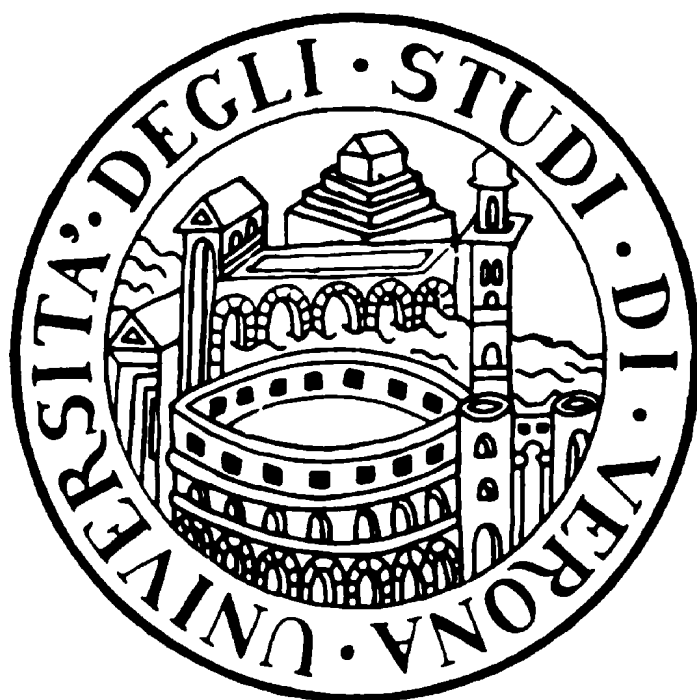


ELABORATO SIS 2019/2020

---

*ARCHITETTURA DEGLI ELABORATORI*

---

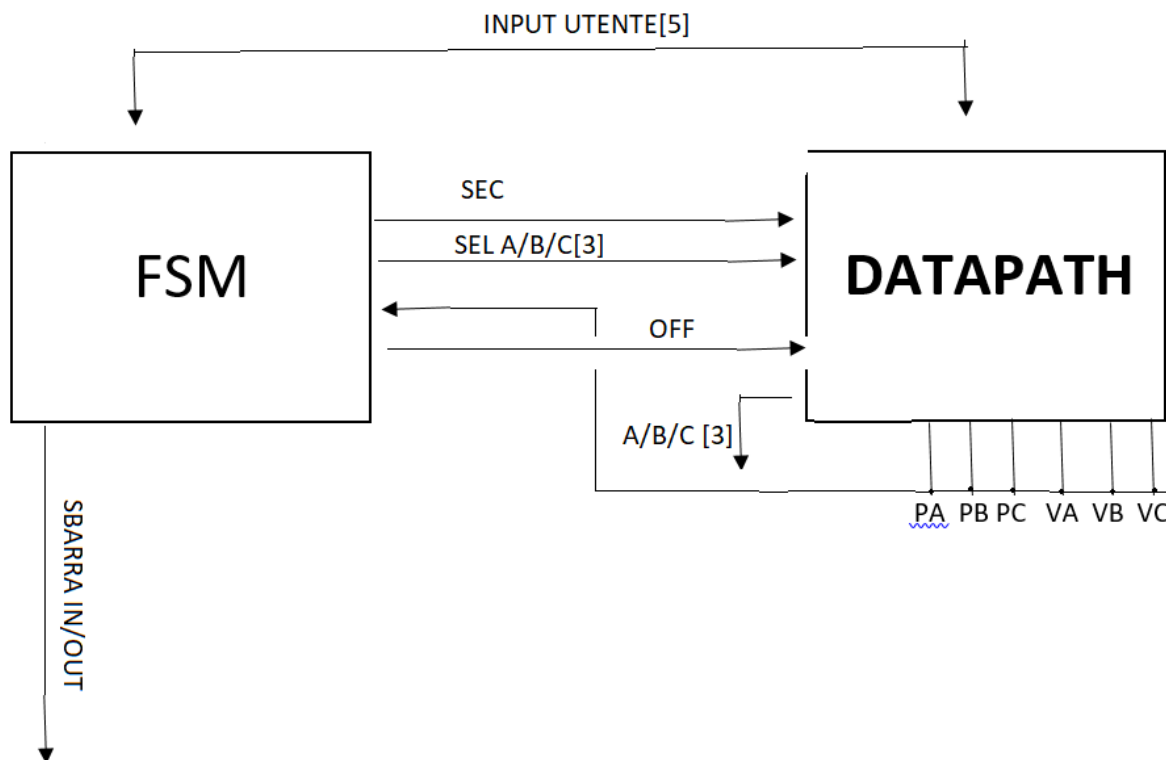


Studenti:

*VR450678 - LEDJO LLESHAJ*

*VR443630 - MARCO CASTELLI*

# ARCHITETTURA DEL CIRCUITO



La comunicazione del circuito avviene mediante un interscambio di bit tra fsm e datapath secondo appunto il modello FSMD.

## **Descrizione degli input:**

Input utente[5]= Questi bit rappresentano durante la fase di memorizzazione dei settori A,B,C le macchine presenti in essi successivamente, durante il funzionamento automatico i primi due specificano se l'utente è in ingresso o in uscita e gli ultimi tre sono relativi al settore in cui l'utente vuole entrare o uscire.

## **Descrizione degli output:**

Sbarra INOUT[2]= specifica se la sbarra in entrata si apre 10, se la sbarra in uscita si apre 01 oppure restano entrambe chiuse 00, entrambe le sbarre aperte 11 è una combinazione che non si può verificare.

## **Comunicazione fra FSM e DATAPATH**

La FSM riceve in input la sequenza di 5 bit inserita dall'utente che, inizialmente nei primi tre cicli corrisponde alle macchine presenti nei settori A, B e C ed inoltre riceve dal DATAPATH 3 bit relativi ai settori A, B, C se sono pieni (PA, PB, PC) e altri 3 bit che specificano se i settore A, B, C sono vuoti (VA, VB, VC). I seguenti bit assumo valore 1 se la condizione è soddisfatta.

La FSM genera come output SBARRA\_(IN/OUT) [2] che in base alle informazioni generate dal DATAPATH (se il relativo settore non è pieno, o non è vuoto e la sequenza inserita è corretta) la relativa sbarra in entrata e in uscita si apre oppure resta chiusa.

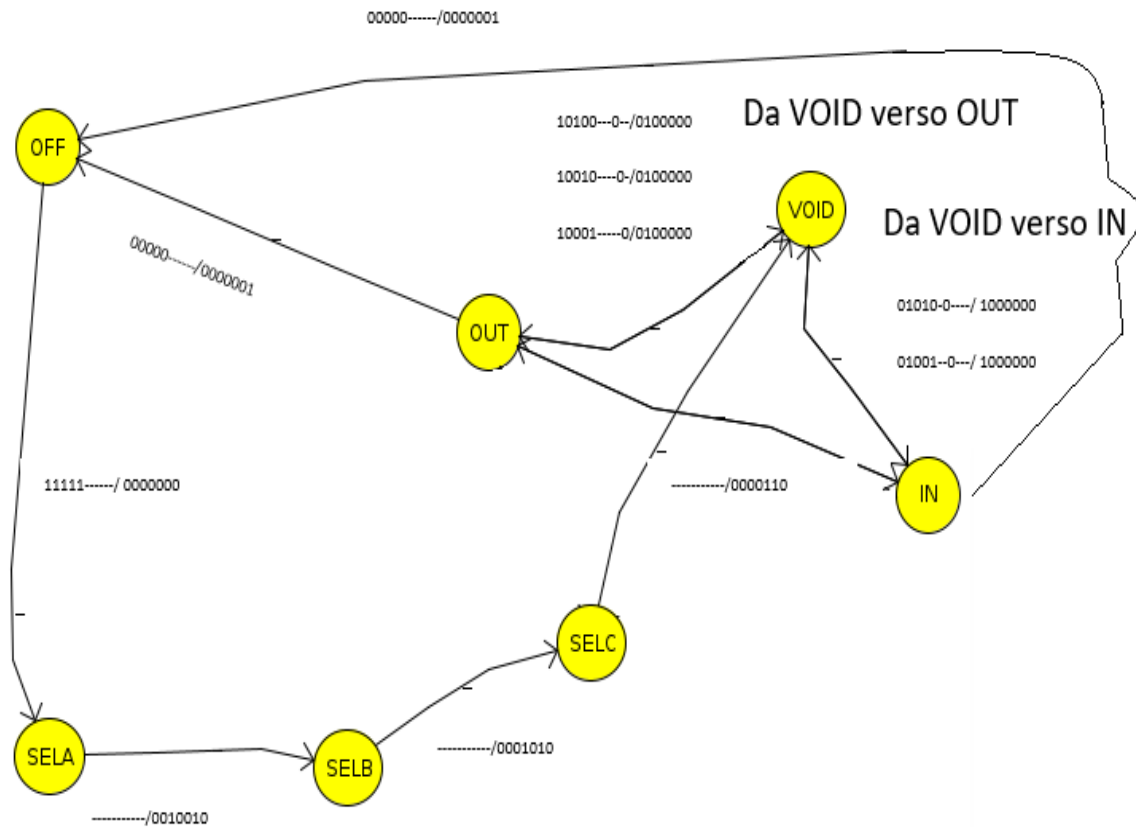
Per gestire l'inserimento delle macchine nei relativi settori la FSM genera 1 bit chiamato SEC che assume valore 1 se siamo nei 3 cicli di memorizzazione iniziale, mentre per specificare il settore in cui memorizzare la sequenza di bit inserita sfrutta i bit 3 bit SEL[3] che specificano il relativo settore (A=100, B=010, C=001).

La FSM inoltre genera OFF (1 bit) che indica se la macchina è spenta o è accesa che viene passato al DATAPATH.

Il DATAPATH controlla la correttezza delle sequenze inserite e nel caso ci fossero degli errori genera un bit ERR.

# CONTROLLORE E RELATIVO STATE TRANSITION GRAPH

## FSM



*Per questioni di compattezza non vengono riportate tutte le combinazioni. Tutte le combinazioni errate portano allo stato VOID, combinazioni giuste permettono di comunicare tra OUT e IN.*

Il controllore è una macchina a stati finiti di Mealy, ovvero una quintupla,  $\{S, \Sigma, \Lambda, T, G\}$ ,

costituita da:

- un insieme non vuoto e finito di stati ( $S$ );
- un insieme finito chiamato alfabeto degli ingressi ( $\Sigma$ );
- un insieme finito chiamato alfabeto delle uscite ( $\Lambda$ );
- una funzione di transizione ( $T : S \times \Sigma \rightarrow S$ );
- una funzione uscita ( $G : S \times \Sigma \rightarrow \Lambda$ ).

### **Descrizione degli stati:**

- OFF → La macchina si trova in questo stato finchè non viene digitata la combinazione 11111 (primi 5 bit) che corrisponde al codice di accensione, ritorna in questo stato una volta inserite le macchine nei tre settori e se viene digitata la combinazione 00000);
- SELA → In questo stato vengono memorizzate le macchine presenti nel settore A;
- SELB → In questo stato vengono memorizzate le macchine presenti nel settore B;
- SELC → In questo stato vengono memorizzate le macchine presenti nel settore C (nel caso venisse inserito un numero >23 il datapath automaticamente inserirà 24 ossia il settore pieno);
- IN → Si passa in questo stato quando la combinazione per entrare e il rispettivo settore specificato non risulta pieno la sbarra in entrata viene aperta;
- OUT → Si passa in questo stato quando la combinazione per uscire e il rispettivo settore non risulta vuoto la sbarra in uscita viene aperta;
- VOID → Nel caso in cui le combinazioni digitate fossero errate ad esempio valori IN/OUT errati (11,00) settori non presenti (101,110,..) oppure nel caso si volesse entrare in un settore pieno o si volesse uscire da un settore vuoto.

### **Descrizione degli input:**

- INOUT[2] → specifica se l'utente è in ingresso 01 oppure in uscita 10;
- S\_A/S\_B/S\_C[3] (corrisponde a sector) → specifica la combinazione del settore in cui si vuole o entrare oppure uscire (100= A;010=B;001=C);
- PIENO\_A → Vale 1 se il settore A è pieno (31) ed è un valore restituito dalla FSM;

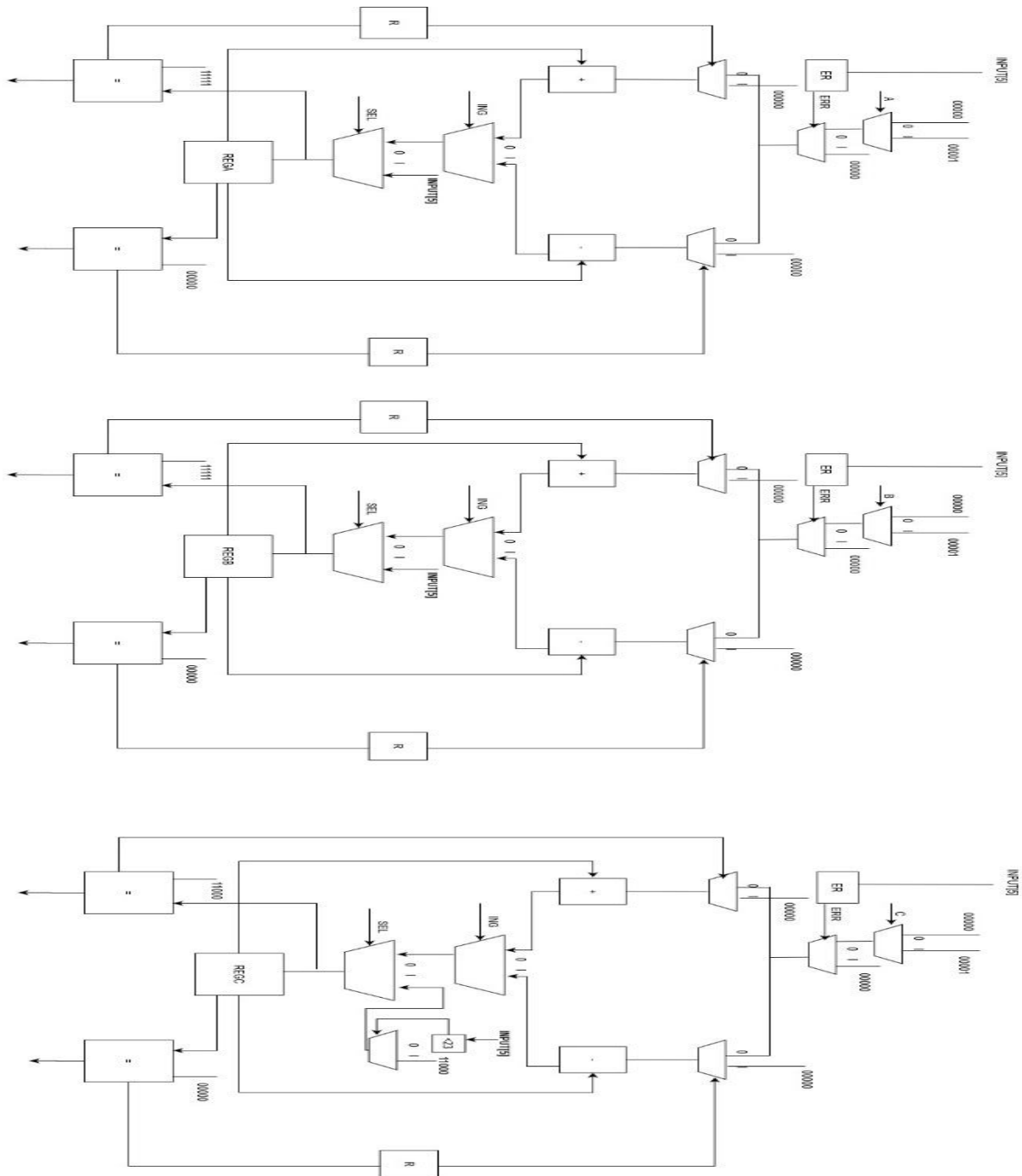
- PIENO\_B → Vale 1 se il settore B è pieno (31) ed è un valore restituito dalla FSM;
- PIENO\_C → Vale 1 se il settore C è pieno (24) ed è un valore restituito dalla FSM;
- VUOTO\_A → Vale 1 se il settore A è vuoto ed è un valore restituito dalla FSM;
- VUOTO\_B → Vale 1 se il settore B è vuoto ed è un valore restituito dalla FSM;
- VUOTO\_C → Vale 1 se il settore C è vuoto ed è un valore restituito dalla FSM.

### **Descrizione degli output**

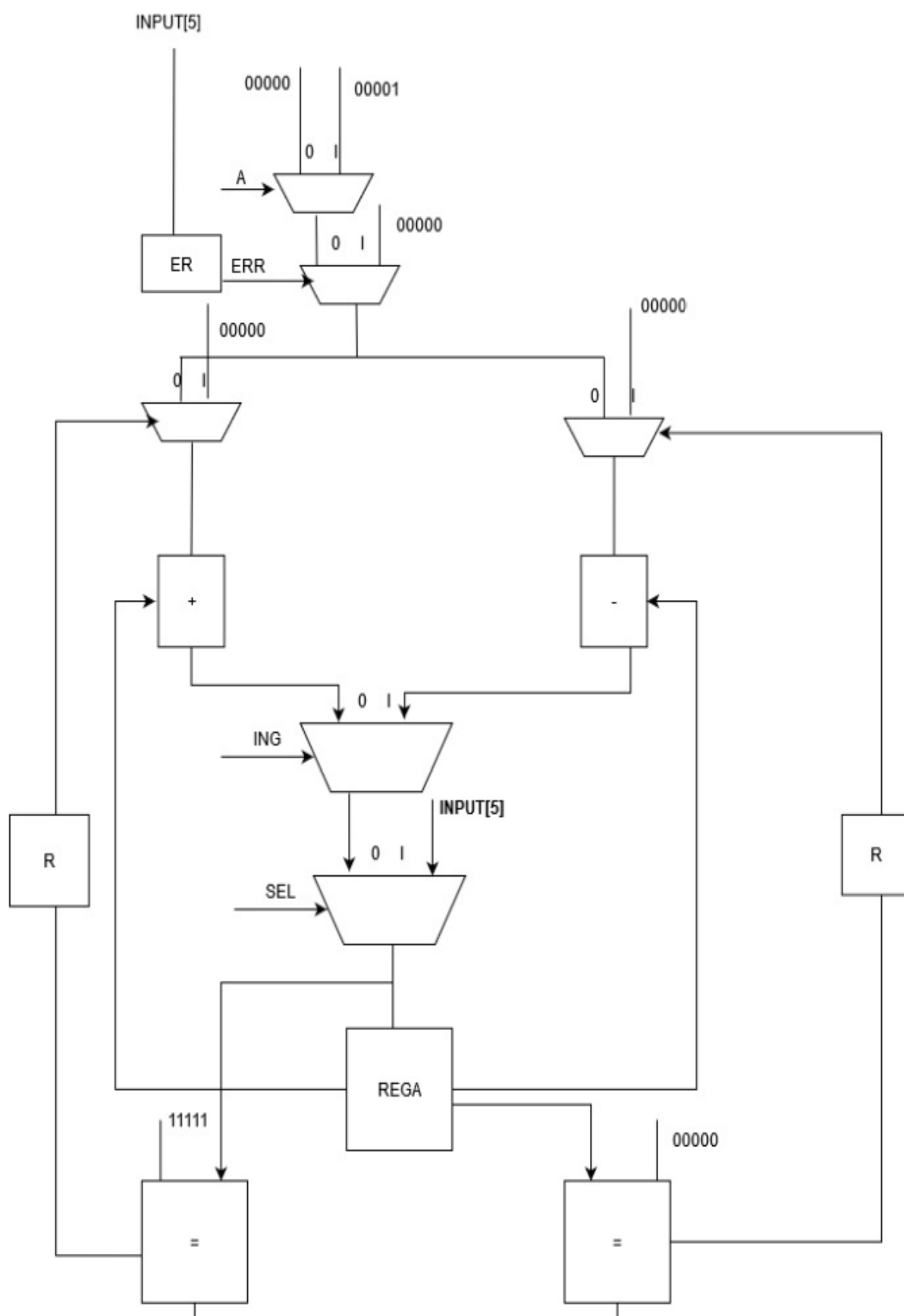
- SBIN → la seguente uscita corrisponde all'apertura della sbarra in entrata (1)
- SBOUT → la seguente uscita corrisponde rispettivamente all'apertura della sbarra in uscita(1)
- SECA → Ci consente di sapere in quale settore siamo se vale 1 siano in A
- SECB → Ci consente di sapere in quale settore siamo se vale 1 siano in B
- SECC → Ci consente di sapere in quale settore siamo se vale 1 siano in C
- SEC → Ci consente di sapere se siamo nei 3 cicli di memorizzazione delle macchine nei settori A,B,C.
- OFFX → permette i verificare se la macchina è spenta o accesa (1 spenta, 0 accesa)

# DATAPATH

Il datapath è composto da due blocchi identici relativi ai settori A,B e un blocco relativo al settore C.

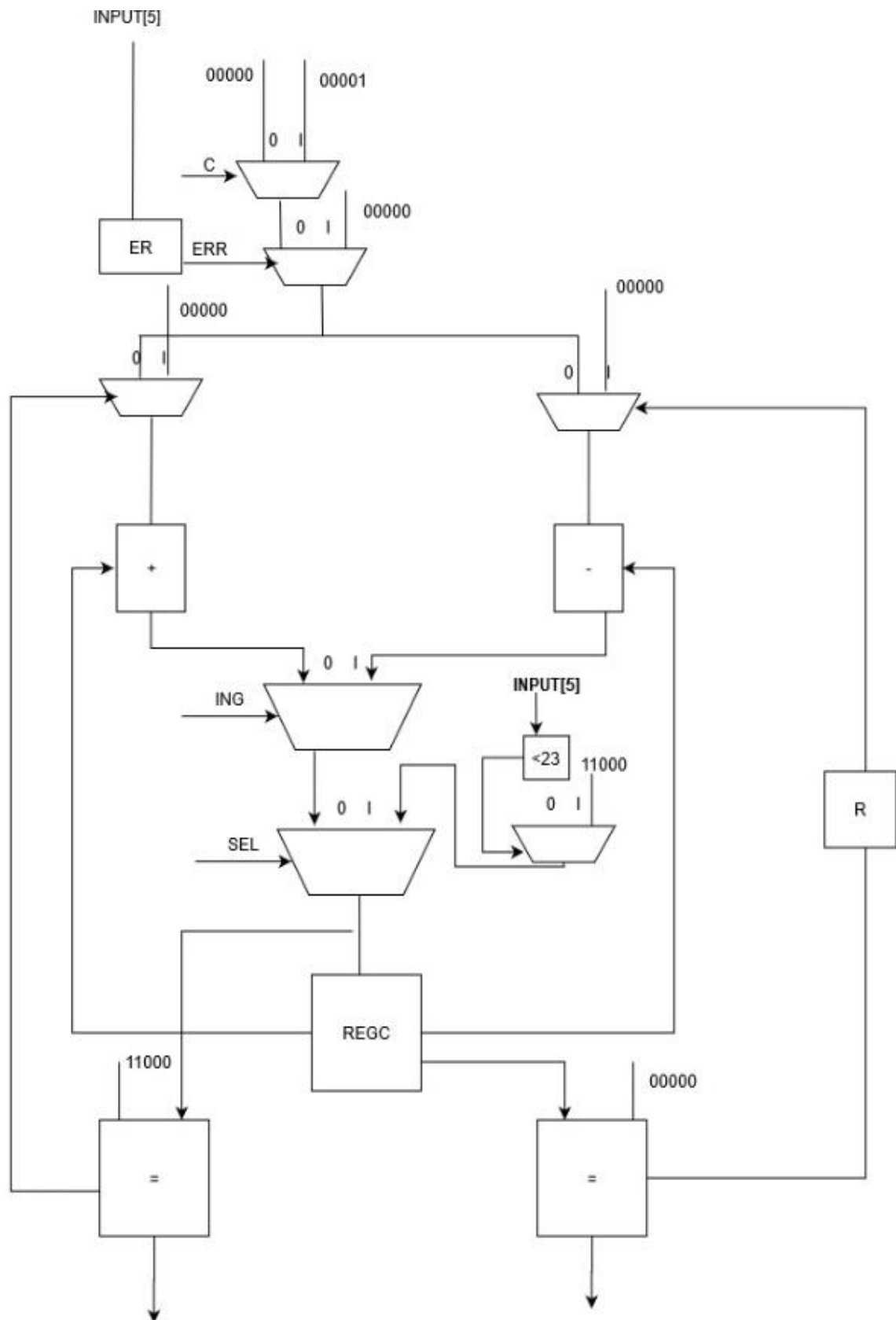


## ZOOM SU A CHE E UGUALE COME B





## ZOOM SU C



## **FUNZIONAMENTO DEL DATAPATH**

Primo mux, se abbiamo 1 allora significa che l'utente vuole entrare oppure uscire quindi entra la costante 00001. Secondo mux, se si è verificato un errore nell'inserimento (ERR vale 1) allora entra 0 e così si fa la somma e la sottrazione con 00000 che non altera i valori presenti nei registri, nel caso in cui non si fosse verificato nessun errore entra 00001 e viene utilizzato per le successive operazioni. Il bit di ERR è creato usando due porte XOR e una porta AND (per questioni di semplicità è riportato solo il componente ERR), la prima XOR è a 2 bit mentre l'altra a 3 bit, la porta AND consente di sapere se entrambe le uscite dalle porte XOR sono giuste in caso affermativo genera 1 altrimenti 0 e abbiamo un errore. Terzo e quarto mux vengono "controllati" dai bit relativi al settore pieno o vuoto (generati dall'uguaglianza con il valore 31/24 e con 0) in modo tale che se il settore fosse pieno la somma viene fatta con il valore 0 in modo che il valore nel registro non venga alterato e rispettivamente se il settore fosse vuoto la differenza viene fatta ancora con 0. Quinto mux, il bit ING permette di sapere se l'utente vuole entrare o uscire nel primo caso scelgo la somma (quando ING=1) nel secondo la differenza. Sesto mux, se siamo nelle prime 3 cicli di memorizzazione il bit SEL vale 1 e quindi viene salvato all'interno del registro il valore INPUT[5] inserito dall'utente, altrimenti se SEL vale 0 viene memorizzato il valore scelto nelle operazioni precedenti. Controllo se il bit che esce dal sesto mux è 31 (A, B) oppure 24 (C) in caso affermativo salvo nel rispettivo registro il bit che occorrerà nel prossimo ciclo per mandarlo come bit di selezione nel 3 mux. Controllo se il bit che esce dal registro VUOTO e lo mando come bit di selezione nel quarto mux. Il meccanismo descritto precedentemente vale sia per il settore A che B.

Nel caso del settore C l'unica variante è che al posto di avere 31 come valore massimo si ha 24 e per tale motivo se viene digitato un valore maggiore di 24 durante la fase di inserimento macchine nel settore, il datapath corregge l'inserimento inserendo 24 che corrisponde al settore pieno.

## MINIMIZZAZIONE DEL CIRCUITO PER AREA

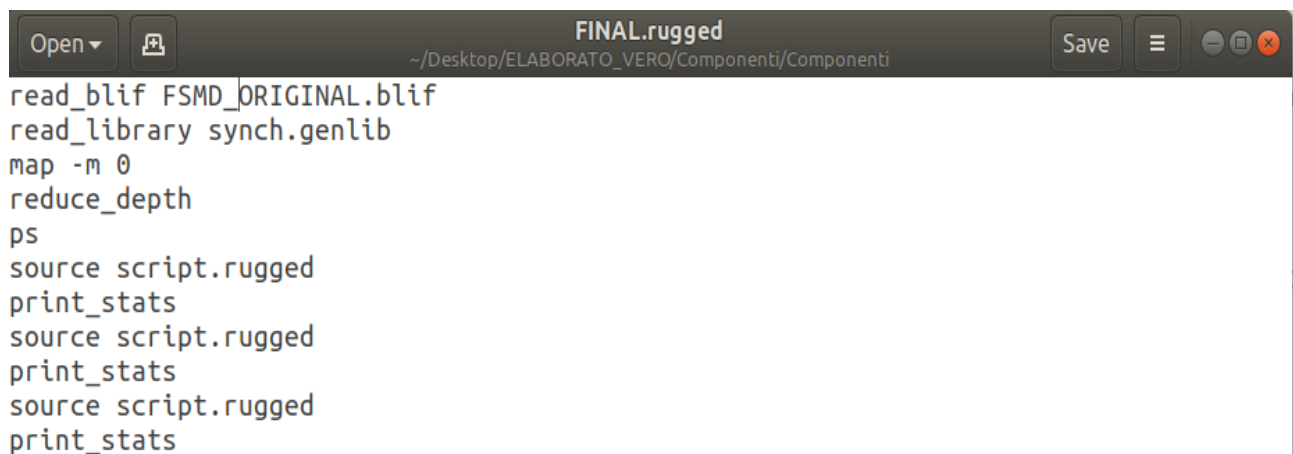
*Print\_stats prima della minimizzazione ma dopo la mappatura della FSMD*




```
FSMD          pi= 5   po= 5   nodes=113   latches=21  
lits(sop)=4353 lits(fac)=1435
```

*Print\_stats dopo la minimizzazione della FSMD*

```
sis> ps  
FSMD          pi= 5   po= 5   nodes= 72   latches=21  
lits(sop)= 368 lits(fac)= 336
```

*I comandi eseguiti scritti in un script*



```
Open ▾  FINAL.rugged Save    
~/Desktop/ELABORATO_VERO/Componenti/Componenti  
read_blif FSMD_ORIGINAL.blif  
read_library synch.genlib  
map -m 0  
reduce_depth  
ps  
source script.rugged  
print_stats  
source script.rugged  
print_stats  
source script.rugged  
print_stats
```

Statistiche prima l'ottimizzazione.

```
sis> map -s
>>> before removing serial inverters <<<
# of outputs:      26
total gate area:    9664.00
maximum arrival time: (37.60,37.60)
maximum po slack:   (-6.80,-6.80)
minimum po slack:   (-37.60,-37.60)
total neg slack:    (-747.40,-747.40)
# of failing outputs: 26
>>> before removing parallel inverters <<<
# of outputs:      26
total gate area:    9664.00
maximum arrival time: (37.60,37.60)
maximum po slack:   (-6.80,-6.80)
minimum po slack:   (-37.60,-37.60)
total neg slack:    (-747.40,-747.40)
# of failing outputs: 26
# of outputs:      26
total gate area:    9104.00
maximum arrival time: (36.60,36.60)
maximum po slack:   (-5.60,-5.60)
minimum po slack:   (-36.60,-36.60)
total neg slack:    (-731.20,-731.20)
# of failing outputs: 26
```

Statistiche dopo l'ottimizzazione.

```
sis> map -s
>>> before removing serial inverters <<<
# of outputs:      26
total gate area:    7744.00
maximum arrival time: (31.20,31.20)
maximum po slack:   (-8.40,-8.40)
minimum po slack:   (-31.20,-31.20)
total neg slack:    (-580.40,-580.40)
# of failing outputs: 26
>>> before removing parallel inverters <<<
# of outputs:      26
total gate area:    7744.00
maximum arrival time: (31.20,31.20)
maximum po slack:   (-8.40,-8.40)
minimum po slack:   (-31.20,-31.20)
total neg slack:    (-580.40,-580.40)
# of failing outputs: 26
# of outputs:      26
total gate area:    7232.00
maximum arrival time: (30.80,30.80)
maximum po slack:   (-8.40,-8.40)
minimum po slack:   (-30.80,-30.80)
total neg slack:    (-572.40,-572.40)
# of failing outputs: 26
```

Numero di gates era 113 ed è diventato 72

Ritardo era 37.60 ed è diventato 31.20

Area era 9664.00 ed è diventata 7744.00

## **Descrizione scelte progettuali effettuate:**

Al inizio volevamo creare un datapath piu complicato rispetto a questo, con un FSM che mandava in DATAPATH solo bit di selezioni per le primi 3 cicli di inserimento e un bit per la entrata/uscita delle machine,ma erano molti casi speciali e molto controlli da fare in questo modo quindi abbiamo fatto un FSM piu complicato e un DATAPATH piu facile a gestire.

Prima nel DATAPATH il bit di segnalazione nel Mux numero 4 era il bit creato dal registro UGUALE 31(24 nel settore C) ma questo causava cicli perche per i mux a funzionare doveva gia avere il bit pronto nell prossimo ciclo.

Nel FSMD quando la machina e stato spento dopo i settori sono riempiti da come output 1 finche si accesa di nuovo per memorizzare il numero delle machine.

Abbiamo cambiato l'output del FSMD quando abbiamo visto che quando la machina e spenta l'input nelle sbarre IN/OUT sono 0 perche prima la avevamo fatto 1 siccome durante la notte l'entrata era libera.

### **Note:**

Quando simuliamo il circuito,SIS elenca dei messaggi di "Warning" dovuti ai riporti del sommatore che genera Delle uscite che non vengono utilizzate. In realtà non sono degli errori, poiché possiamo perdere questi dati, che corrispondono al bit di overflow nelle operazioni di somma.