uniVR - Dipartimento Informatica

Elaborato SIS - Laboratorio Architettura degli Elaboratori

Progettazione Bancomat

A.A. 2020/2021

Gruppo di lavoro:
Amos Lo Verde (vr456585)
Nicolò Piccoli (vr459373)
Simone Moratti (vr456083)

Indice

Specifiche

Si progetti il circuito sequenziale che controlla l'erogazione di denaro di un bancomat.
Il circuito ha 4 ingressi nel seguente ordine:

- BANCOMAT_INSERITO (1 bit)
- CODICE (4 bit)
- CASH_RICHIESTO (10 bit)
- CASH_DISPONIBILE (16 bit)

Gli output sono i seguenti e devono seguire il seguente ordine:

- REINSERIRE_CODICE (1 bit)
- ABITILAZIONE_EROGAZIONE (1 bit)
- BLOCCO_BANCOMAT (1 bit)
- CASH_DA_DA_EROGARE (10 bit)

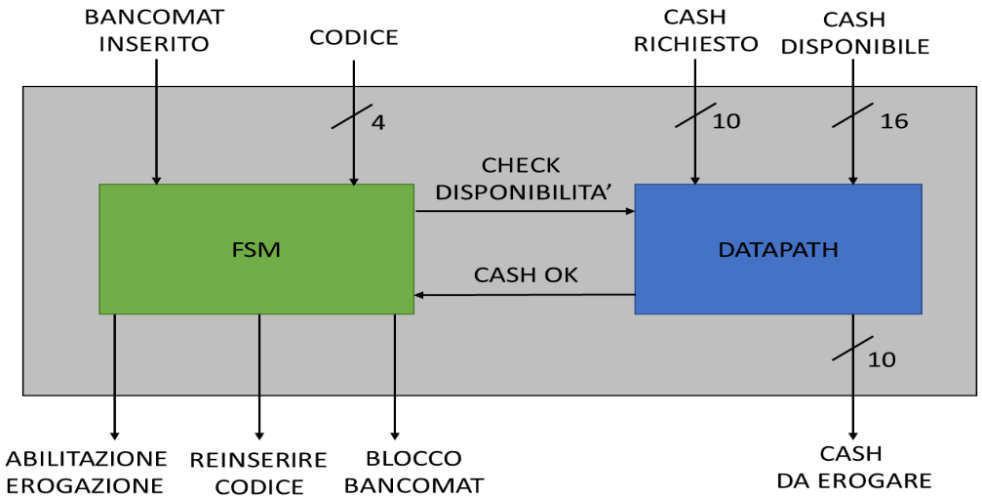
Input e output devono essere definiti nell'ordine sopra specificato (da sinistra verso destra).
Il meccanismo è guidato come segue:

- Il segnale di ingresso BANCOMAT_INSERITO (da considerare derivante da un circuito esterno che rileva la presenza di un bancomat valido inserito nel macchinario) se uguale a 1 abilita la codifica dei numeri inseriti tramite il segnale di ingresso CODICE. Se uguale a 0, disabilita (pone a zero) tutte le uscite del circuito.
- L'analisi del CODICE inizia soltanto dopo che il BANCOMAT è stato inserito. Non è possibile inserire il BANCOMAT e la prima cifra del codice nello stesso momento.
- Una volta che il bancomat è stato inserito, viene inserito nel circuito il codice di autenticazione tramite il segnale di ingresso CODICE composto da 3 numeri inseriti in 3 istanti consecutivi, con range 0..9, codificati quindi con 4 bit.
- Una volta accertato che la sequenza numerica corrisponde a 5 5 0, il circuito riceve l'ammontare del cash richiesto dall'ingresso CASH_RICHIESTO (ammontare da 0 a 1023 euro, codificato con 10 bit) e attiva il controllo della disponibilità di banconote nella cassaforte, tramite il segnale interno CHECK_DISPONIBILITÀ (1 bit).
  Il controllo verifica se il cash richiesto è inferiore a 1/4 del cash disponibile in

cassaforte, quest'ultimo ricevuto dal segnale di ingresso CASH_DISPONIBILE. Se inferiore allora il circuito abilita il segnale interno CASH_OK (1 bit), il quale fa abilitare il segnale di uscita ABILITAZIONE_EROGAZIONE, e riporta sul segnale di uscita CASH_DA_EROGARE l'importo richiesto. Altrimenti, tutti questi segnali rimangono posti a 0.

- Se il codice viene inserito in modo errato, il circuito abilita l'uscita REINSERIRE_CODICE.
- REINSERIRE_CODICE viene alzato solo al termine dell'inserimento dei codici che compongono il pin. Per esempio, se il codice inserito fosse 123, la porta viene messa ad 1 solo al termine dell'inserimento dell'intero codice e non già alla prima cifra inserita.
- Se il codice viene inserito in modo errato per 3 volte consecutive, il circuito abilita l'uscita BLOCCO_BANCOMAT.

Lo schema generale del circuito deve rispettare la FSMD riportata di seguito:



- È possibile aggiungere degli ulteriori segnali interni per la comunicazione tra FSM e DATAPATH
- Le porte di input e di output devono rispettare l'ordine definito ed essere collegate al rispettivo sotto modulo (i.e. BANCOMAT INSERITO e CODICE sono input della FSM, mentre CASH RICHIESTO e CASH DISPONIBILE input del DATAPATH).
- Il DATAPATH deve essere unico: se volete definire più DATAPATH, questi devono essere inglobati in un unico modello.

Architettura generale del circuito

Il circuito è composto da una **FSM** (controllore) e un **DATAPATH** (elaboratore) che comunicano tra loro.

I file che contengono la rappresentazione di queste componenti sono presenti nella cartella con i nomi di *CONTROLLORE.blif* e *DATAPATH.blif*.

Il file che permette il collegamento tra la macchina a stati finiti e l'elaboratore ha il nome di *FSMD.blif*.

Nelle pagine seguenti verranno descritte nel dettaglio analizzando le componenti utilizzate.

Diagramma del controllore

Il **controllore** del bancomat è una macchina a stati finiti del tipo **Mealy**.

Si presentano i seguenti segnali:

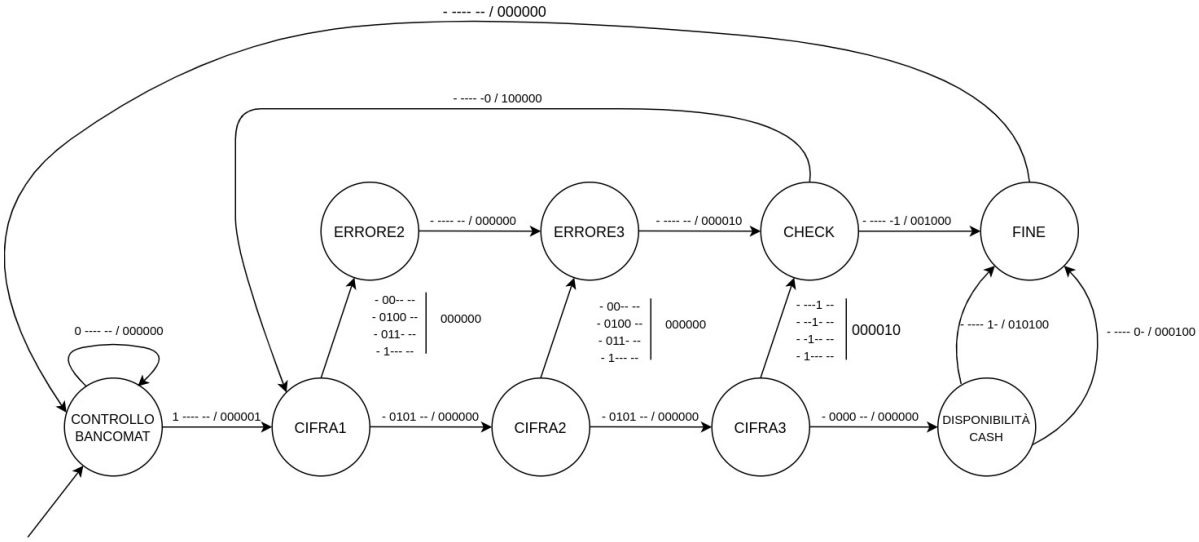
Segnali di input	**Segnali di output**
BANCOMAT_INSERITO[1] CODICE[4] CASH_OK[1] TENTATIVI_ESAURITI[1]	REINSERIRE_CODICE[1] ABILITA_EROGAZIONE[1] BLOCCO_BANCOMAT[1] CHECK_DISPONIBILITA[1] CONTROLLO_TENTATIVI[1] RESET[1]

La **FSM** presenta 9 stati:

- CONTROLLO BANCOMAT: qui viene controllato se il bancomat è inserito o se bisogna ancora inserirlo. Nel momento in cui viene inserito ci si sposta nello stato **cifra1** e si alza il segnale di RESET, altrimenti si rimane in attesa di inserimento.
- CIFRA1: qui si riceve la prima cifra, nel caso sia 0101 (5) il controllore passa in **cifra2**, altrimenti passa in **errore2**.
- CIFRA2: qui si riceve la seconda cifra, nel caso sia 0101 (5) il controllore passa in **cifra3**, altrimenti passa in **errore3**.
- CIFRA3: qui si riceve la terza cifra, nel caso sia 0000 (0) il controllore passa in **disponibilita_cash**, altrimenti passa in **check**.
- ERRORE2: qui qualunque sia la cifra inserita il controllore passa in **errore3**.
- ERRORE3: qui qualunque sia la cifra inserita il controllore passa in **check**.
- CHECK: qui si presentano due casi:
  1. Se in input il segnale TENTATIVI_ESAURITI è 1, allora il controllore passa allo stato **fine** e alza il segnale di **blocco_bancomat** a 1.
  2. Se in input il segnale TENTATIVI_ESAURITI è 0, allora il controllore passa allo stato **cifra1** e alza il segnale di **reinserire_codice** a 1.
- DISPONIBILITA CASH: questo stato significa che l'inserimento del codice è andato a buon fine, tramite il segnale CHECK_DISPONIBILITA avvio il processo di verifica di disponibilità del denaro nel **DATAPATH**:

1. Se viene restituito CASH_OK con valore 1, allora il denaro è disponibile, di conseguenza viene alzato il segnale ABILITA_EROGAZIONE e il controllore passa in **fine**.
2. Se viene restituito CASH_OK con valore 0, allora il denaro non è disponibile, di conseguenza non viene alzato il segnale ABILITA_EROGAZIONE e il controllore passa in **fine**.

- FINE: in questo stato la **FSM** si riporta in un ciclo di clock allo stato iniziale in attesa che venga inserito un nuovo bancomat.



Architettura del Datapath

Il **DATAPATH** si sviluppa in due parti:

1. La prima ha come obiettivo la conta dei tentativi di inserimento del codice;
2. La seconda ha lo scopo di controllare se l'ammontare dei soldi richiesti (CASH_RICHIESTO) è minore o uguale a $\frac{1}{4}$ del denaro disponibile (CASH_DISPONIBILE).

-Conta dei tentativi:

Le componenti utilizzate sono:

- Un MULTIPLEXER (MUX1): composto da 2 ingressi aventi le costanti 0 e 1 (a 2 bit), con un selettore (CONTROLLA_TENTATIVI) che esce dalla **FSM** alzato a 1 se il codice inserito è errato;
- Un MULTIPLEXER (MUX2): composto da 2 ingressi dove in uno passa la costante 0 e nell'altro passa il valore memorizzato nel registro (a 2 bit entrambi), con un selettore (RESET) che esce dalla **FSM** alzato a 1 solo quando viene inserito il bancomat;
- Un SOMMATORE binario che prende in input il segnale del MUX1 e il segnale proveniente dal MUX2;
- Il REGISTRO (REG) è inizializzato a 0 e sovrascrive al suo interno il risultato della somma effettuata a ogni controllo;
- Un COMPARATORE che confronta l'output della somma e lo eguaglia alla costante 11.

Nel caso in cui l'output del comparatore sia 1, il segnale TENTATIVI_ESAURITI causa il blocco del bancomat.

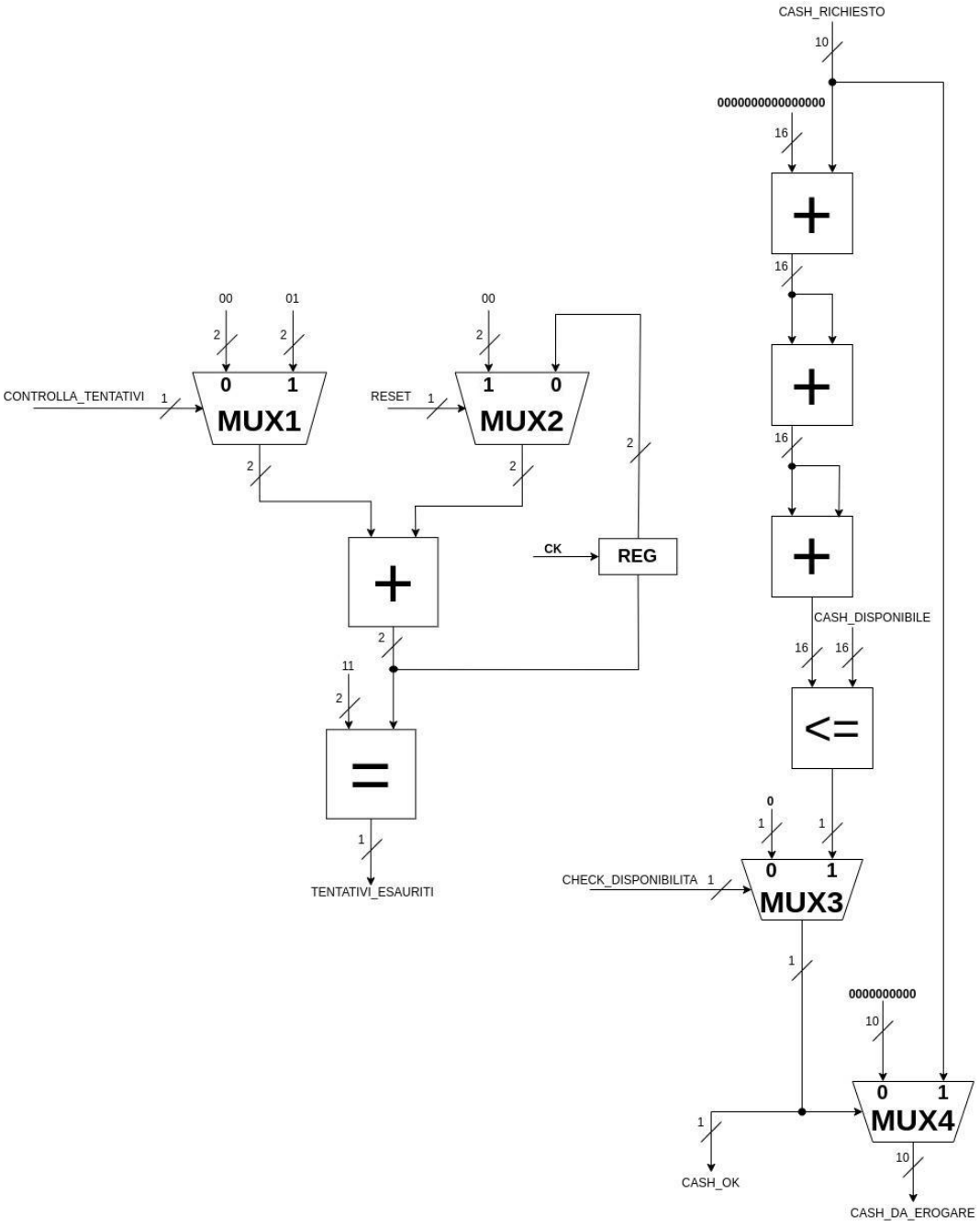
-Controllo del cash:

Le componenti utilizzate sono:

- Un SOMMATORE: ha come ingressi il denaro che si desidera prelevare (CASH_RICHIESTO) e una costante 0 a 16 bit, ciò permette di rendere CASH_RICHIESTO (inizialmente a 10 bit) in 16 bit;
- Due SOMMATORI in serie:
  - Il primo prende in input l'output sdoppiato a 16 bit del sommatore precedente.
  - Il secondo compie la stessa azione, ma con l'output del secondo sommatore specificato.

- Un COMPARATORE: riceve il risultato dell'ultimo sommatore e controlla che sia minore o uguale al denaro disponibile sul conto (CASH_DISPONIBILE);
- Un MULTIPLEXER (MUX3): ha come input una costante 0 a 1 bit e il risultato del comparatore, il selettore è CHECK_DISPONIBILITA che dipende dalla correttezza del codice inserito dall'utente;
- Un MULTIPLEXER (MUX4): ha come input una costante 0 a 10 bit e il CASH_RICHIESTO, il selettore invece è l'output del MUX3.

Gli output di questa fase sono CASH_OK (il quale raccoglie il risultato del MUX3) che torna alla FSM e CASH_DA_EROGARE che proviene dal MUX4.



Statistiche del circuito

5.1 Prima della ottimizzazione

Le statistiche del circuito prima di effettuare l'ottimizzazione per area indicano:

```
sis> print_stats
FSMD            pi=31   po=13   nodes=235       latches= 6
lits(sop)= 787
sis>
```

- **Numero di nodi:** 235
- **Numero di letterali:** 787

5.2 Dopo la ottimizzazione

L'ottimizzazione è stata eseguita lanciando due volte il comando SOURCE SCRIPT.RUGGED che permette la distruzione e ricostruzione della **FSMD**.

Inoltre è stato anche lanciato successivamente il comando FX che permette di sostituire un nodo interno con un insieme di nodi la cui funzionalità sia equivalente a quella del nodo sostituito, però solo quando una parte è comune a due nodi.

```
sis> source script.rugged
sis> print_stats
FSMD            pi=31   po=13   nodes= 34       latches= 6
lits(sop)= 284
sis> source script.rugged
sis> print_stats
FSMD            pi=31   po=13   nodes= 32       latches= 6
lits(sop)= 283
sis> fx
sis> print_stats
FSMD            pi=31   po=13   nodes= 45       latches= 6
lits(sop)= 158
sis>
```

- **Numero di nodi:** 45
- **Numero di letterali:** 158

Mapping: gate e ritardi

Dopo aver ottimizzato il circuito lo si mappa così da visualizzare le statistiche verosimili riguardo area e ritardo. È stata assegnata la libreria **synch.genlib**.
Il circuito mappato presenta le seguenti statistiche:

```
>>> before removing serial inverters <<<
# of outputs:          19
total gate area:       2864.00
maximum arrival time: (19.40,19.40)
maximum po slack:     (-6.60,-6.60)
minimum po slack:     (-19.40,-19.40)
total neg slack:      (-304.60,-304.60)
# of failing outputs:  19
>>> before removing parallel inverters <<<
# of outputs:          19
total gate area:       2864.00
maximum arrival time: (19.40,19.40)
maximum po slack:     (-6.60,-6.60)
minimum po slack:     (-19.40,-19.40)
total neg slack:      (-304.60,-304.60)
# of failing outputs:  19
# of outputs:          19
total gate area:       2816.00
maximum arrival time: (19.40,19.40)
maximum po slack:     (-6.60,-6.60)
minimum po slack:     (-19.40,-19.40)
total neg slack:      (-304.20,-304.20)
# of failing outputs:  19
```

Il **total gate area** (area) è 2864.00 mentre l'**arrival time** (cammino critico) è 19.40.

Scelte progettuali

- I segnali introdotti CONTROLLO_TENTATIVI e TENTATIVI_ESAURITI tra **FSM** e **DATAPATH** servono allo scopo di contare quanti tentativi l'utente ha utilizzato durante gli inserimenti del codice.
  Se arriva al terzo tentativo, e il codice è nuovamente errato, allora il bancomat viene bloccato.

- È stato introdotto un segnale RESET che ha lo scopo di resettare il registro a ogni inserimento del bancomat, così che i tentativi caricati precedentemente sul registro, dovuti all'uso del bancomat da una persona, vengano azzerati. In questo modo ogni persona ha 3 tentativi senza perderne alcuni dagli usi precedenti.

- Dal momento in cui il bancomat viene inserito non risulta importante specificare se sia presente o no negli stati successivi, tanto che il segnale BANCOMAT_INSERITO viene posto a **-** (*don't care*).
  Una volta finite le operazioni, ovvero l'evoluzione della macchina è arrivata allo stato **fine**, il bancomat verrà restituito alla persona; quando lo spostamento dallo stato **fine** allo stato **controllo bancomat** sarà concluso, il segnale BANCOMAT_INSERITO tornerà a essere 0 fino a quando non ne verrà inserito un altro.

- Tra lo stato **check** e **cifra1** si denota una sequenza di cifre tutte poste a **-** (*don't care*), tranne TENTATIVI_ESAURITI che risulta a 0, poiché la macchina deve avvisare l'utente di reinserire il codice in quanto non si è superato il numero di tentativi.

- Tra lo stato **check** e **fine** si denota una sequenza di cifre tutte poste a **-** (*don't care*), tranne TENTATIVI_ESAURITI che risulta a 1, poiché la macchina deve avvisare l'utente che non ha altri tentativi e quindi blocca il bancomat.

- Quando viene inserita la terza cifra corretta non eroga direttamente il cash, ma valuta con uno stato intermedio (**disponibilita_cash**).
  Se il CASH_RICHIESTO è $\frac{1}{4}$ del CASH_DISPONIBILE, allora abilita il segnale ABILITA_EROGAZIONE a 1 ed eroga il denaro, altrimenti rimane a 0 ed eroga 0.