ELABORATO SIS

ARCHITETTURA DEGLI ELABORATORI

A.A. 2020/2021 – Corso di Laurea in Informatica

Bianchini Davide (VR456697)

Bragastini Enrico (VR456374)

Mafficini Andrea (VR462441)

Sommario

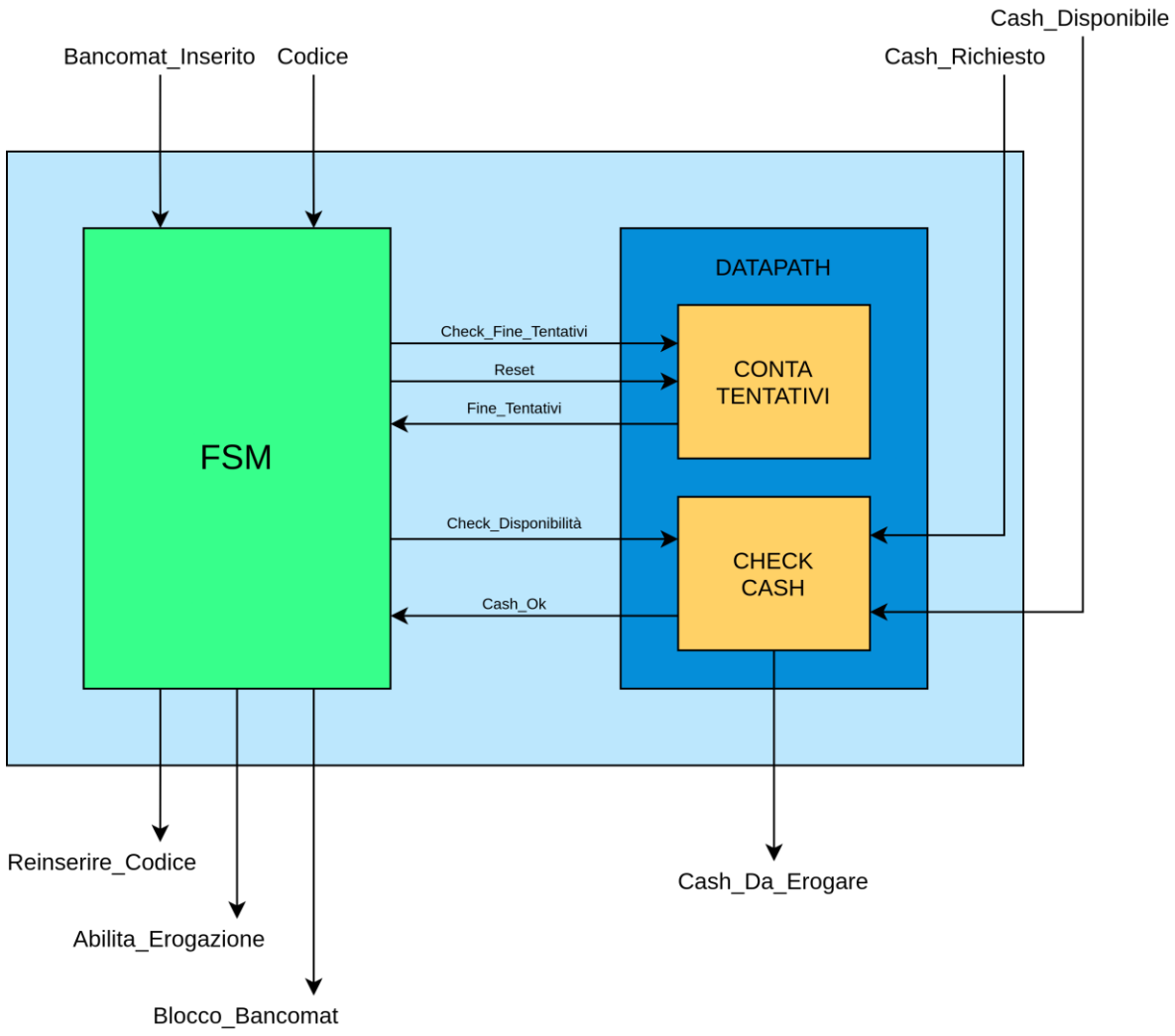
FSMD

Lo schema generale del circuito da noi utilizzato rispecchia lo schema indicato nelle specifiche del progetto.

Il Datapath, per i motivi successivamente illustrati, viene diviso concettualmente in due circuiti distinti. Questo ha comportato l'aggiunta di alcuni segnali interni di comunicazione tra la FSM e il Datapath.

In particolare è stata fatta la *scelta progettuale* di lasciare il compito del conteggio dei tentativi errati al Datapath, per ridurre così il numero di stati presenti nel controllore. Come si può quindi notare dallo schema seguente, il Datapath è composto di due sezioni: una che si occupa di avvisare la FSM nel caso in cui l'utente dovesse inserire un codice errato per due volte di seguito procedendo quindi con il terzo tentativo a disposizione, l'altra sezione si occupa di leggere il *cash richiesto* dall'utente e il *cash disponibile* nella cassaforte, verificare che il primo sia minore di ¼ del secondo, e avvisare la FSM se questa condizione è stata rispettata. In caso positivo, scriverà sull'uscita dedicata l'importo richiesto dall'utente che verrà poi erogato.

Oltre ai segnali interni previsti dalla specifica, sono stati aggiunti i segnali che comunicano con la sezione di Datapath che si occupa di gestire contare il numero di tentativi.



Controllore FSM

1. Interpretazione della specifica in linguaggio naturale

Il circuito richiesto deve controllare l'erogazione di denaro di un bancomat.

Gli ingressi sono: bancomat_inserito, codice, cash_richiesto, cash_disponibile;
Le uscite sono: reinserire_codice, abilitazione_erogazione, blocco_bancomat e cash_da_erogare.

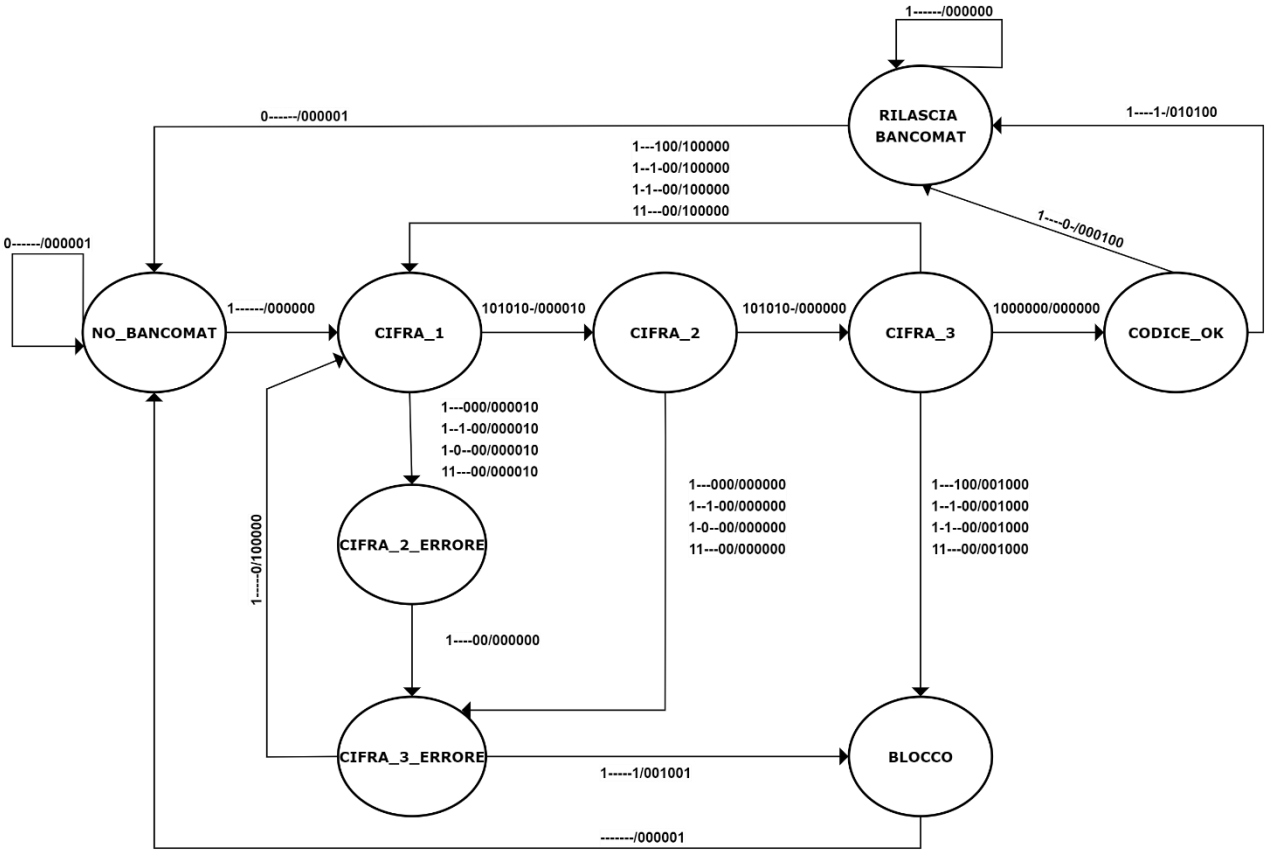
Dopo aver inserito il bancomat, verrà chiesto di inserire un pin di 3 cifre, una volta inserite tutte le cifre, il circuito deve controllare la correttezza del pin. A questo punto, verrà chiesta la quantità di denaro che si vuole prelevare. Il circuito, una volta controllato che ci siano contanti sufficienti per il prelievo, abilita l'uscita abilitazione_erogazione, erogando il denaro richiesto, il cui quantitativo viene scritto sui bit del segnale cash_da_erogare.

Si hanno 3 tentativi per l'inserimento del pin. Nel caso in cui si inserisce il pin scorretto, viene richiesto nuovamente di reinserirlo (uscita reinserire_codice a 1). Se il pin viene inserito errato per 3 volte consecutive, il bancomat si bloccherà (uscita blocco_bancomat a 1).

Se non ci sono abbastanza soldi per il prelievo, non verrà erogato alcun denaro.

Si suppone che il bancomat trattenga la tessera dell'utente per tutto il procedimento. Se il bancomat dovesse finire nello stato di blocco, ovvero dopo aver inserito il pin errato per tre volte consecutive, si suppone che la tessera venga trattenuta dallo stesso, tornando poi allo stato iniziale per procedere servendo un nuovo utente.

2. Rappresentazione in state-transition-graph (STG) Mealy



3. Scelte progettuali

Uno ad uno, analizziamo gli stati con i rispettivi ingressi e uscite.

Ingressi = {Bancomat_Inserito, Codice3, Codice2, Codice1, Codice0, Cash_Ok, Fine_Tentativi}

Uscite = {Reinserire_Codice, Abilitazione_Erogazione, Blocco_Bancomat, Check_Disponibilita, Check_Fine_Tentativi, Reset}

Stato NO_BANCOMAT: Stato in cui si attente che il bancomat venga inserito.
Finché non viene inserito il bancomat, la macchina rimane nello stato NO_BANCOMAT. Una volta inserito il bancomat, il primo bit di input sale a 1, per permettere di passare allo stato successivo: CIFRA_1.

Stato CIFRA_1: Stato in cui si attende l'inserimento della prima cifra del pin. Se nei 4 bit del codice viene inserito il numero 5 (0101) si passerà allo stato CIFRA_2.
Per ogni altro numero inserito (input diverso da 0101), si passerà allo stato CIFRA_2_ERRORE.
In entrambi i casi per scelta progettuale si abilita a 1 l'uscita Check_Fine_Tentativi, in modo tale da chiedere al datapath se l'attuale tentativo è l'ultimo a disposizione dell'utente. La risposta dal datapath verrà valutata successivamente negli stati CIFRA_3 e CIFRA_3_ERRORE.

Stato CIFRA_2: Stato in cui si attende che venga inserita la seconda cifra del pin. Se nei 4 bit del codice viene inserito il numero 5 (0101) si passerà allo stato CIFRA_3.
Per ogni altro numero inserito (input diverso da 0101), si passerà allo stato CIFRA_3_ERRORE.

Stato CIFRA_2_ERRORE: Stato in cui si attende l'inserimento della seconda cifra, sapendo che la prima è stata inserita errata. Qualsiasi sia l'input dei bit del codice (----), si passerà allo stato CIFRA_3_ERRORE. Questo perché una cifra è già stata sbagliata e non ha senso verificare cosa viene inserito in questo stato, l'inserimento del codice continuerà tenendo conto che è stato inserito errato.

Stato CIFRA_3_ERRORE: Stato in cui si attende l'inserimento della terza cifra, sapendo che nell'inserimento delle prime due c'è sicuramente un errore. Se il datapath restituisce 1 sulla linea Fine_Tentativi, l'utente ha terminato i tentativi e quindi si abilita l'uscita blocco che porterà la macchina allo stato BLOCCO.
Altrimenti, qualsiasi sia l'input del codice (----), si passerà allo stato CIFRA_1 e verrà abilitato l'output Reinserire_Codice, per permettere di reinserire il pin.

Stato CIFRA_3: se nei 4 bit del codice viene inserito il numero 0 (0000) significa che l'intero pin è stato inserito correttamente. Si passerà quindi allo stato successivo CODICE_OK. Nel caso in cui non venga inserito 0, analogamente allo stato CIFRA_3_ERRORE, valutando la linea Fine_Tentativi, si passerà allo stato CIFRA_1 oppure allo stato BLOCCO.

Stato BLOCCO: la macchina, dopo aver bloccato il bancomat, lo ritira (per scelta progettuale) e torna allo stato NO_BANCOMAT per ripartire con un nuovo prelievo.

Stato CODICE_OK: Stato in cui l'utente inserisce il denaro da prelevare e viene letto il denaro presente in cassaforte. Se la cifra inserita dall'utente, in relazione con i soldi in cassaforte, è congrua per essere prelevata, viene settato a 1 l'uscita Abilitazione_Erogazione. Per effettuare il controllo viene abilitata l'uscita Check_Disponibilita e allo stesso tempo viene valutata l'entrata Cash_Ok. (Ai calcoli ci penserà il Datapath)
Se dopo aver fatto il controllo, la cifra non risulta congrua, il bit Abilitazione_Erogazione rimane a 0.
In entrambi i casi, lo stato successivo sarà RILASCIA_BANCOMAT.

Stato RILASCIA_BANCOMAT: In questo stato la macchina aspetta che venga espulsa la carta bancomat, ovvero che venga settato a 0 il bit Bancomat_Inserito. Torna quindi allo stato NO_BANCOMAT per ripartire con un nuovo prelievo.

4. Rappresentazione in state-transition-table (STT)

Legenda (Ingressi e Uscite):

- i6: Bancomat_Inserito
- i5 - <i2: Codice
- i1: Cash_Ok
- i0: Fine_Tentativi
- o5: Reinserire_Codice
- o4: Abilitazione_Erogazione
- o3: Blocco_Bancomat
- o2: Check_Disponibilita
- o1: Check_Fine_Tentativi
- o0: Reset

STATO ATTUALE	i6	i5	i4	i3	i2	i1	i0	STATO PROSSIMO	o5	o4	o3	o2	o1	o0
NO_BANCOMAT	0	-	-	-	-	-	-	NO_BANCOMAT	0	0	0	0	0	1
NO_BANCOMAT	1	-	-	-	-	-	-	CIFRA_1	0	0	0	0	0	0
CIFRA_1	1	-	-	-	0	0	0	CIFRA_2_ERRORE	0	0	0	0	1	0
CIFRA_1	1	-	-	1	-	0	0	CIFRA_2_ERRORE	0	0	0	0	1	0
CIFRA_1	1	-	0	-	-	0	0	CIFRA_2_ERRORE	0	0	0	0	1	0
CIFRA_1	1	1	-	-	-	0	0	CIFRA_2_ERRORE	0	0	0	0	1	0
CIFRA_1	1	0	1	0	1	0	-	CIFRA_2	0	0	0	0	1	0
CIFRA_2	1	-	-	-	0	0	0	CIFRA_3_ERRORE	0	0	0	0	0	0
CIFRA_2	1	-	-	1	-	0	0	CIFRA_3_ERRORE	0	0	0	0	0	0
CIFRA_2	1	-	0	-	-	0	0	CIFRA_3_ERRORE	0	0	0	0	0	0
CIFRA_2	1	1	-	-	-	0	0	CIFRA_3_ERRORE	0	0	0	0	0	0
CIFRA_2	1	0	1	0	1	0	-	CIFRA_3	0	0	0	0	0	0
CIFRA_3	1	-	-	-	1	0	0	CIFRA_1	1	0	0	0	0	0
CIFRA_3	1	-	-	1	-	0	0	CIFRA_1	1	0	0	0	0	0
CIFRA_3	1	-	1	-	-	0	0	CIFRA_1	1	0	0	0	0	0
CIFRA_3	1	1	-	-	-	0	0	CIFRA_1	1	0	0	0	0	0
CIFRA_3	1	-	-	-	1	0	0	BLOCCO	0	0	1	0	0	0
CIFRA_3	1	-	-	1	-	0	0	BLOCCO	0	0	1	0	0	0
CIFRA_3	1	-	1	-	-	0	0	BLOCCO	0	0	1	0	0	0
CIFRA_3	1	1	-	-	-	0	0	BLOCCO	0	0	1	0	0	0
CIFRA_3	1	0	0	0	0	0	0	CODICE_OK	0	0	0	0	0	0
CIFRA_2_ERRORE	1	-	-	-	-	0	0	CIFRA_3_ERRORE	0	0	0	0	0	0
CIFRA_3_ERRORE	1	-	-	-	-	-	0	CIFRA_1	1	0	0	0	0	0
CIFRA_3_ERRORE	1	-	-	-	-	-	1	BLOCCO	0	0	1	0	0	0
CODICE_OK	1	-	-	-	-	0	-	RILASCIA_BANCOMAT	0	0	0	1	0	0
CODICE_OK	1	-	-	-	-	1	-	RILASCIA_BANCOMAT	0	1	0	1	0	0
RILASCIA_BANCOMAT	1	-	-	-	-	-	-	RILASCIA_BANCOMAT	0	0	0	0	0	0
RILASCIA_BANCOMAT	0	-	-	-	-	-	-	NO_BANCOMAT	0	0	0	0	0	1
BLOCCO	-	-	-	-	-	-	-	NO_BANCOMAT	0	0	0	0	0	1

Datapath: Contatentativi

Il Datapath Contatentativi, è un componente aggiuntivo che abbiamo deciso di aggiungere, per rendere possibile una funzionalità necessaria per il completamento e il funzionamento della macchina.
Questo Datapath viene utilizzato per contare ciclicamente quante volte viene sbagliato il codice d'ingresso, tramite un registro. Il registro viene incrementato all'inserimento della prima cifra del pin. Quando il registro avrà assunto il valore 3 e il codice viene nuovamente sbagliato, significa che è stato sbagliato per la terza volta. La FSMD manda quindi in blocco il bancomat e non lo rilascia.
In seguito saranno elencati i componenti utilizzati per la creazione di questa sezione di datapath con le relative specifiche di ogni componente:

- **Multiplexer a 2 Bit:** Il multiplexer a due bit d'ingresso viene utilizzato concettualmente per passare una costante di 2 bit al sommatore. Il bit di Check_Fine_Tentativi entra come input di selezione nel multiplexer, il quale lo riceve e se quest'ultimo è a 1 manda in output la costante 01, se invece il valore di Check_Fine_Tentativi è a 0 manderà in output la costante 00.
  Per semplificazione pratica, questo MUX è stato sostituito mettendo in AND il segnale Check_Fine_Tentativi e il bit meno significativo della costante da mandare al sommatore.
- **Sommatore a 2 Bit:** La funzionalità del sommatore è di prendere in ingresso l'output del multiplexer e sommarlo al valore interno del Registro, una volta fatto questo l'output diventerà l'input del registro, quindi il suo nuovo valore.
- **Multiplexer a 2 Bit (Reset):** questo multiplexer si intrapone tra il sommatore e il registro. Se il segnale di selezione di questo MUX, ovvero il segnale di Reset vale 0, allora l'output del sommatore viene lasciato entrare nel Registro, altrimenti viene inserito 00 nel Registro, resettandolo.
- **Registro a 2 Bit:** Il registro viene utile per memorizzare un valore preso in ingresso, in questo caso dal sommatore. Viene poi utilizzato sia in un passaggio successivo di comparazione, sia come addendo in ingresso al sommatore.
- **Comparatore a 2 bit:** Il comparatore è l'ultimo componente del nostro Datapath contatentativi. Il suo scopo è di prendere in ingresso il valore a due bit del Registro e compararlo con il valore binario 11. Nel caso in cui i due valori coincidessero, viene abilitato il segnale Fine_Tentativi in ingresso alla FSM, il quale poi persiste finché nel registro rimane il valore 11, ovvero finché non viene resettato mediante l'apposito segnale. In caso contrario, invece, lascia a 0 il bit di Fine_Tentativi.

Datapath: Check_Cash

Questa sezione del datapath è quella che si occupa di verificare che il quantitativo di denaro richiesto dall'utente (Cash_R) sia inferiore di ¼ del denaro presente nella cassaforte del Bancomat (Cash_D).

La verifica da effettuare deve rispettare la seguente disequazione:

$$Cash\_R < \frac{Cash\_D}{4} \rightarrow \ 4 \cdot Cash\_R < Cash\_D$$

L'importo di Cash_R viene moltiplicato per 4 utilizzando uno *Shifter Register* a 12 bit che esegue lo spostamento dei bit verso sinistra di due posizioni. L'input a questo shifter è di 10 bit e l'output è di 12, in quanto per quadruplicare Cash_R si spostano verso sinistra di due posizioni tutti i suoi bit, aggiungendo due zeri a destra.

Esempio: Cash_R: 0110110101_2 (= 437_{10}) → Cash_R (shift): $0110110101\boxed{00}_2$ (=1748_{10})

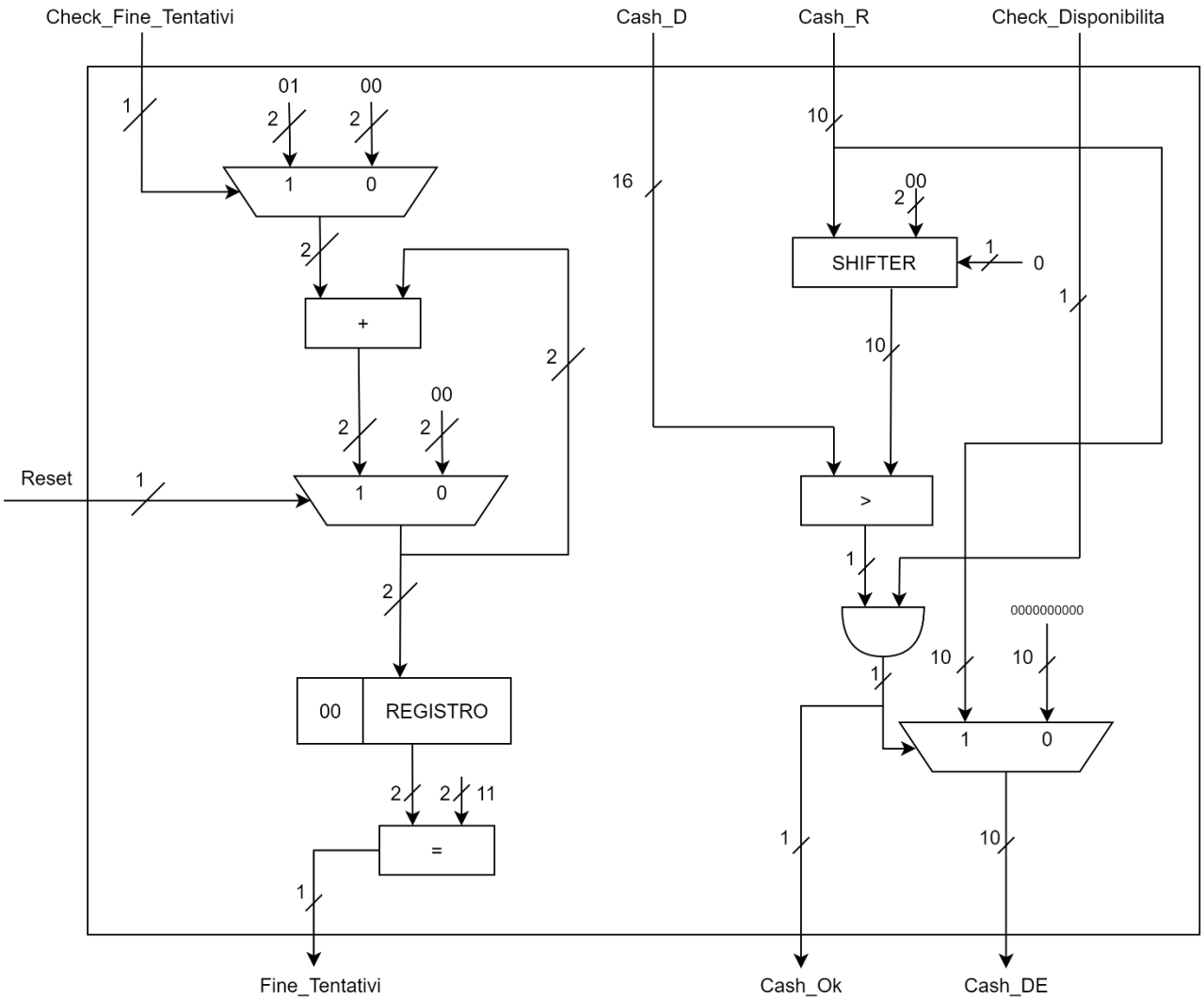
L'importo che esce dallo Shifter viene quindi confrontato con Cash_D utilizzando un comparatore di maggioranza. Quindi se $Cash\_D > 4 \cdot Cash_{\mathrm{R}}$ il comparatore mette 1 in output. Infine se l'AND tra l'uscita del comparatore e l'input Check_Disponibilità risulta vera, Cash_Ok viene messo a 1 e su Cash_DE (cash da erogare) viene riportato il cash richiesto dall'utente.

Componenti utilizzati:

- **Shifter a SX di 2 Bit**: Il componente utilizza 12 registri. L'input da 10 bit viene trascritto sui primi 10 registri "a sinistra", ovvero i più significativi, moltiplicando così il valore per 4. I due bit restanti, i meno significativi, vengono riportati a 0. L'entrata di 10 bit corrisponderà quindi a un'uscita a 12 bit.
- **Comparatore di maggioranza**: Il componente prende in ingresso due valori a 16 bit (Cash_R di 12 bit, viene portato a 16 bit aggiungendo 4 zeri a sinistra). Il comparatore sfrutta l'operatore *xor* per verificare se il primo valore inserito (Cash_D) è maggiore del secondo (4 · Cash_R). Restituisce 1 in uscita se la condizione è verificata.
- **Porta AND**: Viene sfruttata per determinare il valore di Cash_Ok. In ingresso alla porta ci sono il segnale d'uscita del comparatore e il segnale Check_Disponibilita. Cash_Ok viene quindi messo a 1 solo se la FSM chiede di fare il controllo (mediante il segnale Check_Disponibilita) e se i valori di Cash_R e Cash_D rispettano la condizione richiesta.
- **Multiplexer**: viene usato da un punto di vista schematico per far in modo che il segnale Cash_DE riporti il cash da erogare solamente quando Cash_Ok vale 1, altrimenti riporta tutti zeri. Per una questione di semplicità, è stato scelto di sostituirlo nel relativo file *.blif* con una serie di controlli AND tra Cash_Ok e ogni singola cifra di Cash_R.

Rappresentazione Grafica del Datapath:

I componenti di sinistra del Datapath costituiscono il *Contatentativi*, mentre i componenti di destra rappresentano il *Check_Cash*.



Ottimizzazione, Mapping e Statistiche

In seguito a una minimizzazione per area, effettuata mediante la libreria *synch.genlib*, con i comandi:

- `source script.rugged` Per minimizzare
- `map -m 0` Per lanciare l'ottimizzazione
- `map -s` Per visualizzare e valutare i risultati ottenuti

Abbiamo ottenuto i seguenti parametri:

Risultati del mapping

```
sis> map -s
>>> before removing serial inverters <<<
# of outputs:          19
total gate area:       2936.00
maximum arrival time: (24.40,24.40)
maximum po slack:     (-5.20,-5.20)
minimum po slack:     (-24.40,-24.40)
total neg slack:      (-336.40,-336.40)
# of failing outputs:  19
>>> before removing parallel inverters <<<
# of outputs:          19
total gate area:       2936.00
maximum arrival time: (24.40,24.40)
maximum po slack:     (-5.20,-5.20)
minimum po slack:     (-24.40,-24.40)
total neg slack:      (-336.40,-336.40)
# of failing outputs:  19
# of outputs:          19
total gate area:       2936.00
maximum arrival time: (24.40,24.40)
maximum po slack:     (-5.20,-5.20)
minimum po slack:     (-24.40,-24.40)
total neg slack:      (-336.40,-336.40)
# of failing outputs:  19
```

Statistiche prima dell'ottimizzazione

```
sis>print_stats
FSMD            pi=31   po=13   nodes= 60       latches= 6
lits(sop)= 336
```

Statistiche dopo l'ottimizzazione

```
sis>print_stats
FSMD            pi=31   po=13   nodes= 33       latches= 6
lits(sop)= 228
```