Prova Finale di Reti Logiche

Montanari Tommaso e Negri Riccardo

1 Aprile 2022

Docente:	Salice Fabio		
Studente 1:	Montanari Tommaso	10661941	932673
Studente 2:	Negri Riccardo	10729927	936820

1 Introduzione

La prova prevede l'implementazione in VHDL di una macchine che opera su una memoria e svolge la seguente operazione.
La macchina deve per prima cosa leggere il primo byte dalla memoria che identifica il numero di parole che sono state fornite come input, questa informazione è importante per capire quando la macchina deve terminare la lettura.
Dopodiché ogni parola successiva viene tradotta in due parole di memoria che vengono scritte progressivamente in un'altra parte della memoria.

1.1 Esempio

Indirizzo	Valore	Codifica binaria
0	2	0000 0010
1	35	0010 0011
2	161	1010 0001

Questo stato della memoria si traduce nell'input [35, 161] e in questo caso la lunghezza dell'input W=2 quindi mi aspetto una lunghezza dell'output Z=4.

Indirizzo	Valore	Codifica binaria
1000	13	0000 1101
1001	206	1100 1110
1002	97	0110 0001
1003	195	1100 0011

Rappresenta l'output [13, 206, 97, 195] dove [13, 206] sono i numeri che sono stati prodotti dal 35 in ingresso mentre [97, 195] sono ottenuti processando 161

1.2 Ipotesi Progettuali

- Si utilizza la scheda Artix-7 FPGA xc7a200tfbg484-1

- Ogni byte può contenere numeri da 0 a 255.
- La quantità di numeri in ingresso (W) è contenuta in una parola da un byte quindi anche il numero massimo di parole da tradurre è 255.
- Dato che l'input occupa al massimo 256 byte posso scrivere sui byte successivi quindi l'output parte sempre dal millesimo indirizzo di memoria che sicuramente non contiene l'input

2 Architetttura

Architetttura

3 Risultati sperimentali

3.1 Sintesi

Il componente sviluppato è sintetizzabile ed implementabile.

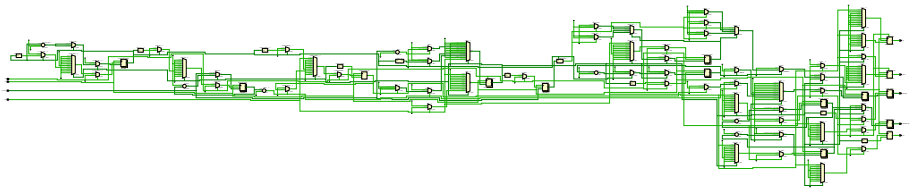

Figure 1: Schema post-sintesi

Nella seguente tabella sono indicati i componenti utilizzati nella sintesi.

Site Type	Used	Available	Utilization%
LUT as Logic	117	134600	0.09
Register as Flip Flop	95	269200	0.04
F7 Muxes	1	67300	<0.01

3.2 Simulazioni

Al fine di verificare il corretto funzionamento del componente sono stati eseguiti diversi test bench in simulazioni Behavioural, Post-synthesis functional e Post-synthesis timing. Di seguito sono riportati i test significativi effettuati con relativa spiegazione, tempi ottenuti e grafico d'onda.

3.2.1 Flussi successivi

Tramite questo test si verifica la correttezza nel caso di codifica di più flussi uno dopo l'altro. Nel testbench usato vengono codificati tre flussi (senza reset dopo ogni flusso).

Behavioural 21850 ns

Post-synthesis functional 22350100 ps

Post-synthesis timing 22353714 ps

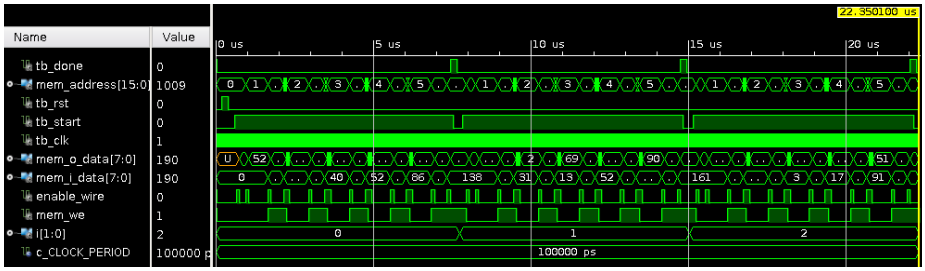

Figure 2: Post-synthesis functional simulation waveform

3.2.2 Reset asincrono

Tramite questo test si verifica il corretto funzionamento del reset asincrono.

Behavioural 10650 ns

Post-synthesis functional 10750100 ps

Post-synthesis timing 10753714 ps

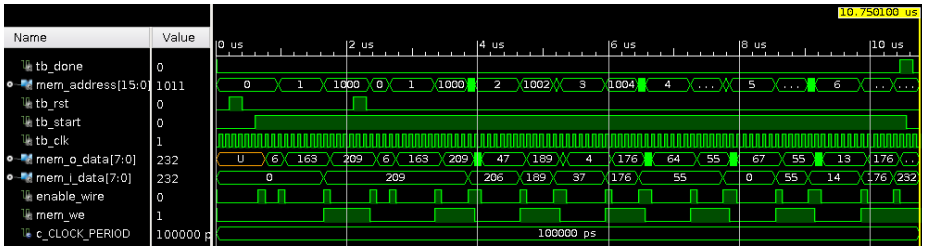

Figure 3: Post-synthesis functional simulation waveform

3.2.3 Sequenza di lunghezza massima

Tramite questo test si verifica il corretto funzionamento nel caso di sequenza di ingresso di lunghezza massima: 255 byte.

Behavioural 332450 ns

Post-synthesis functional 332550100 ps

Post-synthesis timing 332553714 ps

In questo caso non si riporta grafico della forma d'onda perché non è possibile distinguere i valori che i segnali assumono nella vista "fit".

3.2.4 Sequenza di lunghezza nulla

Tramite questo test si verifica il corretto funzionamento nel caso di sequenza di ingresso di lunghezza minima: 0 byte.

Behavioural 950 ns

Post-synthesis functional 1050100 ps

Post-synthesis timing 1053714 ps

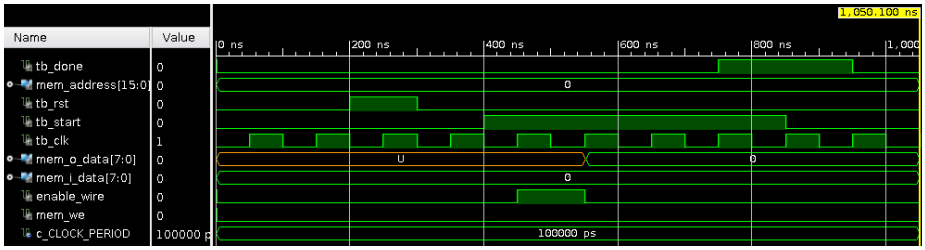

Figure 4: Post-synthesis functional simulation waveform

3.2.5 Double processing sulla stessa RAM

Tramite questo test si verifica il corretto funzionamento nel caso di scrittura sulla stessa RAM.

Behavioural 9250 ns

Post-synthesis functional 9550100 ps

Post-synthesis timing 9553714 ps

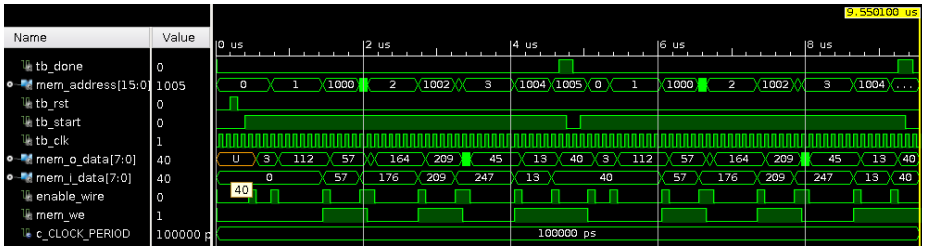

Figure 5: Post-synthesis functional simulation waveform

3.2.6 Tests con reset

Test bench, con reset dopo ogni test, che effettua 1000 test (generati casualmente con uno script Python).

Behavioural 172743250 ns

Post-synthesis functional 172943150100 ps

Post-synthesis timing 172943153714 ps

In questo caso non si riporta grafico della forma d'onda perché non è possibile distinguere i valori che i segnali assumono nella vista "fit".

3.2.7 Tests senza reset

Test bench, senza reset dopo ogni test, che effettua 1000 test (generati casualmente con uno script Python).

Behavioural 172543450 ns

Post-synthesis functional 172743350100 ps

Post-synthesis timing 172743353714 ps

In questo caso non si riporta grafico della forma d'onda perché non è possibile distinguere i valori che i segnali assumono nella vista "fit".

3.3 Osservazioni

Osservazioni su tempo di clock minimo.
Osservazioni su come reset non genera ritardi(?).

4 Conclusioni