è la prima della memoria di programma.

CAUSE CHE PROVOCANO IL RESET

La reinizializzazione del PIC si può provocare nei seguenti modi:

1°. Collegamento dell'alimentazione.

STATO

GPWVF	-	PA0	TO#	PD#	Z	DC	C

I bit GPWUF, TO# e PD# determinano la causa che ha provocato il Reset.

INDIRIZZO RAM	REGISTRO	RESET PER COLLEGAMENTO DELL'ALIMENTAZIONE		Reset per attivazione di MCLR#, overflow del WDT o cambio di stato su GPIO	
00	INDF	xxxx	xxxx	uuuu	uuuu
01	TMR0	xxxx	xxxx	uuuu	uuuu
02	PC	1111	1111	1111	1111
03	STATO	0001	1xxx	q00q	quuu
04	FSR	111x	xxxx	111u	uuuu
05	OSCCAL	1000	00--	uuuu	uu--
06	GPIO	11xx	xxxx	11uu	uuuu
-	OPTION	1111	1111	1111	1111
-	TRIS	--11	1111	--11	1111
-	W	qqqq (calibrazione)	qqxx	qqqq (calibrazione)	qquu

Abbreviazioni / -: Non implementato; u: Non cambia; x: Sconosciuto; q: Dipende dalla condizione.

Valori che assumono i bit dei registri di controllo del PIC12C508A dopo un Reset.

2°. Overflow del Watchdog in modo normale o in modo riposo.

3°. Attivazione di MCLR# in modo normale o in modo riposo. Caduta della tensione di alimentazione sul pin GP3/MCLR#/Vpp fra 2,1 e 1,5 VDC.

4°. Al cambiamento di stato del pin GP0, GP1 e GP3. Quando si provoca un Reset, il microcontroller esce dal modo riposo.

Per conoscere la causa che ha provocato il Reset, si utilizzano tre flag del registro di STATO: GPWUF, TO# e P#, che sono presentati nella figura.

Nella tabella della figura sono mostrati i valori che assumono i bit del registro di STATO per le differenti cause di Reset.

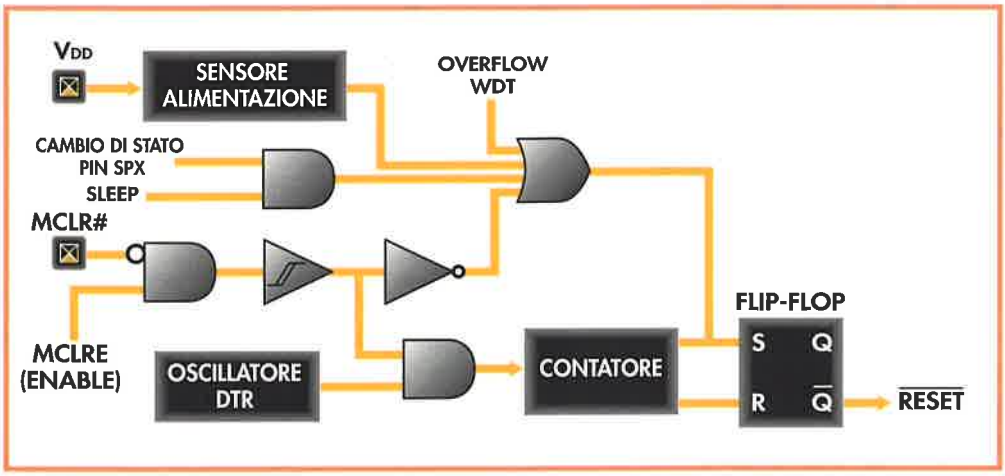
Esistono diverse forme per provocare un Reset, e non tutte le volte i registri di controllo rimangono caricati in modo uguale. Nella tabella della figura sono riportati i valori che assumono i registri SFR del PIC 12C508A dopo che si è prodotto un Reset, sia per il collegamento dell'alimentazione che per l'attivazione del piedino MCLR#, il debordamento del WDT o il cambio di stato su qualcuno dei pin GPx.

GPWUF	TO#	PD#	CAUSA CHE HA ORIGINATO IL RESET
0	0	0	Overflow del WDT in modo riposo
0	0	1	Overflow del WDT in modo normale
0	1	0	Risveglio dal modo riposo per attivazione del MCLR#
0	1	1	Collegamento dell'alimentazione
1	1	0	Reset al risveglio dal modo riposo per cambio pin GPIO
0	non cambia	non cambia	Attivazione MCLR# nel modo normale

Valori dei bit del registro di STATO per le differenti cause di Reset.

MICROCONTROLLER

Reinizializzazione e repertorio delle istruzioni



Schema elettronico del controllo dell'attivazione del Reset.

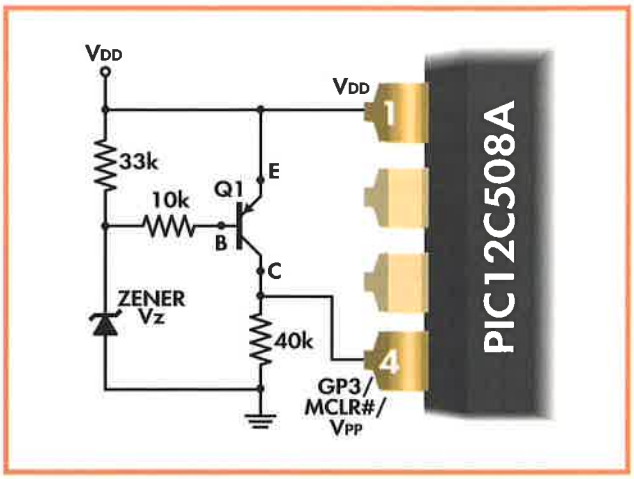
Il circuito per rilevare il buco di tensione sul pin MCLR# si può semplificare eliminando il diodo zener, come si può vedere nella figura, dove è stato sostituito dalla coppia di resistenze R1 e R2. In questo caso il transistor Q1 si blocca e attiva il Reset portando a livello basso il pin MCLR# quando la tensione di alimentazione scende al di sotto di 0,7 [R1 / (R1 + R2)].

SCHEMA ELETTRONICO

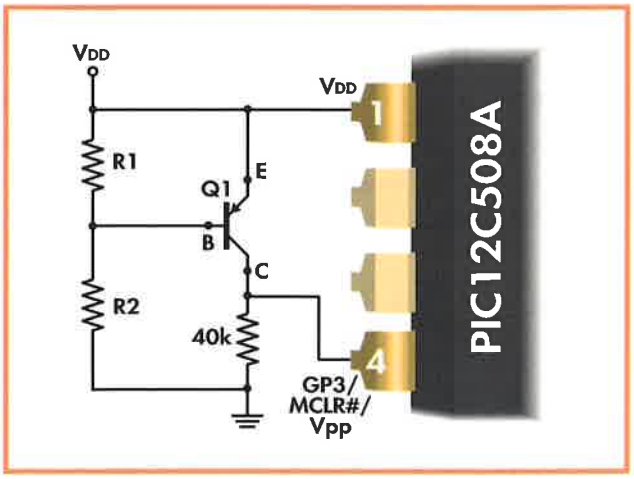
Nello schema elettronico della figura, l'attivazione del Reset interno del processore è provocato in base al valore Q# del flip-flop R-S. Questo dispositivo si pone a 1 quando si attiva il suo ingresso Set, che è controllato dall'uscita di una porta OR a 4 ingressi, ognuno dei quali corrisponde ad una delle cause che generano il Reset. L'attivazione del fronte attivo del Reset è ritardata da un oscillatore DTR e da un contatore a 8 bit, per aspettare che si siano stabilizzate sia la tensione di alimentazione che la frequenza di lavoro.

RESET PER BUCO DI TENSIONE

Si produce un buco di tensione nell'alimentazione quando la tensione VDD che riceve il PIC scende sotto un valore minimo, e poi recupera. In questa situazione è conveniente generare un Reset per evitare malfunzionamenti nel sistema e nei circuiti associati. Questo Master Clear Reset (MCLR) si può programmare in modo che si produca internamente, quando si abilita il segnale MCLRE dello schema elettronico, e anche esternamente, nel qual caso questa funzione è assegnata al pin GP3/MCLR/Vpp che genera un Reset quando la tensione applicata ad esso scende a valori fra 2,1 e 1,5 VDC. Esistono due circuiti classici per implementare questa funzione di rivelazione dell'abbassamento della tensione sul pin menzionato. Nello schema mostrato nella figura al centro, si produce il Reset quando la tensione VDD scende al di sotto del valore della tensione di zener, Vz, più 0,7 Volt corrispondenti alla giunzione emettitore-base del transistor Q1.



Il Reset si produce quando la tensione VDD scende al di sotto di Vz + 0,7 VDC.



Il pin MCLR# genera un Reset quando la tensione scende al di sotto di 0,7 [R1 / (R1 + R2)].

IL SONNO DEL GUERRIERO

In molte applicazioni è frequente che il PIC non abbia niente da fare per lunghi periodi di tempo. Questo succede, ad esempio, nelle macchine distributrici, inoltre è comune nei prodotti piccoli, in cui i PIC nani sono idonei, alimentati a pile, come ad esempio nei microrobot. Ridurre il consumo di energia in queste situazioni è vitale per mantenere il sistema operativo per periodi accettabili. Se il sistema sta aspettando che l'utente prema un tasto, bisogna fare in modo che durante questa attesa, a volte molto lunga, il consumo di energia sia minimo. Per questo si "addormenta" il PIC.

Per fare in modo che il microcontroller entri in stato di riposo, o di basso consumo, bisogna eseguire l'istruzione SLEEP. In questo modo passa dal consumare alcuni milliampere ad aver bisogno solamente di pochi microampere, cosicché una pila può durare degli anni.

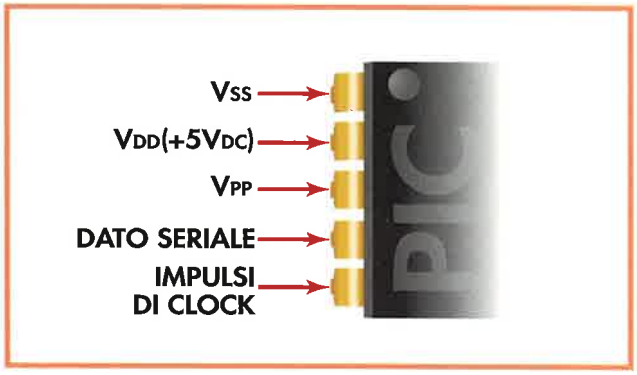
Quando il PIC entra in stato di riposo, i piedini di I/O mantengono lo stato precedente, cessano di eseguirsi le istruzioni e se si utilizza un oscillatore al quarzo questo si fermerà, riducendo ulteriormente il consumo.

Per svegliare il PIC, e farlo uscire dal suo stato di congelamento, bisogna produrre una situazione speciale. Dato che il PIC12C508A non dispone di interrupt, il suo risveglio si realizza generalmente mediante un Reset, che può essere provocato dall'overflow del WDT, dall'attivazione di MCLR# o per il cambio di stato di alcuni piedini GPx. Il bit PC# del registro di STATO si attiva quando si esegue l'istruzione SLEEP, mentre il TO# si attiva quando va in overflow il Watchdog.

PROGRAMMAZIONE SERIALE

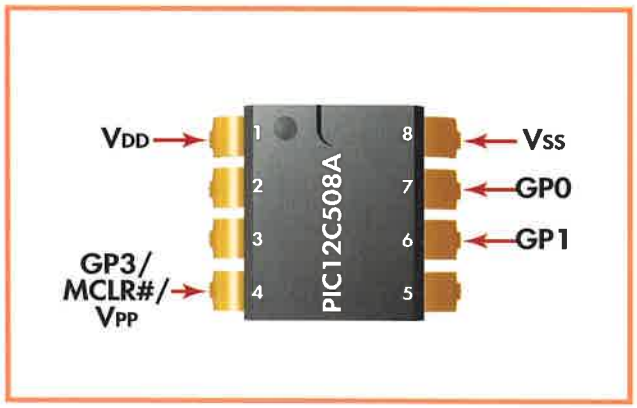
Si può scrivere nella memoria di programma del PIC12C508A in modo seriale, sullo stesso circuito dell'applicazione. Questo permette al costruttore del prodotto finale di produrre la scheda elettronica con il microprocessore senza doverlo programmare, e realizzare la scrittura del programma solo al momento della vendita, garantendo al cliente il firmware più recente e perfezionato.

Come si può vedere nello schema della figura, per la programmazione del PIC sono necessari cinque segnali: VDD e VSS per l'alimentazione, Vpp per l'ingresso della tensione speciale per la programmazione, una linea per inviare i dati in modo seriale al PIC e un'altra per inviare gli impulsi di clock che sincronizzano l'invio dell'informazione seriale.



Collegamenti base per la programmazione in modo seriale del PIC.

Nel caso del PIC12C508A, il pin GP0 funziona come ingresso dei dati seriali, il pin GP1 come ingresso del clock di sincronismo, e il pin GP3/MCLR#Vpp per l'ingresso della tensione speciale di programmazione, come indicato graficamente in figura.



Assegnazione dei pin del PIC12C508A per la sua programmazione seriale.

IL REPERTORIO DI ISTRUZIONI

Sino ad ora abbiamo gestito l'insieme di istruzioni del PIC16F84 che funziona come cervello di Monty, che ha 35 istruzioni da 14 bit di dimensione cadauna. Per manipolare la pinza che possiede Monty su un braccio, si utilizza il PIC12C508A, che è un nano della gamma base. Il suo repertorio di istruzioni ne comprende solo 33, da 12 bit ognuna. Questo fa sì che sia molto facile comprendere la programmazione del nano, dato che la

Reinizializzazione e repertorio delle istruzioni

ISTRUZIONI DEI PIC DI GAMMA BASE

OPERAZIONI CON BIT			OPERAZIONI CON BIT		
NOP		: No Operation	BCF	f,b:	Cancella bit b in f
MOVWF	f	: Muove W a f	BSF	f,b:	Pone a 1 il bit b in f
CLRW		: Cancella W	BTFSC	f,b:	Sottrae b e salta se è 0
CLRF	f	: Cancella f	BTFSS	f,b:	Sottrae b e salta se 1
SUBWF	f,d	: Sottrae W da f			
DECF	f,d	: Decrementa f	**OPERAZIONI CON IMMEDIATI E DI CONTROLLO**		
IORWF	f,d	: OR fra W e f	SLEEP		: Passa al modo riposo
ANDWF	f,d	: AND fra W e f	CLRWDT		: Cancella WDT
XORWF	f,d	: XOR fra W e f	RETLW	k:	Ritorna e carica k in W
ADDWF	f,d	: Somma W e f	OPTION		: Carica in OPTION W
MOVF	f,d	: Muove f	TRISF	f:	Configura le porte di I/O
COMF	f,d	: Complementa f	CALL	a:	Chiamata alla subroutine a
INCF	f,d	: Incrementa f	GOTO	a:	Salta all'indirizzo a
DECFSZ	f,d	: Decrementa f, salta se 0	MOVLW	k:	Muove letterale k a W
RRF	f,d	: Ruota a destra f	ANDLW	k:	AND di W con k
RLF	f,d	: Ruota a sinistra f	IORLW	k:	OR di W con k
SWAPF	f,d	: Scambia le metà di f	XORLW	k:	XOR di W con k
INCFSZ	f,d	: Incrementa f, salta se 0			

Abbreviazioni / **f**: Indirizzi di un registro; **d**: Destinazione; **a**: Indirizzo di programma; **b**: Bit; **k**: Letterale immediato.

Tabella che raccoglie le 33 istruzioni dei PIC della gamma base.

maggior parte delle istruzioni sono simili a quelle del PIC16F84.

ISTRUZIONI PECULIARI

Analizzeremo solo il comportamento delle istruzioni della gamma base che comportano differenze con quelle già note della gamma media.

L'istruzione TRIS serve per configurare le linee della porta GPIO come ingressi o come uscite; con questa istruzione, i bit che precedentemente erano stati caricati nel registro W si caricano sul flip-flop TRIS e configurano la direzione della linea. Un 1 fa sì che la linea corrispondente sia ingresso, uno 0 che sia uscita. Se si desidera configurare le sei linee GPx come ingressi, bisogna eseguire le seguenti istruzioni:

```
MOVLW   b'1111 1111'   ; Si carica il W.
TRIS    GPIO           ; Le GPx funzionano come ingressi.
```

Un'altra istruzione speciale della gamma base è OPTION, che serve per caricare il registro OPTION per il controllo del TMR0 e del WDT con il contenuto di W. L'istruzione GOTO contiene, inclusi nel suo codice a 12 bit, i 9 bit che corrispondono all'indirizzo dell'istruzione a cui bisogna saltare. Ricordatevi che siccome il PIC12C508A dispone solo di 512 parole nella memoria di programma, l'indirizzamento si effettua con 9 bit. Nell'istruzione CALL gli 8 bit meno significativi del suo codice di istruzione sono caricati nel registro PCL dell'area SFR della RAM. Il nono bit del PCL viene messo a 0. Qualcosa di simile accade con le istruzioni che spostano un valore al contatore di programma, nel qual caso gli 8 bit dell'operando si scrivono negli 8 bit meno significativi del PC, mettendo a zero il nono. Secondo le stime di MICROCHIP, la dimensione del codice che si crea per la gamma base è inferiore a quella dei competitori, come si mostra nel grafico seguente.

Per quanto riguarda la velocità di esecuzione, il vantaggio che si ottiene utilizzando i PIC della gamma base è molto più marcato, come si può vedere dal relativo grafico.

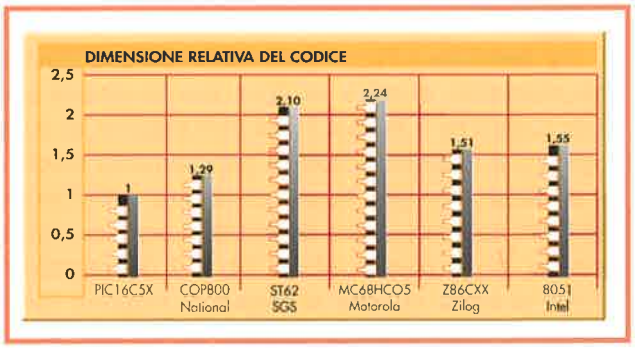
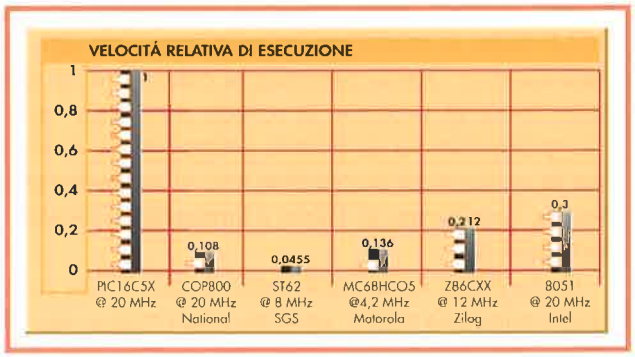


Grafico della dimensione del codice dei PIC base confrontato con quello dei suoi competitori.

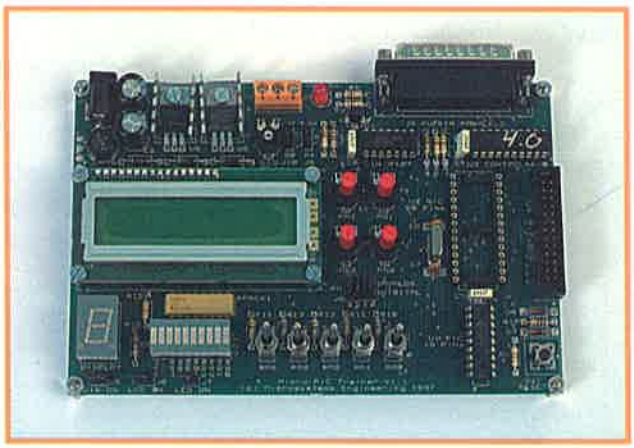


Il grafico evidenzia la differenza di velocità di esecuzione dei PIC base.

Applicazioni con la Micro'PIC Trainer Plus

SCHEDA DI ESPANSIONE DELLE PERIFERICHE

In quest'opera sono presentati diversi strumenti di riconosciuto prestigio internazionale, per gestire i PIC. Fra questi notiamo il sistema di sviluppo Micro'PIC Trainer, che oltre a funzionare come uno scrittore di memorie dispone delle periferiche più comuni utilizzate nelle applicazioni: display LCD, display a sette segmenti,



Fotografia del sistema di sviluppo Micro'PIC Trainer.

interruttori, pulsanti, potenziometri, barra di LED, ecc. il che permette di eseguire in tempo reale i programmi in fase di messa a punto e prova.

Al Micro'PIC Trainer mancano ancora numerose periferiche tipiche dei progetti con i microcontroller, per questo il produttore ha sviluppato altre schede di espansione che ne forniscono di nuove. Queste schede si collegano al sistema di sviluppo tramite il connettore PICBUS. La ditta "Ingeniería de Microsistemas Programados S.L.", ha progettato tre schede di espansione che sono commercializzate sotto il marchio MICROSYSTEMS ENGINEERING:

1. Micro'PIC Trainer Plus
2. Micro'PIC IO
3. MSx84

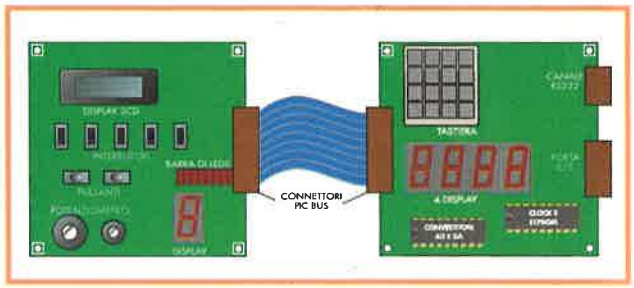
Ognuna di queste schede supporta un insieme di periferiche specifiche, destinate a risolvere lavori comuni in diverse aree di applicazione. Il lettore può ottenere informazioni più dettagliate su questi prodotti, nelle pagine del sito Internet del costruttore. In linea di massima la Micro'PIC Trainer Plus mette a disposizione del progetto da implementare risorse quali tastiera, convertitore AD e DA, porte di I/O, ecc. La Micro'PIC IO è specializzata in applicazioni industriali e la MSx84 nel controllo di motori e sensori.

LA MICRO'PIC TRAINER PLUS

Aggiunge alle risorse della Micro'PIC Trainer quelle elencate di seguito e schematizzate nel disegno della figura:

- Tastiera matriciale 4 x 4
- Barra di diodi LED
- Visualizzatore da quattro display a 7 segmenti
- Convertitori AD e DA
- Canale di comunicazione seriale RS-232
- Porta di I/O da 8 linee
- Clock/calendario e RAM non volatile

La disposizione di tutti gli elementi di questa scheda è riportata nella fotografia alla pagina seguente, in cui si possono vedere i suoi tre connettori di comunicazione con l'esterno: il connettore corrispondente al canale seriale RS-232, quello del PICBUS con cui si adatta alla Micro'PIC Trainer e una fila di morsetti a vite con cui si pongono a disposizione dell'utente le linee di I/O di tutte le risorse.



Schema dell'espansione delle risorse della Micro'PIC Trainer mediante il collegamento alla Micro'PIC Trainer Plus tramite il connettore PICBUS.



Fotografia della scheda Micro'PIC Trainer Plus.

IL BUS I2C

Alcuni PIC hanno integrato sul silicio il bus I2C, e dedicano i piedini RC3 e RC4 a supportare lo scambio di informazioni in serie e gli impulsi di clock. Altri PIC più semplici non dispongono di questa risorsa, e la devono implementare via software, utilizzando le linee RB6 e RB7 per la loro gestione.

Come abbiamo potuto vedere quando è stata descritta la Micro'PIC Trainer Plus, questa scheda utilizza quattro circuiti integrati specializzati per controllare le principali periferiche che supporta:

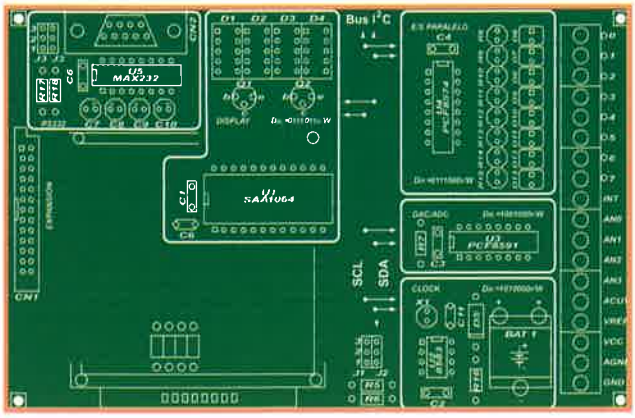
1°. SAA1064: Controller per quattro display a sette segmenti

2°. PCF8591: Convertitore AD e DA

3°. PCF8574A: Porta di I/O da 8 linee digitali

4°. PCF8583P: Clock/calendario più memoria RAM

Sia lo schema dei collegamenti sia il funzionamento delle sezioni che formano la scheda sono state spiegate in precedenza; nella figura si mostra la distribuzione delle risorse sulla superficie della PCB.



Distribuzione dei principali componenti sopra la scheda della Micro'PIC Trainer Plus.

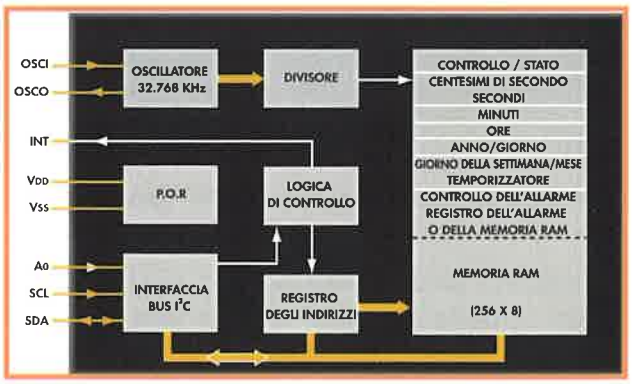
IL SOFTWARE

Insieme all'hardware della Micro'PIC Trainer Plus, il costruttore fornisce un disco che contiene tutti i programmi e le routines che si utilizzano nelle diverse applicazioni e nei progetti dimostrativi del sistema, per iniziare l'utente alla sua gestione. Questi programmi sono progettati esclusivamente a fini didattici e sono abbondantemente documentati per facilitarne la comprensione, andando oltre la descrizione tecnica di pura programmazione.

GESTIONE DEL MODULO CLOCK/CALENDARIO E SRAM

Per fornire al lettore un'idea delle grandi possibilità della Micro'PIC Trainer Plus, studiamo in modo sommario la sezione che controlla il circuito integrato I2C PCF8583, che contiene un clock/calendario e una memoria RAM. Si tratta di un circuito CMOS a basso consumo controllato con il bus I2C.

Contiene una memoria SRAM da 256 byte, e un registro che la indirizza e si autoincrementa leggendo o scrivendo una delle sue posizioni. Dispone di una linea esterna A0 che permette il collegamento al medesimo sistema di due di questi dispositivi. Per il controllo delle risorse interne sono utilizzati diversi registri che sono



Schema a blocchi del PCF8583.

implementati sui primi 16 indirizzi della memoria SRAM. Lo schema a blocchi è rappresentato nella figura riportata qui sopra.

Si alimenta con una tensione compresa fra 2,5 e 6 VDC e la tensione per il mantenimento dei dati nella memoria può oscillare fra 1 e 6 VDC. Il clock/calendario funziona con formato da 12 o 24 ore, con una base tempi da 50 o da 32.768 Hz.

Ha funzione di allarme, di temporizzazione, contatore di eventi e interrupt. La piedinatura, la descrizione

di ogni pin e le caratteristiche del funzionamento sono riportate nelle figure seguenti.

DESCRIZIONE FUNZIONALE DEL PCF8583

Funziona in base a un oscillatore interno da 32,768 Hz e dispone di un circuito POR (Power On Reset) che inizializza il dispositivo quando si collega la tensione di alimentazione. I primi 8 indirizzi della RAM (00 – 07) funzionano come registri specifici di controllo. Il primo registro, situato all'indirizzo 00, funziona come registro di controllo e di stato, mentre i successivi 7 contengono i parametri della funzione oraria. Gli indirizzi da 08 a 0F si possono utilizzare come registri per il controllo di allarmi, oppure come indirizzi di utilizzo generale.



Piedinatura del PCF8583.

Mediante il registro di controllo e di stato, si imposta il temporizzatore per lavorare o con la base tempi, che può essere 50 e 32.768 Hz, o come contatore di eventi esterni. Lavorando in modo orario i giorni, mesi, ore, minuti, secondi e centesimi sono espressi nel formato BCD. Nel modo contatore di eventi esterni, gli impulsi da contare sono applicati al piedino OSCI, e si possono scrivere, dato che può funzionare come contatore di eventi da 6 digit.

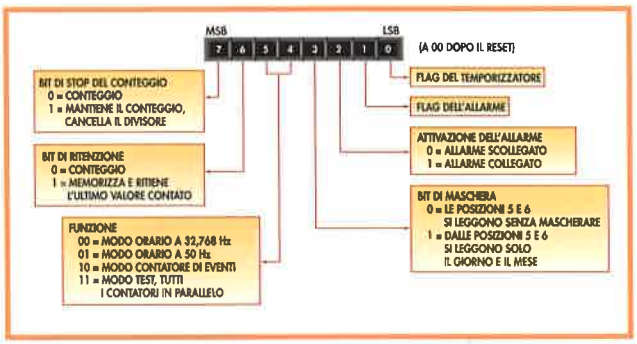
Nella figura riprodotta qui sopra è riportata la distribuzione del registro di controllo e di stato, in essa si può apprezzare l'esistenza di un bit che attiva l'allarme se si desidera. Questo registro si pone a zero dopo il reset. Quando si produce una situazione di allarme si attiva il flag corrispondente. L'uscita INT passa a zero. I flag mantengono il loro stato fino a che si realizza su di essi un'operazione di scrittura. Se non si usa la funzione di allarme, i registri da 09 a 0F possono essere adibiti a contenere dati.

N° DI PIN	SEGNALE	DESCRIZIONE
1	OSCI	Ingresso dell'oscillatore da 50 Hz o ingresso degli eventi
2	OSCO	Uscita dell'oscillatore
3	A0	Ingresso esterno degli indirizzi
4	Vss	Negativo dell'alimentazione
5	SDA	Linea dei dati del bus I²C
6	SCL	Linea del clock del bus I²C
7	INT	Uscita dell'interrupt in collettore aperto
8	Vdd	Positivo dell'alimentazione

Descrizione funzionale dei piedini del PCF8583.

PARAMETRO	SIMBOLO	MINIMO	MASSIMO	UNITÀ
Tensione di alimentazione	VDD	-0,8	+7	V
Consumo di corrente	IDD		50	mA
Tensione di ingresso	VI	-0,8 a VDd	VDD+0,8	V
Corrente di ingresso	II		10	mA
Corrente di uscita	IO		10	mA
Dissipazione di potenza	PTOT		300	MW
Potenza di uscita	PO		50	mW

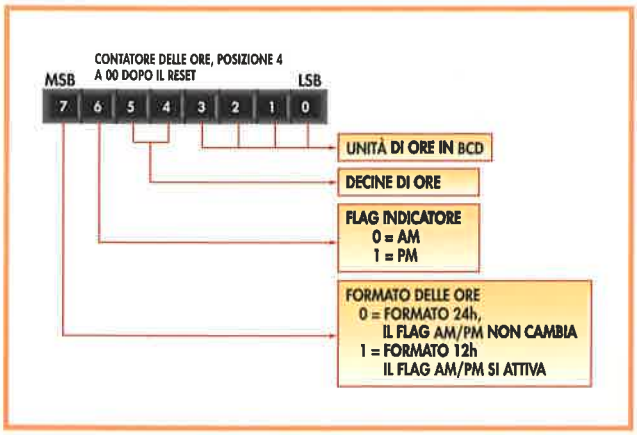
Caratteristiche di funzionamento del PCF8583.



Struttura e assegnazione dei bit del registro di controllo e di stato.



Mappa dei registri specifici.



Registro contatore delle ore.

LA MAPPA DEI REGISTRI

Nella figura alla pagina precedente è rappresentata la distribuzione e l'assegnazione delle funzioni dei differenti registri specifici per il modo orario e per il modo contatore di eventi.

Nel modo orario si può scegliere il formato 12 o 24 ore, attivando il bit più significativo del registro contatore di ore, che è riportato nella figura in alto.

Il registro che occupa la posizione 05 contiene l'anno e il giorno, il registro 06 il giorno della settimana e il mese. Queste figure mostrano la distribuzione interna di entrambi i registri. Attivando il bit di maschera del registro di controllo e di stato, si legge direttamente, tramite questi indirizzi, il giorno e il mese, mentre il giorno della settimana e l'anno si leggono a 0. Il formato BCD è utilizzato anche dal contatore di eventi esterni.



Registro del giorno e dell'anno.



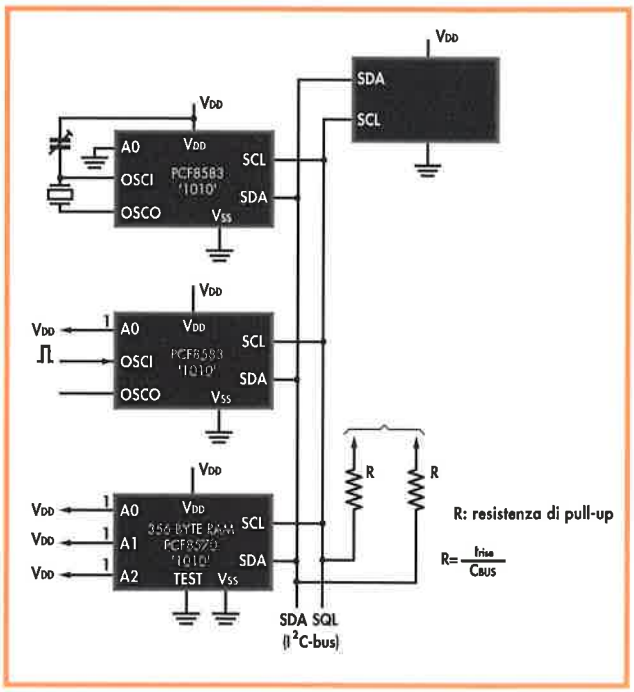
Registro del mese e del giorno della settimana.

SCHEMA DI UNA APPLICAZIONE PRATICA

Nello schema della figura sono rappresentati i collegamenti dei due PCF8583 e un PCF8570, che contiene 256 byte di memoria RAM. Tutti questi dispositivi sono per il bus I2C. Questi tre hanno assegnati gli stessi indirizzi (1010), per cui bisogna differenziarli mediante le linee, o pin di indirizzamento. Così il primo PCF8583 avrà A0 = 0 e il secondo A0 = 1. Il PCF8570 ha le sue tre linee di indirizzo A0 = A1 = A2 = 1. Il primo dei PCF8583 lavora in modo orario con il clock al quarzo a 32.768 Hz, mentre il secondo lavora come contatore di eventi applicato al piedino OSCI.

ESEMPI DI APPLICAZIONE CON IL CLOCK IN TEMPO REALE

Come esempio di gestione per il Micro'PIC Trainer Plus, il dischetto allegato al kit contiene un file con il programma CRONO.ASM, che implementa tutte le funzioni di un cronometro classico, in base al dispositivo PCF8583.



Schema del collegamento dei vari dispositivi al bus I2C.

Applicazioni della Micro'PIC IO (I)

ESPANSIONE DELLA MICRO'PIC TRAINER

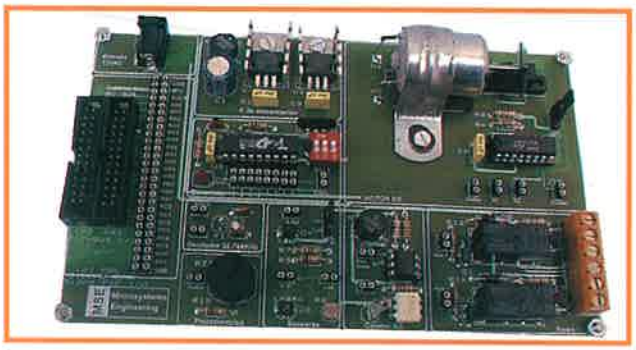
Il grande successo a livello internazionale del sistema di sviluppo Micro'PIC Trainer, ha indotto il suo costruttore a potenziarne l'efficacia, mediante l'ampliamento con nuove e poderose periferiche implementate su tre schede che ricevono il nome di:

1. Micro'PIC Trainer Plus
2. Micro'PIC IO
3. MSx84

In precedenza sono state analizzate in profondità le possibilità di applicazione della Micro'PIC Trainer Plus orientata all'uso intensivo del bus I2C. Qui parleremo della Micro'PIC IO, che espande le risorse base del sistema di sviluppo come indica la seguente tabella:

- Controllo della velocità di motori DC
- Encoder ottico per il controllo del senso di rotazione e della velocità dei motori
- Generatore di funzioni multiplo
- Controllo di potenza per angolo di accensione dei triacs
- Controllo di illuminazione
- Oscillatore
- Sensori analogici di luce e temperatura
- Micro-relè Omron da 5 VDC
- Cicalino piezoelettrico
- Alimentatore stabilizzato

La Micro'PIC IO è adatta a risolvere le applicazioni industriali inerenti al controllo dei motori, controllo di potenza, gestione dei segnali analogici, generazione di treni d'onda, attivazione di relè, ecc. Una informazione più completa su questa scheda e gli elementi ausiliari che l'accompagnano, come il Manuale dell'Utente, il dischetto con il software di appoggio, gli zoccoli di

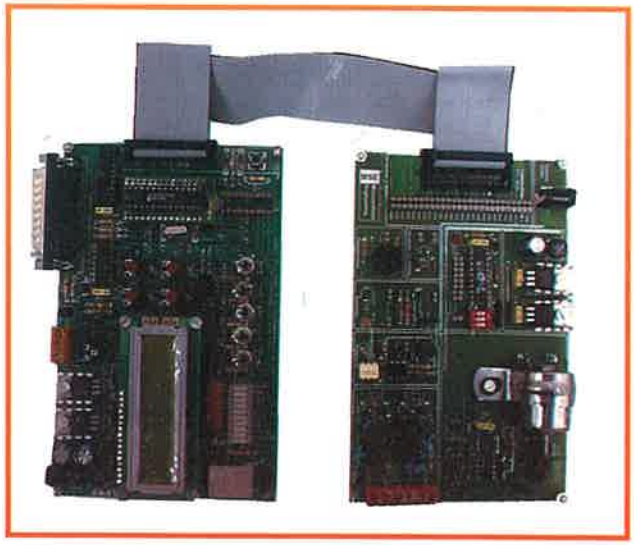


Fotografia della Micro'PIC IO

adattamento a nuovi PIC, ecc., si possono trovare sulle pagine di Internet del costruttore "Ingeniería de Microsistemas Programados S.L." che sono ubicate al sito www.microcontroladores.com. Nella figura si può vedere la distribuzione di tutte le risorse della Micro'PIC IO.

LA CONNESSIONE TRAMITE IL PIC-BUS

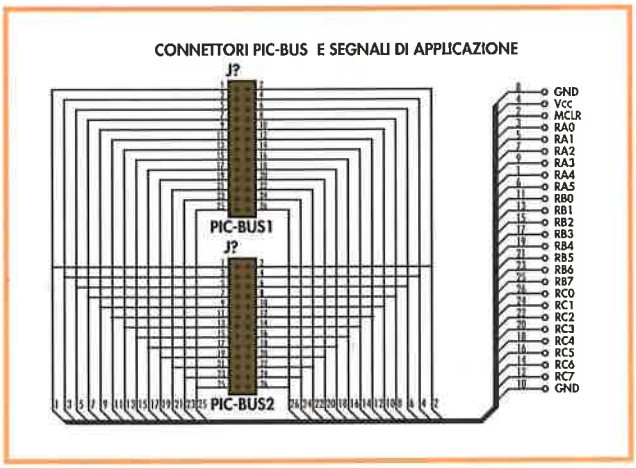
Per espandere la Micro'PIC Trainer, è stato progettato un connettore normalizzato chiamato PIC-BUS, che



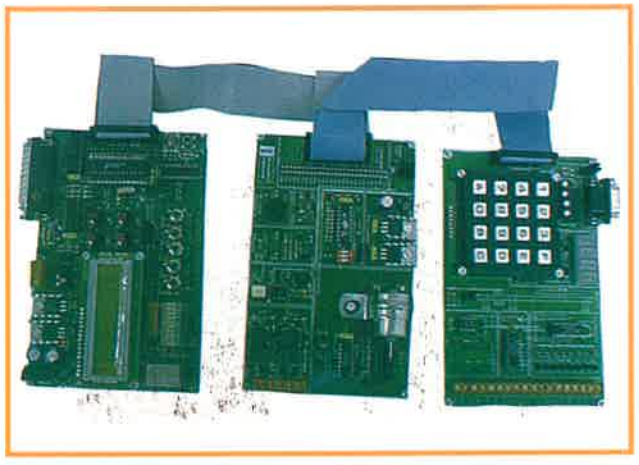
Accoppiamento mediante cavo piatto fra i connettori PIC-BUS del sistema di sviluppo Micro'PIC Trainer e la Micro'PIC IO.

nella Micro'PIC IO è formato da due connettori identici a 26 pin, che si possono riconoscere a sinistra nella fotografia. Utilizzando il cavo piatto accluso al sistema, si collega questa scheda alla Micro'PIC Trainer, che supporta il microcontroller PIC in cui è stato scritto il programma di applicazione, per permettergli di utilizzare le nuove risorse della scheda Micro'PIC IO durante la fase di messa a punto. La fotografia mostra l'accoppiamento delle due schede.

Nello schema elettrico del connettore PIC-BUS e delle linee di connessione esterna, si può vedere che CN2 (PIC-BUS 1) e CN3 (PIC-BUS 2) sono in parallelo, quindi possiamo usare uno qualsiasi dei due, cioè possiamo collegare simultaneamente le tre schede: Micro'PIC Trainer, Micro'PIC Trainer Plus e Micro'PIC IO.



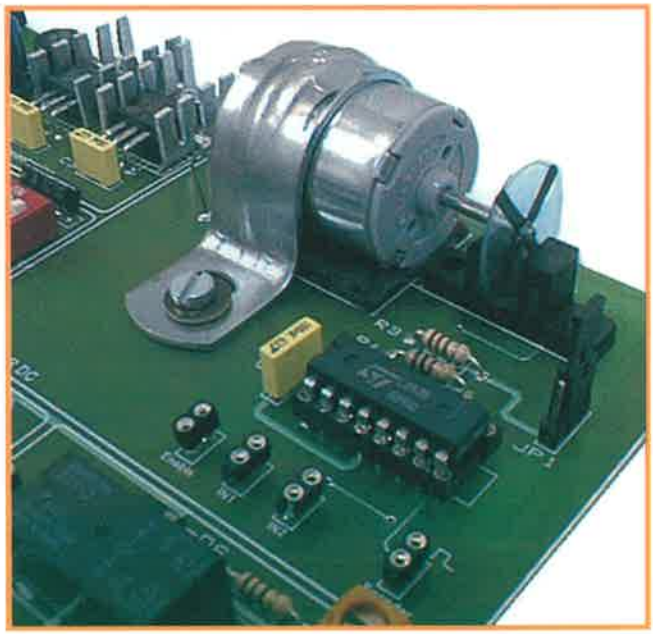
Schema elettrico del collegamento dei connettori che supportano i segnali del PIC-BUS.



Collegamento tramite i connettori PIC-BUS della Micro'PIC Trainer, la Micro'PIC Trainer Plus e la Micro'PIC IO.

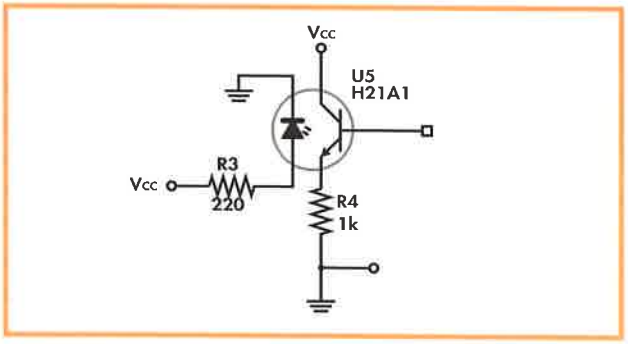
CONTROLLO DEL MOTORE A CORRENTE CONTINUA

Il controllo del motore a corrente continua è una delle applicazioni più rilevanti dell'area industriale, per questo è stato incorporato un motore DC nella Micro'PIC IO, insieme a un circuito integrato che lo controlla. Si vuole abituare l'utente a gestire il controllo della velocità e il senso di rotazione di questi motori, con la tecnica della modulazione di ampiezza degli impulsi (PWM). Sotto l'asse del motore c'è un fotoaccoppiatore a forcella, che permette di misurare la velocità angolare dell'asse del motore quando questo è accoppiato a un encoder. Come possiamo osservare nella fotografia, per realizzare un semplice encoder, basta ritagliare un disco di materiale rigido trasparente, tipo plastica, e perforare il suo centro per poterlo accoppiare all'asse del motore. Sulla superficie circolare del disco vengono dipinte diverse strisce radiali nere, o opache ai raggi infrarossi. Il numero



Posizione dell'encoder del motore a corrente continua.

delle strisce determina la risoluzione dell'encoder. In questo modo con una sola striscia, si produrrà un impulso ogni giro dell'asse, mentre se ci sono quattro raggi, per ogni giro dell'asse del motore si genereranno quattro impulsi. Nello schema della seguente figura è riportato il collegamento del fotoaccoppiatore, che rileva i passaggi dei raggi opachi dell'encoder. Il fotoaccoppiatore a forcella è il modello H21A1, e il diodo emettitore è permanentemente attivo, per cui il transistor ricevito-



Schema del collegamento del fotoaccoppiatore per l'encoder del motore.

re è saturo. Ogni volta che una striscia taglia il raggio di luce si ottiene un impulso che si applica ad una linea di ingresso del PIC per il suo successivo trattamento.

IL DRIVER L293B

Per controllare il motore DC si utilizza un circuito di eccitazione, o driver tipo push-pull, a quattro canali integrati sullo stesso chip L293B di SGS-THOMSON, le cui caratteristiche più importanti sono indicate qui sotto.

- Corrente di uscita da 1 A per canale
- Corrente di uscita di picco da 2 A per canale
- Segnale per l'abilitazione delle uscite
- Alta immunità al rumore
- Alimentazione per il carico i lata dall'alimentazione di controllo
- Protezione con o le sovratemperature

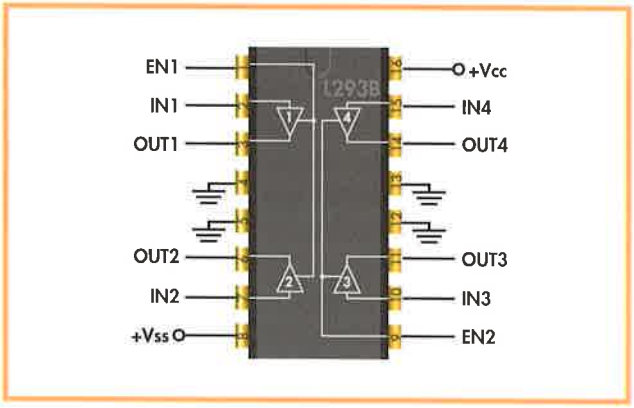
Nella figura è riporta la piedinatura del driver L293B insieme alla sua nomenclatura e descrizione.



PIN	NOME	DESCRIZIONE
1	EN	Abilitazione dei canali 1 e 2
2	IN1	Ingresso del canale 1
3	OUT1	Uscita del canale 1
4	GND	Massa dell'alimentazione
5	GND	Massa dell'alimentazione
6	OUT2	Uscita del canale 2
7	IN2	Ingresso del canale 2
8	VDD	Alimentazione del carico
9	EN2	Abilitazione dei canali 3 e 4
10	IN3	Ingresso del canale 3
11	OUT3	Uscita del canale 3
12	GND	Massa dell'alimentazione
13	GND	Massa dell'alimentazione
14	OUT4	Uscita del canale 4
15	IN4	Ingresso del canale 4
16	Vcc	Alimentazione a + 5VDC

Piedinatura dell'L293B e descrizione dei suoi piedini.

Nello schema a blocchi riportato in figura sono rappresentati i quattro canali capaci di fornire una corrente di uscita di 1 A. Ogni canale è controllato da un segnale di ingresso TTL, e ogni coppia di canali dispone di un segnale di abilitazione che scollega le uscite del medesimo. Il segnale di controllo EN1, attiva la coppia di canali formata dai driver 1 e 2. EN2 attiva i driver 3 e 4.

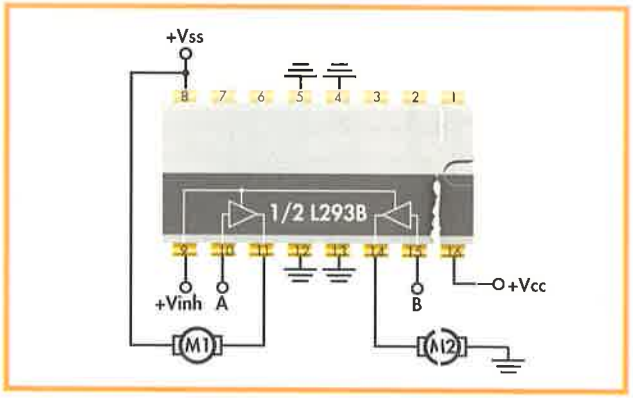


Schema a blocchi dell'L293B.

Le uscite OUTn sono associate ai corrispondenti ingressi INn. i segnali di uscita sono amplificati rispetto a quelli di ingresso sia in tensione (fino a + Vss) che in corrente (fino a 1A).

CIRCUITI PER IL CONTROLLO DEI MOTORI

Nello schema della figura è riportato il controllo dei due motori DC che girano in un verso unico. M1 si attiva con livello 0 applicato all'ingresso, mentre M2 lo fa applicando un livello 1 all'ingresso B, come si può verificare nella tabella della verità. Un altro schema mostra



Controllo dei due motori DC che girano in un unico verso.

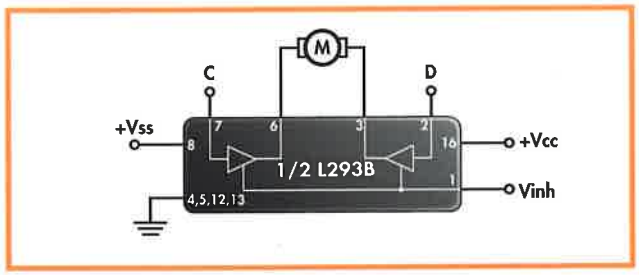
Vinh	A	M1	B	M2
H	H	Arresto rapido del motore	H	Rotazione
H	L	Rotazione	L	Arresto rapido del motore
L	X	Motore scollegato, rotazione libera	X	Motore scollegato, rotazione libera

L=Livello "0"; H=Livello "1"; X=Irrilevante

Tabella della verità per il controllo dei due motori DC che girano in un unico verso.

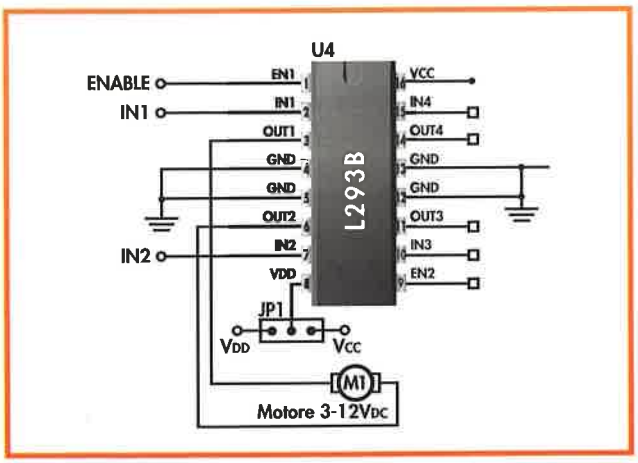
il controllo di entrambi i versi di rotazione del motore, mediante due canali dell' L293B. Se l'ingresso C è a 0 e quello D è a 1, il motore riceve una certa polarità che lo fa girare in un verso. Cambiando i livelli logici degli ingressi C e D, si produce la rotazione in senso contrario. Se entrambi gli ingressi sono allo stesso livello logico, si produce un arresto rapido del motore.

Nello schema della figura successiva è riportato il collegamento del motore DC alla scheda Micro'PIC IO.



Controllo di un motore DC nei due versi di rotazione.

I segnali IN1 e IN2, provenienti da qualsiasi linea dell'uscita del PIC, ricevono i livelli logici che faranno girare il motore in uno dei due versi secondo la tabella della



Collegamento del driver L293B e del motore DC della Micro'PIC IO.

verità. Quando il segnale ENABLE riceve un livello 0 da qualsiasi uscita del PIC, il motore si ferma, indipendentemente dallo stato di IN1 e IN2. È proprio attraverso questo ingresso che si può applicare un segnale con modulazione di ampiezza di impulsi (PWM), che permette di regolare la velocità di rotazione del motore in un verso o nell'altro. Un segnale PWM è un'onda quadra, in cui il tempo di permanenza del livello a 1 si può variare. Tanto maggiore sarà il tempo del livello a 1, più tempo resterà attivato il motore che girerà più velocemente.

IN1	IN2	ROTAZIONE
0	0	Stop
0	1	Orario
1	0	Antiorario
1	1	Stop

Tabella della verità del motore DC della Micro'PIC IO.

IL SOFTWARE

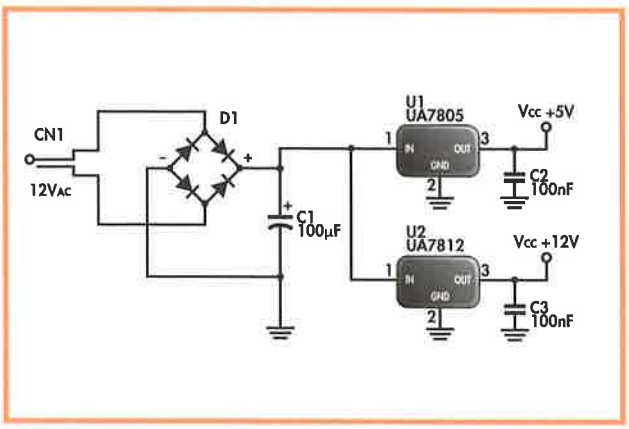
Il kit della Micro'PIC IO è accompagnato da un completo e chiaro manuale dell'utilizzatore, così come da un dischetto che contiene un insieme di progetti da realizzare con la suddetta scheda. In ognuno di questi sono esposti con abbondanti commenti gli obiettivi, i dispositivi di materiale necessario e il listato sorgente del programma del compito. Per maggiori informazioni si raccomanda di accedere alle pagine di Internet di "Ingeniería de Microsistemas Programados S.L." nel sito www.microcondroladores.com. Tutti gli esempi sono preparati per essere eseguiti su un PIC16F873, della gamma media, che studieremo a fondo più avanti, perché dispone dello stesso repertorio di istruzioni del PIC16F84. La cosa più interessante sta nel fatto che sono state aggiunte risorse molto importanti come il convertitore AD e DA, PWM, comparatori, catturatori di segnale, ecc.

Applicazioni della Micro'PIC IO (II)

DISPOSITIVI PER LE APPLICAZIONI INDUSTRIALI

Nel fascicolo precedente abbiamo iniziato lo studio della scheda di espansione Micro'PIC IO e la sua sezione di controllo del motore a corrente continua. Questa scheda dispone anche di altri interessantissimi dispositivi, che permettono di emulare in tempo reale i programmi di applicazione comune nell'industria, eseguiti sul PIC che supporta il sistema di sviluppo Micro'PIC Trainer che è collegato con la Micro'PIC IO tramite il connettore PIC-BUS.

La Micro'PIC IO dispone di un alimentatore proprio, quindi è sconsigliabile alimentarla tramite l'ingresso di tensione +5 VDC che arriva attraverso la linea del connettore PIC-BUS tramite il sistema di sviluppo. Questo ci permetterebbe di fare a meno del trasformatore, però il motore DC non avrebbe l'alimentazione di 12 VDC. Inoltre non potrebbero funzionare nemmeno il circuito di rivelazione del passaggio per lo zero, e l'optotriac per la regolazione alternata. Questi circuiti necessitano di una alimentazione indipendente di corrente alternata. Il ponte raddrizzatore D1 dell'alimentatore, insieme a C1, configura il circuito di raddrizzamento e filtraggio. La tensione così ottenuta si applica agli stabilizzatori 7805 (U1) e 7812 (U2). Il primo fornisce una tensione stabilizzata di +5 VDC per la circuiteria elettronica generale, mentre il secondo genera 12 VDC che si utilizzano principalmente nell'alimentazione del motore DC.

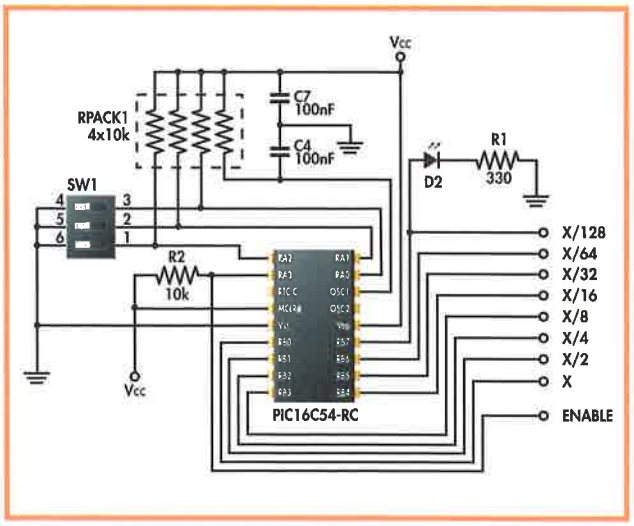


Schema dell'alimentatore della Micro'PIC IO.

IL GENERATORE DI FUNZIONI

Produce un segnale a onda quadra, la cui frequenza principale si può selezionare fra otto possibili valori, e in seguito, si possono ottenere fino a otto sottomultipli del valore scelto. Lo schema elettrico della figura mostra che il circuito è basato su un PIC16C54RC (U3). Il firmware del microcontroller è stato progettato da "Ingeniería de Microsistemas Programados S.L.". Il segnale ENABLE abilita o disabilita la generazione di onde quadre. Se questo ingresso è a livello 1 o non collegato, fornisce ai suoi ingressi le frequenze corrispondenti. Quando questo ingresso è a livello 0, il generatore resta scollegato e tutte le uscite si pongono a livello 0.

Mediante i tre microswitch di SW1 si seleziona, fra otto, il valore della frequenza campione che si ottiene dall'uscita X. Le uscite X/2, X/4, X/8, X/16, X/32, X/64, X/128 sono dei sottomultipli della frequenza scelta. Il diodo D2 sull'uscita X/128 visualizza il funzionamento



Schema elettrico del generatore di funzione.

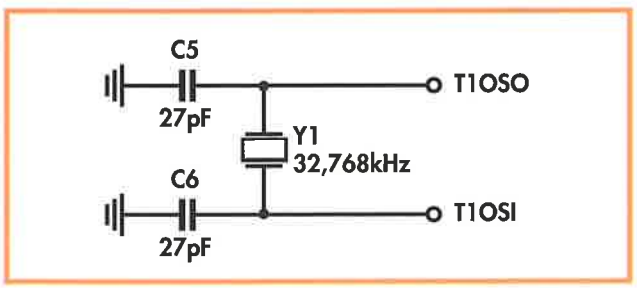
del generatore. Lampeggia quando lavora a frequenze più basse. Per il resto delle frequenze sembra rimanere acceso in modo permanente. Nella tabella della figura qui sotto sono indicate le distinte frequenze ottenute all'uscita, in funzione dello stato dei microinterruttori di SW1. La frequenza massima è attorno ai 200 KHz e la minima è di 1 KHz.

SW1			USCITE							
SW1-1	SW1-2	SW1-3	X	X/2	X/4	X/8	X/16	X/32	X/64	X/128
0	0	0	20kHz	10kHz	5kHz	2,5kHz	1,2kHz	625Hz	312Hz	156Hz
0	0	1	10kHz	5kHz	2,5kHz	1,2kHz	625Hz	312Hz	156Hz	78Hz
0	1	0	5kHz	2,5kHz	1,2kHz	625Hz	312Hz	156Hz	78Hz	39Hz
0	1	1	2,5kHz	1,2kHz	625Hz	312Hz	156Hz	78Hz	39Hz	19,5Hz
1	0	0	1,2kHz	625Hz	312Hz	156Hz	78Hz	39Hz	19,5Hz	9,75Hz
1	0	1	625Hz	312Hz	156Hz	78Hz	39Hz	19,5Hz	9,75Hz	4,87Hz
1	1	0	312Hz	156Hz	78Hz	39Hz	19,5Hz	9,75Hz	4,87Hz	2,43Hz
1	1	1	156Hz	78Hz	39Hz	19,5Hz	9,75Hz	4,87Hz	2,43Hz	1,21Hz

Tabella con le frequenze fornite dal generatore di funzioni a seconda della posizione dei microinterruttori di SW1.

OSCILLATORE A 32,768 KHZ

La funzione principale di questo circuito è fornire impulsi di clock ai temporizzatori o timer di cui dispongono i microcontroller, che realizzano misure di tempo. Così, ad esempio, il TMR1 della famiglia PIC16F87x può utilizzare come base dei tempi il segnale fornito da un oscillatore esterno. In questo modo l'overflow del TMR1 si può utilizzare per risvegliare un PIC che si trova in stato di riposo. Come mostra la figura è stato scelto un cristallo di quarzo da 32,768 KHz, poiché è un valore molto speciale. In concreto si ottiene un tempo pari ad un secondo, quando si conta un valore degli impulsi che è uguale alla potenza 15 in base 2 ed è la base per tutte le applicazioni orarie.

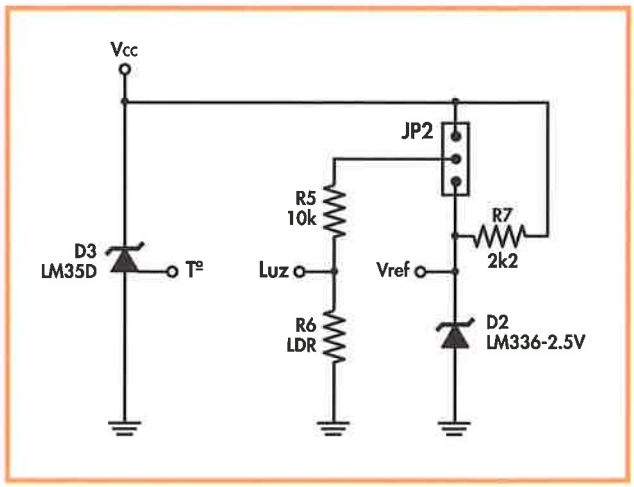


Schema elettrico dell'oscillatore da 32,768 KHz.

I SENSORI ANALOGICI E LA TENSIONE DI RIFERIMENTO

La Micro'PIC IO possiede due sensori analogici, uno misura la luminosità e l'altro la temperatura. Entrambi sono alimentati da una tensione di riferimento di 2,5 VDC, come mostra lo schema della figura. Come generatore della tensione di riferimento, è stato utilizzato LM336Z2.5 (D4) di National Semiconductor. È basato fondamentalmente su di un diodo zener da 2,5 VDC. Tramite la resistenza di assorbimento R7, si alimenta da +5VDC, ottenendo ai suoi capi 2,5 VDC generando quindi la tensione di riferimento Vref, che si può applicare, se si desidera, alla linea RA3/AN3/VREF+ del PIC16F87x in modo che sia utilizzata nella conversione AD.

Il sensore di luminosità è formato da una cellula LDR, che ha in serie la resistenza R5. La tensione che alimenta questo divisore può provenire da 5 o da 2,5 VDC, a seconda della posizione in cui si trova il jumper JP2. In questo modo sul terminale "Luce" dello schema si ottiene una tensione che varia fra 0 e 5 V, oppure, fra 0 e 2,5 V, a seconda della luce che incide sull'LDR. L'utilizzatore deve preoccuparsi che la tensione che alimenta questo



Schema elettrico del generatore di tensione di riferimento, che alimenta i sensori di luminosità e di temperatura.

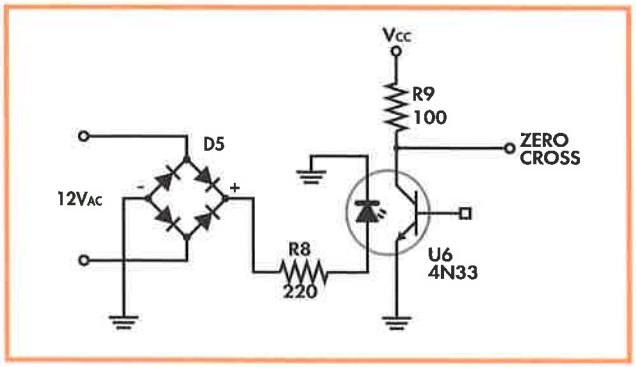
divisore coincida con la tensione di riferimento a cui lavora il convertitore ADC del PIC. In nessun caso la tensione di riferimento può essere minore della tensione analogica di ingresso da misurare. Il sensore di temperatura è il modello LM35D (D3), e su uno dei suoi terminali si ottiene una tensione " Tª" che è proporzionale alla temperatura esistente. Fornisce una tensione di 10 mV/°C e il margine di temperatura a cui può funzionare è compreso fra -55° e 150°C.

Nella configurazione impiegata nella scheda Micro'PIC IO il range di temperature è compreso tra 2 e 150°C. Quando si alimenta il sensore con 5 VDC per la temperatura massima permessa di 150°C, il sensore fornisce 1,5 V sul terminale "Tª".

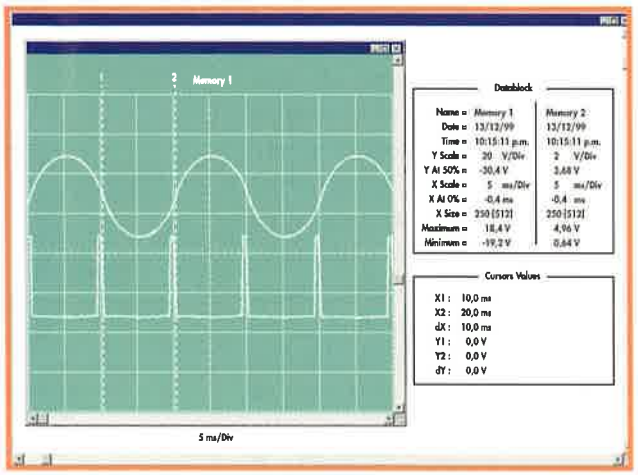
CONTROLLO E REGOLAZIONE DELLA POTENZA DELLA CORRENTE ALTERNATA

Una grande quantità di applicazioni industriali sono derivate dal controllo e dalla regolazione della potenza alternata che si applica ad un carico. Nella Micro'PIC IO sono utilizzati per questo compito dei circuiti elementari ed è necessario conoscere l'istante in cui la corrente alternata passa per lo zero e disporre di un segnale pilota che si attivi quando avviene questo passaggio. Il circuito base è mostrato nella figura.

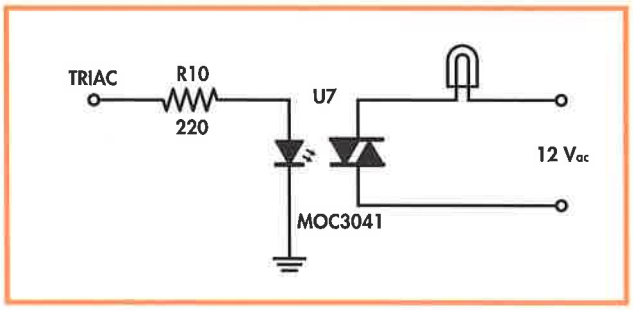
La tensione alternata, che arriva da un secondario di un trasformatore del valore di 12 VAC, è applicata al ponte raddrizzatore D5. Sul terminale D5 di questo ponte si ottiene un impulso doppio di 100 Hz, pari ad una frequenza di rete di 50 Hz: Questo segnale si applica al diodo emettitore del fotoaccoppiatore 4N33 (U6). Ogni volta che il segnale vale 0, il transistor ricevitore di questo fotoaccoppiatore resta bloccato, e la tensione del collettore sale sino a +Vcc. Quando l'impulso di ingresso ha un valore diverso da zero, il transistor rimane saturo, e la tensione sul collettore è 0V. Tramite il connettore "ZEROCROSS" si ottiene un impulso positivo ogni volta che il segnale alternato arriva a valori intorno allo 0 V, come è rappresentato nell'oscillogramma della figura. Questo impulso si può utilizzare per provocare un interrupt nel PIC. Nella figura si offre un semplice schema per la regolazione della potenza alternata



Schema del circuito utilizzato per il controllo della corrente alternata.



Oscillogramma della rilevazione del passaggio dallo zero della tensione alternata.

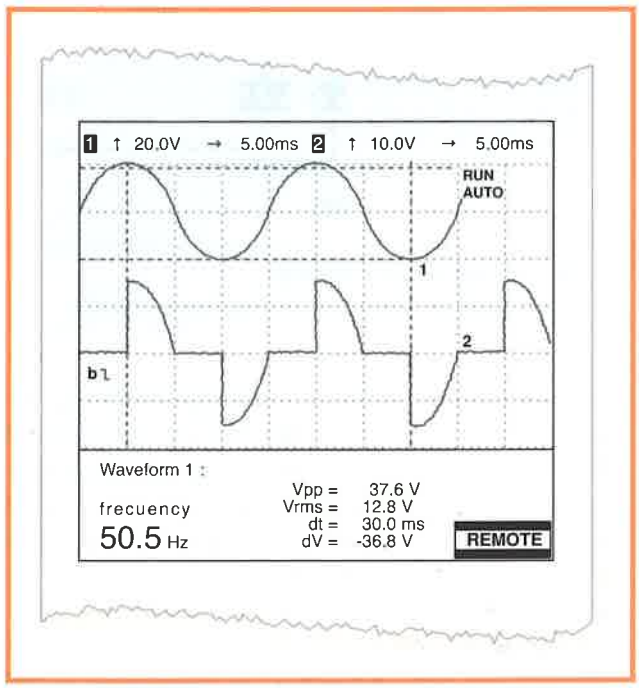


Regolazione della potenza alternata in un carico resistivo, che in questo caso è una lampada.

applicata a un carico resistivo, come può essere una lampada. Un livello 1, che arriva dal PIC sul terminale "TRIAC" dello schema, attiva il diodo emettitore dell'optotriac MOC3041 (U7) che provoca l'innesco del triac, facendolo condurre e applicando la tensione alternata al carico, che nel caso della scheda Micro'PIC IO si tratta di una lampada ad incandescenza (LP1).

La potenza dipende dal momento in cui si porta in conduzione il triac, applicando la tensione di innesco tramite l'ingresso "TRIAC". Questo istante prende come riferimento il momento di passaggio per lo zero, che si

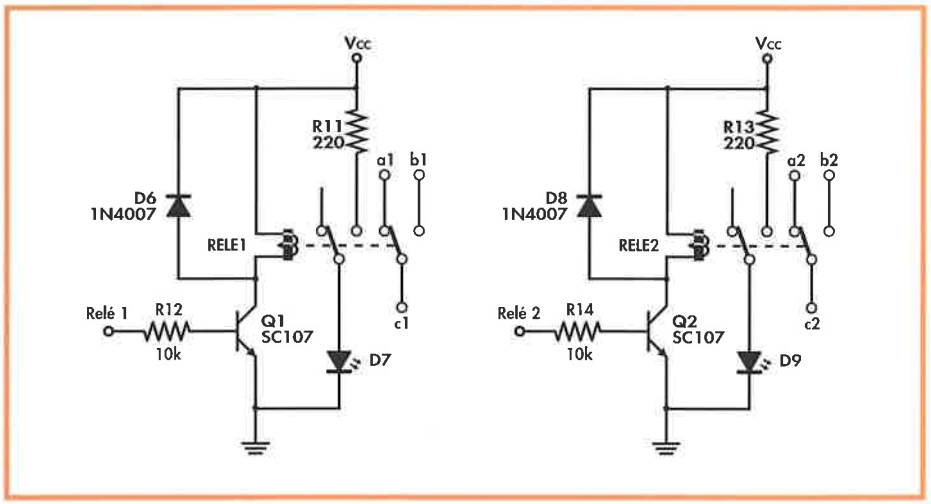
produce ogni 10 ms. Se l'innesco dell'optotriac si produce nel momento del passaggio per lo zero, si dice che l'angolo di innesco è 0°. La potenza che si applica al carico è la massima nei due semicicli. Se l'innesco del triac si produce a 5 ms del passaggio per lo zero, si applica la metà della potenza totale al carico. Se l'innesco si produce a 10 ms dal passaggio dallo zero, abbiamo un ritardo di 180° nell'innesco e il carico non riceve potenza. In altre parole controllando il ritardo con cui si produce il segnale di innesco, prendendo come punto di riferimento il segnale di passaggio per lo zero, si controlla la potenza totale che si applica al carico: il ritardo può variare tra 0 e 10 ms, che corrisponde a un angolo di ritardo da 0° a 180°, rispettivamente. Nella figura è riportato un oscillogramma che illustra la tensione alternata di ingresso e la tensione reale applicata al carico. Si applica al carico solo la metà della potenza di ogni semiciclo. L'angolo di ritardo è di 90°, che corrisponde a un ritardo di 5 ms.



Esempio di innesco del triac con ritardo di 90°, che corrisponde a un ritardo di 5 ms.

I RELÈ

Sono periferiche di uscita molto facili da controllare tramite qualsiasi linea di uscita del microcontroller. Basta un semplice circuito di eccitazione per pilotare la possibile attivazione delle bobine incaricate di aprire o chiudere i contatti, come si vede dalla figura. I relè impiegati nella Micro'PIC IO sono due, le loro bobine vengono alimentate con +5 VDC e i loro contatti sono doppi e a due posizioni (DPDT). Un livello logico 1, applicato ai connettori "Relè1/Relè2", in arrivo da qualsiasi linea di uscita del PIC, fa sì che si attivi il corrispondente relè. Il livello logico 0 produce un effetto contrario. Questo livello si applica alla base dei transistor Q1/Q2, mediante le resistenze di polarizzazione R12/R14. Questi transistor lavorano in commutazione. Quando stanno conducendo, la loro tensione sul collettore Vce è praticamente zero. I diodi D6/D7, collegati ai capi delle bobine, dissipano le correnti inverse che si originano ogni volta che viene tolta tensione. Ogni relè ha un doppio contatto e ogni contatto è a due posizioni. Uno dei due contatti si utilizza per attivare i LED pilota D7/D9, che monitorizzano lo stato di ogni relè. Il LED illuminato indica che è stato attivato il relè a cui fa riferimento. L'altro contatto del relè è a disposizione dell'applicazione dell'utente tramite due morsettiere a tre contatti ognuna. Un relè a riposo mantiene chiuso il contatto Cx-Ax, mentre Cx-Bx rimane aperto. Eccitando la bobina del relè si cambia lo stato di questi contatti. Ora Cx-Ax rimane aperto e Cx-Bx chiuso.

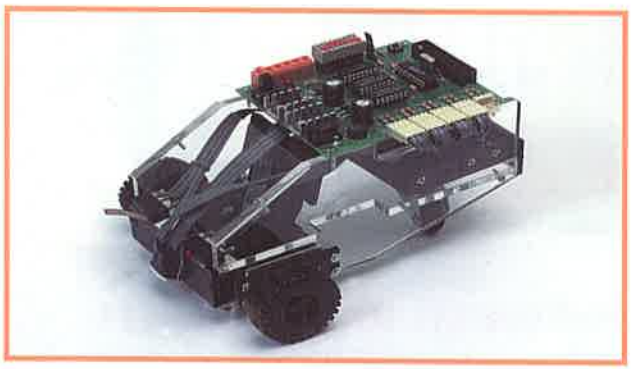


Circuito di eccitazione dei relè.

Applicazioni con la scheda di controllo MSx84

UNA SCHEDA AUTONOMA

In precedenza sono stati descritti i sistemi Micro'PIC Trainer Plus e Micro'PIC IO, composti da schede che espandevano le risorse e le possibilità del sistema di sviluppo Micro'PIC Trainer. Con la scheda di controllo MSx84 il suo costruttore "Ingeniería de Microsistemas Programados S.L." ha perseguito un altro obiettivo: offrire all'utilizzatore una scheda autonoma e indipendente che in base ad un PIC16F84 sia capace di pilotare lo stato di due motori DC o di uno passo a passo (PAP) e fino a cinque sensori digitali di ingresso. Queste caratteristiche sono tipiche in molte applicazioni industriali e la MSx84 è indicata per quattro tipi di impiego, che si adattano bene alle sue caratteristiche e che sono elencati nel riquadro.



Fotografia del microrobot PICBOT-2 basato sulla MSx84.

1° Manipolazione dei sensori di ingresso che permettono la sperimentazione con differenti tecniche per la rilevazione di fronti, eliminazione dell'effetto rimbalzo, misura della dimensione di un impulso, misura della frequenza, cadenza fra impulsi ecc.

2° Controllo di motori DC con azionamento digitale tutto o niente, tecniche di PWM, controllo del verso di rotazione, retroazione, posizionamento, ecc.

3° Controllo di motori PAP.

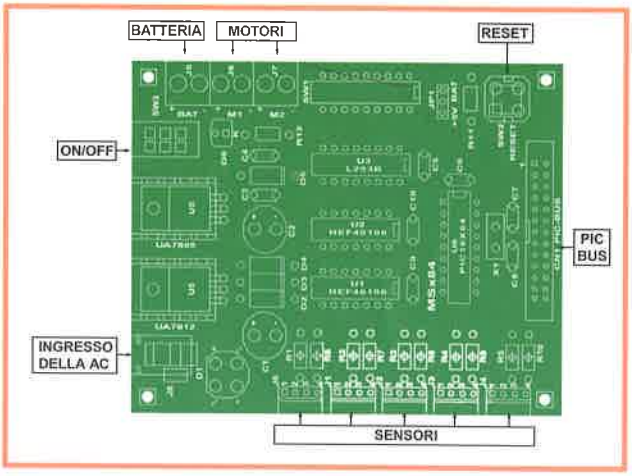
4° Microrobotica. La MSx84 si adatta perfettamente alle necessità di un microrobot. Può controllare direttamente due motori DC e sino a 5 sensori che raccolgano informazioni dall'ambiente. In base a questa scheda il costruttore ha progettato il microrobot commerciale PICBOT-2 che consiste in una struttura di metacrilato che supporta la MSx84, due motori DC e vari sensori.

CARATTERISTICHE GENERALI

Nel riquadro allegato possiamo notare le specifiche più importanti della MSx84.

CARATTERISTICHE TECNICHE DELLA MSx84

- Scheda da 80 x 100 mm a basso costo
- Alimentazione tramite trasformatore da 15 VAC, o mediante batterie da 6, 9 o 12 VDC, potendo utilizzare batterie di piombo da 12 VDC ricaricabili
- Circuito proprio di raddrizzamento, filtraggio e stabilizzazione
- Circuito di ricarica per batterie al piombo da 12 VDC
- Circuiti di controllo basati sul PIC16F84
- Connettori per l'applicazione di 5 sensori di ingresso
- Driver per l'azionamento di due motori DC o uno PAP
- Connettore di espansione Pic-Bus che permette di espandere le sue risorse, e di collegarla direttamente al sistema di sviluppo Micro'PIC Trainer, per realizzare la messa a punto 'in-circuito', così come la scrittura del PIC.

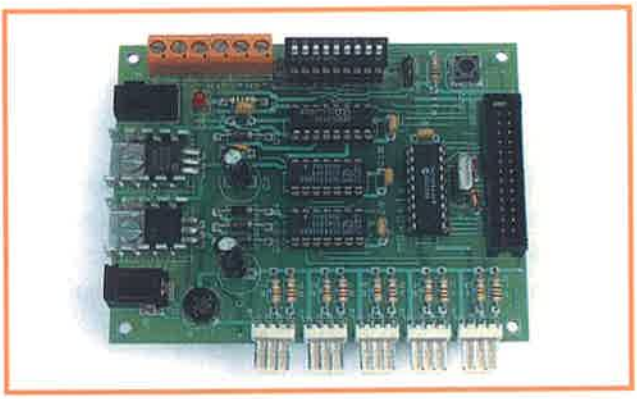


Serigrafia del circuito stampato della MSx84.

MICROCONTROLLER

Controllo dei motori e sensori con la MSx84

Nella figura della pagina precedente è riportata la serigrafia del circuito stampato, dove si vede la distribuzione di tutti i suoi componenti.



Fotografia della scheda di controllo MSx84.

Qui sotto sono descritti i quattro blocchi di cui è composta la MSx84.

1°. Alimentatore

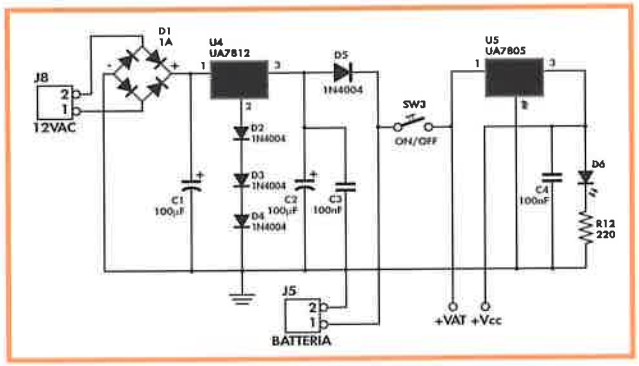
2°. Sezione del microcontroller

3°. Sezione dei sensori di ingresso

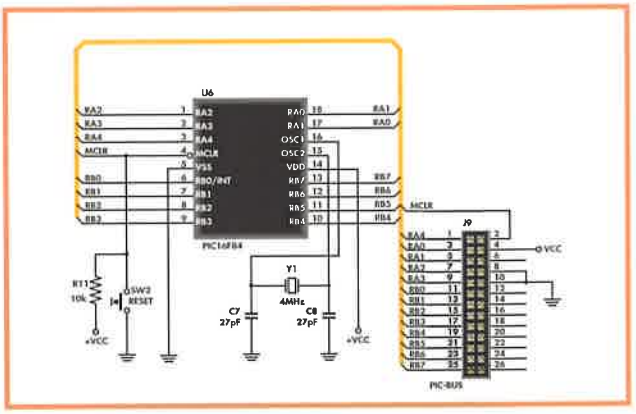
4°. Sezione del controllo dei motori

ALIMENTATORE

La MSx84 è totalmente autonoma, perché dispone di un proprio alimentatore. Nella figura riportiamo lo schema elettrico del medesimo. Tutta l'elettronica della scheda funziona con +5 VDC, che si ottiene a partire da una tensione alternata di 12 VAC, o da un insieme di batterie. Mediante il connettore J8 vengono forniti 12 VAC derivanti dal secondario di un trasformatore esterno. Il ponte rettificatore, composto da D1 e C1, ha il compito di rettificarla e filtrarla per ottenere una tensione di 12 VDC, che si stabilizza a 13 VDC grazie al regolatore 7812 (U4) più la tensione dei diodi D2-D5. SW3 applica questa tensione al controllore dei motori (+VBAT) e anche al regolatore 7805 (U5), quindi sul piedino 3 si ottengono +5 VDC, che sono applicati al circuito elettronico. Il diodo D6 monitorizza questa tensione. L'alimentatore funziona anche applicando una tensione di corrente continua derivata da una batteria tramite il connettore J5. Chiudendo SW3 si applica questa tensione a +VBAT e al regolatore 7805, per ottenere +5 VDC.



Schema dell'alimentatore.



Schema della sezione del microcontroller PIC16F84.

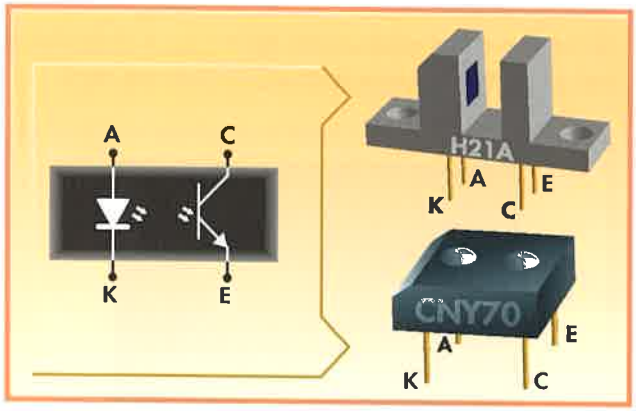
SEZIONE DEL MICROCONTROLLER

Nella figura è riportato lo schema del collegamento del PIC16F84, cuore e cervello della MSx84. Il quarzo Y1 insieme ai condensatori C7 e C8, genera la frequenza di funzionamento di 4 Mhz. Il circuito di Reset è formato dal pulsante SW2 e dalla resistenza R11. Il connettore Pic-bus è compatibile 'pin-to-pin' con quello del sistema di sviluppo Micro'PIC Trainer. Tramite esso tutti i segnali del PIC sono a disposizione dell'utente. Questo permetterà di collegare e/o aggiungere altri tipi di periferiche tra quelle supportate dalla scheda MSx84, cosl come programmare il PIC della MSx84 senza estrarlo dallo zoccolo.

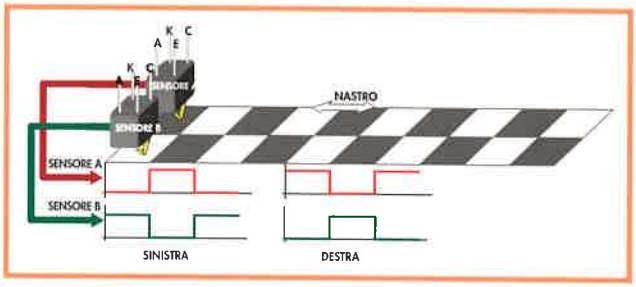
SEZIONE DEI SENSORI DI INGRESSO

I sensori sono i sensi del sistema per captare lo stato dell'ambiente. Nella sezione "Sensori e Attuatori" ne sono stati descritti molti, e in microrobotica quelli ottici spiccano per la loro precisione, semplicità, efficacia, ridotte dimensioni e prezzo.

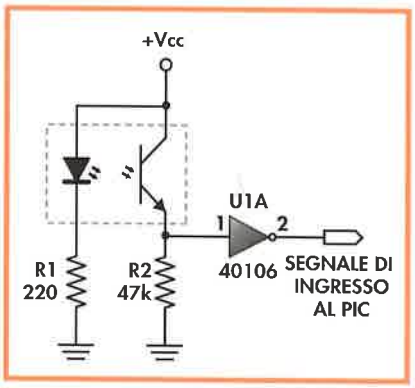
Il modello CNY70 e il H21A sono i più popolari. Il primo contiene l'emettitore e il ricevitore ad infrarossi, nella stessa capsula, ed è ideale per rilevare cambi di tono sulla stessa superficie. La luce emessa dal LED riflessa dall'oggetto viene rilevata da un fototransistor. Il contenitore a forma di U del H21A ha l'emettitore separato di tre millimetri rispetto al ricevitore, però entrambi sono alla stessa altezza, in modo che il raggio di luce emessa colpisca il fototransistor. È l'ideale per rilevare oggetti che, frapponendosi tra l'emettitore e il ricevitore, taglino il raggio di luce, come si vede nella figura. Un'applicazione tipica del sensore H21A è quella di implementare un circuito di misura della velocità angolare di rotazione dell'asse del motore, in base a un encoder formato da un disco trasparente con raggi radiali opachi, come presentato nella figura. Conoscendo il numero di raggi si determina il numero di giri che ha dato l'asse. Nella figura si presenta una possibile applicazione del sensore CNY70, che consiste in un nastro trasportatore con il fondo formato da riquadri neri e bianchi. Quando la luce dell'emettitore coincide con un riquadro nero sarà assorbita e, quindi, il fototransistor non riceverà luce. Se coincide con un riquadro bianco si rifletterà sul ricevitore, cambiando il livello logico della sua uscita. Nella figura successiva sono utilizzati due sensori CNY70 per conoscere non solo la velocità del nastro trasportatore, ma anche il verso del movimento. Lo schema della figura mostra i collegamenti tipici del sensore ad infrarossi con i pin di ingresso del PIC.



Contenitore e disposizione degli elementi dei sensori CNY70 e H21A.

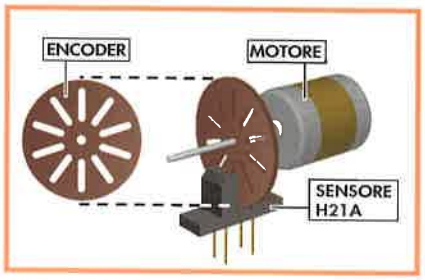


Controllo di velocità e verso del movimento con due sensori CNY70.

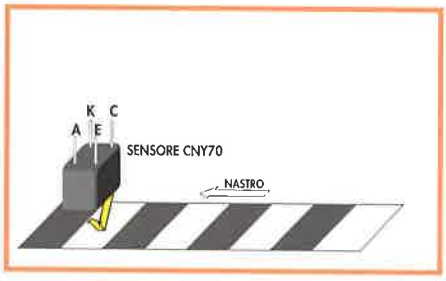


Schema di collegamento di un sensore ottico ai pin di ingresso del PIC.

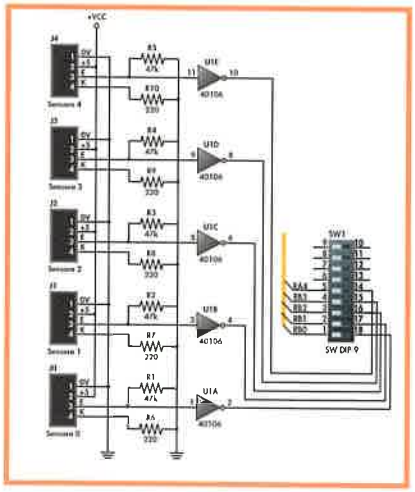
Per poter gestire sino a cinque sensori digitali di ingresso, la MSx84 dispone di un circuito il cui schema è mostrato in figura e nel quale possiamo vedere l'associazione fra i connettori dei sensori di ingres-



Esempio di applicazione del sensore ottico H21A.



Controllo di velocità di un nastro trasportatore con un sensore CNY70.



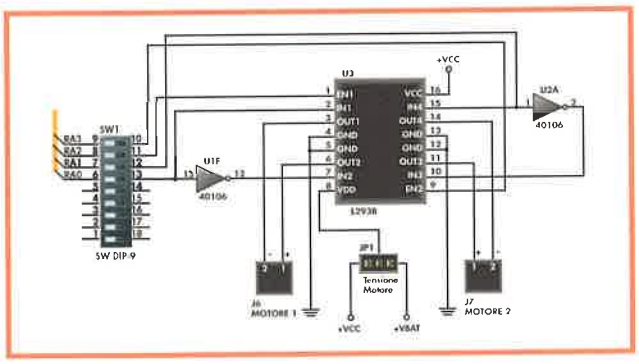
Schema della sezione degli ingressi digitali della MSx84.

so, i microinterruttori e le linee del PIC. Il sensore collegato a J4 resta associato alla linea RA4 del PIC. Questa linea funziona come ingresso degli impulsi T0CKI del temporizzatore TMR0, il quale quando funziona in modo contatore di eventi, può contare il numero di impulsi ricevuti da questo ingresso, permettendo misure di frequenza, dimensione di impulsi, intervalli fra impulsi, ecc.

SEZIONE DEL CONTROLLO DEI MOTORI

Uno dei principali impieghi a cui è destinata la MSx84 è quello del controllo di due motori DC o uno PAP. A questo scopo utilizza il driver L293B, il cui funzionamento e le applicazioni al controllo di motori DC è stato oggetto di studio del capitolo precedente. In questa sezione studieremo il controllo di un motore PAP, lo schema di questa sezione è riportato qui sotto.

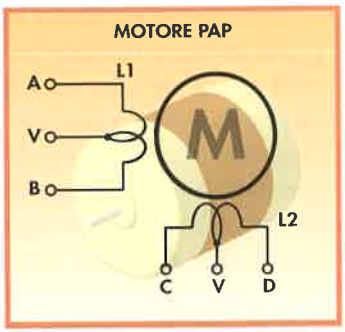
La linea RA0 del PIC tramite l'interruttore 6, si applica al segnale IN1 e, mediante l'inverter U1F, al segnale IN2 dell'L293B. Questo segnale pilota le uscite OUT1 e OUT2 che controllano il motore M1. La stessa spiegazione vale anche per la linea RA1 del PIC, che pilota OUT3 e OUT4 per il motore M2.



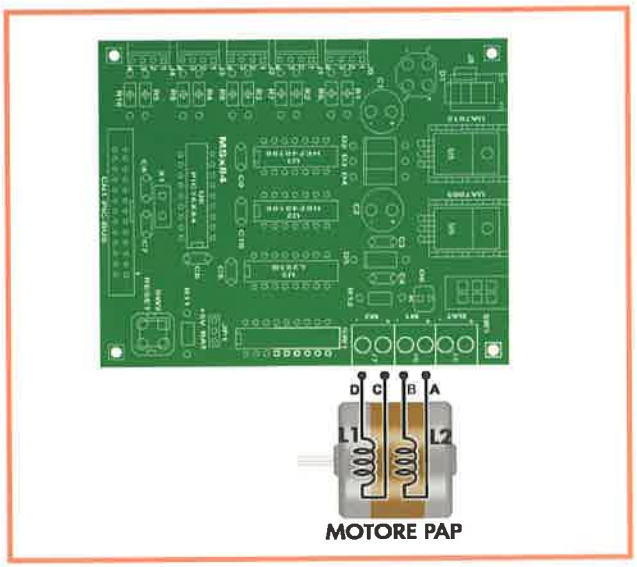
Schema della sezione di controllo dei motori della MSx84.

Per quanto riguarda il controllo dei motori PAP, essi possono avere 4 o 6 cavi, a seconda se sono bipolari o unipolari rispettivamente. I quattro cavi di quelli bipolari, corrispondono agli estremi delle bobine di cui sono composti. Nel caso di quelli unipolari, quattro cavi corrispondono agli estremi delle due bobine, e gli altri due alle prese intermedie di queste, come si può vedere nella figura.

La figura successiva offre lo schema di collegamento di un motore PAP a quattro cavi alla scheda MSx84. La bobina L2 si collega ai morsetti segnati come M2, e la bobina L1 a quelli M1. Tramite le linee RA3 e RA2 del PIC, si ottengono le combinazioni binarie da applicare al driver L293B, in modo che si attivino le uscite M1 e M2, in accordo con le polarità indicate dal costruttore, riportate nella tabella della figura. Per maggiori informazioni tecniche sulla MSx84, consigliamo di consultare le pagine che il costruttore ha messo a disposizione nel sito Internet www.microcontroladores.com.



Schema di collegamento di un motore PAP unipolare.



Schema di collegamento di un motore PAP a quattro fili con la MSx84.

SENSO ORARIO

RA3	RA2	RA1	RA0	PASO	M1+(A)	M1-(B)	M2+(C)	M2-(D)
1	1	1	1	1	-	+	-	+
1	1	1	0	2	+	-	-	+
1	1	0	0	3	+	-	+	-
1	1	1	0	4	-	+	+	-

SENSO ANTIORARIO

RA3	RA2	RA1	RA0	PASO	M1+(A)	M1-(B)	M2+(C)	M2-(D)
1	1	1	0	1	-	+	-	+
1	1	0	0	2	+	-	-	+
1	1	1	0	3	+	-	+	-
1	1	1	1	4	-	+	+	-

Tabella che indica i valori binari che devono assumere le linee della porta A del PIC, per ottenere il senso di rotazione adeguato.

Display a cristalli liquidi LCD

PRESENTAZIONE E APPLICAZIONI

I display a cristalli liquidi LCD, sono i terminali di visualizzazione più utilizzati negli strumenti basati su microcontroller; il loro compito è presentare informazioni, valori o qualsiasi altro tipo di dato. Uno dei primi prodotti in cui sono stati utilizzati gli LCD su larga scala sono state le calcolatrici digitali, come quella mostrata nella figura.



Le calcolatrici sono state il primo prodotto a impiegare i display LCD su larga scala.

Il display LCD consiste in un condensatore planare formato da due placche metalliche, una delle quali è talmente sottile da risultare trasparente. Fra queste placche si trova un dielettrico isolante, che è un cristallo liquido in fase nematica. Quando non si applica tensione fra le placche il cristallo rimane trasparente, ma è sufficiente applicare una piccola tensione, in modo che si generi un piccolo assorbimento di corrente, per produrre una variazione nella riflessione e nella rifrazione della luce, tale da rendere opaco il cristallo.

Una delle principali disfunzioni a cui sono soggetti i diodi LED, si riscontra quando c'è molta luce, perché questi si distinguono meno, specialmente se la luminosità dell'ambiente è grande. Invece nei display LCD succede l'opposto, utilizzano la luce ma non la generano, perciò maggiore è l'illuminazione ambientale, più elevato è il contrasto dell'informazione presentata.

Un tipico settore di applicazione dei display LCD è la strumentazione. La maggioranza degli strumenti di misura utilizza un LCD per visualizzare i dati. È il caso del multimetro, che si può considerare lo strumento più comune nei laboratori di elettronica, e di cui riportiamo una fotografia nella figura.



Il tester è attualmente uno degli strumenti che più utilizza visualizzatori LCD.

Nei multimetri digitali, è molto frequente l'utilizzo di display LCD a tre digit e mezzo, come quello che si vede in figura.



Struttura di un display LCD da tre digit e mezzo, utilizzato nei multimetri digitali.

MICROCONTROLLER

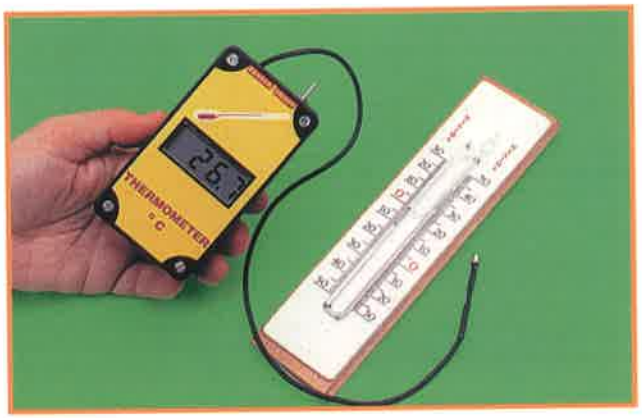
Caratteristiche dei display LCD

Oltre agli strumenti di misura e le calcolatrici, esistono molti dispositivi che utilizzano gli LCD, fra cui ricordiamo i telefoni fissi e mobili, le agende elettroniche, le macchine fotografiche, i distributori automatici, i pannelli di comando dei veicoli, ecc.

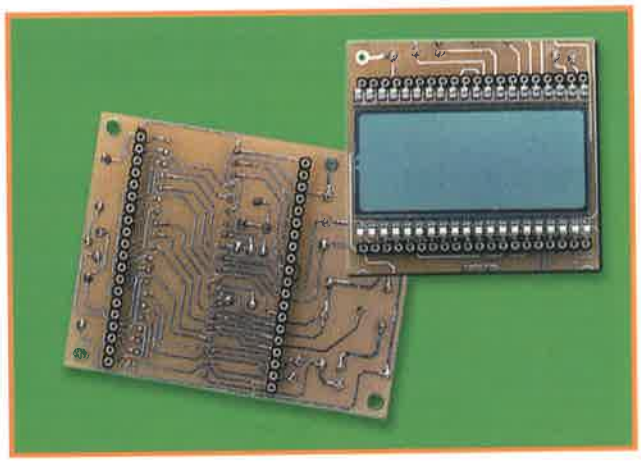
Come esempio di una semplice applicazione, presentiamo nella fotografia sottostante un termometro, che dispone di un sensore di temperatura esterno, collegato tramite un cavo di una certa lunghezza, che permette di misurare la temperatura all'interno di un frigorifero, fuori dalla casa, fuori dalla macchina, nell'acqua, ecc.

In questo termometro si utilizza un display a tre digit e mezzo, tipo SP525, che ha due file di piedini, e che ha bisogno di trovare sul circuito stampato due connettori da 20 pin cadauno, per il proprio sostegno, come si può vedere nella figura in basso.

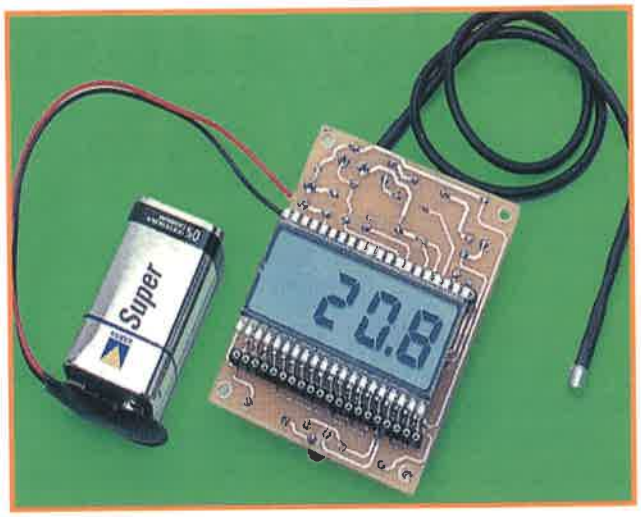
Infine il sistema di sviluppo Micro'PIC Trainer, utilizzato per supportare lo sviluppo completo dei progetti basati sui microcontroller PIC, utilizza come terminale principale un display LCD da due linee con 16 caratteri ognuna, come si vede nella fotografia della figura.



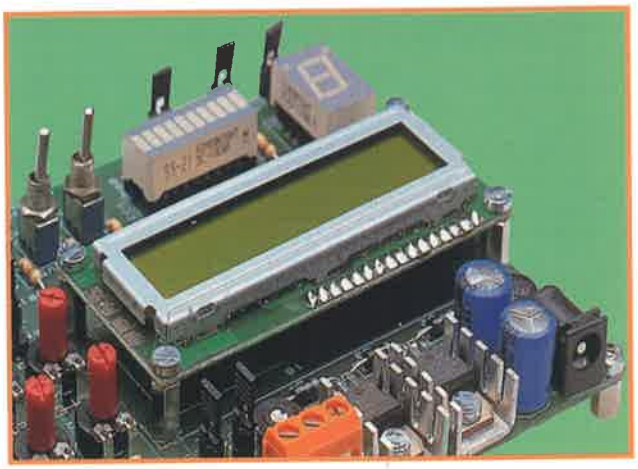
Termometro che supporta il display LCD.



Scheda del termometro che supporta il display LCD.



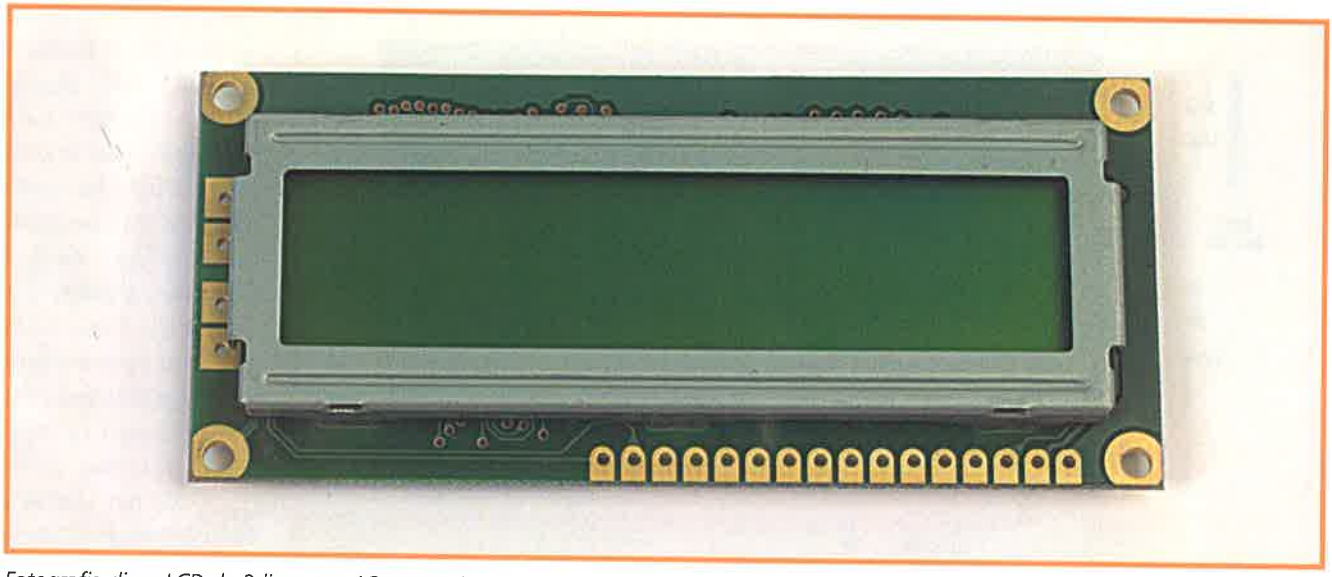
Aspetto dei collegamenti della scheda del termometro che sostiene l'LCD con il sensore di temperatura e la batteria.



Il sistema di sviluppo Micro'PIC Trainer utilizza un display LCD come principale terminale di visualizzazione.

DESCRIZIONE TECNICA

I display alfanumerici hanno la capacità di visualizzare diverse linee composte da un determinato numero di caratteri. Possiamo trovare LCD da 1 linea per 16 caratteri, 2 x 16, 4 x 16, 1 x 20, 2 x 20, 2 x 24, 4 x 20 ecc. Inoltre possono, o meno, avere l'opzione di una lampada di retro-illuminazione, cioè illuminazione da dietro, che migliora notevolmente il contrasto dei caratteri in ambienti poco illuminati.



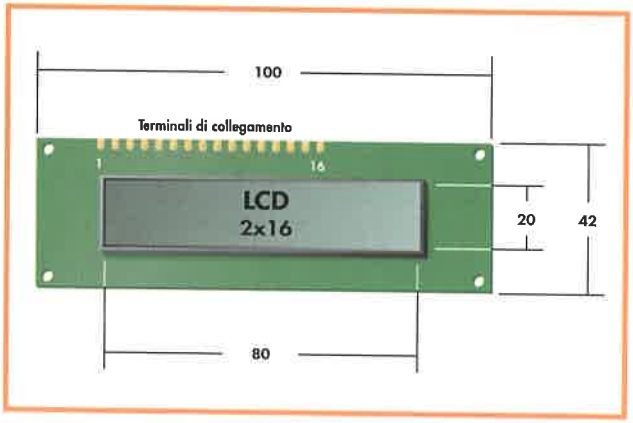
Fotografia di un LCD da 2 linee per 16 caratteri.

Per rendere l'idea, abbiamo scelto di descrivere un LCD tipico, modello C-1602, da 2 linee per 16 caratteri, la cui fotografia è riportata qui sopra.

Le misure dell'LCD selezionato sono approssimativamente 100 x 42 mm, e la dimensione del display di visualizzazione 80 x 20 mm.

I 16 terminali di collegamento del modulo LCD corrispondono ai segnali di controllo e di trasferimento dei dati, in accordo con l'assegnazione delle funzioni riportata nella tabella.

Nella figura della pagina successiva troviamo lo schema di connessione dei segnali esterni ai terminali dell'LCD. Fondamentalmente si dividono in segnali di alimentazione, di controllo, di dati e di retro-illuminazione.



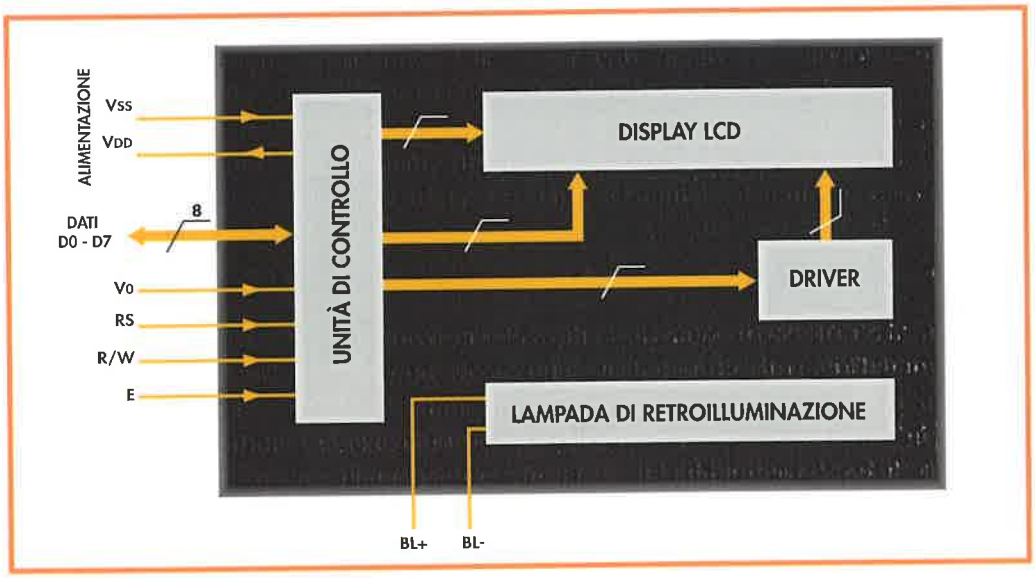
Dimensioni dell'LCD e del display di visualizzazione.

TERMINALI DI COLLEGAMENTO

NUMERO	SIMBOLO	FUNZIONE
1	Vss	Massa dell'alimentazione
2	Vdd	Positivo +5 VDC dell'alimentazione
3	Vo	Contrasto del cristallo liquido. Generalmente si collega a un potenziometro che applica una tensione da 0 a +5 V
4	RS	Seleziona il registro di controllo o dei dati RS = 0, registro di controllo; RS = 1, registro dei dati
5	R/W	Segnale di lettura o scrittura R/W = 0, scrittura; R/W = 1, lettura dell'LCD.
6	E	Segnale di abilitazione dell'LCD E = 0, LCD non attivo; E = 1, LCD attivo
7-14	D0-D7	Bus dei dati bidirezionale
15-16	BL+/BL-	Alimentazione + e – della lampada di retro-illuminazione

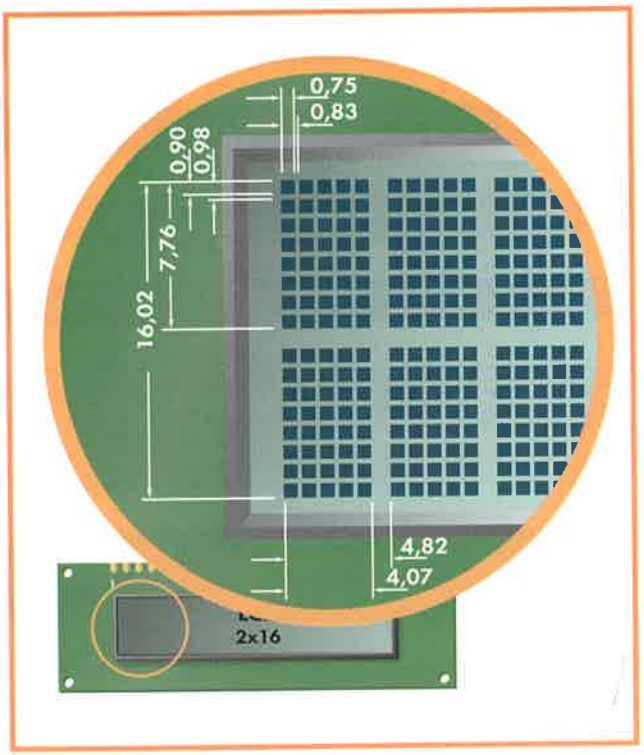
MICROCONTROLLER

Caratteristiche dei display LCD



Schema di interconnessione dei segnali di un modulo LCD.

L'insieme dei caratteri che si possono vedere nel display LCD è riportato nella figura. In essa le posizioni indicate come CGRAM (n) corrispondono a uno degli 8 possibili caratteri grafici da 5 x 7 punti che possono essere liberamente definiti dall'utente. Questi caratteri si definiscono introducendo nei successivi indirizzi della CGRAM dei byte con determinati parametri binari. La CGRAM è una memoria da 64 byte, di cui se ne utilizzano 8 per definire un carattere 5 x 7 e 16 se si tratta di un carattere 5 x 10.



Matrice da 5 x 7 punti di due caratteri sovrapposti, uno per ogni linea, con le dimensioni più importanti.

Ogni carattere si rappresenta sul display LCD mediante una matrice di punti o pixel da 5 x 7. Nella figura sono indicate le matrici dei due caratteri sovrapposti, ognuno dei quali corrisponde a una delle linee.

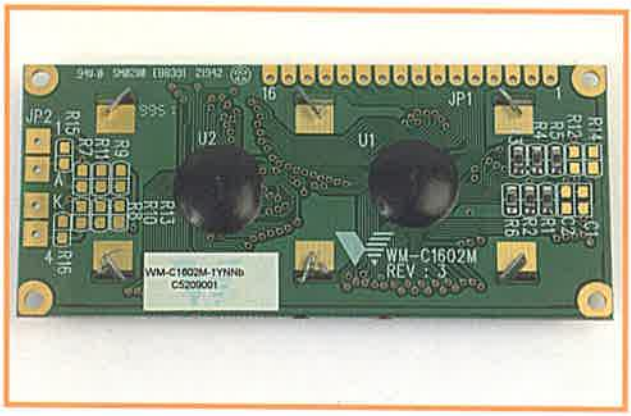


Insieme di caratteri ASCII del modulo LCD.

Funzionamento dei display LCD

IL CONTROLLO DELL'LCD

Come abbiamo già commentato in precedenza, il display LCD è un visualizzatore di caratteri, sia di testo che grafici, molto potente e flessibile, che si può pilotare facilmente tramite un microcontroller inserito all'interno del prodotto. Nella fotografia qui sotto si mostra la parte posteriore dell'LCD, in cui si può apprezzare la sua struttura elettronica.



Vista posteriore del display LCD pilotato da un sistema basato su un microcontroller integrato sulla scheda.

Nella descrizione dell'architettura del modulo LCD, le linee del collegamento esterno si dividono nei seguenti gruppi:

- Linee di alimentazione: Vss e Vdd
- Linee dei dati: BD0 – BD7
- Linee di controllo:
  E: Impulsi di convalida dell'operazione
  Vo: Tensione di controllo del contrasto
  RS: Selezione del controllo (0: Istruzione di ingresso e 1: Carattere di ingresso)
  R/W#: 0: Si scrive LCD; 1: Si legge
- Linee di retro-illuminazione: BL+ e BL- (Ingressi per l'alimentazione per la lampada di retro-illuminazione)

Nella figura in alto a destra si mostra il diagramma dei collegamenti dell'LCD.



Schema dei collegamenti del modulo LCD.

COLLEGAMENTO DISPLAY LCD-MICROCONTROLLER PIC

Per pilotare un display LCD tramite un PIC, si utilizzano 11 linee di I/O digitali del microcontroller.

Le otto linee della porta B (RB0-RB7) servono per supportare i trasferimenti bidirezionali di informazioni tramite il bus dei dati dell'LCD (DB0-DB7).

Le linee di controllo dell'LCD (RS, R/W# ed E) si pilotano con le linee RA0, RA1 e RA2 della porta A rispettivamente, come si può vedere nello schema seguente.

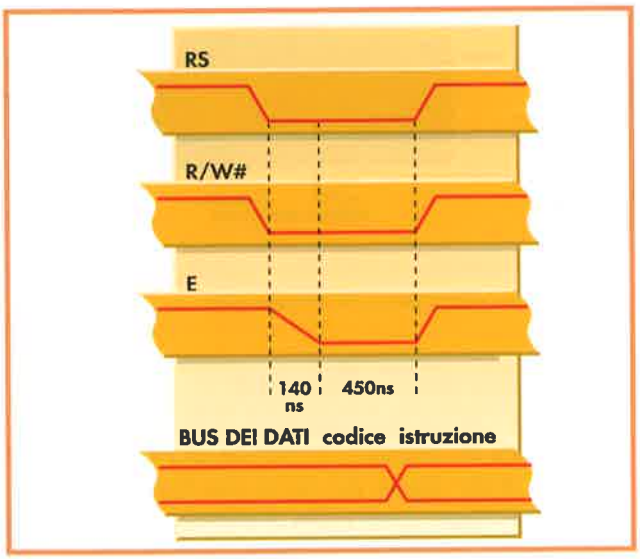


Collegamento delle porte A e B del PIC con le linee dei dati di controllo del display LCD.

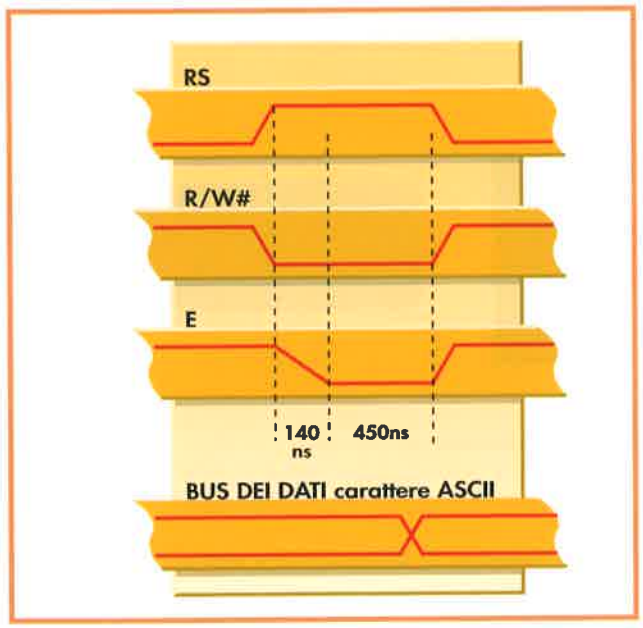
La tensione di alimentazione dell'LCD coincide con quella del PIC, e se si desidera variare il contrasto dei caratteri che sono visualizzati, si utilizza un potenziometro da 10 K con cui si applica una tensione variabile all'ingresso, V0, che regola il contrasto del display.

TRASFERIMENTO DI ISTRUZIONI E CARATTERI

Nella comunicazione del PIC con il display LCD si possono inviare comandi da eseguire o caratteri da visualizzare in qualsiasi delle 32 matrici esistenti nel display, che si compone di due linee da 16 caratteri ognuna. Quando il PIC invia, tramite il bus dei dati, una istruzione o un comando all'LCD, può realizzare operazioni molto interessanti, come cancellare tutto il display, muovere il cursore a una determinata posizione, indirizzare una delle matrici, ecc. Il PIC colloca il codice dell'istruzione nella porta B, e per indicare che si tratta di un comando che si deve scrivere nel display pone la linea RS (RA0) = 0 e la linea R/W# (RA1) = 0.



Cronogramma di un'operazione di scrittura di un'istruzione nell'LCD tramite il PIC.



Cronogramma di un'operazione di scrittura di un carattere ASCII nell'LCD.

Nel cronogramma riportato nella figura a sinistra si mostra il valore digitale che assumono i diversi segnali che partecipano all'operazione di scrittura di un'istruzione nell'LCD tramite il PIC.

La scrittura del codice di un carattere ASCII sopra una delle 32 matrici del video è abbastanza simile alla scrittura di una istruzione.

Varia solo il livello logico del segnale RS. In questa situazione la porta B del PIC prende il codice ASCII, allo stesso tempo in cui RA0, RA1 e RA2 prendono i valori che sono mostrati nel cronogramma in basso.

REPERTORIO DI ISTRUZIONI DELL'LCD

Il display LCD risponde a un insieme di comandi che interpreta ed esegue quando riceve tramite il bus dei dati il codice corrispondente e RS = R/W# = 0.

Tra i comandi più importanti troviamo quelli di cancellazione del video, i movimenti del cursore, i lampeggiamenti, e gli indirizzamenti delle memorie interne dell'LCD.

La memoria DDRAM è quella che contiene i codici dei caratteri che devono essere visualizzati nelle due linee.

Gli indirizzi da 80 a 8F Hex corrispondono ai 16 caratteri della prima linea e gli indirizzi da C0 a CF Hex a quelli della seconda linea.

La memoria CGRAM ha 64 indirizzi di dimensione byte e in essa sono contenuti i codici necessari per definire 4 o 8 caratteri grafici definibili dall'utente, secondo le necessità dell'applicazione. Ogni carattere grafico definito dall'utente è composto da 8 o 16 byte che si memorizzano negli indirizzi successivi della memoria CGRAM.

In seguito sono descritte le principali istruzioni utilizzate con il display LCD.

CANCELLAZIONE DEL VIDEO (CLEAR DISPLAY)

Cancella tutti i caratteri del video e colloca il cursore nella prima posizione della prima linea.

Inoltre pone il bit di controllo I/D = 1 per default, il che significa che con ogni carattere il cursore incrementa la sua posizione.

Impiega 1,64 microsecondi ad essere eseguito, sup-

ponendo che funzioni a 4 Mhz, e il suo codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

CURSORE A CASA (HOME)

Colloca il cursore nella posizione di inizio, prima posizione della prima linea, e forza il display a visualizzare da quel punto.

Il contenuto dei dati nella DDRAM non varia e l'indirizzo della visualizzazione si pone a zero.

Impiega 1,64 microsecondi ad essere eseguito e il suo codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	1	x

SELEZIONE DEL MODO (ENTRY MODE SET)

Stabilisce la direzione del movimento del cursore e specifica se la visualizzazione si sposta alla posizione seguente del display o no.

Questa operazione si esegue durante la lettura o la scrittura della DDRAM o la CGRAM.

Per visualizzare normalmente si pone il bit di controllo S = 0. Se S = 1, la visualizzazione si sposta ogni volta che si scrive un dato.

Impiega 40 microsecondi ad essere eseguito e il suo codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	1	I/D	S

Se I/D = 1 si incrementa il cursore, in caso contrario ciò non accade.

CONTROLLO ON/OFF DEL DISPLAY (DISPLAY ON/OFF CONTROL)

Attiva o disattiva il display (D) e il cursore (C), stabilendo se questo deve lampeggiare (B) o no.

Impiega 40 microsecondi ad essere eseguito e il suo codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

SPOSTAMENTO DEL CURSORE O DISPLAY(CURSOR OR DISPLAY SHIFT)

Muove il cursore e sposta il display senza cambiare il contenuto della memoria dei dati DDRAM.

Si esegue in 40 microsecondi. Se il bit S/C vale 1 sposta il display e se vale 0 muove il cursore. Il codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	S/C	R/L	X	X

SELEZIONE DI FUNZIONE (FUNCTION SET)

Stabilisce la dimensione della comunicazione con il bus dei dati, che può essere da 4 bit (DL = 0) o 8 bit (DL = 1).

Determina il numero di linee del display fra una (N = 0) o due (N = 1) e il tipo di carattere, se è 5x7 (F = 0) o 5 x 10 (F = 1).

Impiega 40 microsecondi ad essere eseguito e il suo codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	F	X	X

SELEZIONE INDIRIZZI CGRAM (SET THE CGRAM ADDRESS)

I caratteri grafici definibili dall'utente sono scritti in una memoria da 64 byte chiamata CGRAM. Con questa istruzione si stabilisce l'indirizzo della memoria CGRAM a partire dal quale si inizierà a memorizzare i byte che definiscono i caratteri grafici.

Eseguendo questo comando tutti i dati che si scrivono e si leggono successivamente, lo faranno da questo indirizzo della CGRAM. Impiega 40 microsecondi e il suo codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	INDIRIZZO DI CGRAM					

SELEZIONE INDIRIZZI DDRAM (SET THE RAM ADDRESS)

I caratteri che si visualizzano sono stati precedentemente scritti nella memoria DDRAM per poi passare al video. Questa istruzione stabilisce l'indirizzo della DDRAM da cui sono memorizzati i dati. L'indirizzo della prima linea corrisponde a 80 fino a 8F Hex e quello

della seconda da C0 a CF Hex. Si esegue in 40 microsecondi e il suo codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	INDIRIZZO DI CGRAM						

LEGGI IL FLAG OCCUPATO (READ BUSY FLAG)

Mentre il modulo LCD esegue istruzioni, è occupato e non si deve inviare un'altra istruzione. Il flag BUSY (BF) indica che è occupato quando vale 1, o che è libero quando vale 0.

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	CODICE ASCII O BYTE DEFINITO DALL'UTENTE							

SCRIVI DATO NELLA CGRAM O DDRAM

Con questo comando si scrivono nella DDRAM i caratteri da visualizzare e si può anche scrivere nella CGRAM i byte definiti dall'utente. Il suo codice è:

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	1	CODICE ASCII O BYTE DEFINITO DALL'UTENTE							

LEGGERE DATI DALLA CGRAM O DDRAM

Con questo comando si leggono i codici ASCII scritti nella DDRAM o i byte nella CGRAM.

RS	R/W#	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	1	BF	INDIRIZZO DI CGRAM						

SEQUENZA DI INIZIO

Il modulo LCD esegue automaticamente una sequenza interna di inizio al momento in cui viene applicata la tensione di alimentazione, se si compiono i seguenti requisiti:

a. Il tempo che impiega a stabilizzarsi la tensione fra 0,2 e 4,5 V è compreso fra 0,1 e 10 millisecondi.

b. Il tempo di scollegamento deve essere come minimo di 1 millisecondo prima di tornare a collegarsi.

La sequenza di inizio è la seguente:

1°. CANCELLAZIONE DEL DISPLAY
Il segnalatore BUSY si mantiene attivo a 1 durante i 15 ms della sequenza di inizio.

2°. SELEZIONE DEL MODO
Si sceglie per default la dimensione del bus dei dati a 8 bit e una linea nel display (N = 0)

3°. CONTROLLO ON/OFF DEL VIDEO
Per default si sceglie il display OFF (D=0), cursore OFF (C=0) e senza lampeggiamento del cursore (B=0).

4°. SELEZIONE DI FUNZIONI
Si sceglie per default l'incremento del cursore (I/D=1), modo normale e non spostamento del display (S=0).

5°. SELEZIONE DELLA PRIMA POSIZIONE DELLA DDRAM

Se non si compiono esattamente le condizioni di alimentazione, la sequenza di inizio dovrà essere realizzata dall'utente via software. È molto importante che la prima istruzione che si invia produca una temporizzazione di 15 ms per poter realizzare una sequenza completa.

ROUTINE DI CONTROLLO

Descriviamo un paio di routines in assembler MPASM che forniscono un'idea della semplicità di gestione dei display LCD:

LCD_E
Routine che genera un impulso di 1 microsecondo tramite la linea RA2 (segnale E). Si suppone sempre che il PIC stia lavorando ad una frequenza di 4 Mhz.

```
LCD_E   bsf     RA,2        ;Attiva E
        nop                 ;Attendi 1 microsecondo
        bcf     RA,2        ;Disattiva E
return
```

LCD_REG
Questa routine deposita il codice di istruzione presente nel registro W del PIC sulla porta B. Attende che il display LCD esegua l'ultima operazione e genera l'impulso di attivazione di E.

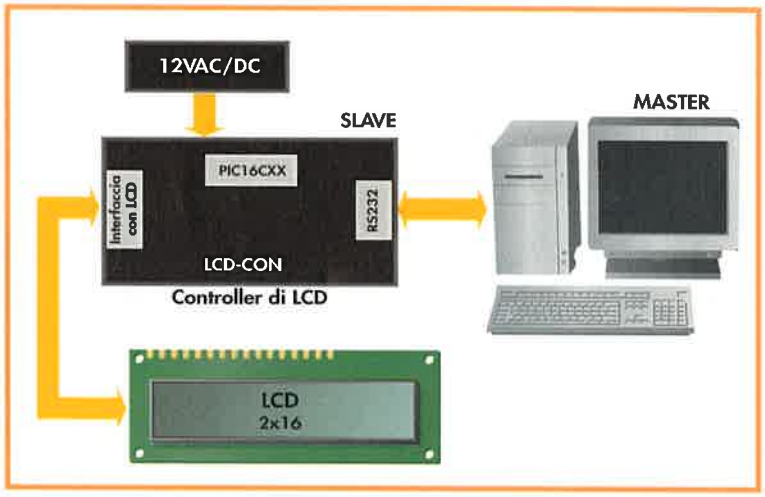
```
LCD_REG bcf     RA,0        ;Disattiva RS (Istruzione)
        ovwf    RB          ;Prende il codice in W tramite PB
        call    LCD_BUSY    ;Attende che si liberi LCD
        goto    LCD_E       ;Genera un impulso in E
```

Controller di LCD (I)

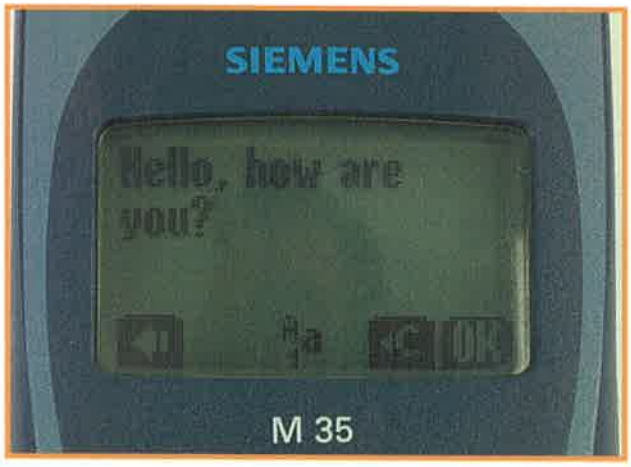
CONTROLLER DI LCD: CONCETTO E APPLICAZIONE

Il display LCD è la periferica più interessante di cui disponiamo per visualizzare informazioni nei prodotti e nei sistemi basati sui microcontroller. Il suo ridotto costo, la sua flessibilità nella presentazione di messaggi e di grafica, le sue piccole dimensioni e il fatto di essere programmabile, sono caratteristiche che lo rendono insostituibile.

Per sfruttare al meglio i display LCD, sono state sviluppate, e si trovano in commercio, schede che facilitano all'utente l'uso e la gestione all'utente per l'applicazione nei progetti. La scheda di controllo rende trasparente il controllo interno del display LCD. Tramite una coppia di linee, si collega la scheda di controllo con qualsiasi sistema capace di comunicare tramite una RS-232, come ad esempio un PC, un terminale di videotastiera, un microcontroller, ecc. Questi dispositivi funzionano come maestri (Master), rispetto alla scheda di controllo che funziona come schiava (Slave). Lo schema dei collegamenti dei blocchi che gestiscono il display LCD è riportato nella figura. La scheda di controllo dell'LCD è basata su un microcontroller che ha il compito principale di interpretare le istruzioni che riceve dal master via RS-232. La decodificazione di queste istruzioni si trasforma in segnali di controllo, che si applicano al display per produrre i differenti effetti di visualizzazione programmati attraverso il master.



Schema di adattamento della scheda controller fra il PC (Master) e il display LCD (Slave).



Il display LCD è diventato uno dei terminali più utilizzati per visualizzare informazioni.

CLASSIFICAZIONE DEI CONTROLLER DI LCD

In generale si possono classificare in due grandi gruppi:

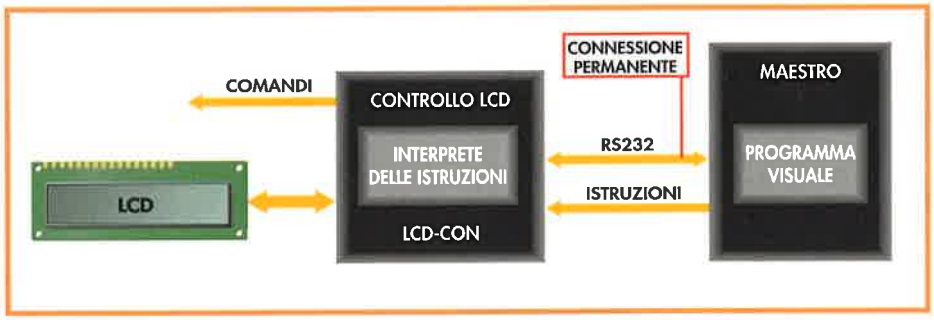
1 → Controller on-line.

2 → Controller programmabili con memoria.

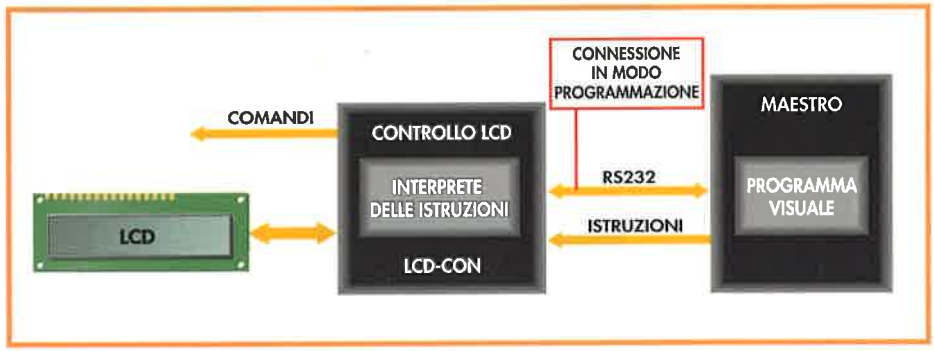
Il controller on-line si pilota in modo permanente tramite il master. Riceve ed esegue sequenzialmente le istruzioni del master, che può essere un PC o un sistema con microcontroller. Le istruzioni di visualizzazione sono inviate nel momento opportuno, in funzione degli avvenimenti del mondo esterno che influenzano il prodotto che incorpora il display LCD. Per esempio, in una macchina distributrice di prodotti, ogni volta che si introduce una moneta si mostra il valore della medesima e il montante totale del credito disponibile. Il controller on-line è progettato per mostrare messaggi nel momento in cui si producono gli eventi corrispondenti.

MICROCONTROLLER

Modelli di controller di LCD



Il master è permanentemente collegato al controller LCD, e contiene il programma di istruzione e di visualizzazione.



Il controller programmabile con memoria, riceve il programma visuale tramite il master in modo programmazione, lo memorizza nella sua EEPROM, e dopo lo esegue in modo autonomo.

In questo caso il programma di istruzione di visualizzazione è contenuto nella memoria del master, come si vede nella figura. L'altro tipo di controller, quelli programmabili con memoria, oltre a disporre delle stesse funzioni dei controller on-line, dispongono di una memoria EEPROM in cui sono scritte le istruzioni di visualizzazione. Queste istruzioni si eseguono al reinizializzarsi del sistema, ciò significa che questi modelli possono eseguire in modo indipendente le istruzioni e i messaggi che in precedenza sono stati scritti nella loro memoria. Questa caratteristica li rende molto adatti ai sistemi automatici, come succede nei cartelli pubblicitari.

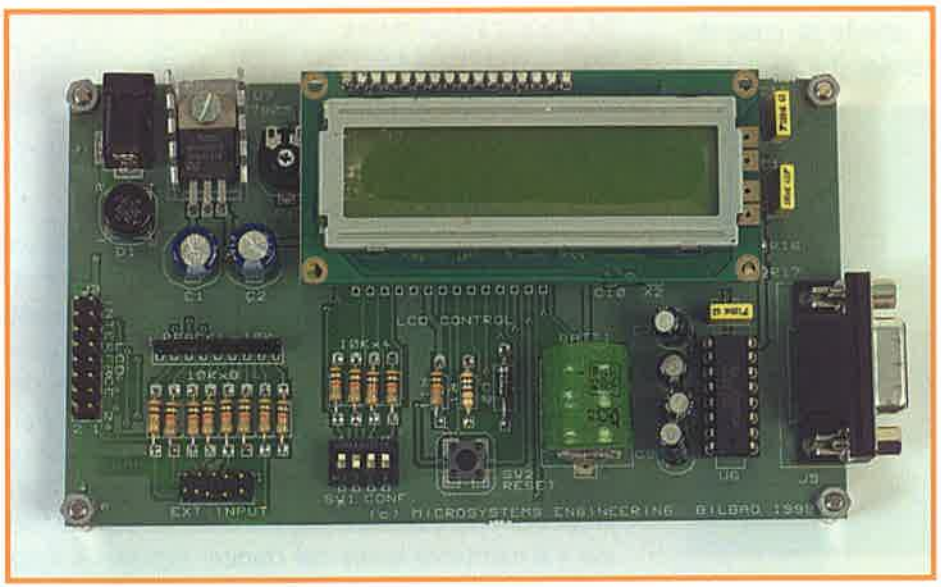
CARATTERISTICHE TECNICHE

Per avvicinare i lettori alla realtà commerciale, descriveremo il modello LCD-CON1, di tipo on-line, e l' LCD-CON2 di tipo programmabile con memoria che possiamo vedere alla pagina seguente.

STRUTTURA E CONFIGURAZIONE

La scheda si compone di dodici blocchi funzionali, elencati di seguito e mostrati nella figura:

1 → Connettore di alimentazione.
2 → Potenziometro di contrasto.
3 → Microcontroller PIC 16C73.
4 → Interfaccia LCD 1. Connettore a singola fila (SIL) per LCD.
5 → Clock/calendario in tempo reale.
6 → Memoria di programma.
7 → Connettore RS232.
8 → Batteria.
9 → Pulsante di Reset.
10 → Microinterruttore di configurazione.
11 → Segnale digitale di ingresso. Ext. Input (J4).
12 → Interfaccia LCD 2. Connettore a doppia fila (DIL) per LCD.



Scheda controller LCD-CON2, programmabile con memoria.

CARATTERISTICHE TECNICHE DEI CONTROLLER "LCD-CON"
Scheda di formato normalizzato da 100 x 75 mm.
Alimentazione 12 VAC/DC, con circuiti di rettificazione, filtraggio e stabilizzazione.
Comunicazione con il master in seriale a 19.200 baude, configurabile con microinterruttori.
Clock/calendario in tempo reale, alimentato con batteria ricaricabile di Ni-Cd.
Microcontroller PIC 16CXX di gamma media.
Potenziometro per la regolazione del contrasto del display.
Capacità di governare display fino a 80 caratteri con e senza retroilluminazione, con diversa configurazione di linee e carattere per linea.
Pulsante di reset.
Memoria EEPROM con 24 Kbyte nella versione LCD-CON2.
Connettore per otto segnali di ingresso che servono per rilevare condizioni di visualizzazione.
Quattro modi di lavoro configurabili con microinterruttori: test, on-line, programmazione ed esecuzione.
Software per Windows che configura il Sistema Integrato di Sviluppo che permette una interfaccia semplice da utilizzare per l'utente.

Cronogramma di un'operazione di scrittura di un'istruzione nell'LCD tramite il PIC.

Il connettore per l'alimentazione, indicato con il numero 1 nello schema, riceve una tensione di 12 V, che può essere alternata o continua. La scheda dispone di un circuito di raddrizzamento (D1), filtraggio (C1-C2), e stabilizzazione (U7). Il potenziometro di contrasto è evidenziato con il numero 2 nello schema.

IL MICROCONTROLLER E I CONNETTORI PER LCD

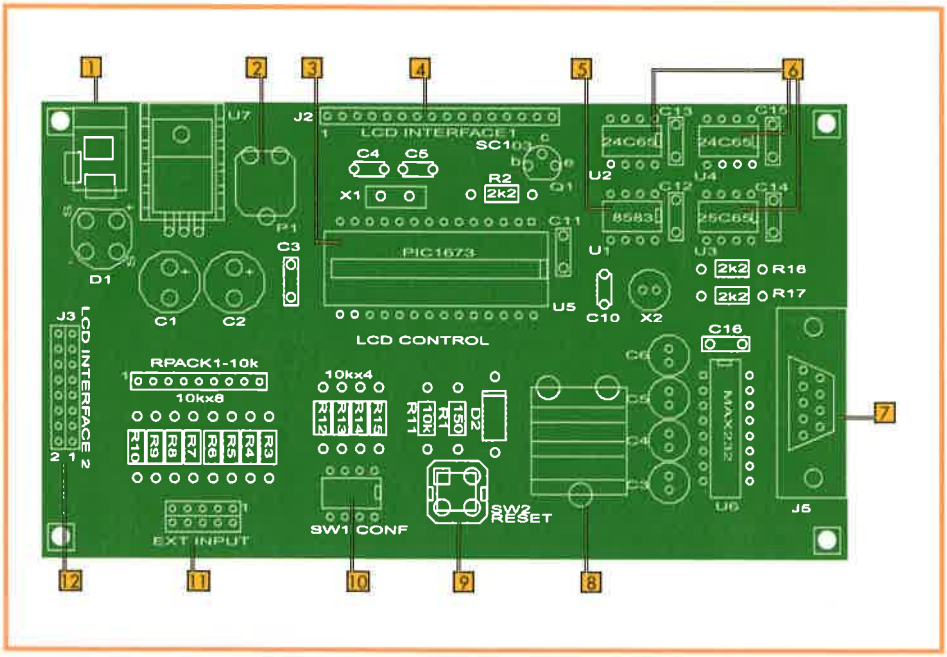
Il PIC16C73 è incaricato di realizzare i due compiti vitali del controller:

1°. Supporta la comunicazione RS232 con il master.

2°. Interpreta le istruzioni che invia il master per controllare il display LCD. Grazie al lavoro del PIC non è necessario che l'utente conosca il funzionamento interno dell'LCD.

Il connettore a fila singola da 16 pin (SIL), è segnato con il numero 4 nello schema, e serve per supportare i moduli LCD che hanno questo formato.

Nella figura sono assegnate le funzioni ai diversi piedini. Per alcuni tipi di moduli LCD, è necessario un connettore a doppia fila per la loro inserzione. Questo connettore è indicato con il numero 12 nello schema, e nella figura è riportato il suo schema che si adatta ai modelli delle case WINTEK, NORITAKE e SHARP.



Schema della scheda controller LCD-CON2 con la distribuzione dei suoi blocchi funzionali.

CLOCK/CALENDARIO

È basato sul dispositivo PCF8583 (U1) di Philips, che contiene un clock/calendario a tempo reale, alimentato tramite le batterie di Ni/Cd.

Il chip è controllato da un cristallo di quarzo da 32.768 Khz e inoltre dispone di una memoria SRAM da 256 byte, che essendo alimentata con la batteria si comporta da memoria non volatile, e in cui si scrivono i differenti parametri

MICROCONTROLLER

Modelli di controller di LCD

INTERFACCIA LCD 1



PIN	SEGNALE	DESCRIZIONE
1	GND	Connessione a massa dell'alimentazione
2	Vcc	Connessione positiva di +5V di alimentazione
3	VADJ	Tensione variabile per la regolazione del contrasto del display
4	RS	Segnale di controllo dei registri interni dell'LCD
5	R/W	Segnale di controllo per lettura e scrittura interna dell'LCD
6	E	Segnale per l'abilitazione interna dell'LCD
7	D0	Linea bidirezionale dei dati n° 1
8	D1	Linea bidirezionale dei dati n° 2
9	D2	Linea bidirezionale dei dati n° 3
10	D3	Linea bidirezionale dei dati n° 4
11	D4	Linea bidirezionale dei dati n° 5
12	D5	Linea bidirezionale dei dati n° 6
13	D6	Linea bidirezionale dei dati n° 7
14	D7	Linea bidirezionale dei dati n° 8
15	L+	Alimentazione positiva di 4,5V per la retroilluminazione (se il modello di LCD dispone di essa)
16	L-	Alimentazione di massa per la retroilluminazione (se il modello di LCD dispone di essa)

Assegnazione dei piedini nel connettore a fila singola per alloggiare l'LCD.

INTERFACCIA LCD 2



Connettore a doppia fila di terminali per supportare alcuni modelli di display LCD.

che si utilizzano nella scheda.

MEMORIA DI PROGRAMMA

La scheda LCD-CON2 è programmabile, e dispone di una memoria per scrivere le istruzioni che riceve dal master, per poi eseguirle in modo autonomo e indipendente. È formata da tre circuiti 24C65 (U2, U3 e U4) che sono memorie EEPROM da 8 Kbyte, con cui si ottiene la capacità totale di 24 Kbyte.

ALTRI ELEMENTI AUSILIARI

Il connettore J5 supporta la comunicazione RS232 e utilizza tre linee. Il pin 3 corrisponde al segnale di ricezione (Rx) il pin 2 a quello di trasmissione (Tx), e il 5 di J5 è la massa. Con il numero 10 si fa riferimento a quattro microinterruttori che selezionano il baude-rate nella comunicazione seriale, e i modi di lavoro, così come riportato nella tabella della figura. In ultimo il connettore J4, segnato con il numero 11 nello schema, permette la gestione di un massimo di otto segnali digitali, che opportunamente gestiti, permettono di selezionare visualizzazioni preprogrammate: a seconda dei segnali che sono attivati, si esegue una routine di visualizzazione memorizzata nella EEPROM.

Il flusso di controllo del programma è governato dagli otto segnali di ingresso che forniscono 255 combinazioni diverse di salto.

Il connettore J4 ha 10 terminali, dei quali il numero 2 è la massa, e quelli compresi tra 3 e 10 supportano gli otto segnali di ingresso. Il terminale 1 corrisponde con il segnale STB, che inizializza l'esecuzione di un programma per il rapido riconoscimento dei segnali di ingresso.

SW1-1	SW1-2	Modo di lavoro
OFF	OFF	On-Line
OFF	ON	Programmazione (solo con LCD-CON2)
ON	OFF	Esecuzione (solo con LCD-CON2)
ON	ON	Test

SW1-3	SW1-4	Baude
OFF	OFF	2400
OFF	ON	4800
ON	OFF	9600
ON	ON	19200

Microinterruttori SW1 per la configurazione della scheda di controllo.

Controller di LCD (II)

IL SISTEMA LOGICO DEI CONTROLLORI DI LCD

Confezionare i programmi necessari per fare in modo che l'utente risolva facilmente i suoi compiti, e che le istruzioni da utilizzare siano semplici e intuitive, è importante tanto quanto progettare dispositivi fisici che adattano il display LCD al master, tramite il quale si stabiliscono i messaggi da visualizzare. La scheda di controllo riceve le istruzioni dal master, le interpreta e le esegue in funzione del programma memorizzato sul PIC16C73. Le istruzioni sono divise in due tipi:

1°. Istruzioni che interpreta il PIC16C73.

2°. Istruzioni che edita l'utente sul master.

Il PIC16C73 deve ricevere le istruzioni dal master, secondo il formato e la sintassi che è capace di interpretare. Però quando l'utente edita il programma di visualizzazione sopra il PC, utilizza un repertorio di istruzioni più pratico e facile da comprendere, con l'obbiettivo di ottimizzare rapidamente lo sfruttamento del display LCD. Questa filosofia di lavoro è mostrata nella figura.



Le istruzioni che utilizza l'utente vengono tradotte dalla CPU del master, per fare in modo che il PIC della scheda di controllo le possa interpretare.

ISTRUZIONI DI ALTO E BASSO LIVELLO

Il PIC16C73 della scheda di controllo contiene nella sua memoria un codice di programma, capace di interpretare un repertorio di istruzioni che converte in segnali di governo per il modulo di LCD finale. La CPU raccoglie le istruzioni che utilizza l'utente, le traduce in formato adatto al PIC della scheda di controllo e le invia a quest'ultima tramite una RS-323. Le istruzioni che utilizza l'utente non sono uguali a quelle che interpreta la scheda di controllo. L'ideale è che l'utente sia una persona con poca specializzazione tecnica, utilizzi istruzioni intuitive per creare i messaggi e disponga di una interfaccia grafica amichevole. Queste istruzioni sono definite di alto livello e la CPU del master le traduce nel codice delle istruzioni che sarà interpretato dal PIC16C73 della scheda di controllo e riceve il nome di "basso livello", molto più vicino alla macchina e molto meno comprensibile.

FORMATO DELLE ISTRUZIONI DI BASSO LIVELLO

L'interprete dei comandi, ubicato nel PIC16C73, ammette un insieme di istruzioni con le quali si accede a tutte le possibilità di visualizzazione del display LCD. Tutte queste istruzioni rispondono alla stessa sintassi. Iniziano con un carattere ASCII a cui fanno seguito uno o più byte esadecimali, in funzione di operandi. Infine tutte le istruzioni terminano con il codice 0x0D Hex (Ritorno a capo).

Dopo aver completato l'esecuzione delle istruzioni ricevute, la scheda di controllo ritrasmette al master un codice di riconoscimento ACK del valore di 0x06 Hex. Alcune istruzioni, inoltre, restituiscono al master i parametri richiesti. Il master non invia altre istruzioni alla scheda di controllo fino a che questa non invia il riconoscimento della precedente. In seguito descriveremo alcune istruzioni di basso livello in cui sono esposti i seguenti campi:

CODICE: Rappresenta il carattere ASCII che definisce l'istruzione.

SINTASSI: Indica la sintassi dell'istruzione.

OPERANDI: Esprime i possibili operandi.

ESEMPIO: Semplice esempio di applicazione delle istruzioni.

RITORNO: Valore che restituisce al master.

COMMENTI: Chiarimenti per comprendere le istruzioni.

MICROCONTROLLER

Modelli di controller di LCD

Codice:	A
Sintassi:	A (catena) CR
Operandi:	(catena) è la catena dei caratteri da visualizzare, max. 40
Esempio:	AmicrosistemsCR
Ritorno:	ACK
Commenti:	Tutte le istruzioni terminano con un ritorno a capo (CR)

Caratteristiche dell'istruzione che visualizza caratteri sull'LCD.

ISTRUZIONE PER VISUALIZZARE CARATTERI

Con questa istruzione si mostra la catena dei caratteri che sarà indicata in seguito a partire dalla posizione attuale del cursore del display LCD.

I caratteri sino ad un massimo di 40, si memorizzano in un buffer interno. Nella figura sono riportate le principali caratteristiche di questa istruzione.

Codice:	C
Sintassi:	CCR
Operandi:	Nessuno
Esempio:	CCR
Ritorno:	ACK
Commenti:	Ogni volta che si esegue questa istruzione, si commuta lo stato del cursore ON/OFF

Caratteristiche dell'istruzione di attivazione e disattivazione del cursore.

ISTRUZIONE PER ATTIVARE O DISATTIVARE IL CURSORE

Mostra o nasconde il cursore nella sua posizione attuale. Il cursore si rappresenta come una linea orizzontale nella posizione inferiore del carattere in cui si trova.

Comando	Sintassi	Descrizione
A	A[catena]CR	Visualizza la catena di caratteri a partire dalla posizione attuale del cursore.
B	B[aa][bb]CR	Produce un effetto di (bb) intermittenza con un periodo stabilito da (aa).
C	CCR	Attiva o no la visualizzazione del cursore.
D	DCR	Disattiva il carattere ACII localizzato nella posizione del cursore.
E	E[indirizzo]CR	Sposta il cursore in una nuova posizione.
F	F[catena]CR	Inserisce una catena di caratteri a partire dalla posizione del cursore spostandolo verso sinistra.
G	G[nn][byte1][byte2]...[byte8]CR	Genera uno degli otto possibili caratteri grafici definibili dall'utente.
H	HCR	Inizializza il display LCD cancellandolo e posizionando il cursore nella posizione iniziale.
I	I[catena]CR	Inserisce una catena di caratteri a partire dalla posizione del cursore spostandolo a destra.
J	JCR	Attiva o disattiva il display LCD.
K	KCR	Attiva o disattiva la visualizzazione dell'ora in corso.
L	LCR	Attiva o disattiva la visualizzazione della data corrente.
M	M[mm][hh]CR	Regola l'orologio con i minuti [mm] e le ore [hh].
N	N[dd][mm]CR	Regola il calendario con il giorno [dd] e il mese [mm].
O	OCR	Attiva o disattiva il lampeggiamento e/o il cursore nella posizione in cui si trova.
P	PCR	Legge e restituisce la posizione attuale dove si trova il cursore.
Q	Q[valor]CR	Temporizza tanti decimi di secondo come si definiscono nel [valore].
R	RCR	Attiva o disattiva le luci di fondo (backlight) nei display che ce l'hanno.
S	S[dirH][dirL]CR	Salto incondizionato all'indirizzo [indH][indL].
T	T[valore][indH][indL]CR	Salto incondizionato all'indirizzo [indH][indL] in funzione del [valore] presente sul connettore esterno J4.
U	UCR	Fine del programma.
V	V[valor]CR	Carica la variabile interna con [valore].
X	X[indH][indL]CR	Decrementa la variabile e salta a [indH][indL]se il risultato è 00.
Y	YCR	Inizializza la scheda di controllo LCD-CON.
?	?CR	Restituisce il valore dell'ultimo indirizzo di memoria programmabile disponibile.

Repertorio di istruzioni di basso livello che ammette la scheda di controllo LCD-CON2

ISTRUZIONE DI SALTO CONDIZIONALE

Devia il flusso di controllo del programma quando si compie una determinata condizione. La scheda di controllo LCD-CON2 scrive il programma di visualizzazione nella memoria EEPROM da 24 Kbyte, che occupa gli indirizzi da 0000 a 5FFF Hex. Le istruzioni si scrivono negli indirizzi correlati iniziando da 0000 Hex. Ogni istruzione occupa un certo numero di byte. La condizione per cui si realizza il salto viene stabilita dal valore binario presente nelle otto linee di ingresso a cui si accede tramite il connettore J4 Ext.Input della scheda di controllo.

REPERTORIO DI ISTRUZIONI DI BASSO LIVELLO

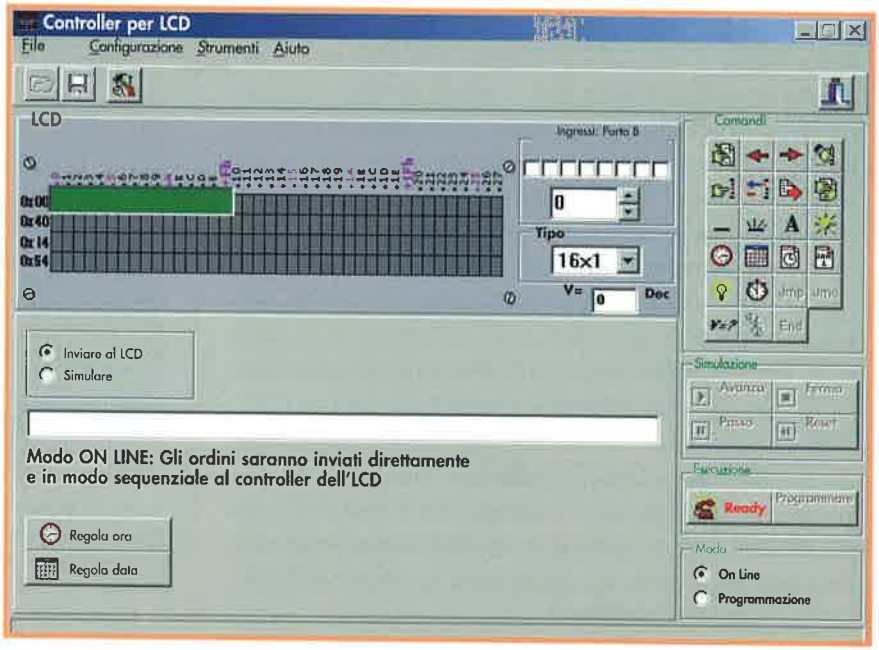
Nella tabella a sinistra sono rappresentate tutte le istruzioni, con le loro principali

caratteristiche, che la scheda di controllo LCD-CON2 è capace di interpretare e convertire nei corrispondenti segnali di controllo che sono inviati al terminale del modulo LCD.

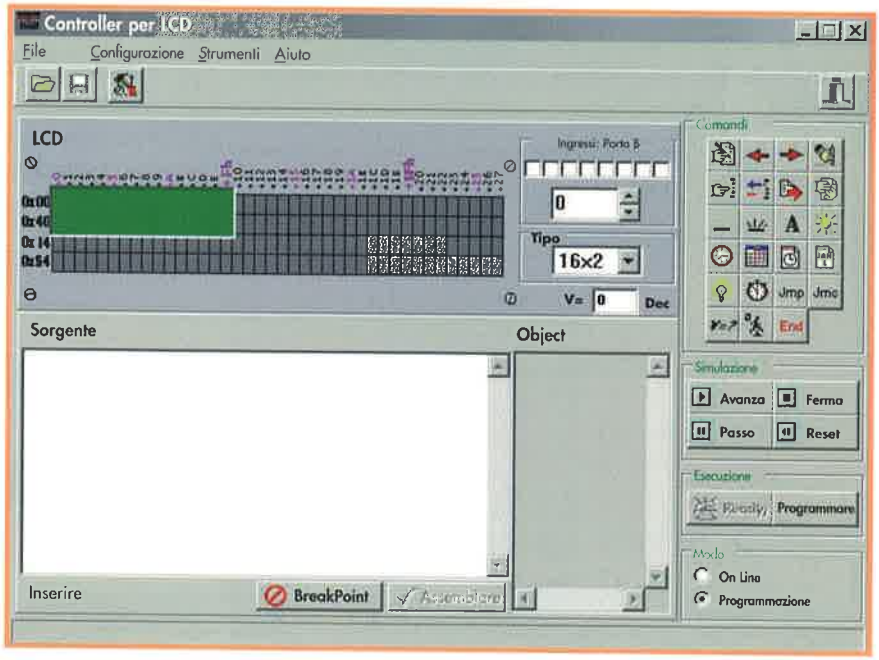
L'INTERFACCIA GRAFICA

Per semplificare la gestione del controller LCD-CON il suo produttore, Ingeniería de Microsistemas Programados S.L., ha sviluppato un software molto pratico che implementa l'interfaccia grafica e che è stato battezzato con il nome di Ambiente Integrato di Sviluppo EID, il quale permette le seguenti funzioni:

- Inviare istruzioni in modo on-line dal master alla scheda di controllo.
- Simulare i programmi di visualizzazione sul monitor del PC prima di eseguirli sopra LCD.
- Editare il programma con il linguaggio di alto livello.
- Assemblare le istruzioni di alto livello e convertirle in istruzioni di basso livello, interpretabili dal PIC della scheda di controllo.
- Mettere a punto il programma.
- Trasferire i programmi assemblati alla scheda di controllo LCD-CON.



Videata di lavoro dell'ambiente EID in modo on-line.



Videata di lavoro di EID in modo programmazione.

L'EID è basato su un repertorio di istruzioni di alto livello, che pur essendo equivalenti a quelle di basso livello, risultano più comprensibili e facili da gestire. Con l'editor dei testi dell'EID si può cancellare, inserire, muovere, duplicare, salvare su disco, recuperare, ecc. L'assembler di questo ambiente traduce le istruzioni di alto livello in altre di basso livello, che possono essere interpretate con il firmware della scheda di controllo LCD-CON. Il simulatore dell'EID è molto interessante, perché unisce la possibilità di presentare nel display del PC i messaggi che vogliamo visualizzare sull'LCD. Infine EID possiede un sistema di aiuto on-line, in modo che posizionando il cursore su qualsiasi icona appare una finestra con un commento. I requisiti per l'installazione dell'EID sono:

- PC compatibile con processore Pentium ad almeno 100 MHz.
- 2 MB come minimo di spazio libero sul disco fisso.
- 8 MB come minimo di memoria RAM.
- Sistema Operativo Windows 95/98.
- Floppy disc da 3,5".
- Porta seriale libera.

MICROCONTROLLER

Modelli di controller di LCD

LA FINESTRA DI LAVORO

Nella figura è riportata la finestra che appare sul PC lavorando nell'ambiente EID in modo on-line, in cui le istruzioni si inviano direttamente alla scheda di controllo. Esiste una finestra a video, denominata LCD, in cui si rappresenta il display LCD disponibile, potendo scegliere tra diverse configurazioni di file e caratteri. Nel modo simulazione i comandi si inviano a questa finestra al posto della scheda di controllo, simulando l'effetto che produrrà ogni istruzione sul display LCD. Sulla linea dei comandi si digitano le differenti istruzioni con i loro operandi. Se si attiva il pulsante "Inviare a LCD", l'istruzione si invia tramite il canale seriale alla scheda di controllo. Se è attivato il pulsante "Simulazione", l'esecuzione dell'istruzione si simula sulla finestra LCD del PC. Inoltre si può lavorare in modo programmazione, descrivendo tutte le istruzioni nella memoria EEPROM della scheda di controllo.

Come si può vedere nella figura, la videata in modo programmazione dispone di una finestra "Sorgente", in cui lavora l'editor dei testi ASCII, con cui si editano le linee del programma sorgente. Ognuna delle linee può terminare con un commento chiarificatore preceduto da REM.

Instruzione	Sintassi	Descrizione
Print"	Print"[catena]	Visualizza la catena di caratteri a partire dalla posizione del cursore. Il numero massimo di caratteri di cui è composta la catena è 40.
		Si possono visualizzare caratteri grafici precedentemente definiti. In questo caso si utilizza la barra inversa (backslash) seguita dal numero del carattere grafico desiderato.
		Print "Ciao **Print** "Addio **Print** "Microsystems\0\1
Insert>"	Insert>"[catena]	Inserisce la catena di caratteri a partire dalla posizione del cursore, spostandosi verso destra.
		La sintassi è simile all'istruzione Print.
		Insert> "Ciao **Insert>** "Microsystems\0\1
Insert<"	Insert<"[catena]	Inserisce la catena di caratteri a partire dalla posizione del cursore, spostandosi verso sinistra.
		La sintassi è simile a quella dell'istruzione precedente.
Home	Home	Cancella il display LCD e posiziona il cursore nella prima posizione della prima linea.
SetPos	SetPos [ind]	Posiziona il cursore all'indirizzo specificato.
GetPos	GetPos	Il controller restituisce al master la posizione attuale del cursore, seguita da ACK.
GetChar	GetChar	Il controller restituisce al master il carattere che c'è nella posizione dove si trova il cursore.
Power	Power	Accende/spegne LCD. Il fatto che LCD sia spento non significa che l'informazione che conteneva sia perduta; la successiva istruzione Power la restaura.
Cursor	Cursor	Attiva/disattiva la visualizzazione del cursore

Simboli e caratteristiche di nove istruzioni di alto livello.



Icone corrispondenti alle 23 istruzioni di alto livello dell'ambiente EID.

ISTRUZIONI DI ALTO LIVELLO

Si tratta di un repertorio di 23 istruzioni che corrispondono a quelle di basso livello già studiate. Nella finestra di lavoro dell'EID esiste un quadro in cui sono rappresentate ognuna di queste istruzioni mediante un'icona: in questo modo le istruzioni del programma si possono editare senza la necessità di digitarle, basta cliccare con il mouse sopra l'icona corrispondente. Nella figura sono riportate le principali caratteristiche delle 9 istruzioni di alto livello.

ESEMPIO DI APPLICAZIONE

Si tratta di costruire un programma che presenta sul display LCD la parola "Ingeniería" e "Microsistemas" alternativamente. Ogni parola si deve mantenere per due secondi sul display e l'intervallo fra entrambe deve essere di un secondo.

LE ISTRUZIONI CHE SI DEVONO USARE SONO:

- **Home:** Inizializza il display LCD.
- **Power:** Alimenta il display LCD.
- **Pause:** Temporizzazione.
- **Print:** Visualizza una catena di caratteri.
- **Jmp:** Salto all'etichetta indicata.

```
Loop                    Home
REM inizializza LCD
Power                   REM attiva LCD
Print "Ingeniería"      REM visualizza Ingeniería
Pause 20                REM temporizza 2 secondi
Home                    REM inizializza e cancella LCD
Pause 10                REM temporizza 1 secondo
Print "Microsistemas"   REM visualizza Microsistemas
Pause20
Power
Pause 10
Jmp Loop                REM salta a etichetta Loop
```

Microinterruttori SW1 per la configurazione della scheda di controllo.

Trasferimento di dati tramite R.F. (I)

UNA COMUNICAZIONE IDEALE

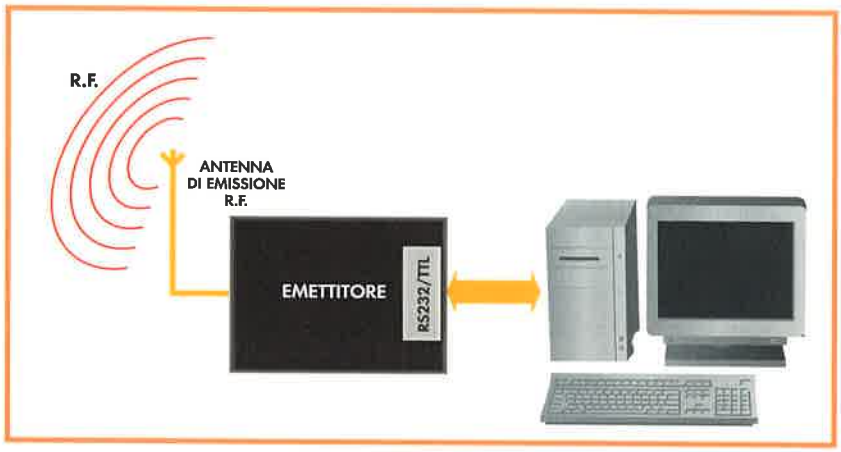
Trasferire informazioni tra sistemi digitali, comandare strumenti remoti, ricevere dati da un ambiente distante senza la necessità di un collegamento fisico mediante cavi fra emettitore e ricevitore, per mezzo di radiofrequenza, risulta una delle applicazioni elettroniche più interessanti. In seguito citeremo alcuni dei settori più interessanti in cui si può impiegare questa tecnologia:

1°. Controllo e gestione dei dispositivi remoti mediante invio di ordini e comandi: connessione/sconnessione di allarmi, controlli di accessi, porte dei garages, ecc.

2°. Ricezione di dati in arrivo da periferiche remote per la successiva elaborazione: trasduttori di temperatura, pressione, umidità, ecc. Tutti i tipi di variabili analogiche e digitali.

3°. Trasferimento di informazioni tra sistemi molto distanti: risultati di determinate operazioni, stato interno di variabili, ecc.

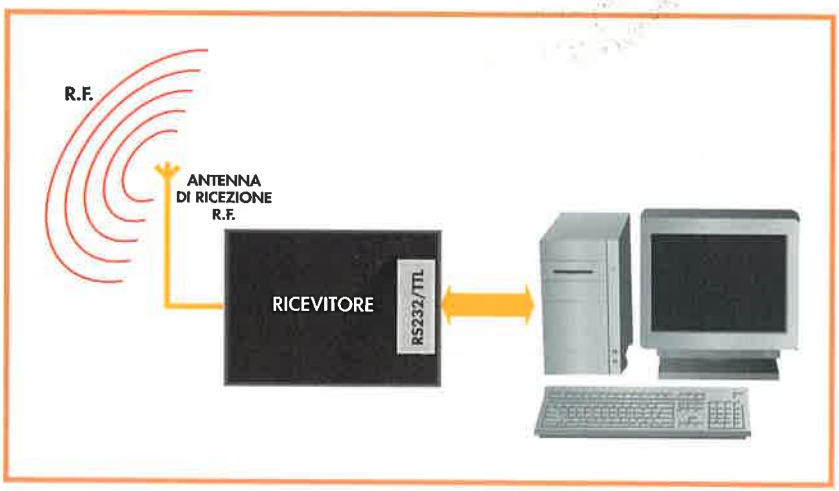
4°. Applicazioni in microrobotica: comunicazione fra microrobot per realizzare compiti congiunti, controllo movimenti, informazioni dell'ambiente, ecc.



Schema di funzionamento dell'emettitore di R.F.

EMISSIONE E RICEZIONE DI R.F.

Esiste una certa similitudine fra la funzione della scheda di controllo LCD e i dispositivi di R.F. Nel primo caso, la scheda di controllo riceveva le istruzioni del master – che poteva essere un PC –, le interpretava e generava segnali di controllo per il display. In R.F. esiste una scheda incaricata di ricevere informazioni in forma di onde elettromagnetiche a R.F., interpretarle e fornire il risultato a un PC per la successiva elaborazione. Nelle seguenti figure si presenta l'attuazione semplificata delle schede di emissione e ricezione di R.F.. La comunicazione fra le schede di emissione e ricezione di R.F. e l'applicazione finale a cui le destina l'utente, si realizza mediante comunicazione seriale RS-232, o mediante livelli logici TTL che si applicano direttamente al microcontroller di queste schede. Nel caso della scheda di emissione, questa riceve via seriale dal PC i comandi di configura-



Schema di funzionamento del ricevitore di R.F.

zione e i dati da inviare all'utente.

Il microcontroller della scheda interpreta i comandi e li invia, insieme con i dati, tramite l'antenna di emissione, secondo un determinato protocollo. Nel caso della ricezione si può imporre che sia un sistema, che fa parte di un altro più grande, che riceve i dati tramite radiofrequenza, li verifica, li impacchetta, e li invia in serie al dispositivo o all'applicazione dell'utente.

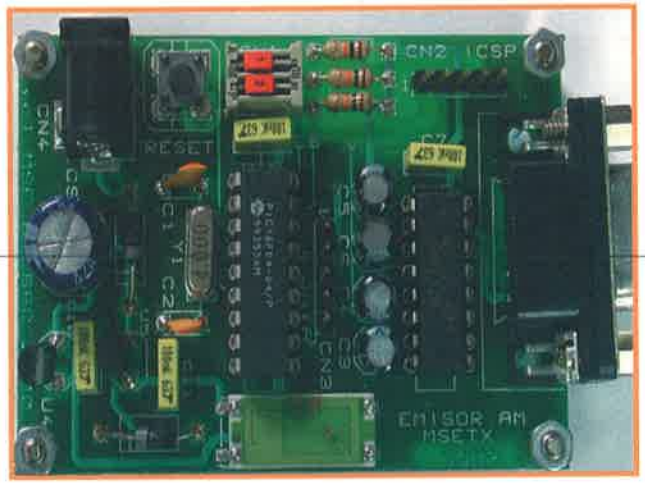
CARATTERISTICHE	Min	Tipico	Max	Unità
Tensione di alimentazione	4		14	VDC
Consumo		3		mA
Frequenza di lavoro		433,92		MHz
Velocità di trasmissione dei dati			4000	Baude
Range di temperatura	-40		+80	ºC

Principali caratteristiche del modulo emettitore RT1-433 della ditta italiana "Telecontrolli".

Riassumendo, una scheda codifica l'informazione digitale per trasmetterla poi via R.F. e l'altra la riceve, sempre via R.F., la codifica e la trasforma in informazione binaria.

EMISSIONE E RICEZIONE DI R.F. COMMERCIALI

Con l'obiettivo di analizzare il comportamento di questi moduli che comunicano tramite R.F., descriviamo l'emettitore MSETX e il ricevitore MSERX, entrambi basati su un PIC16F84, che vediamo nella figura.



Emettitore MSETX

Nella scheda di emissione si utilizza un PIC16F84 per controllare la trasmissione alla frequenza standard di 433,9 MHz. Il microcontroller riceve l'informazione digitale da trasmettere via RS-232, la decodifica e la invia tramite R.F.

La scheda di ricezione raccoglie i segnali di R.F. tramite un'antenna e dispone di un altro PIC16F84 che è incaricato di decodificarla, convertirla nell'informazione digitale iniziale e portarla all'esterno via RS-232.

L'EMETTITORE MSETX

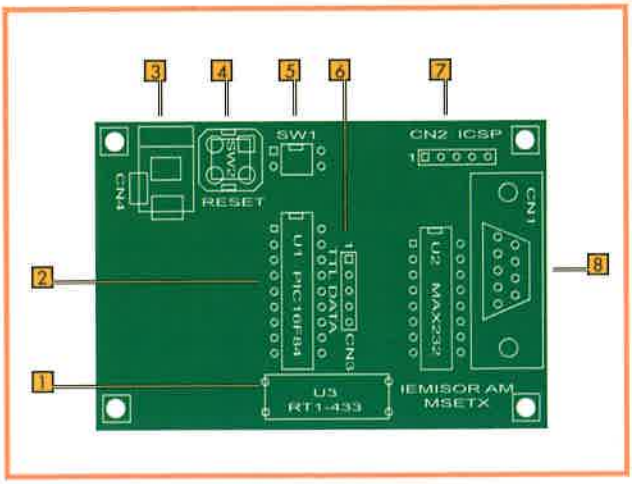
Il cuore dell'emettitore R.F. è il modulo RT1-433, prodotto dalla ditta italiana "Telecontrolli", che consiste fondamentalmente in uno stadio di potenza che genera una portante di 433,92 MHz, modulata in ampiezza (AM). È un circuito ibrido che, mediante un processo di regolazione che utilizza tecniche laser, offre una grande stabilità di frequenza. L'antenna di emissione è integrata nel circuito, ed è possibile ottenere una portata di varie decine di metri, a seconda delle condizioni ambientali. Le sue caratteristiche principali sono riportate nella tabella della figura.

CIRCUITO DI CONTROLLO

La gestione dell'emettitore MSETX è basata sul programma che alloggia nel PIC16F84 e che supporta i seguenti compiti:

a. Gestisce la comunicazione seriale con il sistema di applicazione dell'utente, che riceve i comandi di configurazione, e i dati che si vorranno inviare.

b. Interpreta i comandi di configurazione e li scrive in



Distribuzione dei blocchi principali sulla scheda MSETX.

una memoria interna non volatile.

I dati sono impacchettati con formato prestabilito, e si genera un'informazione di intestazione e un'altra di test o "checksum" in modo da garantire la trasmissione.

Pin Nº	Segnale	Descrizione
1	Rx	Ricezione dei dati tramite l'applicazione dell'utente verso l'emettitore MSETX.
2	Tx	Trasmissione dei dati dall'emettitore MSETX verso l'applicazione dell'utente.
3	Vcc	Tensione di alimentazione di +5VDC. Non necessaria.
4	MCLR	Inizializzazione dell'emettitore MSETX. L'utente dalla sua applicazione può resettare l'emettitore.
5	GND	Connessione della massa dell'alimentazione.

Assegnazione delle funzioni dei cinque terminali del connettore CN3 (TTL DATA).

c. La trama, ottenuta con l'intestazione, i dati, il controllo ecc. viene inviata in modo sequenziale, bit a bit, al modulo RT-433, da cui sono irradiati tramite R.F. I blocchi principali della scheda di emissione MSETX sono riportati nella figura e sono i seguenti:

1°. Modulo RT1-433.
2°. PIC16F84.
3°. Ingresso dell'alimentazione.
4°. Reset.
5°. Microinterruttore di configurazione.
6°. Connettore TTL DATA (CN3).
7°. Connettore ICSP (CN2)
8°. Connettore DB9 (CN1).

ALIMENTAZIONE, RESET E CONFIGURAZIONE

Il connettore CN4 riceve l'alimentazione, che deve arrivare dal secondario di un trasformatore che fornisca 12 VAC. All'interno della scheda esiste uno stadio di raddrizzamento, di filtro e di stabilizzazione per generare +5VDC per l'elettronica di controllo, e +13VDC per il modulo RT1-433. Per resettare il sistema si usa il pulsante SW-2, che resetta il PIC16F84, senza modificare i parametri interni di configurazione. Gli interruttori SW-1 a due contatti, servono per determinare i baude con cui la scheda comunica con il sistema di applicazione dell'utente, che può essere un PC. Se SW-1 e SW-2 sono a 0 la comunicazione si stabilisce a 2.400 baude. Se SW-1 = 0 e SW-2 = 1 si realizza a 4.800 baude, e se i due sono a 1 la comunicazione è a 9.600 baude.

I TRE CONNETTORI

Il connettore CN3 a cinque contatti è chiamato TTL DATA, e serve per supportare la comunicazione con l'utente, mediante livelli logici compatibili TTL.

I segnali che invia l'utente si applicano direttamente al PIC16F84 e nella tabella della figura si mostra l'assegnazione di ognuno dei terminali di questo connettore.

Il costruttore della scheda, Ingeniería de Microsistemas Programados S.L., utilizza il connettore

Pin Nº	Segnale	Descrizione
1	SPC	Serial Programing Clock. Segnale di clock di sincronismo al momento della scrittura.
2	SPD	Serial Programing Data. Segnale dei dati seriali al momento della scrittura.
3	Vcc	Tensione di alimentazione di +5VCC.
4	MCLR/VPP	Segnale di reset del dispositivo da registrare e punto di applicazione della tensione Vpp di registrazione.
5	GND	Massa dell'alimentazione.

Assegnazione delle funzioni ai terminali del connettore CN2 (ICSP)

Pin Nº	Segnale	Descrizione
2	TxD	Trasmissione dei dati RS232 tramite l'emettitore MSETX verso il sistema di applicazione dell'utente.
3	RxD	Ricezione dei dati RS232 dall'applicazione dell'utente verso l'emettitore MSETX.
5	GND	Segnale di massa comune.

Assegnazione di funzioni ai tre terminali che supportano la comunicazione seriale RS-232 nel connettore CN1.

CN2 (ICSP) per scrivere nella memoria del PIC il firmware di controllo.

Mediante questo connettore il fabbricante può aggiornare i suoi clienti dell'ultima versione del firmware in modo rapido ed economico.

Nella figura si vede la tabella di assegnazione delle funzioni dei terminali di questo connettore. Infine, il connettore CN1 (DB9) possiede nove contatti, e supporta i segnali utilizzati nella comunicazione RS-232. Il circuito integrato MAX232 adatta i livelli TTL alla normativa RS-232 e viceversa. Utilizza solamente tre contatti, le cui funzioni sono rappresentate nella figura.

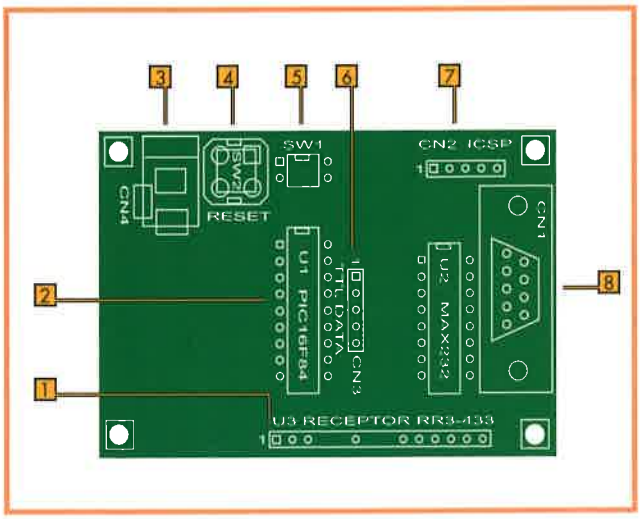
CARATTERISTICHE	Min	Tipico	Max	Unità
Tensione di alimentazione	4,5	5	5,5	VDC
Consumo		2,5	3,5	mA
Frequenza di lavoro	200		450	MHz
Tolleranza di sintonia		+/- 0,5		MHz
Ampiezza di banda a –3dB		+/- 2	+/- 3	MHz
Velocità di ricezione dei dati			2000	Baude
Sensibilità RF	100	-105		.dBm
Livello di spettro emesso		-65	-60	.dBm
Livello "0" di uscita			0,6	V
Livello "1" di uscita	3,6			V
Range di temperatura	-25		+80	°C

Principali caratteristiche del modulo ricevitore RR3-433 di Telecontrolli.

IL RICEVITORE MSERX

Realizza il lavoro inverso dell'emettitore MSETX, cioè riceve tramite l'antenna le onde R.F., le interpreta e le codifica in informazioni digitali, quindi le invia all'utente mediante RS-232 o segnali TTL. Le principali sezioni del ricevitore sono le seguenti:

1ª. Modulo ricevitore.
2ª. PIC16F84.
3ª. Alimentazione.
4ª. Reset.
5ª. Switches di configurazione.
6ª. Connettore TTL DATA (CN3).
7ª. Connettore ICSP (CN2).
8ª. Connettore DB9 (CN1).



Distribuzione delle sezioni principali del ricevitore MSERX.

I blocchi del ricevitore realizzano la stessa funzione degli omologhi dell'emettitore, però al contrario. Il modulo ricevitore è il modello RR3-433 di "Telecontrolli". Ha una grande stabilità di frequenza e offre una buona resistenza alle vibrazioni meccaniche, oltre ad un ampio margine di temperatura.

La sua sensibilità si può migliorare saldando un filo, con funzione di antenna, sul terminale ANT della scheda, con una lunghezza di circa 35 cm per funzionare come antenna a mezza onda. Riceve informazioni via R.F. e le invia al PIC16F84 che le interpreta, le prepara e le invia in serie all'utente.

Le principali caratteristiche del modulo RR3-433 sono riportate nella tabella in alto.

IL MICROCONTROLLER RICEVITORE

I tre compiti che realizza il PIC del ricevitore sono:

1°. Gestisce la comunicazione seriale.

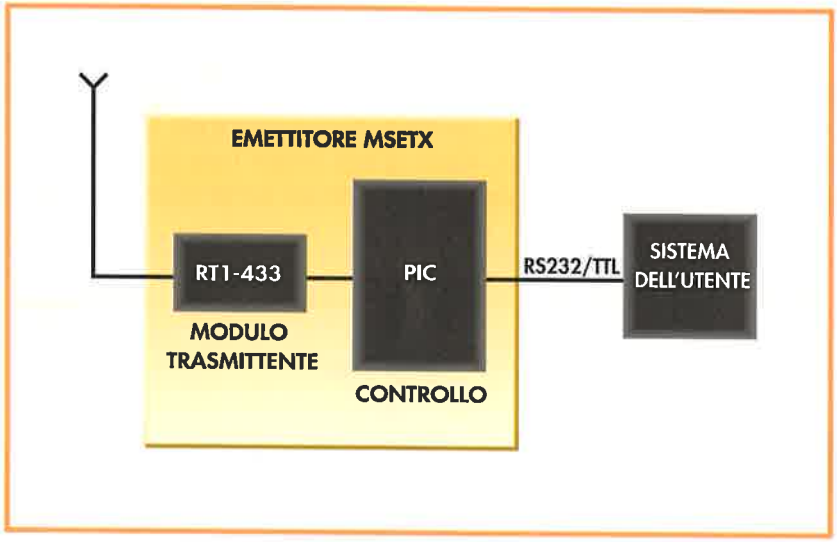
2°. Interpreta i comandi di configurazione ricevuti e li scrive nella memoria non volatile. Scompone i dati ricevuti via R.F., analizza le informazioni delle intestazioni, determina se il messaggio è diretto al ricevitore adeguato. Infine verifica la validità dei dati mediante il checksum.

3°. Dal totale della trama ricevuta estrae l'informazione utile che invia all'utente in via seriale.

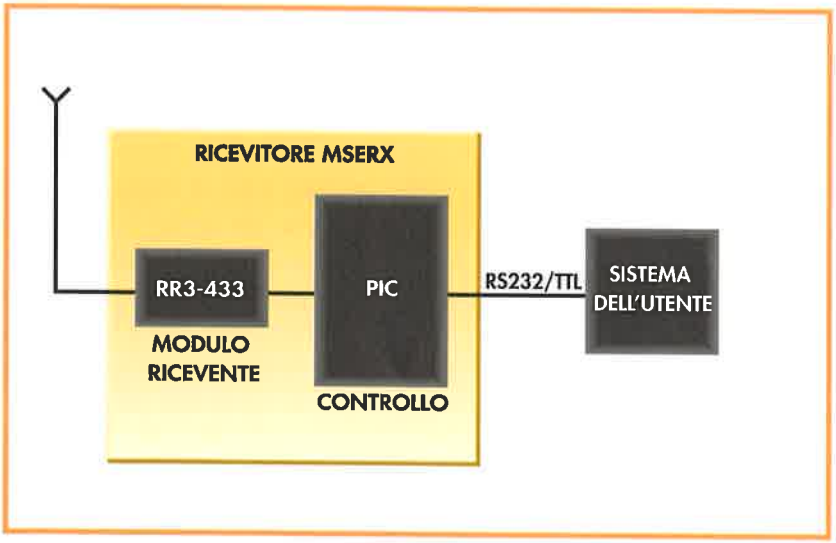
Trasferimento dei dati tramite R.F. (II)

PROTOCOLLO DI COMUNICAZIONE

L'emettitore di R.F. studiato, MSETX, comunica con il sistema dell'utente in modo seriale e bidirezionale. Anche il modulo trasmettitore RT1-433 comunica in modo seriale e riceve le informazioni bit a bit, per inviarle successivamente tramite R.F. così come è rappresentato in modo schematico nella figura. Il ricevitore MSERX supporta un flusso di informazioni simili al trasmettitore. Da un lato la comunicazione con l'utente è in modo seriale e di verso bidirezionale, e dall'altro l'utente invia i comandi di configurazione, e riceve lo stato interno, così come i dati ricevuti tramite R.F., in serie, bit a bit, come si vede nella figura. Nella comunicazione seriale dobbiamo tener conto di tre aspetti rilevanti per garantire un corretto trasferimento e una successiva interpretazione.



Flusso di informazioni nell'emettitore MSETX.



Flusso di informazioni nel ricevitore MSERX.

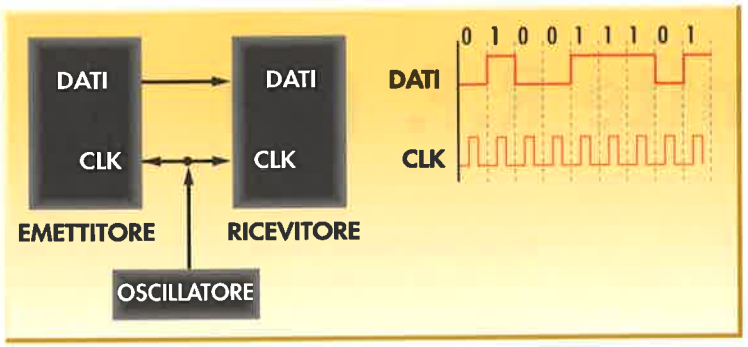
1°. **Sincronismo dei bit.**
2°. **Sincronismo del carattere.**
3°. **Sincronismo del messaggio.**

SINCRONISMO DEI BIT

Questo sincronismo ha il compito di ottenere che tutti i bit emessi durante un intervallo di tempo determinato siano raccolti nello stesso intervallo. In questo modo "non si perde" nessun bit nel viaggio e per fare questo esistono due alternative che spiegheremo a seguire:

- **Modo sincrono:** ogni bit emesso va accompagnato da un impulso di clock. Questo segnale di clock è comune sia per l'emettitore che per il ricevitore. Quest'ultimo utilizza il fronte o il livello per stabilire il momento di acquisizione del bit sulla linea dei dati. Per supportare questo sincronismo sono necessarie due linee fra emettitore e ricevitore: la linea dei dati (DATA) e quella del clock (CLK), così come si presenta graficamente nella figura.

- **Modo asincrono:** non necessita di un clock ausiliario. Sia l'emettitore che il ricevitore, trasferiscono e raccolgono i loro bit in un intervallo di tempo "t"

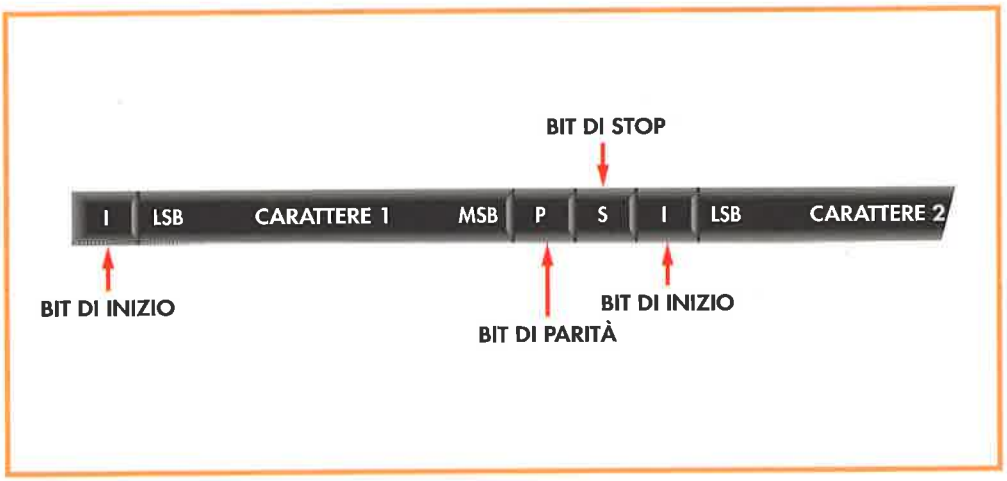


Schema di funzionamento del sincronismo di bit in modo sincrono, ed esempio di trasferimento.

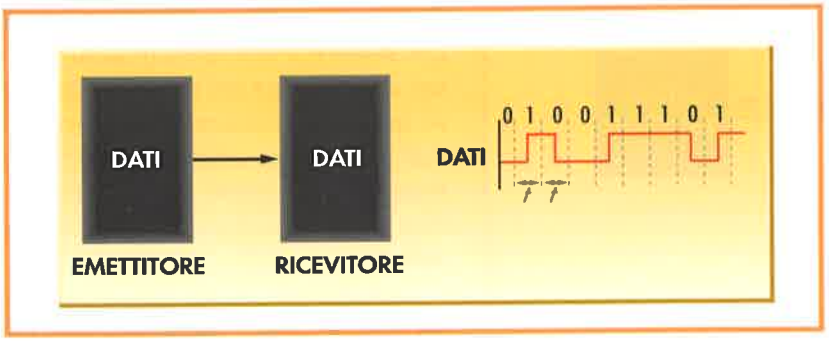
uguale per entrambi. Se per esempio si decide di realizzare un trasferimento di 1.000 bit al secondo (1.000 baude) ogni bit che fornisce il trasmettitore deve rimanere sulla linea dei dati per 1 millisecondo, che è il tempo che ha a disposizione il ricevitore per raccoglierlo. Di solito, per motivi di sicurezza, il ricevitore fa un campionamento e raccoglie il dato a metà dell'intervallo del tempo, come si vede nella figura. Nel caso particolare dell'emettitore e del ricevitore che stiamo descrivendo in queste pagine, entrambi utilizzano il sincronismo dei bit in modo asincrono, che è quello impiegato nella norma RS-232.

SINCRONISMO DI CARATTERE

Superato il sincronismo di ogni bit dobbiamo risolvere la delimitazione fra ogni pacchetto di bit trasferito, ossia, dove inizia e dove finisce ognuno dei caratteri che formano il messaggio. Questo si chiama sincronismo di carattere. Il metodo più comune per implementare il sincronismo di carattere è rappresentato graficamente nella figura, e consiste nell'iniziare ogni carattere con un bit denominato di "inizio". Dopo arrivano i bit di informazione che configurano il carattere, e in seguito, in modo opzionale, può esserci un bit di "parità", terminando con uno o più bit di "stop" o "fermata". Come si può vedere nella figura, davanti a ognuno dei caratteri c'è un bit di inizio. Secondo la



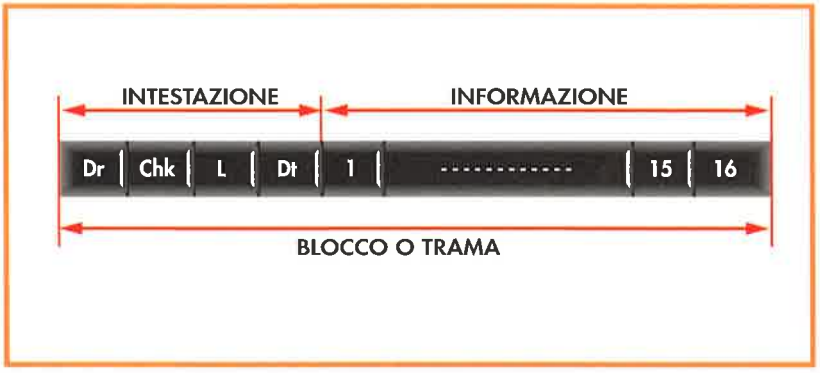
Struttura dei bit che compongono ogni carattere.



Nel modo asincrono non esiste segnale di clock e ogni bit rimane stabile sulla linea dei dati per un tempo fisso.

norma RS-232, il bit di inizio consiste in una transizione del livello 1 al livello 0 sulla linea dei dati. Dopo vengono i bit che compongono il carattere, iniziando da quelli meno significativi (LSB). In base al dispositivo utilizzato, il carattere può essere di 5, 6, 7 o 8 bit. Attualmente il carattere è quasi sempre di 8 bit. Il bit di parità è opzionale. Si può lavorare con parità pari o dispari. Se si utilizza la parità pari, il livello logico del bit P è tale che la somma di tutti i bit che valgono 1 nel

carattere, più il proprio bit P, deve dare un totale di 1 che sia pari. La parità dispari funziona al contrario. Il bit S di stop è quello che termina tutti i caratteri. Nella norma RS-232 questo bit è di livello 1, in questo modo, quando la linea dei dati torna a 0 si capirà che si tratta di un nuovo bit di inizio per il carattere successivo. Alcuni sistemi permettono di utilizzare 1, 1,5 o 2 bit di stop. Tutti i bit hanno la stessa durata, per questo se si lavora a 1.000 baude ogni bit rimane stabile sulla linea dei dati 1 ms.



Ogni blocco o trama è composto da 20 caratteri o byte, dei quali i primi 4 corrispondono all'informazione di testa.

ESERCIZIO

Quanto tempo impiegheremmo a trasferire ogni carattere lavorando a 1.000 baude e sapendo che ogni carattere è formato da 7 bit e 1,5 bit di stop?

Soluzione: Ogni bit impiega ad essere trasferito 1 ms lavorando a 1.000 baude. Il carattere ha 7 bit di informazione più uno di inizio, più un altro di parità e 1,5 bit di stop. In totale 10,5 bit, il che suppone un tempo di 10,5 ms per trasferire ogni carattere. Nel caso dei dispositivi commerciali MSETX e MSERX, il formato di ogni carattere è composto da 1 bit di inizio, 8 bit di informazione e 2 bit di stop. Lavorando a 9.600 baude per trasferire ogni carattere, si utilizza un tempo approssimato di 1 ms.

SINCRONISMO DI MESSAGGIO

Serve per garantire l'integrità dell'informazione trasferita. Il totale delle informazioni da trasferire si divide in blocchi, ognuno dei quali può avere un numero fisso o variabile di caratteri. Ogni blocco dispone di un'informazione di testa, che determina una serie di parametri: chi invia l'informazione, a chi è diretta, quanti caratteri ci sono in ogni blocco, ecc. Nei moduli MSERX e MSETX ogni blocco o trama è di lunghezza fissa e contiene 20 caratteri o byte, dei quali i primi 4 corrispondono alle informazioni di testa e i 16 successivi compongono una informazione utile. I quattro byte di testa hanno il seguente significato:

Dr: Indirizzo del ricevitore. Può valere 253, tutti rimangono all'ascolto, però accetterà l'informazione solo quello il cui indirizzo coincide con il valore Dr. L'indirizzo 254 è universale e con esso tutti i ricevitori accettano l'informazione.

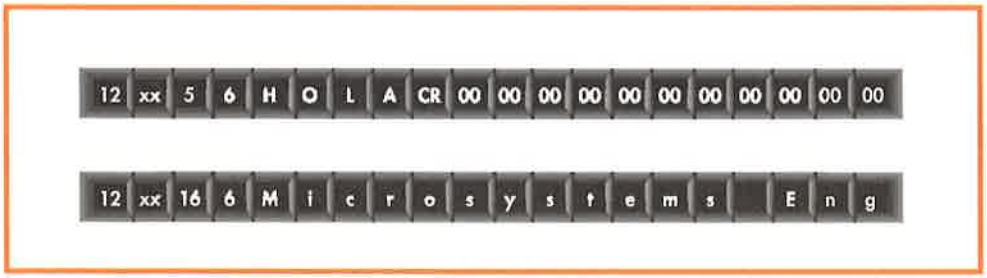
Chk: Byte di Checksum. Si tratta di un byte calcolato dall'emettitore mediante una serie di operazioni logico-aritmetiche con tutti i byte da trasferire. Ha il compito di evitare la corruzione dell'informazione durante il trasferimento.

L: indica il numero di byte utili dell'informazione. Nel formato della figura può avere un massimo di 16 caratteri, però sono validi solo i 'n' primi

Dt: Indirizzo del trasmettitore.

DATI: Dopo i quattro byte di testa seguono quelli utili dell'informazione. Si trasferisce sempre una catena di 16 byte, e quelli che non si utilizzano si riempiono come caratteri NULL (000).

Quando tramite l'applicazione dell'utente si inviano i caratteri da trasmettere mediante la norma RS-232, il trasmettitore lo fa scrivendo in buffer interno, e quando si completa con 16 byte si procede alla trasmissione automatica di tutta la trama. Se al posto di ricevere i 16 caratteri si riceve prima un carattere di "ritorno a capo" CR = 013, il resto del buffer si riempie con codice NULL (000) e si procede alla trasmissione, considerando CR come un carattere in più. Nella figura sono riportate due trame distinte, nella prima si trasferisce una catena con meno di 16 caratteri (Hola) e nella seconda un altro messaggio. In entram-



Esempio della struttura delle due trame, in cui tutti i byte sono espressi in esadecimale.

bi si suppone che siano diretti al ricevitore 12 e che arrivino dall'emettitore 6. Si fa riferimento al byte di checksum con XX.

CALCOLO DEL TEMPO DI TRASFERIMENTO

Per calcolare il tempo totale che dura un trasferimento, dobbiamo considerare i seguenti tempi parziali:

1°. - Tempo di trasferimento fra l'applicazione dell'utente e l'emettitore MSETX.

Si suppone di lavorare a 9.600 baude e che ogni carattere impieghi per trasferirsi 1 ms. Il sistema utente invia al MSETX il byte che definisce l'indirizzo del ricevitore (Dr) e quelli dei dati, con un massimo di 16. Totale 17 ms, come si può vedere dalla figura.

2°. - Tempo di emissione in R.F. tramite MSETX.

Dopo aver ricevuto il messaggio dall'applicazione dell'utente, MSETX scompone la trama completa di 20 byte, dei quali calcola il byte di checksum (Chk), il numero di caratteri valido (L), e aggiunge il proprio indirizzo (Dt). Dato che la velocità massima della R.F. è di 2.000 baude, un carattere impiega 5 ms ad essere trasmesso, e la trama completa approssimativamente 100 ms.

3°.- Tempo di ricezione del messaggio tramite R.F.

Il ricevitore MSERX inizia a ricevere tramite R.F. la trama dei 20 byte durante gli stessi 20 ms. in cui viene trasmessa. L'emettitore e il ricevitore devono essere sincronizzati alla stessa frequenza di 2.000 baude. Ricevuta la trama si separa l'informazione dall'intestazione e si calcola il checksum.

4°.- Tempo di trasferimento fra il ricevitore e l'applicazione dell'utente.

All'utente vengono inviati solo i byte utili e il byte di indirizzo del trasmettitore (Dt), in totale 17 byte, che a 9.600 baude impiegano 17 ms. ad essere trasferiti.

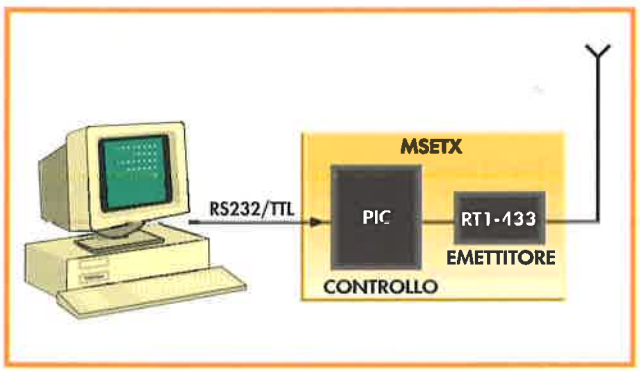
TEMPO TOTALE

Tempo di invio di una trama tramite R.F. 100 ms

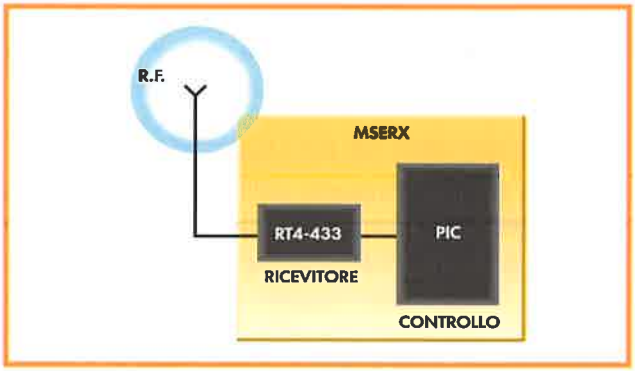
Tempo di invio del ricevitore all'utente 17 ms

Tempo in calcoli intermedi 3 ms

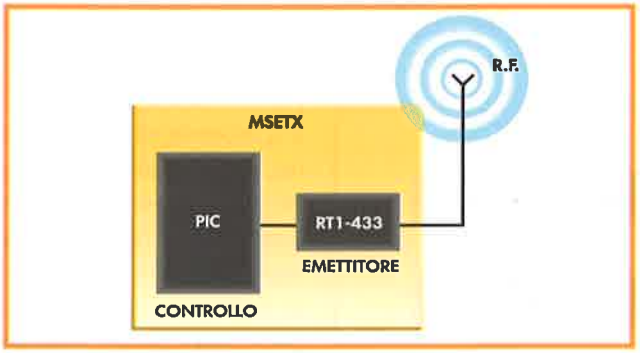
TOTALE: 120 MS



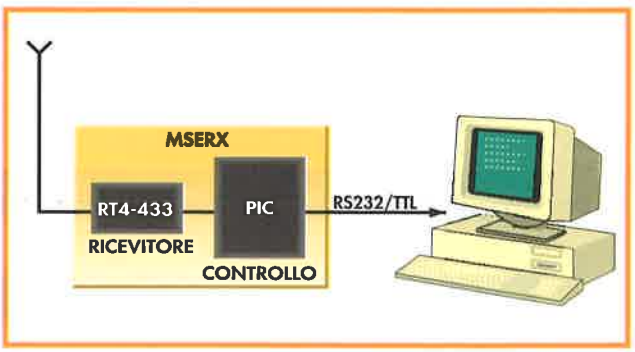
Il tempo di trasferimento fra l'utente e l'emettitore è di 17 ms.



La ricezione tramite R.F. della trama impiega gli stessi 100 ms che dura la trasmissione.



Il tempo di emissione di una trama via R.F. è di circa 100 ms.



Il trasferimento di una trama tra il ricevitore e l'applicazione dell'utente a 9.600 baude dura 17 ms.

Invasione di microcontroller con memoria flash

L'ORIGINE DEL SUCCESSO DEL PIC16F84

Il nostro vecchio amico, il PIC16F84, è stato scelto per controllare Monty, ed è utilizzato anche in molti prodotti sviluppati in tutto il mondo.

Risulta un dispositivo "piccolo, povero, però essenziale". Non ha molte risorse, iniziando dalla capacità della sua memoria, passando per i temporizzatori, terminando con lo scarso numero di pin I/O digitali. Tutto questo spiega l'appellativo di piccolo. Quello di povero fa riferimento alle risorse abituali in altri microcontroller. Così ad esempio non ha convertitori AD, né porte di comunicazione.

Però tutto si risolve con il fatto che è "essenziale" perché i progettisti fanno addirittura miracoli con le sue limitate possibilità. Le altre caratteristiche di spicco del PIC16F84 si riferiscono al suo basso costo e alla sua semplicità di funzionamento e di programmazione.

Non dà problemi e sia il produttore che molte ditte che lo applicano offrono una enorme quantità di informazioni e di strumenti per il suo utilizzo. Però, che cos'ha di speciale questo PIC? La sua caratteristica principale è la memoria di programma di tipo FLASH: questa è la ragione per cui la Arizona Microchip ha deciso di potenziare la linea di prodotti basata sulla tecnologia FLASH.



L'umile PIC16F84
si impiega in migliaia di prodotti in tutto il mondo.

I VANTAGGI DELLA TECNOLOGIA FLASH

La "flessibilità" è la parola chiave che definisce questa tecnologia e le sue principali caratteristiche sono esposte nella tabella.

CARATTERISTICHE DELLA TECNOLOGIA FLASH
1°. **Facilità della** cancellazione elettrica dell'informazione contenuta, in comparazione con le memorie EEPROM.
2°. Attrezzatura e software per realizzare le cancellazioni minime.
3°. **Basso** costo degli strumenti di sviluppo.
4°. La scheda finale contiene il microcontroller montato e il programma si scrive alla fine della costruzione della scheda.
5°. Si facilita **la messa a punto del** codice **nel chip** stesso, **già** installato sopra il prodotto.
6°. Vantaggioso rispetto ai chip OTP (che si programmano una sola volta), perché sono economici e si possono scrivere e cancellare più di 1.000 volte.
7°. Si può sempre inserire **nel prodotto l'ultima versione** del firmware sviluppato.
8°. La programmazione è ammessa per un vasto range di tensioni.
9°. Il cliente finale può riscrivere il codice.
10°. Si producono programmi di manutenzione e rilevazione degli errori che possono essere controllati a distanza.
11°. Non esiste una differenza apprezzabile fra il costo delle memorie FLASH, OTP e ROM.

Nello sforzo per migliorare le specifiche e il comportamento dei circuiti integrati con questa tecnologia, Microchip ha ottenuto all'inizio del secolo XXI delle elevate prestazioni, passando da una produzione di 0,9 micron ad una di 0,5 micron: così si riduce considerevolmente il tempo di scrittura per ciclo dai 10 ms per parola a 1-2 ms per blocco di programma, che a se-

MICROCONTROLLER

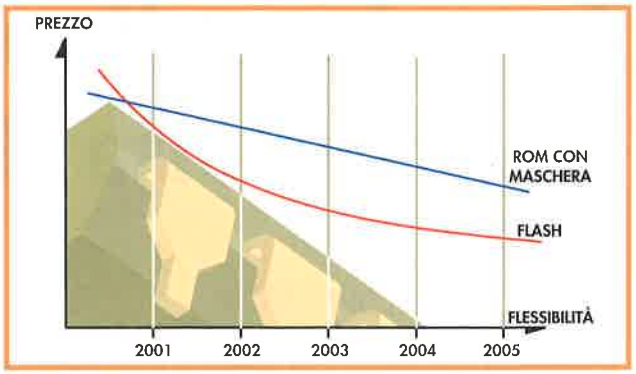
Il successo della memoria flash

conda dei modelli può contenere 8 K parole, 32 o solo una. In altre parole, nei moderni microcontroller PIC con memoria FLASH, si può scegliere fra cancellare e scrivere tutta la memoria, un blocco di diverse parole, o una sola parola. In questo modo si arriva a gestire programmi fino a 64 K parole in meno di 2 secondi.

Inoltre la capacità di autoprogrammazione di questi nuovi PIC permette di realizzare cambi e modifiche del programma a distanza. Sotto questo aspetto, sia Microchip che diversi costruttori OEM, come Ingeniería de Microsistemas Programados S.L., dispongono già di chip e di strumenti focalizzati a potenziare l'autocaricamento dei programmi, e la messa a punto in tempo reale. Nella tabella si mostrano i dati riguardanti il comportamento e le caratteristiche rilevanti delle memorie FLASH classiche, e quelle migliorate con la pompa di carica. Alla fine del 2001 si è arrivati all'equiparazione delle memorie FLASH migliorate, con le memorie ROM con maschera, rispetto al rapporto PREZZO/FLESSIBILITA' così come si può vedere graficamente nella figura. Il nuovo processo di produzione della memoria FLASH con tecnologia a 0,5 micron, non solo riduce il costo dei microcontroller, sino a permettere l'integrazione congiunta della memoria EEPROM dei dati, ma ha portato le capacità di queste fino a 128 Kbyte, e ridotto le dimensioni delle celle di informazione a un solo transistor.

CARATTERISTICA	MEMORIA FLASH VECCHIA	MEMORIA FLASH MIGLIORATA
ALIMENTAZIONE Vdd	2-5,5 VDC	2-5,5 VDC
TENSIONE DI PROGRAMMAZIONE	12,5-13,5 Esterna 5V Vdd	LO STESSO RANGE DI Vdd
AUTOPROGRAMMABILE	NO	SI
DEBUG IN CIRCUITO	NO	SI
TECNOLOGIA	0,9µm-0,7µm	0,5µm
CICLI DI CANCELLAZIONE/SCRITTURA	100 (Programma)	100k (Dati) 1k (Programma)
EEPROM DATI	NO	SI
TEMPO DI PROGRAMMAZIONE PER CICLO	10ms	1-2ms

Tabella comparativa delle caratteristiche della tecnologia FLASH vecchia e attuale, migliorata e dotata di pompa di carica.



Andamento dell'evoluzione della memoria FLASH migliorata, e della memoria ROM con maschera.

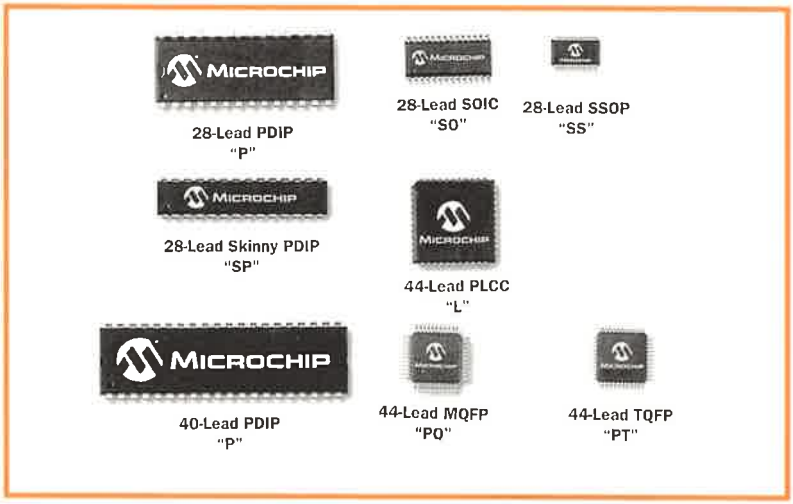
PICmicro MICROCONTROLLER FAMILY PRODUCTS

Product	Program Memory Bytes	Program Memory OTP/FLASH Words	Program Memory ROM Words	EEPROM Data Memory Bytes	RAM Bytes	I/O Ports	Packages	Analog 8-Bit ADC Channels	Analog Comparators	Digital PWM 10-Bit	Digital Timers/WDT	Digital Serial I/O	MAX Speed MHz	ICSP	BOR/ PBOR	LVD/ PLVD	CCP/ ECCP	Other Features
PIC16&CXXX: 4-12 Interrupts, 200ns Instruction Execution, 35 Instruction, 4/5 Oscillator Selections, Upward Compatible with PIC16C5X/PIC12CXXX (continued)																		
PIC16F84A	1792 (FLASH)	1024x14 (FLASH)	-	64	68	13	18P,18SO,20SS	-	-	-	1-8bit,1-WDT	-	20	✓	-	-	-	25mA source/sink per I/O
PIC16F870	3584 (FLASH)	2048x14 (FLASH)	-	64	128	22	28SP,28SO,28SS	5 (10-bit)	-	1	1-16bit,2-8 bit 1-WDT	USART	20	✓	✓	-	1	25mA source/sink per I/O, 9-bit
PIC16F871	3584 (FLASH)	2048x14 (FLASH)	-	64	128	33	40P,44L,44PT	8 (10-bit)	-	1	1-16bit,2-8 bit 1-WDT	USART	20	✓	✓	-	1	25mA source/sink per I/O, 9-bit,PSP
PIC16F872	3584 (FLASH)	2048x14 (FLASH)	-	64	128	22	28SP,28SO,28SS	5 (10-bit)	-	1	1-16bit,2-8 bit 1-WDT	MI²C/SPI	20	✓	✓	-	1	ICD, Self-Programing
PIC16F873	7168 (FLASH)	4096x14 (FLASH)	-	128	192	22	28SP,28SQ	5 (10-bit)	-	2	1-16bit,2-8 bit 1-WDT	USART MI²C/SPI	20	✓	✓	-	1	25mA source/sink per I/O,ICD, Self Programing, 9-bit
PIC16F874	7168 (FLASH)	4096x14 (FLASH)	-	128	192	33	40P,44L 44PQ,44PT	8 (10-bit)	-	2	1-16bit,2-8 bit 1-WDT	USART MI²C/SPI	20	✓	✓	-	1	25mA source/sink per I/O,ICD, PSP Self-Programing, 9-bit
PIC16F876	14336 (FLASH)	8192x14 (FLASH)	-	256	368	22	28SP,28SQ	5 (10-bit)	-	2	1-16bit,2-8 bit 1-WDT	USART MI²C/SPI	20	✓	✓	-	1	25mA source/sink per I/O,ICD, Self-Programing, 9-bit
PIC16F877	14336 (FLASH)	8192x14 (FLASH)	-	256	368	33	40P,44L 44PQ,44PT	8 (10-bit)	-	2	1-16bit,2-8 bit 1-WDT	USART MI²C/SPI	20	✓	✓	-	1	25mA source/sink per I/O,ICD, PSP, Self-Programing, 9-bit

Nuovi microcontroller PIC a 8 bit fabbricati da Microchip con tecnologia FLASH.

I FRATELLI MAGGIORI DEL PIC16F84

Una volta provata la tecnologia FLASH con il PIC16F84, gli utenti hanno richiesto con forza modelli più potenti e poderosi, con cui poter sviluppare progetti più ambiziosi. La regola fondamentale nello sviluppo di un progetto basato su microcontroller, è che la soluzione completa del sistema sia contenuta in un chip. Per questo al PIC16F84 bisognava aumentare le capacità di memoria, aggiungere dispositivi esterni, includere un convertitore AD, delle porte di comunicazione, e aumentare il numero delle linee di I/O digitali. Con questa finalità Microchip ha messo in vendita alla fine del millennio una famiglia di modelli di microcontroller con tecnologia FLASH, ai quali auguriamo un successo maggiore di quello conseguito con il conosciuto PIC16F84. Per chiarire alcune caratteristiche di questi nuovi PIC presentiamo una tabella, presa dal catalogo di Microchip. Come si può vedere dall'analisi della tabella, il PIC16F84 rimane all'ultimo posto in quanto a capacità e risorse. Tra le caratteristiche più rappresentative, che riflettono la differenza tra questo PIC e i nuovi modelli, le più importanti sono rappresentate nella tabella della figura. Da questa tabella si apprende che i fratelli maggiori del PIC16F84 hanno molta più capacità di memoria, molte più risorse e più piedini. Esistono versioni da 28 e da 40 pin. Per quanto riguarda la memoria FLASH, lo superano tutti, dato che hanno il doppio di capacità (2 K), il quadruplo (4 K) ed esistono alcuni modelli che arrivano fino a 8 K parole di 14 bit ognuna. Non dobbiamo dimenticare che questi PIC appartengono alla gamma media e utilizzano lo stesso formato e repertorio di istruzioni del PIC16F84. Nella figura si possono vedere i contenitori più utilizzati in cui sono introdotti questi nuovi PIC.



Contenitori tipici dei nuovi PIC con memoria FLASH.

MICROCONTROLLER	MEMORIA PROGRAMMA FLASH	MEMORIA DEI DATI EEPROM	MEMORIA DEI DATI RAM	LINEE I/O	PIN CONTENITORE
PIC16F84	1k x 14	64 x 8	68 x 8	13	18
PIC16F870	2k x 14	64 x 8	128 x 8	22	28
PIC16F871	2k x 14	64 x 8	128 x 8	33	40
PIC16F872	2k x 14	64 x 8	128 x 8	22	28
PIC16F873	4k x 14	128 x 8	192 x 8	22	28
PIC16F874	4k x 14	128 x 8	192 x 8	33	40
PIC16F876	8k x 14	256 x 8	368 x 8	22	28
PIC16F877	8k x 14	256 x 8	368 x 8	33	40

Principali differenze fra il PIC16F84 e i suoi fratelli maggiori.

I COMPORTAMENTI DELLE RISORSE AUSILIARIE

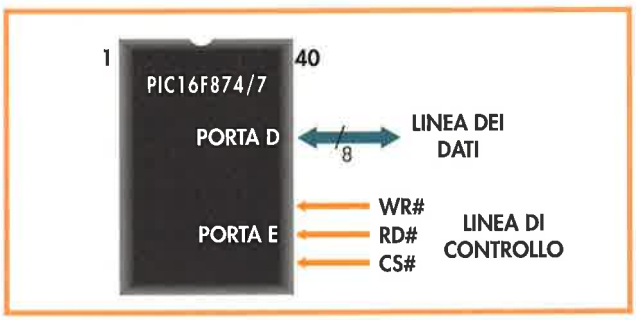
In un microcontroller, tanto importante quanto la capacità di memoria, è il numero

MICROCONTROLLER	CONVERTITORE A/D (10 BIT) CANALI	PWM 10BIT	TEMPORIZZATORI	PORTA SERIALE	CCP	PORTA PARALLELA
PIC16F84	-	-	TMR0-8BITS	-	-	-
PIC16F870	5	1	TMR0-8 TMR1-16 TMR2-8	USART	1	-
PIC16F871	8	1	TMR0-8 TMR1-16 TMR2-8	USART	1	PSP
PIC16F872	5	1	TMR0-8 TMR1-16 TMR2-8	MI^{2}C/SPI	1	-
PIC16F873	5	2	TMR0-8 TMR1-16 TMR2-8	USART MI^{2}C/SPI	1	-
PIC16F874	8	2	TMR0-8 TMR1-16 TMR2-8	USART MI^{2}C/SPI	1	PSP
PIC16F876	5	2	TMR0-8 TMR1-16 TMR2-8	USART MI^{2}C/SPI	1	-
PIC16F877	8	2	TMR0-8 TMR1-16 TMR2-8	USART MI^{2}C/SPI	1	PSP

Dotazione delle risorse ausiliarie nei nuovi PIC.

di linee di I/O e la quantità e la potenza delle risorse ausiliarie che realizzano i lavori specifici e scaricano di lavoro il processore. Ricordiamo le seguenti:

1°. Convertitore AD. I convertitori AD dei PIC FLASH sono da 10 bit e a diversi canali.
2°. PWM (modulazione di ampiezza di impulso).
3°. Temporizzatore (TMR).
4°. Porte di comunicazione seriali (USART, I2C e SPI).
5°. Modulo di Campionamento e Comparazione (CCP).
6°. Porta di Comunicazione Parallela (PSP).

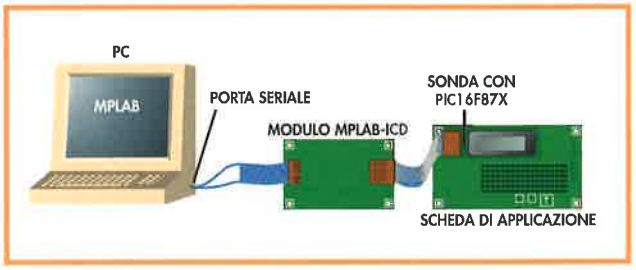


La porta di comunicazione parallela è supportata solo dai PIC-FLASH da 40 pin.

Nella tabella della figura riprodotta nella pagina precedente in basso, sono riportate le risorse ausiliarie dei nuovi PIC. La porta di comunicazione parallela è supportata solo dai nuovi PIC a 40 piedini, dato che per la sua implementazione si richiedono 11 linee di I/O, delle quali 8 sono destinate a supportare le 8 linee di trasferimento bidirezionale, e tre per il segnale di controllo (WR#, RD#, e CS#). I PIC FLASH a 40 pin destinano le porte D e E alle porte parallele. Nella figura sono riportate le implementazioni di questa porta di comunicazione con i PIC16F874/7 a 40 pin.

STRUMENTI PER LAVORARE CON I PIC16F87X

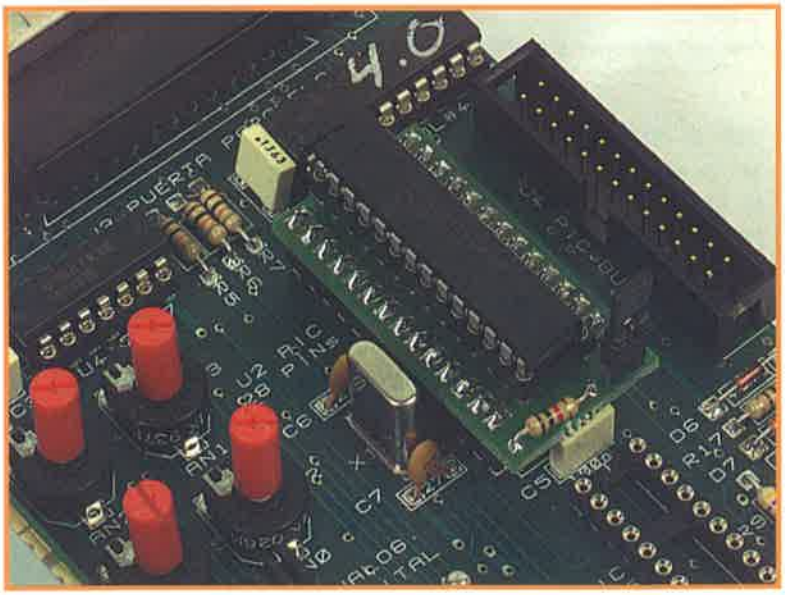
Microchip ha tenuto in grande considerazione il fatto di accompagnare i nuovi PIC FLASH con potenti strumenti che facilitano lo sviluppo dei progetti. Per ottenere ciò ha commercializzato il kit MPLAB-ICD, adattabile all'ambiente MPLAB, che già conosciamo e che utilizziamo in questa opera. Il kit è caratterizzato dalle seguenti prestazioni:



Il kit MPLAB-ICD di Microchip permette di lavorare con i PIC16F87X sul prodotto finito.

1°. Incorpora un programmatore o scrittore di PIC16F87X.
2°. Strumenti per la messa a punto del microcontroller collocato sopra il circuito di applicazione.
3°. Esecuzione in tempo reale con punti di fermata.
4°. Esecuzione passo a passo.
5°. Funzionamento con range di tensione 3-5,5 VDC e con range di frequenza 20-32 MHz.

La possibilità di scrivere un programma nei PIC FLASH montati sul prodotto finale riduce considerevolmente il costo del prodotto, diminuisce il tempo di realizzazione, rende possibile la calibrazione finale del prodotto e abilita l'inclusione di codici di identificazione sul prodotto terminato. Per poter lavorare con il PIC16F87X mediante il sistema di sviluppo MicroPIC Trainer, il suo costruttore ha progettato un economico adattatore, denominato ZOC87X, che si inserisce su uno degli zoccoli e permette di accoppiare, scrivere e sviluppare progetti con nuovi PIC.



Per lavorare con i PIC16F87X sul sistema di sviluppo MicroPIC Trainer, bisogna utilizzare lo zoccolo adattatore ZOC87X, che è mostrato nella fotografia.

I quattro moschettieri PIC-FLASH

MICROCONTROLLER	Memoria di programma (parole da 14 bit)	Memoria dati RAM (byte)	Memoria EEPROM (byte)	Linee I/O	Porte	Piedini
PIC16F873	4 k	192	128	22	3	28
PIC16F874	4 k	192	128	33	5	40
PIC16F876	8 k	386	256	22	3	28
PIC16F877	8 k	386	256	33	5	40

Differenze rilevanti fra i 4 modelli di PIC16F87X.

SCELTA DEI QUATTRO MIGLIORI

Considerata la somiglianza in termini di hardware e software che hanno i nuovi modelli di PIC con memoria FLASH, che potenziano la linea di lavoro del noto PIC16F84, per descriverli abbiamo scelto i quattro che coprono la maggioranza delle aspettative commerciali e tecniche. Si tratta dei modelli PIC16873, PIC16F874, PIC16F876 e PIC16F877. Due di essi hanno 28 pin e gli altri due 40. In ogni modello con la stessa piedinatura ne esiste uno con doppia capacità di memoria dell'altro: la differenza più visibile ricade nel maggior numero di linee di I/O che possiedono quelli da 40 pin, dato che dispongono di cinque porte: A, B, C, D, E. Quelli da 28 pin hanno solo tre porte A, B, C. Nella tabella della figura si mostrano le caratteristiche che differenziano i quattro modelli. Analizzando la tabella comparativa, si deduce che la principale differenza fra i 4 modelli da 28 e da 40 pin è quella relativa al numero di linee di I/O. Questo dato può essere importante in alcune applicazioni che necessitano di gestire molte periferiche diverse. Inoltre, i PIC16F874 e 877 dispongono di un dispositivo extra: la Porta Parallela Slave (PSP), che si configura con le otto linee di I/O della porta E che supporta le linee bidirezionali dei dati, e le tre linee della porta D, che servono per implementare i tre segnali di controllo: WR#, RD# e CS#.

Come abbiamo già spiegato, i vantaggi dei PIC16F87X consistono non solo nella maggiore capacità di memoria, ma anche nella dotazione di abbondanti risorse ausiliarie che facilitano moltissimo le applicazioni più esigenti. Un convertitore AD con diversi canali, tre temporizzatori, porte di comunicazione seriali tipo USART, I2C e SPI, porta di comunicazione parallela, PWM, moduli di captur e comparazione e sino a 14 possibili sorgenti di interrupt, sono alcuni degli elementi che sono stati integrati nel chip dei PIC16F87X e vengono presentati nella tabella seguente, che vuole sottolineare i vantaggi di questi moderni microcontroller.

MODELLO		PIC16F84A	PIC16F873	PIC16F874	PIC16F876	PIC16F877
MEM PROG (FLASH)	Bytes	1792	7168	7168	14336	14336
	Parole	1024x14	4096x14	4096x14	8192x14	8192x14
MEM DATI	Bytes EEPROM	64	128	128	256	256
	Bytes RAM	68	192	192	368	368
C A/D: Canali		No	5 (10 bits)	8 (10 bits)	5 (10 bits)	8 (10 bits)
BOD (Rivelazione di bassa tensione)		No	Si	Si	Si	Si
Linee I/O		13	22	33	22	33
COMUNICAZIONE SERIALE		No	USART/ MSSP	USART/ MSSP	USART/ MSSP	USART/ MSSP
CCP		No	2	2	2	2
TEMPORIZZATORE		1-8 bit, 1-WDT	1-16 bit, 2-8BIT, 1-WDT	1-16 bit, 2-8BIT, 1-WDT	1-16 bit, 2-8BIT, 1-WDT	1-16 bit, 2-8BIT, 1-WDT
FREQ. MAX, MHz		20	20	20	20	20
ICSP (Programmazione seriale in circuiti)		Si	Si	Si	Si	Si
PIEDINATURA		18P, 18SO, 20SS	28P, 28SO	40P, 44L, 44PQ,44PT	28SP, 28SO	40P, 44L,44PQ,44PT
SORGENTI DI INTERRUPT		4	13	14	13	14
COMUNICAZIONE PARALLELA (PSP)		No	No	Si	No	Si

Tabella comparativa del PIC16F87X con il PIC16F84A.

MICROCONTROLLER

Presentazione dei PIC16F87X

Cercando di riassumere le prestazioni che accompagnano i quattro nuovi modelli, offriamo una descrizione di quelle più importanti e il loro funzionamento.

RISORSE FONDAMENTALI DEI PIC16F87X

1°. Architettura RISC avanzata, tipo Harvard.

2°. Insieme di 35 istruzioni da 14 bit ognuna.

3°. Tutte le istruzioni si eseguono in un ciclo di istruzione, meno quelle di salto che ne impiegano due.

4°. Frequenza massima di funzionamento di 20 MHz.

5°. Modelli con 4 e 8 K parole da 14 bit per memoria di codice tipo FLASH.

6°. Sino a 356 byte di memoria dei dati RAM.

7°. Sino a 256 byte di memoria dei dati EEPROM.

8°. Contenitore compatibile con i PIC16C73/4/677.

9°. Sino a 14 possibili sorgenti di interrupt.

10°. Stack con 8 livelli.

11°. Modi di indirizzamento degli operandi immediato, diretto e indiretto.

12°. Watchdog (WDT).

13°. Codice di protezione programmabile.

14°. Modo di riposo SLEEP a basso consumo.

15°. Programmazione della memoria in modo seriale, utilizzando solo due pin.

16°. Tensione di alimentazione compresa fra 2 e 5,5 VDC.

17°. Basso consumo. Meno di 2 mA a 5 VDC e 5 MHz.

In quanto alla disponibilità delle risorse ausiliarie presentiamo la seguente tabella, che le raccoglie ordinatamente.

DISPOSITIVI PERIFERICI AUSILIARI

1°. Tre temporizzatori:
TMR0 da 8 bit con predivisore da 8 bit.
TMR1 da 16 bit con predivisore.
TMR2 da 8 bit con predivisore e postdivisore.

2°. Un convertitore AD da 10 bit con vari canali di ingresso (da 5 a 8).

3°. Due moduli di captur e comparazione e di PWM (CCP).

4°. Porta seriale sincrona con SPI e I2C.

5°. USART.

6°. Porta Parallela Slave (PSP) nei modelli da 40 pin.

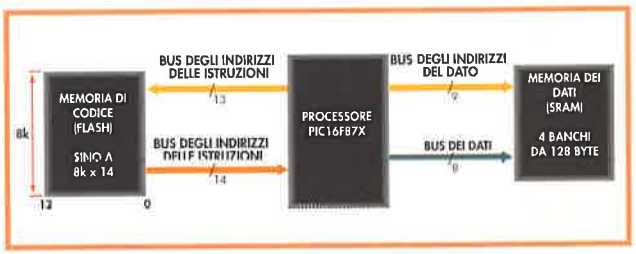
POTENZIAMENTO DELL'ARCHITETTURA HARVARD

Questa architettura utilizzata da Microchip nell'implementazione dei suoi microcontroller da 8 bit, è caratterizzata dall'indipendenza fra la memoria di codice e quella dei dati.

Con questa filosofia la capacità di ogni memoria, così come la dimensione dei suoi bus, si adatta alla necessità del progetto di ogni modello, rendendo possibile il lavoro in parallelo delle due memorie: questo si riflette in modo positivo sul rendimento e sulla produttività del processore. Nella figura è riportata la struttura dei PIC16F87X con le due memorie indipendenti.

Poiché quella di programma può arrivare ad una capacità di 8 K parole da 14 bit ognuna, necessita di un bus di indirizzo da 13 linee e di un bus per trasferire al processore le istruzioni da 14 linee, che è la dimensione di ogni istruzione.

La memoria dei dati SRAM accetta una capacità massima di 4 banchi da 128 byte ognuno, il che esige un bus di indirizzi da 9 linee, due delle quali sono destinate a selezionare il banco e le 7 restanti a selezionare una delle 128 posizioni di ogni banco.



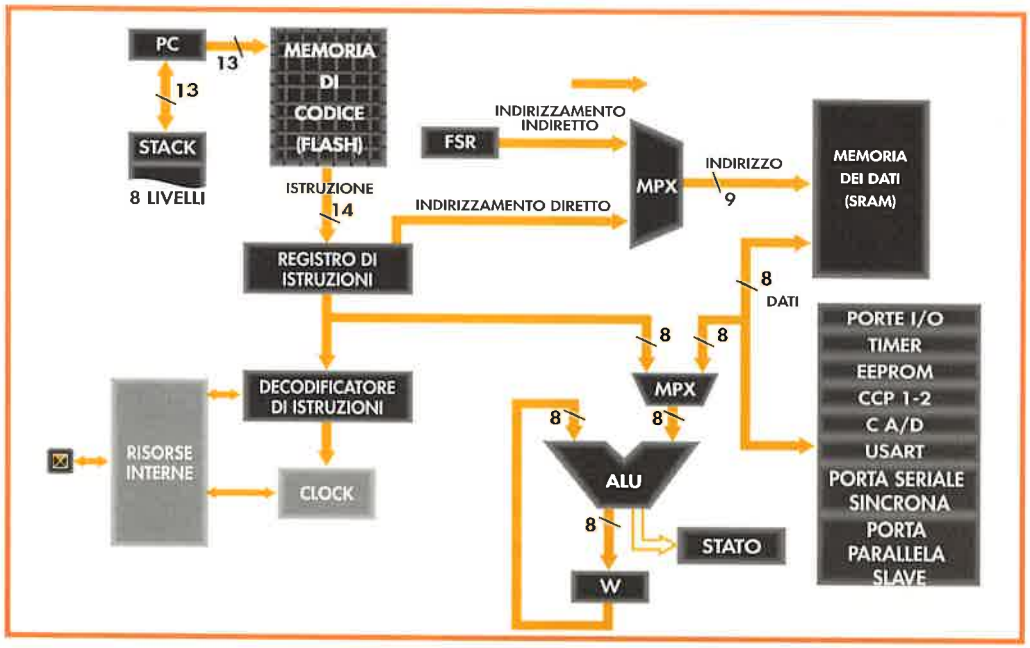
I bus delle memorie hanno la dimensione richiesta dalle caratteristiche specifiche di ognuna di esse.

Nella figura è rappresentato uno schema semplificato dell'architettura Harvard dei PIC16F87X, dove si può vedere che il contatore di programma è composto da 13 bit per indirizzare le memorie di codice tipo FLASH. Il contenuto del PC può essere salvato e recuperato nello stack che possiede 8 livelli.

La memoria dei dati, a seconda se si impiega l'indirizzamento diretto o indiretto, ottiene l'indirizzo dell'operando tramite l'informazione contenuta nella propria istruzione, o in quella che risiede rispettivamente nel registro FSR. La sezione della ALU, legata al registro di lavoro W e al registro di stato con i flag, è simile a quella studiata per il PIC16F84.

Le risorse interne e le diverse periferiche che accompagnano ogni modello di microcontroller sono gli ele-

menti di differenziazione. Infine si offre una figura che rappresenta lo schema completo dell'architettura interna di un PIC16F87X a 40 pin. Nei modelli con 28 pin sono state eliminate le porte D ed E. La filosofia di lavoro del processore è identica a quella studiata per il PIC16F84, rinforzata unicamente dalla capacità di memoria e dall'inclusione di diverse periferiche e risorse complementari.



Architettura Harvard semplificata dei PIC16F87X.

PIEDINATURA

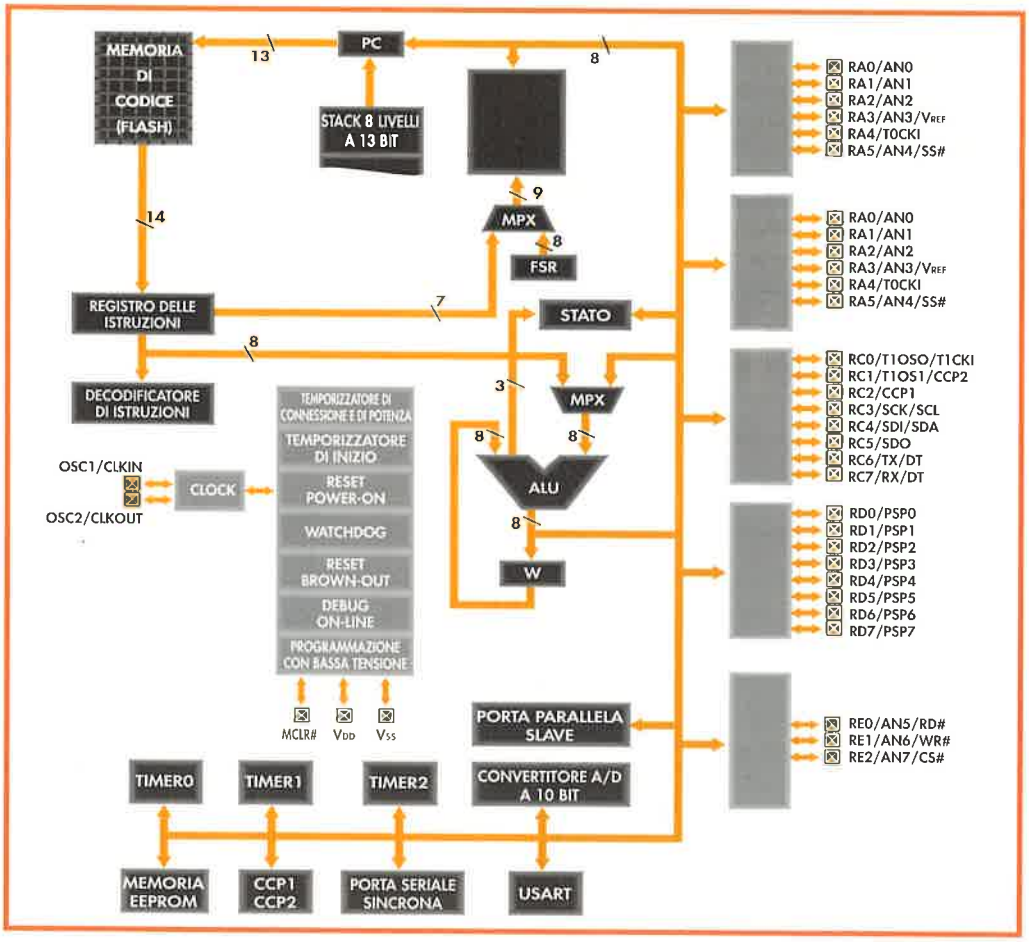
Nella figura della pagina successiva è riportata la distribuzione e l'assegnazione dei piedini nei modelli PIC16F87X.

Le piedinature con 28 pin mancano delle porte D ed E. In seguito riportiamo una descrizione sommaria delle funzioni e delle nomenclature di ognuno dei pin dei nuovi PIC-FLASH.

Piedini di utilizzo generale:

– OSC1/CLKIN e OSC2/CLKOUT: servono per supportare i collegamenti del cristallo di quarzo interno. Nel modo oscillatore R-C il pin CLKIN riceve la frequenza di lavoro e CLKOUT fornisce questa frequenza divisa per 4.

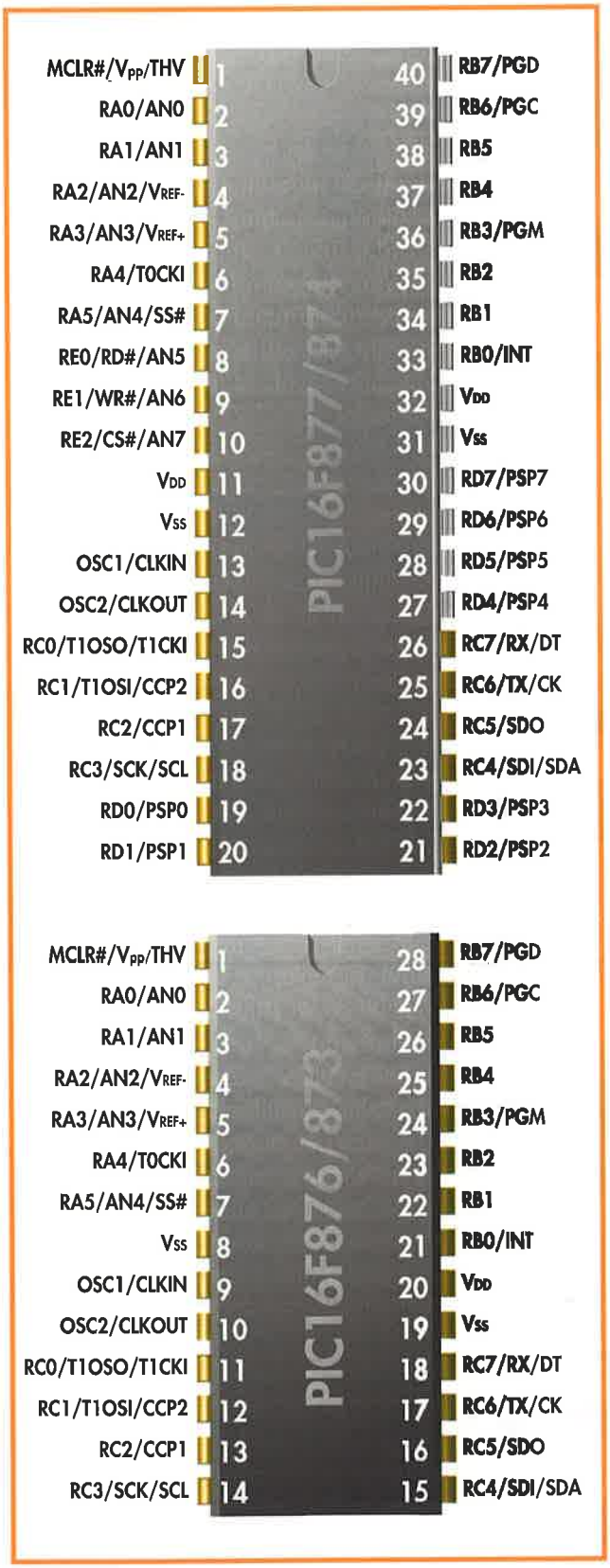
– Vss e VDD: piedi-



Architettura interna dettagliata dei PIC16F87X a 40 pin.

MICROCONTROLLER

Presentazione dei PIC16F87X



Piedinatura dei circuiti integrati corrispondenti ai PIC16F87X.

ni a cui si applica la tensione di alimentazione.

– MCLR#/VPP/THV: Ingresso di Reset/Tensione di Programmazione /Tensione alta in modo Test.

Piedini della porta A:
Supportano le 6 linee di I/O digitali, RA0-RA5, multiplexate con quelle corrispondenti ai canali di ingresso del convertitore AD AN0-AN4, con quelle della tensione positiva e negativa di riferimento VREF+ e VREF-, con quella di ingresso degli impulsi di clock per il TMR0, chiamata T0CKI, e con quella di selezione come slave (SS#) per la porta serie sincrona.

Piedini della porta B:
Supportano le 8 linee di I/O digitali, RB0-RB7, multiplexate con la linea PGM di ingresso della tensione bassa per la programmazione della memoria, e delle linee PGC e PGD per la programmazione seriale (segnale di clock e dei dati).

Piedini della porta C:
Supportano le 8 linee di I/O digitali, RC0-RC7, multiplexate con l'uscita del TMR1 (T1OSO), l'ingresso del clock TMR1 (T1CKI), l'ingresso dell'oscillatore TMR1 (T1OSI), l'ingresso del modulo CCP2, l'ingresso al CCP1, l'ingresso del clock sincrono (SCK/SCL) dei modi SPI e I2C.

Inoltre supportano le linee di ingresso dei dati in modo SPI (SDI) o quelle dei dati in modo I2C (SDA), l'uscita dei dati in modo SPI (SD0), la linea di trasmissione dell'USART (Tx) e quella del clock sincrono (CK), quella del ricevitore dell'USART (Rx) e la linea dei dati in modo sincrono (DT).

Piedini della porta D:
Oltre a supportare le 8 linee di I/O digitali, RD0-RD7, ha multiplexate sopra queste linee, quelle di trasferimento bidirezionali dei dati della Porta Parallela Slave (PSP0-PSP7).

Piedini della porta E:
Questa porta ha solo tre linee che funzionano come I/O digitali, RE0-RE2, e hanno multiplexate le seguenti funzioni: segnali di controllo della porta parallela PSP (RD#, WR# e CS#).

Inoltre possono anche funzionare come canali di ingresso per il convertitore AD nei modelli a 40 pin (AN5, 6 e 7).

Organizzazione delle memorie e repertorio delle istruzioni

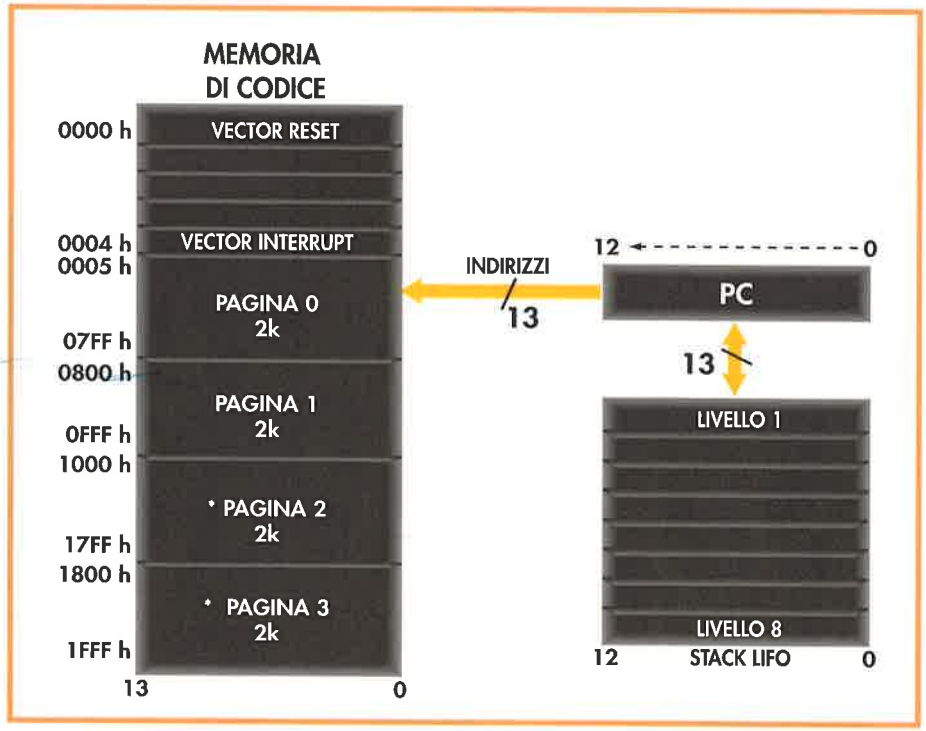
LA MEMORIA DI PROGRAMMA

L'architettura Harvard dei PIC16F87X determina l'esistenza delle due memorie indipendenti, una dedicata a contenere le istruzioni e l'altra per i dati. Questa caratteristica migliora notevolmente il rendimento del processore potendo accedere simultaneamente alle due memorie, di modo che mentre una localizza il codice di una istruzione, nell'altra si legge o si scrive un operando o il risultato di una operazione. Un altro vantaggio inerente alla separazione delle memorie, è collegato alla possibilità di progettare la capacità e la dimensione dei bus per ognuna di queste memorie. La memoria di programma dei PIC16F87X è di tipo FLASH, il che indica la sua capacità di tollerare fino a circa 1.000 cicli di scrittura /cancellazione. Dato che la scrittura e la cancellazione si realizzano mediante energia elettrica, non è necessario rimuovere dallo zoccolo il microcontroller per eseguire queste operazioni; tali caratteristiche hanno convinto molti tecnici ad utilizzare dispositivi con memoria FLASH. La capacità massima supportata dai quattro modelli di PIC16F87X che sono stati selezionati, è di 8 K parole da 14 bit ognuna. La lunghezza delle parole è legata alla dimensione del formato del codice delle istruzioni dei PIC della gamma media. Per poter indirizzare 8 K posizioni, il Contatore di Programma, o PC, ha una dimensione da 13 bit ed è legato a uno stack LIFO a 8 livelli. Il valore del PC si può scrivere nello stack e, parimenti, il primo livello dello stack può essere caricato nel PC. Queste operazioni di salvataggio e di carica del PC con lo stack si realizzano automaticamente senza l'intervento del programmatore. Non esistono istruzioni che gestiscono lo stack. Ogni volta che si esegue una istruzione CALL o si produce un interrupt, il valore del PC si salva automaticamente nello stack. Ogni volta che si esegue una istruzione di ritorno dalla subroutine, RETURN, o una istruzione di ritorno dall'interrupt, RETFIE, il valore che si ritrova in cima allo stack si deposita nel PC. C'è anche un'istruzione simile a queste ultime di ritorno, però con una particolarità, ed è la RETLW.

Come si vede nella figura, le 8 K parole da 14 bit ognuna che ammette come massimo la memoria FLASH dei PIC16F87X, sono divise in 4 pagine da 2 K l'una. Così come nel PIC16F84, il vector di Reset e quello di Interrupt si trovano rispettivamente negli indirizzi 0000 e 0004 Hex.

LA MEMORIA DEI DATI

Per memorizzare i dati e gli operandi, i PIC16F87X dispongono di due zone diverse. Esiste una zona di



Organizzazione interna e indirizzamento della memoria di programma FLASH.

MICROCONTROLLER

Memoria e istruzioni dei PIC16F87X

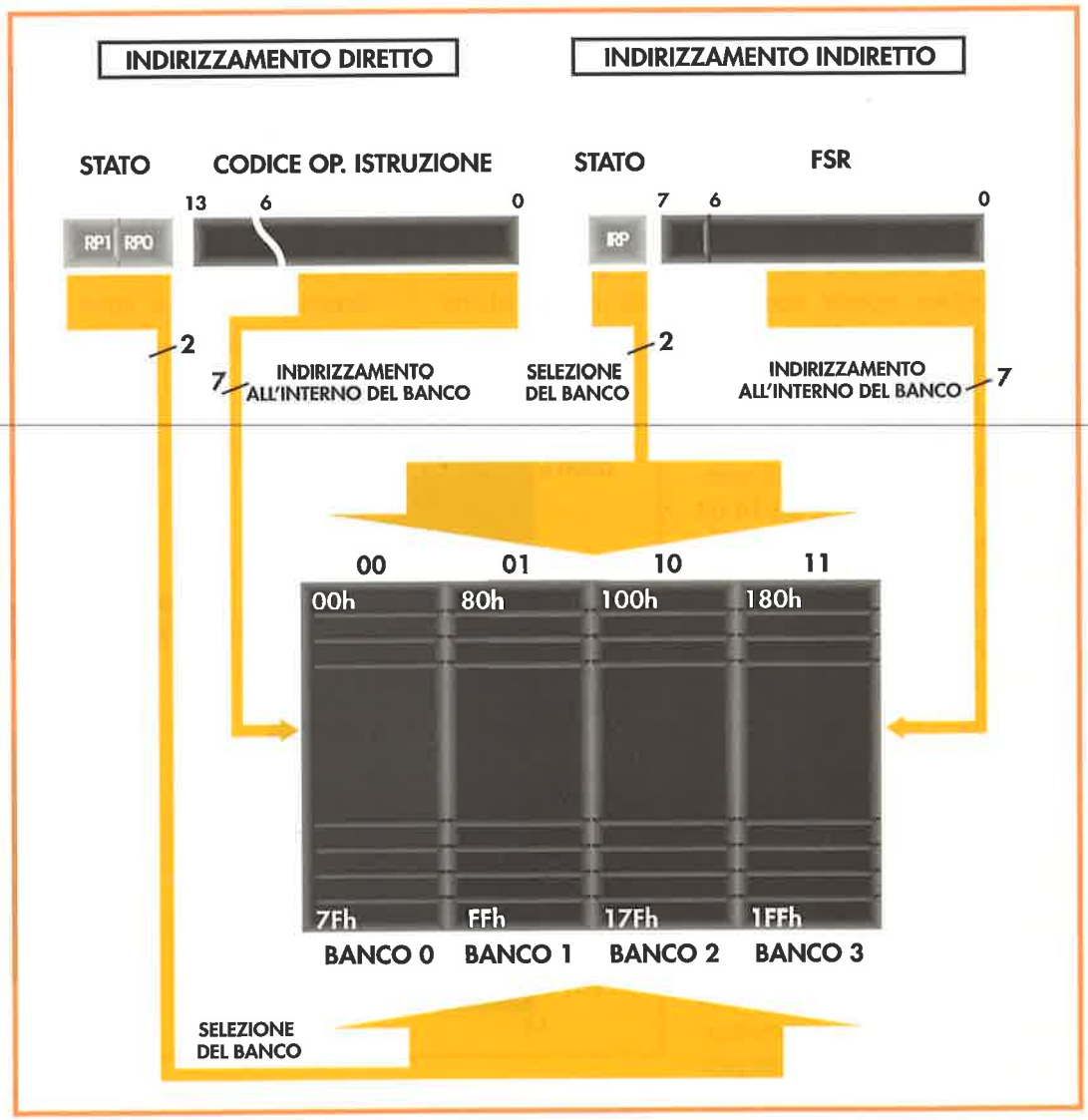


Struttura interna del Registro di Stato, i cui bit 6 e 5, RP1 e RP0, selezionano il banco della RAM dove sono posizionati gli indirizzi a cui accedere.

memoria dei dati non volatili, di tipo EEPROM e un'altra indipendente SRAM volatile. A seconda del modello di PIC la EEPROM può arrivare ad una capacità di 256 byte, che si gestiscono mediante dei registri appositi che contengono gli indirizzi, il dato e il bit di controllo. Il loro funzionamento e la loro gestione sono stati studiati nel PIC16F84, e sono identici per i suoi fratelli maggiori. Nella zona della memoria dei dati di tipo RAM, sono ubicati i registri specifici (SFR), i cui bit controllano il funzionamento delle risorse, i dispositivi del processore e i registri di utilizzo generale (GPR), dove si scrivono i dati che desidera il programmatore. La RAM si organizza in banchi da 128 byte ognuno. Ci sono due modelli di PIC16F87X che dispongono di 192 byte e gli altri due arrivano a 368 byte. Per selezionare uno dei quattro possibili banchi dove si trovano le posizioni a cui accedere, si utilizzano due bit: RP1 e RP0, che occupano le posizioni 6 e 5 del Registro di Stato, come si vede nella figura.

MODI DI INDIRIZZAMENTO

La memoria RAM ha due forme di indirizzamento, che sono chiamate "modo di indirizzamento" e che sono il modo diretto e il modo indiretto. Nel modo di indirizzamento diretto i bit RP1 e RP0 del Registro di Stato selezionano il banco. Per puntare ad una delle 128 posizioni del banco scelto, sono necessari 7 bit che arrivano dal codice delle istruzioni dove occupano le sette posizioni meno significative. Nell'indirizzamento indiretto la posizione del banco si determina con i sette bit meno significativi contenuti dal registro FSR, dell'area SFR



Modi di realizzare l'indirizzamento diretto e indiretto nella memoria RAM.

della RAM. Per trovare il banco in questo ultimo modo di indirizzamento, si utilizza il bit meno significativo del FSR insieme al bit IRP, che occupa anche la posizione meno significativa del Registro di Stato.

Entrambi i modi di indirizzamento sono spiegati graficamente nella figura. Per quanto riguarda la distribuzione interna degli indirizzi della RAM, com'è abituale nell'architettura dei microcontroller PIC, le prime posizioni di ogni banco sono destinate a contenere i registri specifici per il controllo delle risorse del processore, questa area si chiama SFR. Gli ultimi indirizzi di ogni banco sono dedicati a contenere i dati di utilizzo generale che necessitano al programmatore. Questa zona si chiama GPR. Come riportato nella figura, ci sono molti registri specifici nelle prime posizioni di ogni banco.

Molti di più che nel PIC16F84. Questo è logico perché i nuovi PIC dispongono di molte risorse (convertitore AD, CCP, porte di



Ubicazione dei registri specifici (SFR) e di utilizzo generale (GPR) nei PIC16F87X dei 368 byte della RAM.

molte risorse (convertitore AD, CCP, porte di

MICROCONTROLLER

Memoria e istruzioni dei PIC16F87X

MNEMONICO	PARAMETRI	OPERAZIONE	CICLI	FORMATO 14 BIT	FLAG
		ISTRUZIONI CHE GESTISCONO REGISTRI			
addwf	f,d	SOMMA W e f	1	00 0111 dfff ffff	C, DC, Z
andwf	f,d	AND di W e f	1	00 0101 dfff ffff	Z
clrf	f	CANCELLAZIONE di f	1	00 0001 1fff ffff	Z
clrw	æ	CANCELLAZIONE di W	1	00 0001 0xxx xxxx	Z
comf	f,d	COMPLEMENTO di f	1	00 1001 dfff ffff	Z
decf	f,d	DECREMENTO di f	1	00 0011 dfff ffff	Z
incf	f,d	INCREMENTO di f	1	00 1010 dfff ffff	Z
iorwf	f,d	OR di W con f	1	00 0100 dfff ffff	Z
movf	f,d	MOVIMENTO di f	1	00 1000 dfff ffff	Z
movwf	f	MOVIMENTO di W a f	1	00 0000 1fff ffff	
nop	æ	NESSUNA OPERAZIONE	1	00 0000 0xx0 0000	
rlf	f,d	ROTAZIONE di f a sinistra con carry	1	00 1101 dfff ffff	C
rrf	f,d	ROTAZIONE di f a destra con carry	1	00 1100 dfff ffff	C
subwf	f,d	SOTTRAZIONE di W a f (f - W)	1	00 0010 dfff ffff	C, DC, Z
swapf	f,d	INTERSCAMBIO dei 4 bit più significativi con i 4 meno significativi	1	00 1110 dfff ffff	
xorwf	f,d	EOR fra W e f	1	00 0110 dfff ffff	Z
		ISTRUZIONI CHE GESTISCONO BIT			
bcf	f,b	MESSA a 0 del bit b di f	1	01 00bb bfff ffff	
bsf	f,b	MESSA a 1 del bit b di f	1	01 01bb bfff ffff	
		ISTRUZIONI DI SALTO			
btfsc	f, b	TEST del bit b di f ; SALTA se 0	1 (2)	01 10bb bfff ffff	
btfss	f, b	TEST del bit b di f ; SALTA se 1	1 (2)	01 11bb bfff ffff	
decfsz	f, d	DECREMENTO di f ; SALTA se 0	1 (2)	00 1011 dfff ffff	
incfsz	f, d	INCREMENTO di f ; SALTA se 0	1 (2)	00 1111 dfff ffff	
		ISTRUZIONI CHE GESTISCONO OPERANDI IMMEDIATI			
addlw	k	SOMMA di un letterale con W	1	11 111x kkkk kkkk	C, DC, Z
andlw	k	AND di un letterale con W	1	11 1001 kkkk kkkk	Z
iorlw	k	OR di un letterale con W	1	11 1000 kkkk kkkk	Z
movlw	k	MOVIMENTO di un letterale a W	1	11 00xx kkkk kkkk	
sublw	k	SOTTRAZIONE di W da un letterale (k -W)	1	11 110x kkkk kkkk	C, DC, Z
xorlw	k	OR esclusivo di un letterale con W	1	11 1010 kkkk kkkk	Z
		ISTRUZIONI DI CONTROLLO E SPECIALI			
call	k	CHIAMATA a subroutine	2	10 0kkk kkkk kkkk	
clrwdt		CANCELLAZIONE del WATCHDOG	1	00 0000 0110 0100	#TO, #PD
goto	k	SALTO a un indirizzo	2	10 1kkk kkkk kkkk	
retfie		RITORNO da interrupt	2	00 0000 0000 1001	
retlw	k	RITORNO scrivendo un letterale in W	2	11 01xx kkkk kkkk	
return		RITORNO da subroutine	2	00 0000 0000 1000	
sleep		MESSA del microprocessore in riposo	1	00 0000 0110 0011	#TO, #PD

Repertorio di istruzioni dei PIC della gamma media, valido per i PIC16F87X.

comunicazione, diversi temporizzatori, ecc.) e ognuna ha bisogno di registri programmabili che ne controllino il funzionamento. Nella figura esiste una zona di registri nei banchi 1, 2 e 3 che è mappata nel banco 0 agli indirizzi 70-7F Hex. Significa che gli indirizzi di questi tre banchi corrispondono con quelli del banco 0, eseguendo un accesso ad essi è come se si realizzasse un accesso al registro corrispondente del banco 0.

REPERTORIO DELLE ISTRUZIONI

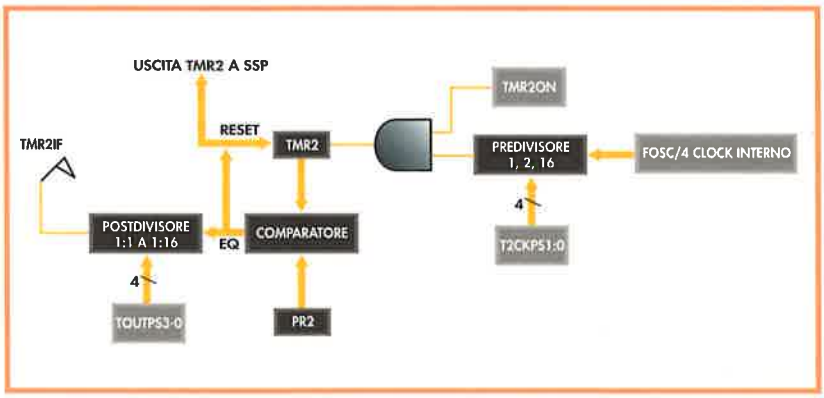
Il lettore conosce e ha utilizzato molte volte le 35 istruzioni corrispondenti ai PIC della gamma media, dato che le utilizza il PIC16F84 di cui è dotato Monty. Visto che i PIC16F87X appartengono alla gamma media dei PIC a 8 bit, supportano lo stesso insieme di istruzioni. Com'è possibile gestire con le stesse istruzioni un microcontrolloer così semplice come il PIC16F84 e altri così potenti e con molte risorse come i PIC16F87X? La differenza non sta nelle istruzioni, ma nei registri specifici di controllo dei dispositivi. Ognuna delle nuove risorse dispone nella zona SFR della RAM di registri adeguati per programmare il proprio comportamento. Inoltre i piedini dei PIC16F87X hanno multiplexate molte funzioni di ingresso, uscita, clock, ecc. di questi elementi. Nella figura è riportata una tabella con l'insieme delle istruzioni dei PIC della gamma media. Il gruppo che gestisce i registri, e il gruppo che gestisce i bit, sono quelli che permettono ai nuovi PIC di operare al meglio delle loro potenzialità, visto che con le loro istruzioni si possono programmare i registri e i bit specifici dei registri che gestiscono le risorse ausiliarie.

Gli altri tre gruppi di istruzioni, che contengono le istruzioni di salto, le istruzioni che gestiscono operandi immediati e le istruzioni speciali e di controllo, hanno praticamente lo stesso significato e la medesima applicazione in tutti i modelli di PIC della gamma media.

Questi sono i poteri dei PIC16F87X

Presentando questi PIC come i fratelli maggiori del PIC16F84, è stato detto che possiedono una capacità di memoria molto maggiore, però sono i dispositivi e le risorse che Microchip ha incorporato al loro interno quelli che forniscono la vera forza.

Anche se in quest'opera non si dispone di spazi sufficienti per descriverli in dettaglio, vogliamo fornire un'idea generale sul comportamento e le capacità degli elementi più interessanti che sono incorporati nei PIC16F87X.



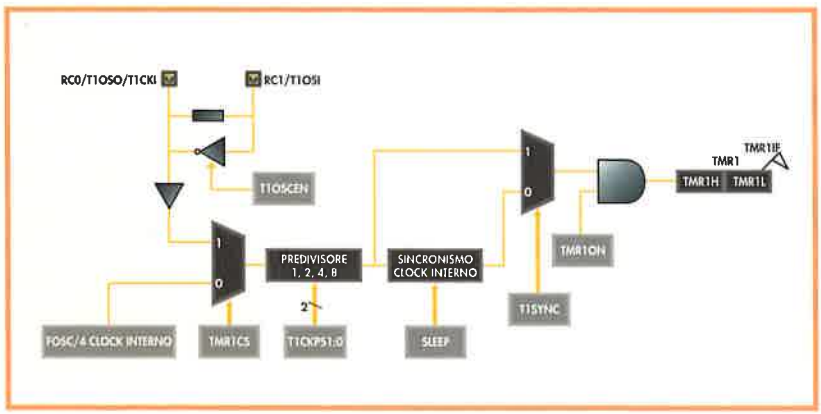
Struttura interna del TMR1 con i blocchi principali e i segnali di controllo.

I nuovi temporizzatori o Timer, i moduli di Capture, Comparazione e Generazione di Impulsi e di Ampiezza Variabile, il Convertitore AD, le Porte di Comunicazione Seriale, sono i dispositivi di cui faremo una breve analisi, senza approfondire il discorso sui registri di controllo, la cui corretta programmazione suppone la gestione dei medesimi.

IL TEMPORIZZATORE TMR1

Dei tre temporizzatori disponibili nei PIC16F87X, questo è l'unico che ha una dimensione di 16 bit e che funziona come un Temporizzatore/Contatore. Per gestire 16 bit è necessario concatenare due registri da 8 bit: TMR1 H:TMR1L, che sono incaricati di memorizzarne il contenuto in ogni istante. Questo valore evolve da 0000 sino a FFFF Hex e quando passa nuovamente sul valore 0000 si attiva il flag TMR1F e se si desidera si può provocare una richiesta di interrupt. Il valore presente in TMR1H:TMR1L può essere letto e scritto e gli impulsi di clock che originano il conteggio ascendente possono provenire dall'esterno o dalla frequenza di funzionamento del microcontroller (Fosc/4). Il TMR1 può funzionare in tre modi:

1°. Come temporizzatore.
2°. Come contatore sincrono.
3°. Come contatore asincrono.



Schema a blocchi del temporizzatore TMR2.

Nel modo temporizzatore il valore concatenato TMR1H:TMR1L si incrementa con ogni ciclo di istruzione (Fosc/4). Nel modo contatore l'incremento si può produrre con i fronti di salita di un clock, il cui ingresso è applicato alla linea RC0 e RC1 della porta C, o tramite gli impulsi applicati sulla linea RC0. Nella figura è riportato il diagramma a blocchi del TMR1, in cui possiamo vedere il predivisore di frequenza.

IL TEMPORIZZATORE TMR2

Questo potente PIC dispone anche del temporizzatore TMR0, però dato che il suo

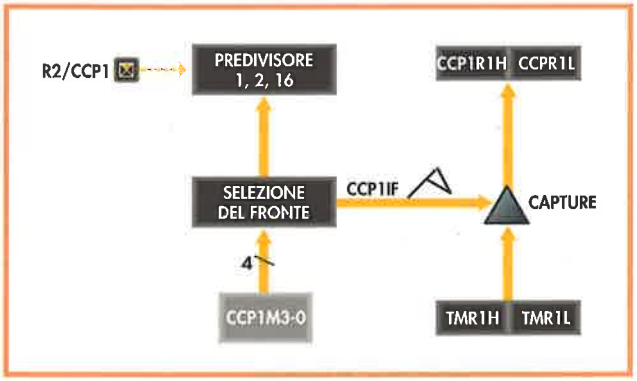
comportamento è simile a quello studiato nel PIC16F84 non lo prenderemo in considerazione e ci concentreremo sui due aggiunti: il TMR1 appena considerato e il TMR2.

Quest'ultimo è un temporizzatore ascendente a 8 bit, all'interno del quale si può leggere e scrivere, inoltre può realizzare operazioni speciali per la Porta Seriale Sincrona (SSP) e per i moduli di Capture e Comparazione.

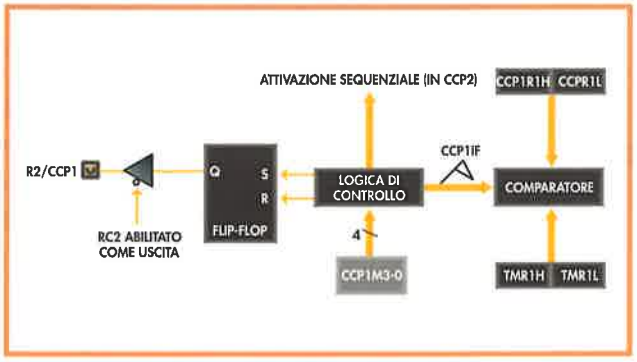
Il segnale di clock del TMR2 e quello interno Fosc/4, prima di essere applicato passa attraverso un prediviso-re di frequenza con range 1:1, 1:4 e 1:16. L'uscita del TMR2 attraversa un postdivisore di frequenza con range compresi da 1:1 a 1:16, passando per 16 valori possibili. Quando il microcontroller entra in modo riposo si ferma l'oscillatore interno e il TMR2 smette di funzionare.

I MODULI CCP IN MODO CAPTURE

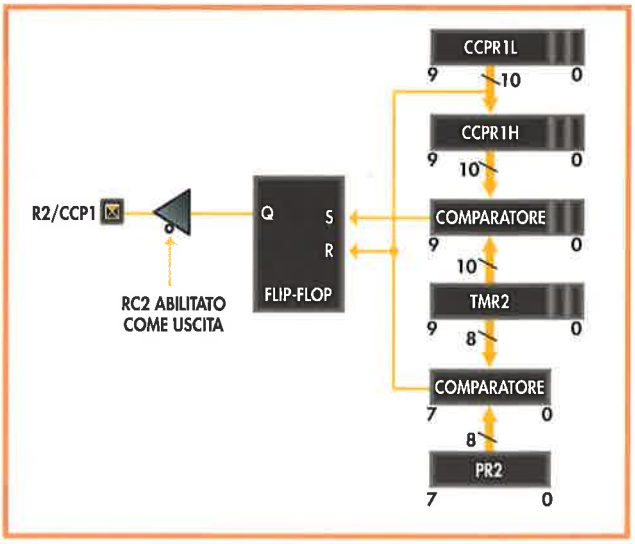
I PIC16F87X dispongono di due moduli CCP, che si differenziano fra loro solo per la nomenclatura, CCP1 e CCP2, visto che il loro funzionamento è praticamente uguale.



Schema generale a blocchi del modulo CCP1 in modo capture.



Schema a blocchi del modulo CCP1 quando funziona in modo comparazione.



Schema semplificato del modulo CCP1 quando funziona in modo PWM.

Quando funzionano in modo capture una coppia di registri cattura il valore che è presente sul TMR1 al verificarsi di un evento speciale sul piedino RC2/CCP1 per il modulo CCP1 e sul piedino RC1/T1OSI/CCP2 per il modulo CCP2. Gli eventi che possono accadere su questi piedini sono:

1°. Un fronte di salita.
2°. Un fronte di discesa.
3°. Ogni 4 fronti di salita.
4°. Ogni 16 fronti di discesa.

Effettuando il capture si attiva il flag CCP1F per il modulo CCP1, e se il bit di abilitazione è attivato si può anche originare una richiesta di interrupt. Un'interessante applicazione del modo capture è la misura degli intervalli di tempo esistenti fra gli impulsi che si applicano a un pin del PIC.

MODO COMPARAZIONE

In questo modo lavoro, la coppia di registri CCPR1H-L compara il suo contenuto, in modo continuo, con il valore del TMR1.

Quando coincidono entrambi i valori viene messo a 1 il flag CCP1IF, e il piedino RC2/CCP1 che deve essere configurato come uscita, supporta uno dei seguenti eventi:

1°. Passa a livello alto.
2°. Passa a livello basso.

3°. Non cambia di stato però si produce un interrupt.

MODO PWM

In questo modo lavoro si ottengono impulsi la cui ampiezza del livello alto è di durata variabile, molto utili per il controllo dei motori. Il piedino RC2/CCP1 è configurato come uscita, e passa dai livelli logici 0 e 1 a intervalli variabili di tempo.

Si vuole ottenere un impulso il cui livello alto abbia un'ampiezza variabile (Duty Cicle), all'interno di un intervallo fisso del periodo dell'impulso.

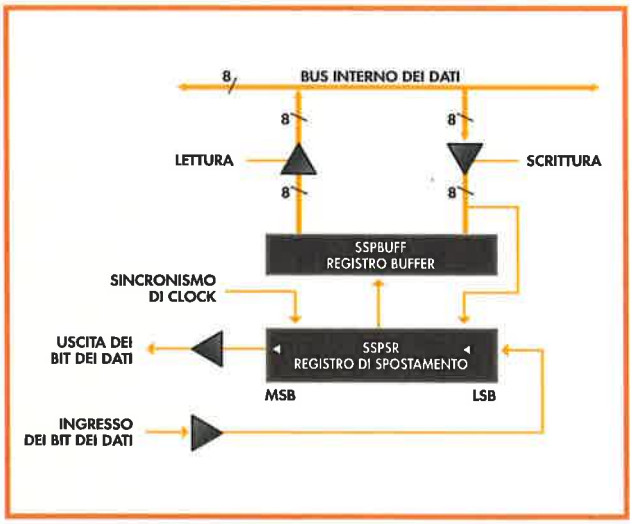
Per ottenere il cambiamento di stato del piedino, si utilizza un comparatore che pone a 1 (Set) un flip-flop quando il valore del registro PR2 coincide con la parte alta del TMR2, momento in cui TMR2 prende il valore 00 Hex.

Dopo il flip-flop si resetta, e si pone a 0 quando un altro comparatore rileva la coincidenza sul valore esistente nel CCPR1H con quello della parte alta del TMR2. Variando i valori che si caricano in PR2 e in CCPR1L si varia l'intervallo di tempo in cui il piedino di uscita sta a 1 e a 0.

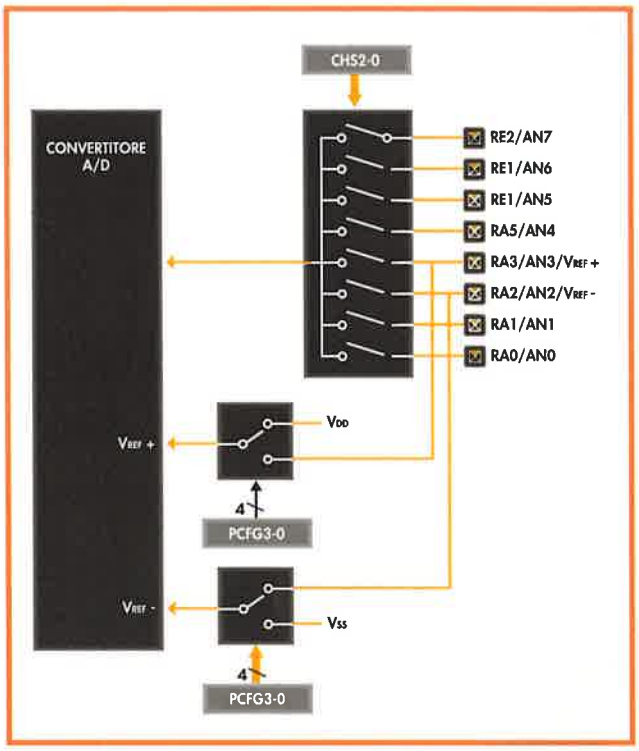
IL CONVERTITORE ANALOGICO-DIGITALE

I PIC16F87X dispongono di un convertitore AD da 10 bit di risoluzione e 5 canali di ingresso nei modelli da 28 pin, e 8 canali in quelli da 40 pin.

Attraverso uno degli ingressi selezionati si applica il segnale analogico ad un condensatore di cattura e



Struttura base del modulo MSSP con i registri di lavoro SSPBUF e SSPSR.



Struttura semplificata del collegamento del convertitore AD.

mantenimento, successivamente questo valore viene portato al convertitore, che utilizzando la tecnica dell'approssimazione successiva fornisce il risultato digitale equivalente su 10 bit.

La tensione di riferimento può essere generata con la tensione interna di alimentazione del PIC oppure con una esterna introdotta con i pin RA3/AN3/VREF+ e RA2/AN2/VREF-.

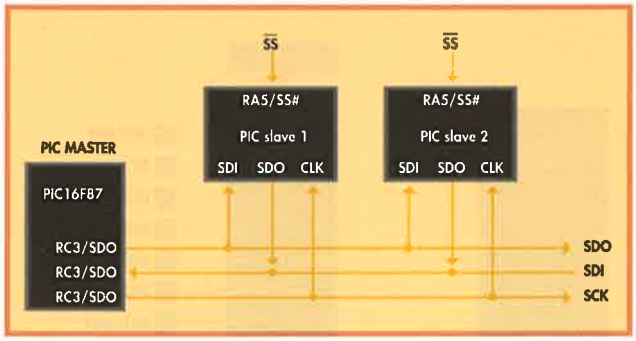
Nella figura è rappresentato uno schema semplificato dei collegamenti del convertitore AD.

PORTA DI COMUNICAZIONE SERIALE SINCRONA

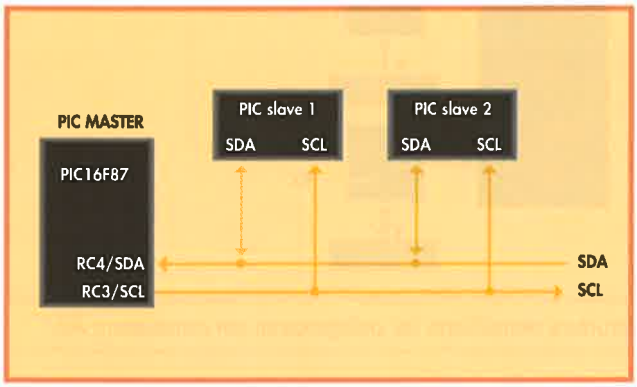
Questo modulo chiamato MSSP, integrato nei PIC16F87X fornisce un eccellente mezzo di comunicazione con gli altri microcontroller o con le periferiche che lavorano in modo seriale. Ha due modi possibili di lavoro:

1°. SPI (Serial Peripheral Interface).
2°. I2C (Inter-Integrated Circuit).

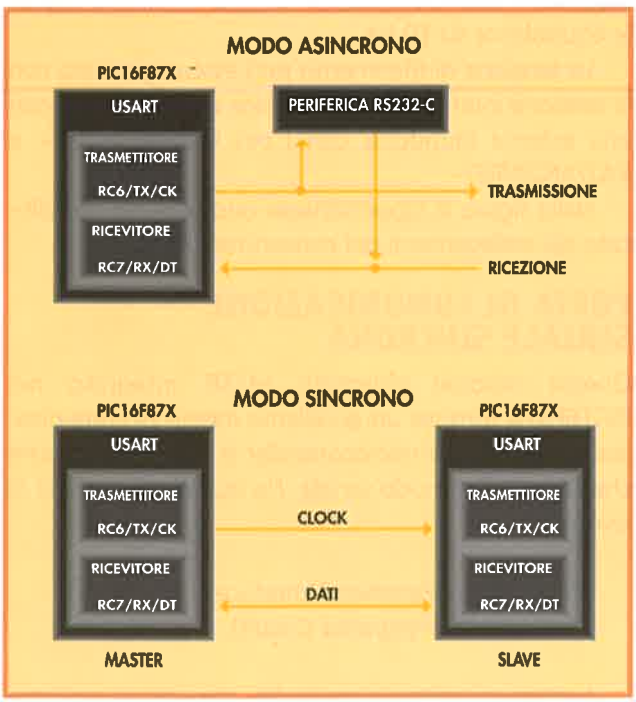
La comunicazione in modo SPI, che utilizza tre linee, viene usata in modo particolare dalle memorie RAM e EEPROM. Il protocollo I2C utilizza solo due linee, ed è



Comunicazione seriale in modo SPI tra un PIC Master e i due dispositivi che funzionano come slave.



Nel modo I2C il Master comunica con tutti gli slave tramite due sole linee.



Nel modo asincrono la comunicazione seriale dell'USART utilizza le linee TX e RX, mentre nel modo sincrono le linee DT e CK.

impiegato soprattutto nella comunicazione fra circuiti integrati diversi.

Fondamentalmente il modulo MSSP è basato su due registri: SSPSR, che è un registro di spostamento che trasporta l'informazione seriale in parallelo e viceversa, e il registro SSPBUF, che funziona come buffer dell'informazione che si riceve o si trasmette in serie. Nella trasmissione il byte che si desidera inviare è caricato nel SSPBUF e automaticamente viene spostato a SSPSR, dove si sposta bit a bit, portandolo all'esterno al ritmo degli impulsi di clock.

In ricezione i bit entrano al ritmo del clock tramite un pin del PIC e si spostano nel registro SSPSR sino al suo riempimento, momento nel quale si trasferisce l'informazione al SSPBUF.

Nel modo SPI si utilizzano tre linee del PIC che funziona come Master nello schema della figura, mentre nel modo I2C si utilizzano solo due linee per la comunicazione del PIC master con i circuiti integrati che funzionano come slave.

L'USART

L'USART supporta la comunicazione seriale sincrona e asincrona. Può funzionare come un sistema di comunicazione bidirezionale sincrono o full duplex, adattandosi a una moltitudine di dispositivi che trasferiscono informazioni in questo modo. Inoltre può lavorare nel modo unidirezionale o half duplex. Riassumendo può lavorare in tre modi:

1°. SERIALE ASINCRONO (Full duplex, bidirezionale).
2°. SERIALE SINCRONO-MASTER (Half duplex, unidirezionale).
3°. SERIALE SINCRONO-SLAVE (Half duplex, unidirezionale).

Nella figura è riportato lo schema del funzionamento dell'USART in modo asincrono e sincrono. Nel primo, i trasferimenti di informazione si realizzano su due linee TX e RX; i bit entrano ed escono da queste due linee a una frequenza controllata internamente dall'USART.

Nel modo sincrono, la comunicazione si realizza su due linee, la DT che trasla i bit in entrambi i versi alla frequenza degli impulsi del clock che escono dalla linea CK del master.

In entrambi i modi, le linee di comunicazione sono le due più significative della porta C: RC6/TX/CK e RC7/RX/DT.