Unità di controllo (informatica)

Da Wikipedia, l'enciclopedia libera.

> **Questa voce o sezione sull'argomento terminologia informatica non cita le fonti necessarie o quelle presenti sono insufficienti**.
> **Commento**: *nessuna fonte, mancano completamente le sezioni Note, Bibliografia e Collegamenti esterni*

L'**unità di controllo** è un componente delle CPU che ha il compito di coordinare tutte le azioni necessarie per l'esecuzione di una istruzione e di insiemi di istruzioni. È il componente che dà la possibilità al microprocessore di eseguire istruzioni diverse. Le azioni che coordinano i vari settori della CPU (la ALU, i registri, il Write-Back) vengono dette **micro-operazioni** o **micro-istruzioni**. Un insieme di micro-istruzioni viene detto **micro-programma**.

Indice

Input e Output

L'unità di controllo riceve in input ad ogni ciclo di clock:

- lo stesso *clock* che sincronizza lo svolgimento delle singole micro-istruzioni
- l'istruzione, contenente il codice operativo, sul quale determina le micro-istruzioni da eseguire
- i segnali di flag, che determinano lo stato della CPU e danno indicazioni sul precedente stato della ALU
- e segnali dal bus di controllo, come interrupt e di acknowledgement dai dispositivi esterni.

In output genera segnali interni alla CPU di trasferimento dati e di controllo della alu, e segnali attraverso il bus di controllo per memoria e I/O per quanto riguarda il resto del sistema.

Tecniche di implementazione

Rete combinatoria

È possibile implementare l'unità di controllo come rete combinatoria che genera delle uscite a seconda degli ingressi, sempre possibile per il Teorema di Shannon. I limiti di questo metodo sono una difficile progettazione e collaudo, e la scarsa flessibilità del sistema. Se si dovesse aggiungere un'istruzione o qualche controllo sarebbe necessario riprogettare il tutto.

Micro-linguaggio

Guardando più attentamente è possibile osservare che il compito dell'unità di controllo è simile a quello di un microprocessore ovvero effettuare delle scelte a seconda degli input e reagire di conseguenza (sequenzializzare le micro-operazioni ecc). È possibile quindi costruire un **linguaggio di micro-programmazione**. Sostanzialmente un segnale di controllo è una linea che varia tra 0 e 1 a seconda se il segnale deve essere spento o attivo. Ogni microistruzione deve quindi associare il valore 1 ai segnali di controllo richiesti dalle micro-operazioni la cui esecuzione è prevista in tale microistruzione. Per ogni codice operativo previsto si deve definire una sequenza di micro-operazioni, ossia un micro-programma. I micro-programmi prendono il nome di firmware, sottolineando la loro appartenenza sia all'hardware sia al software. Il *firmware* è memorizzato in una memoria interna chiamata *memoria di controllo*. Ovviamente anche in questo caso occorre effettuare delle scelte rivolte all'efficienza e al numero di microprogrammi richiesti. Nei moderni microprocessori vi sono molti registri e controlli da eseguire, per cui l'unica soluzione è quella di implementare microistruzioni composte da molti bit per gestire il gran numero di controlli da eseguire. La lunghezza delle microistruzioni è composta da tre fattori: dal massimo numero di micro-istruzioni diverse da gestire contemporaneamente, dalle modalità di codifica e rappresentazione dei segnali di controllo e dalle modalità con cui si specifica l'indirizzo della successiva microistruzione da eseguire. Si distinguono qui due modelli:

- la *microprogrammazione verticale* dove ogni microistruzione specifica poche microoperazioni da eseguire. In questo caso le microistruzioni hanno pochi bit, dove n segnali di controllo sono rappresentati da log2 n bit, ed è quindi necessaria una decodifica successiva dell'istruzione per generare i segnali di controllo
- o una *microprogrammazione orizzontale* dove si possono specificare molti segnali di controllo in parallelo e avere tanti bit quanti sono i segnali di controllo da generare. In questo caso è possibile effettuare un elevato parallelismo, e la codifica dei segnali è nulla o limitata.

Si può tuttavia implementare una via di mezzo, dividendo le microistruzioni in gruppi disgiunti, ottenendo quindi un ragionevole parallelismo con un numero non enorme di bit.

CAR e CBR

Si definisce *CAR* il registro indirizzi di controllo (*Control Address register*), e *CBR* il registro buffer di controllo (*Control Buffer Register*). Un'unità di controllo microprogrammata deve effettuare due operazioni fondamentali nell'unità temporale di riferimento che è il clock. Prelievo di una microistruzione attraverso la determinazione dell'indirizzo della memoria di controllo al quale si deve accedere (e quindi sequenzializzazione delle microistruzioni) e l'esecuzione della microistruzione. L'unità di controllo opera quindi nel seguente modo:

- il sequenziatore emette un comando di lettura
- La microistruzione contenuta nella locazione della memoria di controllo specificata dal Registro Indirizzi del Controllo (CAR) è trasferita nel Registro Buffer del Controllo (CBR)
- Il registro CBR in base al suo contenuto genera, direttamente o previa decodifica, i segnali di controllo e l'informazione relativa al successivo indirizzo
- Il sequenziatore inserisce il nuovo indirizzo nel registro CAR sulla base delle informazioni fornitegli dal registro CBR e dei flag provenienti dall'ALU

L'indirizzo successivo dipende dal contenuto del CBR e dei flag provenienti dall'ALU. A questo punto l'unità di controllo o prende l'istruzione successiva aumentando di uno il registro CAR, o salta ad un nuovo microprogramma copiando il contenuto di CBR in CAR attraverso una microistruzione di salto, o ancora saltare ad un altro microprogramma attraverso una vera e propria istruzione del linguaggio macchina in fase di IR.

Schema di Wilkes

Dal punto di vista prettamente fisico, è possibile implementare una unità di controllo microprogrammata attraverso lo schema di *Wilkes*, ovvero costruendo una griglia di conduttori a formare una enorme matrice, con dei diodi collegati agli incroci. Ad ogni colpo di clock viene attivata una riga della matrice, ottenendo dei segnali d'uscita alle colonne collegate con i diodi. La prima parte delle colonne riguarda le istruzioni da eseguire, la seconda parte l'istruzione successiva da eseguire. Ogni riga rappresenta quindi la microistruzione da eseguire e l'intera matrice è la memoria

delle microistruzioni. Ovviamente questo schema è rigorosamente orizzontale, e richiede un gran numero di bit. È possibile utilizzare schemi più complessi per la generazione dell'indirizzo successivo che richiedono moduli di sequenzializzazione.

Supponiamo che l'unità di controllo debba generare K segnali, sia esterni che interni. Con lo schema di Wilkes abbiamo 2k configurazioni possibili. Non tutte queste vengono utilizzate per vari motivi (due sorgenti non possono essere inviate alla stessa destinazione nello stesso ciclo, un registro non può essere sorgente e destinazione nello stesso ciclo, in un ciclo l'ALU può ricevere un unico segnale, in un ciclo il bus di controllo esterno può ricevere un unico segnale), e si potrebbe quindi pensare di ridurre il numero di segnali di ingresso della matrice. Ma questo complicherebbe di parecchio la programmazione e la decodifica in uscita, per cui si usano più bit di quelli strettamente necessari, ma alcune configurazioni non si codificano perché inutilizzate. Una tecnica di codifica delle istruzioni è quella del dividerle in campi, ogni campo, che attiva i suoi segnali di controllo, con la sua codifica. I campi sono indipendenti, e quindi campi diversi possono dare il via ad azioni diverse simultaneamente mentre ogni campo può dare il via a una sola azione per ciclo di clock.

Pipeline

In presenza di pipeline all'interno del microprocessore, l'unità di controllo dovrà essere in grado di rilevare hazard sui dati e sui controlli e modificare le istruzioni in corso di esecuzione. Tutti i controlli vengono salvati nei *registri di pipeline* e scalate via via in avanti finché non viene completata l'istruzione. L'introduzione della pipeline ha complicato notevolmente la struttura interna dell'unità di controllo, per prevenire i problemi legati agli hazard dei dati e dei controlli.

Voci correlate

- CPU
- Linguaggio di programmazione
- Rete combinatoria