Prova Finale
Progetto di Reti Logiche

Anno Accademico 2018-2019

Mattia Massarini

Scuola di Ingegneria Industriale e dell'Informazione

Politecnico di Milano

Laurea di primo livello

Corso di Ingegneria Informatica – II3

Matricola 825664

Codice Persona 10496090

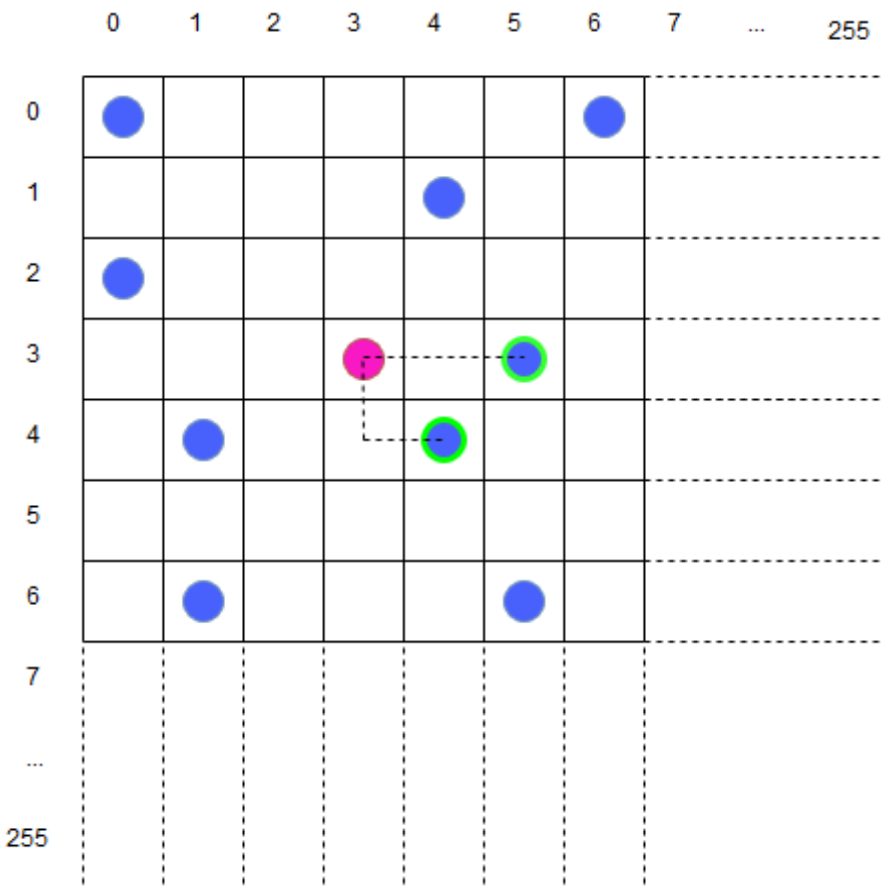
Docente: Fabio Salice

Indice

1. Introduzione

Lo scopo del progetto è la realizzazione di un componente Hardware mediante il software Vivado utilizzando il linguaggio VHDL.

Data una matrice bidimensionale 256x256 e otto punti appartenenti a tale spazio detti centroidi, il componente dovrà trovare tutti i punti la cui distanza sia la più breve rispetto ad un nono punto d'interesse.
Una maschera di ingresso formata 8 bit determina quali degli otto centroidi dovranno essere considerati al fine del calcolo: se l'i-esimo bit è a 1 verrà valutato, se è a 0 invece no.
Il bit più significativo rappresenta l'ottavo centroide mentre il meno significativo il primo centroide.



In figura è presentata una porzione della matrice di dimensione 256x256.
Il centroide in rosso è il punto da valutare, in blu sono quelli su cui calcolare la distanza, mentre quelli circondati in verde sono i centroidi a distanza minima trovati (due in questo caso).

Il risultato della computazione consiste in una maschera di uscita di 8 bit che rappresenta l'insieme dei centroidi a distanza minima dal punto di interesse.
L'i-esimo bit è posto a 1 se esso si trova a distanza minima, 0 altrimenti.
Se esistono più centroidi a distanza minima, essi verranno tutti considerati.
Per "distanza" si intende la Distanza di Manhattan, definita nel modo seguente:

Siano x_0, y_0 le coordinate del punto da valutare e x , y le coordinate di un generico centroide, la distanza di Manhattan è:

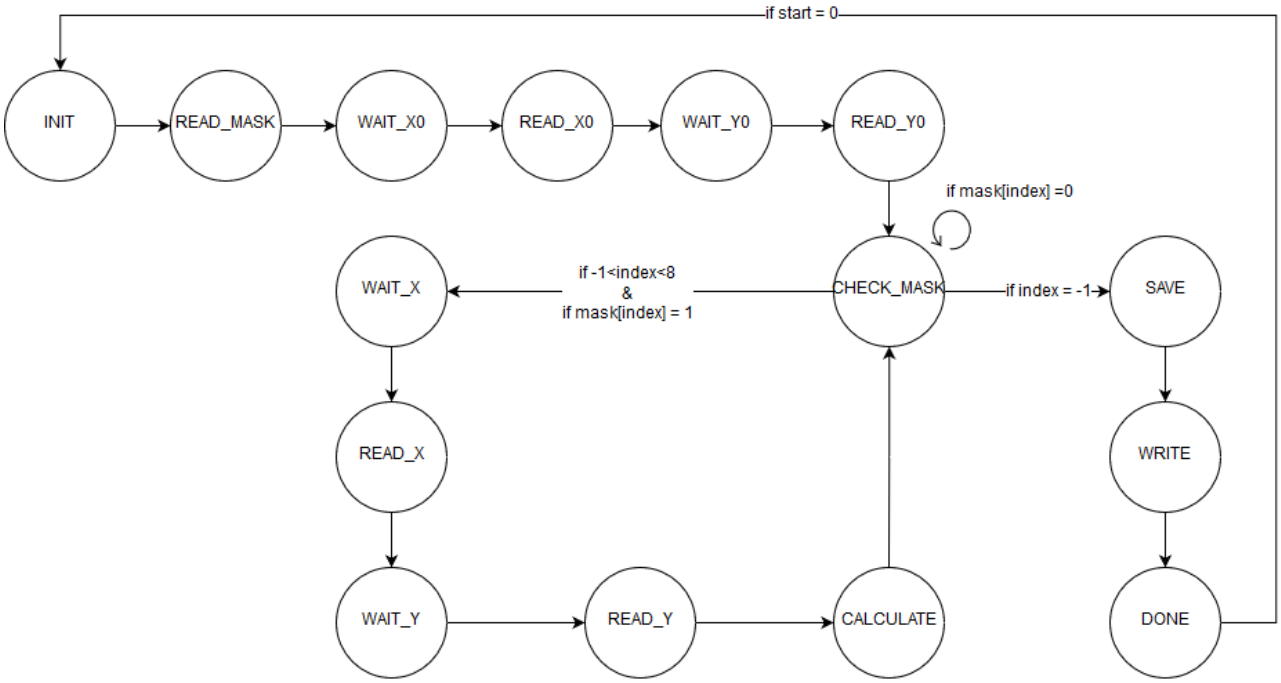
$$D = |x - x_0| + |y - y_0|$$

Una memoria esterna con indirizzamento al byte, non facente parte del progetto, verrà utilizzata in lettura per acquisire le informazioni della maschera di input e delle coordinate dei centroidi, in scrittura invece per salvare la maschera di output.

2. Scelte progettuali

La progettazione del componente è iniziata con la stesura di un algoritmo in pseudocodice il cui scopo è quello di rappresentare l'insieme di operazioni da eseguire per generare il risultato corretto.
Successivamente si è passati alla realizzazione di una macchina a stati finiti composta inizialmente da 19 stati, ridotti a 15 in fase di ottimizzazione.
Infine si è passati alla descrizione del componente in linguaggio VHDL.

Il componente è stato descritto mediante l'utilizzo di un singolo process sincronizzato sul fronte di salita del clock, la cui sensitivity list comprende i segnali *i_clk* e *i_rst*.



Inizialmente la macchina rimane ferma in attesa del segnale di reset *i_rst*, che consente di inizializzare segnali e variabili, successivamente si passa allo stato **init**.

Si attende il segnale di start e si passa allo stato **read_mask**.

Vengono lette dalla memoria la maschera di input e le coordinate del punto da valutare x0 e y0 attraversando gli stati **read_x0** e **read_y0**, fino ad arrivare allo stato **check_mask**.

In questo stato, se il valore di index è compreso tra 0 e 7 (estremi compresi) e nel caso in cui il bit corrispondente della maschera di ingresso sia uguale a 1, percorrendo gli stati **read_x** e **read_y** vengono lette dalla memoria e salvate nelle variabili *int_x* e *int_y* le coordinate del centroide, l'indice è decrementato.

Nello stato **calculate** viene salvata la distanza di Manhattan in un array e nel caso in cui la distanza appena salvata sia più piccola di *min*, quest'ultima viene sovrascritta.

Questo ciclo è percorso fino ad aver letto tutta la maschera (index = -1), passando allo stato di **save**, in cui l'array viene scandito confrontando il valore della i-esima cella con *min*. Se entrambi sono uguali il bit i-esimo viene posto a 1, altrimenti 0.

Nello stato di **write** viene salvata la maschera di uscita nella memoria e il segnale *i_done* è portato ad 1.

Si rimane nello stato di **done** fino a quando il segnale i_start non è stato portato a 0.

Infine il segnale *o_done* è portato a 0 e la macchina viene riportata nello stato di *init*, pronta ad una nuova computazione.

I segnali di attesa (**wait_x0**, **wait_y0**, **wait_x**, **wait_y**) sono utilizzati per dare il tempo ai segnali di propagarsi correttamente, evitando letture da memoria errate.

Se il segnale *i_rst* dovesse essere portato alto durante l'esecuzione, il componente viene resettato e portato nello stato **init**.

3. Testing

Il componente è stato sottoposto a diversi test per verificarne il corretto funzionamento, scegliendo opportunamente la configurazione dei centroidi.

I casi di test sono i seguenti:

- Test in cui tutti i centroidi sono stati disattivati (maschera di ingresso a 0).
- Test in cui tutti i centroidi sono attivi e hanno le stesse coordinate
- Test in cui tutti i centroidi condividono la stessa distanza, ma diverse coordinate
- Test in cui tutti i centroidi sono attivi e coincidenti
- Test in cui tutti i centroidi sono attivi e hanno diversa distanza.

4. Conclusione

A seguito del testing il componente funziona secondo le specifiche, superando positivamente le simulazioni proposte dal software Vivado: Behavioral Simulation, Post-Synthesis Simulation e Post-Implementation Simulation (delle ultime due sia functional che timing).