UNIVERSITÀ DI VERONA
Facoltà di scienze ed ingegneria
Anno accademico 2019/2020
Corso di laurea in Informatica

ELABORATO #1

Progettazione centro di esportazione di truciolo

LABORATORIO DI ARCHITETTURA DEGLI ELABORATORI

A CURA DI

STEFANO CALDANA
VR437143

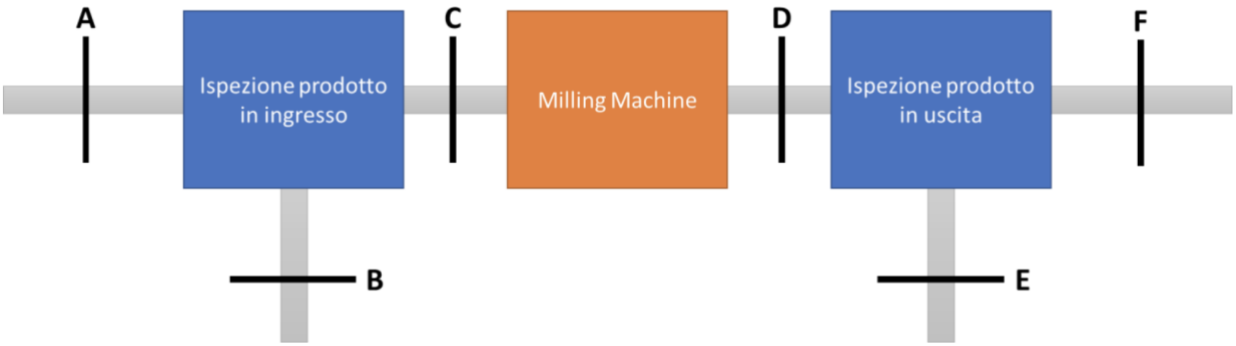
ANDREA TOFFALETTI
VR437535

INDICE

Si riporta l'indice del testo:

SPECIFICHE ELABORATO

Si progetti un dispositivo per la gestione di un centro di lavoro per asportazione di truciolo (fresa). Il sistema, rappresentato schematicamente in figura, è composto da una macchina utensile a controllo numerico (Milling Machine) e due unità di controllo qualità poste prima dell'ingresso e dopo l'uscita del pezzo dalla macchina.



Il dispositivo di controllo prende in ingresso una sequenza di 7 bit, che assumono significato diverso in funzione dello stato in cui si trova l'impianto.

Impianto Off	ON2	ON1	ON0	-	-	-	-
Check In	QI	VIN5	VIN4	VIN3	VIN2	VIN1	VIN0
Milling Machine	EM	VOUT5	VOUT4	VOUT3	VOUT2	VOUT1	VOUT0
Check Out	QO	-	-	-	-	-	-

Il principio di funzionamento dell'impianto è espresso dalle seguenti fasi:

1. L'impianto è inizialmente spento (**O/I=0**). L'impianto si accende quando il comando di accensione (**ON**) di 3 bit è composto dalla sequenza 111 (**ON=111**). L'impianto si accende (**O/I=1**) e inizia a lavorare aprendo il gate A (**GA=1**) e permettendo il caricamento di un pezzo grezzo nell'unità di controllo in ingresso. Il dispositivo deve inoltre incrementare un contatore dei pezzi in ingresso (**NA**);
2. 2. Nel modulo di ispezione prodotti in ingresso viene controllata la qualità del pezzo grezzo (**QI**, 1=buono, 0=scarto) e ne viene misurato il volume (**Vin**) espresso in 6 bit;
    a. Se il pezzo risulta scarto (**QI=0**), il dispositivo deve aprire il gate B (**GB=1**) mandando il pezzo in un deposito di scarti e aumentare il contatore dei pezzi scartati in ingresso (**NB**). Al ciclo successivo il dispositivo ricomincia il ciclo aprendo il gate A;
    b. Se il pezzo risulta buono (**QI**=1) si apre il gate C (**GC**=1) ed il pezzo può passare alla fresa incrementando il contatore **NC**;
3. Nella macchina il pezzo grezzo viene fresato per ottenere un pezzo lavorato; la macchina fornisce un valore binario EM che indica la riuscita o meno della lavorazione, ed il volume del pezzo lavorato (**Vout**) espresso in 6 bit;
    a. se la macchina ha generato un errore (**EM**=0) l'impianto deve spegnersi generando il codice di errore **ERR**=001.
    b. Se la lavorazione è andata a buon fine (**EM**=1), il pezzo viene scaricato attraverso il gate D (**GD**=1) e passa al controllo qualità in uscita. La differenza tra il volume del

pezzo grezzo **Vin** e quello del pezzo lavorato **Vout** è scarto che va a finire nel serbatoio della macchina. Quando la quantità di scarto nella macchina supera il valore 200 l'impianto deve spegnersi generando il codice di errore **ERR**=010.

4. Il prodotto lavorato entra nel modulo di ispezione in uscita che restituisce l'indicatore di qualità del pezzo lavorato (**QO**, 1=buono, 0=scarto).
    a. Se il pezzo risulta buono (**QO**=1), il dispositivo apre il gate F (**GF**=1) per mandare il pezzo a magazzino, incrementando il contatore dei pezzi lavorati correttamente (**NF**).
    b. Se il pezzo risulta scarto (**QO**=0), il dispositivo apre il gate E (**GE**=1) per mandare il pezzo in un deposito scarti lavorati ed aumenta il contatore relativo (**NE**).
5. In entrambi i casi al ciclo successivo la macchina riprende il ciclo aprendo il gate A.

Tutti i contatori dei pezzi (**NA**, **NB**, **NC**, **NE** e **NF**) sono espressi in 4 bit. Nel caso in cui il numero di pezzi nel deposito degli scarti in ingresso (**NB**) generi un overload l'impianto deve spegnersi con codice di errore **ERR**=101. Analogamente per quanto riguarda il deposito degli scarti in uscita (**NE**), generando il codice **ERR**=110.

Ad ogni ciclo di clock il dispositivo deve restituire una sequenza di 30 bit indicativi delle seguenti variabili (nell'ordine di seguito indicato!!!):

O/I ERR GA GB GC GD GE GF NA NB NC NE NF

NB: ad ogni ciclo di clock può essere aperto al massimo un gate!

INTRODUZIONE

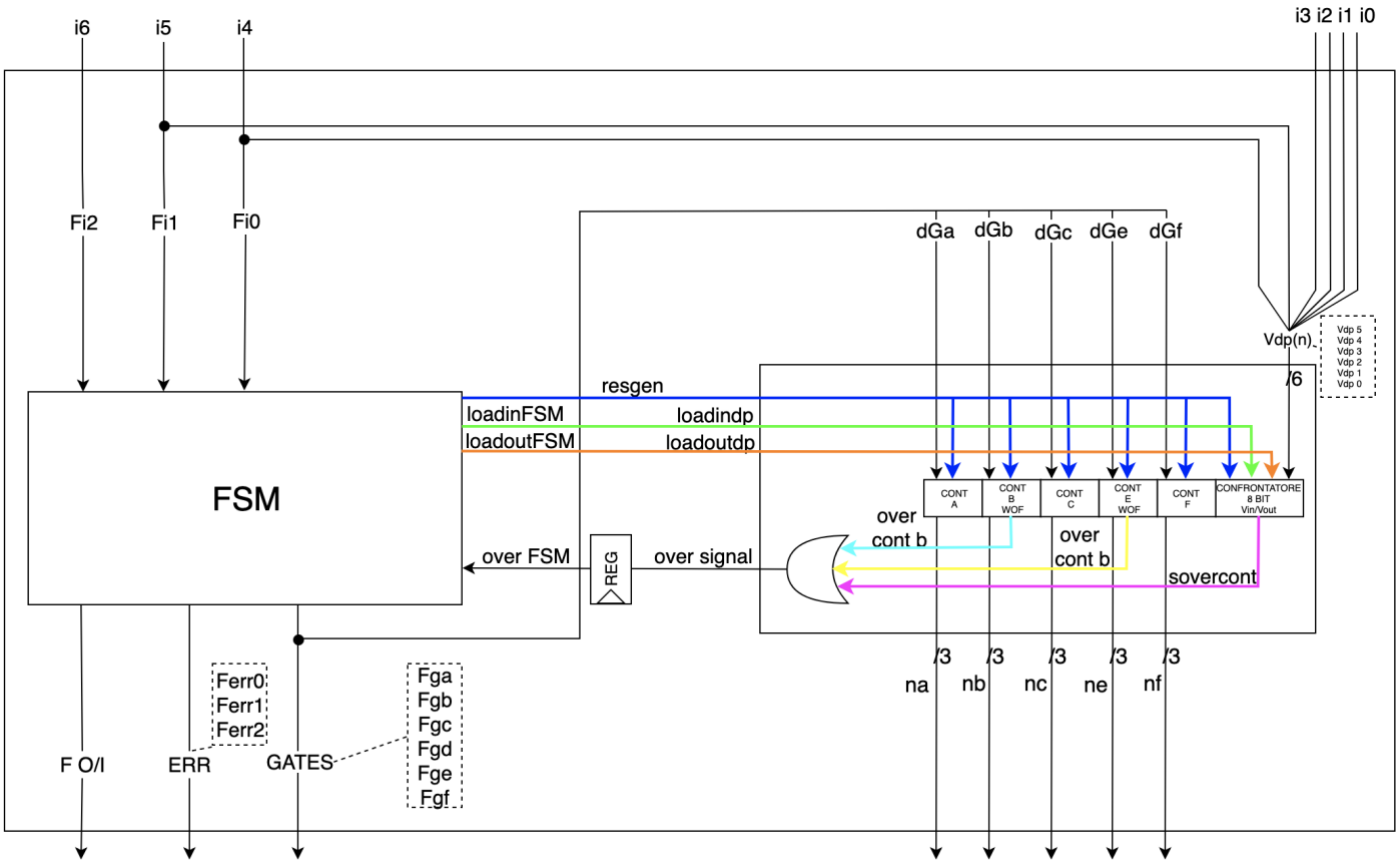
Questa relazione è seguita dai file contenuti nella cartella *./blif/.*

La relazione espone le principali componenti e le principali caratteristiche della macchina, ovvero l'*fsm*, il *data-path* e le sue componenti, la stampa dei ritardi ottenuti e i risultati ottenuti nell'ottimizzazione.

All'interno si trovano due file legati all'*fsmd* prodotta:

1. *Fsmd_beforeop.blif*: è il file contenente la *fsm* prima che la stessa venisse ottimizzata;
2. *Fsm.blif*: è il file contenente la *fsm* dopo l'ottimizzazione e la mappatura.

FSM - DATAPATH



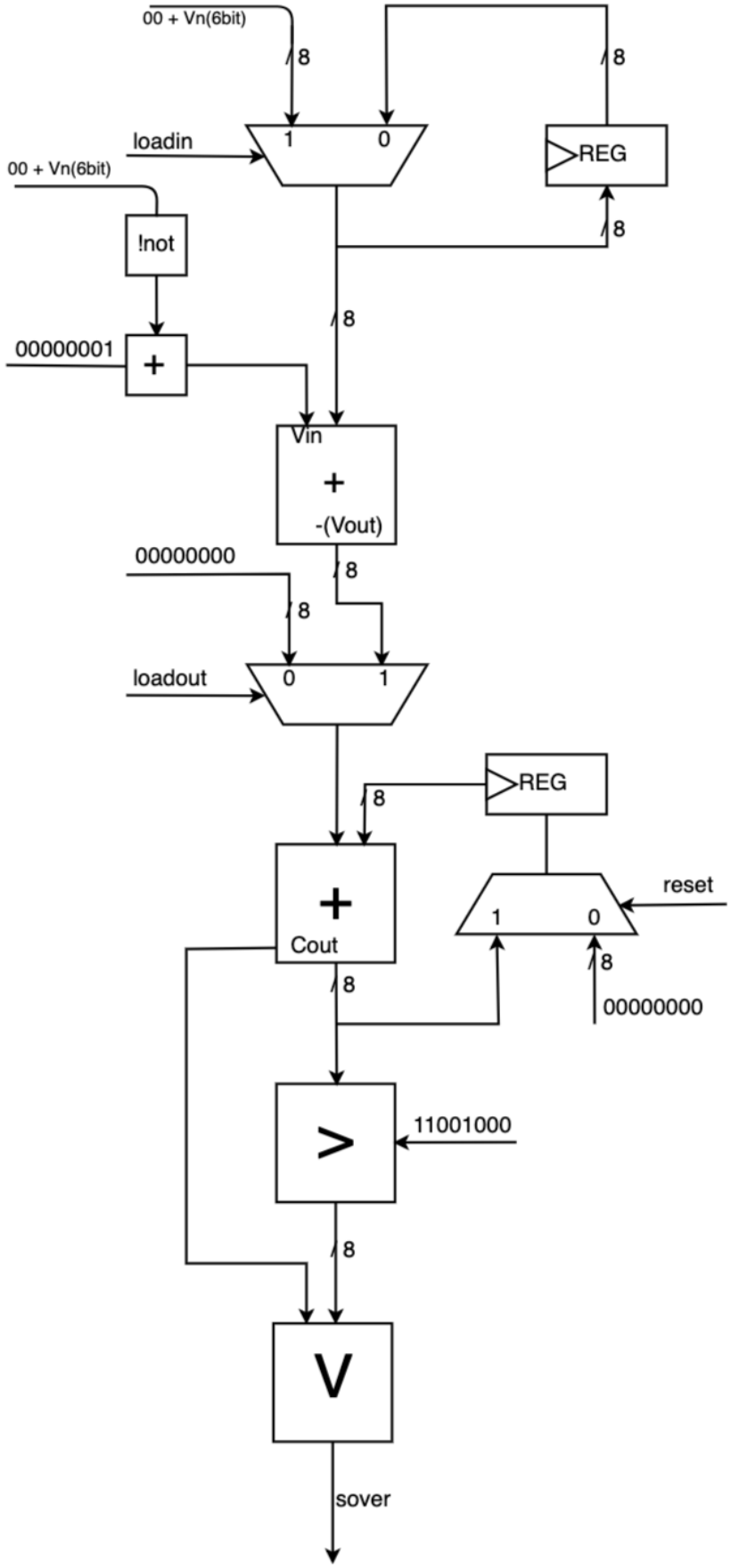
DESCRIZIONE

È la componente principale di tutto il sistema. È dedita al coordinamento della macchina (compito legato all'*fsm*) ed all'elaborazione dei dati (compito legato al *data-path*).

Il *data-path* è composto da:

1. *Contatori* a 4 bit: relativi ai Gate A, B, C, E, F. Sono circuiti operanti a 4bit dediti al conteggio dei pezzi passati dai vari gate;
2. *Componente di confronto* a 8 bit: circuito di confronto dei volumi di scarto a 8bit.
3. *Unita logica* OR: dedita alla gestione dei segnali di *overflow*.

COMPONENTE DI CONFRONTO A 8 BIT



La componente viene utilizzata per il confronto dei volumi di scarto *pre* e *post* lavorazione. I valori vengono ricavati dal *data-path*.

In caso di superamento della soglia massima del serbatoio, la componente manderà un segnale all'*fsm* che, a sua volta, produrrà un corrispondente segnale d'errore.

Il controllo del circuito viene effettuato dall'*fsm*, mediante i segnali *loadin*, *loadout* e *resgen*.

I primi due servono a selezionare i valori di *Vn* relativi ai volumi d'ingresso o d'uscita.

Il segnale *resgen*, invece, viene utilizzato per controllare i *registri*, quando viene posto a 0, essi vengono azzerati.

Il componente è formato da:

- 3 mux;
- 2 registri;
- 3 sommatori;
- 1 modulo "maggiore di";
- 1 not;
- 1 or.

Riceve come ***segnali di ingresso***:

- Reset;
- Loadin;
- Loadout.
- Vn: volumi in ingresso (si noti che a sinistra viene effettuato un *padding* per giungere a 8bit).

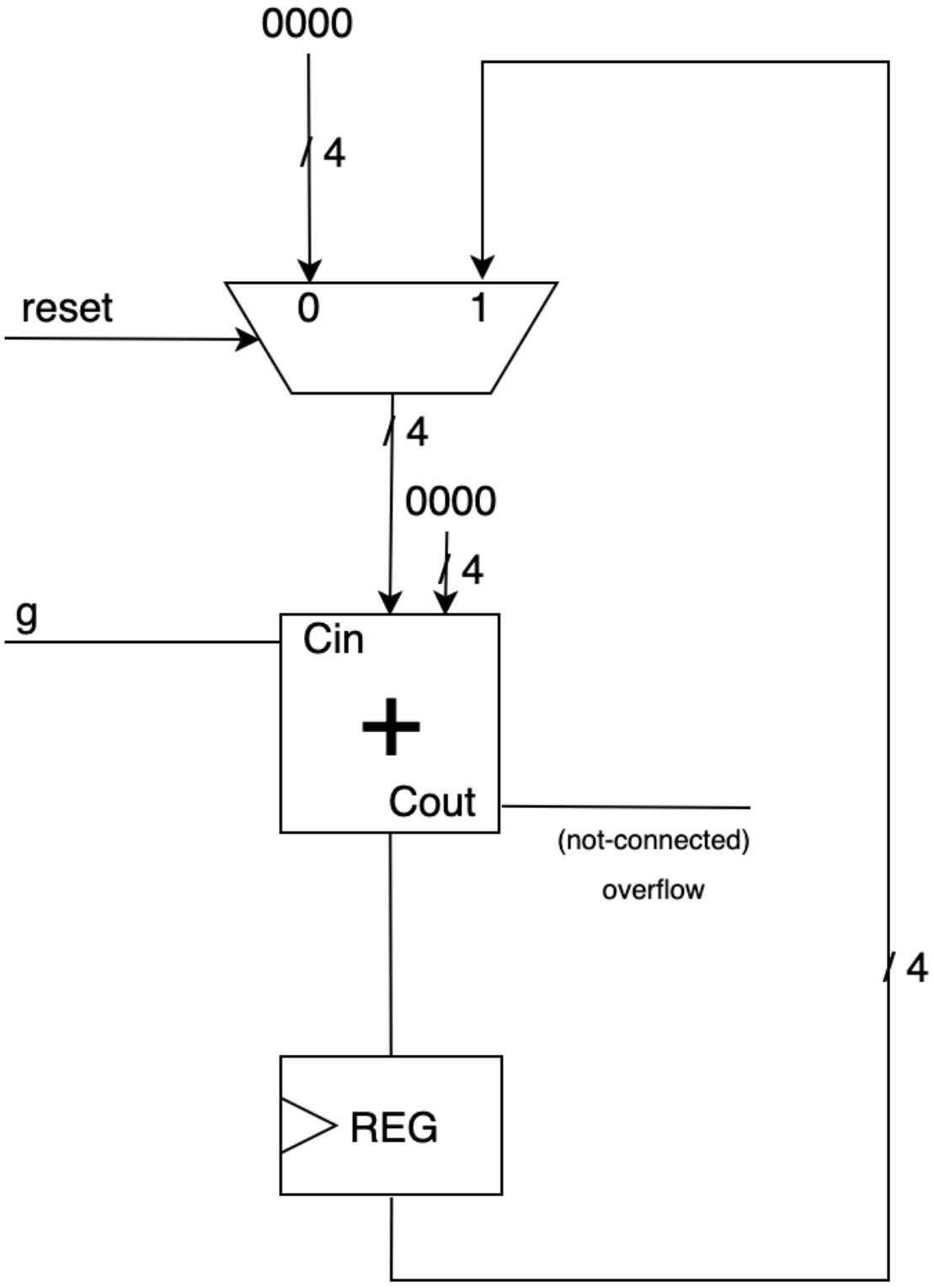
Espelle come ***segnali di uscita***:

- Sover.

Se il segnale in uscita *Sover* risulta 1 la macchina si fermerà generando l'errore 010.

COMPONENTE CONTATORE A 4 BIT

La componente viene utilizzata per il conteggio delle aperture dei gate, è resettata mediante il segnale *resgen* prodotto dall'*fsm*.



Il componente è formato da:

- 1 mux;
- 1 sommatore;
- 1 registro.

Riceve come segnali di ingresso:

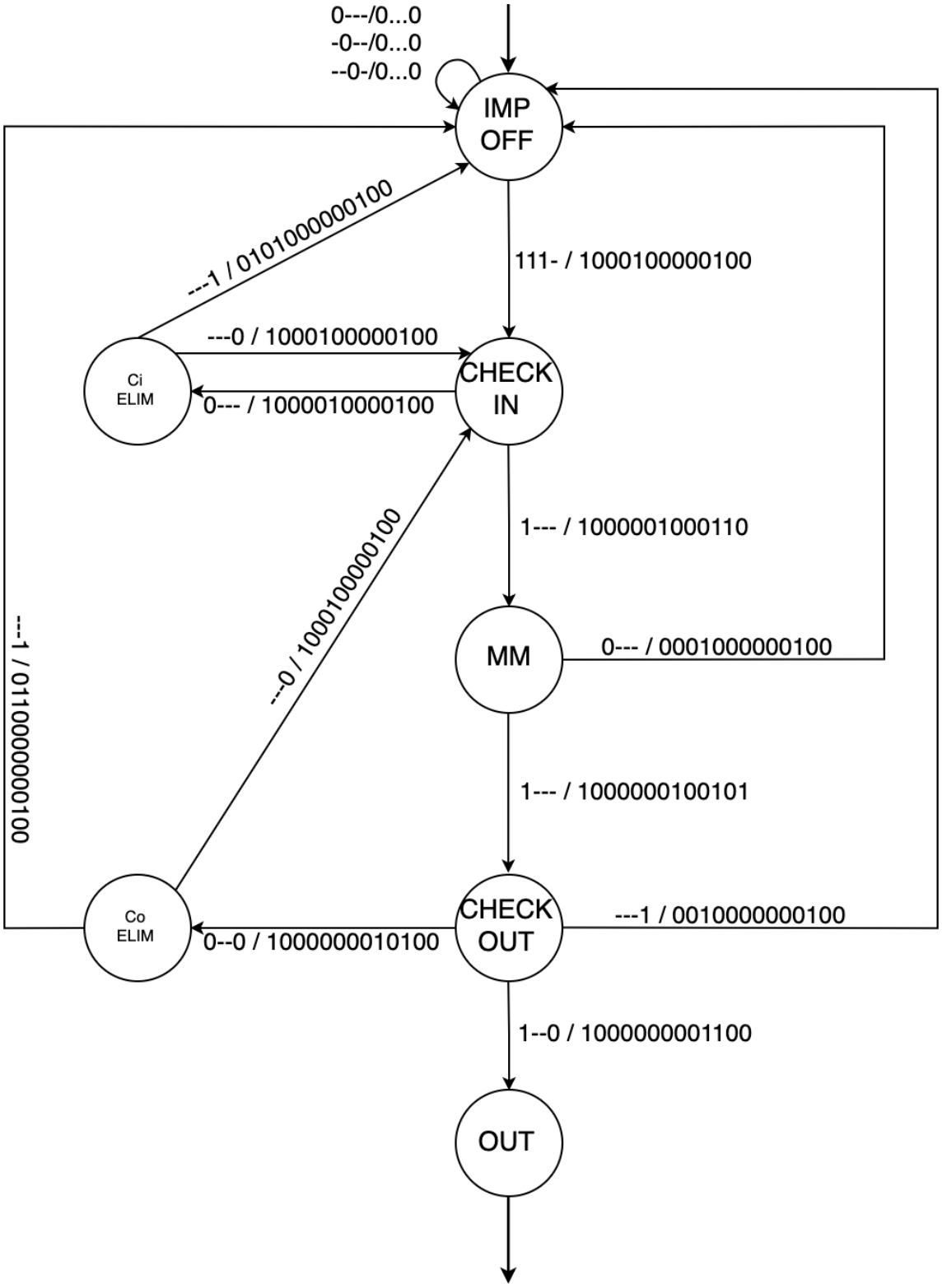
- Reset;
- G (segnale corrispondente a ogni clock a uno specifico gate, scelto dall'*fsm*).

Si specifica che l'overflow viene utilizzato solamente nel caso dei contatori legai al GATE B e al GATE E (segnati nella *fsmd* come CONT B WOF e CONT E WOF)(WOF=With *overflow*).

> Questi producono il segnale di *overflow* quando superano il valore massimo in modulo legato al contatore (ovvero 15).

Se un segnale di *overflow* viene generato, esso viene sommato agli altri segnali di *overflow* mediante un OR e poi il risultato viene condotto all'*fsm*.

FSM


0---/0...0
-0--/0...0
--0-/0...0
IMP
OFF
---1 / 0101000000100
111- / 1000100000100
---0 / 1000100000100
Ci
ELIM
0--- / 1000010000100
CHECK
IN
1--- / 1000001000110
--0 / 1000100000100
---1 / 0110000000100
MM
0--- / 0001000000100
1--- / 1000000100101
Co
ELIM
0--0 / 1000000010100
CHECK
OUT
---1 / 0010000000100
1--0 / 1000000001100
OUT

DESCRIZIONE

L'*fsm* è quella componente che si occupa di coordinare e controllare i risultati ottenuti dal *data-path*. Essa aprirà e chiuderà i GATE nella sequenza corretta, controllerà la capienza ottenuta nel serbatoio a ogni passaggio di scarti da parte della macchina e, in fine, nelle fasi di CHECK IN e CHECK OUT valuterà l'idoneità del pezzo.

Se necessario, inoltre, genera i segnali di errore nelle eventualità in cui il *data-path* segnali un *overflow* (i segnali di overflow NB NE e Sover sono collegati ad un OR).

In caso di ricezione di tali segnali l'*fsm* genera un errore opportuno in base allo stato in cui si trova.

STAMPA DEI RITARDI

Mediante il comando *printf_delay* sono stati selezionati i 30 ritardi legati ai 30 bit in uscita:

```
sis> print_delay
 ... using library delay model
                {oi}: arrival = (15.60 15.60)
                {err2}    : arrival = ( 6.60  6.60)
                {err1}    : arrival = ( 9.00  9.00)
                {err0}    : arrival = ( 7.60  7.60)
                {ga}      : arrival = (12.60 12.60)
                {gb}      : arrival = ( 9.00  9.00)
                {gc}      : arrival =(12.20 12.20)
                {gd}      : arrival =( 6.00  6.00)
                {ge}      : arrival = ( 8.60  8.60)
                {gf}      : arrival =( 9.00  9.00)
                {na3}     : arrival = (32.40 32.40)
                {na2}     : arrival = (29.80 29.80)
                {na1}     : arrival = (24.20 24.20)
                {na0}     : arrival = (25.60 25.60)
                {nb3}     : arrival = (24.20 24.20)
                {nb2}     : arrival = (26.80 26.80)
                {nb1}     : arrival = (24.00 24.00)
                {nb0}     : arrival = (24.20 24.20)
                {nc3}     : arrival = (29.80 29.80)
                {nc2}     : arrival = (27.20 27.20)
                {nc1}     : arrival = (24.20 24.20)
                {nc0}     : arrival = (24.20 24.20)
                {ne3}     : arrival = (24.20 24.20)
                {ne2}     : arrival = (26.80 26.80)
                {ne1}     : arrival = (24.00 24.00)
                {ne0}     : arrival = (24.20 24.20)
                {nf3}     : arrival = (29.80 29.80)
                {nf2}     : arrival = (27.20 27.20)
                {nf1}     : arrival = (24.20 24.20)
                {nf0}     : arrival = (24.20 24.20)
```

OTTIMIZZAZIONE

Lo script *script.rugged* è specifico per l'ottimizzazione. Esso utilizza diversi comandi, come: *sweep*, *eliminate*, *resub*, *fx*, *simplify* e *full_simplify*.

Originale

fsmd *pi* = 7 *po* = 30 *nodes* = 220 *latches* = 40 *lits*(sop) = 1074

Minimizzazione

fsmd *pi* = 7 *po* = 30 *nodes* = 118 *latches* = 40 *lits*(sop) = 506

Mapping n°1

fsmd *pi*= 7 *po* = 30 *nodes* = 304 *latches* = 40 *lits*(sop) = 699

Ottimizzazione

fsmd *pi* = 7 *po* = 30 *nodes*=123 *latches* = 40 *lits*(sop) = 504

Mapping n° 2

fsmd *pi* = 7 *po* = 30 *nodes* = 298 *latches* = 40 *lits*(sop) = 695

Si possono ricavare le seguenti differenze tra la macchina *non ottimizzata* e quella *ottimizzata*:

ORIGINALE

PI	PO	NODI	LATCH	LETTERALI
7	30	220	40	1074

MAPPING n°2

PI	PO	NODI	LATCH	LETTERALI
7	30	298	40	695

DIFFERENZE

PI	PO	NODI	LATCH	LETTERALI
/	/	+78	/	-379

Riducendo il numero di letterali di 379 si è ridotto il tempo di propagazione del segnale e si è, quindi, incrementata l'efficienza.