Anno accademico 2016/2017

STUDENTI:
De Marchi Mirco - VR408481
Guadagnini Filippo - VR408802

ELABORATO DI ARCHITETTURA DEGLI ELABORATORI

RELAZIONE

Dispositivo per monitoraggio di un impianto chimico industriale

INDICE:

Architettura generale del circuito

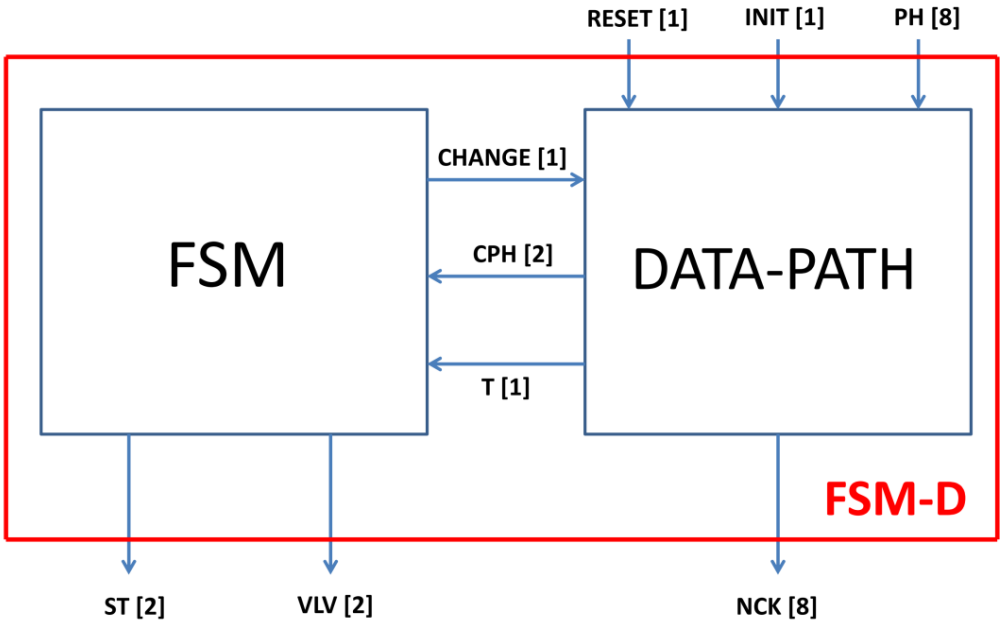
Il dispositivo da noi implementato, tramite SIS, permette di identificare lo stato di una soluzione della quale viene dato in ingresso il pH. Ogni volta che viene effettuata una simulazione, se il pH letto in ingresso equivale (in range di stato) a quello precedente, viene fatto partire un contatore il quale fornisce in uscita il numero di cicli nello stato corrente. Nel caso che questo cambi, il contatore riparte da zero. Se questo contatore arriva a contare fino a 5 cicli di clock, il dispositivo provvederà ad aprire una valvola in base allo stato in cui esso si trova: una valvola basica se ci si trova in uno stato acido e una acida se si è in uno stato basico. In questo modo al sesto ciclo di clock il sistema cercherà di riportare la soluzione allo stato neutro. Il sistema inoltre dispone di un segnale di “INIT” per spegnere la macchina e portare dunque tutte le uscite a 0 e un segnale di “RESET”, che permette di azzerare tutte le uscite quando il sistema è acceso. Il progetto da noi implementato dispone di un'unità di controllo e una di elaborazione.

Il sistema presenta i seguenti ingressi:

- RESET [1]: se uguale ad 1, il controllore deve essere resettato e le uscite poste a 0;
- INIT [1]: se uguale a 1 il sistema è acceso, quando vale 0 il sistema è invece spento e tutti i bit in uscita devono essere 0 indipendentemente dagli ingressi;
- PH [8]: valore del pH della soluzione con una risoluzione di 0.1, quando il valore, convertito in modulo, supera 80 lo stato è basico, quando il valore è minore di 60 lo stato è acido mentre quando ho un valore tra 80 e 60 la soluzione è neutra.

Il sistema presenta le seguenti uscite:

- ST [2]: rappresenta lo stato attuale della soluzione, 01 equivale allo stato acido, 10 a neutro mentre 11 rappresenta lo stato basico;
- VLV [2]: rappresenta la valvola da aprire quando sono passati più di 5 cicli nello stesso stato che non sia neutro, 01 rappresenta la valvola basica mentre 10 quella acida;
- NCK [8]: rappresenta il numero di cicli di clock passati nello stato corrente.



Come illustrato nella rappresentazione della FSM-D, oltre agli ingressi e le uscite sopra elencate, abbiamo posto 3 segnali di scambio tra il controllore e il datapath:

- CHANGE [1] (Cambio di stato): segnale che parte dal controllore e arriva al datapath, indica quando avviene un cambio di stato in modo da poter resettare il conteggio di NCK;
- CPH [2] (ControlPH): segnale che parte dal datapath e arriva al controllore, indica lo stato attuale della soluzione dato in base al pH ed è ottenuto dal confronto di questo con bit rappresentanti in codifica modulo i numeri 80 e 60, e il passaggio da un MUX.
  Possibili configurazioni di CPH:

  Acido ———> 10
  Basico ———> 01
  Neutro ———> 00
  Stato di Reset ———> 11

  Il segnale 11 (Stato di Reset) è un segnale trasmesso su CPH per resettare il controllore. Questo tipo di segnale dipende da INIT e RESET.

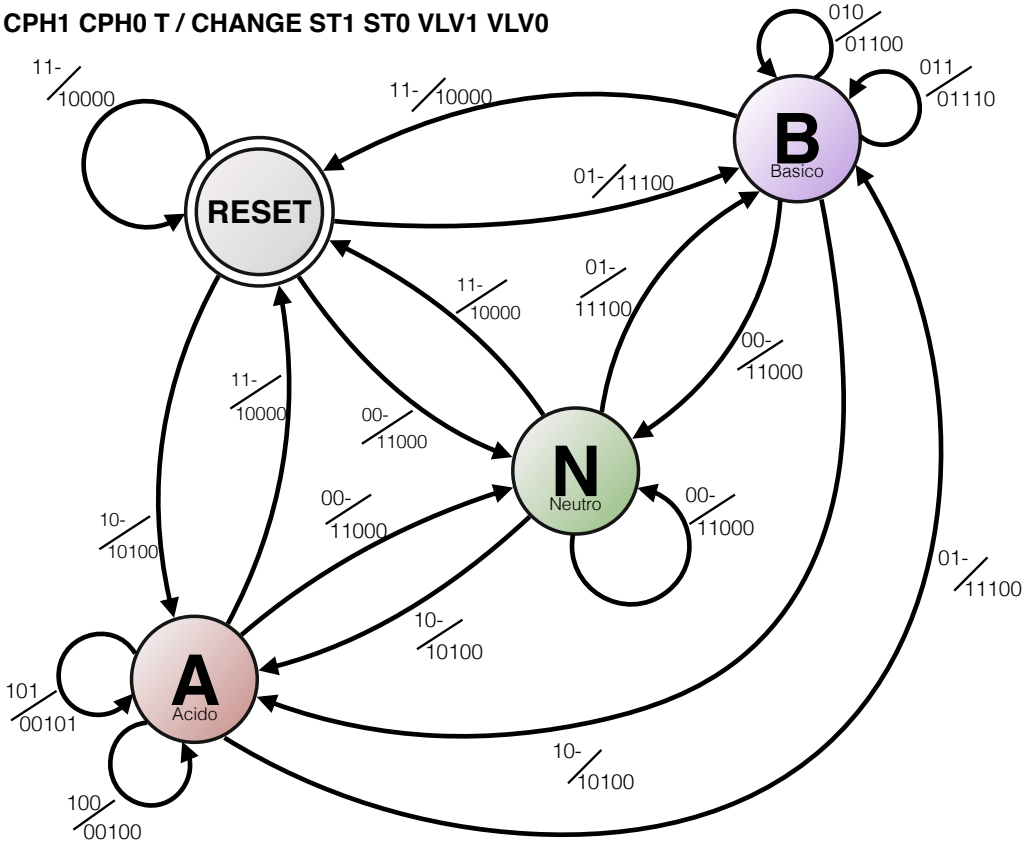
- T [1] (Time): segnale che parte dal datapath e arriva al controllore, quando vale 1 vuol dire che sono passati più di 5 cicli nello stesso stato e indica che verrà attivata la valvola specifica per lo stato basico o acido.

Diagramma degli stati del controllore

La FSM, ovvero *Finite State Machine (*Macchina stati finiti), rappresenta l'unità di controllo, cioè quella parte di sistema che permette di descrivere le transizioni, di stato in stato, del nostro progetto. La FSM riceve come ingressi CPH[2] e T[1], tutti dal datapath e ha come outputs i segnali ST[2], VLV[2] e CHANGE[1] (segnale di scambio con il data path). I segnali ST e CPH rappresentano entrambi il pH della soluzione, tuttavia lo rappresentano in due codifiche diverse. La principale funzione della FSM è quella di convertire il CPH in ST, in questo modo:

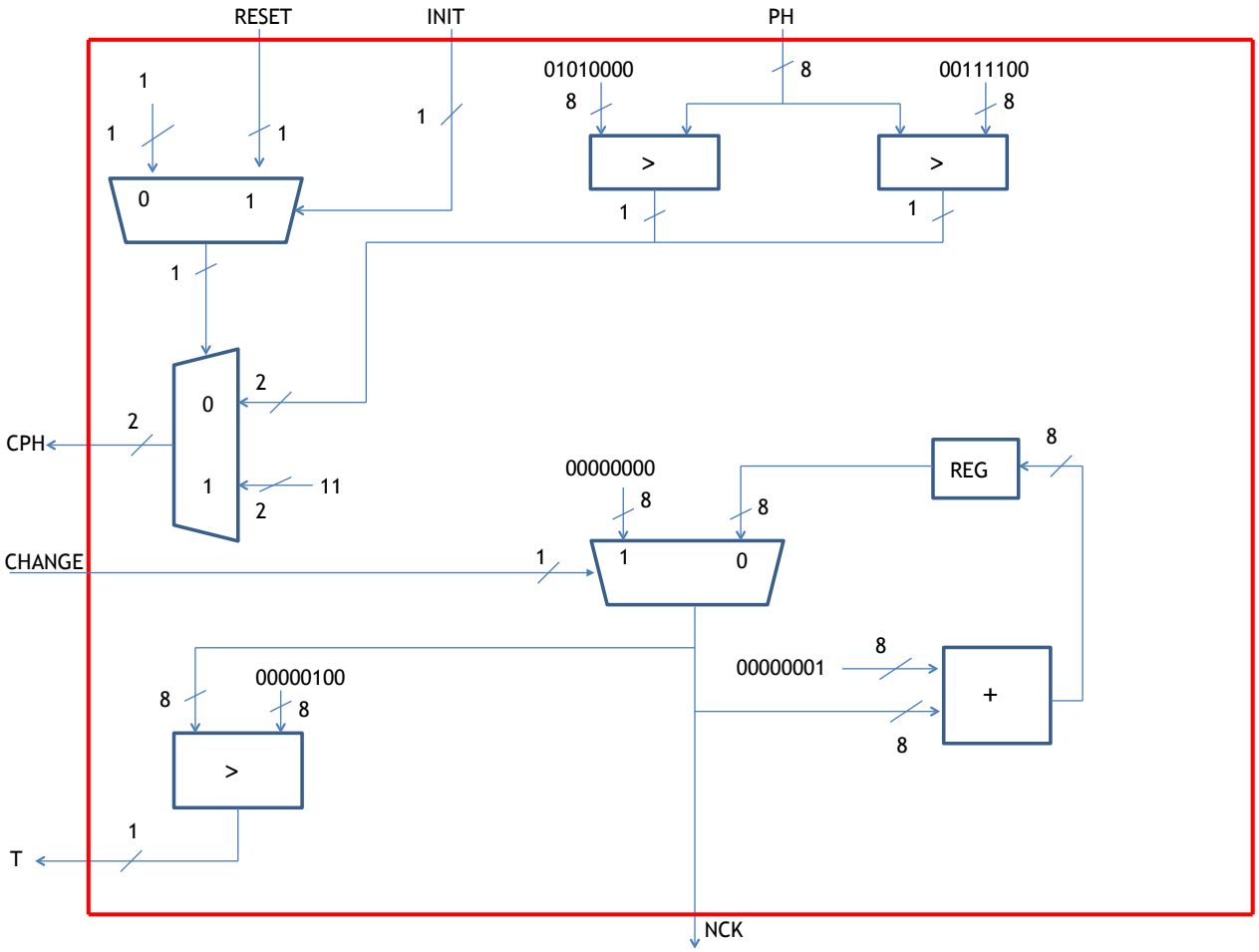
	CPH	**ST**
Acido	———> 10	———> 01
Basico	———> 01	———> 11
Neutro	———> 00	———> 10
Stato di Reset	———> 11	———> 00

Inoltre la FSM si occupa anche di aprire la valvola giusta in base allo stato di acido o basico in cui si trova e in base al segnale di T. VLV[2] sarà posta a 10 (valvola acida) quando il sistema è nello stato basico e 01 (valvola basica) quando è nello stato acido. Ma questo potrà avvenire solo quando T trasmette 1, cioè quando sono trascorsi più di 5 cicli di clock nello stato corrente. Al sesto ciclo dunque verrà aperta la valvola e continuerà ad essere mantenuta aperta fino ad un prossimo cambio di stato.
Il cambio di stato viene gestito dal segnale CHANGE, il quale trasmetterà 1 solo durante una transizione e nello stato di RESET.
Questa FSM non è minimizzabile.
La FSM è la seguente:

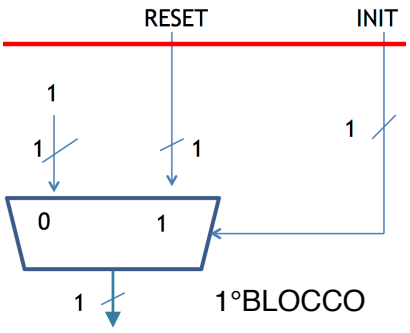


Architettura del Datapath

Il datapath è l'unità di elaborazione, rappresenta cioè la parte del sistema addetta a eseguire calcoli e confronti. Nel nostro progetto, il datapath prende in input RESET[1], INIT[1] e PH[8] come ingressi esterni mentre CHANGE[1] come segnale che proviene dal controllore. T è 1 bit in uscita, trasmesso alla FSM che indica quando sono trascorsi 5 cicli di clock; NCK rappresenta a 8 bit in uscita il numero di cicli passati nello stato corrente. Lo schema da noi progettato è il seguente:

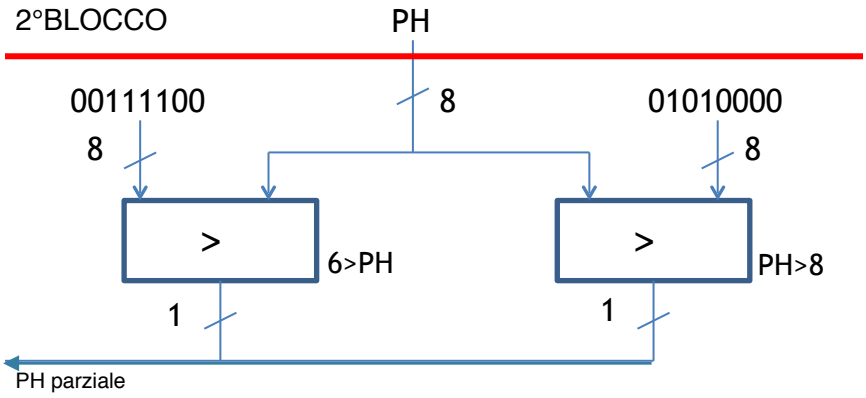


Analizziamo il datapath a blocchi:

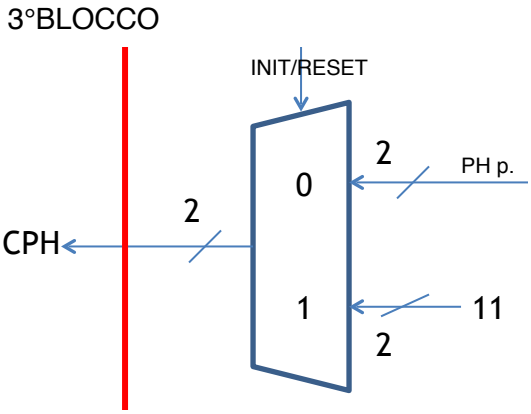


Il primo blocco è di selezione di INIT su RESET. Questo blocco è formato da un MUX che sceglie tra la costante 1 e RESET in base a INIT.
Se RESET va a 1, la macchina deve resettarsi, ma se INIT vale 0 (quindi macchina spenta) vuol dire che RESET deve valere 1 in qualsiasi caso. L'output di questo blocco servirà per ottenere lo stato di reset dell'unità di elaborazione e dell'unità di controllo, cioè lo stato in cui tutte le uscite devono essere 0.

Il secondo blocco è di controllo del pH. Fornisce in parte le possibili configurazioni di CPH. Con due operatori “maggiore”, controllo se 6,0 > pH (ovvero se pH < 6,0) e se pH > 8,0. In realtà le costanti limite di pH non sono 6,0 e 8,0 ma sono, se convertite in modulo, 60 e 80. Questa perché il pH è dato in input tra 0 e 14 con una risoluzione di 0,1. Dunque per convertire in modo corretto il valore di pH bisogna fare la conversione in modulo in base 10 e poi dividere il valore ottenuto per 10.

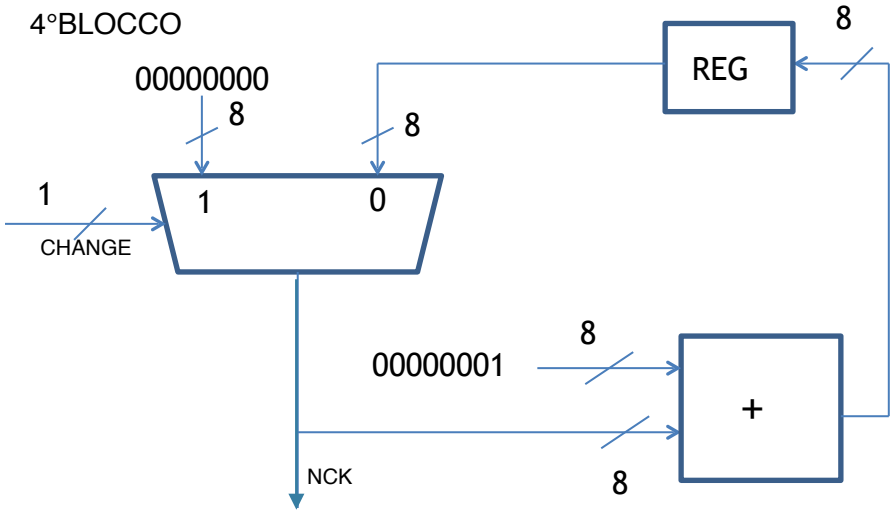


Esempio delle costanti limite: 00111100 = 60 / 10 = 6,0; 01010000 = 80 / 10 = 8,0.
Il controllo del pH può dare le seguenti configurazioni: 10 = acido; 01 = basico; 00 = neutro.



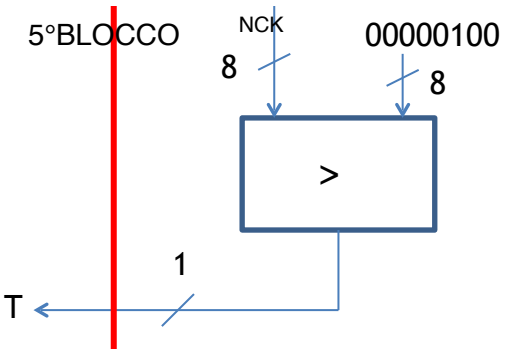
Nel terzo blocco queste configurazioni di controllo vengono inserite in un MUX, che ha come selettore l’uscita del blocco INIT/RESET (1° blocco) e ha come seconda entrata la costante 11. Dunque questo implementa una quarta configurazione al CPH: 10 = acido; 01 = basico; 00 = neutro, 11 = reset. La configurazione 11 si ottiene sempre se il selettore è 1, dunque nel caso in cui la macchina è spenta oppure è in fase di reset, altrimenti le altre configurazioni si ottengono quando il selettore è 0.

Il quarto blocco inizializza il processo di conteggio. Questo processo viene inizializzato attraverso un MUX, che ha come selettore la variabile in ingesso CHANGE[1] e come ingressi la costante 0 (a 8 bit) e il valore salvato nel registro della somma. Se CHANGE vale 1, vuol dire che c’è stato un cambio di stato e il processo di conteggio deve essere portato a 0, invece se CHANGE è 0 passa il valore del registro.



L’uscita del MUX va ad un sommatore al quale viene inserita la costante 1 (a 8 bit).
In fine il valore della somma viene salvato in un registro che al ciclo di clock successivo riporta il valore del conteggio al MUX, facendo dunque un ciclo.

Il quinto ed ultimo blocco determina la fine dei 5 cicli di clock e ha come ingresso l'uscita del quinto blocco, cioè l'uscita del MUX descritto poche righe sopra. Questo blocco confronta il conteggio dei cicli di clock e la costante 4 (a 8 bit e in base 2) con un operatore "maggiore". Quindi se il conteggio è maggiore di 4, allora l'uscita di questo blocco diverrà 1, in caso contrario rimarrà 0. L'uscita è esattamente la variabile T, che poi sarà un ingresso della FSM.



Per questo confronto è stata scelta la costante 4 e non la costante 5 per far trascorrere i 5 cicli di clock per un ritardo dovuto a una scelta progettuale sulla FSM.
In realtà spesso questo tipo di ritardo avviene per colpa di un registro nel contatore. In questo progetto invece il registro è stato messo dopo il sommatore e non provoca dunque alcun ritardo. Il ritardo è dovuto alla FSM per cui il CHANGE viene attivato durante la transizione. Ciò significa che il primo valore mostrato da NCK in un nuovo stato è 00000000 che in sequenza arriva a 4 (00000100) esattamente dopo 5 cicli di clock. Quindi al sesto ciclo di clock, cioè quando il contatore è a 5 (00000101) verrà attivato il segnale T.

Statistiche del circuito prima e dopo l'ottimizzazione

Per ottimizzare il circuito per area, come richiesto da consegna abbiamo cercato di applicare i comandi di ottimizzazione in SIS in modo tale da ottenere il minor numero possibile di lits, ovvero di letterali.

Per ottimizzare abbiamo utilizzato prevalentemente 3 script:

- script.rugged
- e due script creati da noi: script.1 e script.2

Lo script.1 è uno script di ottimizzazione molto lungo ottenuto componendo lo script.rugged e altri script trovati nelle librerie di default fornite da sis, tra i quali lo script.algebraic e lo script.boolean.
Lo script.2 è invece uno script di ristrutturazione, più corto che metto in seguito:

```
sweep
decomp -q
resub -a -d
sweep
eliminate -l 100 -1
simplify -l
full_simplify -l
sweep
decomp -q
fx -l
```

Questo script è lo script.delay trovato nelle librerie di SIS, rivisto e modificato, togliendo i comandi che non conoscevamo o che non venivano eseguiti correttamente.

Inoltre una strategia che si è rivelata molto efficace per minimizzare a fondo la nostra FSMD è stata eseguendo i seguenti comandi dopo aver applicato lo script.1:

```
eliminate -1
decomp
full_simplify
```

Risultato prima dell'ottimizzazione:

```
sis> rl FSMD_nomin.blif
"DATAPATH.blif", line 173: search file costanti.blif not found
sis> rl FSMD_nomin.blif
Warning: network `DATAPATH', node "COUT" does not fanout
Warning: network `FSMD-ControlloPH', node "COUT" does not fanout
sis> psf
FSMD-ControlloPH        pi=10   po=12   nodes=113       latches=10
lits(sop)= 417
```

Numero di lits: 417
Numero di nodes: 113

Dopo aver inserito e confermato il comando di lettura del file blif della FSMD sono comparsi 2 warning, i quali dicono di aver trovato all'interno del codice un nodo chiamato "COUT" al quale non è stato specificato un punto di arrivo.
COUT è il riporto in uscita del sommatore, il quale in effetti non mi è servito in nessun'altra parte del datapath, di conseguenza questo warning si può trascurare poiché il segnale COUT non ci serve.

Risultato dopo ottimizzazione:

```
FSMD-ControlloPH        pi=10   po=12   nodes= 35       latches=10
lits(sop)= 105
```

Numero di lits: 35
Numero di nodes: 105

Statistiche della mappatura

Per fare il mapping ho prima di tutto caricato la libreria richiesta dalla consegna: synch.genlib, dopodichè, come richiesto, ho mappato per area con il comando “map -m 0”. Per stampare le statistiche della mappatura uso il comando “map -s”.

```
sis> map -s
>>> before removing serial inverters <<<
# of outputs:          22
total gate area:       2840.00
maximum arrival time: (29.20,29.20)
maximum po slack:     (-6.20,-6.20)
minimum po slack:     (-29.20,-29.20)
total neg slack:      (-367.80,-367.80)
# of failing outputs:  22
>>> before removing parallel inverters <<<
# of outputs:          22
total gate area:       2808.00
maximum arrival time: (29.20,29.20)
maximum po slack:     (-6.20,-6.20)
minimum po slack:     (-29.20,-29.20)
total neg slack:      (-361.20,-361.20)
# of failing outputs:  22
# of outputs:          22
total gate area:       2440.00
maximum arrival time: (29.20,29.20)
maximum po slack:     (-5.60,-5.60)
minimum po slack:     (-29.20,-29.20)
total neg slack:      (-359.60,-359.60)
# of failing outputs:  22
sis> psf
FSMD-ControlloPH        pi=10   po=12   nodes= 61       latches=10
lits(sop)= 137
```

Numero di Gates: 2440.00
Ritardo massimo: 29.20

Come si può notare dall’immagine dopo aver stampato le statiche della mappatura ho fatto anche un print_stats che mostra i nodi e i letterali ottenuti, che risultano essere senza dubbio maggiori rispetto i nodi e i letterali mostrati prima con l’ottimizzazione della FSMD. Questo accade perché il processo di mapping potrebbe automaticamente aggiungere nodi e letterali per poter adattare nel modo da lui più opportuno le librerie caricate.
Sopratutto nei progetti più grandi questo processo diventa critico per l’area e il ritardo.

Visto che la consegna richiedeva di ottimizzare il circuito per area, aggiungiamo un commento su questo.
Abbiamo realizzato diverse macchine, confrontando tra loro i “Total Gate Area” e alla fine abbiamo scelto la macchina meglio ottimizzata per area.
In particolare abbiamo realizzato una macchina con una FSM con 3 stati, ma ci siamo accorti che poi mappando il tutto risultava che il datapath era troppo pesante e ci dava un’area di 2900 circa. Abbiamo capito che in realtà appesantire la FSM spesso è meglio di appesantire il datapath.

Inoltre abbiamo anche realizzato la macchina proposta sulla consegna dell'elaborato e siamo riusciti ad ottenere un'area di 2440, esattamente come questa.
Ma abbiamo preferito tenere questa macchina come risultato finale poiché abbiamo ritenuto più opportuno presentare un progetto diverso rispetto a quello dato nella consegna.

Scelte progettuali

La più sostanziale scelta progettuale fatta è stata quella di inserire 11 come possibile configurazione di CPH[2] per lo stato di reset.
Infatti quando nel datapath, dato in ingresso il pH a 8 bit, si cercava di definire se il pH fosse neutro, acido o basico, tramite due operatori “maggiore” abbiamo ottenuto la seguente configurazione:

Acido ———> 10
Basico ———> 01
Neutro ———> 00

Tuttavia ci siamo accorti che avendo a disposizione 2 bit, avevamo che la configurazione 11 non poteva mai verificarsi, poiché una soluzione non può essere sia acida, sia basica.
Così abbiamo deciso di implementare anche la configurazione 11 per lo stato di reset.
Abbiamo inserito RESET e INIT nel datapath, poiché aggiungendo la configurazione 11 avrebbe potuto controllare la FSM e quando il segnale di RESET in funzione di INIT sarebbe diventato 1, allora il segnale di CPH in uscita dal datapath sarebbe diventato 11 indipendentemente dal PH in ingresso.
Così abbiamo aggiunto la configurazione:

Stato di Reset ———> 11.

Come già anticipato un’altra scelta progettuale riguarda INIT e RESET.
La consegna non specificava che tipo di relazione avevano INIT e RESET tra di loro, la consegna diceva solo che INIT e RESET dovevano settare tutte le uscite a 0.
Noi abbiamo preferito metterli in relazione in questo modo: il RESET funzionerà solo se la macchina è accesa cioè solo se INIT=1., invece se INIT=0 la macchina setterà tutte le uscite a 0 indipendentemente dal segnale di reset.
Questa scelta è stata fatta poiché le specifiche dicevano che INIT avrebbe spento l’intera macchina, invece su RESET dicevano che doveva semplicemente portare tutte le uscite a 0. Dunque abbiamo pensato che INIT dovesse essere più importante di RESET.

Un’altra scelta progettuale riguarda il CHANGE. Esistevano altri modi in cui potevamo implementarlo: ad esempio potevamo chiamarlo “start” oppure “rt” (reset_time) e farlo comportare in altri modi.
Tuttavia abbiamo preferito chiamarlo CHANGE poiché forse rende più l’idea di quello che fa: ad ogni cambio di stato, oppure quando siamo nello stato di reset, il segnale CHANGE passa a 1 e permette il reset del conteggio. Di conseguenza quando CHANGE rimane a 0 vuol dire che rimaniamo nello stato corrente e quindi si può continuare il conteggio.

Un’ultima scelta progettuale prende in esame il 5° blocco del datapath. Questo blocco confronta il conteggio dei cicli di clock e la costante 4 con un operatore “maggiore”.
Per questo confronto è stata scelta la costante 4 e non la costante 5 per far trascorrere i 5 cicli di clock e per passare a 1 la variabile T al sesto ciclo. Questa scelta è stata fatta per un ritardo dovuto alla FSM.
In realtà spesso questo tipo di ritardo avviene per colpa di un registro nel contatore. In questo progetto invece il registro è stato messo dopo il sommatore e non provoca dunque alcun ritardo. Il ritardo in questo caso è dovuto invece alla FSM per cui il CHANGE viene attivato durante la transizione. Ciò significa che il primo valore mostrato da NCK in un nuovo stato è 00000000 ed esattamente dopo 5 cicli di clock arriva al 4 (00000100).

Quindi al sesto ciclo di clock, cioè quando il contatore è a 5 (00000101) verra attivato il segnale T.