Relazione dell’elaborato di architettura degli elaboratori

A.A. 2021/2022

Studenti:
Caprioli Silvia VR476714
Bellamoli Damiano VR481878

Sommario

Schema generale del circuito…… 3

Il controllore (FSM)…….. 5

L’esecutore (Datapath)…… 6

Statistiche del circuito..... 8

Mapping..... 10

Scelte Progettuali..... 11

SCHEMA GENERALE

Il circuito da noi creato permette di controllare un macchinario chimico che rilevi il valore di PH di una soluzione e di portare essa al valore di neutralità in base all'aggiunta, attraverso specifiche valvole, di soluzioni acide o basiche. Nel caso in cui la soluzione iniziale risulti essere acida, il circuito rileverà il ph e permetterà di aprire la valvola della soluzione basica per l'aggiunta di una determinata quantità, la quale porterà la soluzione di partenza alla neutralità. Lo stesso meccanismo avviene per le soluzioni basiche a differenza della valvola di apertura che sarà quella della soluzione acida. Il circuito è composto da un fsm e un datapath, i quali comunicano costantemente al fine di controllo e analisi della soluzione per produrre un risultato finale.

Il circuito presenta i seguenti ingressi e uscite:
INGRESSI

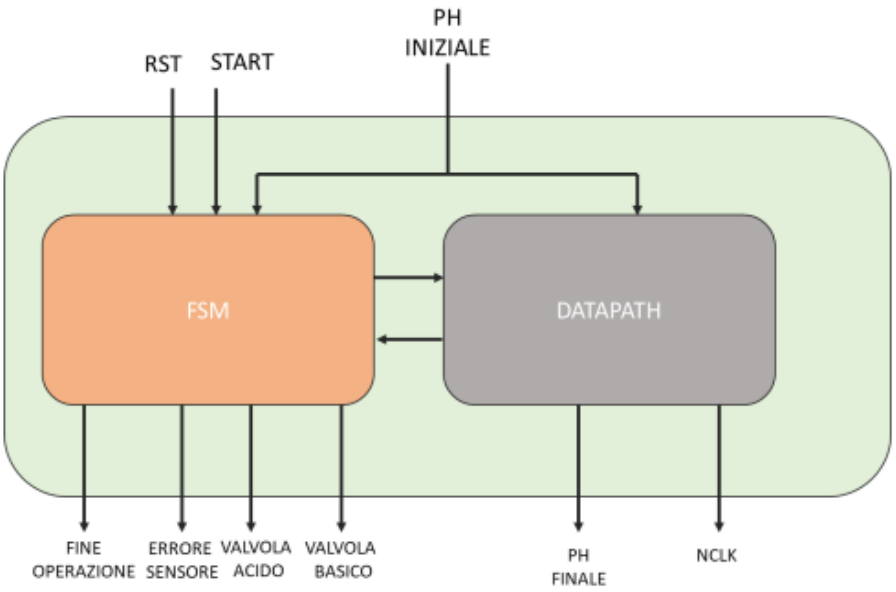
- RST [1bit]: funge da reset ovvero quando ha il valore 1 il circuito azzera tutto e ritorna allo stato iniziale di reset.
- START [1bit]: quando vale 1 il sistema acquisisce il ph iniziale per un solo ciclo di clock, procedendo con la fase di elaborazione.
- pH [8bit]: il valore iniziale del ph della soluzione rilevato dal sensore; è codificato in virgola fissa ad 8 bit con 4 bit dedicati alla parte intera e 4 alla parte decimale.

USCITE

- FINE_OPERAZIONE [1bit]: il segnale, quando è a 1, indica la fine dell'elaborazione e porta in uscita i valori del ph finale e del numero di clock che sono stati necessari alla conclusione dell'operazione.
- ERRORE_SENSORE [1bit]: segnala da subito la presenza di un errore in caso di ph superiore a 14 impedendo l'avvio dell'elaborazione, rimanendo nello stato di reset
- VALVOLA_ACIDO [1bit]: quando viene messo ad uno, il sistema apre la valvola della soluzione acida, aumentando il ph della soluzione di 0,25 ad ogni ciclo di clock.
- VALVOLA_BASICO [1bit]: quando viene messo ad uno, il sistema apre la valvola della soluzione basica ed abbassa il ph della soluzione di 0,5 ad ogni ciclo di clock.
- PH_FINALE [8 bit]: valore finale del ph che viene portato in output solo a fine operazione, 4 bit per parte intera e 4 per parte decimale
- NCLK [8 bit]: numero di clock che ha eseguito il circuito per portare la soluzione alla neutralità; viene portato in output solo a fine operazione.

Il controllore e il datapath contenuti nel circuito si scambiano i seguenti 4 segnali:

- START [1 bit]: segnale che riceve fsm e che gira al datapath ad inizio operazione per azzerare il contatore dei cicli di clock e acquisire il valore del ph iniziale.
- VALORE [2 bit]: segnale in arrivo da FSM e che fornisce il valore di acido, basico e neutro a seguito della prima analisi della soluzione, permettendo al datapath di selezionare la corretta operazione da eseguire.
- FINE_OPERAZIONE [1 bit]: nel momento in cui la soluzione finale è alla neutralità, l'FSM fornisce il segnale, oltre che in output generale del circuito, anche al datapath in modo da permettere l'uscita dei segnali nclk e ph finale.
- FINE [1 bit]: segnale che da datapath va in FSM per segnalare la fine dei calcoli e farle quindi chiudere la valvola finora aperta.



Il ciclo si avvia tramite il controllo del ph da parte dell'FSM, che produce un valore a 2 bit (VALORE) per indicare al datapath se la soluzione è acida, basica o neutra e permettergli quindi di eseguire i calcoli di conseguenza. Il datapath calcola il numero di cicli da eseguire per portare la soluzione alla neutralità e il valore finale del ph e li riporta in uscita solo quando riceve il segnale di fine operazione da FSM.

ESEMPI

Se all'interno del circuito finisce un valore di ph iniziale acido pari a 6, il circuito porterà il ph a 7 tramite 4 cicli di clock e aggiungendo la soluzione basica per 4 volte (che aumenta il ph di 0,25 alla volta).

Se il ph iniziale è basico e pari a 9,3, il circuito porterà il ph a 7,8 tramite 3 cicli di clock e aggiungendo la soluzione acida per 3 volte (che abbassa il ph di 0,5 alla volta).

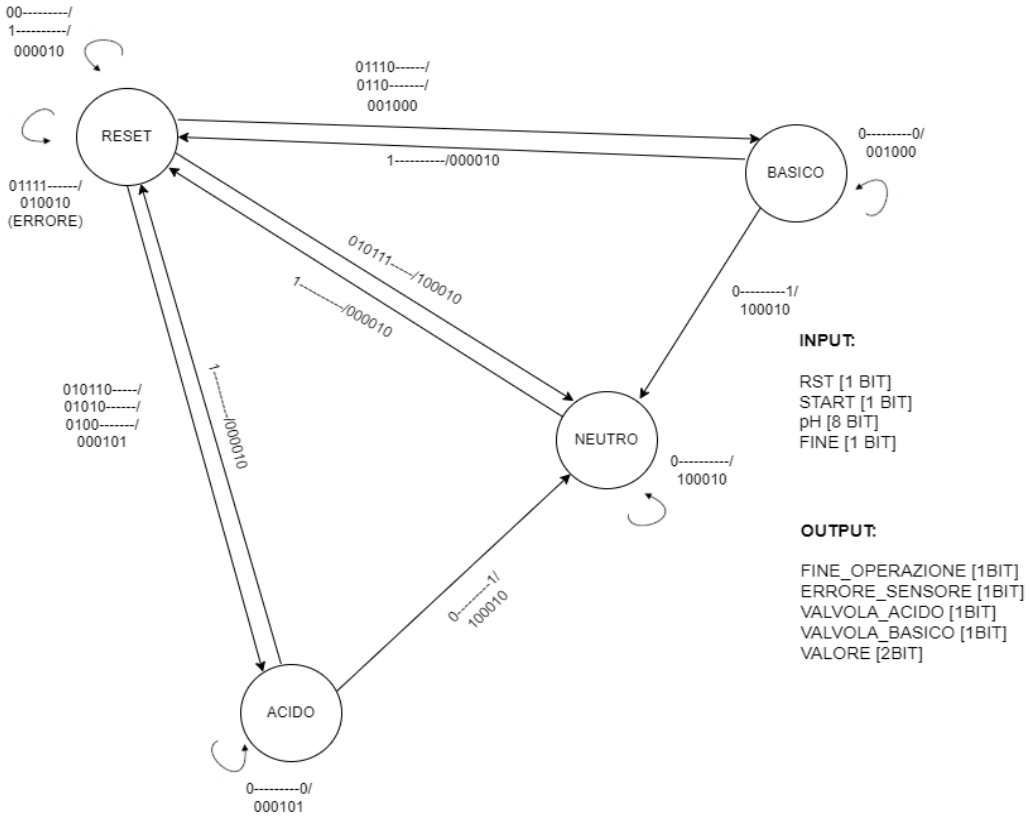
Se il ph iniziale è neutro, uscirà invariato nella porta PH_FINALE e il conteggio dei cicli di clock rimarrà a 0.

IL CONTROLLORE

Il controllore è una macchina FSM (Finite State Machine) di Mealy che presenta 4 ingressi (RST, START, PH_INIZIALE e FINE) e 5 uscite (FINE_OPERAZIONE, ERRORE_SENSORE, VALVOLA_ACIDO, VALVOLA_BASICO e VALORE). “FINE” è il segnale che arriva al controllore dal datapath e che se vale 0 significa che FSM deve tenere aperta la valvola in quanto l’operazione non è terminata, mentre se vale 1 significa che datapath ha terminato i calcoli e FSM deve chiudere la valvola. “VALORE” è un segnale a 2 bit che FSM manda al datapath per indicare se il ph iniziale è basico (00), acido (01) o neutro (10). Il valore 11 non si verificherà mai, in quanto non necessario al caso specifico di questo circuito.

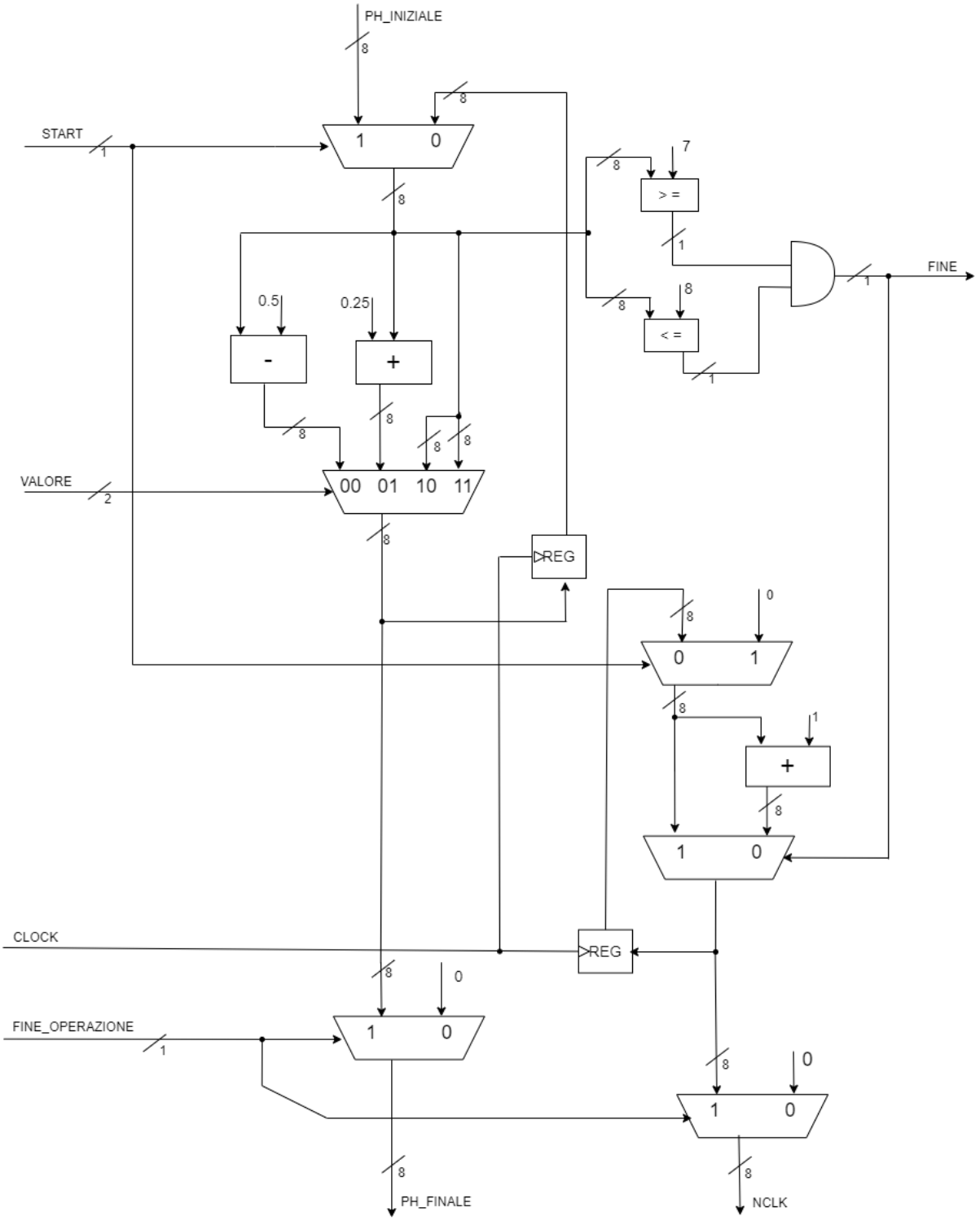
Il controllore può assumere 4 differenti stati:

- RESET → è lo stato iniziale del FSM, nel quale nessuna valvola viene aperta e tutto il circuito viene azzerato; questo stato viene poi raggiunto anche nel caso in cui venga prodotto un errore di lettura del ph (maggiore di 14).
- ACIDO→ FSM passa a questo stato quando rileva un ph acido (strettamente inferiore a 7) e vi rimane fino a quando non si è raggiunto un ph neutro; mentre è in questo stato, FSM tiene aperta la valvola della soluzione basica.
- BASICO→ FSM passa a questo stato quando rileva un ph basico (strettamente superiore ad 8) e vi rimane fino a quando non si è raggiunto un ph neutro; mentre è in questo stato, FSM tiene aperta la valvola della soluzione acida.
- NEUTRO→ nello stato neutro, non verrà apportata alcuna modifica al valore del ph e al conteggio dei cicli di clock, in quanto la soluzione ha un ph neutro; si arriva a questo stato direttamente dallo stato di reset se la soluzione è a ph neutro fin da subito, oppure nel momento in cui datapath indica ad FSM con il segnale FINE ad 1 il termine dei calcoli e il raggiungimento della neutralità.



IL DATAPATH

Il datapath è la parte del circuito dedita al calcolo del ph finale e al conteggio del numero di clock necessari a portare il ph alla neutralità. Il datapath ha 4 segnali di ingresso (PH_INIZIALE, START, VALORE e FINE_OPERAZIONE).
In uscita dal datapath abbiamo invece 3 segnali (PH_FINALE, NCLK e FINE).



Il datapath ha il ruolo di esecutore all'interno del circuito.
Il lavoro del datapath inizia quando riceve il segnale di START dal controllore, che quando vale 1 permette al datapath, tramite 2 multiplexer dedicati (entrambi a 2 ingressi da 8 bit ciascuno), di acquisire il PH_INIZIALE della soluzione e azzerare il contatore dei cicli di clock.
Nel contempo arriva anche il segnale VALORE da FSM, che, tramite un multiplexer a 4 ingressi da 8 bit ciascuno, permette di selezionare la corretta operazione in uscita: se VALORE vale 00 (soluzione basica), esce il ph sottratto di 0,5 ad ogni ciclo di clock; se VALORE è 01 (soluzione acida), esce il ph incrementato di 0,25 ad ogni ciclo di clock; infine se VALORE è 10 (soluzione neutra), continuerà a girare lo stesso valore in quanto non dovrà più essere modificato. VALORE non assumerà mai 11, in quanto non necessario al caso specifico del nostro circuito.
L'uscita selezionata riporterà ogni volta il ph finale aggiornato, che verrà dato ad un registro che lo salva ad ogni ciclo di clock e lo fa rientrare in ciclo per il calcolo e ad un mux che lo metterà sulla porta di output PH_FINALE solamente una volta raggiunta la stabilità.
Contemporaneamente, il ph aggiornato ad ogni ciclo di clock finisce in due comparatori i quali determinano se la soluzione è neutra, ovvero maggiore o uguale di 7 e minore o uguale di 8. Il segnale potrà procedere oltre solo ed esclusivamente se entrambe le condizioni saranno rispettate, tramite una porta AND. Se la porta AND vale 0, il segnale di FINE arriva alla FSM per dirle che i calcoli non sono terminati poiché non si è raggiunta la neutralità e deve quindi tenere aperta la valvola della relativa soluzione, mentre se vale 1 le dice che i calcoli sono terminati e deve chiudere la valvola.
Il segnale FINE passa quindi ad un altro multiplexer (all'interno del datapath) a 2 ingressi da 8 bit ciascuno, che fa passare nel contatore dei cicli di clock il conteggio invariato in caso valga 1 (poiché significa neutralità raggiunta) oppure il conteggio incrementato di 1 in caso valga 0 (poiché significa neutralità non raggiunta, è servito un ulteriore ciclo di clock).
A questo punto, il conteggio sempre aggiornato che esce dal mux viene inviato ad un registro che lo memorizza e lo rimette in circolo e ad un multiplexer a 2 ingressi da 8 bit ciascuno che lo mette nella porta output NCLK solamente quando il segnale FINE_OPERAZIONE che arriva dalla FSM vale 1.
Se FINE_OPERAZIONE vale 0, entrambi i mux del PH_FINALE e del NCLK danno in uscita 0 (codificato in modulo ad 8 bit).

STATISTICHE DEL CIRCUITO

Descriviamo qui di seguito le statistiche del circuito prima e dopo l'ottimizzazione.
Partiamo innanzitutto dalla FSM, provando a ridurre gli stati con il comando *state_minimize stamina*; vediamo che non ci sarà una riduzione in quanto gli stati sono già minimizzati:

```
sis> rl controllore.blif
sis> state_minimize stamina
Running stamina, written by June Rho, University of Colorado at Boulder
Number of states in original machine : 4
Number of states in minimized machine : 4
```

Facciamo quindi l'assegnazione degli stati con il comando *state_assign jedi* e stampiamo le statistiche dell'FSM:

```
sis> state_assign jedi
Running jedi, written by Bill Lin,  UC Berkeley
sis> print_stats
CONTROLLORE     pi=11   po= 6   nodes=  8       latches= 2
lits(sop)= 161  #states(STG)=   4
```

Creiamo le funzioni di output e di stato prossimo con il comando *stg_to_network* e procediamo alla minimizzazione di λ e δ con il comando *source script.rugged* e con altri comandi come *sweep, eliminate, fx, resub, simplify* e *full_simplify* inseriti in un nostro script riusciamo a ridurle ulteriormente. Stampiamo le statistiche dopo la migliore ottimizzazione ottenuta (considerando l'ottimizzazione per area, quindi meno letterali):

```
sis> print_stats
CONTROLLORE     pi=11   po= 6   nodes=  9       latches= 2
lits(sop)=  38  #states(STG)=   4
```

Facciamo la stessa cosa per ottimizzare il datapath, ed otteniamo:
Prima dell'ottimizzazione:

```
sis> rl datapath.blif
Warning: network `SOTTRATTORE8', node "[59]" does not fanout
Warning: network `SOTTRATTORE8', node "[123]" does not fanout
Warning: network `DATAPATH', node "[59]" does not fanout
Warning: network `DATAPATH', node "[91]" does not fanout
Warning: network `DATAPATH', node "[225]" does not fanout
Warning: network `DATAPATH', node "[123]" does not fanout
sis> print_stats
DATAPATH        pi=12   po=17   nodes=186       latches=16
lits(sop)= 908
```

I warning “does not fanout” li ignoriamo in quanto non ci danno problemi al circuito; sono dovuti al carry out che otteniamo dalle somme all’interno del datapath ma che non ci serve e va in overflow.

Dopo l’ottimizzazione:

```
sis> print_stats
DATAPATH        pi=12   po=17   nodes= 53       latches=16
lits(sop)= 226
```

Ora analizziamo le statistiche della FSMD a cui abbiamo collegato FSM e datapath ottimizzati:

```
sis> print_stats
FSMD            pi=10   po=20   nodes= 63       latches=18
lits(sop)= 265
```

Proviamo ad ottimizzare ulteriormente la FSMD e ne stampiamo le statistiche:

```
sis> print_stats
FSMD            pi=10   po=20   nodes= 58       latches=18
lits(sop)= 257
```

MAPPATURA

Con il mapping tecnologico per area, caricando prima la libreria dei componenti con *read_library synch.genlib* e usando poi il comando *map -m 0 -s*, otteniamo le seguenti statistiche di mappatura tra cui numero di gates e ritardo:

```
>>> before removing serial inverters <<<
# of outputs:           38
total gate area:        5624.00
maximum arrival time: (34.40,34.40)
maximum po slack:     (-5.40,-5.40)
minimum po slack:     (-34.40,-34.40)
total neg slack:      (-809.20,-809.20)
# of failing outputs:  38
>>> before removing parallel inverters <<<
# of outputs:           38
total gate area:        5624.00
maximum arrival time: (34.40,34.40)
maximum po slack:     (-5.40,-5.40)
minimum po slack:     (-34.40,-34.40)
total neg slack:      (-809.20,-809.20)
# of failing outputs:  38
# of outputs:           38
total gate area:        5416.00
maximum arrival time: (33.60,33.60)
maximum po slack:     (-5.40,-5.40)
minimum po slack:     (-33.60,-33.60)
total neg slack:      (-799.40,-799.40)
# of failing outputs:  38
```

SCELTE PROGETTUALI

In fase di progettazione ed esecuzione del progetto abbiamo apportato alcune scelte progettuali fondamentali per la realizzazione del circuito:

- Nella FSM abbiamo optato per la realizzazione di 4 stati in modo da poter coprire i 3 casi di ph e lo stato di reset, come si può vedere nello schema riportato sopra.
- Nel datapath, il valore 11 al selettore del mux dedito alla scelta dell'operazione da eseguire per il raggiungimento della neutralità non si presenta mai in quanto gli stati di ph sono 3 e non ci sono altri casi da coprire.
- Il segnale START, input di FSM, viene utilizzato anche nel datapath per l'azzeramento del conteggio dei cicli di clock e per determinare l'acquisizione del ph_iniziale per cominciare il calcolo.
- Analogamente al segnale START sopra citato, l'output della FSM FINE_OPERAZIONE viene mandato al datapath, il quale mette in output PH_FINALE e NCLK solo nel momento in cui esso vale 1.
- Il segnale FINE viene invece utilizzato per mandare un segnale da datapath a FSM, per indicare che i calcoli sono terminati e si deve chiudere la valvola in quanto non è più necessaria una correzione della sostanza.
- Abbiamo inoltre ipotizzato che, una volta raggiunto lo stato NEUTRO, il segnale di FINE_OPERAZIONE, il PH_FINALE e il NCLK rimarranno in output fino a quando non si prema il tasto RESET.