Prova Finale

(Progetto di Reti Logiche)

Anno Accademico 2018/2019

Sommario

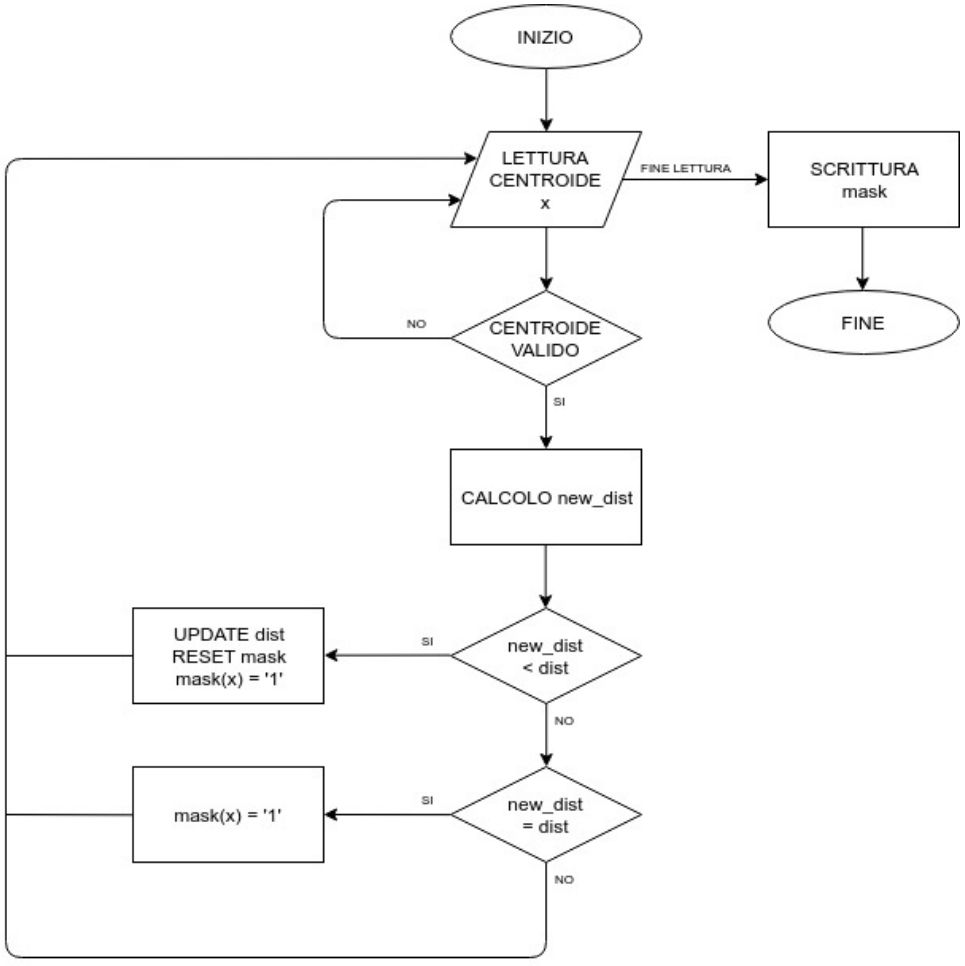
Introduzione 2
Algoritmo 2
Macchina a stati 3
Vincoli strutturali 3
Macchina a stati ottimizzata 4
Testing 5

Introduzione
Il progetto prevedeva la realizzazione in VHDL di un componente che risolvesse il seguente problema:

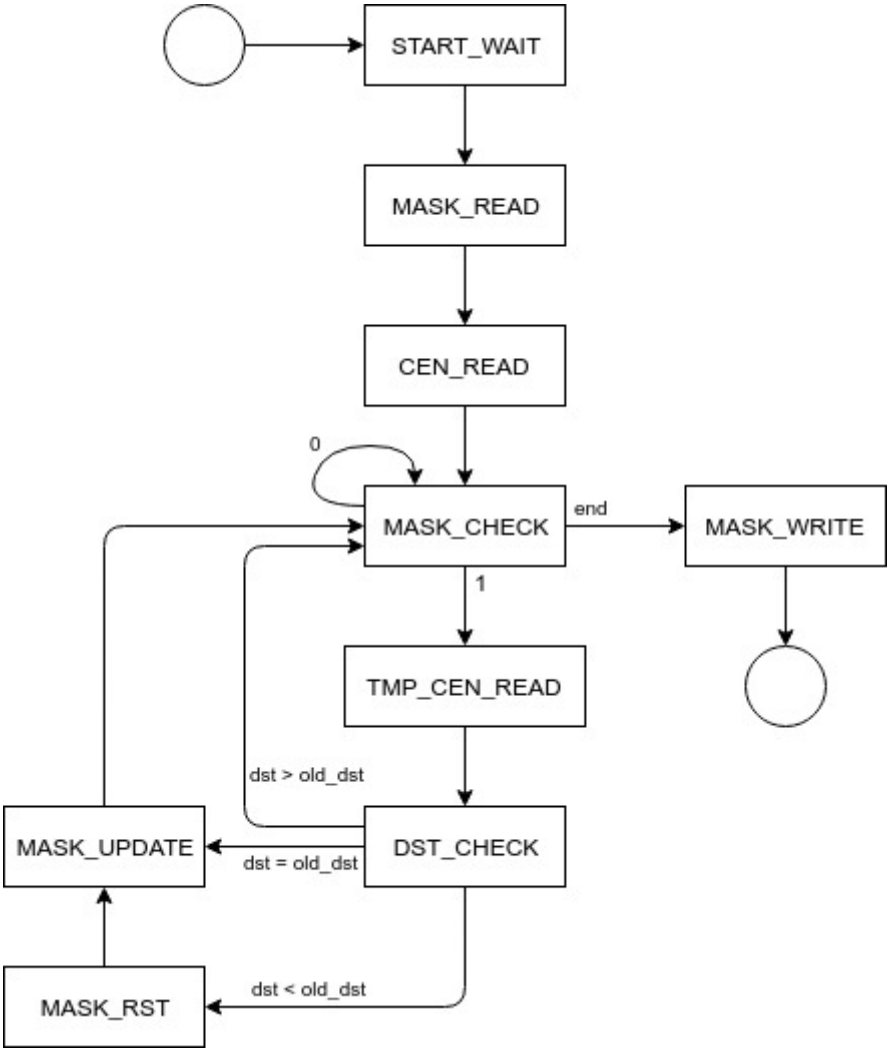
"In uno spazio 256x256 ci sono 8 punti detti centroidi, più uno di riferimento. Identificare il centroide più vicino (o i centroidi più vicini) a quello di riferimento, trascurando quelli marcati come non validi"

Algoritmo
L'algoritmo che ho utilizzato, schematizzato col diagramma di flusso riportato di seguito, è costituito da tre blocchi:

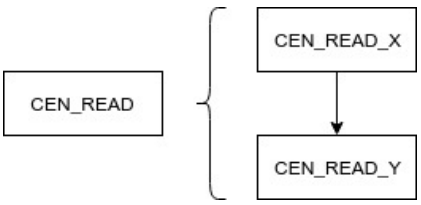
1) *Inizializzazione* - In questa fase vengono inizializzati i valori di controllo e vengono letti il centroide di riferimento e la maschera con i bit di validità dei centroidi da verificare
2) *Iterazione* - Per ogni centroide valido viene calcolata la distanza da quello di riferimento, e in base al risultato viene modificata la maschera in uscita (se la distanza calcolata è inferiore a quella minima attuale, la maschera in uscita viene azzerata e viene portato a '1' il bit relativo al centroide corrente; se la distanza calcolata corrisponde con quella minima attuale, viene portato a '1' il bit relativo al centroide corrente senza azzerare la maschera in uscita; se la distanza calcolata è superiore a quella minima attuale si passa al centroide successivo)
3) *Scrittura del risultato* - La maschera in uscita viene scritta nell'indirizzo di memoria preposto e viene segnalata la fine della computazione



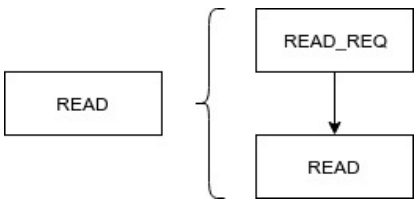
Macchina a stati
Dopo aver definito l'algoritmo, ho realizzato la macchina a stati corrispondente (non sintetizzabile in VHDL a causa dei vincoli strutturali analizzati nella sezione apposita)



Vincoli strutturali
A questo punto della progettazione ho identificato gli stati della macchina per i quali era irrealizzabile una descrizione fedele in VHDL, e ho apportato le seguenti modifiche:

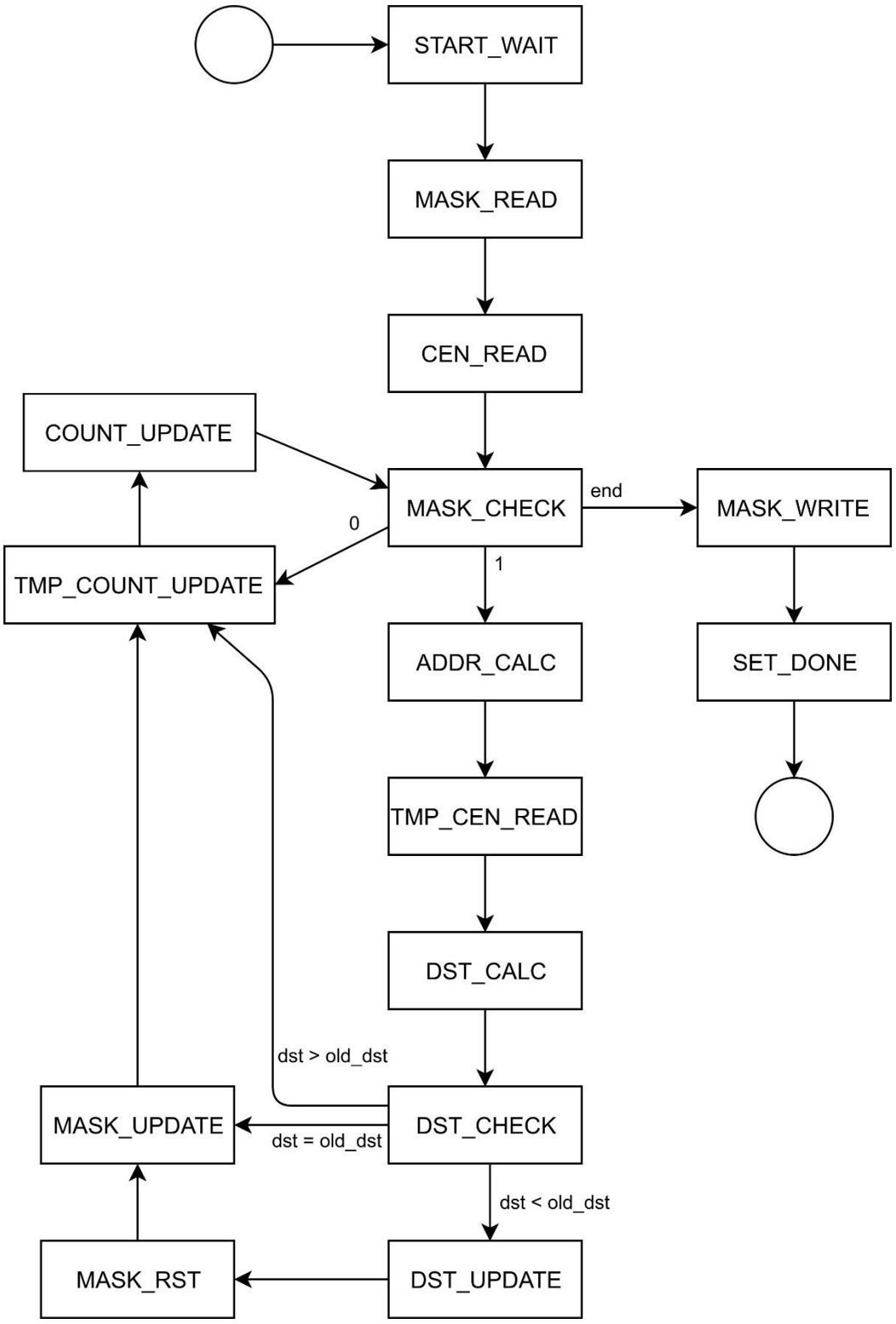


▪ *Lettura centroidi* - All'interno della memoria, i centroidi erano memorizzati mediante coordinate cartesiane, una per ogni indirizzo di memoria; sono quindi stati necessari due stati per ogni centroide letto



▪ *Lettura da RAM* - La memoria descritta richiedeva un ciclo di clock per rendere disponibile il dato dopo la richiesta mediante segnali di controllo; ho quindi utilizzato un primo stato per la richiesta del dato e un secondo stato per la lettura effettiva

Macchina a stati ottimizzata
Apportate le modifiche necessarie per rendere la macchina a stati sintetizzabile in VHDL, il diagramma a stati che ne deriva è il seguente (i nomi degli stati sono il più possibile auto-esplicativi)



Testing

Conclusa la descrizione e la sintesi del componente, ho effettuato una serie di test per verificarne il funzionamento; più in particolare, ho svolto un elevato numero di test generati casualmente, e un numero più ristretto di test specifici, volti a sollecitare funzionalità particolari del componente. Quanto segue è relativo alla simulazione comportamentale pre-sintesi, alla simulazione funzionale post-sintesi e alla simulazione temporale post-sintesi (con un periodo di clock di 100 ns, come da specifica).

- *Test casuali* - Il componente ha risposto correttamente a 100/100 casi di test generati casualmente
- *Test specifici* - Ho effettuato test specifici per testare tutte le diramazioni comportamentali possibili, dopodichè ho concluso con la verifica dei seguenti casi limite
    - Maschera in input nulla
    - Centroidi tutti equidistanti
    - Centroidi tutti coincidenti
    - Centroidi sovrapposti a quello di riferimento

  Di seguito sono riportati i tempi di esecuzione dei casi limite, che costituiscono dei bound sul tempo di esecuzione del componente

  Caso ottimo 7,450000 µs
  Caso pessimo 23,450000 µs

Il componente risponde correttamente a tutti i test effettuati, sia casuali che specifici.