Università degli studi di Parma
Dipartimento di Ingegneria dell'Informazione

Circuiti Sequenziali

Introduzione ai circuiti sequenziali sincroni

Circuiti combinatori e sequenziali
Concetto di memoria e di stato
Dalla specifica al dispositivo

Introduzione

- I circuiti digitali si possono classificare in due categorie
- Circuiti combinatori
  - Il valore delle uscite ad un determinato istante dipende unicamente dal valore degli ingressi in quello stesso istante
- Circuiti sequenziali
  - Il valore delle uscite in un determinato istante dipende
    - Dalla condizione di partenza del circuito
    - Dal valore degli ingressi in quell'istante
    - Dal valore degli ingressi in istanti precedenti
  - Per definire il comportamento di un circuito sequenziale è necessario tenere conto della storia passata del circuito stesso

Memoria e concetto di stato

- Le uscite di un circuito sequenziale in un dato istante di tempo dipendono
  - Dalla condizione iniziale del circuito
  - Dalla sequenza di ingressi, applicata in un arco temporale finito, fino all'istante considerato
- Questo aspetto implica che il dispositivo ha memoria degli eventi passati
  - All'istante t l'informazione relativa al "contenuto" di questa memoria è rappresentata nel concetto di stato
    - Le reti combinatorie possono essere considerate un caso particolare di sistema sequenziale dove lo stato è unico

Memoria e concetto di stato

- Trattiamo solo macchine a stati finiti deterministiche
  - Per ragioni legate alla realizzabilità fisica, la memoria di una rete sequenziale è di dimensioni finite
  - Dato uno stato ed una configurazione di ingresso il nuovo stato è identificato univocamente

Esempio: Specifiche

- Serratura a combinazione di una porta

  "Fornire la combinazione di 3 valori in sequenza per aprire una porta. Se si riconosce un errore la serratura deve rimanere chiusa ed è necessario ripartire dall'inizio. Quando la porta viene chiusa, il circuito di controllo della serratura deve essere ri-inizializzato in attesa di una nuova sequenza."

- Ingressi:
  - Valore in ingresso che appartiene alla sequenza
  - Reset (per inizializzare il sistema)
- Uscite:
  - Porta aperta/chiusa

Esempio: Specifiche

- Combinazione:
  - Funzionalità cablata
    - La combinazione è parte della struttura del dispositivo da realizzare e staticamente definta
  - Funzionalità programmata
    - Fornita in ingresso

Esempio: Implementazione software

```
int SerraturaCombinazione()
{
  int c[3] = { 3, 4, 8 };          /* Sequenza da riconoscere              */
  int v[3];                        /* Sequenza immessa                     */
  int error = 0;

  while( ! WaitValue() ) ;         /* Attendi nuovo valore                 */
  v[0] = ReadValue();              /* Leggi nuovo valore: prima cifra      */
  if( v[0] != c[0] )               /* Controllo della cifra                */
    error = 1;

  while( ! WaitValue() ) ;         /* Attendi nuovo valore                 */
  v[1] = ReadValue();              /* Leggi nuovo valore: seconda cifra    */
  if( v[1] != c[1] )               /* Controllo della cifra                */
    error = 1;

  while( ! WaitValue() ) ;         /* Attendi nuovo valore                 */
  v[2] = ReadValue();              /* Leggi nuovo valore: terza cifra      */
  if( v[2] != c[2] )               /* Controllo della cifra                */
    error = 1;

  return error;
}
```

Esempio: Implementazione hardware

- La specifica è incompleta
  - Alcuni aspetti richiedono successive riflessioni
    - Aspetti impliciti da evidenziare
    - Aspetti rilevanti e non espressi nemmeno implicitamente
- Aspetti non definiti e non impliciti
  - Numero bit per la codifica del valore in ingresso e in uscita
    - Si specifica che le cifre lette sono 3 ma non se ne definisce l'intervallo d'appartenenza
    - Non è implicito che la tastiera sia costituita dalle cifre 0-9
- Aspetti impliciti da evidenziare
  - È indispensabile identificare che è stato fornito un nuovo valore in ingresso per evitare che l'inserimento di un valore sia considerato più di una volta
    - Dipende da come sarà il dispositivo: sincrono o asincrono

Esempio: Implementazione hardware

- Il sistema è sequenziale
  - È necessario fornire una sequenza di valori ricordando la storia degli ingressi
  - È necessario ricordare se si è verificato un errore
  - L'errore deve essere rivelato solo alla fine della sequenza
    - Per ragioni di sicurezza

Esempio: Implementazione hardware

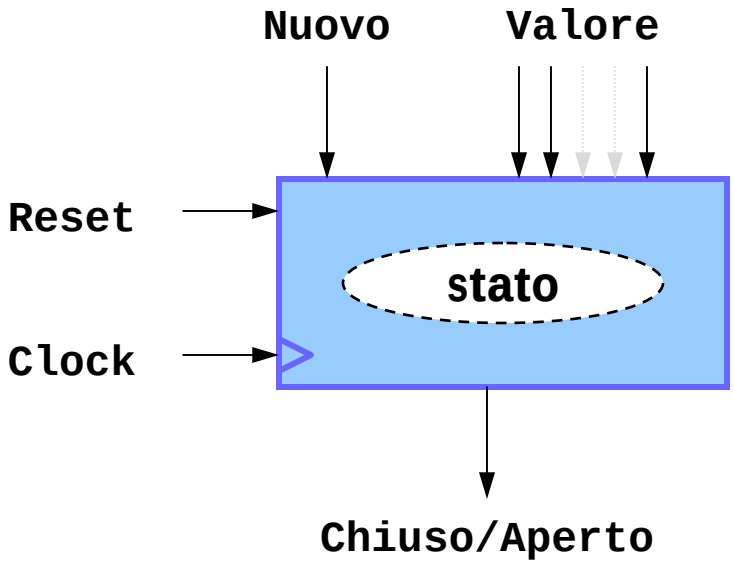
- Identificare che è stato immesso un nuovo valore
  - Dispositivo sincrono
    - Il clock regola l'evoluzione temporale dei sistema
    - Gli ingressi devono essere campionati solo quando sono stabili
    - È indispensabile avere un segnale di ingresso che rappresenti la validità del dato e la sua unicità
    - Non si vuole che la pressione di un tasto venga preso in considerazione più di una volta
  - Il dispositivo è asincrono
    - Il sistema evolve sulla base di eventi di ingresso

Esempio: Considerazioni

- Si ragiona su sistemi con stati stabili (astrazione)
  - Osservazione delle uscite avviene solo dopo che è trascorso un tempo sufficientemente lungo affinché il sistema possa effettuare i cambiamenti e stabilizzare le uscite
- I progettisti implementano tale astrazione
  - La memoria del sistema è rappresentata dal suo stato
  - I cambiamenti di stato del sistema sono ammessi solo in particolari istanti di tempo controllati da un segnale esterno periodico detto clock
  - Il periodo del clock è il tempo tra i cambiamenti di stato
    - Deve essere sufficientemente lungo per permettere al sistema di raggiungere lo stato stabile prima che avvenga il successivo cambiamento di stato al termine del periodo

Esempio: Implementazione hardware

- Rappresentazione logica del dispositivo
  - Pin-out logico



Esempio: Implementazione hardware

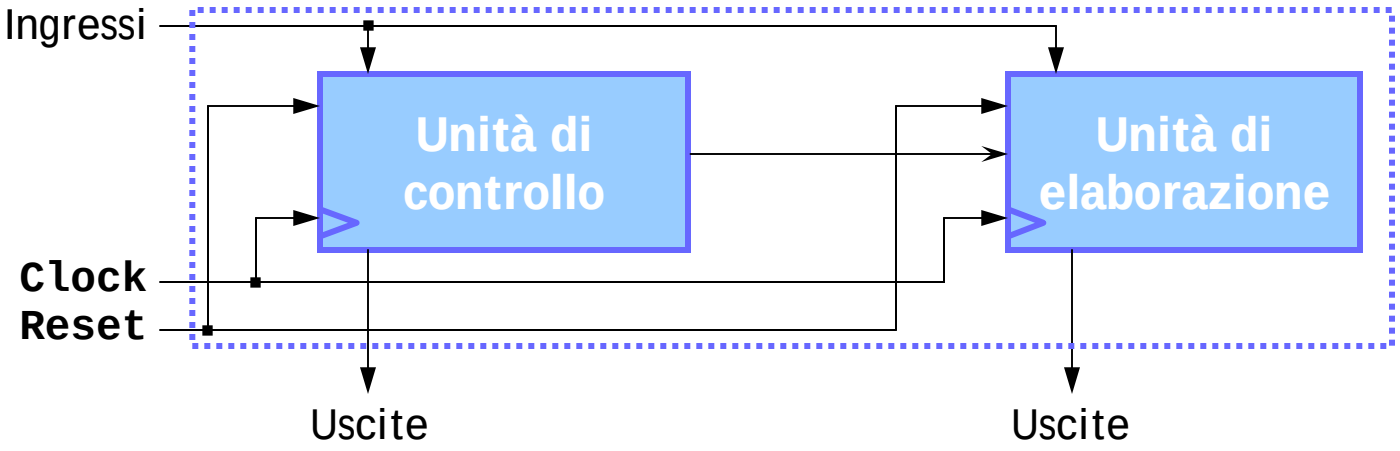
- Si hanno due possibili modi per implementare il sistema
- Come un'unica Macchina a Stati Finiti
  - FSM: Finite State Machine
- Come Unità di Controllo e Unità di Elaborazione
  - FSMD: Finite State Machine + Data-path

Esempio: Implementazione hardware

- Come un'unica Macchina a Stati Finiti
  - Ogni ingresso porta il sistema in uno stato che corrisponde
    - Ad uno stato che relativo alla sequenza riconosciuta, oppure
    - Ad uno stato che identifica la presenza di un errore
  - L'identificazione della validità del valore immesso è definita nel comportamento della macchina a stati. Si hanno due possibilità:
    - Gli ingressi raggiungono direttamente la macchina a stati
      - Struttura priva di flessibilità
    - Rete combinatoria per la codifica dell'ingresso alla macchina a stati (normalizzazione degli ingressi)
      - La struttura è più flessibile rispetto alla precedente poiché una modifica della sequenza incide solo sulla parte combinatoria di codifica degli ingressi

Esempio: Implementazione hardware

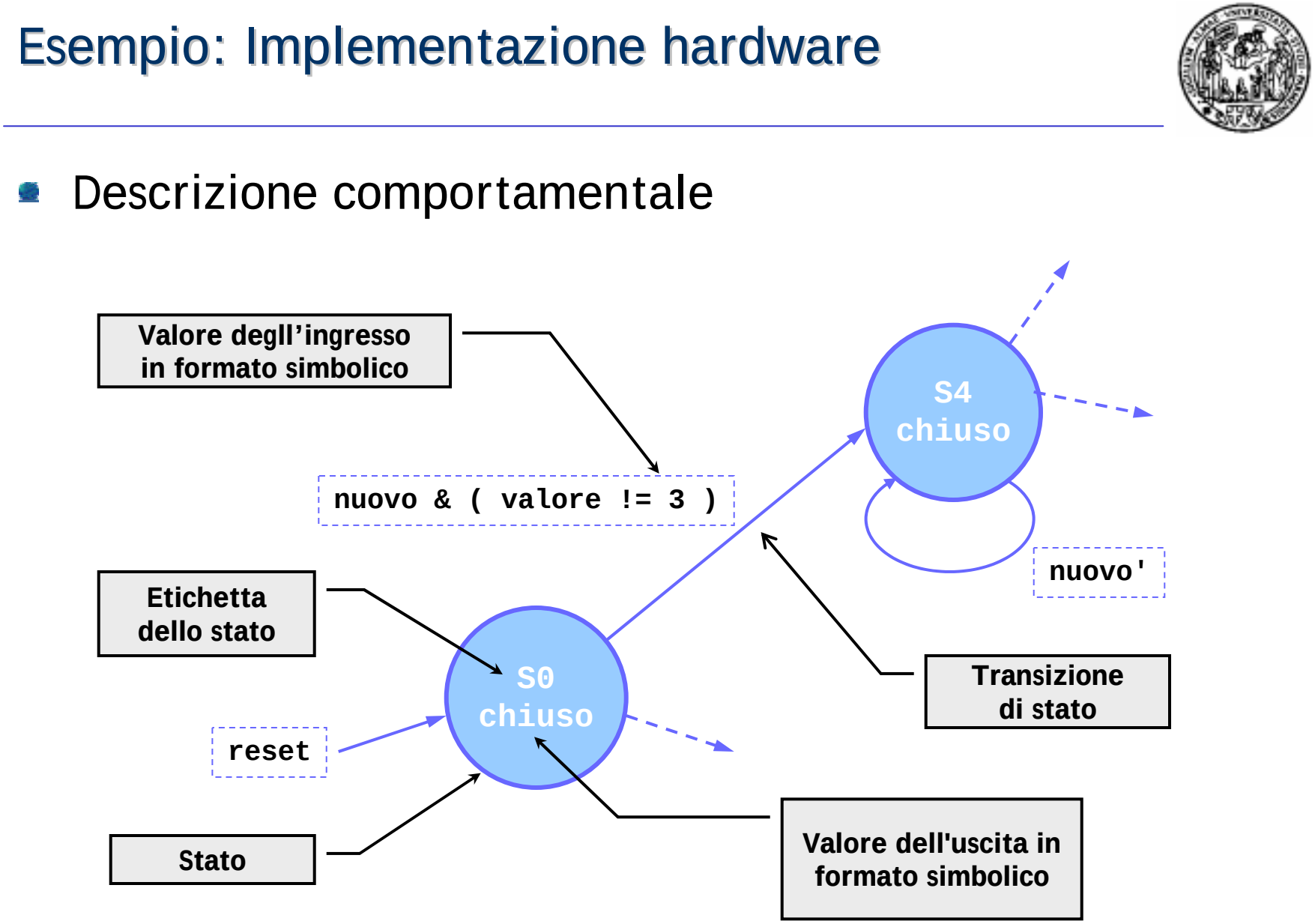
- Come Unità di Controllo e Unità di Elaborazione
  - L'unità di controllo è realizzata come macchina a stati
    - È generale ed indipendente dalla sequenza da riconoscere
    - Comanda l'unità di elaborazione
  - L'unità di elaborazione
    - È controlla i valori immessi
    - Restituisce all'unità di controllo un valore che modifica la sequenza di controllo



Esempio: Implementazione hardware

- La descrizione del comportamento di una macchina a stati è ottenuto attraverso un diagramma degli stati
  - Rappresenta il modo con cui la macchina a stati evolve
    - Gli stati
    - Il modo in cui gli ingressi agiscono causando il passaggio da uno stato all'altro (transizioni di stato)
  - Una configurazione d'uscita
    - È associata ad ogni stato (come in questo caso), oppure
    - È associata ad ogni transizione tra gli stati

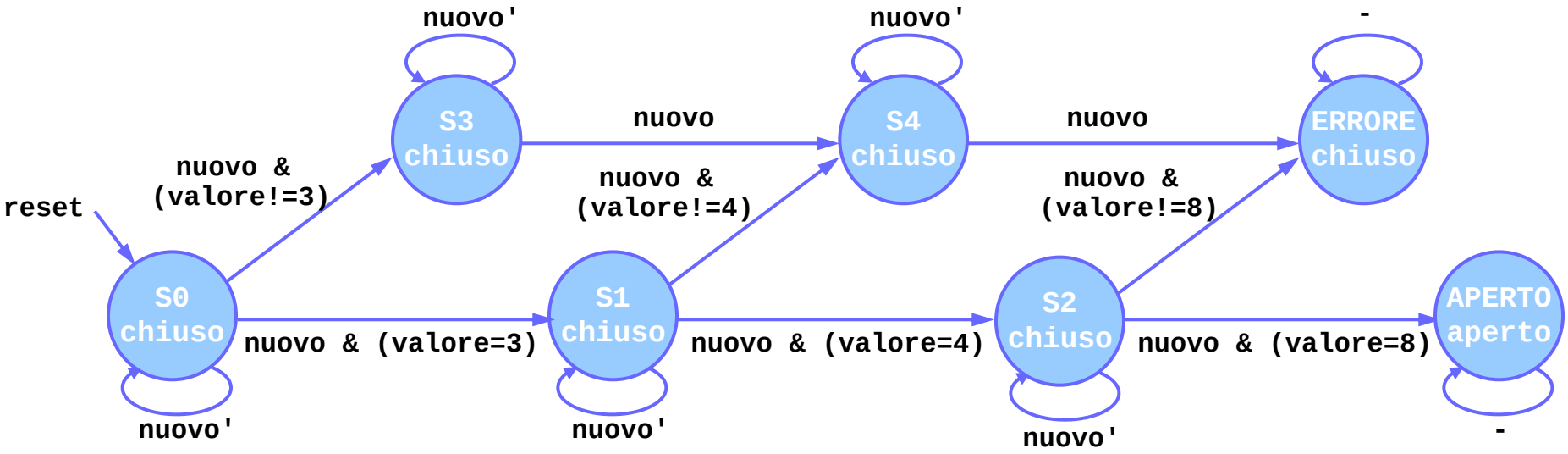
Esempio: Implementazione hardware



- Descrizione comportamentale

Esempio: Implementazione hardware

- Ipotesi:
  - Valori da riconoscere: 3, 4, 8
  - 4 bit di ingresso (minimo valore)

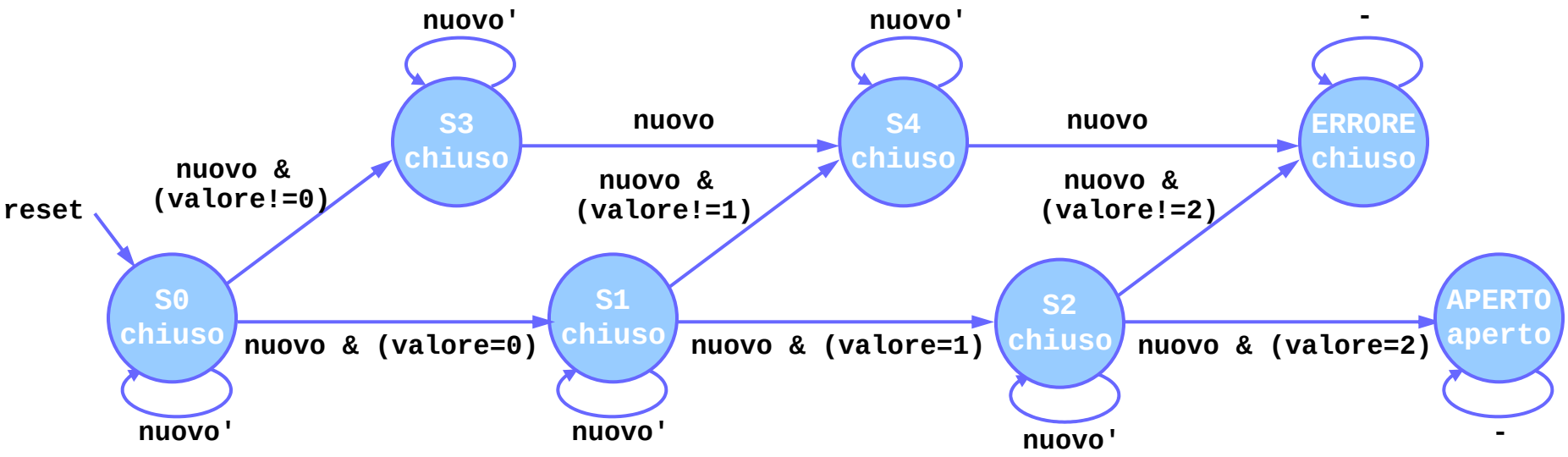


Dimensione dei segnali:

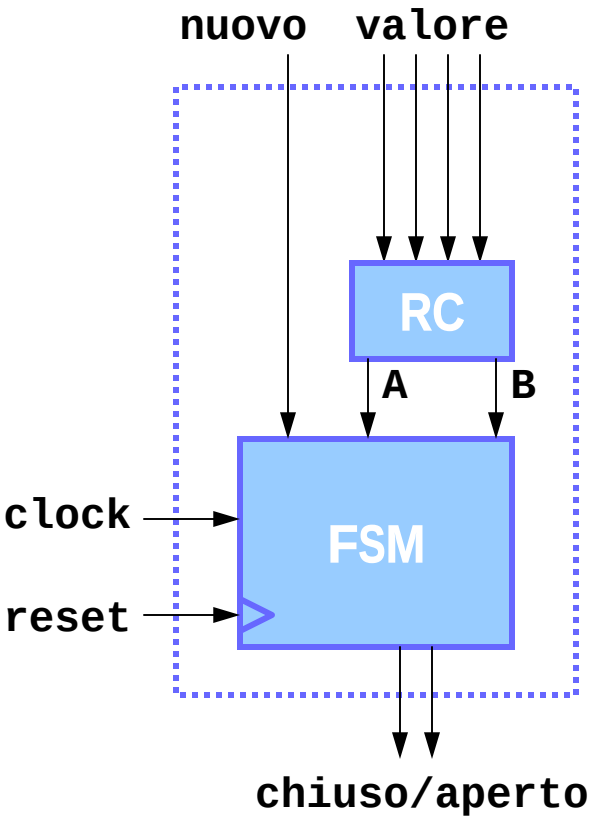
reset	1 bit	**nuovo**	1 bit
valore	4 bit	**chiuso/aperto**	2 bit

Esempio: Codifica degli ingressi

- La rete combinatoria codifica i valori di ingresso
  - Uniformandoli a quelli per cui la macchina è progettata
    - Il funzionamento è indipendente dai valori da identificare
  - Valori da riconoscere arbitrariamente scelti in {0, 1, 2}
    - 2 bit (minimo) o 3 bit (massimo) di ingresso



Esempio: Codifica degli ingressi



Esempio: Sequenza (3, 4, 8)

valore

$V_3V_2V_1V_0$	A	B
0 0 1 1	0	0
0 1 0 0	0	1
1 0 0 0	1	0

$A = P_0*(V_3+V_2'+V_1+V_0)$
$B = P_0*(V_3'+V_2+V_1+V_0)$
$P_0 = (V_3+V_2+V_1'+V_0')$

Esempio: Sequenza (7, 0, 15)

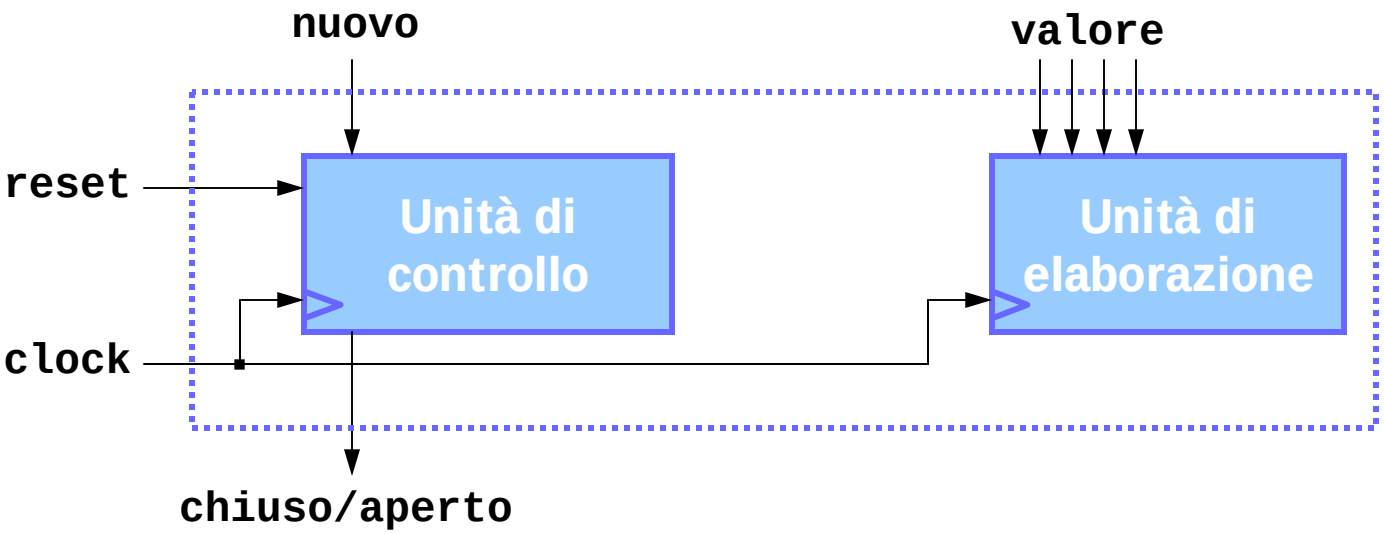
valore

$V_3V_2V_1V_0$	A	B
0 1 1 1	0	0
0 0 0 0	0	1
1 1 1 1	1	0

$A = P_0*(V3+V2+V1+V0)$
$B = P_0*(V3'+V2'+V1'+V0')$
$P_0 = (V3+V2'+V1'+V0')$

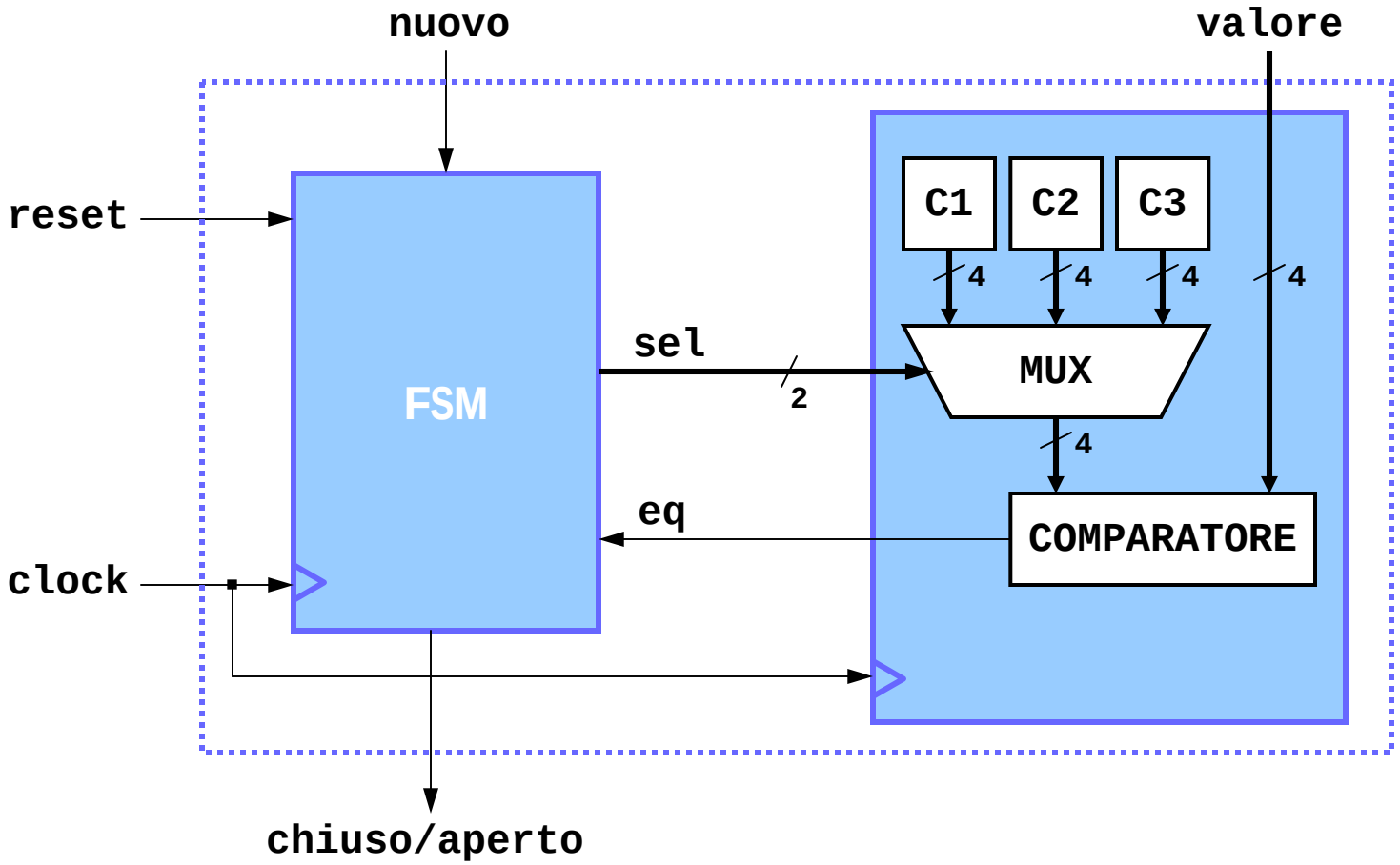
Esempio: Controllo e Elaborazione

- È possibile realizzare funzionalità programmabili
- Le configurazioni sono memorizzate e confrontate con i valori immessi passo dopo passo
  - Flessibilità molto alta
  - Maggior compessità di realizzazione



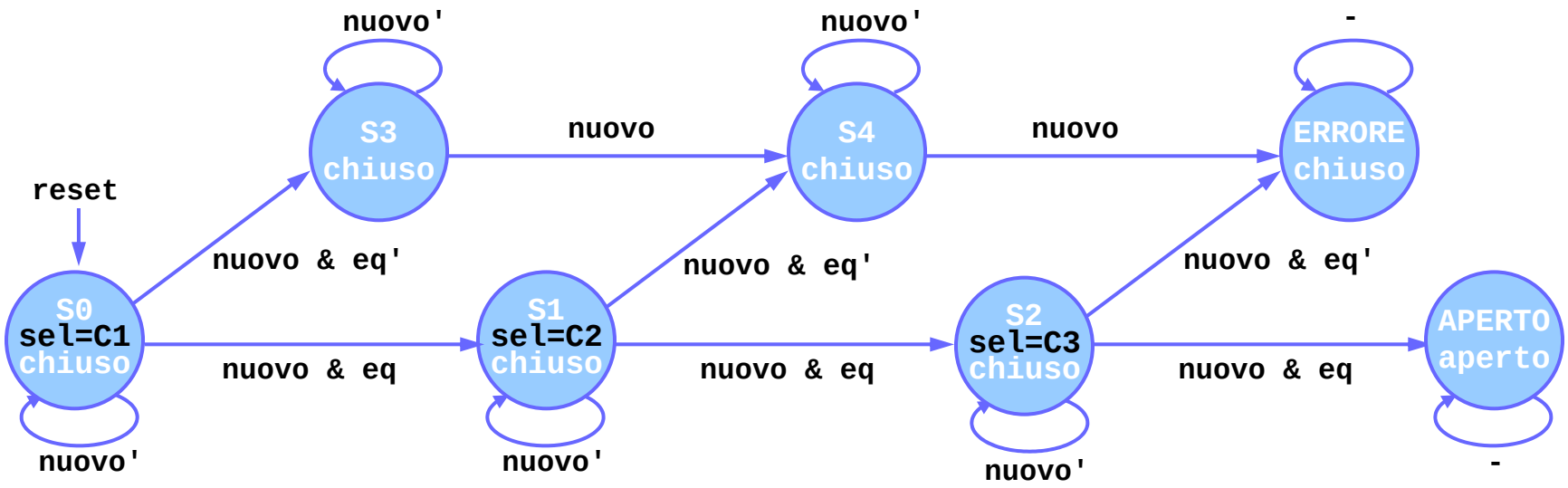
Esempio: Struttura Circuitale

Architettura



Esempio: Macchina a Stati

Diagramma degli stati



Esempio: Macchina a Stati

Tabella degli stati

reset	nuovo	eq	stato-presente	stato-futuro	sel	uscita
1	-	-	-	S0	C1	chiuso
0	0	-	S0	S0	C1	chiuso
0	1	1	S0	S1	C2	chiuso
0	1	0	S0	S3	-	chiuso
0	0	-	S1	S1	C2	chiuso
0	1	1	S1	S2	C3	chiuso
0	1	0	S1	S4	-	chiuso
0	0	-	S2	S2	C3	chiuso
0	1	1	S2	APERTO	-	aperto
0	1	0	S2	ERRORE	-	chiuso
0	0	-	S3	S3	-	chiuso
0	1	-	S3	S4	-	chiuso
0	0	-	S4	S4	-	chiuso
0	1	-	S4	ERRORE	-	chiuso
0	-	-	ERRORE	ERRORE	-	chiuso
0	-	-	APERTO	APERTO	-	aperto

Esempio: Macchina a Stati - Codifica

- Nella tabella degli stati compaiono valori simbolici
  - Devono essere tradotti in bit
  - Si hanno molti gradi di libertà
    - Numero di bit
    - Valore
  - Ad ogni possibile scelta corrisponde una rappresentazione circuitale differente anche in costo
- SI tratta di scegliere una opportuna codifica per tutti i valori simbolici che compaiono nella descrizione del comportamento

Esempio: Macchina a Stati - Codifica

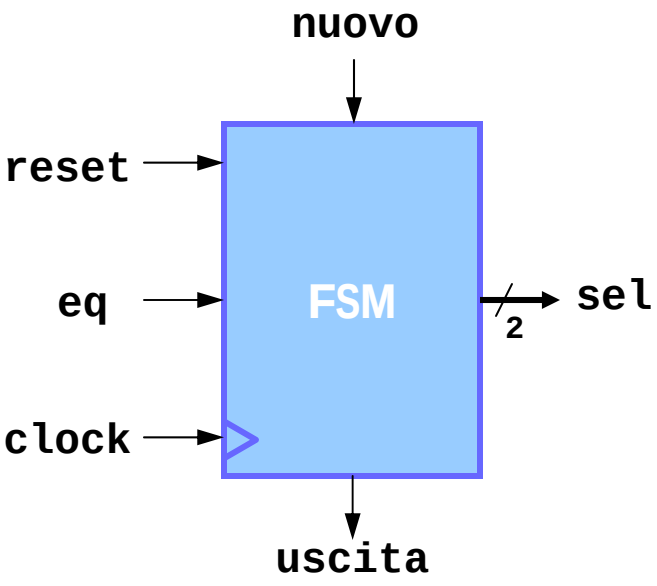
- Stati: **`S0`**, **`S1`**, **`S2`**, **`S3`**, **`S4`**, **`APERTO`**, **`ERRORE`**
  - Codifica minima: 7 stati richiedono 3 bit (scelta)
    `001`, **`010`**, **`100`**, **`000`**, **`011`**, **`101`**, **`110`**
  - Codifica One-hot: 7 stati richiedono 7 bit
    `0000001`, **`0000010`**, **`0000100`**, **`0001000`**, **`0010000`**, **`0100000`**, **`1000000`**
- Uscita del **`MUX`**: **`C1`**, **`C2`**, o **`C3`**
  - Codifica minima: 3 valori richiedono 2 bit
    `00`, **`01`**, **`10`**
  - Codifica minima: 3 valori richiedono 3 bit (scelta)
    `001`, **`010`**, **`100`**
- Uscita: **`aperto`**, **`chiuso`**
  - Codifica minima: 2 valori richiedono 1 bit (scelta)
    `0`, **`1`**
  - Codifica minima: 2 valori richiedono 2 bit
    `01`, **`10`**

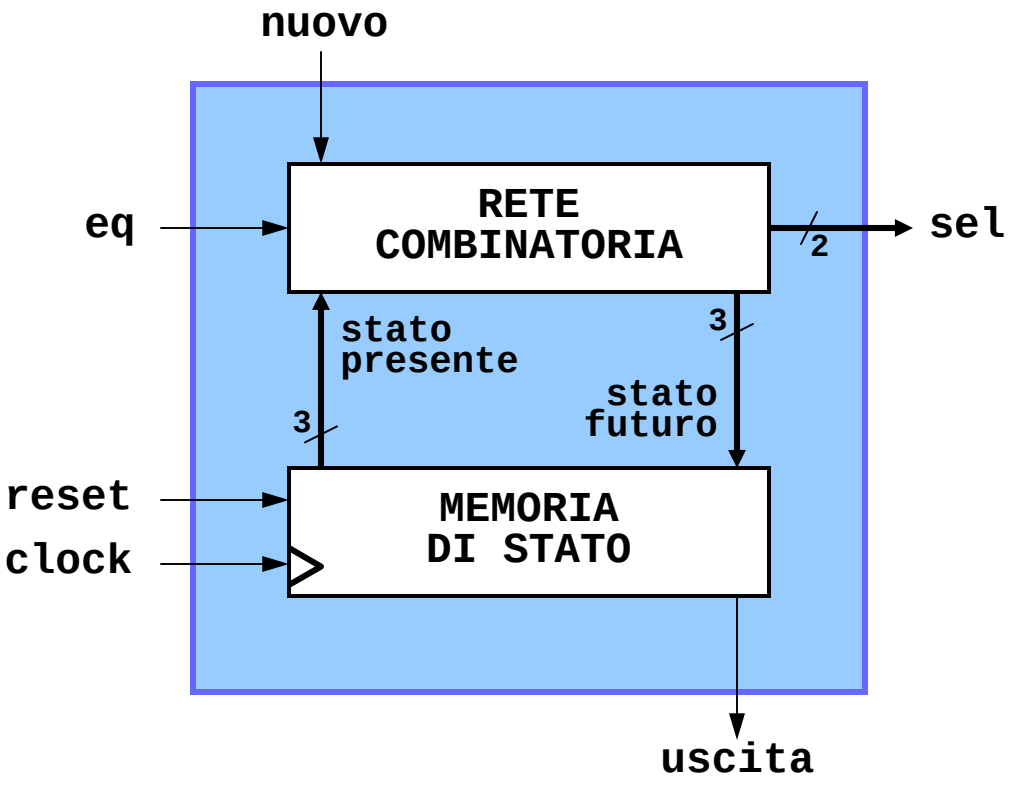
Esempio: Macchina a Stati - Codifica

Tabella delle transizioni

reset	nuovo	eq	stato-presente	stato-futuro	sel	uscita
1	-	-	---	001	001	0
0	0	-	001	001	001	0
0	1	1	001	010	010	0
0	1	0	001	000	---	0
0	0	-	010	010	010	0
0	1	1	010	100	100	0
0	1	0	010	011	---	0
0	0	-	100	100	100	0
0	1	1	100	110	---	1
0	1	0	100	101	---	0
0	0	-	000	000	---	0
0	1	-	000	011	---	0
0	0	-	011	011	---	0
0	1	-	011	101	---	0
0	-	-	101	101	---	0
0	-	-	110	110	---	1

Esempio: Unità di Controllo





Esempio: Unità di Controllo

- Passi per l'implementazione della unità di controllo (in generale, di una Macchina a Stati Finiti)
  - Separare la Rete Combinatoria e la memoria
  - Identificare i dispositivi di memoria da utilizzare
    - Modificano la rete combinatoria
  - Trasformare la tabella delle transizioni in tabella delle eccitazioni
    - Nuova tabella degli implicanti che descrive la funzione della rete combinatoria
  - Sintetizzare la rete combinatoria a più uscite utilizzando le procedure note
  - Collegare nuovamente la memoria alla rete combinatoria sintetizzata