Unità di controllo e datapath

▼ Creatore originale: @Samuele Gentile

Controllore e datapath

Fino ad ora, ci siamo concentrati sulla realizzazione di unità dedicate ad implementare funzioni specifiche. In generale, però, nei sistemi digitali queste unità sono interconnesse e comandate tramite opportuni segnali dedicati alla selezione dell'operazione da svolgere e su quali valori in ingresso.

Le operazioni possono essere selezionate e svolte sulla base di ulteriori criteri:

- stato del sistema;
- temporizzazione;
- altre condizioni.

Inoltre, nonostante a livello teorico una FSM possa rappresentare un gran numero di sistemi, ci sono casi in cui possono risultare troppo complessi per essere modellati efficacemente. Per esempio:

- un registro a scorrimento a 8 bit è descrivibile con una FSM con 256 stati differenti;

- un singolo registro a 32 bit definisce uno spazio degli stati con 2^{32} stati differenti.

Definizioni

Si utilizzano, quindi, due parti importanti nel circuito:

- controllore: è un'unità logica che, basandosi su ingressi e sul proprio stato interno, genera segnali di controllo per guidare il comportamento di un sistema digitale**.**

  È spesso realizzato come una macchina a stati finiti (FSM**)** e determina quali operazioni devono essere eseguite e **in** quale ordine, coordinando i vari componenti di un sistema (ad esempio ALU, registri, bus, memorie, ecc.).

  Un esempio pratico è la Control-Unit di una CPU, che coordina il ciclo di fetch-decode-execute.

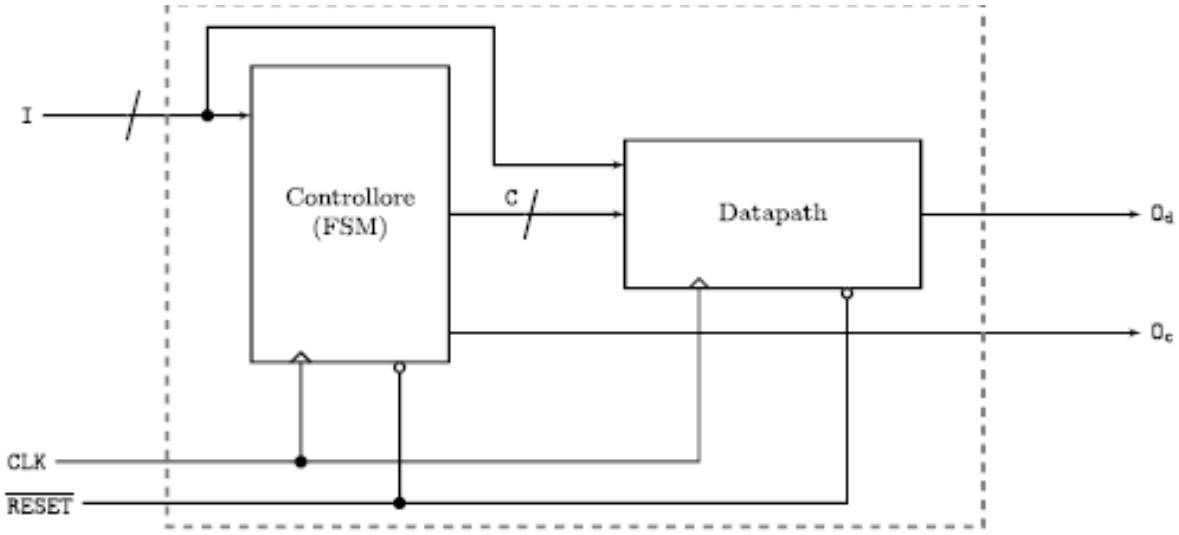
- datapath: parte del circuito che si occupa dell'elaborazione e del trasporto dei dati.

Sistemi di controllo aperto

Con controllo aperto, il controllore invia comandi e segnali al datapath senza ricevere nessuna retroazione.

Le uscite del sistema possono essere generate sia dal controllore, sia dal datapath.

Gli ingressi del sistema possono interessare entrambi i componenti.

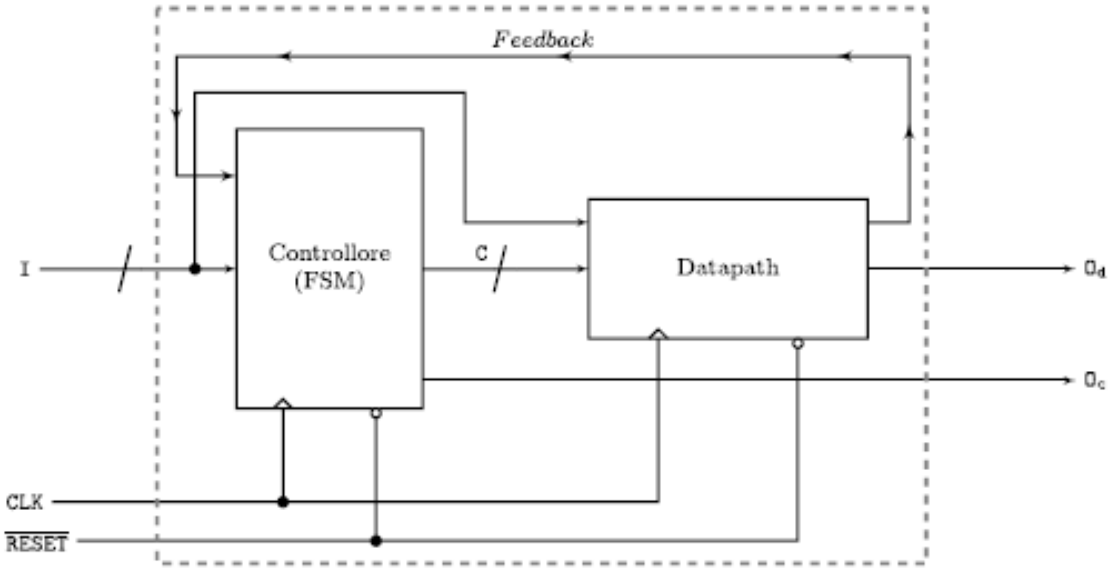


Esempio di sistema di controllo aperto

Sistemi a controllo retroazionato

Con controllo retroazionato, il controllore riceve informazioni di ritorno dal datapath ai fini del controllo dell'operato del sistema, in modo da poterlo usare come dato durante le successive fasi dell'algoritmo.

Per esempio, se dobbiamo fare una serie di shift fino ad esaurire il contenuto di un registro, il controllore imporrà di eseguire uno shift fino a quando il datapath non manda un segnale per dire che il registro è stato svuotato completamente.



Esempio di sistema a controllo retroazionato

Design di sistemi con controllore e datapath

Per creare un design di sistemi con controllore e datapath il procedimento è come segue:

1. descrivere le funzionalità del sistema da modellare;
2. determinare quali componenti del datapath siano necessari (registri, ALU, MUX, DEMUX ecc...);
3. definire le interconnessioni tra elementi del datapath;
4. identificare i segnali di input e output del controllore;
5. definire la sequenza di segnali che il controllore deve generare perché il sistema presenti il comportamento atteso;
6. definire la FSM adatta a rappresentare il controllore necessario;
7. verifica, manuale o automatica, del sistema complessivo. In caso di errori e/o incongruenze, rivedere i passi da (2) a seguire;
8. implementazione e test del sistema definitivo. In caso di errori e/o incongruenze rivedere i passi da (2) a seguire.

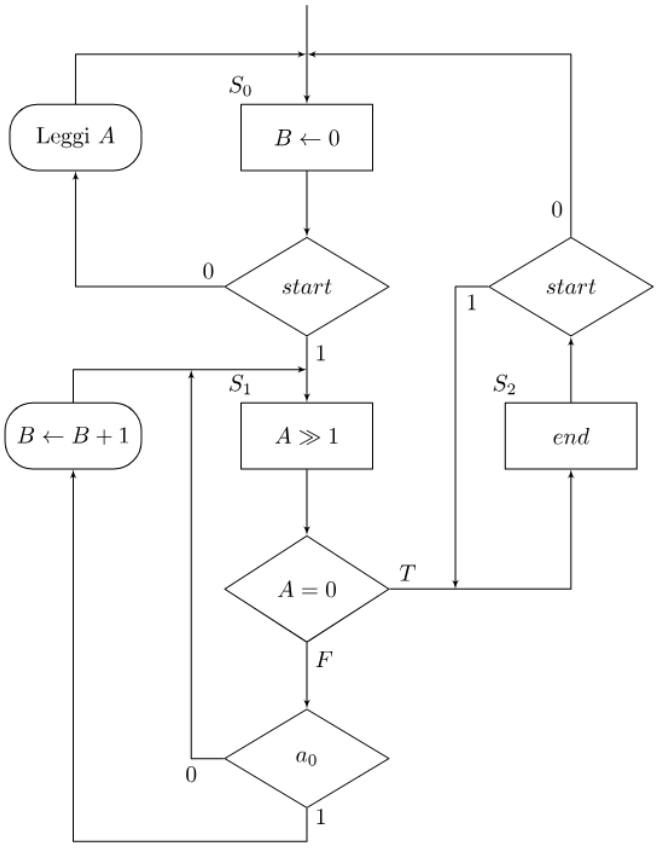
Esempio - Bit counter

L'obiettivo è partire da un registro A e contare quanti bit sono ad 1.

1. Si inizializza un registro B a zero;
2. Con un ciclo while si va a prendere il bit meno significativo di A, fino a quando A non è uguale a 0, mentre se è 1 incrementiamo B;
3. Per ogni iterazione del ciclo, si shifta il valore del registro A a destra.

$$
\begin{aligned}
&B \leftarrow 0 \\
&\textbf{while } A \neq 0 \textbf{ do} \\
&\quad \textbf{if } a_0 = 1 \textbf{ then} \\
&\qquad B \leftarrow B + 1 \\
&\quad A \gg 1
\end{aligned}
$$

Pseudocodice dell'algoritmo



Visualizzazione della ASM

Definizione del circuito

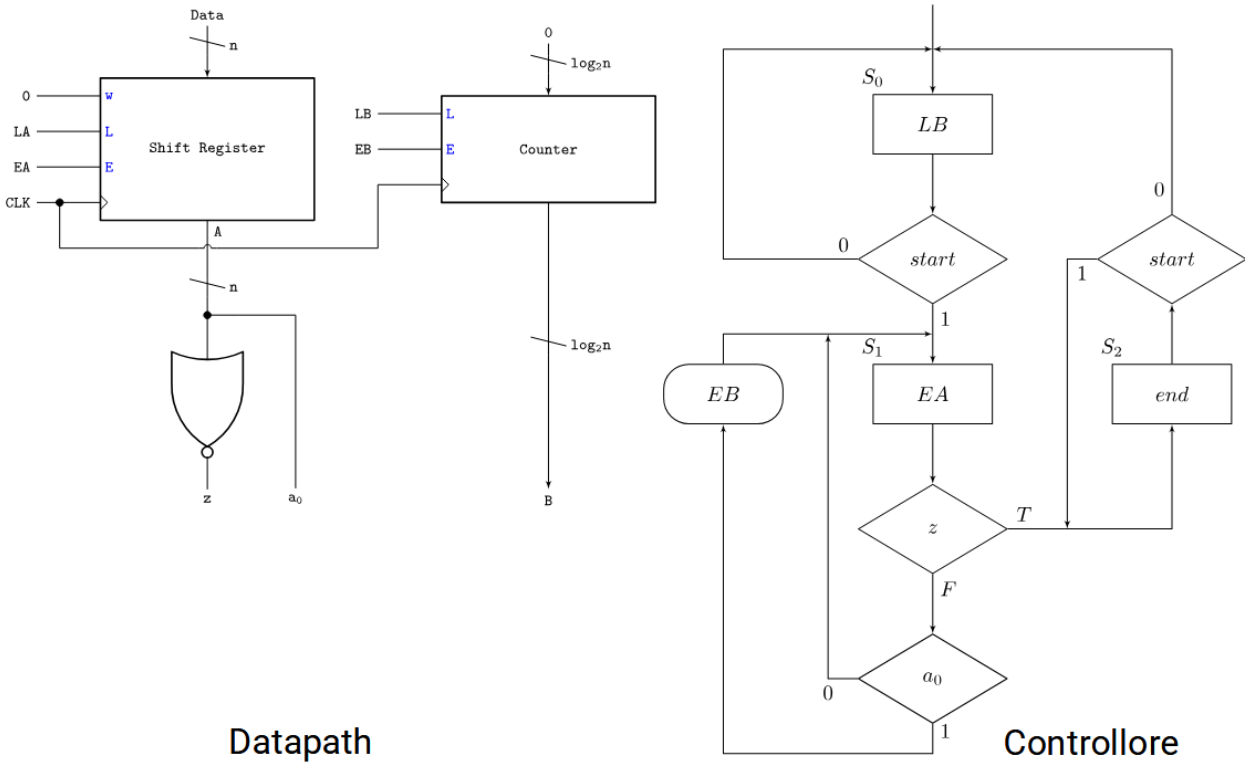
Nello pseudocodice e nel grafico a blocchi:

- la freccia puntata verso sinistra è un'assegnazione, mentre il simbolo di uguale è una verifica di uguaglianza (come `==` in C);
- esiste una condizione di start che inizia ad operare quando gli è stato detto di farlo, e finché non usciamo dallo start si rimane in attesa (nel loop superiore);

- una volta che la condizione di start è a 1, la prima operazione che incontriamo è lo shift di A, ma ricordiamo che il risultato dell'operazione di shift sarà "visibile" solo al colpo di clock successivo, ovvero quando il valore del registro verrà aggiornato.

Dallo studio dell'ASM, capiamo che le operazioni che dobbiamo implementare nel datapath sono definite da uno Shift-Register $(A \gg 1)$, da un contatore per B $(B \leftarrow B + 1)$ e da due blocchi di condizione:

- uno per verificare se A vale 0 $(A \neq 0)$;
- uno per verificare se a0 vale 0 $(a0 = 1)$, sapendo che può essere solo un valore binario.



Datapath e controllore ottenuti

Analisi del datapath

Lo Shift-Register:

- riceve in ingresso Data gli n bit iniziali di A;

- l'ingresso w è impostato a 0_2 serve per inserire un nuovo bit, il quale verrà inserito quando shiftiamo il registro;
- il segnale LA è il segnale che serve per abilitare l'ingresso dei bit di Data;
- il segnale EA è il segnale di enable, necessario per abilitare lo shift register, in modo che ogni volta che arriva un colpo di clock avvenga effettivamente l'operazione di shift.

Il Counter:

- riceve in ingresso il valore formato da $\log_2 n$ cifre da cui partire, definito solo da 0;
- il parallelismo è solo $\lfloor \log_2 n + 1 \rfloor$, poiché se, ad esempio, ho un registro A in ingresso di 16 bit, il suo numero di bit a 1 può essere al massimo 16, portando a una codifica in binario di 5 bit;
- Il segnale LB (Load B) serve per prendere il valore iniziale di B;
- Il segnale EB è il segnale che, se attivato ogni colpo di clock, incrementa B di 1.

La porta NOR con, in ingresso, l'uscita A dello Shift-Register, permette di implementare la funzione di confronto di A con il valore 0.

Analisi del controllore

Dato che il controllore è la parte di circuito che deve preoccuparsi della gestione dei vari segnali in ingresso, ci sarà un primo momento in cui LA e LB dovranno essere alti, in modo da impostare i dati al valore iniziale e, successivamente, dovranno essere impostati al valore basso per evitare che i dati vengano reimpostati al valore iniziale.

Il controllore deriva dall'ASM, dai suoi stati e dalle sue condizioni, come descritto in seguito:

- nel blocco della ASM, dove avevamo $B \leftarrow 0$, ora abbiamo il relativo segnale che svolge il compito nel counter, ovvero LB;
- $A \gg 1$ diventa EA;
- $B \leftarrow B + 1$ diventa EB;

- la prima condizione $A = 0$ viene definita dal segnale z, ottenuto dall'uscita della porta NOR;
- la seconda condizione viene definita dal segnale a0;
- si assume che il caricamento e la gestione del valore di A siano fatti da circuiteria esterna.

Esempio - Moltiplicatore binario SHIFT-ADD

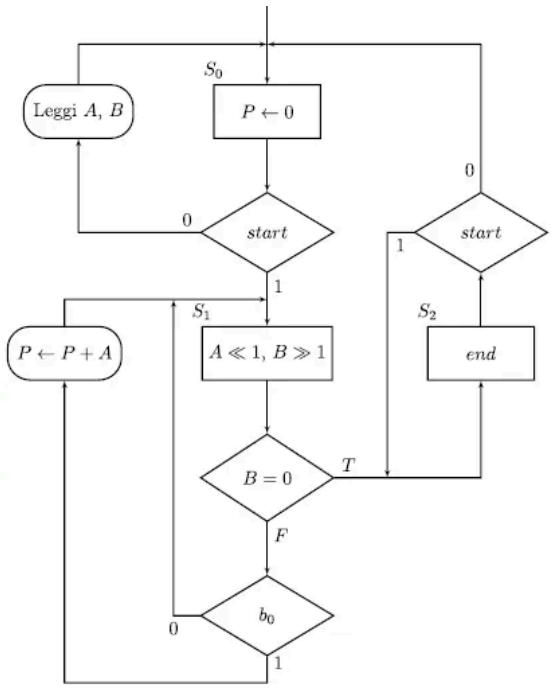
In questo esempio, vedremo come implementare un moltiplicatore binario sfruttando il paradigma controller/datapath.

L'algoritmo è lo stesso visto quando si è trattato il moltiplicatore parallelo.

$P \leftarrow 0$
for $i \leftarrow 0$ to $n-1$ **do**
  if $b_0 = 1$ **then**
    $P \leftarrow P + A$
  $A \ll 1$
  $B \gg 1$

$$
\begin{array}{cccccccc}
 & & & & 1 & 1 & 0 & 1 \leftarrow A \\
 & & & \times & 1 & 0 & 1 & 1 \leftarrow B \\
\hline
 & & & & 1 & 1 & 0 & 1 \\
 & & & 1 & 1 & 0 & 1 & \\
 & & 0 & 0 & 0 & 0 & & \\
 & 1 & 1 & 0 & 1 & & & \\
\hline
1 & 0 & 0 & 0 & 1 & 1 & 1 & 1
\end{array}
$$

Pseudocodice dell'algoritmo



Visualizzazione della ASM

Analisi del circuito

Il circuito è molto simile a quello implementato nel primo esempio:

- due Shift-Register, uno con scorrimento a sinistra (Shift-Left Register, per A) e uno con scorrimento a destra (Shift-Right Register, per B), con possibilità di caricare dati in parallelo e con segnale di abilitazione;
- un registro con enable per il valore di P, permettendo il mantenimento delle somme parziali;
- un sommatore per calcolare il valore P+A;

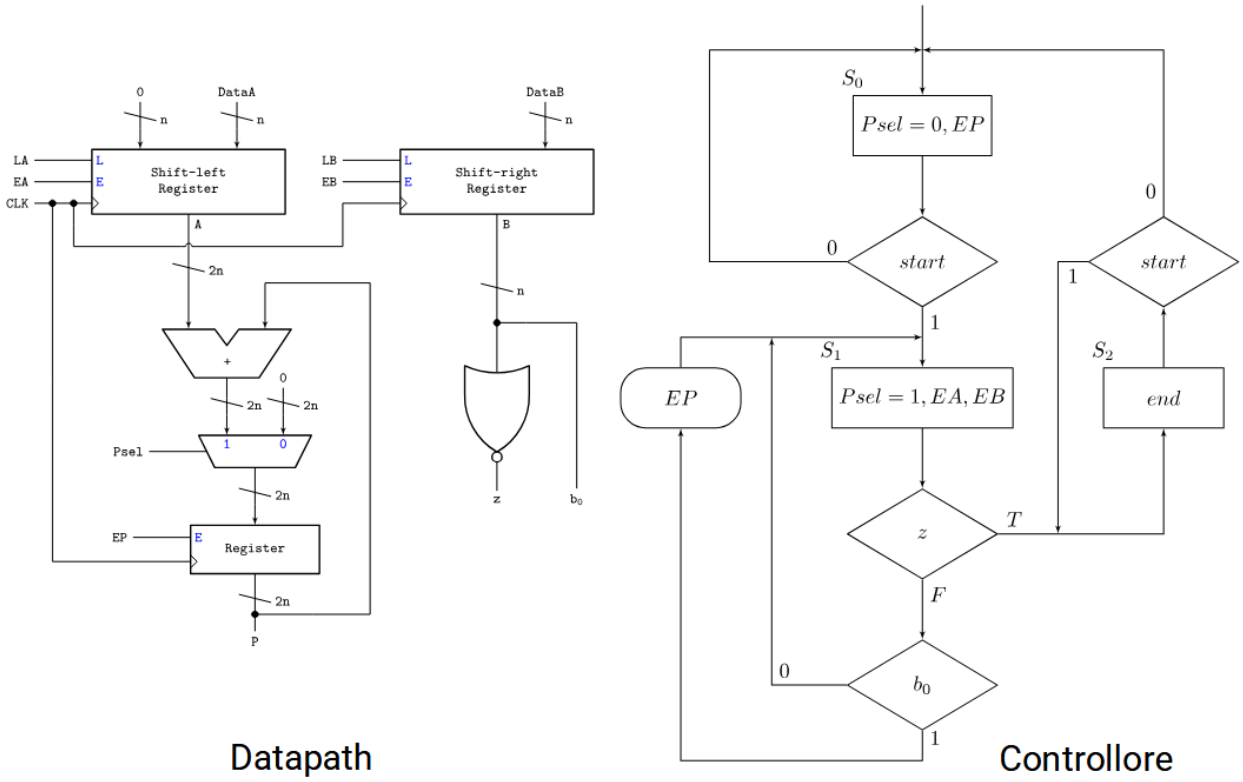
- un multiplexer 2 a 1 per controllare i dati caricati nel registro dedicato a P, permettendo la definizione di Psel, ovvero un segnale che permette di reimpostare la somma parziale se Psel=0, oppure di continuare ad eseguire le somme se Psel=1).

La porta logica NOR con, in ingresso, l'uscita B dello Shift-Right Register, implementa la funzione di confronto di B con il valore 0.

Analisi del controllore

Il controllore deriva dall'ASM, dai suoi stati e dalle sue condizioni, come descritto in seguito:

- il blocco $P \leftarrow 0$ viene implementato attraverso Psel=0 ed EP attivo alto, in modo da impostare tutti i valori a 0 per iniziare la somma nel Register;
- si assume che il caricamento e la gestione del valore di A e di B siano fatti da circuiteria esterna;
- il blocco $A \ll 1$ e $B \gg 1$ vengono realizzati tramite EA, EB e con Psel=1, in modo da eseguire le somme. Le somme verranno sempre eseguite, ma verranno salvate nel registro solo se b0=1, dato che b0 pilota il segnale EP;
- il confronto su z si fa sempre, come prima, con il blocco NOR.



Datapath e controllore ottenuti