**Prova Finale (Progetto di reti logiche)**

**AA. 2018/19**

**Professore di riferimento:** William Fornaciari

**Tutor di riferimento:** Davide Zoni

**Componenti del gruppo**: Edoardo Carrer

Cod\_persona: 10561353

Matricola: 870718

Amedeo Cavallo

Cod\_persona: 10562259

Matricola: 868665

**Indice**

1. Specifiche di progetto. p. 3
2. Scelte progettuali. pp. 4-5
   1. Reset
   2. Bitmask
   3. Y
   4. X
   5. Done
3. Risultati del testing. p. 5
4. Risultati della sintesi. p. 6

4.1 Presintesi

4.2 Postsintesi

1. Ottimizzazioni. p.7
2. **Specifiche di progetto**

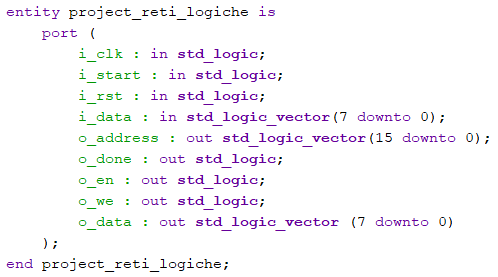
Il progetto svolto consiste nella specifica in vhdl di un componente hardware a cui una volta fornite le coordinate di otto centroidi e un punto da valutare, sia in grado di individuare quale tra questi ha “Manhattan distance" inferiore. La modalità di acquisizione dei dati avviene tramite lettura da una RAM sincrona, questa operazione viene effettuata solamente dopo che il segnale di START viene portato a ‘1’. All’indirizzo ‘0’ della RAM è salvata una bitmask da otto bit in cui ogni bit rappresenta la validità del centroide, dove il bit meno significativo rappresenta il primo centroide, e il più significativo l’ottavo. Negli indirizzi di memoria successivi sono presenti alternativamente le coordinate X e Y dei centroidi e infine quelle del punto da valutare. La scrittura del risultato va fatta all’indirizzo 19 di memoria solamente dopo che un segnale di DONE viene portato a ‘1’. Infine, il segnale di START viene portato a ‘0’ almeno fino a quando anche il segnale di DONE viene portato a ‘0’.

Il componente è costituito da 4 ingressi:

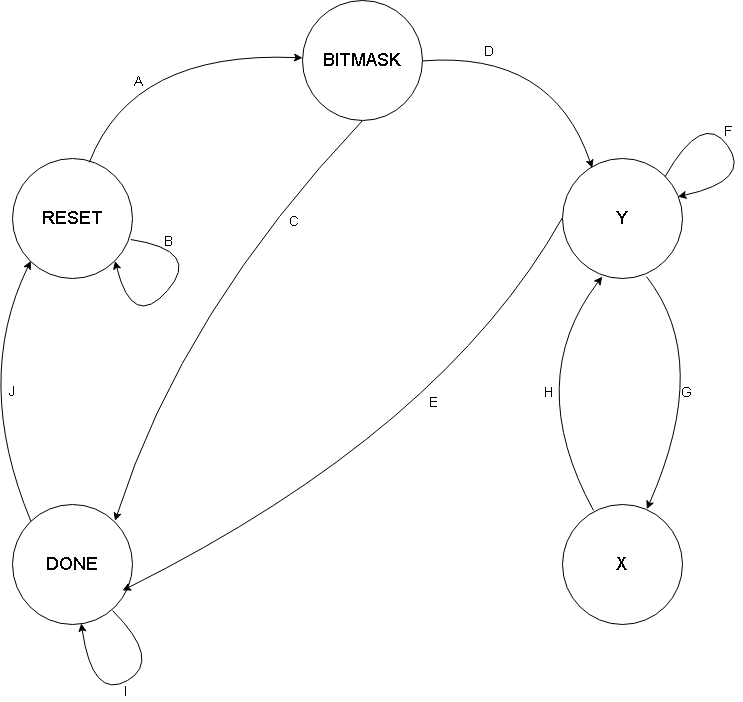
1. i\_clock: segnale di clock;
2. i\_start: segnale di inizio del processo;
3. i\_rst: segnale che quando vale ‘1’ riporta allo stato di RESET;
4. i\_data: segnale che legge i dati dalla RAM;

e 5 output:

1. o\_address: segnale di selezione dell’indirizzo della RAM;
2. o\_done: segnale che posto a 1 indica la fine dell’elaborazione per il componente;
3. o\_en: segnale che abilita l’accesso alla RAM;
4. o\_we: segnale che abilita l’accesso in scrittura alla RAM;
5. o\_data: segnale che contiene il risultato finale da scrivere in RAM;



1. **Scelte progettuali**



Per modellare il design del progetto abbiamo deciso di utilizzare una macchina di Mealy a 5 stati con trigger sul fronte di salita del clock. Di seguito una breve descrizione dei segnali e dei componenti.

**Segnali:**

State: segnale utilizzato per salvare lo stato corrente.

Mindist: segnale utilizzato per salvare la distanza minima calcolata.

Counter: segnale utilizzato per salvare l’indirizzo utilizzato per accedere alla memoria.

X: segnale utilizzato per salvare la coordinata X del punto.

Y: segnale utilizzato per salvare la coordinata Y del punto.

Yc: segnale utilizzato per salvare la coordinata Y del centroide.

**Stati:**

* 1. **RESET:** in questo stato tutti i segnali vengono portati al valore di default, quando i\_start vale ‘1’ viene scelta la transizione A e ad o\_en viene assegnato il valore 1, mentre il resto degli output rimane invariato. Finché i\_start rimane a 0 viene scelto l’autoanello B e tutti i segnali rimangono uguali.
  2. **BITMASK:** in questo stato si legge la bitmask dei centroidi dalla RAM, viene decrementato di 1 il contatore e viene scelta la transizione D, tranne nel caso in cui sia presente nella bitmask un unico centroide attivo, in questo caso si sceglie la transizione C e si pongono o\_we ad 1, o\_address all’indirizzo 19, e o\_data uguale alla bitmask appena letta.
  3. **Y:** è lo stato in cui si legge la coordinata Y del punto e dei centroidi, ed inoltre si scrive il risultato finale in RAM, l’autoanello F viene scelto nel caso il centroide non sia attivo, e o\_en posto uguale 1, se invece il centroide è attivo viene scelta la transizione G, o\_en viene posto a 1 e o\_address uguale al contatore decrementato di 2. Una volta finito di leggere le coordinate di tutti i centroidi viene scelta la transizione E, o\_we e o\_en vengono posti ad 1, o\_address a 19 ed o\_data uguale alla bitmask.
  4. **X:** è lo stato in cui si legge la coordinata X del punto e dei centroidi, ed inoltre si calcola la distanza tra loro, nel caso in cui la distanza appena calcolata sia minore di Pmindist, quest’ultima viene aggiornata. In ogni caso si sceglie la transizione H si pone o\_en uguale ad 1 e o\_address uguale al contatore decrementato di 1.
  5. **DONE:** è lo stato che indica la fine del processo, viene scelta la transizione I finchè i\_start è uguale ad 1 e o\_done viene posto ad 1. Quando i\_start viene posto a 0 si sceglie la transizione J e il componente si predispone ad un'altra elaborazione.

1. **Risultati del testing**

Abbiamo individuato N casi di test critici per verificare l’affidabilità del nostro componente. I casi critici con i relativi risultati sono:

1. 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 calcolando per ogni centroide la distanza e salvandola su Pmindist.

**Risultati**: behavioral: time: 2550ns

iteration: 2

postsintesi: time: 2550.1ns

iteration 4

1. Test con bitmask tutta a 0, o un solo 1

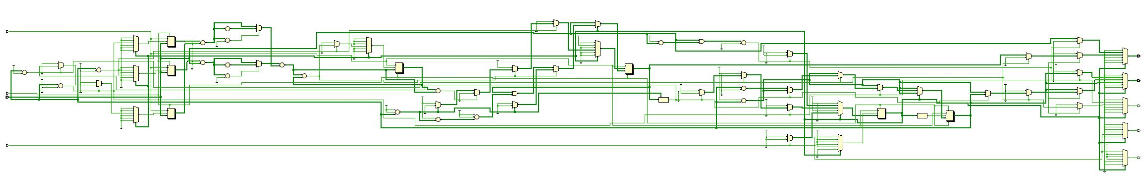
**Risultati:**

1. Test con due 1:

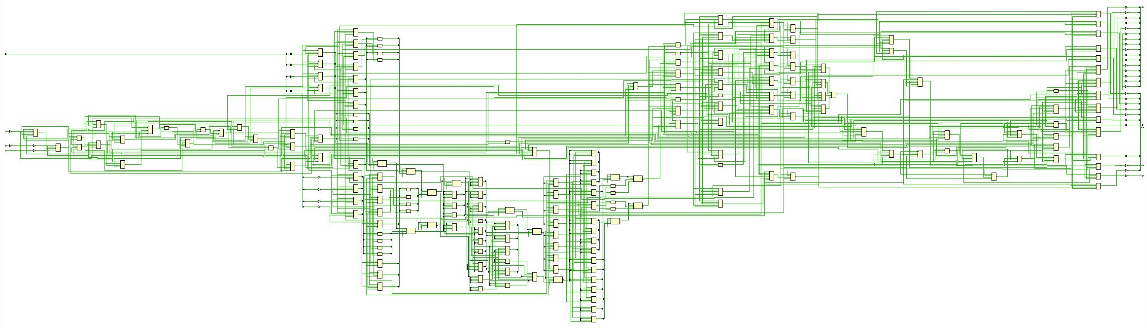
**Risultati**

Inoltre abbiamo sviluppato un programma in linguaggio python in grado di generare test casuali andando a modificare direttamente i valori presenti nel testbench, questo ci ha permesso di fare un elevato numero di test e di verificare che il nostro componente li passasse tutti.

1. **Risultati della sintesi**
2. **Pre sintesi**

****

1. **Post sintesi**

****

**5-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 centeroidi 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.