Progetto finale di Reti Logiche

Politecnico di Milano

Anno Accademico 2018-2019

Documentazione

Antonino Elia Mandri
Matricola: 870882
Codice persona 10533811

Prof. Fabio Salice

1 Obiettivo e specifiche:

1.1 Descrizione generale

Sia dato uno spazio bidimensionale definito in termini di dimensione orizzontale e verticale, siano date le posizioni di N punti, detti "centroidi", appartenenti a tale spazio. Si vuole implementare un componente hardware descritto in VHDL che, una volta fornite le coordinate di un punto appartenente a tale spazio, sia in grado di valutare quale/i e dei centroidi risulti più vicino secondo la distanza di Manhattan.
Degli N centroidi K <= N sono quelli su cui calcolare la distanza dal punto dato. I K centroidi sono indicati da una maschera di ingresso a N bit: il bit a '1' indica che il centroide è valido e deve quindi essere calcolata la distanza, viceversa il bit a '0' indica che il centroide non va esaminato. La vicinanza al centroide viene espressa tramite una maschera di uscita di N bit dove ogni bit corrisponde ad un centroide: il bit viene posto a 1 se il centroide è il più vicino al punto fornito, 0 negli altri casi. Nel caso in cui il punto considerato sia equidistante da 2 o più centroidi, i bit della maschera di uscita corrispondenti a tali centroidi saranno tutti posti a 1. Per entrambe le maschere il bit meno significativo è quello più a destra e il bit i-esimo si riferisce al centroide i-esimo. Per il progetto il numero di centroidi N è pari a 8.
Lo spazio in questione è un quadrato di dimensione 256x256, le coordinate dei centroidi e del punto da valutare, così come la maschera di ingresso sono memorizzate in una memoria (la cui implementazione non è parte del progetto).
La distanza di Manhattan nello spazio in questione è definita come la somma dei valori assoluti delle differenze tra le coordinate.

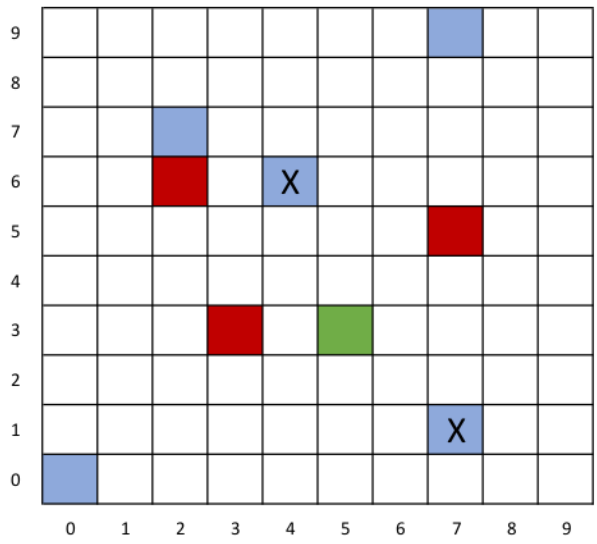

Figura 1.

In figura 1 è mostrato uno spazio 10x10 in cui la cella verde identifica il punto da cui misurare la distanza. In rosso i centroidi non validati dalla maschera (relativi bit posti a 0). In blu i centroidi validi, il punto risulta equidistante dai due centroidi blu contrassegnati dalla X.

1.2 Dati

I dati, ciascuno di dimensione 8 bit, sono memorizzati in una memoria con indirizzamento al Byte partendo dalla posizione 0.

- L'indirizzo 0 è usato per memorizzare la maschera di ingresso.
- Gli indirizzi da 1 a 16 sono usati per memorizzare le coordinate a coppie X e Y dei centroidi:
    - 1 - Coordinata X 1° centroide.
    - 2 - Coordinata Y 1° centroide.
    - 3 - Coordinata X 2° centroide.
    - 4 - Coordinata Y 2° centroide.
    - ...
    - 15 - Coordinata X 8° centroide.
    - 16 - Coordinata Y 8° centroide.
- Gli indirizzi 17 e 18 sono usati per memorizzare le coordinate X e Y rispettivamente del punto da valutare.
- L'indirizzo 19 è usato per scrivere la maschera di uscita.

1.3 Interfaccia del componente

Il componente descritto ha la seguente interfaccia.

```
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;
```

In particolare:

- i_clk è il segnale di CLOCK in ingresso generato dal TestBench;
- i_start è il segnale di START generato dal TestBench;
- i_rst è il segnale di RESET che inizializza la macchina pronta per ricevere il primo segnale;
- i_data è il segnale (vettore) che arriva dalla memoria in seguito ad una richiesta di lettura;
- o_address è il segnale (vettore) di uscita che manda l'indirizzo alla memoria;
- o_done è il segnale di uscità che comunica la fine dell'elaborazione e la conseguente scrittura in memoria del dato di uscita;
- o_en è il segnale di ENABLE da dover mandare alla memoria (=1) per poter comunicare sia in lettura che in scrittura;
- o_we è il segnale di WRITE ENABLE da dover mandare alla memoria (=1) per poter scriverci. Per leggere da memoria esso deve sempre essere a 0;
- o_data è il segnale (vettore) di uscita dal componente verso la memoria.

1.4 Note ulteriori sulla specifica

Il modulo si partirà nella elaborazione quando il segnale di START in ingresso verrà posto a 1. Il segnale di START rimarrà alto fino a che il segnale di DONE non verrà portato alto. Al termine della computazione e una volta scritto il risultato in memoria, il modulo deve impostare a 1 il segnale di DONE che notifica la fine dell'esecuzione. Il segnale DONE rimarrà alto fino a che il segnale di START non è riportato a 0. Un nuovo segnale di start non può essere dato fin tanto che DONE non è nuovamente portato a 0.

1.5 Strumenti di sintesi utilizzati

- XILINX VIVADO WEBPACK;
- Target FPGA (xc7a200tfbg484--1) ;

2 Implementazione

2.1 Descrizione generale

La soluzione progettata si basa su una macchina a stati finiti che descrive il funzionamento e le diverse fasi dell'algoritmo di risoluzione implementato. La descrizione informale dell'algoritmo implementato dalla macchina a stati finiti è la seguente.
In primo luogo viene salvata la maschera d'ingresso in un registro, successivamente vengono salvate in due registri le coordinate del punto di cui valutare le distanze dei centroidi. Un registro di supporto denominato *alpha* tiene traccia di quale centroide viene esaminato in dato momento, per fare ciò viene inizializzato a "00000001" ad indicare che si sta analizzando il centroide 1. Ad ogni iterazione *alpha* viene shiftato di una posizione a sinistra. Per determinare se il centroide i-esimo è valido, e quindi procedere alla lettura e memorizzazione delle coordinate del centroide, al calcolo della distanza, si esegue l'and bit a bit tra *alpha* e la maschera d'ingresso, se il risultato dell'operatore and è diverso da "00000000" si procede all'acquisizione delle coordinate del centroide quindi al calcolo della distanza e successivamente al controllo se la distanza appena calcolata è minore della minima distanza calcolata fino a quel punto dell'esecuzione. Se la nuova distanza è strettamente minore della precedente distanza minima il registro che memorizza durante l'esecuzione l'output che verrà scritto a fine computazione, viene posto con tutti i bit a 0 escluso il bit in posizione i-esima (secondo la codifica delle maschere) che presenterà un 1. Se la nuova distanza è minore uguale a quella precedente, all'output verrà aggiunto un 1 in posizione i-esima. Se invece la nuova distanza è maggiore della precedente l'output rimane invariato. Il ciclo procede in questo modo fino a che il vettore *alpha* non diventa uguale a "00000000" dopo di chè si procede a scrivere in memoria l'output così ottenuto e la computazione termina.

2.2 Descrizione implementazione

2.2.1 Segnali interni

Il componente usa diversi segnali interni durante la computazione. I più importanti dal punto di vista concettuale sono:

- *signal current_state: state_type;* memorizza lo stato corrente della macchina.

- *signal mask: STD_LOGIC_VECTOR(7 downto 0);* Vettore da 8 bit per memorizzare la maschera di ingresso.
- *signal Xp: STD_LOGIC_VECTOR(8 downto 0);*
- *signal Yp: STD_LOGIC_VECTOR(8 downto 0);* memorizzano le due coordinate del punto di cui calcolare la distanza dei centroidi.
- *signal coorX: STD_LOGIC_VECTOR(8 downto 0);*
- *signal coorY: STD_LOGIC_VECTOR(8 downto 0);* memorizzano le coordinate del centroide da valutare. Le coordinate vengono estese, concatenando uno 0 in posizione più significativa, su 9 bit per poter calcolare la distanza, anch'essa su 9 bit senza incorrere in overflow, altrimenti possibile se la distanza dovesse risultare maggiore di 255.
- *signal distance: UNSIGNED(8 downto 0);* viene memorizzata la distanza calcolata tra il punto ed un centroide.
- *signal min_distance: UNSIGNED(8 downto 0);* tiene traccia della distanza minima calcolata nei cicli precedenti.
- *signal current_address: UNSIGNED(15 downto 0);* memorizza l'indirizzo corrente della memoria a cui si accede.
- *signal alpha: STD_LOGIC_VECTOR(7 downto 0);* tiene traccia, mediante codifica posizionale, del centroide corrente in fase di valutazione.
- *signal output: STD_LOGIC_VECTOR(7 downto 0);* memorizza l'output temporaneo durante l'esecuzione, a fine computazione sarà scritto all'indirizzo di memoria della maschera di uscita.

I segnali interni che memorizzano le coordinate del punto e dei centroidi, così come *distance* e *min_distance* sono di 9 bit. In fase di lettura delle coordinate viene concatenato uno '0' in posizione più significativa, questa scelta è dettata dalla possibilità che la distanza tra punto e centroide possa variare tra 0 e 510 e quindi necessita di 9 bit per essere memorizzata.
Altri segnali utilizzati e qui non riportati sono di supporto per le fasi di modifica dei segnali precedentemente elencati. Segnali dichiarati come UNSIGNED sono stati dichiarati tali per semplificare e rendere più leggibile il codice riducendo il numero di cast necessari ad eseguire operazioni aritmetiche.

2.2.2 Descrizione Stati

- INIT: stato di attesa del primo segnale di reset, una volta ricevuto tale segnale la macchina passa allo stato di reset e non tornerà più nello stato di INIT.
- RESET: stato di reset, vengono inizializzati tutti i segnali necessari per l'esecuzione della FSM. In particolare *min_distance* viene impostato a "111111111", *o_done* viene messo a '0', *o_address* viene inizializzato a "0000000000000000" e *current_address* a "0000000000000001". Questi assegnamenti devono essere effettuati nello stato di RESET in modo che l'esecuzione possa ripartire più di una volta.
- START: stato di start, attiva la memoria impostando il segnale *o_en* a 1 e avvia l'esecuzione alla ricezione del segnale alto su *i_start;*
- STORE_MASK: salva la maschera di ingresso.
- READ_XP: assegna a *o_address* l'indirizzo della coordinata X del punto.
- STORE_XP: salva la coordinata X del punto.
- READ_YP: assegna a *o_address* l'indirizzo della coordinata Y del punto.
- STORE_YP: salva la coordinata Y del punto.
- CHECK_VALIDITY: controlla se il centroide è valido. Se valido vengono lette le coordinate del centroide e calcolata la distanza, altrimenti incrementa l'indirizzo corrente di 2, shifta il segnale *alpha* e procede nello stato INCREASE_ADDRESS;
- READ_X: assegna a *o_address* l'indirizzo della coordinata X del centroide.
- STORE_X: salva la coordinata X del centroide.
- READ_Y: assegna a *o_address* l'indirizzo della coordinata Y del centroide.
- STORE_Y: salva la coordinata Y del centroide.
- CALC_DISTANCE: calcola la distanza tra punto e centroide.
- CHECK_MIN: confronta distanza attuale con la distanza minima e in base a ciò aggiorna il segnale *output*.
- INCREASE_ADDRESS: allinea il segnale *o_address* con *current_address* nel caso in cui il centroide fosse risultato non valido dal confronto con la maschera.
- WRITE_OUTPUT: scrive nell'indirizzo 19 il valore di output dopo aver scandito tutti i centroidi validi, inoltre imposta *o_done* a '1'.
- DONE: stato terminale della computazione, assegna 0 ai segnali *o_done, o_en, o_we.* Successivamente la macchina ritorna nello stato di RESET.

2.3 FSM

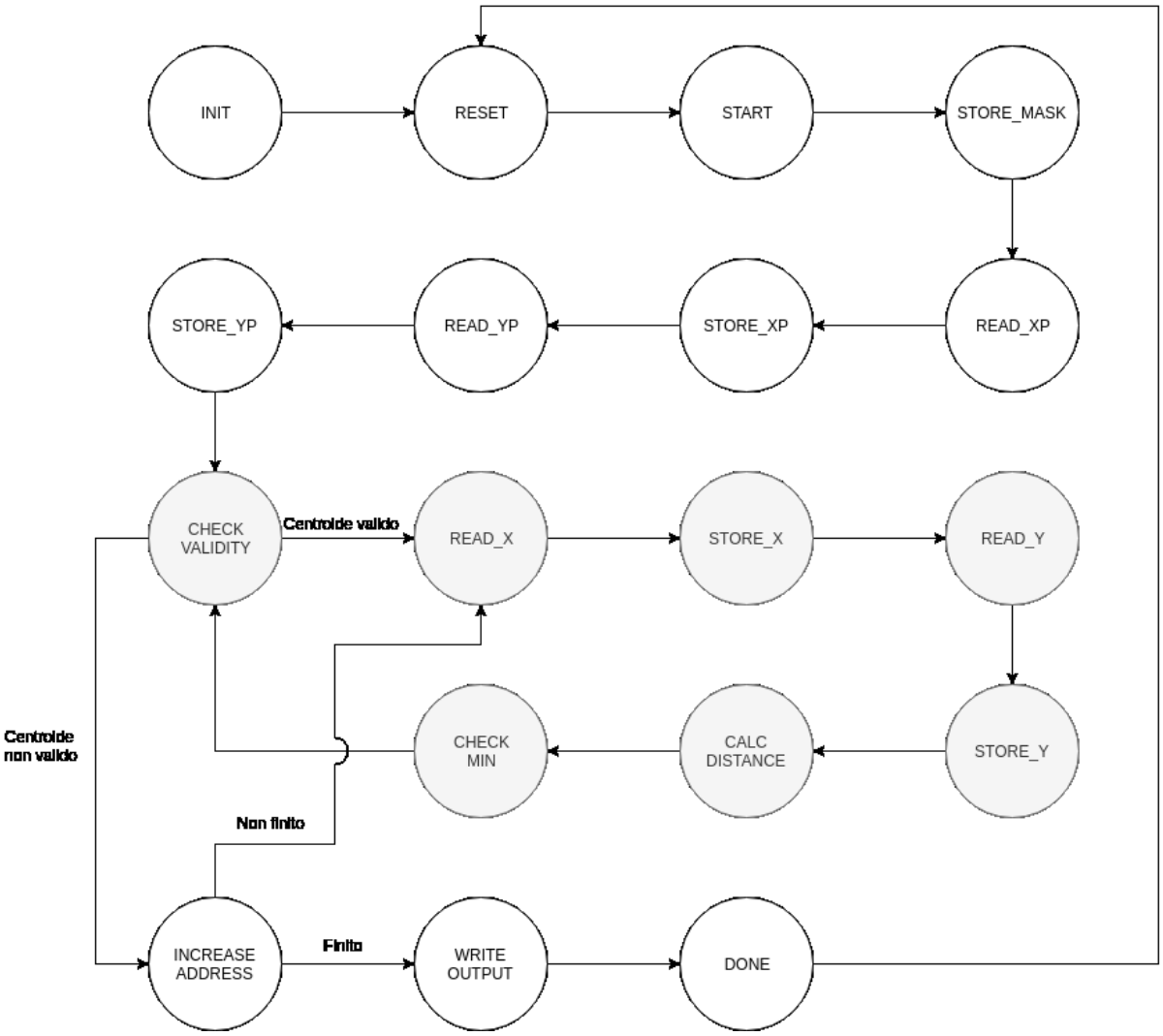

Figura 2

Per rendere più chiaro il grafo rappresentante la macchina a stati finiti sono stati omessi gli archi uscenti da ogni stato e diretti verso lo stato di RESET che vengono percorsi se il segnale i_rst viene posto a 1 indipendentemente dallo stato complessivo della macchina in quel istante di tempo.

3. Test

Il componente supera tutti i casi di test a cui è stato sottoposto, ogni test è stato eseguito in Behavioral Simulation, Post-Synthesis Functional Simulation, Post-Synthesis Timing Simulation, Post-Implementation Functional Simulation e Post-Implementation Timing Simulation. In particolare oltre al Test Bench fornito, ne sono stati effettuati altri casi determinati sulla base di due approcci differenti: testing black box e white box. Per ogni test vengono riportati i tempi di esecuzione. Nota: per rendere più leggibili ed autoesplicative le tabelle riassuntive dei test effettuati, entrambe le maschere sono invertite, cioè con il bit più significativo a destra, rispetto alle maschere lette e scritte in memoria.

3.1 Testing Black Box

Occorre dividere i possibili input in gruppi i cui elementi si ritiene che saranno trattati similmente dal processo elaborativo. Questi gruppi formeranno delle classi di equivalenza
La prima suddivisione avviene con i due casi limite della maschera d'ingresso, che sono:
maschera uguale a "11111111" e maschera uguale a "00000000".

3.1.1 Maschera d'ingresso "11111111"

Centroidi equidistanti

In questo test si è predisposto che tutti gli 8 centroidi siano distinti tra loro e il punto ed equidistanti dal punto.

Maschera d'ingresso		1	1	1	1	1	1	1	1
	Punto	Centroide 1	Centroide 2	Centroide 3	Centroide 4	Centroide 5	Centroide 6	Centroide 7	Centroide 8
Coordinata X	100	110	115	95	80	85	103	83	99
Coordinata Y	100	110	95	115	100	105	83	103	119
Distanza	---	20	20	20	20	20	20	20	20
Maschera di uscita		1	1	1	1	1	1	1	1

	Tempo di esecuzione	Esito
Behavioral Simulation	7 µs	Corretto
Post-Synthesis Functional Simulation	7,100100 µs	Corretto
Post-Synthesis Timing Simulation	7,203737 µs	Corretto

Centroidi coincidenti

In questo test si è predisposto che tutti e gli 8 centroidi siano coincidenti al punto.

Maschera d'ingresso		1	1	1	1	1	1	1	1
	Punto	Centroide 1	Centroide 2	Centroide 3	Centroide 4	Centroide 5	Centroide 6	Centroide 7	Centroide 8
Coordinata X	189	189	189	189	189	189	189	189	189
Coordinata Y	32	32	32	32	32	32	32	32	32
Distanza	---	0	0	0	0	0	0	0	0
Maschera di uscita		1	1	1	1	1	1	1	1

	Tempo di esecuzione	Esito
Behavioral Simulation	7 µs	Corretto
Post-Synthesis Functional Simulation	7,100100 µs	Corretto
Post-Synthesis Timing Simulation	7,203737 µs	Corretto

Centroidi a posizione e distanza pseudo-casuali

In questo test si è predisposto che i centroidi siano in posizione pseudo-casuale, i valori delle coordinate sono stati ottenuti tramite il generatore di numeri pseudo-casuali (con distribuzione uniforme) fornito dalla libreria standard del C.

Maschera d'ingresso		1	1	1	1	1	1	1	1
	Punto	Centroide 1	Centroide 2	Centroide 3	Centroide 4	Centroide 5	Centroide 6	Centroide 7	Centroide 8
Coordinata X	235	124	24	4	234	139	231	172	195
Coordinata Y	200	178	63	229	184	90	26	223	53
Distanza	---	133	348	260	17	206	178	86	187
Maschera di uscita		0	0	0	1	0	0	0	0

	Tempo di esecuzione	Esito
Behavioral Simulation	7 µs	Corretto
Post-Synthesis Functional Simulation	7,100100 µs	Corretto
Post-Synthesis Timing Simulation	7,203737 µs	Corretto

3.1.2 Maschera d'ingresso "00000000"

In questo caso, avendo la maschera di ingresso posta a "00000000", nessun punto, come da specifica, viene valutato. In ragione di ciò un solo caso di test con coordinate pseudo-casuali è sufficiente.

Maschera d'ingresso		0	0	0	0	0	0	0	0
	Punto	Centroide 1	Centroide 2	Centroide 3	Centroide 4	Centroide 5	Centroide 6	Centroide 7	Centroide 8
Coordinata X	235	124	24	4	234	139	231	172	195
Coordinata Y	200	178	63	229	184	90	26	223	53
Distanza	---	133	348	260	17	206	178	86	187
Maschera di uscita		0	0	0	0	0	0	0	0

	Tempo di esecuzione	Esito
Behavioral Simulation	3 µs	Corretto
Post-Synthesis Functional Simulation	3,100100 µs	Corretto
Post-Synthesis Timing Simulation	3,203737 µs	Corretto

3.1.3 Maschera pseudo casuale

Quest'ultimo caso stimola il comportamento del componente in condizioni non limite.

Maschera d'ingresso		0	1	1	0	0	1	0	0
	Punto	Centroide 1	Centroide 2	Centroide 3	Centroide 4	Centroide 5	Centroide 6	Centroide 7	Centroide 8
Coordinata X	235	124	24	4	234	139	231	172	195
Coordinata Y	200	178	63	229	184	90	26	223	53
Distanza	---	133	348	260	17	206	178	86	187
Maschera di uscita		0	0	0	0	0	1	0	0

	Tempo di esecuzione	Esito
Behavioral Simulation	4,500 µs	Corretto
Post-Synthesis Functional Simulation	4,600100 µs	Corretto
Post-Synthesis Timing Simulation	4,703737 µs	Corretto

3.2 Testing funzionale

Questa tipologia di testing si basa sulla conoscenza dell'implementazione e quindi dei punti a maggiore criticità che devono essere testati.

3.2.1 Reset

In questo caso di test è stato testato il funzionamento del componente nel caso in cui durante la computazione, precisamente 1100 ns dopo il segnale di start, il segnale di reset viene portato alto per un ciclo di clock. Il componente ritorna nello stato di reset, reinizializzando i segnali necessari e riesegue per intero la computazione. Un test con gli stessi valori di input, ma senza il reset aggiuntivo mostra una riduzione del tempo di esecuzione di 1200 ns.

Maschera d'ingresso		0	1	0	0	1	1	1	1
	Punto	Centroide 1	Centroide 2	Centroide 3	Centroide 4	Centroide 5	Centroide 6	Centroide 7	Centroide 8
Coordinata X	196	155	7	150	213	29	64	235	56
Coordinata Y	143	189	253	232	198	134	183	132	200
Distanza	---	87	299	135	72	176	172	50	197
Maschera di uscita		0	0	0	0	0	0	1	0

	Tempo di esecuzione	Esito
Behavioral Simulation	6,700 µs	Corretto
Post-Synthesis Functional Simulation	6,800100 µs	Corretto
Post-Synthesis Timing Simulation	6,803737 µs	Corretto

3.2.2 Distanza massima

Questo caso di test vuole verificare il caso in cui la distanza dei centroidi dal punto è la massima possibile nello spazio, al fine di stimolare possibili overflow.

Maschera d'ingresso		1	1	1	1	1	1	1	1
	Punto	Centroide 1	Centroide 2	Centroide 3	Centroide 4	Centroide 5	Centroide 6	Centroide 7	Centroide 8
Coordinata X	0	255	255	255	255	255	255	255	255
Coordinata Y	0	255	255	255	255	255	255	255	255
Distanza	---	510	510	510	510	510	510	510	510
Maschera di uscita		1	1	1	1	1	1	1	1

	Tempo di esecuzione	Esito
Behavioral Simulation	7 µs	Corretto
Post-Synthesis Functional Simulation	7,100100 µs	Corretto
Post-Synthesis Timing Simulation	7,103737 µs	Corretto

Per completezza viene fornito anche il valore del segnale interno *min_distance*.
min_distance = 111111110 valore in codifica binaria naturale corrispondente al valore 510 in codifica decimale.

4 Conclusioni

Dai casi di test effettuati si traggono informazioni sul tempo di computazione. Il caso pessimo è dato dalla maschera d'ingresso posta a 11111111 in cui il tempo in esecuzione in Behavioral functional simulation è di 7 µs, corrispondenti a 70 cicli di clock. Il caso ottimo è invece dato dalla maschera posta a 00000000 con un tempo in Behavioral functional simulation di 3 µs, corrispondenti a 30 cicli di clock.
Non sono stati riportati altri casi di test generati pseudo casualmente in quanto poco significativi alla verifica della correttezza della soluzione progettata. Infatti le possibili combinazioni di input d'ingresso sono 255^{19} rendendo dunque computazionalmente intrattabile la generazione e successiva simulazione di una suite di casi di test che possa stimolare ulteriormente percorsi critici e quindi rilevare errori nel componente.