**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.
2. Scelte progettuali.
   1. Reset
   2. Bitmask
   3. Y
   4. X
   5. Done
3. Risultati del testing.
4. Risultati della sintesi.

4.1 Presintesi

4.2 Postsintesi

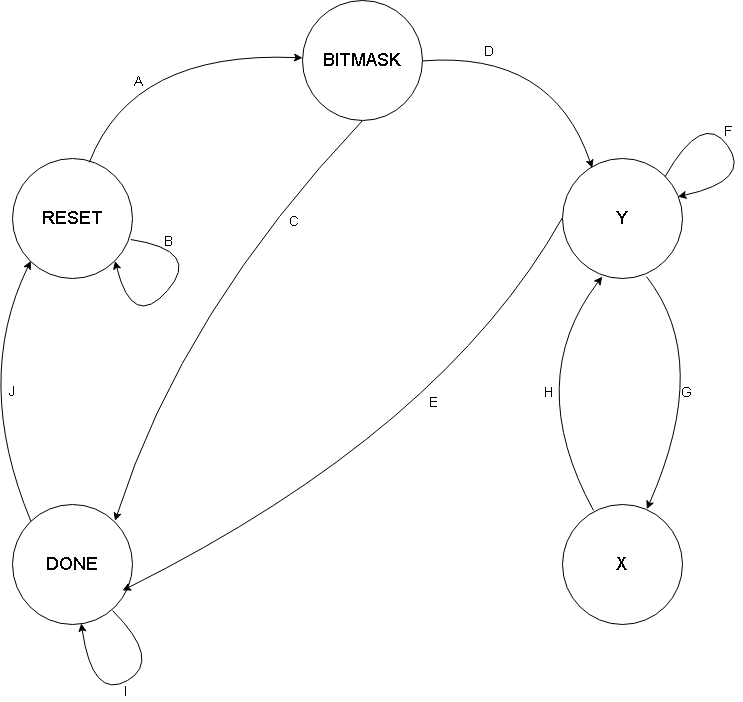
1. Ottimizzazioni.
2. **Specifiche di progetto**

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 RAM, 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’. Alla fine 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 se vale ‘1’ riporta allo stato di RESET e inizializza i segnali ai valori di default;
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 se vale 1 indica la fine del processo;
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;
6. **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 del componente con i valori di default e gli stati con le relative transizioni(con condizione e segnali di output corrispondenti).

**Valori di defalut**:

Nstate = Pstate; (segnali che memorizzano stato presente e futuro)

Nbitmask = Pbitmask; (segnali che memorizzano la bitmask di ingresso)

Nx = Px; (segnali che memorizzano la X del punto)

Ny = Py; (segnali che memorizzano la Y del punto)

Nc = Pc; (segnali che memorizzano la Y del centroide)

Ncounter = Pcounter; (contatori che memorizzano l’indirizzo per leggere le

coordinate dalla RAM);

Nmindist = Pmindist; (segnali che memorizzano la minima “Manhattan

distance” calcolata)

o\_address = ‘0…0’ (16 bit);

o\_done = '0';

o\_en = '0';

o\_we = '0';

o\_data = ‘0…0’ (8 bit);

1. **RESET:** è lo stato in cui tutti i segnali vengono portati al valore di default.

**A**: **condizione:** i\_start = ‘1’;

**output**: o\_en = ‘1’;

**B**: **condizione**: i\_start = ‘0’;

1. **BITMASK:** è lo stato in cui si legge la bitmask dei centroidi dalla RAM.

**C: condizione:** se è presente un solo centroide attivo.

**output:** o\_we = ‘1’;

o\_address = ‘0…010011’;

o\_data = i\_data;

**D: condizione:** transizione di default.

**output:** o\_address = Pcounter;

o\_en = 1;

1. **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.

**E: condizione:** se sono state lette tutte le coordinate.

**output:** o\_we = '1';

o\_address = ‘0…010011’;

o\_data = Pbitmask(bitmask da scrivere in memoria);

o\_en = 1;

**F: condizione:** se il centroide non è attivo.

**output:** o\_address = Pcounter – 2;

o\_en = 1;

**G:** **condizione:** transizione di default.

**output:** o\_en = 1;

o\_address = Pcounter;

1. **X:** è lo stato in cui si legge la coordinata X del punto e dei centroidi, ed inoltre si calcola la distanza.

**H: condizione:** transizione di default.

**output:** o\_en = 1;

o\_address = Pcounter – 1;

1. **DONE:** è lo stato che indica la fine del processo.

**I: condizione:** finchè i\_start = 1;

**output:** o\_done = 1;

**J: condizione:** quando i\_start = 0;

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.

**Risutati**:

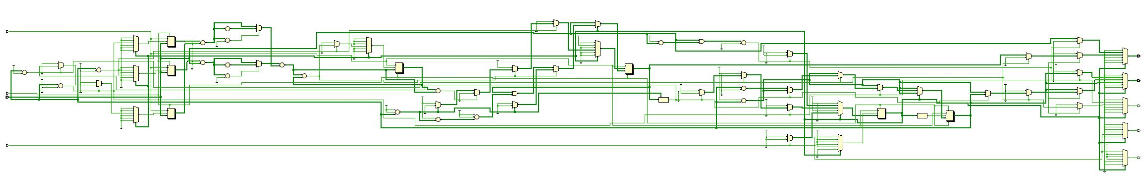
1. Test con bitmask tutta a 0, o un solo 1

**Risultati:**

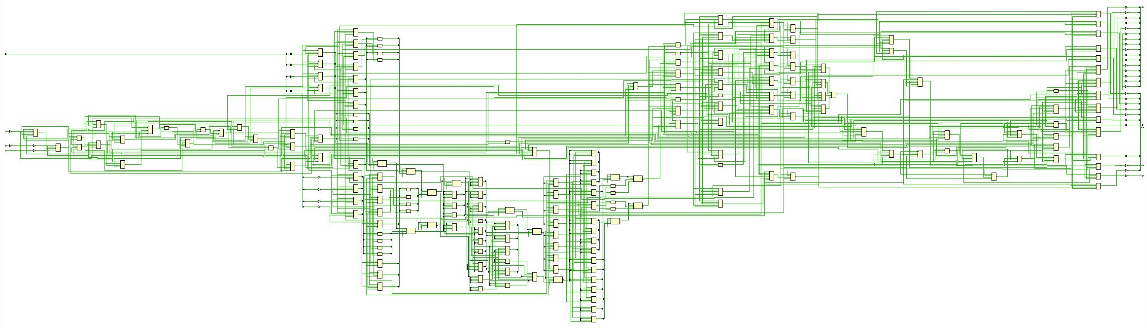
1. Test con due 1:

**Risultati**

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

****

1. **Post sintesi**

****

**5-Ottimizzazioni**

**Notevole attenzione ha successivamente richiesto la codiﬁca degli stati della FSM: in seguito a successivi tentativi si `e scoperto che la scelta iniziale del compilatore di seguire la codiﬁca one-hot sempliﬁcava la logica della FSM stessa, ma introduceva nuove sﬁde dal punto di vista del routing.**

**La scelta manuale della codiﬁca Gray ha permesso di recuperare alcune frazioni di ns dovute al ritardo di propagazione all’interno delle matrici di interconnessione del chip.**

**Il componente supera correttamente la simulazione Behavioral, la simulazione Post-Syntesis Functional e Timing**

**Ai 4 test ne `e stato aggiunto un quinto che spinge il componente verso due corner case:**

**Durante l’ultimo ciclo di clock la computazione `e 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 `e IDLE che predispone il componente ad una nuova elaborazione.**

**Secondo le speciﬁche, il testbench richiede che il risultato della computazione sia salvato nella memoria RAM. Esso `e un numero a 16 bit, pertanto va diviso in due byte: 0x0001 ne conterr`a la parte pi`u signiﬁcativa, 0x0000 la parte meno signiﬁcativa.**

**Il protocollo di comunicazione in scrittura `e gi`a stato analizzato e visualizzato nel paragrafo RAM interface**

**Durante lo stato DIM vengono elaborati i risultati dell’algoritmo di bounding box. Scrivo equzione manhattan distance**

**Design: Data la presenza di un segnale di start e di uno di reset, `e ragionevole pensare che sia necessario un automa a stati ﬁniti (FSM). Mealy per ciclo di clock**

**L’interfaccia del componente, cos`ı come presentata nelle speciﬁche [1], `e la seguente:**

**entity project\_reti\_logiche is port ( i\_clk : in std\_logic; i\_start : in std\_logic; i\_rst : in std\_logic; i\_data : in std\_logic\_vector(7 downto 0); o\_address : out std\_logic\_vector(15 downto 0); o\_done : out std\_logic; o\_en : out std\_logic; o\_we : out std\_logic; o\_data : out std\_logic\_vector (7 downto 0) ); end project\_reti\_logiche;**

**Abstract**

**Lo scopo del progetto `e la realizzazione di un componente hardware in VHDL. Esso riceve in ingresso un’immagine in toni di grigio e, dopo aver applicato un algoritmo di thresholding ad una soglia data, scrive in output l’area del rettangolo minimo che racchiude completamente l’immagine B/W risultante.**