Prova Finale
(Progetto di Reti Logiche)

Vittorio Antonicelli

Cod. Persona **10492832**

Matricola **846674**

Corso di Reti Logiche
Anno accademico 2018-2019
Professore: William Fornaciari
Tutor: Davide Zoni

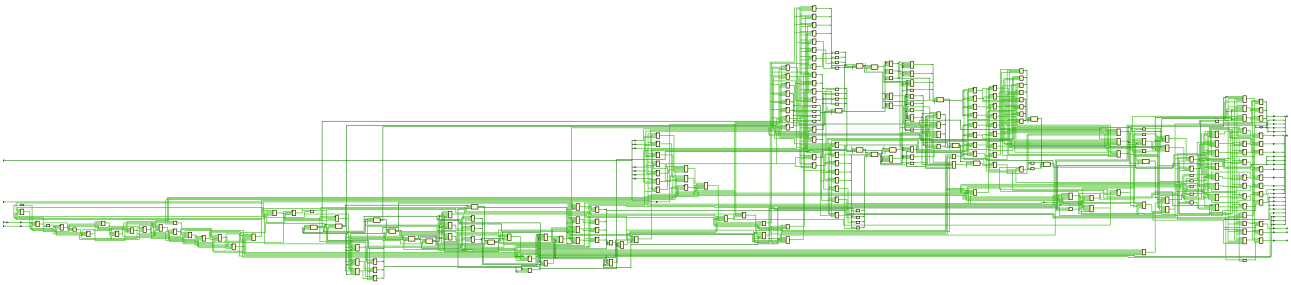
INDICE

Scelte progettuali e descrizione del componente

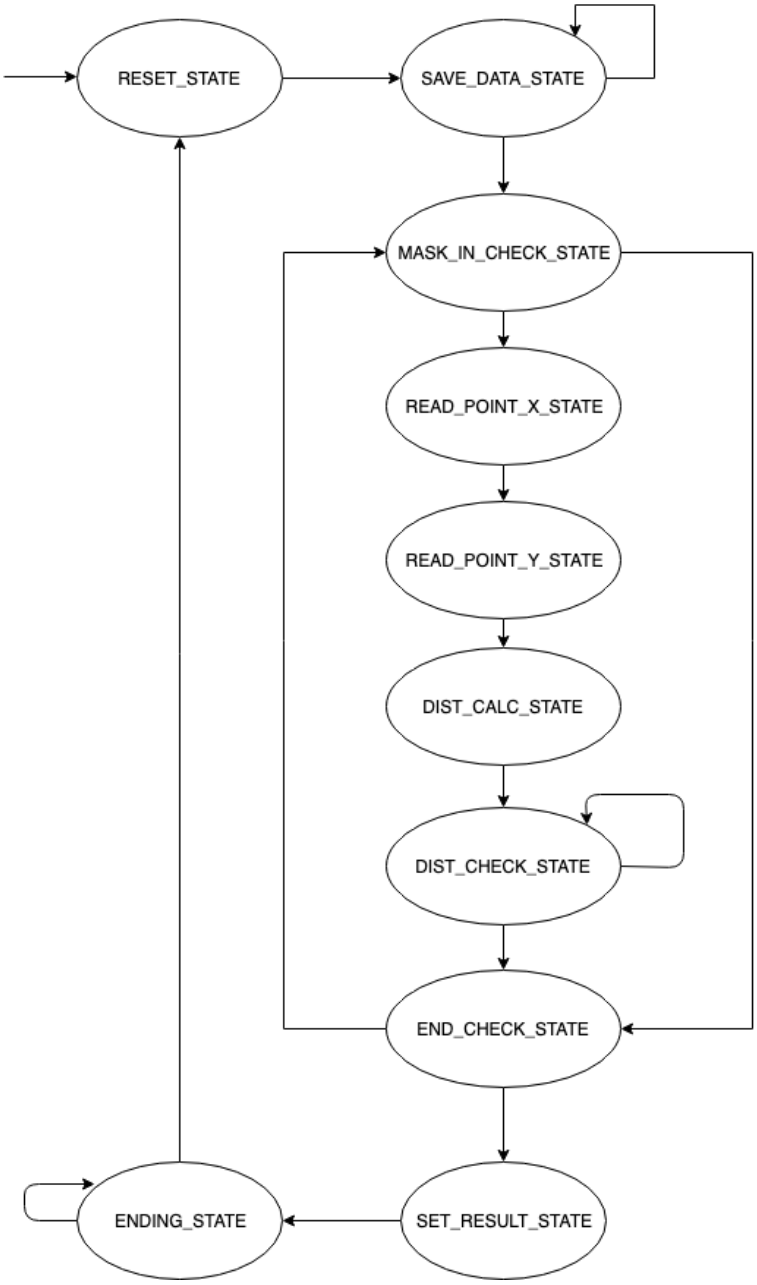
Per implementare il componente richiesto secondo specifiche di progetto, si è ritenuto opportuno realizzare un componente monoprocesso sensibile al clock che implementa una macchina a stati finiti, optando quindi per un approccio descrittivo puramente behavioral (comportamentale), basato cioè su descrizioni algoritmiche del comportamento del circuito. In particolare si è scelto di lavorare sul fronte di discesa del clock, in corrispondenza del quale viene commutato lo stato e viene eseguita la corrispondente porzione di codice che esegue uno step intermedio della computazione.
Un segnale di tipo enumerato indica lo stato, mentre per i valori utili all'elaborazione (variabili e costanti) vengono usati segnali di tipo *std_logic_vector*. E' stata fatta particolare attenzione ai possibili overflow derivanti dal calcolo della distanza di Manhattan, così definita:

$$L_1(P_1, P_2) = |x_1 - x_2| + |y_1 - y_2|$$

Essendo lo spazio considerato nella specifica di 256 x 256 punti, i punti possono essere codificati con coppie di segnali std_logic_vector da 8 bit.
Per la distanza invece sono necessari almeno 9 bit. Infatti, supponendo ad esempio P1 = (0, 0) e P2 = (255, 255), risulterebbe che L(P1, P2) = | 0 - 255 | + | 0 - 255 | = 255 + 255 = 510
Un risultato codificabile con 9 bit di un *std_logic_vector* in modo unsigned, oppure con 10 bit di un *std_logic_vector* in modo signed.
Un segnale di particolare importanza è il contatore (*point_counter*) che fornisce l'indicazione di quanti centroidi sono già stati calcolati nel ciclo di calcolo. E' uno *standard_logic_vector* contenente la codifica di un numero tra 0 e 7 (per identificare gli 8 centroidi da calcolare).



Schema degli stati



NOTA: ad ogni ciclo di clock viene controllato se il segnale di reset (*i_rst*) è alto. In caso positivo, a partire da qualsiasi stato in cui si trovi la macchina lo stato viene commutato in *reset_state*. E' quindi implicita la presenza di un arco che esca da ogni stato e vada nel suddetto stato di reset.

Descrizione degli stati

Il fulcro dell'elaborazione è individuabile nel ciclo *mask_in_check -> read_point -> dist_calc -> dist_check -> end_check* che, per ogni centroide, verifica la validità tramite check della maschera di ingresso, calcola la distanza dal punto da valutare, la confronta con la distanza minima trovata aggiornando di conseguenza la maschera di uscita, verifica la fine del ciclo.

- RESET_STATE: stato di reset in cui vengono azzerati gli output (*o_en, o_we, o_data, o_done*) e vengono inizializzati tutti i segnali utili alla computazione. In particolare il segnale della distanza minima trovata viene settato al massimo teoricamente possibile +1, cioè 511 (codificato come "111111111") e viene azzerato un contatore (*point_counter*) che fornirà l'indicazione di quanti centroidi sono già stati calcolati nel ciclo di calcolo. Se il segnale di start (*i_start*) è alto, inizia il processo partendo dallo stato *save_data_state* in cui si provvederà ad acquisire dalla memoria la maschera di ingresso e le coordinate x e y del punto da valutare.

- SAVE_DATA_STATE: in 3 cicli di clock vengono letti dalla memoria RAM rispettivamente la maschera di ingresso, la coordinata x e la coordinata y del punto da valutare. Una volta acquisiti, si passa al ciclo di calcolo dei centroidi, che inizia con il check della maschera di ingresso nello stato *mask_in_check_state*.

- MASK_IN_CHECK_STATE: viene verificata la validità del centroide attuale (identificato dal contatore point_counter). Se il centroide è valido, si passa alla lettura dello stesso dalla memoria (read_point_x_state), altrimenti si passa direttamente alla verifica di fine del ciclo (end_check_state).

- READ_POINT_X_STATE: viene acquisita la coordinata X del centroide attuale (identificato dal contatore *point_counter*).

- READ_POINT_Y_STATE: viene acquisita la coordinata Y del centroide attuale (identificato dal contatore *point_counter*).

- DIST_CALC_STATE: viene calcolata la distanza di Manhattan del centroide attuale dal punto da valutare.

- DIST_CHECK_STATE: se la distanza appena calcolata è minore della distanza minima trovata, resetta la maschera di uscita, imposta questo valore come nuova distanza minima trovata, e rimane in questo stato per il prossimo ciclo di clock. Se la distanza calcolata è uguale alla distanza minima trovata, aggiorna la maschera di uscita aggiungendo un '1' per il centroide attuale e passa allo stato di verifica di fine ciclo (*end_check_state*). Se la distanza calcolata è maggiore della distanza minima trovata, non aggiorna la maschera di uscita (che è già inizializzata a zero per il centroide attuale) e passa allo stato di verifica di fine ciclo (*end_check_state*). La durata di questo stato è quindi di due cicli di clock nel caso di centroidi la cui distanza è minore delle distanze già trovate, di un clock negli altri casi.

- END_CHECK_STATE: viene verificato il valore del contatore. Se sono già stati computati tutti i centroidi, passa allo stato di scrittura dell'output (*set_result_state*), altrimenti incrementa il contatore e passa nuovamente allo stato di check della maschera di ingresso (*mask_in_check_state*) per iniziare un nuovo ciclo in cui si valuterà il prossimo centroide.

- SET_RESULT_STATE: abilita la memoria in scrittura e in modo da poter memorizzare l'output finale della maschera di uscita.

- ENDING_STATE: viene alzato il segnale di done che viene tenuto alto finchè il segnale di start non viene abbassato. Quando quest'ultima condizione accade, il segnale di done viene riportato basso e si passa ad uno stato di reset per poter eventualmente riavviare la computazione non appena un segnale di start viene nuovamente alzato.

Ottimizzazioni

Nell'algoritmo scelto l'ottimizzazione è evidente quando si è in presenza di una maschera di ingresso contenente degli zeri, in quanto la verifica della maschera di ingresso avviene ancor prima della lettura dei centroidi dalla memoria.

Test

Oltre al test bench fornito, al quale il componente risponde correttamente sia in presintesi che in postsintesi, sono stati provati anche i seguenti test:

- Maschera di ingresso tutta nulla.
- Maschera di ingresso completamente valida.
- Punto da valutare molto distante dal centroide, in modo da verificare l'assenza di overflow nel calcolo della distanza di Manhattan.
- Tutti i centroidi e il punto da valutare in (0, 0).
- Vari altri test generati randomicamente.

A questo link (https://github.com/vintonic/ProgettoRL2019) è possibile vedere i testbench usati.