UNIVERSITÀ di VERONA | Dipartimento di INFORMATICA

ELABORATO SIS
ARCHITETTURA DEGLI ELABORATI

Università degli Studi di Verona
Corso di laurea in Informatica
Anno scolastico 2020-2021

Elaborato: Bancomat

STUDENTI:
Iliescu Michele Eduardo (VR460003) e Pagliarusco Eleonora (VR446508)

STRUTTURA DELL' ELABORATO

Il nostro progetto è composto da:

1)Introduzione
2)FSM (Final State Machine)
3)Datapath
4)FSMD
5)Mapping tecnologico in SIS

Spieghiamo la struttura di queste fasi:

1) Introduzione:

Abbiamo realizzato il circuito sequenziale che controlla l'erogazione di denaro di un bancomat.

Quest'ultimo ha 4 ingressi nel seguente ordine:

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

I suoi output sono i seguenti e hanno il seguente ordine:

- REINSERIRE_CODICE (1 bit)
- ABILITAZIONE_EROGAZIONE (1 bit)
- BLOCCO_BANCOMAT (1 bit)
- CASH_DA_EROGARE (10bit)

Spiegazione del funzionamento:

- 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.

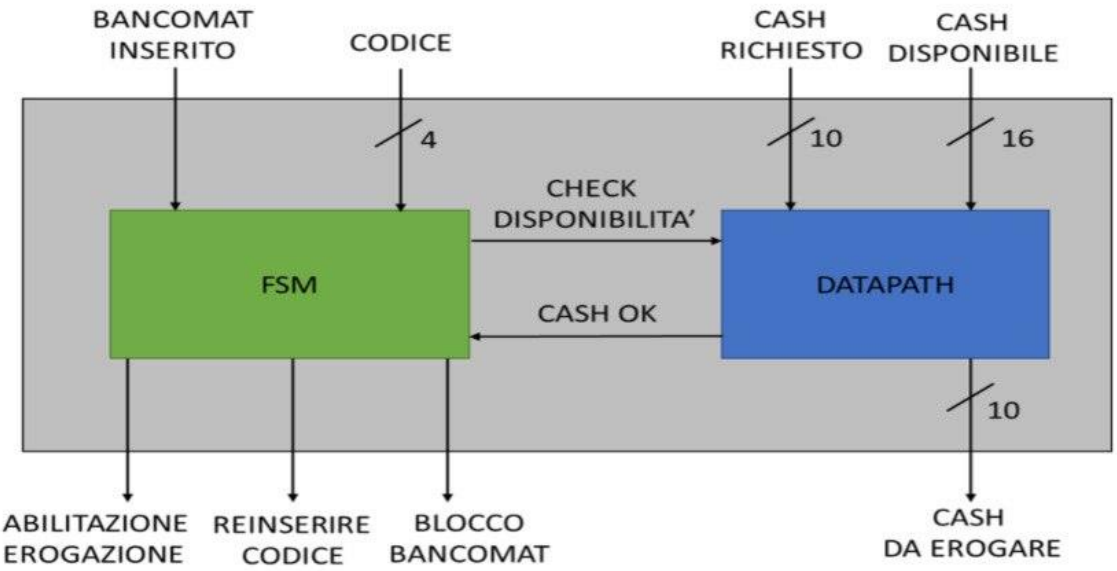
- Una volta accertato che la sequenza numerica corrisponde a 5 5 0, il circuito riceve l'ammontare del cash richiesto dall'ingresso CASH_RICHIESTO (ammontare da 0 a 1023 euro, codificato con 10 bit) e attiva il controllo della disponibilità di banconote nella cassaforte, tramite il segnale interno CHECK_DISPONIBILITA (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.

Schema generale del circuito

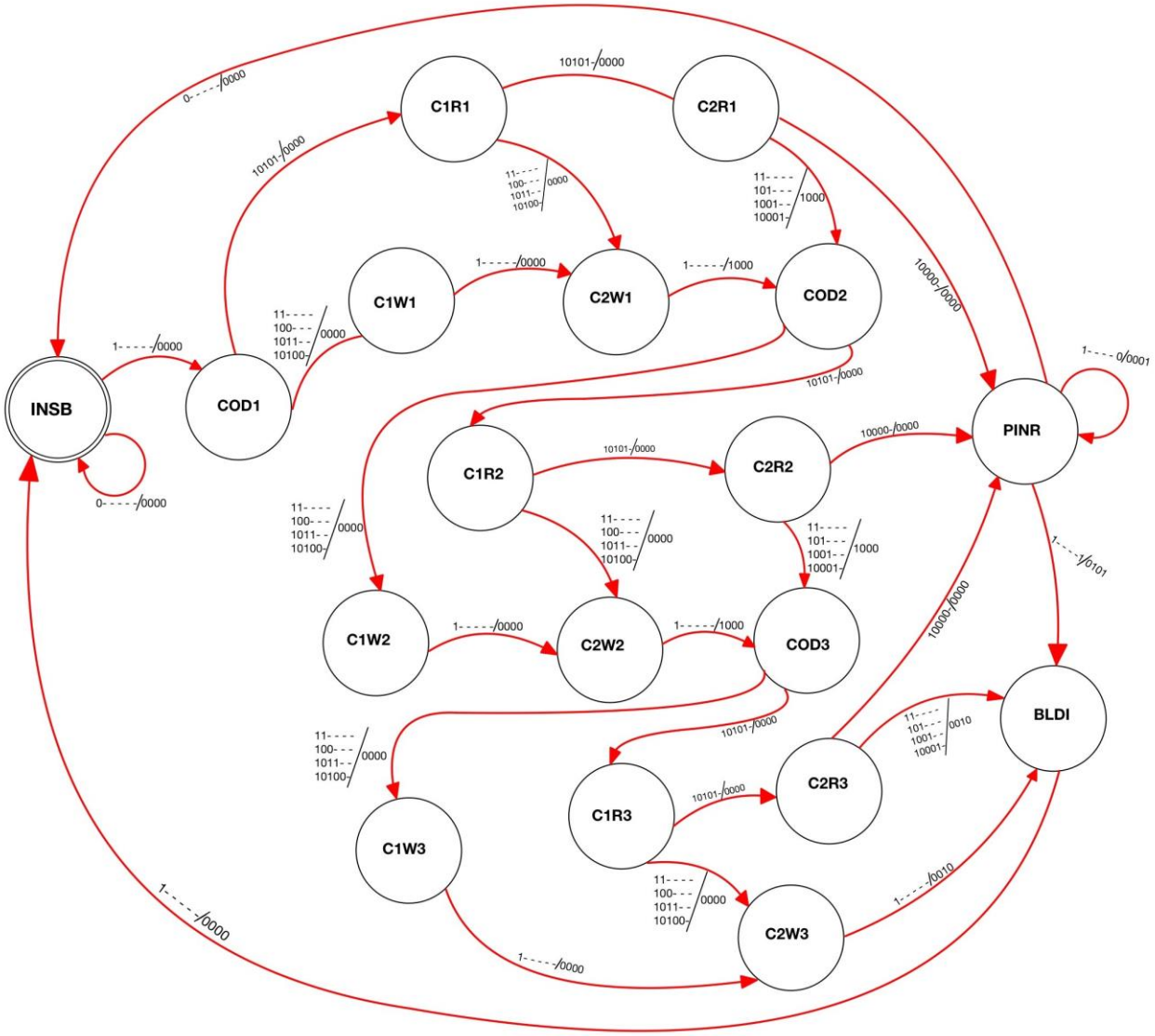


In SIS:

- BANCOMAT_INSERITO (1bit) = “BC”
- CODICE (4bit) = “C3”, “C2”, “C1”, “C0”
- CASH_OK (1bit) = “OK”
- CASH_RICHIESTO (10bit) = “X9”, “X8”, “X7”, “X6”, “X5”, “X4”, “X3”, “X2”, “X1”
- CASH_DISPONIBILE (16bit) = “Y15”, “Y14”, “Y13”, “Y12”, “Y11”, “Y10”, “Y9”, “Y8”, “Y7”, “Y6”, “Y5”, “Y4”, “Y3”, “Y2”, “Y1”, “Y0”
- REINSERIRE_CODICE (1bit) = “RE”
- ABILITÀ_EROGAZIONE (1bit) = “AB”
- BLOCCO_BANCOMAT (1 bit) = “BLK”
- CASH_DA_EROGARE (10bit) = “Q9”, “Q8”, “Q7”, “Q6”, “Q5”, “Q4”, “Q3”, “Q2”, “Q1”, “Q0”

2) FSM (Final State Machine)

STG (State Transition Graph)



Presentazione degli stati

- **INSB** è lo stato iniziale in cui viene controllato se il bancomat (fisico) è stato inserito. Rappresenta il segnale di input BANCOMAT_INSERITO, in cui se è uguale a 1 passa allo stato successivo, ossia COD1. Se invece è 0 rimane nello stato attuale. Le uscite rimangono poste a 0.

- **COD1** è lo stato che rappresenta il primo tentativo di inserimento del codice (per quanto riguarda la prima cifra). Se il CODICE è uguale a "0101", ossia 5 in decimale (primo numero richiesto da inserire), passa allo stato "C1R1". Se invece il CODICE è diverso da 5 allora passa allo stato "C1W1". Le uscite rimangono poste a 0.

- **COD2** è lo stato che rappresenta il secondo tentativo di inserimento del codice (per quanto riguarda la prima cifra). Se il CODICE è uguale a "0101", ossia 5 in decimale (primo numero richiesto da inserire), passa allo stato "C1R2". Se invece il CODICE è diverso da 5 allora passa allo stato "C1W2". Le uscite rimangono poste a 0.

- **COD3** è lo stato che rappresenta il terzo tentativo di inserimento del codice (per quanto riguarda la prima cifra). Se il CODICE è uguale a "0101", ossia 5 in decimale (primo numero richiesto da inserire), passa allo stato "C1R3". Se invece il CODICE è diverso da 5 allora passa allo stato "C1W3". Le uscite rimangono poste a 0.

- **C1R1** è lo stato in cui il primo numero del codice (5) è giusto (primo tentativo). Se il CODICE è uguale a "0101", ossia 5 in decimale (secondo numero richiesto da inserire), passa allo stato "C2R1". Se il CODICE invece è diverso da 5, passa allo stato "C2W1". Le uscite rimangono poste a 0.

- **C2R1** è lo stato in cui il secondo numero del codice (5) è giusto (primo tentativo). Se il CODICE è uguale a "0000", ossia 0 in decimale (terzo numero richiesto da inserire), passa allo stato "PINR". Se il CODICE invece è diverso da 0, passa allo stato "COD2". Il segnale di uscita REINSERIRE_CODICE viene posto a 1.

- **C1W1** è lo stato in cui il primo numero del codice (che doveva essere uguale a 5) è sbagliato (primo tentativo). Indipendentemente dal segnale CODICE va allo stato successivo "C2W1". Le uscite rimangono poste a 0.

- **C2W1** è lo stato in cui il secondo numero del codice (che doveva essere uguale a 5) è sbagliato (primo tentativo). Indipendentemente dal segnale CODICE va allo stato successivo “COD2”. Il segnale di uscita REINSERIRE_CODICE viene posto a 1.

- **C1R2** è lo stato in cui il primo numero del codice (5) è giusto (secondo tentativo). Se il CODICE è uguale a “0101”, ossia 5 in decimale (secondo numero richiesto da inserire), passa allo stato “C2R2”. Se il CODICE invece è diverso da 5, passa allo stato “C2W2”. Le uscite rimangono poste a 0.

- **C2R2** è lo stato in cui il secondo numero del codice (5) è giusto (secondo tentativo). Se il CODICE è uguale a “0000”, ossia 0 in decimale (terzo numero richiesto da inserire), passa allo stato “PINR”. Se il CODICE invece è diverso da 0, passa allo stato “COD3”. Il segnale di uscita REINSERIRE_CODICE viene posto a 1.

- **C1W2** è lo stato in cui il primo numero del codice (che doveva essere uguale a 5) è sbagliato (secondo tentativo). Indipendentemente dal segnale CODICE va allo stato successivo “C2W2”. Le uscite rimangono poste a 0.

- **C2W2** è lo stato in cui il secondo numero del codice (che doveva essere uguale a 5) è sbagliato (secondo tentativo). Indipendentemente dal segnale CODICE va allo stato successivo “COD3”. Il segnale di uscita REINSERIRE_CODICE viene posto a 1.

- **C1R3** è lo stato in cui il primo numero del codice (5) è giusto (terzo tentativo). Se il CODICE è uguale a “0101”, ossia 5 in decimale (secondo numero richiesto da inserire), passa allo stato “C2R3”. Se il CODICE invece è diverso da 5, passa allo stato “C2W3”. Le uscite rimangono poste a 0.

- **C2R3** è lo stato in cui il secondo numero del codice (5) è giusto (terzo tentativo). Se il CODICE è uguale a “0000”, ossia 0 in decimale (terzo numero richiesto da inserire), passa allo stato “PINR”. Se il CODICE invece è diverso da 0, passa allo stato “BLDI”. Il segnale di uscita BLOCCO_BANCOMAT viene posto a 1.

- **C1W3** è lo stato in cui il primo numero del codice (che doveva essere uguale a 5) è sbagliato (terzo tentativo). Indipendentemente dal segnale CODICE va allo stato successivo “C2W3”. Le uscite rimangono poste a 0.

- **C2W3** è lo stato in cui il secondo numero del codice (che doveva essere uguale a 5) è sbagliato (terzo tentativo). Indipendentemente dal segnale CODICE va allo stato successivo “BLDI”. Il segnale di uscita BLOCCO_BANCOMAT viene posto a 1.

- **PINR** è lo stato in cui ogni cifra inserita (pin) del bancomat è corretta (5 5 0). Se riceve in ingresso il segnale di input BANCOMAT_INSERITO uguale a 0 ritorna allo stato di Reset (INSB) e le uscite rimangono poste a 0 (questo in caso non sia possibile prelevare cash dallo sportello in quanto il denaro disponibile è minore ad ¼ del denaro richiesto e quindi sia impossibile ritirare del denaro). Se invece il segnale di ingresso BANCOMAT_INSERITO sia uguale a 1 viene controllato il segnale interno CASH_OK: se uguale a 1 passa allo stato “BLDI” ponendo le uscite CHECK_DISPONIBILITÀ e ABILITAZIONE_EROGAZIONE uguali a 1; se uguale a 0 resta nello stato “PINR” ponendo l’uscita CHECK_DISPONIBILITÀ uguale a 1.

- **BLDI** è l’ultimo stato dove viene eseguito il blocco oppure l’erogazione del cash. Indipendentemente dai segnali d’ingresso (BANCOMAT_INSERITO rimane comunque posto a 1) torna allo stato iniziale “INSB”.

Codifica degli stati

La codifica degli stati effettuata in SIS tramite il comando “state_assign jedi”, che ha generato inoltre
la funzione stato prossimo δ e la funzione di uscita λ, risulta la seguente:

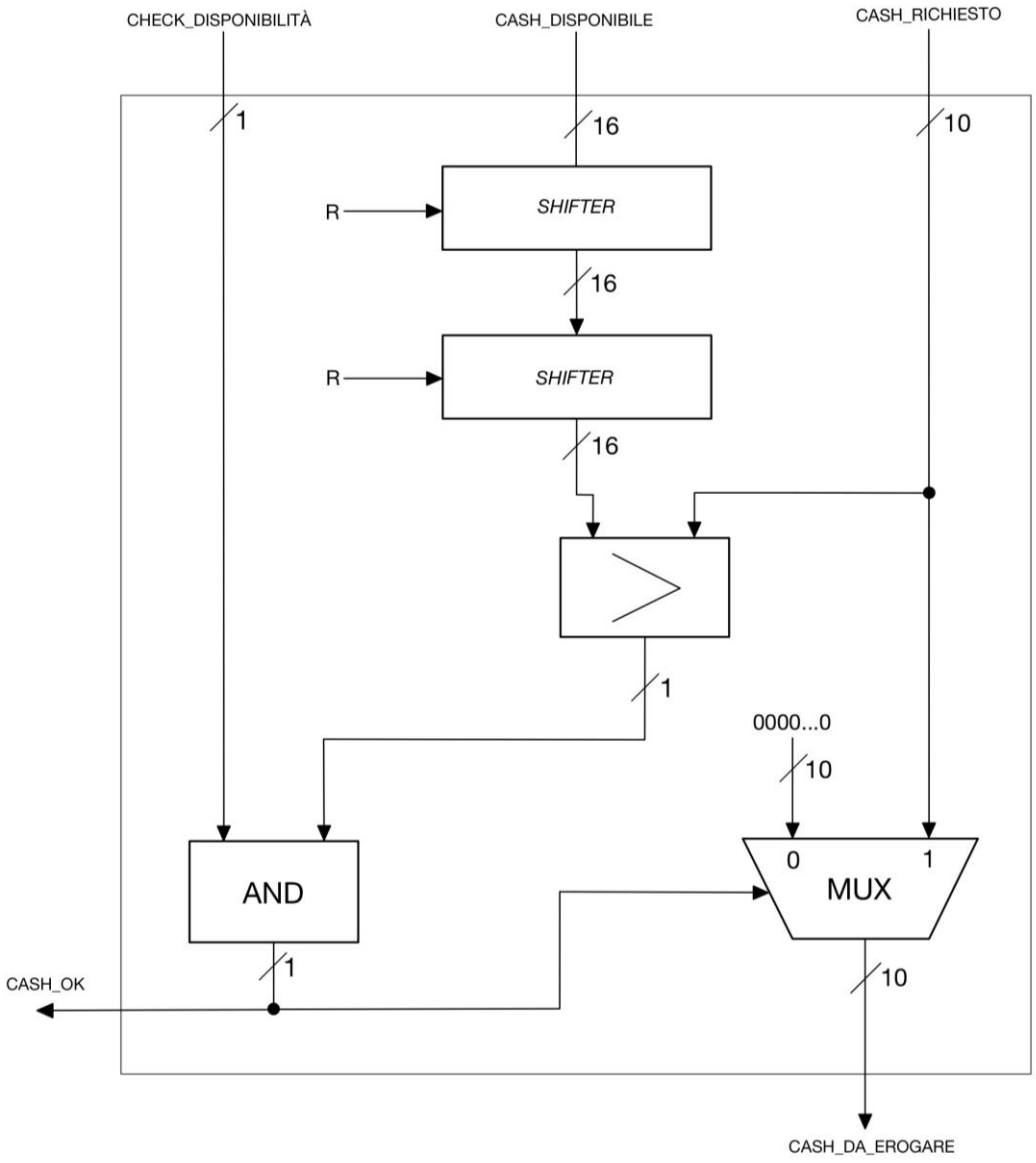
.code insb 01110
.code cod1 11110
.code c1r1 00001
.code c1w1 00101
.code c2r1 01001
.code c2w1 01101
.code pinr 00100
.code cod2 11100
.code bldi 01100
.code c1r2 10001
.code c1w2 10101
.code c2r2 11001
.code c2w2 11101
.code cod3 10100
.code c1r3 10000
.code c1w3 11000
.code c2r3 00000
.code c2w3 01000

Ottimizzazione FSM in SIS

L'ottimizzazione del file "controllore.blif" corrispondente all'FSM del circuito, risulta la seguente:

```
sis> read_blif controllore.blif
Warning: network `controllore.blif', node "C3" does not fanout
Warning: network `controllore.blif', node "C2" does not fanout
Warning: network `controllore.blif', node "C1" does not fanout
Warning: network `controllore.blif', node "C0" does not fanout
Warning: network `controllore.blif', node "OK" does not fanout
sis> print_stats
controllore       pi= 6    po= 4   nodes=  9       latches= 5
lits(sop)= 181  #states(STG)=  18
sis> source script.rugged
sis> fx
sis> print_stats
controllore       pi= 6    po= 4   nodes= 15       latches= 5
lits(sop)=  64   #states(STG)=  18
```

3) DATAPATH


CHECK_DISPONIBILITÀ
CASH_DISPONIBILE
CASH_RICHIESTO
1
16
10
R
SHIFTER
16
R
SHIFTER
16
1
0000...0
10
AND
0
1
MUX
1
10
CASH_OK
CASH_DA_EROGARE

Funzionamento

Il Datapath del circuito riceve in ingresso tre segnali di input:

- CHECK_DISPONIBILITA': In sis "CD"; è il segnale interno derivante dall'FSM.
  Se uguale a 1 il circuito controlla se il segnale d'ingresso CASH_RICHIESTO è minore a ¼ del segnale di ingresso CASH_DISPONIBILE: se la condizione è verificata l'output CASH_DA_EROGARE viene posto uguale a CASH_RICHIESTO e il segnale interno CASH_OK viene posto a 1 altrimenti tutte le uscite rimangono poste a 0.

- CASH_RICHIESTO: segnale composto da 10 bit, in sis "X0 … X9"; corrisponde al denaro richiesto per l'erogazione quindi con range da 0 a 1023. Per essere erogato deve essere minore a ¼ del CASH_DISPONIBILE.

- CASH_DISPONIBILE: segnale composto da 16 bit, in sis "Y0 … Y15"; corrisponde al denaro depositato nella cassaforte dello sportello automatico con range da 0 a 65535.
  Il circuito restituisce come output il segnale interno CASH_OK che, se verificata la condizione precedente quindi uguale a 1, abilita l'erogazione e restituisce in output il valore del segnale CASH_RICHIESTO altrimenti tutte le uscite rimangono poste a 0.

Il datapath del circuito è composto dai seguenti file:

- xor.blif
- mux.blif
- minore.blif
- datapath.blif

OTTIMIZZAZIONE DATAPATH IN SIS

L'ottimizzazione del file Datapath è la seguente:

Per il file "xor.blif" :

```
sis> read_blif xor.blif
sis> print_stats
xor               pi= 2    po= 1   nodes=  1       latches= 0
lits(sop)=   4
sis> source script.rugged
sis> fx
sis> print_stats
xor               pi= 2    po= 1   nodes=  1       latches= 0
lits(sop)=   4
```

Per il file "mux.blif" :

```
sis> read_blif mux.blif
sis> print_stats
mux               pi= 2    po= 1   nodes=  1       latches= 0
lits(sop)=   2
sis> source script.rugged
sis> fx
sis> print_stats
mux               pi= 2    po= 1   nodes=  1       latches= 0
lits(sop)=   2
```

Per il file "minore.blif" :

```
sis> read_blif minore.blif
sis> print_stats
minore          pi=20   po= 1   nodes= 11       latches= 0
lits(sop)= 114
sis> source script.rugged
sis> fx
sis> reduce_depth
sis> source script.rugged
sis> source script.rugged
sis> fx
sis> print_stats
minore          pi=20   po= 1   nodes= 15       latches= 0
lits(sop)=  66
```

Per il file "datapath.blif" :

```
sis> read_blif datapath.blif
Warning: network `datapath', node "Y1" does not fanout
Warning: network `datapath', node "Y0" does not fanout
sis> print_stats
datapath        pi=27   po=11  nodes= 22       latches= 0
lits(sop)= 154
sis> source script.rugged
sis> source script.rugged
sis> fx
sis> print_stats
datapath        pi=27   po=11  nodes= 28       latches= 0
lits(sop)= 100
```

4) FSMD

L'ottimizzazione dell'FSMD del circuito corrispondente al file "FSMD.blif" è la seguente:

```
sis> read_blif FSMD.blif
Warning: network `controllore.blif', node "C3" does not fanout
Warning: network `controllore.blif', node "C2" does not fanout
Warning: network `controllore.blif', node "C1" does not fanout
Warning: network `controllore.blif', node "C0" does not fanout
Warning: network `controllore.blif', node "OK" does not fanout
Warning: network `datapath', node "Y1" does not fanout
Warning: network `datapath', node "Y0" does not fanout
Warning: network `FSMD', node "Y1" does not fanout
Warning: network `FSMD', node "Y0" does not fanout
sis> print_stats
FSMD            pi=31   po=13  nodes= 43       latches= 5
lits(sop)= 164
sis> source script.rugged
sis> reduce_depth
sis> source script.rugged
sis> fx
sis> source script.rugged
sis> reduce_depth
sis> source script.rugged
sis> source script.rugged
sis> source script.rugged
sis> fx
sis> print_stats
FSMD            pi=31   po=13  nodes= 42       latches= 5
lits(sop)= 150
```

5) Mapping tecnologico

```
sis> read_blif FSMD.blif
Warning: network `FSMD', node "Y1" does not fanout
Warning: network `FSMD', node "Y0" does not fanout
sis> print_stats
FSMD            pi=31   po=13  nodes= 42       latches= 5
lits(sop)= 150
sis> read_library synch.genlib
sis> map -m 0
warning: unknown latch type at node '{[4]}' (RISING_EDGE assumed)
warning: unknown latch type at node '{[5]}' (RISING_EDGE assumed)
warning: unknown latch type at node '{[6]}' (RISING_EDGE assumed)
warning: unknown latch type at node '{[7]}' (RISING_EDGE assumed)
warning: unknown latch type at node '{[8]}' (RISING_EDGE assumed)
WARNING: uses as primary input drive the value (0.20,0.20)
WARNING: uses as primary input arrival the value (0.00,0.00)
WARNING: uses as primary input max load limit the value (999.00)
WARNING: uses as primary output required the value (0.00,0.00)
WARNING: uses as primary output load the value 1.00
sis> map -s
>>> before removing serial inverters <<<
# of outputs:         18
total gate area:      2752.00
maximum arrival time: (20.20,20.20)
maximum po slack:     (-6.60,-6.60)
minimum po slack:     (-20.20,-20.20)
total neg slack:      (-275.80,-275.80)
# of failing outputs:  18
>>> before removing parallel inverters <<<
# of outputs:         18
total gate area:      2752.00
maximum arrival time: (20.20,20.20)
maximum po slack:     (-6.60,-6.60)
minimum po slack:     (-20.20,-20.20)
```

```
total neg slack:     (-275.80,-275.80)
# of failing outputs:  18
# of outputs:        18
total gate area:     2720.00
maximum arrival time: (20.20,20.20)
maximum po slack:    (-6.60,-6.60)
minimum po slack:    (-20.20,-20.20)
total neg slack:     (-275.00,-275.00)
# of failing outputs:  18
sis> print_stats
FSMD            pi=31   po=13  nodes= 82       latches= 5
lits(sop)= 198
sis> source script.rugged
sis> source script.rugged
sis> fx
sis> map -s
>>> before removing serial inverters <<<
# of outputs:        18
total gate area:     2712.00
maximum arrival time: (23.80,23.80)
maximum po slack:    (-5.80,-5.80)
minimum po slack:    (-23.80,-23.80)
total neg slack:     (-317.00,-317.00)
# of failing outputs:  18
>>> before removing parallel inverters <<<
# of outputs:        18
total gate area:     2712.00
maximum arrival time: (23.80,23.80)
maximum po slack:    (-5.80,-5.80)
minimum po slack:    (-23.80,-23.80)
total neg slack:     (-317.00,-317.00)
# of failing outputs:  18
# of outputs:        18
total gate area:     2696.00
maximum arrival time: (23.80,23.80)
maximum po slack:    (-5.80,-5.80)
minimum po slack:    (-23.80,-23.80)
total neg slack:     (-316.60,-316.60)
# of failing outputs:  18
```

```
sis> reduce_depth
sis> source script.rugged
sis> source script.rugged
sis> fx
sis> print_stats
FSMD            pi=31   po=13  nodes= 42       latches= 5
lits(sop)= 147
sis> map -s
>>> before removing serial inverters <<<
# of outputs:         18
total gate area:      2704.00
maximum arrival time: (25.20,25.20)
maximum po slack:     (-6.80,-6.80)
minimum po slack:     (-25.20,-25.20)
total neg slack:      (-333.40,-333.40)
# of failing outputs:  18
>>> before removing parallel inverters <<<
# of outputs:         18
total gate area:      2704.00
maximum arrival time: (25.20,25.20)
maximum po slack:     (-6.80,-6.80)
minimum po slack:     (-25.20,-25.20)
total neg slack:      (-333.40,-333.40)
# of failing outputs:  18
# of outputs:         18
total gate area:      2688.00
maximum arrival time: (25.20,25.20)
maximum po slack:     (-6.80,-6.80)
minimum po slack:     (-25.20,-25.20)
total neg slack:      (-333.20,-333.20)
# of failing outputs:  18
```

SIMULAZIONI

- ✓ Processo completo per l’erogazione del denaro:

sis> read_blif FSMD.blif
Warning: network `FSMD', node "Y1" does not fanout
Warning: network `FSMD', node "Y0" does not fanout

#bancomat/tessera inserita
sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 11110

#primo tentativo inserimento prima cifra del codice (5)
sis> simulate 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 00001

#primo tentativo inserimento seconda cifra del codice (5)
sis> simulate 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 01001

#primo tentativo inserimento terza cifra del codice (0)
sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 00100

#controllo cash : cash_richiesto = 500 ; cash_disponibile = 30000
sis> simulate 1 0 0 0 0 0 1 1 1 1 1 0 1 0 0 0 1 1 1 0 1 0 1 0 0 1 1 0 0 0 0
Network simulation:
Outputs: 0 1 0 0 1 1 1 1 1 0 1 0 0
Next state: 01100

#ritorno allo stato di reset/iniziale

sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 01110

- Processo completo per il blocco del bancomat:

#bancomat/tessera inserita

sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 11110

#primo tentativo inserimento prima cifra del codice (8) primo tentativo errato:

sis> simulate 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 00101

#primo tentativo inserimento seconda cifra del codice (8) primo tentativo errato:

sis> simulate 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 01101

#primo tentativo inserimento terza cifra del codice (0) primo tentativo errato:

sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 1 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 11100

#secondo tentativo inserimento prima cifra del codice (5) secondo tentativo errato:

sis> simulate 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 10001

#secondo tentativo inserimento seconda cifra del codice (1) secondo tentativo errato:

sis> simulate 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 11101

#secondo tentativo inserimento terza cifra del codice (5) secondo tentativo errato:

sis> simulate 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 1 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 10100

#terzo tentativo inserimento prima cifra del codice (1) terzo tentativo errato:

sis> simulate 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 11000

#terzo tentativo inserimento seconda cifra del codice (1) terzo tentativo errato:

sis> simulate 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 01000

#terzo tentativo inserimento prima cifra del codice (1) terzo tentativo errato --> blocco bancomat:

sis> simulate 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 1 0 0 0 0 0 0 0 0 0 0

Next state: 01100

#ritorno allo stato di reset/iniziale

sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 01110

- Diverse simulazioni:

#bancoma/tessera non inserita
sis> simulate 0 0 1 0 1 1 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 01110

#bancomat/tessera inserita
sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 11110

#primo tentativo inserimento prima cifra del codice (5)
sis> simulate 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 00001

#primo tentativo inserimento seconda cifra del codice (5)
sis> simulate 1 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 01001

#primo tentativo inserimento terza cifra del codice (0)
sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 00100

#controllo cash : cash_richiesto = 300; cash_disponibile = 900
sis> simulate 1 0 0 0 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 1 0 0
Network simulation:
Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0
Next state: 00100

“dato che il cash_richiesto è maggiore di ¼ del cash_disponibile si ritenta con un’altra somma, che questa volta sia minore di ¼ del cash_disponibile

#controllo cash : cash_richiesto = 100; cash_disponibile = 900

sis> simulate 1 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 1 0 0

Network simulation:

Outputs: 0 1 0 0 0 0 1 1 0 0 1 0 0

Next state: 01100

#ritorno allo stato di reset/iniziale

sis> simulate 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

Network simulation:

Outputs: 0 0 0 0 0 0 0 0 0 0 0 0 0

Next state: 01110