Zeno Martini (admin)

ALCUNE RISPOSTE DI: PAOLINO, UN PIC PER AMICO

2 August 2010

Presentazione

Paolino, è un componente dello "zoccolo duro", se così posso dire, del Forum e del sito.

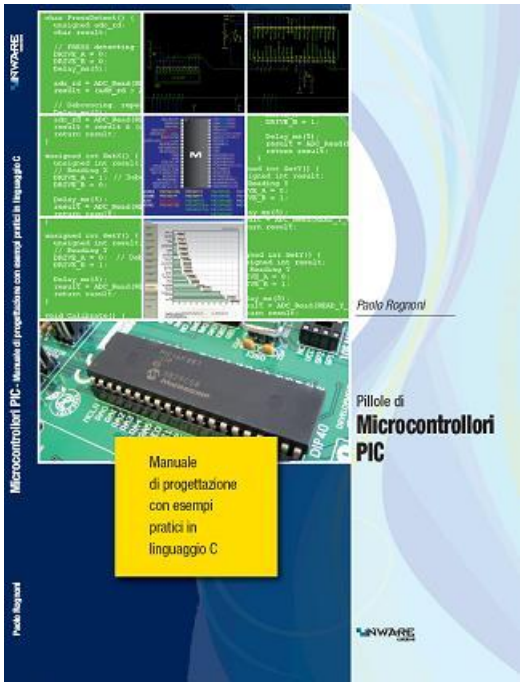
Ha sempre creduto in questa attività e, coerentemente, vi ha contribuito con costanza cercando di conferire al forum **serietà** e **spirito di collaborazione**.

Si occupa professionalmente di sviluppo di applicazioni dei PIC e costituisce, per i microcontrollori, il punto di riferimento del nostro Forum.

Numerosi sono anche gli **articoli prodotti per ElectroYou**, che illustrano progetti completi con lo scopo di guidare chi si addentra in un mondo i cui limiti sono, si potrebbe dire, solo quelli imposti dalla propria creatività.

Paolino, alias **Paolo Rognoni**,è l'autore del libro

- **Pillole di microcontrollori PIC**:



Pillole di Microcontrollori PIC.JPG

Nel suo sito personale, **PIC-EXPERIENCE**, amplia ed approfondisce gli argomenti che affronta nella nostra community e che riguardano la sua attività.

Qualche informazione generale sui PIC

Programmatore

Che programmatore mi potete consigliare per iniziare con i microcontrollori?

R: Una soluzione, ad ampio raggio è il PicStartPlus di MicroChip, che permette di programmare anche i dispositivi CMOS, come il 16C84 che hai citato. Però, perché spendere di più utilizzando dei dispositivi CMOS quando con la tecnologia FLASH si hanno molti più vantaggi a costi decisamente inferiori?

Per quanto concerne la programmazione: sono un amante del C anche se ho sviluppato (e sviluppo) firmware in assembly. Per i PIC ci sono molti compilatori; uno di quelli che ha, a mio avviso, un buon rapporto qualità prezzo lo puoi trovare su http://www.ccsinfo.com. Per l'assembly credo che il corso di TANZILLI possa essere una buona base per partire; tieni conto che i PIC sono macchine RISC ed hanno, per le versioni a 8 bit (esclusi i PIC18) veramente poche istruzioni.

Per l'assembler, Microchip fornisce **gratuitamente** un ambiente di sviluppo integrato che prende il nome di **MPLAB**, scaricabile dal sito sia come pacchetto completo, sia con installazione guidata via internet.

MPLAB contiene l'assembler e il linker per tutti i PIC prodotti. Nel menu di help è disponibile tutto l'instruction set. Attenzione: come ho già avuto modo di dire, consiglio sempre di scaricarsi il datasheet del dispositivo in quanto su quel documento si trovano le impostazioni da attribuire ai registri per abilitare determinate funzioni.

Hobby o professione?

Il PIC è molto usato in applicazioni hobbistiche. Ci sono esempi in cui i PIC vengono utilizzati in un campo diverso da quello hobbistico?

R: Come tutti i dispositivi elettronici, anche i PIC nascono per scopi professionali e ben si prestano all'uso hobbistico. Spesso è proprio grazie a questo uso che se ne deve le diffusione ad ampio spettro. Ne sei testimone tu che hai trovato a basso costo un programmatore "spartano" ad una fiera.

L'uso dei PIC per uso professionale va dall'automotive al controllo industriale in generale. Ne ho trovato uno per il controllo di una elettrovalvola per l'irrigazione del giardino di casa. Personalmente li uso sia in ambito hobbistico ma anche per scopi professionali: controllo motori, comunicazioni di sicurezza, conversioni A/D e filtraggio digitale. Ci sono molte aziende (anche italiane) che usano pesantemente i PIC; basta partecipare a qualche incontro organizzato dal Costruttore o dai rivenditori per rendersene conto.

Porta ISCP

Come funziona la "Porta ICSP"?

R: Microchip prevede che per i propri dispositivi FLASH di ultima generazione sia possibile la programmazione in-circuit (ICSP: In-Circuit Serial Programming) mediante il programmatore/debugger ICD2. Puoi acquistare ICD2 direttamente sul sito di Microchip Direct (http://www.microchipdirect.com) ad un prezzo sostanzialmente buono (circa 140 Euro) viste le potenzialità di questo dispositivo. Personalmente lo uso sia in ambito professionale sia a casa e non me ne pento!

MPLAB

Cerco informazioni riguardo al compilatore per pic MPLAB. Non mi riconosce i registri del PIC...

R: MPLAB è un ambiente di sviluppo che integra al suo interno l'assembler ed il linker per realizzare firmware per PICMicro, in linguaggio Assembly. Da qualche anno a questa parte si integra molto bene con compilatori nativi Microchip e di terze parti.

Che si utilizzi l'assembly oppure, ad esempio, il linguaggio C, è necessario compiere una operazione: creare un nuovo progetto. Tu questo lo hai fatto? Hai seguito tuttii passi indicati?

Il fatto che il compilatore (o l'assembler) non riconosca i registri può essere dettato dal fatto che non hai incluso i file di specifica per il PIC di tuo interesse.

Se non lo hai ancora, installa l'ultima versione di MPLAB che trovi a questo link. Dopodiché accedi al menu Project e seleziona la voce Project Wizard. Ti si apre una procedura guidata abbastanza semplice.

Inizia così, poi se hai dei dubbi, approfondiamo.

Clock

Quali sono i vantaggi che potrei avere a far funzionare il PIC a 4 Mhz, a parte lo svantaggio di andare piano?

R: Non perdi di affidabilità. Tieni presente che una istruzione del PIC (escluse quelle relative ai salti) viene eseguita in 1 ciclo macchina e che ogni ciclo macchina sono 4 colpi di clock. Se hai un clock a 4Mhz il PIC lavora a 1MIPS, a 20MHz lavora a 5MIPS. Altri vantaggi sono derivati dal fatto che il clock serve diverse periferiche, come ad esempio la porta seriale. Se vuoi utilizzare la porta seriale ad alta velocità è obbligatorio un quarzo veloce. Gli svantaggi credo siano sostanzialmente legati al consumo maggiore di potenza elettrica. I PIC della famiglia 18 hanno introdotto il clock-swapping che permette di lavorare a bassa frequenza anche per ridurre i consumi. Ciao.

Durata di una routine

Quanto tempo impiega il microcontrollore PIC16F84A a svolgere la seguente routine, usando un quarzo da 4 MHz

```
                        movlw    d'200'
                        movwf    a1
loop1                 movlw     d'100'
                        movwf    a2
loop2                 movlw     d'50'
                        movwf    a3
loop3                 decfsz     a3,1
                        goto       loop3
                        decfsz    a2,1
                        goto       loop2
                        decfsz    a1,1
                        goto       loop1
                        return
```

R: I PIC sono macchine RISC con architettura Harvard (modificata). L'esecuzione di ciascuna istruzione avviene ogni 4 colpi di clock cioè alla frequenza Fcy = Fosc/4, dove Fosc è la frequenza dell'oscillatore principale. Fanno eccezione le istruzioni di salto (branch) per le quali per la loro esecuzione va considerato un tempo pari a 2Tcy, dove Tcy = 1/Fcy.

Ci sono poi altre eccezioni, quelle (che non riguardano il tuo caso) della risposta all'interrupt dove il tempo di latenza all'interrupt "prende qualche Tcy".

Ora, se leggi il datasheet è mostrato per ciascuna istruzione il valore del numero dei cicli macchina che vengono impiegati. Nel tuo caso si tratta di tre cicli annidati. Il modo migliore per verificare la durata di una routine è quella di simularla. Con MPLAB puoi abilitare MPSIM e verificare quanto dura. Un conto spannometrico mi fa risultare oltre 9 milioni di cicli macchina... Verificalo con MPSIM.

Suggerimenti per

controllare due uscite con un ingresso

La mia intezione è realizzare un programma che da un solo Pin di ingresso usando delle funzioni tipo (Pulsin o Count) attivi due Pin di uscita a seconda della necessità.

R: Riassumiamo le specifiche così:

1) Quando il PIC vede attivo l'ingresso (RA0), fa partire un TIMER con una base dei tempi prefissata.

2) Se entro il tempo prefissato non giunge un altro impulso, viene attivata l'uscita RB0.

3) Se entro il tempo prefissato giunge un altro impulso (sempre su RA0), viene attivata l'uscita RB1.

Il TIMER0 va configurato con la base dei tempi che si desidera (ad es. 2 secondi). Il firmware resta in polling su RA0; quando giunge l'impulso (vanno considerati e filtrati i rimbalzi), si azzera il TIMER0. Il firmware controlla sia la presenza di un successivo impulso di RA0 sia lo scadere del TIMER0. Se viene verificata la condizione espressa in 2), si attiva RB0, se viene verificata la condizione in 3), si attiva RB1.

A parte i fronzoli, questa può essere l'ossatura.

generare frequenze con PWM

Come generare con un pic 18F2550 in modalità PWM 2 frequenze con duty cycle 50%

R: Se devi generare due segnali a frequenze diverse, con duty al 50%, potrebbe non servire nemmeno l'utilizzo del modulo CCP! Bastano infatti i due timer programmati in modo che vadano inoverflow al doppio della frequenza da generare (cioè ogni semiperiodo del segnale). La risposta all'interrupt dell'overflow del timer si occuperà di fare il toggle del pin che ti interessa.

analizzare segnali a frequenza variabile

Vorrei analizzare 2 segnali TTL (0 - 5V) ad onda quadra che hanno una frequenza variabile tra 20 e 400 Hz. Qual è il miglior modo di acquisire i segnali? Io vorrei cronometrare un intero periodo di salita e discesa dei 2 segnali e poi fare dei confronti.

R:Con frequenze così basse (ma anche molto più alte, se scegli bene la frequenza

dell'oscillatore) puoi utilizzare il modulo CCP dei PIC configurato in modalità **CAPTURE**.

Ti consiglio di leggere [http://ww1.microchip.com/downloads/en/devicedoc/31000a.pdf questa guida] almeno per le parti che riguardano il modulo CCP, i timer e gli interrupt.

Il PIC18F2550 ha due moduli CCP e per ciascuno di essi devi usare un timer differente. Credo che il lavoro che hai in mente tu lo si possa fare solo con un PIC18. Non mi sembra che i PIC16 (anche quelli con 2 canali CCP) permettano di utilizzare due timer differenti per CCP1 e CCP2.

Con il modulo CCP hai il vantaggio che puoi scatenare interrupt ogni fronte (di salita o di discesa, a scelta e sempre programmabile) oppure ogni 4 o 16 fronti di salita. Inoltre, ti risparmi una serie di trasferimenti, via firmware, del contenuto dei registri dei timer. Direi che il modulo è studiato appositamente anche per questo genere di applicazioni.

tombola elettronica

Voglio realizzare una tombola elettronica..

R: Beh, tanto per cominciare ti do qualche consiglio:

- scegli il PIC che intendi utilizzare: un 18 pin dovrebbe essere sufficiente, se vai a pilotare due display a 7 segmenti in multiplexing;
- stendi uno schema elettrico del circuito che poi dovrai realizzare;
- avvia il progetto con MPLAB, iniziando con le cose più semplci, quelle che sai fare;
- poi si impostano timer ed interrupt;
- infine la gestione del timer e del buffer di indicazione estratto/non estratto.

Se suddividi il problema in piccoli compiti da svolgere, vedrai che la cosa non è poi così complicata. L'importante è avere le idee chiare su cosa vuoi ottenere.

Suggerimento: utile sarebbe l'adozione di un CD4511 che è un decoder BDC per display a 7 segmenti. Un'applicazione di tale chip la trovi in questo articolo, ad esempio.

Dai un'occhiata a questo articolo. C'è una sezione che riguarda la generazione dei numeri casuali. È vero, è scritta in C, ma la teoria che ci sta dietro è facilmente implementabile anche in assembly.

generare e selezionare frequenze

Voglio realizzare con un PIC un generatore di onde quadre con possibilità di selezionare 6 frequenze fisse sui 100 MHz, con la possibilità di farle emettere in sequenza automaticamente accendendo il rispettivo led di segnalazione ad ogni sequenza.

R: Mi sono preso qualche tempo per pensarci su e per darti alcune risposte su come potresti procedere.

Prima idea.

Viste le specifiche del tuo progetto, sembra interessante poter sfruttare il modulo hardware PWM che molti PIC hanno al proprio interno. Visto che il conteggio del pin-out sembra farevole (in funzione del numero di I/O impiegati), ti suggerisco di vagliare l'ipotesi di un PIC16F819. Per ottenere delle frequenze così basse con il PWM hardware, è necessario collegare al PIC un quarzo con frequenza bassa, diciamo 32768 Hz. Così facendo, se non ho sbagliato i conti, riesci a regolare tutte e 6 le frequenze in uscita (da 6.25 Hz a 100 Hz) con duty-cycle al 50%. E qui viene il il bello della regolazione del duty-cycle. Stando al datasheet, mi sembra che i valori che richiedi siano ottenibili senza troppi problemi.

Seconda idea

Con un quarzo da 32768 MHz il PIC funziona, ma certamente non come se avesse un quarzo da 4 Mhz o da 20 MHz... Se preferisci usare un quarzo con frequenze maggiori da 32768 Hz, puoi pensare di affrontare nuovamente il progetto come illustrato precedentemente (cioè con il modulo PWM integrato) e di generare frequenze con valori di 10 o di 100 volte superiori a quelli che suggerisci e poi di applicare un divisore (per 10 o per 100) all'uscita del PIC.

Terza idea

La cosa più laboriosa a livello software è quella di generarsi autonomamente le frequenze con il relativo duty-cycle, ma questo richiede l'uso dei TIMER e una buona dose di conoscenza della programmazione. In questo caso, dato che le frequenze non sono critiche, un codice scritto in C può alleviare il fastidio dell'assembly. Un vantaggio c'è: puoi usare un PIC senza il modulo PWM integrato e risparmiare qualche centesimo di Euro... Se hai in casa un PIC16F84A può andar bene allo scopo, anche con un quarzo da 4MHz.

Regolare la velocità di una ventola

Devo poter regolare la velocità di rotazione di 6 ventole da 12 V

R:Mi sembra un problema "facilmente" risolvibile con un microcontrollore. Se non interpreto male, trattasi di 6 ventole in DC; l'idea che mi viene è la seguente: adottare un microcontrollore con (almeno) un ingresso analogico (potenziometro), 4 digitali oppure uno analogico (selezione del numero della ventola: in caso di input digitale si interpone un multiplex, nel caso in input analogico si impiega una tecnica di input con più resistenze), e 6 uscite PWM indipendenti. I comandi in PWM possono essere realizzati via firmware, con tecniche consolidate di generazione PWM a mezzo dei timer, in quanto microcontrollori "a basso costo" e con 6 PWM indipendenti non li ho ancora trovati. Può servirti qualche altro input per dare START/STOP/RESET al sistema, magari un LED di segnalazione o quant'altro. Poi c'è il firmware da scrivere, ovviamente...

Usare l'interrupt

Generare temporizzazioni lunghe

Con un 16f84a devo effettuare un ritardo di 12 ore senza una grande precisione.

R:

Interrupt e precisione del timer non sono correlati. Se durante l'esecuzione del timer il PIC deve svolgere altre funzioni, allora è utile l'interrupt.

Colgo l'occasione per suggerire un PIC diverso, se possibile. Ti suggerisco in particolare un PIC che abbia il modulo del TIMER1 al quale puoi collegare un quarzo esterno da 32768 Hz. Se configurato bene (e non è difficile da farsi) il contatore del TIMER1 può andare in saturazione ogni 2 secondi e darti un interrupt. All'interno della routine aggiorni un contatore che, se supera il limite delle 12 ore, ti segnala lo scadere del timer. Facendo due conti:

12ore=3600*12=43200 secondi.

Con un interrupt ogni 2 secondi, devi allocare in memoria un contatore a 16 bit che se supera 43200/2=21600 significa che sono trascorse 12 ore.

Questa è una soluzione che dà precisione, semplicemente perché si usa un quarzo esterno dedicato solo al TIMER1.

Per quanto concerne i PIC che si possono usare ti consiglio il PIC16F819 oppure il PIC16F648A che sono pin-to-pin compatibili con il 16F84A.

Overflow

Che succede quando il contatore va in overflow, se non ho attivato l'interrupt (ammesso che sia possibile), rimane nella cella di allocazione del conteggio il risultato massimo, cioè i bit tutti settati ad 1 fin quando non viene azzerato il TMR1?

R:La funzione di interrupt è sempre attivabile/disattivabile a piacere. Se il contatore va in overflow, il registro ha superato il valore 0xFFFF. Il contatore dovrebbe azzerarsi e ripartire. In questo caso (overflow) il flag T1IF (Timer 1 Interrupt Flag) viene posto a 1 ma non scatena interrupt se la funzionalità non è attivata. Se non vuoi attivare l'interrupt è chiaro che devi andare in polling su T1IF; se T1IF vale 1, allora c'è stato overflow. Devi porre T1IF a zero.

Riutilizzo di un codice che gestisce l'interrupt

*Ho letto con interesse **questo articolo**.Vorrei svolgere un altro tipo di programma, sempre utilizzando un interrupt, invece di accendere un led. Devo inserirlo quando viene chiamato il sotto programma T0ISR ??*

R: Esattamente.

Quello che devi rispettare sono le operazioni di context switch (che salvano e ripristinano il conenuto dei registri) e la cancellazione del flag T0IF. Puoi utilizzare la stessa struttura che ho usato io e al posto del codice inserito nella routine T0ISR ci metterai quello che tu vorrai che faccia il PIC. L'esempio del lampeggio del LED è stato fatto proprio per poter dimostrare, in modo tangibile all'occhio umano, come il circuito lavora.

Attenzione: la simulazione mostra come, con un oscillatore da 4MHz e le impostazioni dei registri che ho scelto io (in modo arbitrario), il periodo di tempo misurato tra un lampeggio e il successivo è di poco superiore a 1s: il PIC "perde" qualche frazione di secondo ad ogni lampeggio del LED (circa 0.0001 s).

Usare il convertitore A/D

Ho un segnale analogico che può variare da 0.5 a 4.5V, mi interessa che il PIC processi questo segnale a partire da 1V, e poi con soglie da 1.2, 1.4, 1.6, ecc.. ad

intervalli di 0.2V fino ad i 4.5V. Ad ogni soglia poi il PIC dovrà eseguire un' istruzione diversa.

R: Se per ipotesi considerassimo di configurare il convertitore A/D a 8 bit, la risoluzione prevista (considerando un fondo scala di 5V) è di:

5/256=0.01953 V = 19.53 mV per ogni valore analogico letto.

Se invece pensassimo di estendere la lettura a 10 bit, la risoluzione diventa:

5/1024=0.0048V = 4.88 mV per ogni valore analogico letto.

Sostanzialmente per ogni variazione di 19.53 mV (nel caso a 8 bit) o di 4.88 mV (nel caso a 10 bit) del tuo segnale in ingresso, il PIC riesce a discriminarne la differenza.

Facilmente si verifica che:

1) caso a 8 bit

- 1V corrisponde a 256/5 analogico, cioè 51.2 che diventa 51 in quanto è un valore intero;
- 1.2V corrisponde a 1.2*256/5 analogico, cioè 61.44 che diventa 61 in quanto è un valore intero;
- 4.5V corrisponde a 4.5*256/5 analogico, cioè 230.4 che diventa 230 in quanto è un valore intero;

2) caso a 10 bit

- 1V corrisponde a 1024/5 analogico, cioè 204.8 che diventa 204 in quanto è un valore intero;
- 1.2V corrisponde a 1.2*1024/5 analogico, cioè 245.76 che diventa 245 in quanto è un valore intero;
- 4.5V corrisponde a 4.5*1024/5 analogico, cioè 921.6 che diventa 921 in quanto è un valore intero.

Se il tuo segnale è molto rumoroso, puoi pensare di acquisirlo a 10 bit e di filtrarlo inanzitutto buttando via i due bit più bassi e poi considerando il dato come se fosse a 8 bit.

Per concludere

Le risposte scelte sono solo un piccolo assaggio di ciò che si può ottenere dai colloqui in diretta con Paolino.

L'invito che è lo scopo della collana **Alcune risposte di:...**, è dunque a viaggiare tra le sue risposte nel forum e, per gli appassionati, a proporre quesiti e portare avanti progetti per fare amicizia con i microcontrollori.

Estratto da "http://www.electroyou.it/mediawiki/index.php?title=UsersPages:Admin:paolino"