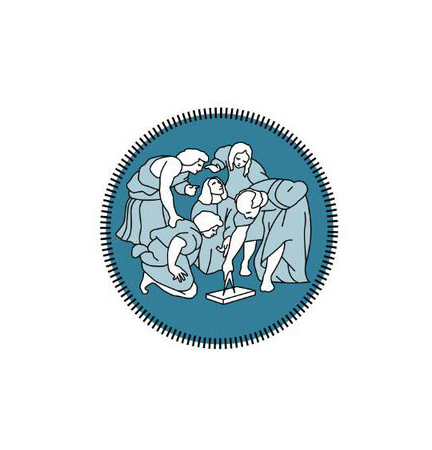
Dipartimento di Elettronica Informazione e Bioingegneria

Corso di Laurea in Ingegneria Informatica

Politecnico di Milano



Prova Finale

(Progetto di Reti Logiche)

Professore di riferimento **William Fornaciari**

Tutor di riferimento **Davide Zoni**

**Edoardo Carrer** Codice Persona 10561353 Matricola 870718

**Amedeo Cavallo** Codice Persona 10562259 Matricola 868665

|  |  |
| --- | --- |
| Politecnico di Milano | 2019 |

Contenuti

[Contenuti ii](#_Toc3366437)

[Introduzione 1](#_Toc3366438)

[Scelte progettuali 2](#_Toc3366439)

[1.1 Stato RESET 3](#_Toc3366440)

[1.2 Stato BITMASK 3](#_Toc3366441)

[1.3 Stato Y 3](#_Toc3366442)

[1.4 Stato X 4](#_Toc3366443)

[1.5 Stato DONE 4](#_Toc3366444)

[Testing 5](#_Toc3366445)

[Risultati della sintesi 6](#_Toc3366446)

[Ottimizzazioni 7](#_Toc3366447)

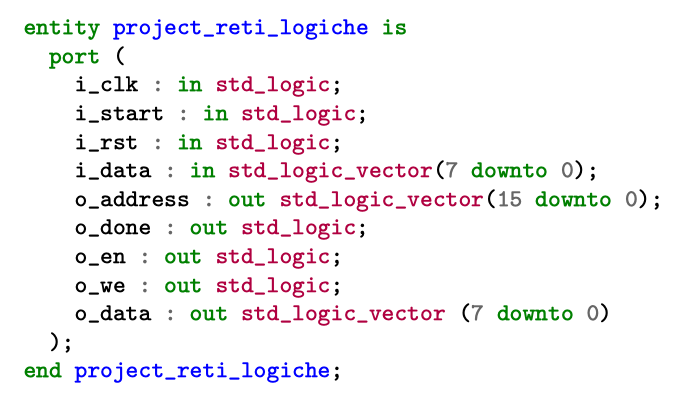
Introduzione

Lo scopo del progetto è la realizzazione di un componente hardware in VHDL. Esso riceve in ingresso le coordinate di otto centroidi e di un punto da valutare. Dopo aver calcolato la “Manhattan Distance” di ogni centroide da tale punto è in grado di individuare quale/i tra questi ha distanza minore.

Il componente si interfaccia con un chip RAM nel quale sono stati precaricati i dati da analizzare:

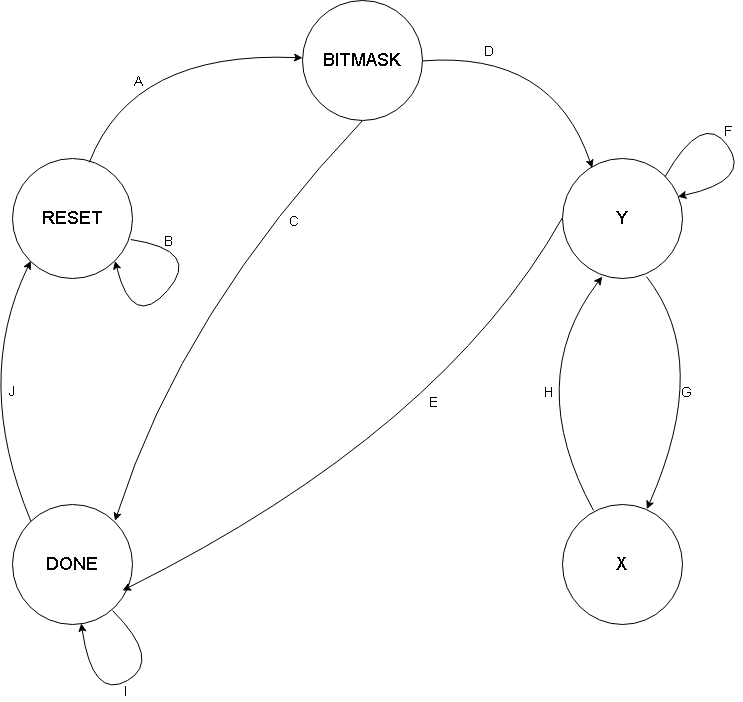
* All’indirizzo 0b0 (0) della RAM è memorizzata una bitmask di 8 bit che indica i centroidi attivi e dunque da valutare.
* Negli indirizzi di memoria successivi sono presenti alternativamente le coordinate X e Y dei centroidi.
* All’indirizzo 0b10001 (17) e 0b10010 (18) della RAM sono memorizzati rispettivamente la coordinata X e la coordinata Y del punto da valutare.
* Secondo le speciﬁche, il testbench richiede che il risultato della computazione sia salvato nella memoria RAM all’indirizzo 0b10011 (19).

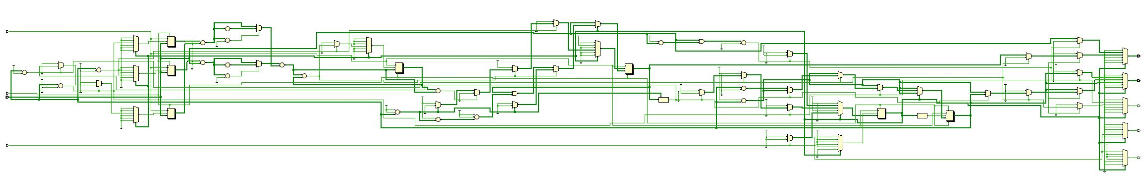
L’interfaccia del componente, così come presentata nelle speciﬁche, è la seguente:



Scelte progettuali

Data la presenza di un segnale di start e di uno di reset, è ragionevole pensare che sia necessario un automa a stati ﬁniti (FSM). In particolare, per modellare il design, abbiamo utilizzato una macchina di Mealy a 5 stati. Di seguito una breve descrizione degli stati con le relative transizioni.



****

## Stato RESET

Lo stato iniziale è RESET, nel quale la macchina attende il segnale di i\_start proveniente dal testbench. Ogni altro stato conﬂuisce ovviamente in RESET ogni qualvolta i\_rst è posto alto.

In questo stato tutti i segnali vengono portati al valore di default.

## Stato BITMASK

In questo stato viene letta dalla RAM la bitmask dei centroidi attivi da analizzare.

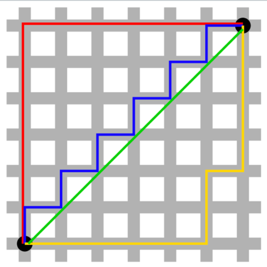
Nel caso in cui non ci siano punti da analizzare o ci sia un solo punto attivo la bitmask letta viene propagata direttamente in uscita per essere scritta in memoria in quanto nessuna modifica è necessaria.

## Stato Y

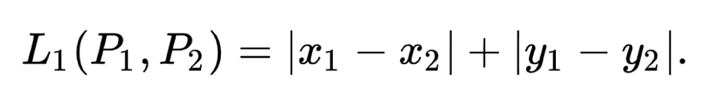
Stato in cui si legge la coordinata Y del punto da valutare e dei centroidi e a computazione terminata viene scritto il risultato finale nella RAM.

L’indirizzo per la lettura nella RAM viene via via decrementato ad ogni ciclo di clock, poiché è possibile accedere alla memoria RAM sequenzialmente. Nel caso in cui il centroide non sia attivo l’indirizzo viene decrementato due volte in modo da non effettuare letture superflue.

## Stato X

Stato in cui si legge la coordinata X del punto e dei centroidi e nel secondo caso in cui si calcola la distanza tra il punto da valutare e il centroide in ingresso.

Nel caso in cui la distanza appena calcolata sia minore della distanza minima quest’ultima viene aggiornata.

Manhattan distance (la distanza tra due punti è la somma del valore assoluto delle differenze delle loro coordinate):

## Stato DONE

Durante l’ultimo ciclo di clock la computazione è ormai terminata e viene posto alto il segnale in uscita o\_done, la memoria RAM viene posta in IDLE abbassando il segnale o\_en. Lo stato successivo è RESET che predispone il componente ad una nuova elaborazione.

Testing

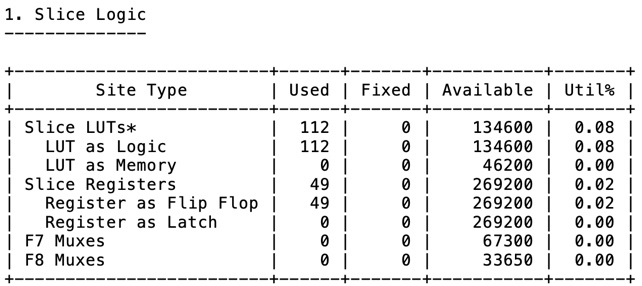
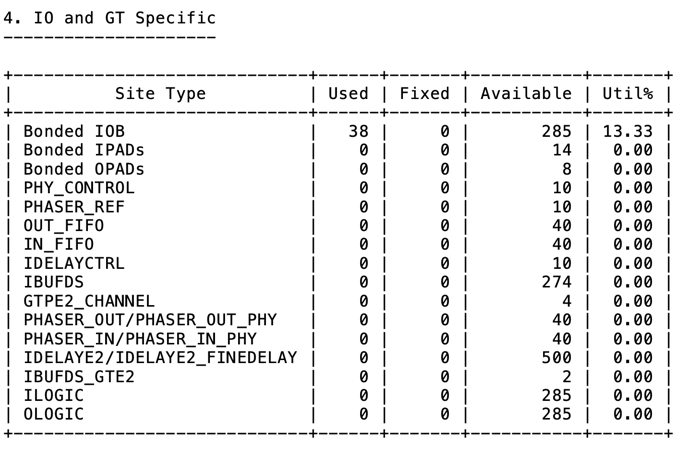
Il componente supera correttamente la simulazione Behavioral, la simulazione Post-Syntesis Functional e la simulazione Post-Syntesis Timing. Ulteriori test sono stati eﬀettuati nel corso della progettazione del componente anche in Post-Implementation

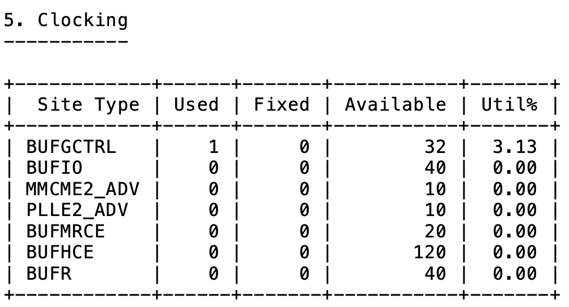
Abbiamo sviluppato un programma in linguaggio Python in grado di effettuare in Behavioral, Post-Syntesis Functional e Post-Syntesis Timing simulazioni con valori casuali. Nel caso di errori durante la sintesi vengono mostrati i log e a fine computazione viene analizzata la percentuale di test passati rispetto al numero test richiesto. Questo ci ha permesso di fare un elevato numero di test e di verificare il corretto funzionamento del componente.

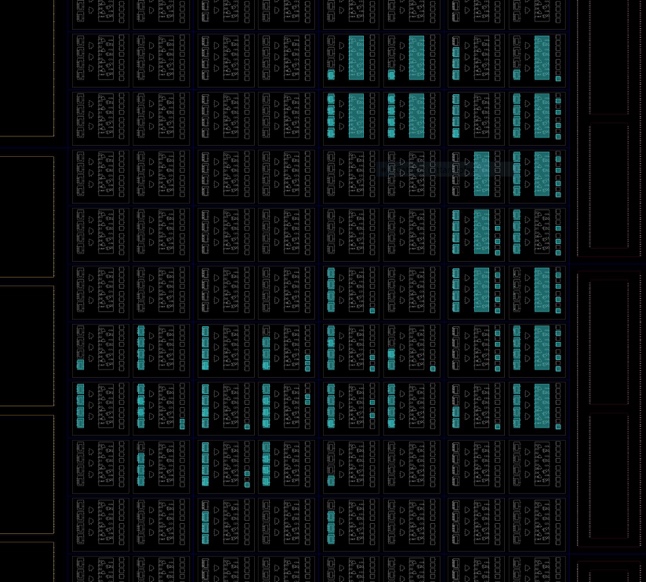
Oltre ai test generati in modo randomico abbiamo individuato altri casi di test critici per verificare l’affidabilità del nostro componente:

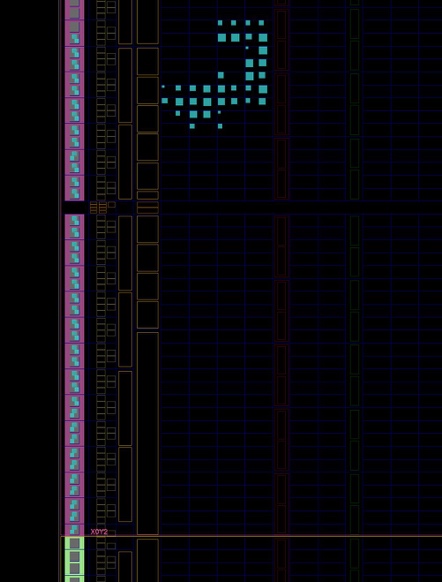
* Test con tutti i centoidi attivi e, a partire dal primo, i centroidi hanno distanza decrescente, in modo che il componente venga stressato al massimo sovrascrivendo ogni volta la distanza minima trovata.
* Test con bitmask tutta a 0.
* Test con un solo 1.
* Test con due 1.

Risultati della sintesi

Vivado Post-Syntesis utilization report:







Come previsto vengono utilizzati 49 registri come Flip Flop e 38 pin di input e output. Per quanto riguarda la logica combinatoria vengono utilizzati 112 LUT su 134600 (0.08 %).

In figura vengono mostrati gli slices e i pin di input e output effettivamente utilizzati.

Ottimizzazioni

Di seguito una breve descrizione delle ottimizzazioni sviluppate per il progetto.

La prima ottimizzazione introdotta gestisce la bitmask nel caso in cui l’ingresso è 0 oppure un multiplo di 2, quindi nella matrice esiste al massimo un unico centroide attivo, di conseguenza si passa direttamente dallo stato BITMASK a quello di DONE scrivendo in memoria direttamente la bitmask appena letta senza dover verificare se gli altri centroidi siano attivi o meno, in quanto o non ci sono centroidi attivi o l’unico centroide attivo sarà sicuramente anche il più vicino.

Un ulteriore ottimizzazione è l’utilizzo della coordinata X dei centroidi utilizzata per il calcolo della distanza senza assegnarla ad un segnale, in modo da poter utilizzarla appena viene resa disponibile dalla RAM.

Infine, quando si è nello stato Y, se un centroide non è attivo il contatore viene decrementato di 2 in modo tale da evitare la transizione allo stato successivo e quindi la lettura della corrispondente X.