Prova finale

(Progetto di Reti Logiche)

Anno Accademico 2019-2020

Professore: William Fornaciari

Riccardo De Santi

Luca Danelutti

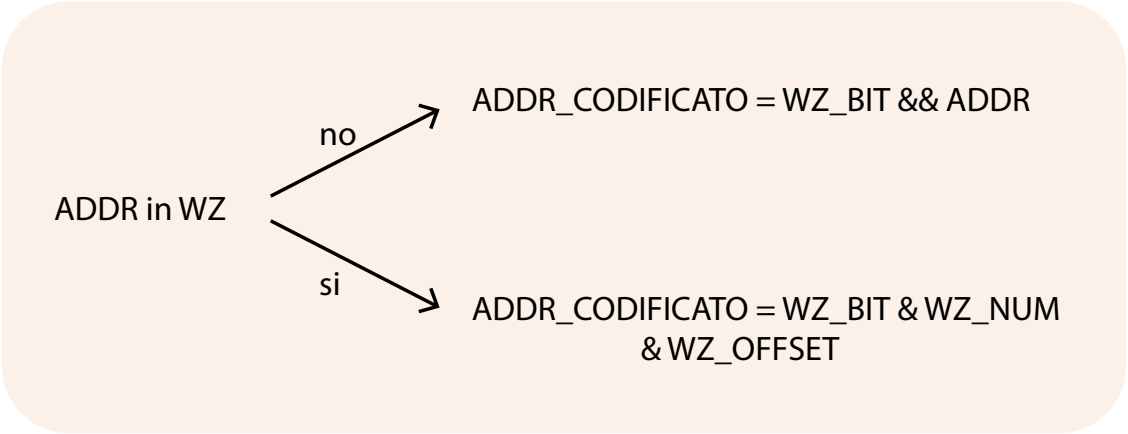
Indice

1. Introduzione

1.1 Il Problema

Il problema affrontato riguarda un metodo di codifica a bassa dissipazione denominato “Working Zone”. Esso consiste nel codificare un determinato indirizzo sulla base della sua appartenenza ad una Working Zone (WZ). Ogni WZ ha una dimensione di 4 indirizzi, di cui il primo appartiene ad un elenco dato.
Se l’indirizzo originale non corrisponde a nessun indirizzo di una WZ allora la sua codifica sarà **ADDR_CODIFICATO = WZ_BIT && ADDR.** Altrimenti la sua codifica sarà: **ADDR_CODIFICATO = WZ_BIT** & **WZ_NUM** & **WZ_OFFSET**.

Schema Riassuntivo:



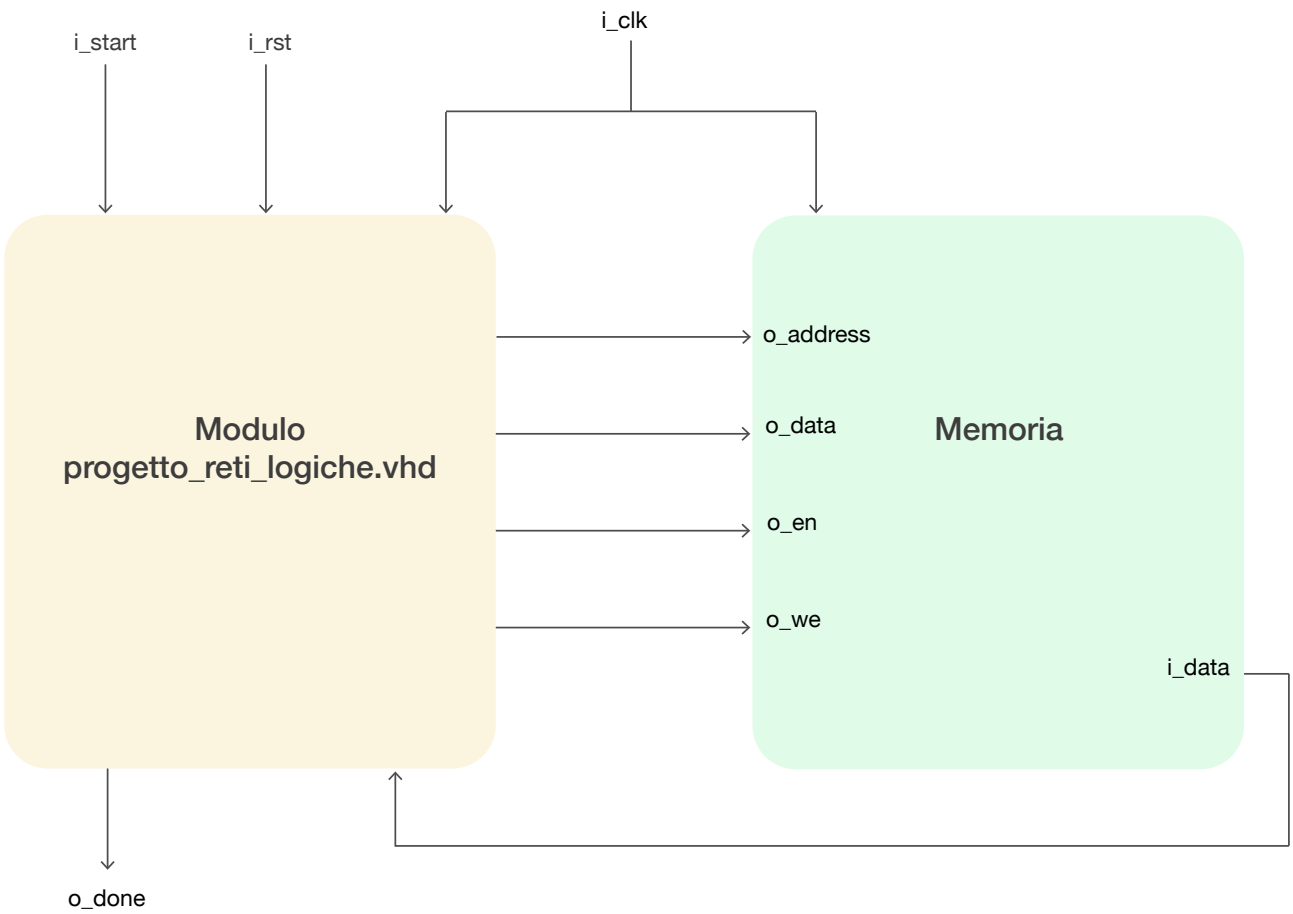
ADDR (7 bit): indirizzo originale da codificare, comprende numeri tra 0 e 127.
WZ_BIT (1 bit): vale 1 se l’indirizzo dato appartiene ad una working zone, 0 altrimenti
WZ_NUM (3 bit): numero della working zone alla quale appartiene ADDR, codificato in codifica binaria. Le working zone definite sono 8.
WZ_OFFSET (4 bit): offset dell’indirizzo della working zone individuata al quale corrisponde ADDR, codificato mediante codifica one-hot.

Il modulo codificato legge da una memoria esterna, descritta nella prossima sezione, l'indirizzo da codificare e quelli delle 8 working zone e produce un indirizzo codificato secondo le regole soprastanti.

1.2 Struttura della memoria

La memoria è caratterizzata da un indirizzamento al byte ed i dati sono disposti a partire dalla posizione 0, ciascuno è codificato su 8 bit. Pertanto le posizioni 0-7 contengono gli indirizzi base delle Working Zone, la posizione 8 contiene l'indirizzo da codificare, e quella successiva (9) è quella in cui viene scritto il valore dell'indirizzo codificato.

1.3 Schema d'interazione tra moduli



2. Aspetti Architetturali

Durante la realizzazione del componente si è cercato di individuare un algoritmo in grado di produrre un risultato nel minor tempo possibile, compatibilmente con i vincoli di lettura e scrittura della memoria.
Quindi si è optato per un componente mono-processo in cui è implementata tutta la logica necessaria. Il processo è eseguito ad ogni fronte di discesa del clock con la possibilità di reset asincrono. L'approccio di base dell'algoritmo implementato si basa su 3 fasi:

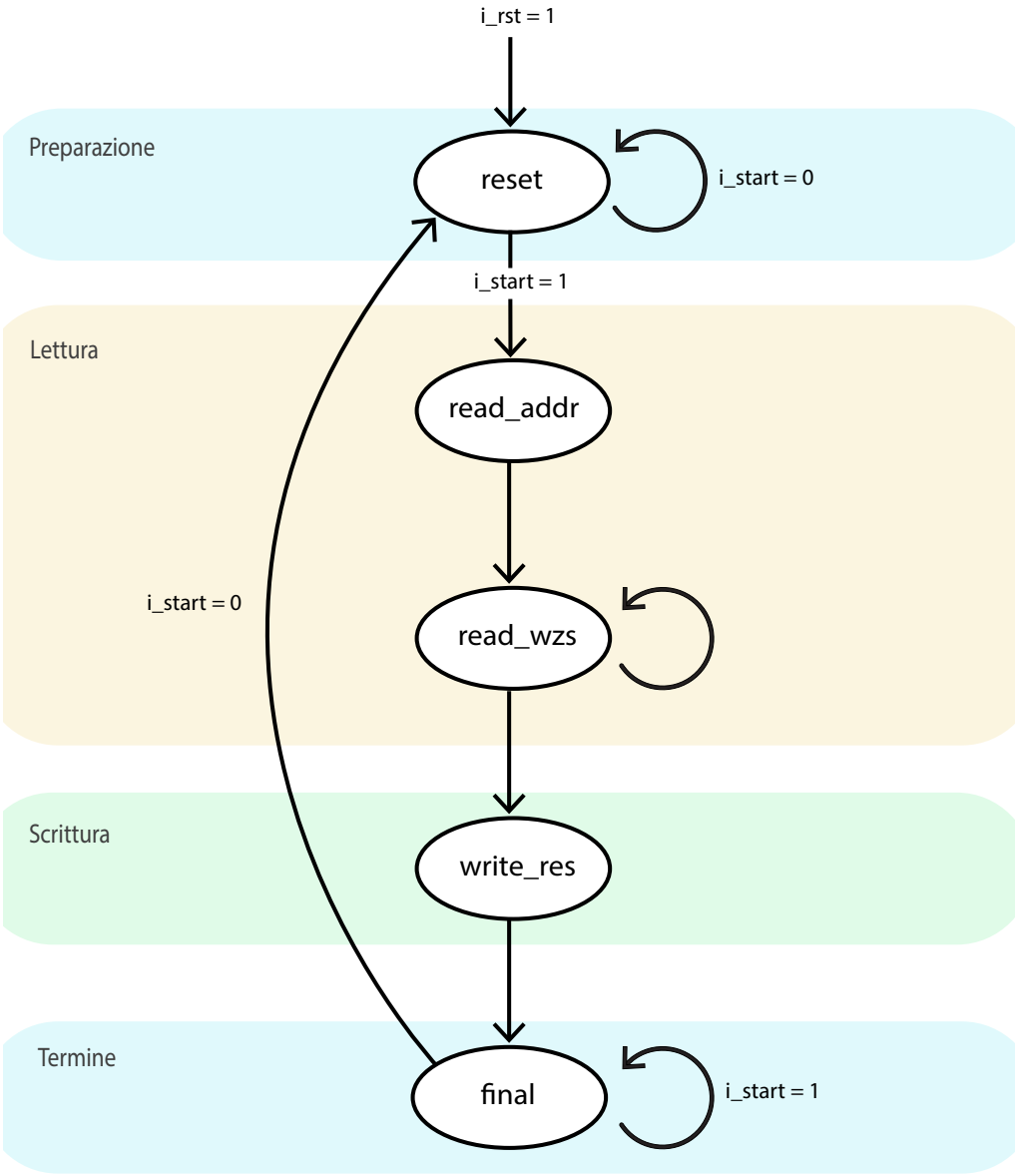
1. Lettura dell'indirizzo da codificare
2. Lettura e confronto tra l'indirizzo da codificare e ogni working zone
3. Scrittura del risultato

Per implementare il process abbiamo ritenuto opportuno introdurre 5 segnali aggiuntivi:

- *state*: rappresenta lo stato della FSM descritta più avanti
- *address_read*: memorizza l'indirizzo da codificare
- *curr_mem_pos*: rappresenta uno stato interno che consente di avere un solo stato per leggere tutte le working zone
- *wz_bit*: memorizza il **WZ_BIT** da generare in output secondo le specifiche
- *wz_offset*: rappresenta l'offset da generare in output calcolato rispetto all'indirizzo base della working zone

2.1 Macchina a stati

Abbiamo implementato nello specifico una FSM a 5 stati distinti: 3 stati per le 3 fasi principali dell'algoritmo e 2 stati ausiliari per la corretta gestione dei segnali di start e reset. Una rappresentazione dettagliata della macchina a stati è la seguente:



In particolare:

- *reset*: è lo stato di reset in cui si porta la macchina dopo aver ricevuto un segnale alto su *i_rst* oppure dopo ogni elaborazione eseguita. In questo stato viene controllato il segnale *i_start* e nel caso questo sia alto viene abilitata la lettura dalla memoria, predisposto l'indirizzo da leggere e portata la macchina in *read_addr*, così da iniziare l'elaborazione.

- *read_addr*: è lo stato in cui viene letto dalla memoria l'indirizzo da codificare, esso viene inoltre salvato in un registro in modo da essere accessibile per la computazione della fase 2.
- *read_wzs*: è lo stato in cui vengono letti, uno per ciclo di clock, gli indirizzi base delle working zones. Contemporaneamente viene determinata l'appartenenza o meno dell'indirizzo da codificare. In caso di appartenenza vengono impostati *wz_bit* e *wz_offset* e la FSM passa alla stato *write_res*, in caso contrario invece la macchina rimane in *read_wzs* e procede alla lettura e confronto della working zone successiva. Se sono già state lette tutte le working zone si procede comunque allo stato di *write_res* per scrivere il risultato secondo le regole esposte in precedenza.
- *write_res*: è lo stato in cui viene scritto in memoria il risultato della computazione. Contemporaneamente viene alzato il segnale *o_done*, la macchina quindi transita allo stato final.
- *final*: è la stato in cui la FSM rimane in attesa di un segnale basso su *i_start*. Una volta ricevuto vengono inizializzati i registri interni *wz_bit* e *wz_offset* ai valori di default e la macchina si riporta allo stato di reset diventando disponibile per una nuova computazione.

Come si evince dall'automa appena descritto il componente è stato sviluppato in modo da fornire all'utente una risposta appena questa è disponibile. Abbiamo quindi un tempo di codifica variabile come illustrato nella sezione *Analisi del tempo di conversione*.

3. Risultati della Sintesi

3.1 Area occupata

Sulla base della specifica del progetto è stata usata la *FPGA xc7a200tfbg484-1.* Nella tabella sottostante sono riportati i dati estratti dall'*Utilization Report* di *Vivado* relativamente alle risorse utilizzate (assolute e percentuali).

Risorsa	Utilizzo	Utilizzo in %
LookUp Table	50	0.04
Flip Flop	57	0.02

3.2 Analisi del tempo di conversione

Nel presente paragrafo intendiamo analizzare il tempo minimo e massimo di conversione di un dato indirizzo nella rispettiva codifica finale. L'unità di misura con la quale confronteremo tali valori è il ciclo di clock, in quanto il tempo effettivo dipende dalla durata del singolo clock.

Le scelte progettuali intraprese, in particolare l'assenza di uno stato iniziale durante il quale vengono salvate tutte le working zone, comporta un minor tempo minimo di conversione (T_{min}).
Nello specifico T_{min} = 8 cicli di clock viene ottenuto quando l'indirizzo iniziale appartiene all'insieme di indirizzi della prima working zone (**WZ_BIT** = 1 && **WZ_NUM** = 000)

Il tempo massimo di conversione (T_{max}) equivale a 15 cicli di clock e viene ottenuto nei seguenti due casi:

- L'indirizzo iniziale appartiene all'ultima working zone (**WZ_BIT** = 1 && **WZ_NUM** = 111).
- L'indirizzo iniziale non corrisponde ad alcun indirizzo appartenente ad una working zone (**WZ_BIT** = 0).

4. Test Bench e Risultati

Una volta sviluppato il componente siamo passati ad una fase di testing durante la quale abbiamo sollecitato il componente mediante diversi test bench con il fine di assicurarne la correttezza con la minima incertezza.
In particolare abbiamo usato due principali approcci:

- In primis abbiamo manualmente verificato la correttezza del comportamento in particolari condizioni critiche (boundary conditions).
- Dopo di che siamo passati ad una fase di testing mediante un test bench generato randomicamente con il fine di scoprire possibili anomalie non analizzate durante il test precedente.

Nello specifico alcune delle condizioni e funzioni che abbiamo testato sono le seguenti:

- Appartenenza a ognuno dei possibili indirizzi appartenenti ad ogni working zone.
- Non appartenenza ad alcuna working zone.
- Computazioni multiple (codifica di due indirizzi diversi in successione) con e senza segnale di reset intermedio.
- Corretta risposta al segnale reset per ogni stato della computazione.

Per quando riguarda il test con numeri randomici (generati tra valori possibili per l'input) il componente non ha riportato errori nonostante il numero elevato (decine di migliaia) di test al quale è stato sottoposto.

Pertanto possiamo concludere che il componente ha superato tutti i Test Bench da noi creati, sia in *Behavioural Simulation*, che in *Post-Synthesis Functional Simulation* e *Post-Synthesis Timing Simulation*.

5. Conclusioni

Il componente realizzato soddisfa pienamente i requisiti della specifica ed è in linea con le scelte progettuali introdotte, tra le quali la minimizzazione del tempo minimo di conversione comparando l'indirizzo originale dopo la lettura di ogni working zone. E' stato infatti realizzato un componente tale che:

- Risulta funzionante in pre-sintesi e post-sintesi.
- Dal testing si deduce che rispetta la specifica convertendo correttamente l'indirizzo dato.
- Rispetta il requirement relativo al periodo di clock ($\tau_{CK} > 100$ ns).