Volfango Furgani

INTRODUZIONE AI MICROPROCESSORI

15 February 2003

global $section,$id; ?>

Indice:

Premessa

Introibo filosofico

Un fondamentale problema didattico nell'insegnamento dell'elettronica è: quanto durerà ‘sta roba che vado dicendo?. Sei mesi, un anno? Pensiero angoscioso. L'evoluzione tecnologica e commerciale, il volere di Bill Gates, signore del libero mercato, tutti quanti insieme, in quanto tempo mi metteranno in netto fuori gioco? Invidia rancorosa per i prof di matematica e italiano. Poco si può fare. Questo è un tentativo di individuare i fondamenti che possano valere ancora qualcosa, per un tempo un pochino più lungo, nel turbine delle mode commerciali. Per dire: gli algoritmi durano più della codifica in un certo linguaggio. La codifica è influenzata dalla versione del linguaggio e, spesso, dal suo tramonto nell'orizzonte fugace del mercato. Gli algoritmi, almeno finora, dimostrano più robustezza nei confronti della variabile tempo. Spero tanto di imitarli, ma si sa come finirà.

Torniamo al micro. Dall' 4004, dallo Z-80 fino al Pentium non so cosa, troppe cose sono cambiate. Qualcuna, spero, meno. Vorrei parlare di queste.

Le tappe di avvicinamento

Un po' di storia

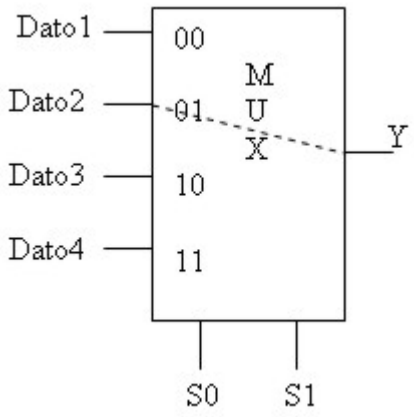
L'evoluzione tecnologica dei circuiti integrati ha permesso di far stare un numero enormemente crescente di dispositivi nello stesso spazio. Con molti transistor si fanno cose più complicate e quindi più specializzate. La specializzazione riduce lo

spazio di mercato ad una nicchia che richiede un numero limitato di unità sulle quali dividere i costi. L'idea, non di poco conto e per di più vincente, fu quella dell'elettronica programmata. In breve, facciamo il circuito più complesso che sappiamo e possiamo, che non abbia però un uso specifico, ma generico. Il µP nasce di qui: è un circuito la cui potenza di elaborazione è cresciuta in modo esponenziale (Che vuol dire? 1 cent a chi me lo sa spiegare in modo inequivoco), ma resta sempre specializzato da programma. E' il programma, non l' hardware che decide cosa il µP fa.

Ancora con la storia

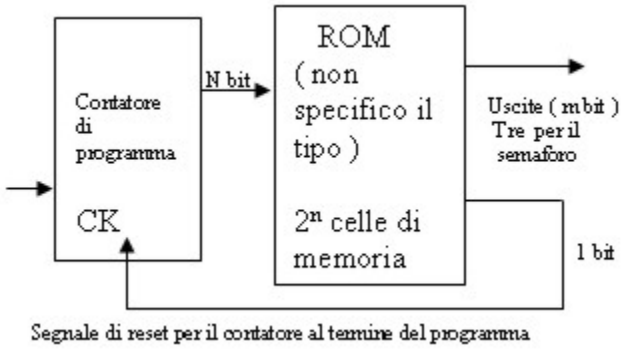
La prof di Italiano mi a detto: Vai tranquillo, ne sanno a pacchi! Che mentisse?

Io voglio comunque parlarvi di quella dell'elettronica programmata o meglio delle strutture. microprogrammate. Comincia con il multiplex. Una rete è programmabile se presenta diverse possibili relazioni ingresso/uscita Il più semplice è il mux con una sola uscita. Cos'è il mux lo sapete, che diamine!, ve l' ho insegnato io! Calmi, calmi, ci riprovo.



Oltre all'uscita il multiplex ha due gruppi di ingressi con funzioni distinte. N ingressi di selezione che in base alla combinazione binaria presente, collegano quello fra i 2^n ingressi dati selezionato all'uscita. In figura n vale 2 e il collegamento fra Dato2 ed Y implica che sugli ingressi di selezione sia presente la coppia 01. La programmabilità sta nel fatto che se i valori logici imposti ai quattro ingressi dati passano da 0001 a 0111, la funzione logica svolta passa da And ad Or. In generale con un mux ad n ingressi di selezione, posso programmare tutte le funzioni logiche di n variabili espresse nella prima forma canonica detta Somma di Prodotti od SP. Per fissare le idee n vale al massimo 4. Oltre, si debbono collegare più mux espandendo il numero di ingressi.

Un passo avanti: la struttura contatore Rom



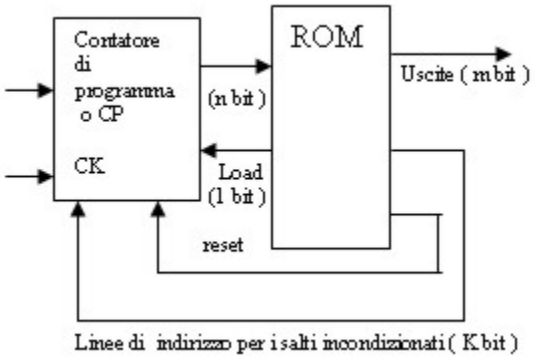
Il contatore scandisce gli indirizzi delle celle di memoria della rom. I dati contenuti nella cella selezionata passano sulle uscite, pilotando, che so, le luci di un semaforo attraverso opportune interfacce di potenza. Cosa vuol dire, interfacce di potenza, intendo, lo potreste imparare in quarta nella remota ipotesi che qualcosa vi smuova dal profondo sonno dei giusti che vi avvince e ristora. Il segnale di reset serve ad evitare che il contatore scandisca tutti i 2^n indirizzi anche se il programma è lungo molto meno. Chi passa all'incrocio se il programma semaforo è terminato e le celle della rom, no? Vi temo tetragoni a queste quisquilie. Questa struttura è rozzamente elementare perché scandisce gli indirizzi in rigida sequenza. Voi sapete (aridaiie!!) che un algoritmo oltre alla scansione sequenziale del programma, richiede la possibilità di scegliere se eseguire o no un blocco di programma e può necessitare della sua ripetizione. Ripartiamo. Il programma semaforo richiede la ripetizione di un ciclo finchè non accade un evento esterno: un incidente, l'ora di punta, le undici di sera o mezzanotte, non so perché quella è l'ora in cui mi sveglio la prima volta, non è che sono in giro. In corrispondenza ad uno di questi eventi, il semaforo salta ad una diversa sequenza. Salta è una parola chiave. Per ripetere un blocco di programma (la sequenza del semaforo) l'indirizzo emesso dal contatore deve tornare ad essere quello iniziale. Si tratta di un salto all'indietro. Se debbo scegliere fra due alternative, realizzate con due distinti blocchi di programma, uno dei due deve essere saltato o sorvolato per non essere eseguito. Punto di vista hardware: quando siamo in presenza della necessità di un salto, il contatore non deve incrementarsi all'indirizzo dell'istruzione successiva, ma deve caricare un diverso indirizzo che sta al programma fornire. Dico in un altro modo: le istruzioni if (scelta) o for (ripetizione) tipiche dei tradizionali linguaggi ad alto livello che avete visto in terza, nascondono dai salti. Il contatore di programma deve avere una discontinuità negli indirizzi emessi: non ci piove. If e for nascondono questo evento che rende il programma di difficile lettura. Non è un problema di poco conto. Provate a rileggere un programma sei mesi dopo che l'avete scritto. Se non lo capite voi, almeno non subito e non del

tutto, figuratevi gli altri che vi hanno sostituito dopo il licenziamento alla scadenza del periodo di formazione e lavoro. Dimenticavo, non si dice più licenziamento; è flessibilità. Potenza delle parole e dei mezzi sui quali viaggiano. Flessibilità suona bene al telegiornale; provate, invece, a parlarne in banca, se vi volete procurare una casa! Chiudo la parentesi socio-politica. Questo fatto del nascondere si chiama astrazione, come dicono gli informatici. Più alto è il livello di astrazione, meno cose dovete sapere su quello che stà sotto. Sembra conveniente. E lo è. Ma a forza di astrarre, la macchina fa delle cose che non vi aspettate. Ecco un caso in cui siamo in due ad avere ragione: noi, che presi dalla vertigine dell'astrazione, bestemmiamo come turchi, e la macchina che, sotto sotto, fa quel che deve. Altro esempio di astrazione: il programma scritto nella rom è fatto di 0 ed 1. Ed alla fine sempre così deve essere. Provate a scrivere un Kbyte di 0 ed 1. E' la via più diretta per la paranoia con susseguente suicidio. Si possono nascondere gli 0 e gli 1 passando all'esadecimale. A che? Pazienza Volfango, ma non esagerate!. Pazienza viene dal latino patior, vale a dire soffrire. Non posso pazientare a tempo indeterminato. Prendete la stringa di bit e dividetela in gruppi di quattro. Esempio 0111|1010. Il primo numero è 7 il secondo 10. Ah, no! Qui invoco l'atto di fede. Chi mi ama mi segua! Sette, come tutti i numeri da 0 a 9 rimane com'è. I numeri da 10 a 15 sono sostituiti dalle lettere maiuscole da A ad F. Astrarre, in questo caso, vuol dire che è meglio scrivere 7 A piuttosto che 01111010. L'attacco di delirio che ne segue è temporaneo.

Il secondo passo (Non è il Pordoi. Guardate che è per contratto che dovete ridere.)

La struttura per i salti incondizionati.

I salti ci vogliono. Se non vi ho convinto fino ad adesso, pazientate voi! Ma sono di due tipi. L'incondizionato: qui sei e qui salta: alle 11 di sera, come ci sia arrivato non importa, il semaforo deve saltare alla sequenza lampeggiante. Poi ci sono i salti condizionati. Il semaforo alle 11 deve passare al lampeggio solo se è soddisfatta la condizione che non sia né venerdì né sabato. Quando arrivano le 11 il salto non è sicuro: è condizionato.

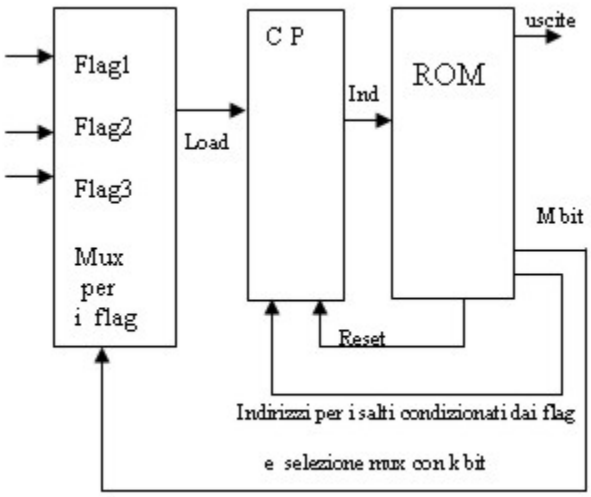


Crescono gli anelli di reazione come le banane sugli alberi. Non fate i grossi! . In sostanza cambia solo il contatore. Quando il segnale di Load si attiva, perché nella rom, in quel bit di Load non c' è più 0 ma 1, allora il contatore non prosegue nella sequenza normale, ma, senza alcuna condizione, carica l'indirizzo, sempre scritto nella cella della rom, portato dalle linee per i salti incondizionati e poi va avanti di lì. E' ovvio che la rom deve avere un numero di uscite adeguato. Possiamo già notare che sulle piste di collegamento viaggiano tre informazioni nettamente distinte: i dati (in questo caso in sola uscita, è una Read Only Memory, ricordate?), gli indirizzi, di salto o meno, e linee tipo load e reset che vengono dette di controllo.

Terzo passo

Ancora un po' di pazienza: dobbiamo vedere i **salti condizionati**.

Una condizione accettabile in queste situazioni deve essere booleana. Che tempo fa? La risposta deve essere o bello o brutto. Così, così non è accettabile salvo ricorrere a logiche a più valori (fuzzy). Una condizione booleana può essere rappresentata con una variabile a due valori che in questo contesto prende il nome di flag (bandiera). Capita spesso che le condizioni da verificare siano più di una: si pongono allora agli ingressi dati di un mux. Sono tre nella figura che segue. L'esecuzione condizionata di una parte di programma, tipo la sequenza lampeggiante, avviene solo se la condizione, né venerdì nè sabato nel nostro esempio, è verificata. Per testare una condizione e cioè lo stato di un flag, il programma deve porre sulle linee di selezione del mux l'appropriato indirizzo in modo che il valore del flag passi in uscita al mux per divenire l' ingresso Load del contatore. Se il flag è attivo il contatore carica l'indirizzo di salto ed il micro esegue la sequenza lampeggiante, altrimenti nisba, tira diritto.



Questa struttura permette, mediante l'esecuzione di salti, di effettuare scelte o ripetizioni. Non ha limitazioni teoriche, ma molte pratiche. Una per tutte, la demenzialità del linguaggio di programmazione. Il mio scopo nel presentare queste strutture è indicare una semplice soluzione hardware in grado di eseguire un programma. Quando mi sono avvicinato al microprocessore, mi pareva un buco nero, quello che inghiotte tutto e non restituisce nulla. Piano, piano mi sono abituato. Spero che questa premessa vi eviti, almeno parzialmente, il salto nel vuoto, adesso che parleremo del micro vero e proprio. Anzi, no! Parliamo prima un po' di un sistema di elaborazione di cui il microprocessore è solo un elemento.