Prova Finale di Reti Logiche

AA. 2019/2020

Stefano Dalla Longa, pID 10535602
Nicolò Brandolese, pID 10531144

Politecnico di Milano, scaglione prof. Fornaciari

Indice

1 Introduzione

This is the first section.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortisfacilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdietmi nec ante. Donec ullamcorper, felis non sodales...

2 Descrizione della macchina

L'implementazione si basa sulla scomposizione in tre sottoproblemi:

1. comunicazione con la RAM, intesa come l'insieme di operazioni di lettura e scrittura
2. computazione per determinare l'appartenenza dell'indirizzo base ad una working zone
3. gestione del flusso d'esecuzione in base al risultato della computazione

La scelta di risoluzione è stata la creazione di una macchina a stati finiti, in particolare una macchina di Mealy, basata su tre macroprocessi:

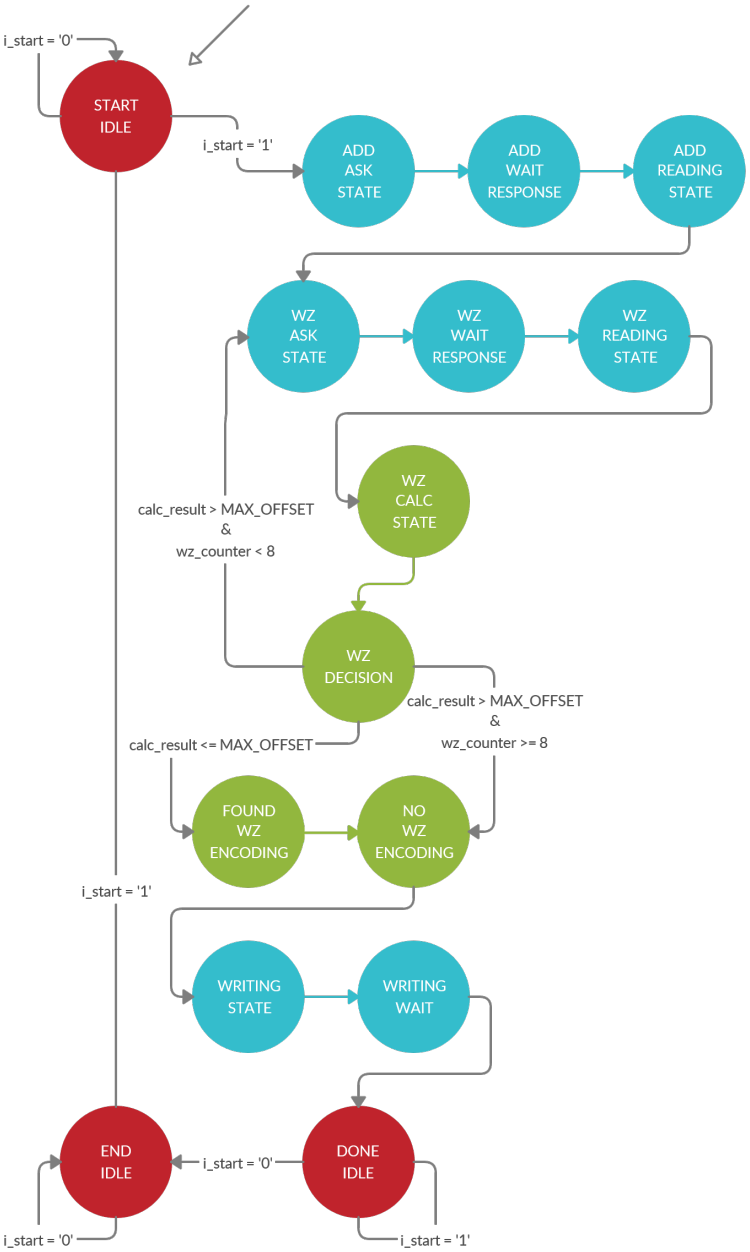
- *state_register*: ad ogni fronte di salita del clock propaga il valore del prossimo stato della macchina *next_state* al registro dello stato corrente *current_state*. Reagisce al segnale di *i_reset* eseguendo un reset sincrono, ovvero assegnando invece il valore di *START_IDLE* a *current_state* sul fronte di salita del clock.
- *speak_with_RAM*: gestisce i valori dei segnali in input e output, ad eccezione di *o_done* il quale è gestito da *calc_process*.
- *calc_process*: esegue i calcoli per determinare se l'indirizzo base appartiene alla working zone considerata

Al quale si aggiungono due ulteriori processi destinati alla gestione degli elementi di memoria interni:

- *wz_counter_process*: aggiorna sul fronte di discesa del clock il contatore *wz_counter* delle working zone già controllate tramite il segnale *count_add_sig*. Permette di resettare il suo contenuto a zero in caso di reset o di segnale di start portato a zero. L'aggiornamento sul fronte di discesa anziché di salita del clock risolve diversi problemi di sincronia tra il contatore e i successivi stati, dal momento che il suo valore deve disponibile allo stato immediatamente successivo all'aggiornamento del suo contenuto.
- *FF_saving*: gestisce i flip-flop relativi a quattro segnali da memorizzare. Non è stato necessario fornirlo di segnale di reset.

Abbiamo scelto di separare *calc_process* e *speak_with_RAM* per ragioni di leggibilità, dal momento che da un punto di vista teorico entrambi costituiscono una componente combinatoria della stessa macchina a stati finiti, e perciò sarebbe stato possibile accorparli nel medesimo processo. Dal momento però che VHDL proibisce di modificare lo stesso segnale da processi diversi, la gestione del flusso modificando il valore di *next_state* è affidata esclusivamente a *calc_process*.

Il funzionamento della macchina a stati è schematizzato nella figura seguente; gli stati in blu sono tutti e i soli stati nei quali opera *speak_with_RAM* e sono progettati per durare un ciclo di clock, gli stati in verde sono gli stati di calcolo e codifica da parte di *calc_process* anch'essi progettati per cambiare ad ogni ciclo di clock, mentre gli stati in rosso sono stati di idle gestiti da *calc_process*, progettati per arrestare il flusso di esecuzione ed eventualmente ricominciare la computazione in base al valore di *i_start*.



3 Test Eseguiti

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisissem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi necante...

4 Limitazioni e ottimizzazioni possibili

4.1 Hardcoding

Gli unici valori hardcodati sono dovuti al formato di codifica del risultato (work in progress)

4.2 Segnale di reset

Dal momento che la specifica non riporta alcun vincolo né assunzione sul segnale di reset `i_rst` e sulla sua durata, abbiamo deciso di implementare un resetting sincrono. In particolare, la macchina è in grado di riconoscere una richiesta di reset a prescindere da quanto a lungo il segnale `i_rst` è rimasto al valore logico 1, ma deve passare un intero ciclo di clock perché tale reset avvenga. Garantiamo comunque un

comportamento coerente con la richiesta di reset: in particolare, non è possibile per la macchina notificare alla RAM la fine dell'elaborazione tramite il segnale *o_done* nei cicli immediatamente successivi all'innalzamento di *i_rst*.

4.3 Ottimizzazione dei dati nella RAM

La macchina è progettata per produrre il risultato corretto anche se gli indirizzi relativi alle working zone contenuti nella RAM non sono in ordine (vedasi la sezione 3 relativa ai test eseguiti). La macchina infatti esegue una scansione lineare di tutte le working zone, decidendo per ogni working zone letta se richiedere la working zone successiva o fermarsi. Perciò, tra tutte le possibili configurazioni, il caso pessimo è quello in cui l'indirizzo base non appartiene a nessuna working zone, oppure quello in cui il base address appartiene alla working zone codificata in *RAM(7)*. Tuttavia, se venisse garantito che gli indirizzi delle working zone nella RAM siano in ordine crescente (decrescente), sarebbero possibili due modifiche:

- *Riduzione dei casi pessimi*: se l'indirizzo della working zone appena esaminata ha un valore più alto dell'indirizzo base, si può concludere che l'indirizzo base non appartiene a nessuna working zone, riducendo così il numero dei casi pessimi. Tale modifica al progetto è piuttosto modesta, dal momento che basterebbe modificare lo stato di *WZ_DECISION*.
- *Ricerca binaria*: anziché eseguire una ricerca lineare della working zone risulterebbe più efficiente eseguire una ricerca binaria. Tale strategia ridurrebbe l'intera complessità temporale asintotica della macchina: se n fosse il numero di working zone, si passerebbe da $T(n) \in \mathcal{O}(n)$ a $T(n) \in \mathcal{O}(\log n)$. Tuttavia, questa modifica è più onerosa di quella del punto precedente, dal momento che richiederebbe una modifica all'attuale metodo di scansione (affidato al semplice contatore di working zone *wz_counter*) e almeno un nuovo stato per il calcolo del prossimo indirizzo di memoria da richiedere.

5 Conclusioni

Quando non sai che cosa dire, è meglio se non dici nulla

no scherzo scriviamo che non abbiamo warning