Università degli Studi di Verona
Facoltà di Scienze MM.FF.NN.
AA 2015/2016
Corso di laurea in Informatica

Elaborato SIS
Laboratorio Architettura degli Elaboratori

Università degli studi di Verona
Corso di laurea Informatica

Mori Samuele (VR397217) - Rocchetti Emmanuel (VR398432)
15/02/2016

Indice:

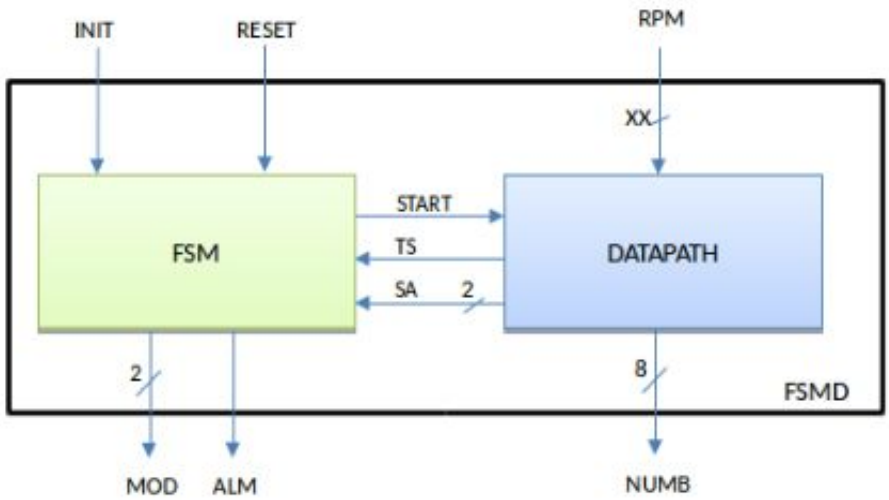
Schema generale del circuito

Il nostro dispositivo permette il monitoraggio di un motore a combustione interna basato su un circuito sequenziale che riceve come input il numero di giri/minuto del motore (RPM) e fornisce in uscita una modalità di funzionamento del motore: sotto-giri (SG), in regime ottimale (OPT) o fuori-giri (FG). In uscita restituisce inoltre da quanto tempo il sistema si trova nello stato attuale ed un ulteriore output di allarme che vale 1 se e soltanto se il sistema è in stato FG da più di 15 secondi (cicli di clock).
Il circuito è composto da un controllore e da un datapath con i seguenti ingressi e uscite.

- INIT[1]: quando vale 1 indica che il circuito ha iniziato la rilevazione del numero di giri. Il controllore passa nello stato di conteggio dei secondi trascorsi nell'attuale modalità. Finché vale 0 non svolge alcun conteggio né indica alcun valore in uscita.
- RESET[1]: se posto a 1 il controllore si resetta, ovvero il contatore dei secondi viene posto a zero.
- RPM[13]: valore del numero di giri ricevuto dal rilevatore (valore massimo 6500).
- MOD[2]: indica in quale modalità di funzionamento si trova l'apparecchio al momento corrente (00 – spento, 01 – SG, 10 – OPT, 11 – FG).
- NUMB[8]: indica i secondi trascorsi nell'attuale modalità.
- ALM[1]: segnala il superamento del tempo limite in FG.

Il controllore è collegato al datapath con tre segnali che hanno il seguente significato:

- START[1]: messo a 1 fa iniziare al datapath la lettura dei RPM, il set dello stato di soglia, e l'inizio del conteggio secondi in soglia.
- SA[2]: imposta il controllore allo stato di soglia attuale.
- TS[1]: segnala al controllore il superamento della soglia di tempo.



Il rilevatore manda un valore RPM al secondo. Tale valore fa parte degli ingressi impostati durante la simulazione. Ad ogni valore RPM in ingresso, il circuito controlla l'attuale soglia, imposta lo stato corrispondente e inizia a contare (o incrementa il contatore). Nel momento in cui il valore RPM non fa parte dell'attuale soglia, il circuito cambia stato e inizia da zero il conteggio. Ad ogni inserimento di RPM, se INIT è attivo e non vi è reset, il circuito riporta i valori aggiornati delle uscite. I valori delle soglie sono i seguenti:

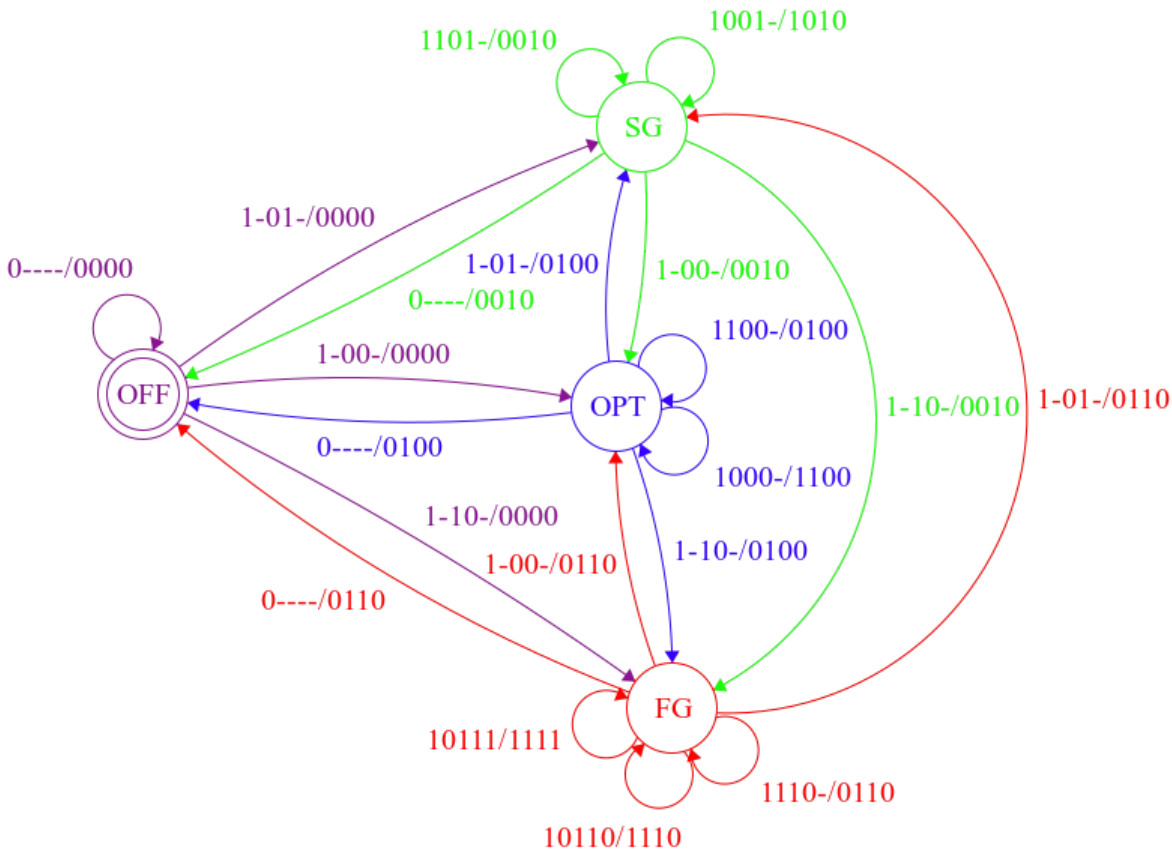
- RPM < 2000 → SG
- 2000 ≤ RPM ≤ 4000 → OPT
- RPM > 4000 → FG

Controllore FSM

Il controllore è una macchina a stati finiti (FSM) di Mealy che presenta questi I/O:
Inputs {INIT, RESET, SA1, SA0, TS}
Outputs {START, MOD1, MOD0, ALM}

Nel nostro STG abbiamo identificato quattro stati

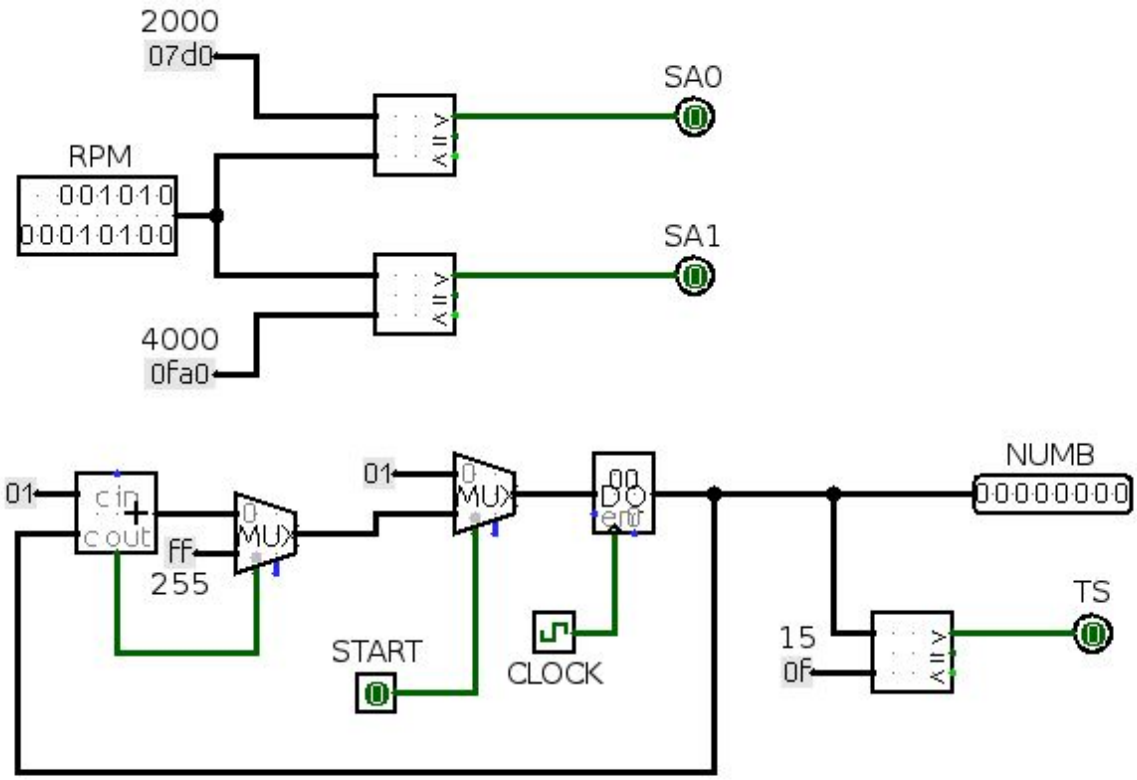
- **ONOFF**: è lo stato di reset, l' FSM rimane in questo stato finché il controllore resta spento (il conteggio rimane invariato a 1);
- **SG**: il controllore arriva in questo stato quando rileva un numero di giri inferiore a 2000, il contatore aumenta senza però attivare l'allarme;
- **OPT**: il controllore arriva in questo stato quando rileva un numero di giri compreso tra 2000 e 4000, il contatore aumenta senza però attivare l'allarme;
- **FG**: il controllore arriva in questo stato quando rileva un numero di giri superiore a 4000 e dopo 15 cicli di clock viene attivato l'allarme.



Datapath

I componenti che abbiamo utilizzato per realizzare il datapath sono:

- SOMMATORE1: usato nel sommatore a otto bit;
- SOMMATORE8: somma il valore NUMB alla costante UNO8 il suo carry-out è il selettore del MUX8 per evitare che superato il valore 255 il contatore si azzeri;
- UNO8: costante a otto bit con valore 1;
- MUX8: multiplexer a due ingressi da otto bit ciascuno utilizzato:
    - nel sommatore per evitare che superato il valore 255 il contatore si azzeri;
    - nel registro NUMB, contenente la variabile contatore; gli ingressi sono la costante 0 espressa in otto bit e il risultato in uscita dal sommatore sempre a otto bit; l'input START fa da selettore tra i due ingressi e ciò corrisponde anche al RESET.
- DUECENTOCINQUANTACINQUE8: genera un valore costante 255 (su 8 bit) usato nel sommatore per evitare che il contatore si resetti dopo il superamento del valore 255;
- REGISTRO8: contiene la variabile NUMB ed ha il suo clock;
- XOR: Operatore logico XOR, utilizzato nel calcolo dei maggiori;
- MAGGIORE8: "maggiore" a 8 bit (restituisce 1 se il numero rappresentato dai primi otto bit è maggiore del numero rappresentato dai successivi otto bit) utilizzato nel controllo dell'allarme, superati i quindici cicli di clock nello stato FG da come output (TS) 1;
- QUINDICI8: genera un valore costante 15 (su 13 bit) usato nel confronto con il contatore per l'attivazione dell'allarme;
- MAGGIORE13: "maggiore" a 13 bit (restituisce 1 se il numero rappresentato dai primi 13 bit è maggiore del numero rappresentato dai successivi 13 bit) utilizzato nei confronti degli RPM (SA1 = RPM>4000, SA0 = 2000>RPM);
- DUEMILA13: genera un valore costante 2000 (su 13 bit) usata nel confronto con gli RPM per stabilire lo stato attuale;
- QUATTROMILA13: genera un valore costante 4000 (su 13 bit) usata nel confronto con gli RPM per stabilire lo stato attuale.


2000
07d0
SA0
RPM
001010
0001010 00
SA1
4000
0fa0
01
cin
cout
ff
255
0
MUX
01
0
MUX
00
DO
enr
NUMB
00000000
TS
START
CLOCK
15
0f

Statistiche circuito

In seguito verranno descritte le statistiche del circuito prima e dopo l'ottimizzazione per area.

FSM

Per la minimizzazione degli stati della FSM abbiamo utilizzato il comando "state_minimize stamina", il numero degli stati è rimasto invariato.
Per la codifica degli stati abbiamo utilizzato il comando "state_assign jedi".

```
sis> rl fsm_base.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
sis> state_assign jedi
Running jedi, written by Bill Lin,  UC Berkeley
sis>
```

La FSM risultante ha le seguenti caratteristiche:

```
samu27:~/workspace $ sis
UC Berkeley, SIS 1.3.6 (compiled 2016-01-23 15:46:56)
sis> rl fsm.blif
Warning: network `fsm.blif', node "RESET" does not fanout
Warning: network `fsm.blif', node "TS" does not fanout
Warning: network `fsm.blif', node "LatchOut_v5" does not fanout
Warning: network `fsm.blif', node "LatchOut_v6" does not fanout
sis> print_stats
fsm             pi= 5   po= 4   nodes=  6       latches= 2
lits(sop)=  31  #states(STG)=   4
```

Che dopo l'ottimizzazione diventano:

```
sis> print_stats
fsm             pi= 5   po= 4   nodes=  5       latches= 2
lits(sop)=  29  #states(STG)=   4
sis>
```

I warning non sono preoccupanti, essi non interferiscono con l'esecuzione del circuito.

Datapath

Il datapath presenta le seguenti caratteristiche:

```
sis> rl datapath.blif
sis> print_stats
DATAPATH        pi=14   po=11   nodes=136       latches= 8
lits(sop)= 564
```

Che dopo l'ottimizzazione diventano:

```
sis> print_stats
DATAPATH        pi=14   po=11   nodes= 17       latches= 8
lits(sop)= 106
sis>
```

FSMD

La FSMD presenta le seguenti caratteristiche:

```
sis> rl controlloMotore.blif
Warning: network `fsm_rugged.blif', node "RESET" does not fanout
Warning: network `fsm_rugged.blif', node "LatchOut_v5" does not fanout
Warning: network `fsm_rugged.blif', node "LatchOut_v6" does not fanout
sis> print_stats
CONTROLLO_MOTORE        pi=15   po=11   nodes= 23       latches=10
lits(sop)= 136
```

Che dopo l'ottimizzazione diventano:

```
sis> source script.rugged
sis> print_stats
CONTROLLO_MOTORE        pi=15   po=11   nodes= 21       latches=10
lits(sop)= 147
```

Mapping

Una volta ottimizzato il circuito, lo abbiamo mappato utilizzando la libreria “synch.genlib”.
Il circuito presenta le seguenti statistiche:

```
samu27:~/workspace $ sis
sis> rl controlloMotore_rugged.blif
sis> print_stats
CONTROLLO_MOTORE        pi=15   po=11   nodes= 21       latches=10
lits(sop)= 147
sis> read_library synch.genlib
sis> map -s
warning: unknown latch type at node '{[13]}' (RISING_EDGE assumed)
warning: unknown latch type at node '{[14]}' (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
>>> before removing serial inverters <<<
# of outputs:          21
total gate area:       2696.00
maximum arrival time: (26.40,26.40)
maximum po slack:     (-0.60,-0.60)
minimum po slack:     (-26.40,-26.40)
total neg slack:      (-200.80,-200.80)
# of failing outputs:  21
>>> before removing parallel inverters <<<
# of outputs:          21
total gate area:       2696.00
maximum arrival time: (26.40,26.40)
maximum po slack:     (-0.60,-0.60)
minimum po slack:     (-26.40,-26.40)
total neg slack:      (-200.80,-200.80)
# of failing outputs:  21
# of outputs:          21
total gate area:       2584.00
maximum arrival time: (25.60,25.60)
maximum po slack:     (-0.60,-0.60)
minimum po slack:     (-25.60,-25.60)
total neg slack:      (-194.40,-194.40)
# of failing outputs:  21
sis> print_stats
CONTROLLO_MOTORE        pi=15   po=11   nodes= 68       latches=10
lits(sop)= 160
```

Scelte progettuali

1) Il contatore superata la soglia di 255 si sarebbe azzerato, nel caso in cui questo fosse accaduto nello stato FG, in pieno allarme, dopo il 256° ciclo di clock l'allarme si sarebbe spento per 15 cicli. Per noi questo era un bug da risolvere e per evitare ciò abbiamo usato un MUX a otto bit i quali ingressi erano l'uscite del sommatore e la costante 255, il suo selettore era il carry-out del sommatore stesso. Una volta superato il 255° ciclo, il carry-out avrebbe attivato come ingresso la costante 255 in modo da non far spegnere l'allarme.

2) Effettuando diverse simulazioni, abbiamo notato che il passagio da uno stato all'altro impiegava un ciclo di clock in più del previsto. Per evitare ciò abbiamo pensato di far partire il conteggio a 1, in modo che i cicli necessari per attivare l'allarme non fossero 17 ma 16.