PROVA FINALE
(PROGETTO DI RETI LOGICHE)

PROGETTO A CURA DI

Riccardo Bassi	10521394	870338
Francesco Battagin	10528227	866153

PROFESSORE

Prof. William Fornaciari

TUTOR

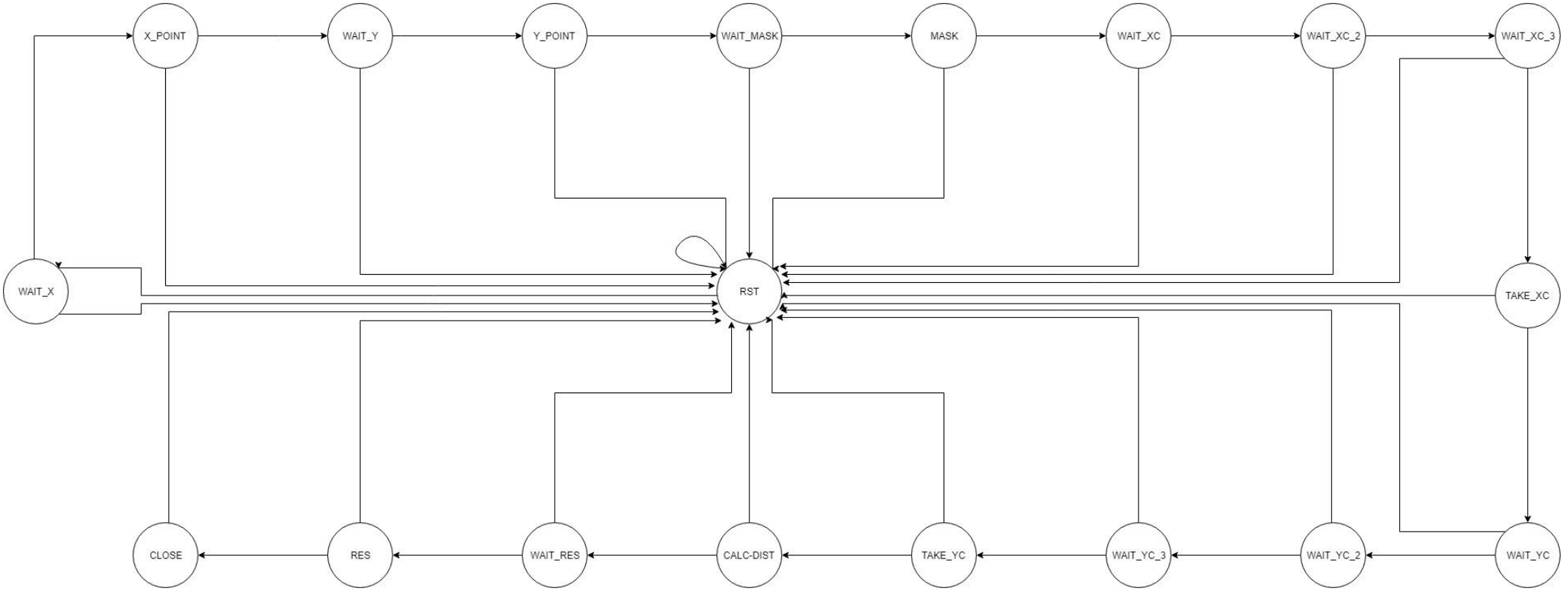
Prof. Davide Zoni

Anno accademico: 2018/19

INDICE

1. Macchina a Stati Finiti (FSM)

1.1 Schema della macchina



1.2 Descrizione del funzionamento

L'automa, descritto graficamente in precedenza, consiste in una Macchina a Stati Finiti, il cui funzionamento è il seguente.

Tale FSM si basa sul settaggio di alcuni signals (xp, yp, xc, yc, beg_mask e Y_N, std_logic_vector di 8 bit ciascuno, e di Y_N_1, un intero) e variables (data, address, dist, dist_min, std_logic_vector di 8 bit, e xp_op, yp_op, xc_op e yc_op, std_logic_vector di 9 bit, considerando possibili overflow).

Lo stato iniziale corrisponde a uno stato di reset (stato RST): vengono inizializzati tutti i signals e le variables, in modo tale che, a meno di segnali di reset e con il segnale di start pari a '1', lo stato successivo della macchina sia d'attesa (WAIT_X) e, successivamente, si possa acquisire in ingresso l'ascissa del punto (salvata in xp, stato X_POINT). Tale procedimento (stato d'attesa e stato d'acquisizione) è ripetuto per salvare l'ordinata del punto (in yp, stati WAIT_Y e Y_POINT) e la maschera iniziale (in beg_mask, stati WAIT_MASK e MASK).
Vengono inoltre settati xp_op e yp_op al valore di xp e yp (con il MSB settato a 0), usati in seguito per poter effettuare le operazioni di sottrazione e addizione senza preoccuparsi di possibili overflow.

Successivamente si susseguono tre stati di wait (WAIT_XC, WAIT_XC_2 e WAIT_XC_3), affinché venga salvata correttamente l'ascissa del primo punto da considerare.

Nel momento in cui viene acquisita l'ascissa del primo punto (stato TAKE_XC) viene eseguito un controllo sul bit della maschera in ingresso relativo a quel centroide, tramite Y_N_1 (un intero che funge da "contatore", tenendo traccia di quale bit si stia considerando sulla maschera); nel caso sia pari a '0', viene incrementato l'indirizzo successivo di due affinché si possa acquisire l'ascissa del punto successivo, così come Y_N_1, e viene aggiornato Y_N (shift a sx, un vector inizializzato a "00000001", utilizzato per costruire passo passo la maschera d'uscita); altrimenti, nel caso in cui sia pari a '1', la variabile Y_N_1 viene incrementata di uno, allo stesso modo dell'indirizzo successivo, cosicché si possa acquisire l'ordinata del punto considerato (in TAKE_YC, a seguito di altri tre stati di wait, WAIT_YC, WAIT_YC_2 e WAIT_YC_3).
Vengono inoltre settati xc_op e yc_op al valore di xc e yc (con il MSB settato a 0).

A questo punto, viene calcolata la distanza di Manhattan del centroide considerato dal punto in esame (stato CALC_DIST); se tale distanza è la minima calcolata fino a questo momento, viene aggiornata la variabile dist_min (inizializzata a '11111111') e viene salvato in data il valore di Y_N (in questo modo, in quest'ultima variabile, inizialmente settata a '00000000', viene aggiornata la maschera di uscita); altrimenti, se è uguale alla precedente dist_min, viene operata una OR tra Y_N e data, in modo tale da settare a '1' il bit del centroide.
Nel caso in cui fosse maggiore, non viene effettuata nessuna operazione.

A seguito di tali operazioni si effettuano ulteriori check: se il centroide preso in esame non è l'ultimo, si ritorna al primo stato di wait dell'ascissa del centroide successivo (WAIT_XC), aggiornando variabili e signals come descritto precedentemente; altrimenti si passa in uno stato di wait (WAIT_RES) e successivamente viene settato o_done a '1' (stato RES).

Infine, viene ri-settato o_done a '0' (stato CLOSE) per concludere il test.

Ogni stato ha un check sul segnale di reset, il quale, se acquisito pari a '1', porta la macchina allo stato di RST, ri-settando tutte le variabili e i signals come precedentemente descritto.

2. Casi di test

2.2 Test effettuati

Si è scelto di programmare uno script (in allegato) atto a generare numerosi casi di test, completamente casuali, molti dei quali sono stati testati ottenendo risultati positivi in behavioural simulation, in post-synthesis simulation e anche in timing simulation.

In aggiunta, sono stati oggetto di test anche i seguenti "casi particolari":

- Maschera iniziale pari a 0: ci si aspetta che la macchina non calcoli alcuna distanza tra il punto e i centroidi, riportando una maschera di uscita settata completamente a '00000000'
- Maschera iniziale pari a 127: l'ultimo centroide della maschera è settato a 0, ci si aspetta che la macchina computi tutti le distanze di tutti i centroidi eccetto l'ultimo
- Maschera iniziale qualsiasi, con centroidi da considerare posti alla stessa distanza
- Maschera iniziale pari a 255: la macchina è composta da soli '1', ci si aspetta che la maschera calcoli le distanze di tutti i centroidi dal punto in esame
    - Tutti i centroidi alla stessa distanza: ci si aspetta un solo aggiornamento della distanza minima e una maschera di uscita composta da 8 bit settati a '1'
    - Tutti i centroidi a distanza differente: si è scelto di considerare anche questo scenario come una sorta di "stress test".
- Segnali di reset nel mezzo della computazione: ci si aspetta che la macchina venga resettata e ricominci la computazione del test in esame
- Calcolo di una distanza maggiore di 255: viene sfruttato il vector di 8 bit per evitare overflow

2.2 Conclusioni

In conclusione, la macchina è stata testata con i precedenti casi di test affinché fossero presi in esame diversi moduli.

In primo luogo, sono stati eseguiti numerosi test_benches di base, con numeri casuali, per verificare che il funzionamento della FSM fosse effettivamente corretto.

In seguito, l'analisi è stata focalizzata principalmente sul comportamento in caso di maschere iniziali "particolari", in modo tale da verificare che tutti i signals e variables venissero settati correttamente anche all'interno del processo; ad esempio, è risultato molto utile il test con una maschera pari a 127, il quale ha rivelato un errore non risultato dai test di base; in aggiunta, anche i test con centroidi posti alla stessa distanza dal punto hanno dimostrato come la macchina non vada a sovrascrivere inutilmente variabili, limitandosi all'aggiunta del centroide in esame nella maschera di uscita. Il test con maschera a 255 e centroidi posti a distanze differenti è risultato efficace affinché si potesse constatare il corretto funzionamento della macchina in situazioni di "carico massimo" (si relaziona in modo corretto con i segnali di ingresso della RAM grazie agli stati di wait).

Infine, i test che prevedono dei segnali di reset nel mezzo dell'esecuzione, hanno riportato come la macchina risponda in modo corretto a tale sollecitazione, resettando segnali e variabili, riportandosi nello stato RST e ricominciando la computazione del test in esame.

3. Possibili ottimizzazioni

Si è scelto di inserire uno stato di wait ogni volta che ci si aspetta di acquisire dati dalla RAM, per adeguare la frequenza di funzionamento della macchina ai tempi di risposta della memoria.

Tale soluzione risulta tuttavia “impegnativa” nel momento in cui si acquisiscono le coordinate dei centroidi: si è stati costretti a inserire tre stati di wait per fare in modo di acquisire tali valori in modo corretto, soprattutto per gestire il caso in cui non si debba considerare un determinato centroide e si debba passare a quello successivo (il valore dell’ascissa viene passato comunque alla macchina, con il rischio di generare “conflitto”, riscontrato in alcuni casi di test).

Una migliore gestione delle acquisizioni unitamente a una RAM dai tempi di risposta più rapidi avrebbe sicuramente comportato una diminuzione degli stati, e quindi di celle.

In aggiunta si sarebbe potuto far uso di variabili da 9 bit con le quali effettuare operazioni aritmetico-logiche solamente quando necessario (ciò avrebbe tuttavia comportato dei check aggiuntivi).