Prova finale, Progetto di Reti Logiche 2018/2019

Studente: Caruso Marco Giuseppe, mat. 866190, cod. pers. 10531943;

Professore: Fornaciari William;

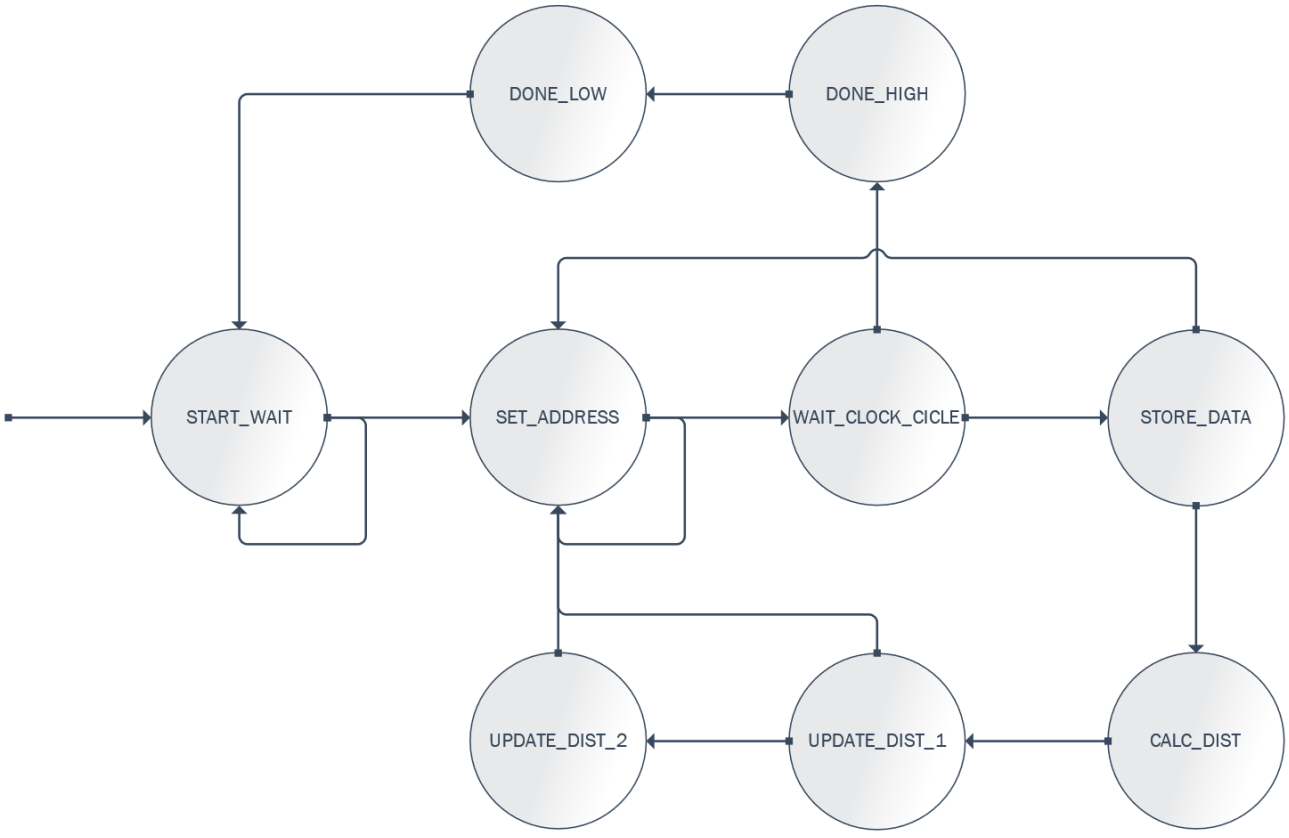
Tutor: Zoni Davide;

Sommario

Descrizione dei segnali (ciascuno si presenta nelle due forme cur_* e next_*)

- **cur_state, next_state:** segnale che indica lo stato;
- **cur_address, next_address:** segnale che contiene l'indirizzo della memoria a cui accedere;
- **cur_operation, next_operation:** segnale di tipo integer che determina quale operazione eseguire tra:
  - leggere dalla ram la maschera di don't care (operation=0);
  - leggere dalla ram le coordinate del punto da valutare (operation=1 or operation=2);
  - leggere dalla ram le coordinate degli 8 centroidi (2<operation<19);
  - scrivere sulla ram il risultato (operation=19);
- **cur_x_centre, next_x_centre:** segnale che contiene la x del punto da valutare;
- **cur_y_centre, next_y_centre:** segnale che contiene la y del punto da valutare;
- **cur_x, next_x:** segnale che contiene la x del centroide corrente;
- **cur_y, next_y:** segnale che contiene la y del centroide corrente;
- **cur_min_distance, next_min_distance:** segnale che contiene la distanza minima tra il punto da valutare e il centroide più vicino (o i centroidi più vicini);
- **cur_distance, next_distance:** segnale che contiene la distanza tra il punto da valutare e il centroide corrente;
- **cur_out_mask, next_out_mask:** segnale che contiene il risultato (è pronto quando operation=19);
- **cur_point_number, next_point_number:** segnale che contiene il numero (da 0 a 7) del centroide corrente;
- **cur_dont_care_mask, next_dont_care_mask:** segnale che contiene la maschera di don't care;

Schema Macchina a Stati Finiti



Descrizione degli stati

- **START_WAIT:** stato iniziale, nel quale vengono inizializzati tutti i segnali next_* e si pone next_state<=SET_ADDRESS appena il segnate i_start diventa 1, altrimenti si rimane in START_WAIT;
- **SET_ADDRESS:** stato nel quale si assegna il valore ai segnali o_address, o_en e o_we. Il valore dei tre segnali viene deciso in base al valore del segnale cur_operation nel seguente modo:
    - se operation=0 si fornisce alla ram l'indirizzo della maschera di don't care;
    - se operation=1 si fornisce alla ram l'indirizzo della x del punto da valutare;
    - se operation=2 si fornisce alla ram l'indirizzo della y del punto da valutare;
    - se 2<operation<19 si determina se il centroide corrente (indicato dal segnale cur_point_number) è da valutare:
        - in caso positivo si fornisce alla ram l'indirizzo della x del centroide corrente, e successivamente l'indirizzo della y del centroide corrente;
        - in caso negativo si passa al centroide successivo e si pone next_state<=SET_ADDRESS;

  In tutti i casi, tranne l'ultimo, si pone next_state<=WAIT_CLOCK_CICLE;
- **WAIT_CLOCK_CICLE:** stato nel quale non si esegue nessuna operazione, permette alla ram di preparare sul segnale i_data il valore richiesto tramite i segnali o_address, o_en e o_we. Lo stato prossimo viene determinato in base al valore di cur_operation nel seguente modo:
    - se cur_operation<19 si pone next_state<=STORE_DATA;
    - se cur_operation=19 si pone next_state<=DONE_HIGH;
- **STORE_DATA:** stato nel quale si salva il valore presente in i_data in base al valore di cur_operation nel seguente modo:
    - se operation=0 si salva i_data in next_dont_care_mask;
    - se operation=1 si salva i_data in next_x_centre;
    - se operation=2 si salva i_data in next_y_centre;
    - se 2<operation<19 si salva i_data in next_x (se cur_operation è pari) opppure in next_y (se cur_operation è dispari);

  Infine, si incrementa il valore di cur_operation e si pone next_state<=SET_ADDRESS, tranne nel caso in cui 2<cur_operation<19 e cur_operation è dispari, in cui si pone next_state<=CALC_DIST;
- **CALD_DIST:** stato nel quale si calcola la distanza di Manhattan tra il punto da valutare e il centroide corrente, si pone next_state<=UPDATE_DIST_1;
- **UPDATE_DIST_1:** stato nel quale avviene il primo aggiornamento della maschera di uscita (out_mask) nel seguente modo:
    - se cur_distance<cur_min_distance si aggiorna la distanza minima, si azzera la maschera di uscita e si pone next_state<=UPDATE_DIST_2;
    - se cur_distance=cur_min_distance non si modifica la maschera di uscita e si pone next_state<=UPDATE_DIST_2;
    - altrimenti di pone next_state<=SET_ADDRESS (il centroide corrente non si trova a distanza minima);
- **UPDATE_DIST_2:** stato nel quale avviene il secondo aggiornamento della maschera di uscita, ponendo a 1 il bit corrispondente al centroide corrente (nel codice si ha next_*out_mask (cur_point_number - 1) <= '1';* in quanto cur_point_number viene incrementato in anticipo) e si pone next_state<=SET_ADDRESS;
- **DONE_HIGH:** stato nel quale si porta a 1 il segnale o_done. Finchè i_start rimane a 1, si rimane nello stato corrente (next_state<=DONE_HIGH), altrimenti si passa allo stato successivo (next_state<=DONE_LOW);
- **DONE_LOW:** stato nel quale si porta a 0 il segnale o_done e si ritorna nello stato iniziale (next_state<=START_WAIT);

Scelte progettuali

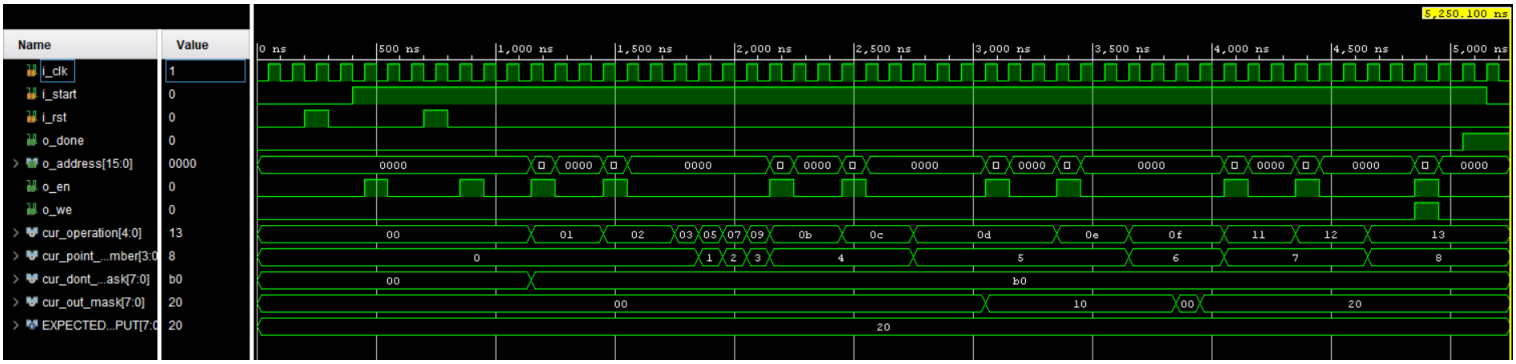
Il componente è stato descritto utilizzando due process:

- Il primo sensibile ai segnali i_clock e i_rst ha due funzioni:
  - se i_rst va a 1 si porta il componente nello stato iniziale START_WAIT;
  - se si ha un evento di clock (rising_edge(i_clk)) si aggiornano tutti i segnali cur_* con i corrispettivi valori next_*;
- Il secondo è sensibile a i_start, i_data e a tutti i segnali cur_*, permette di calcolare lo stato prossimo e il valore dei segnali di tipo next_*;

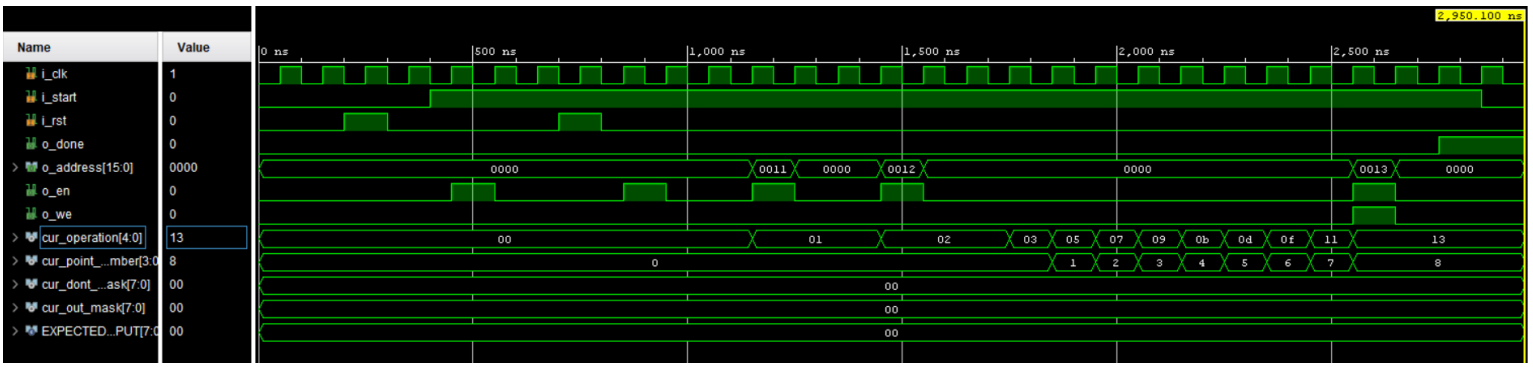
Descrizione dei casi di test

Il componente è stato sottoposto a due tipi di test:

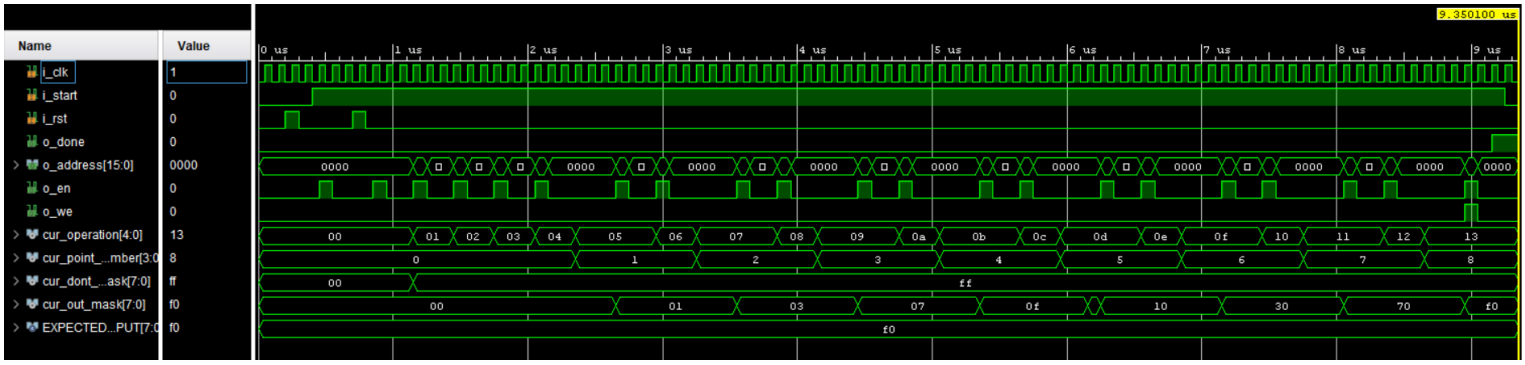
- **Test generati automaticamente:** mediante un programma scritto in C (in allegato) sono stati generati dei file di test-bench in modo da testare il componente in scenari d'uso "tipici". In particolare, è stato inserito un reset asincrono a 700ns, in modo da verificare se la macchina torni effettivamente nello stato di START_WAIT dopo un reset.
  Sono stati eseguiti 30 diversi test-bench, di seguito il diagramma temporale di uno di essi, in simulazione post sintesi funzionale:



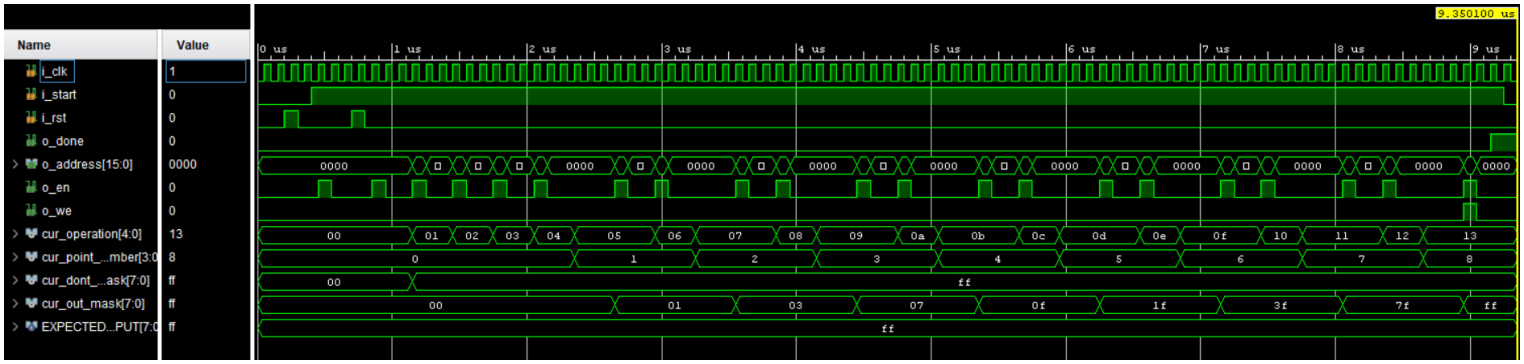
- **Test specifici:** sulla base di uno dei test generati automaticamente, sono stati scritti tre test-bench specifici per tre casi particolari:
  - **Maschera di don't care tutta a 0:** caso particolare in cui la maschera di don't care è tutta a 0, quindi nessun centroide deve essere considerato, di conseguenza il risultato atteso è out_mask tutta a 0.
    Di seguito il diagramma temporale in simulazione post sintesi funzionale:



- **Maschera di don’t care tutta a 1:** caso particolare in cui la maschera di don’t care è tutta a 1, quindi tutti i punti devono essere considerati.
  Tale caso testa in particolar modo l’accesso alla ram.
  Di seguito il diagramma temporale in simulazione post sintesi funzionale:



- **Tutti i punti equidistanti e maschera di don’t care tutta a 1:** caso particolare in cui la maschera di don’t care è tutta a 1, quindi tutti i punti devono essere considerati, inoltre i centroidi tutti equidistanti dal punto da valutare, quindi il risultato atteso è out_mask tutta a 1.
  Di seguito il diagramma temporale in simulazione post sintesi funzionale:



Eventuali ottimizzazioni

Una possibile ottimizzazione consiste nell’unione dei due stati CALC_DIST_1 e CALC_DIST_2 in un unico stato CALC_DIST, in modo da non utilizzare due cicli di clock, ma uno, riducendo il tempo di esecuzione. Tale modifica non è stata effettuata, poiché l’aggiornamento di next_out_mask sarebbe risultato troppo macchinoso e poco chiaro a livello di codice nel caso in cui cur_distance<cur_min_distance.

Non sono state rilevate altre ottimizzazioni.