Università degli Studi di Verona

Facoltà di Scienze MM. FF. NN.

Corso di Laurea in Informatica

Architettura degli Elaboratori

A.A. 2013/2014

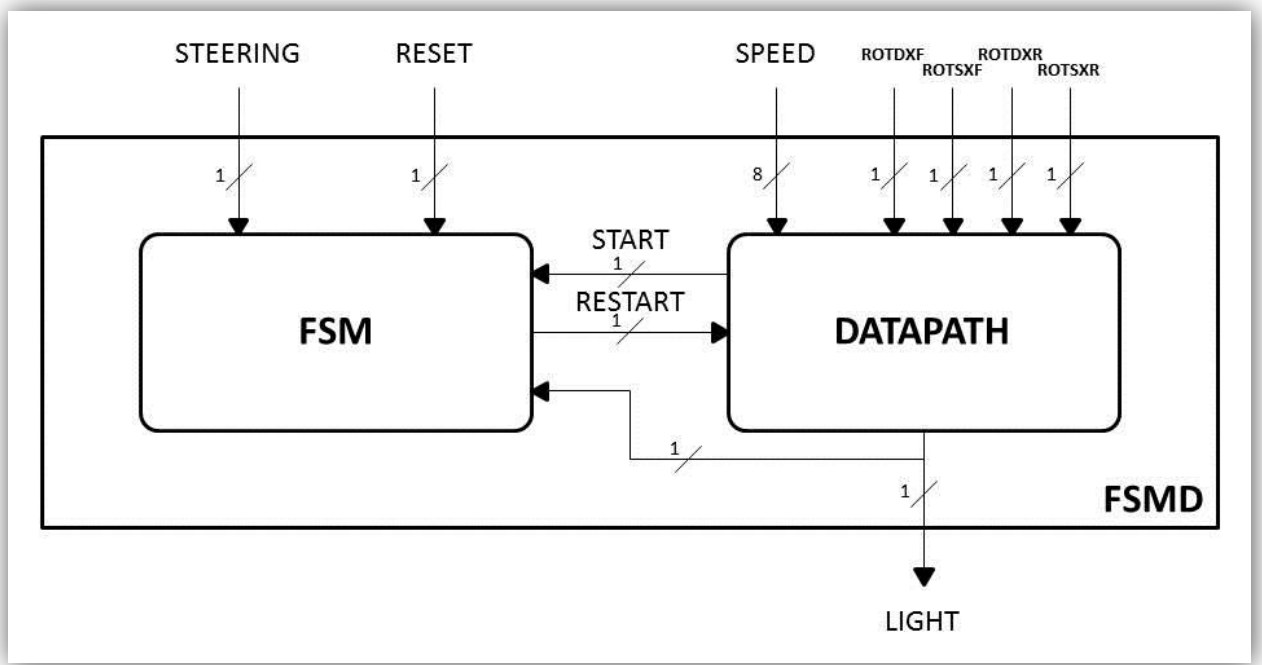
Elaborato SIS

Dispositivo per la rilevazione di foratura pneumatici di un'automobile

Lara Scarpari
Matricola: VR361106

Componenti del circuito sequenziale

FSMD



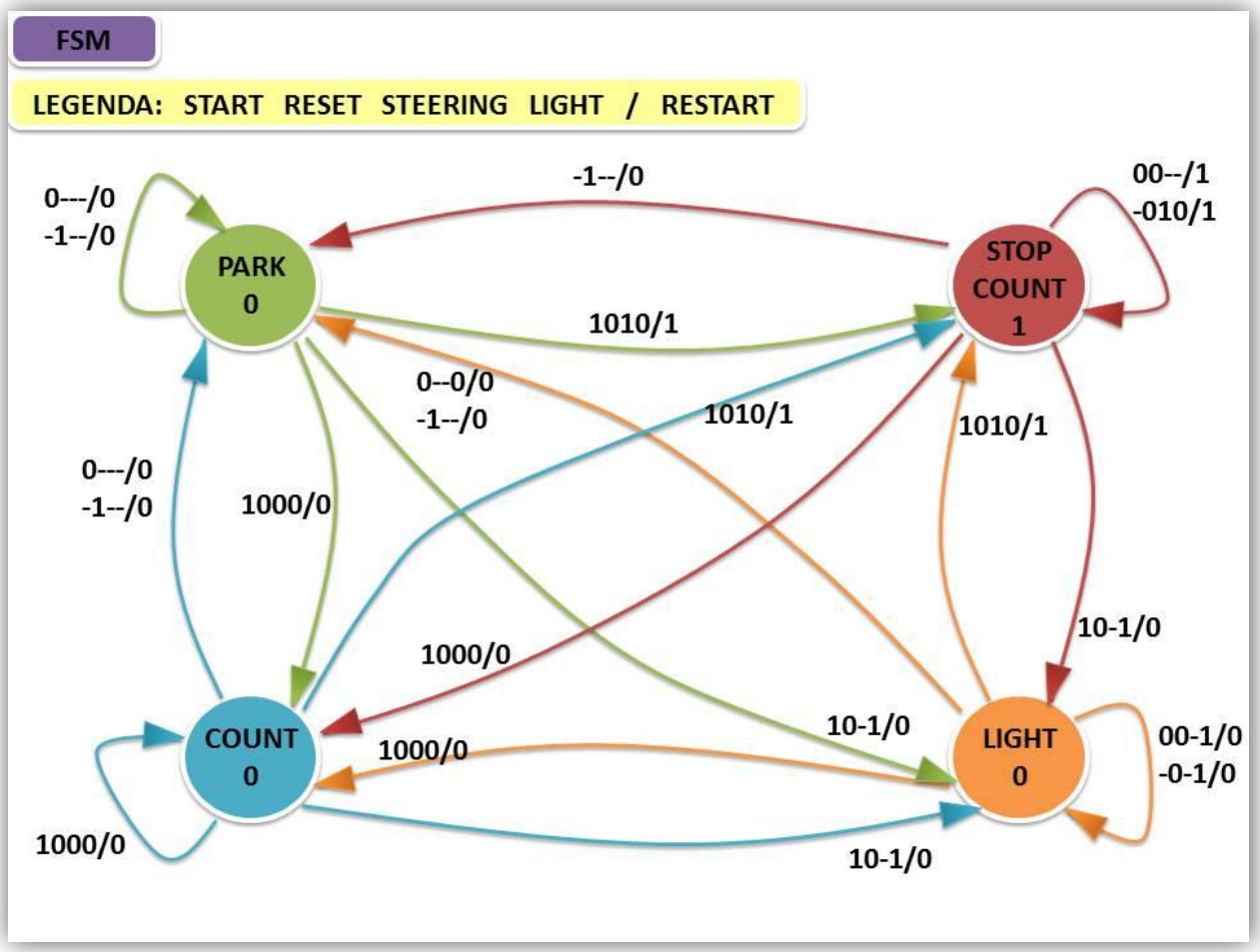
Inputs: *STEERING [1 bit], RESET [1 bit],*
SPEED [8 bit], ROTDXF [1 bit], ROTSXF [1 bit], ROTDXR [1 bit], ROTSXR [1 bit]
Outputs: *LIGHT [1 bit]*

È un circuito sequenziale costituito da FSM (controllore) e DATAPATH.

Questo dispositivo in particolare serve per rilevare la foratura degli pneumatici di una automobile.
Quando l'uscita LIGHT del dispositivo vale 1, indica l'accensione di una spia che comunica il problema.

Il circuito è realizzato dal file FSMD.blif, costituito da FSM_INIZIALE.blif e DATAPATH.blif. Non è stato utilizzato FSM.blif, in cui sono state fatte minimizzazione e codifica degli stati, poiché produceva degli errori.
Tuttavia, componendo FSM_INZIALE.blif e DATAPATH.blif nel file FSMD.blif, si ricevono comunque degli warnings che non si è stati in grado di risolvere.
I file FSM.blif e DATAPATH.blif sembrano funzionare correttamente se testati separatamente, ma riuniti in FSMD.blif, pur funzionando correttamente in molti casi (vedere script di simulazione), per qualche input non danno il risultato atteso.

FSM



Inputs: *START [1 bit], RESET [1 bit], STEERING [1 bit], LIGHT [1 bit]*
Outputs: *RESTART [1 bit]*
Stati: *PARK, COUNT, STOP_COUNT, LIGHT*

Inputs

START è un output del DATAPATH.
Vale 1 se la velocità è maggiore di 10 Km/h.

RESET rappresenta un ingresso che serve per resettare il dispositivo.
Nel nostro caso il bit di reset blocca semplicemente il conteggio dei giri dei pneumatici, senza azzerare il numero di giri già calcolato.

STEERING rappresenta la fase di sterzo.
Vale 1 se l'automobile sta sterzando.

LIGHT è un output del DATAPATH.
Vale 1 se almeno uno pneumatico è forato.

Outputs

RESTART è l'output della FSM e un input del DATAPATH.
Vale 1 se, con una velocità maggiore di 10 Km/h, l'automobile è in fase di sterzo, 0 altrimenti.

Stati

PARK è lo stato iniziale della FSM. L'automobile si trova in questo stato sostanzialmente quando la sua velocità è inferiore a 10 Km/h o quando si comunica al dispositivo di resettarsi.
In questo stato non viene incrementato il numero di giri dei pneumatici in quanto la velocità dell'automobile è troppo bassa. Inoltre lo stato PARK rappresenta una sorta di stato di reset ed è raggiungibile quando RESET vale 1.

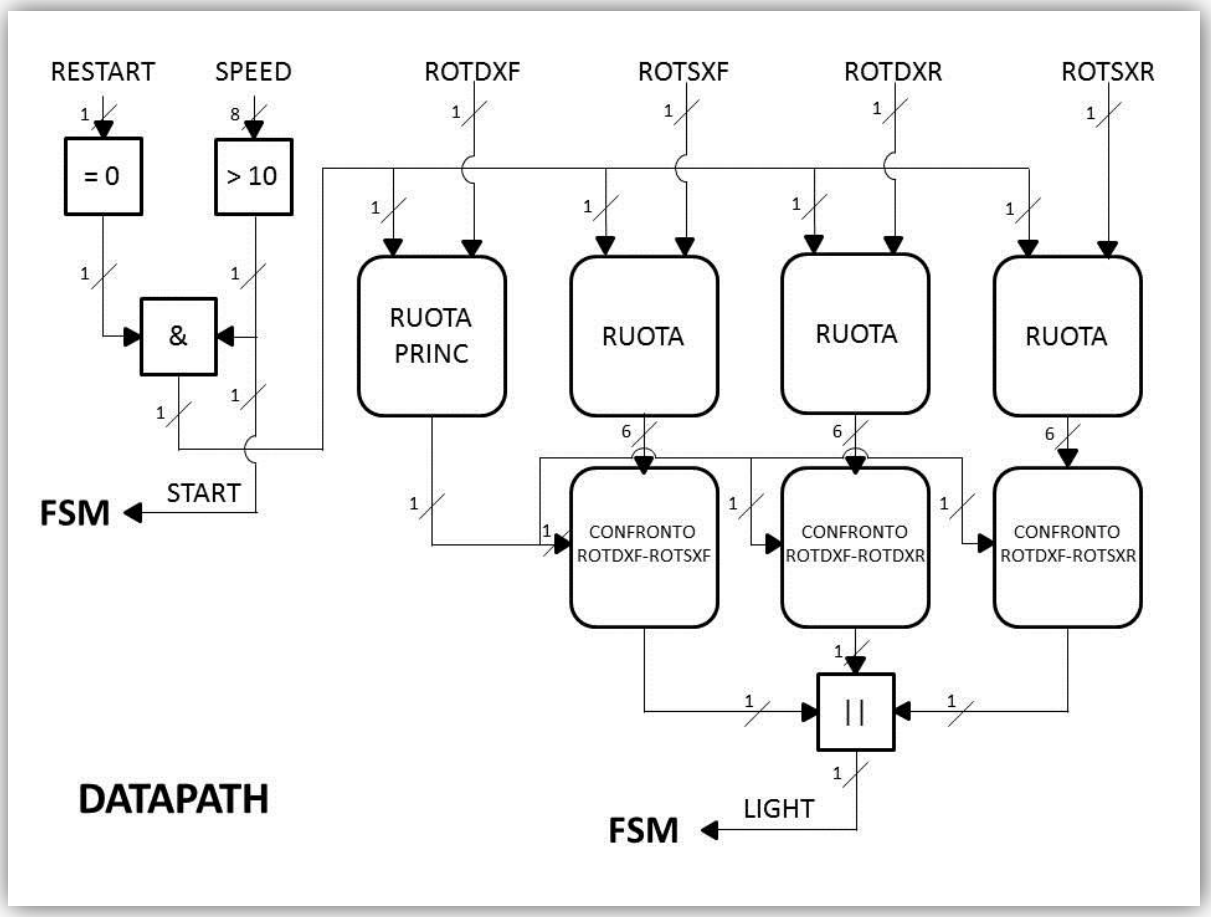
COUNT è lo stato in cui si avviano i conteggi del numero di giri di ogni pneumatico. Questo stato è raggiungibile se e solo se START vale 1 e gli altri ingressi sono posti a 0.

STOP_COUNT è lo stato in cui si interrompono, ma non si azzerano, i conteggi del numero di giri di ogni pneumatico poiché l'automobile è in fase di sterzo. Questo stato è raggiungibile da tutti gli altri stati se e solo se START e STEERING valgono 1, mentre gli altri ingressi sono posti a 0. In questo caso l'automobile ha una velocità maggiore di 10 Km/h ed è in fase di sterzo.

LIGHT è lo stato in cui si accende la spia che indica la foratura di uno pneumatico. Questo stato è raggiungibile dagli altri se e solo START e LIGHT valgono 1 e RESET vale 0, indipendentemente dal fatto che l'automobile stia o meno sterzando.

Nella FSM è stata data maggiore importanza al bit di RESET e al valore della velocità dell'automobile. Infatti in caso di velocità inferiore a 10 Km/h o di indicazione di RESET si è preferito rimanere o ritornare nello stato PARK.
Inoltre, come già accennato, il fatto che l'automobile stia sterzando blocca semplicemente il conteggio del numero di giri dei pneumatici, il quale verrà ripreso non appena l'automobile riprenderà a percorrere un tratto rettilineo con velocità superiore a 10 Km/h.

DATAPATH



Inputs: *RESTART [1 bit], SPEED [8 bit],*
ROTDXF [1 bit], ROTSXF [1 bit], ROTDXR [1 bit], ROTSXR [1 bit]
Outputs: *START [1 bit], LIGHT [1 bit]*

Inputs

RESTART è l'output della FSM.
Vale 1 se, con una velocità maggiore di 10 Km/h, l'automobile è in fase di sterzo, 0 altrimenti.

SPEED rappresenta la velocità dell'automobile.
È espressa in 8 bit e può assumere valori da 0 a 255.

ROTDXF rappresenta la ruota anteriore destra.
Vale 1 se la ruota fa un giro completo, 0 altrimenti.

ROTSXF rappresenta la ruota anteriore sinistra.
Vale 1 se la ruota fa un giro completo, 0 altrimenti.

ROTDXR rappresenta la ruota posteriore destra.
Vale 1 se la ruota fa un giro completo, 0 altrimenti.

ROTSXR rappresenta la ruota posteriore sinistra.
Vale 1 se la ruota fa un giro completo, 0 altrimenti.

Outputs

START è un output del DATAPATH e un input della FSM.
Vale 1 se la velocità dell'automobile risulta maggiore di 10 Km/h, 0 altrimenti.

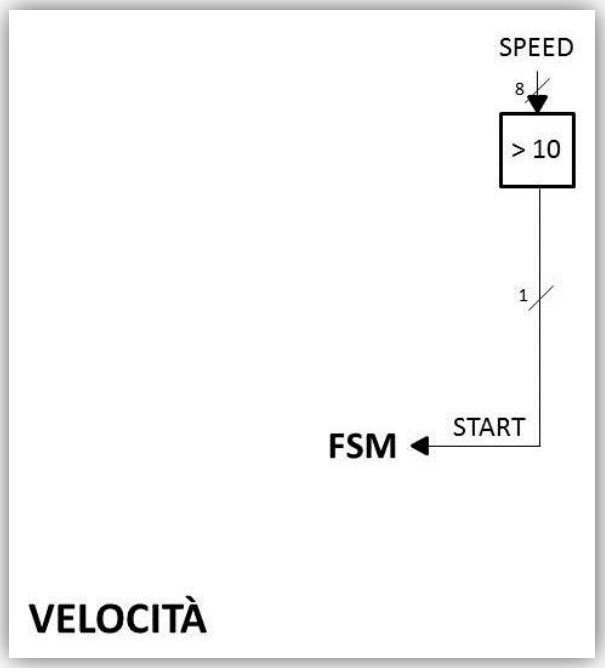
LIGHT è un output del DATAPATH e un input della FSM. Esso è anche l'output del circuito sequenziale.
Vale 1 se è stato riscontrato che almeno uno pneumatico ha effettuato un numero di giri troppo basso per poter sembrare non forato, 0 altrimenti.

Composizione

Il DATAPATH è costituito dai seguenti file con estensione .blif:

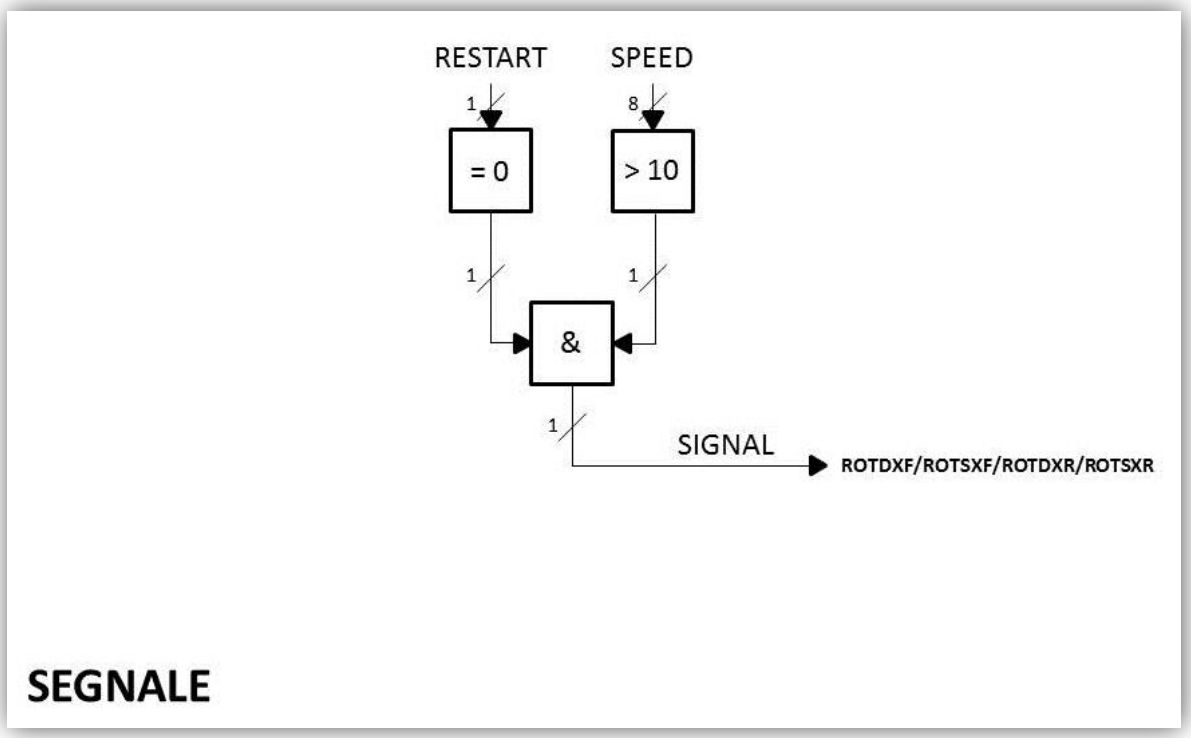
- ***velocita.blif***
- ***segnale.blif***
- ***ruota_princ.blif***
- ***ruota.blif***
- ***differenza.blif***
- ***or3.blif***

File velocita.blif



Restituisce 1 se e solo se gli 8 bit di input rappresentano un valore maggiore di 10.
In questo caso l'output di velocita.blif costituisce l'output START del DATAPATH. Esso indica che la velocità dell'automobile è superiore a 10 Km/h.

File segnale.blif



Fa riferimento ai seguenti file:
zero.blif
uguale_zero.blif
velocita.blif
and.blif

File zero.blif
Restituisce la costante zero a un bit.

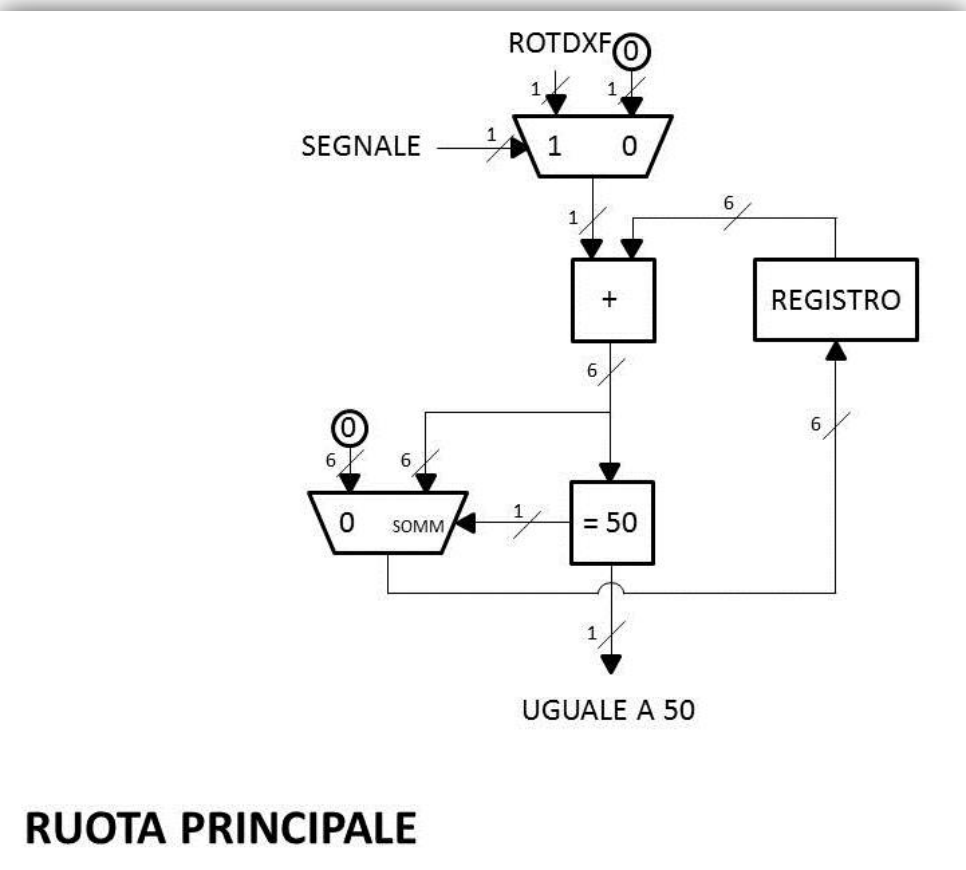
File uguale_zero.blif
Controlla se il bit passato in input sia uguale a 0. Restituisce 1 se e solo se l'input è uguale a zero.

File velocita.blif
Restituisce 1 se e solo se gli 8 bit di input rappresentano un valore maggiore di 10.

File and.blif
Restituisce 1 se e solo se entrambi i bit di input valgono 1.

segnale.blif riceve come input i valori di RESTART (1 bit) e SPEED (8 bit). Se il bit di RESTART vale 0 e la velocità è superiore a 10 km/h, allora restituisce 1, altrimenti restituisce 0.

File ruota_princ.blif



Fa riferimento ai seguenti file:
zero.blif
zero6.blif
mux.blif
contatore6.blif
uguale_cinquanta.blif
mux6.blif
registro6.blif

File zero.blif
Restituisce la costante zero a un bit.

File zero6.blif
Restituisce la costante zero a sei bit.

File mux.blif
Rappresenta un componente, multiplexer, che permette di selezionare un input tra due tramite un apposito segnale di selezione. In questo caso se il segnale restituito da segnale.blif vale 1, il multiplexer selezionerà il valore della ruota, altrimenti selezionerà la costante 0 a un bit.

File contatore6.blif
Riceve in input un numero a 6 bit, che rappresenta il numero eventualmente da incrementare, e una costante, 0 oppure 1, che rappresenta la cifra da sommare. Se la costante in input vale 0, il contatore restituisce il numero invariato. Altrimenti, se la costante vale 1, il contatore restituisce il numero in input incrementato di 1.

File uguale_cinquanta.blif
Riceve in input un numero a 6 bit e restituisce 1 se e solo se esso è uguale a 50.

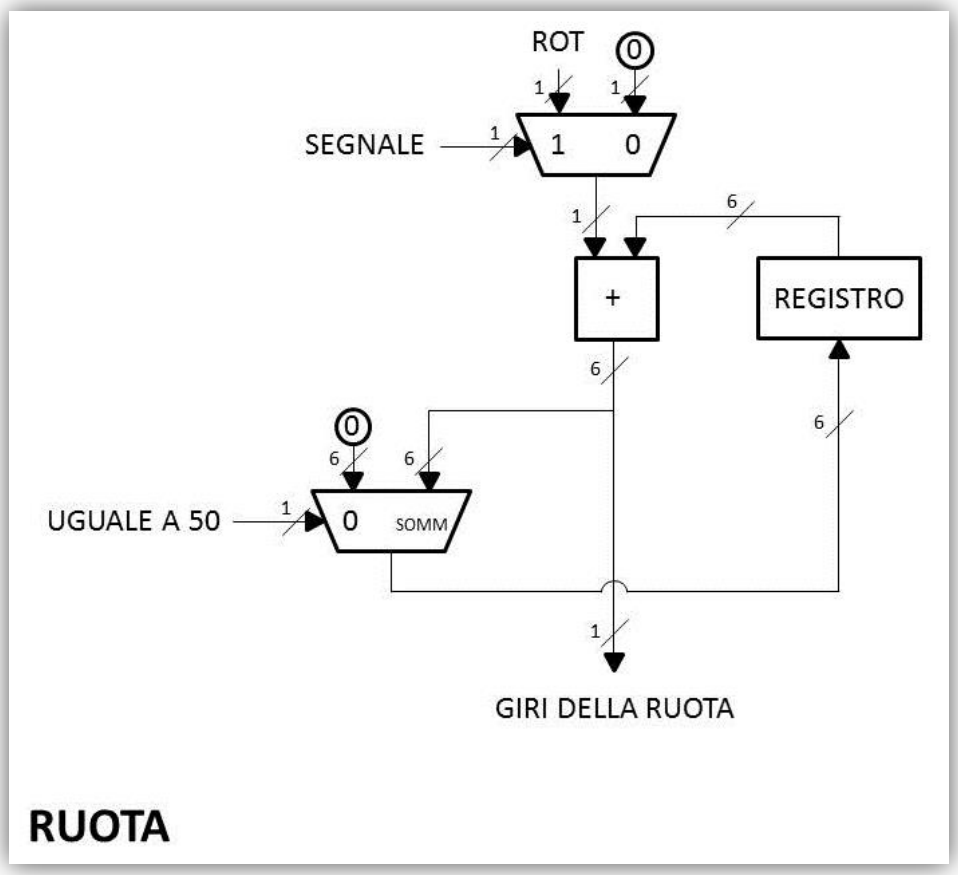
File mux6.blif
Rappresenta un multiplexer avente due ingressi di 6 bit ciascuno. Tramite un segnale il multiplexer seleziona l'uno o l'altro ingresso da restituire come output.

File registro6.blif
Realizza un registro parallelo/parallelo inizializzato ad un valore 0 a 6 bit. Questo registro permette di memorizzare valori numerici a 6 bit.

ruota_princ.blif riceve in input il bit restituito dal file segnale.blif SIGNAL e ROTDXF.
Se SIGNAL vale 0, il multiplexer seleziona la costante 0. Questo caso si verifica quando l'automobile si trova in fase di sterzo oppure ha una velocità troppo bassa. In queste due condizioni il dispositivo non incrementa il numero di giri del pneumatico, ma continua a sommare 0 al numero che rappresenta i giri del pneumatico, il quale rimane invariato.
Altrimenti, se SIGNAL vale 1, il multiplexer seleziona il valore della ruota, che potrebbe essere 0, se la ruota non ha effettuato un giro completo, oppure 1, se la ruota ha effettuato un giro completo.
Il registro, inizializzato a 0, memorizza un numero a 6 bit che rappresenta il numero di giri completi effettuati dalla ruota.
Il contatore prende in input il valore contenuto dal registro e la costante appena selezionata dal multiplexer, sommando al primo la costante e restituendo un numero che potrebbe essere invariato, se è stato sommato 0, oppure incrementato di 1, se è stato sommato 1.
A questo punto il multiplexer deve selezionare quale valore salvare nel registro: il numero restituito dal contatore oppure una costante 0 a 6 bit, la quale serve per azzerare il contenuto del registro.
Nel frattempo un altro componente controlla se il numero appena restituito dal contatore sia uguale a 50. In questo caso il multiplexer seleziona la costante 0 a 6 bit e la restituisce come output, permettendo al registro di salvarla al suo interno e azzerarsi. Altrimenti, se il numero è minore di 50, il multiplexer seleziona il numero restituito dal contatore, il quale si salverà nel registro.
ruota_princ.blif restituisce in output un bit, il quale vale 0 se il numero di giri del pneumatico è inferiore a 50, altrimenti vale 1 se il pneumatico ha effettuato 50 giri.

È stato indicato che in condizioni ottimali la ruota di un'automobile effettua un giro completo ogni 2 metri. Ogni 100 metri i contatori dei giri di ogni pneumatico devono essere resettati. Se la ruota gira sempre correttamente, in 100 metri effettuerà 50 giri circa.
Per resettare tutti i contatori ogni 100 metri, si è preso come riferimento la ruota anteriore destra. Per cui quando quest'ultima arriva a 50 giri, azzera il proprio contatore e lo "comunica" alle altre ruote, che a loro volta andranno ad azzerare i loro contatori.

File ruota.blif

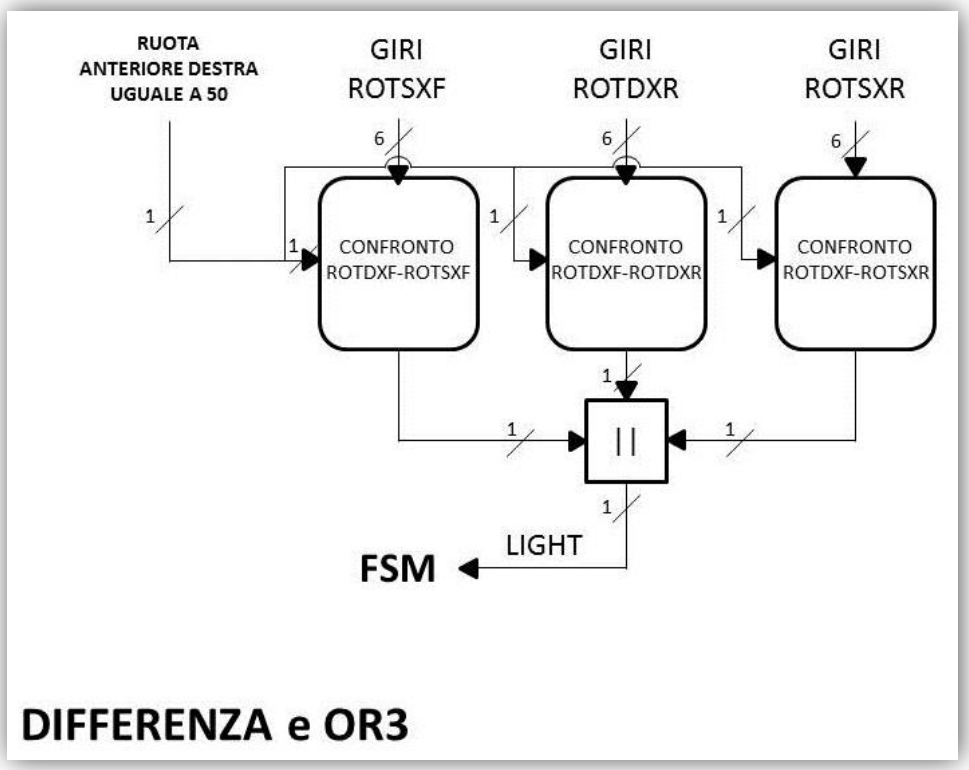


Si occupa del calcolo dei giri di un generico pneumatico dell'automobile, diverso da quello anteriore destro.
Per cui, a seconda della ruota considerata, riceverà in input ROTSXF oppure ROTDXR oppure ROTSXR.

La struttura di questo file è analoga a quella di ruota_princ.blif, ma presenta qualche lieve differenza.

Anzitutto presenta un input in più, cioè un segnale UG50 che corrisponde all'output della ruota anteriore destra e vale 1 se quest'ultima ha raggiunto i 50 giri, altrimenti vale 0. Il multiplexer dovrà leggere ogni volta il valore di questo bit per decidere quale valore restituire al registro perché quest'ultimo possa memorizzarlo al suo interno.
Inoltre ruota.blif restituisce come output il valore del registro appena incrementato dal contatore.
Infine, il contatore utilizzato in questo caso è implementato in contatore6bis.blif. Esso realizza un contatore analogo a contatore6.blif ma conta fino a 60, azzerandosi successivamente. Questo permette di continuare a contare i giri delle ruote anteriore sinistra e posteriore destra e sinistra anche quando la ruota di riferimento per qualche motivo gira più lentamente senza essere bucata. I confronti sui giri dei pneumatici si avviano quando la ruota di riferimento arriva a 50 giri. Per cui se il pneumatico di riferimento effettua in 100 metri 45 giri, mentre gli altri ne effettuano 50, il confronto avverrà quando il primo sarà arrivato a 50 giri e gli altri a 55 giri, supponendo che abbiano girato sempre correttamente. La differenza tra 55 e 50 è infatti minore del 20%.

File differenza.blif



Questo file di occupa di controllare che la differenza tra il numero di giri della ruota di riferimento, cioè quella anteriore destra, e ognuna delle altre tre ruote sia inferiore al 20%.
differenza.blif riceve in input un segnale S, che indica se la ruota anteriore destra ha effettuato 50 giri, e un numero a 6 bit, che rappresenta il numero di giri effettuato dalla ruota che deve essere confrontata con quella di riferimento. Il confronto ha senso solo se il segnale S vale 1
Quindi, se la differenza tra 50, cioè il numero di giri del pneumatico anteriore destro, e il numero di giri del pneumatico da confrontare è inferiore al 20%, allora l'uscita di differenza.blif vale 0. In caso contrario, per cui di differenza superiore al 20%, l'uscita vale 1.

Si è preferito confrontare la ruota anteriore destra con tutte le altre tre ruote e non confrontare le due anteriori e le due posteriori tra loro. Infatti nel caso in cui per esempio siano bucate entrambe le due ruote posteriori, confrontando tra loro i numeri di giri dei pneumatici, potrebbe non essere rilevata alcuna anomalia in quanto la differenza potrebbe risultare sempre inferiore al 20%. Questo problema non dovrebbe verificarsi confrontando la ruota di riferimento con tutte le altre, in quanto anche in presenza di due ruote forate, per esempio le due anteriori, verrebbe riscontrato che almeno una è bucata poiché la ruota anteriore destra viene confrontata con tutte le altre. In questo caso si riscontrerebbe che la ruota anteriore destra è bucata confrontandola con ciascuna delle due ruote posteriori.

File or3.blif

Rappresenta un componente or a 3 bit. Questo ha uscita zero se e solo se tutti e 3 i bit in entrata valgono 0.
or3.blif riceve in input 3 bit, ognuno dei quali rappresenta l'esito di un confronto sul numero di giri tra la ruota "principale" e ciascuna delle altre 3 ruote.
Se tutte le ruote, compresa quella di riferimento, hanno girato correttamente, gli input di OR3 saranno 3 zeri e l'uscita avrà valore 0.
Altrimenti, se almeno una ruota non ha girato correttamente, almeno un input di OR3 sarà 1 e quindi anche la sua uscita avrà valore 1.

L'output di OR3 è LIGHT. Questo è un output del DATAPATH e anche l'unico della FSMD. Esso costituisce inoltre un input per la FSM.

Se LIGHT vale 1, significa che il dispositivo ha rilevato un problema sul numero di giri dei pneumatici su una distanza pari a 100 metri.

Mapping tecnologico

FSM

Caratteristiche iniziali

sis> print_stats

FSM_INIZIALE pi= 4 po= 1 nodes= 1 latches= 0
lits(sop)= 0 #states(STG)= 4

Mapping tecnologico

File script.mapping_fsm

read_blif FSM_INIZIALE.blif
state_minimize stamina
state_assign jedi
fx
source script.rugged
read_library synch.genlib
map -m 0
print_map_stats
print_gate

sis> source script.mapping_fsm

Running stamina, written by June Rho, University of Colorado at Boulder
Number of states in original machine : 4
Number of states in minimized machine : 2

Running jedi, written by Bill Lin, UC Berkeley
warning: unknown latch type at node '{[6]}' (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

Caratteristiche finali

sis> print_map_stats

Total Area = 200.00
Gate Count = 5
Buffer Count = 1
Inverter Count = 1
Most Negative Slack = -5.40
Sum of Negative Slacks = -10.80
Number of Critical PO = 2

sis> print_gate

[505] nand2_comb 24.00
[518] oai22_comb 40.00
[535] invor_comb 32.00
[6] dff_re 88.00
{RESTART} inv_comb 16.00

sis> print_stats

FSM_INIZIALE pi= 4 po= 1 nodes= 5 latches= 1
lits(sop)= 10 #states(STG)= 2

DATAPATH

Caratteristiche iniziali

sis> print_stats

DATAPATH pi=13 po= 2 nodes= 90 latches=24
lits(sop)=9471

Mapping tecnologico

File script.mapping_datapath

read_blif DATAPATH.blif
fx
full_simplify
fx
full_simplify
source script.rugged
fx
read_library synch.genlib
map -m 0
print_map_stats
print_gate

sis> source script.mapping_datapath

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

Caratteristiche finali

sis> print_map_stats

Total Area = 6992.00
Gate Count = 192

Buffer Count = 3
Inverter Count = 51
Most Negative Slack = -28.40
Sum of Negative Slacks = -673.80
Number of Critical PO = 26

sis> print_gate

[1337] nand2_comb 24.00
[1345] or2_comb 32.00
[804] aoi12_comb 32.00
[887] invand_comb 32.00
{START} or4_comb 48.00
[1346] invor_comb 32.00
[1347] invor_comb 32.00
[1348] invor_comb 32.00
[1338] inv_comb 16.00
[1846] nand2_comb 24.00
[1349] nand2_comb 24.00
[302] nor2_comb 24.00
[1335] inv_comb 16.00
[1786] inv_comb 16.00
[1354] nand3_comb 32.00
[1355] invor_comb 32.00
[806] aoi22_comb 40.00
[1336] inv_comb 16.00
[1356] oai22_comb 40.00
[807] aoi22_comb 40.00
[1334] inv_comb 16.00
[808] aoi22_comb 40.00
[1588] oai22_comb 40.00
[1581] inv_comb 16.00
[881] nor2_comb 24.00
[2082] oai12_comb 32.00
[1578] oai12_comb 32.00
[1259] invand_comb 32.00
[1358] nand2_comb 24.00
[1580] inv_comb 16.00
ROT1_O5MUX6 dff_reset_re 104.00
ROT1_O4MUX6 dff_reset_re 104.00
[1359] nand2_comb 24.00
[1350] inv_comb 16.00
[2197] aoi12_comb 32.00
ROT1_I3REG6 dff_re 88.00
[1351] inv_comb 16.00
[1932] oai12_comb 32.00
[2198] and3_comb 40.00
ROT1_I2REG6 dff_re 88.00
[1589] inv_comb 16.00
ROT1_O1MUX6 dff_reset_re 104.00
[1807] nand3_comb 32.00
[2066] nand4_comb 40.00
[2200] nand2_comb 24.00
[894] dff_re 88.00
[1364] inv_comb 16.00
[1369] inv_comb 16.00
[1370] inv_comb 16.00
[1371] nand2_comb 24.00
[1368] invor_comb 32.00
[1860] inv_comb 16.00
[1755] oai12_comb 32.00
[1372] invor_comb 32.00
[1893] inv_comb 16.00
[1373] inv_comb 16.00
[1374] nand2_comb 24.00
[281] nor2_comb 24.00
[1377] nand3_comb 32.00
[793] aoi12_comb 32.00
[1378] oai22_comb 40.00
[2054] or3_comb 40.00
[2056] oai12_comb 32.00
[1511] nand2_comb 24.00
[1512] inv_comb 16.00
O5MUX6 dff_reset_re 104.00
[1514] inv_comb 16.00
O4MUX6 dff_reset_re 104.00
[1375] invor_comb 32.00
[1380] nand2_comb 24.00
[1381] nand2_comb 24.00
[1382] or2_comb 32.00
[2050] nand3_comb 32.00
[1384] oai12_comb 32.00
[1519] inv_comb 16.00
O3MUX6 dff_reset_re 104.00
[2040] nand3_comb 32.00
[1386] oai12_comb 32.00
[1940] oai22_comb 40.00
[1941] inv_comb 16.00
[1387] oai12_comb 32.00
[1523] inv_comb 16.00
O2MUX6 dff_reset_re 104.00
[1812] inv_comb 16.00
[1145] invand_comb 32.00
[1365] inv_comb 16.00
[1942] nand2_comb 24.00
[1390] ao222_comb 72.00
[1526] inv_comb 16.00
O1MUX6 dff_reset_re 104.00
[1739] inv_comb 16.00
[848] nor2_comb 24.00
[1391] oai12_comb 32.00
O0MUX6 dff_reset_re 104.00
[1392] inv_comb 16.00
[1397] inv_comb 16.00
[1398] inv_comb 16.00
[1399] nand2_comb 24.00
[1396] invor_comb 32.00
[1864] inv_comb 16.00
[1758] oai12_comb 32.00
[1400] invor_comb 32.00
[1901] inv_comb 16.00
[1401] inv_comb 16.00
[1402] nand2_comb 24.00
[282] nor2_comb 24.00
[1405] nand3_comb 32.00
[798] aoi12_comb 32.00
[1406] oai22_comb 40.00
[2020] or3_comb 40.00
[2022] oai12_comb 32.00
[1532] nand2_comb 24.00
[1533] inv_comb 16.00
[135] dff_reset_re 104.00
[1535] inv_comb 16.00
[136] dff_reset_re 104.00
[1403] invor_comb 32.00
[1408] nand2_comb 24.00
[1409] nand2_comb 24.00
[1410] or2_comb 32.00
[2016] nand3_comb 32.00
[1412] oai12_comb 32.00
[1540] inv_comb 16.00
[137] dff_reset_re 104.00
[2006] nand3_comb 32.00
[1414] oai12_comb 32.00
[1950] oai22_comb 40.00
[1951] inv_comb 16.00
[1415] oai12_comb 32.00
[1544] inv_comb 16.00
[138] dff_reset_re 104.00
[1803] inv_comb 16.00
[1043] invand_comb 32.00
[1393] inv_comb 16.00
[1952] nand2_comb 24.00
[1418] ao222_comb 72.00
[1547] inv_comb 16.00
[139] dff_reset_re 104.00
[1743] inv_comb 16.00
[827] nor2_comb 24.00
[1419] oai12_comb 32.00
[140] dff_reset_re 104.00
[1421] inv_comb 16.00
[1423] inv_comb 16.00
[1428] invor_comb 32.00
[1429] invor_comb 32.00
[1430] invor_comb 32.00
[1431] invor_comb 32.00
[1992] or4_comb 48.00
[1426] inv_comb 16.00
[1797] nand2_comb 24.00
[1850] nand2_comb 24.00
[1001] nor2_comb 24.00
[802] aoi12_comb 32.00
[1420] inv_comb 16.00
[1434] oai12_comb 32.00
[1553] inv_comb 16.00
[161] dff_reset_re 104.00
[820] nor3_comb 32.00
[1840] invor_comb 32.00
[983] nor2_comb 24.00
[975] aoi12_comb 32.00
[1435] nand2_comb 24.00
[1439] oai12_comb 32.00
[1560] inv_comb 16.00
[162] dff_reset_re 104.00
[1982] nand2_comb 24.00
[1958] oai12_comb 32.00
[1440] nand2_comb 24.00
[163] dff_reset_re 104.00
[1424] inv_comb 16.00
[1926] nand2_comb 24.00
[1442] nand3_comb 32.00
[164] dff_reset_re 104.00
[1425] inv_comb 16.00
[1928] nand2_comb 24.00
[1444] nand3_comb 32.00
[165] dff_reset_re 104.00
[1446] nand3_comb 32.00
[166] dff_reset_re 104.00
[1563] inv_comb 16.00
[1455] oai22_comb 40.00
[1529] inv_comb 16.00
[811] nor4_comb 40.00
[809] and4_comb 48.00
[1550] inv_comb 16.00
[810] nor4_comb 40.00
[913] nor4_comb 40.00
[1458] nand2_comb 24.00
[814] nor2_comb 24.00
[904] nor3_comb 32.00
{LIGHT} aoi12_comb 32.00

sis> print_stats

DATAPATH pi=13 po= 2 nodes=192 latches=24
lits(sop)= 432

FSMD

Caratteristiche iniziali

sis> print_stats

FSMD pi=14 po= 1 nodes= 91 latches=24
lits(sop)=9471

Mapping tecnologico

File script.mapping_fsmd

read_blif FSMD.blif
full_simplify
full_simplify
full_simplify
fx
full_simplify
fx
source script.rugged
fx
read_library synch.genlib
map -m 0
print_map_stats
print_gate

sis> source script.mapping_fsmd

Warning: network `FSMD', node "STEERING" does not fanout
Warning: network `FSMD', node "RESET" does not fanout
Warning: network `FSMD', node "START" does not fanout
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

Caratteristiche finali

sis> print_map_stats

Total Area = 6184.00
Gate Count = 165
Buffer Count = 5
Inverter Count = 39
Most Negative Slack = -37.60
Sum of Negative Slacks = -810.00
Number of Critical PO = 25

sis> print_gate

[1382] inv_comb 16.00
[1383] inv_comb 16.00
[1392] or2_comb 32.00
[882] aoi12_comb 32.00
[984] invand_comb 32.00
[1511] or4_comb 48.00
[1393] nand2_comb 24.00
[2085] nand2_comb 24.00
[1901] oai12_comb 32.00
[1395] or2_comb 32.00
[980] nor2_comb 24.00
[1385] nand2_comb 24.00
[1558] oai12_comb 32.00
[2079] nand2_comb 24.00
[1398] nand2_comb 24.00
[1341] nor2_comb 24.00
[976] nor2_comb 24.00
[2071] oai12_comb 32.00
[1835] inv_comb 16.00
[974] nor3_comb 32.00
[2073] oai12_comb 32.00
[1401] nand2_comb 24.00
[1327] nor2_comb 24.00
[2067] nand3_comb 32.00
[2069] oai12_comb 32.00
[2167] nand2_comb 24.00
[1315] nor2_comb 24.00
[2063] nand3_comb 32.00
[1406] or2_comb 32.00
[2065] oai12_comb 32.00
[2166] nand2_comb 24.00
[1407] or2_comb 32.00
[965] nor2_comb 24.00
[883] nor3_comb 32.00
[1409] oai22_comb 40.00
[961] nor3_comb 32.00
[1384] inv_comb 16.00
[1412] nand3_comb 32.00
[1795] nand2_comb 24.00
[962] aoi12_comb 32.00
[2164] or2_comb 32.00
[991] dff_re 88.00
[2165] inv_comb 16.00
[1796] dff_re 88.00
[992] dff_re 88.00
[993] dff_re 88.00
[1551] inv_comb 16.00
ROT1_O1MUX6 dff_reset_re 104.00
[2169] inv_comb 16.00
[1559] dff_re 88.00
[1414] inv_comb 16.00
[1415] inv_comb 16.00
[1416] inv_comb 16.00
[1854] nand4_comb 40.00
[1856] inv_comb 16.00
[1420] nand2_comb 24.00
[2051] nand2_comb 24.00
[1929] oai12_comb 32.00
[1423] nand2_comb 24.00
[1620] inv_comb 16.00
[957] nor3_comb 32.00
[889] aoi12_comb 32.00
[1421] inv_comb 16.00
[1425] oai22_comb 40.00
[1426] nand2_comb 24.00
[1427] or2_comb 32.00
[1235] nor2_comb 24.00
[888] aoi12_comb 32.00
[1429] oai22_comb 40.00
[387] nor2_comb 24.00
[1431] xor_comb 40.00
[386] nor2_comb 24.00
[1433] xor_comb 40.00
[368] nor2_comb 24.00
[898] xorbar_comb 48.00
O5MUX6 dff_reset_re 104.00
[1601] inv_comb 16.00
O4MUX6 dff_reset_re 104.00
[1606] inv_comb 16.00
O3MUX6 dff_reset_re 104.00
[1610] inv_comb 16.00
O2MUX6 dff_reset_re 104.00
[1615] inv_comb 16.00
O1MUX6 dff_reset_re 104.00
O0MUX6 dff_reset_re 104.00
[1436] inv_comb 16.00
[1437] inv_comb 16.00
[1438] inv_comb 16.00
[1857] nand4_comb 40.00
[1859] inv_comb 16.00
[1442] nand2_comb 24.00
[2021] nand2_comb 24.00
[1935] oai12_comb 32.00
[1445] nand2_comb 24.00
[1685] inv_comb 16.00
[941] nor3_comb 32.00
[892] aoi12_comb 32.00
[1443] inv_comb 16.00
[1447] oai22_comb 40.00
[1448] nand2_comb 24.00
[1449] or2_comb 32.00
[1153] nor2_comb 24.00
[891] aoi12_comb 32.00
[1451] oai22_comb 40.00
[390] nor2_comb 24.00
[1453] xor_comb 40.00
[389] nor2_comb 24.00
[1455] xor_comb 40.00
[369] nor2_comb 24.00
[899] xorbar_comb 48.00
[142] dff_reset_re 104.00
[1666] inv_comb 16.00
[143] dff_reset_re 104.00
[1671] inv_comb 16.00
[144] dff_reset_re 104.00
[1675] inv_comb 16.00
[145] dff_reset_re 104.00
[1680] inv_comb 16.00
[146] dff_reset_re 104.00
[147] dff_reset_re 104.00
[1458] inv_comb 16.00
[1459] inv_comb 16.00
[1460] inv_comb 16.00
[1860] nand4_comb 40.00
[1862] inv_comb 16.00
[1464] nand2_comb 24.00
[1991] nand2_comb 24.00
[1941] oai12_comb 32.00
[1467] nand2_comb 24.00
[1750] inv_comb 16.00
[925] nor3_comb 32.00
[895] aoi12_comb 32.00
[1465] inv_comb 16.00
[1469] oai22_comb 40.00
[1470] nand2_comb 24.00
[1471] or2_comb 32.00
[1071] nor2_comb 24.00
[894] aoi12_comb 32.00
[1473] oai22_comb 40.00
[393] nor2_comb 24.00
[1475] xor_comb 40.00
[392] nor2_comb 24.00
[1477] xor_comb 40.00
[370] nor2_comb 24.00
[900] xorbar_comb 48.00
[168] dff_reset_re 104.00
[1731] inv_comb 16.00
[169] dff_reset_re 104.00
[1736] inv_comb 16.00
[170] dff_reset_re 104.00
[1740] inv_comb 16.00
[171] dff_reset_re 104.00
[1745] inv_comb 16.00
[172] dff_reset_re 104.00
[173] dff_reset_re 104.00
[1482] nand3_comb 32.00
[1961] or2_comb 32.00
[903] aoi12_comb 32.00
[1907] oai12_comb 32.00
[1908] inv_comb 16.00
[1905] oai12_comb 32.00
[1906] inv_comb 16.00
[1479] oai22_comb 40.00
[881] nor3_comb 32.00
{LIGHT} nor2_comb 24.00

sis> print_stats

FSMD pi=14 po= 1 nodes=165 latches=24
lits(sop)= 368

Comandi utilizzati e ottimizzazione

read_blif è un comando auto-esplicativo che serve per leggere un file con estensione .blif.

state_minimize stamina è serve per ridurre, ove possibile, il numero di stati, mantenendo il funzionamento della macchina a stati finiti uguale al precedente. Utilizzando una relazione di equivalenza NUMERO_STATI_INIZIALI >= NUMERO_STATI_RIDOTTI.

state_assign jedi è il comando utilizzato per assegnare i letterali in SOP(somma di prodotti) che rappresentano l'effettiva grandezza del circuito.

fx è un comando di estrazione: si può trovare una sotto-espressione comune a due funzioni associabile a due nodi distinti. Così facendo si giunge ad una semplificazione della rete.

source script.rugged è il comando che invoca lo script.rugged contenuto nei file di sistema di sis.

read_library synch.genlib è il comando utilizzato per caricare in memoria la libreria synch.genlib utilizzata per "mappare per area" ossia per rendere fisico tutto il circuito logico: mappando in pratica è possibile capire quanto effettivamente nella realtà sia grande il circuito e quanto semiconduttore (silicio) sia necessario per poterlo creare.

map -m 0 è il vero e proprio comando di mappatura per area. E' praticamente l'opposto di "map -n 1" che serve per mappare per ritardo.

print_map_stats è il comando utilizzato per visualizzare le caratteristiche del circuito dopo l'ottimizzazione e la mappatura per area.

print_gate è il comando con cui si visualizzano le porte logiche e i rispettivi letterali. In pratica permette di capire quanto grande, in termini di letterali spesi, sia una porta (gate).

print_stats visualizza importanti informazioni sul circuito.

PI	numero di segnali in input
PO	numero di segnali in output
nodes	numero di nodi
latches	numero di elementi di memoria
lits	numero di letterali

Le scelte dei comandi per l'ottimizzazione sono dovute principalmente a ridurre il più possibile l'area occupata dal circuito. Quindi con il comando "fx" e "script.rugged" si riesce a raggiungere il massimo livello di ottimizzazione possibile e il minor numero di nodi della rete.

Il mapping tecnologico è stato applicato separatamente a FSM e DATAPATH, producendo rispettivamente fsm_mapping.blif e datapath_mapping.blif. Questi file non sono stati riuniti per creare il file del circuito finale poiché producevano degli errori. È stato quindi applicato il mapping tecnologico anche a FSMD.blif, costituito da FSM_INIZIALE.blif e DATAPATH.blif.