Prova Finale Reti Logiche

Politecnico di Milano

POLITECNICO
MILANO 1863

Cardinale Simona
890171

Colli Stefano
891018

Anno Accademico 2019/2020

Indice

Capitolo 1

Introduzione

1.1 Scopo del progetto

Lo scopo del progetto è quello di creare un dispositivo hardware che implementi la tecnica di codifica "Working Zones", in modo da ridurre l'attività dei bus di indirizzo. Come si può dedurre dal nome attribuito, quest'ultima si basa sul concetto di Working Zones("WZ"), ovvero aree di memoria che vengono predilette dai programmi nel loro spazio di indirizzi. Il componente ideato ha l'obiettivo di determinare se un dato indirizzo appartenga o meno ad una WZ.

1.2 Specifiche generali

Lo schema di codifica preleva da una memoria 8 indirizzi base, uno per ciascuna Working Zone, e un indirizzo da codificare (per ipotesi assumeremo validi quelli da 0 a 127). Il principio guida di codifica sarà il seguente:

- Se l'indirizzo non appartiene a nessuna delle Working Zones, verrà trasmesso invariato;
- Se l'indirizzo appartiene ad una Working Zone, verrà rielaborato come concatenazione dei seguenti parametri: `WZ_BIT` & `WZ_NUMBER` & `WZ_OFFSET`.

  In tale rappresentazione `WZ_BIT` è un flag che se posto pari a 1 implica l'appartenenza dell'indirizzo ad una WZ; `WZ_NUMBER` rappresenta il numero della WZ individuata; `WZ_OFFSET` è l'offset, in codifica one-hot, rispetto all'indirizzo base di tale WZ.

1.3 Descrizione di memoria

Gli indirizzi, da 8 bit, sono allocati in una memoria RAM così rappresentata:

INDIRIZZO	CONTENUTO
0	INDIRIZZO BASE WZ 0
1	INDIRIZZO BASE WZ 1
2	INDIRIZZO BASE WZ 2
3	INDIRIZZO BASE WZ 3
4	INDIRIZZO BASE WZ 4
5	INDIRIZZO BASE WZ 5
6	INDIRIZZO BASE WZ 6
7	INDIRIZZO BASE WZ 7
8	INDIRIZZO DA CODIFICARE (ADDR)
9	INDIRIZZO CODIFICATO

Tabella 1.1: Schema della memoria RAM

1.4 Esempio esplicativo

Mostriamo il funzionamento richiesto mediante un piccolo esempio.

INDIRIZZO	CONTENUTO	CODIFICA BINARIA DEL CONTENUTO
0	2	00000010
1	10	00001010
2	25	00011001
3	36	00100100
4	55	00110111
5	72	01001000
6	87	01010111
7	93	01011101
8	58	00111010
9	200	11001000

Tabella 1.2: Rappresentazione della RAM

La tabella riporta la descrizione della memoria al seguito della codifica: l'indirizzo in analisi è "00111010"(58, contenuto in ottava posizione).
La sua codifica sarà quindi la seguente:

WZ_BIT	WZ_NUMBER	WZ_OFFSET
1	100	1000

Capitolo 2

Design

2.1 Architettura

Quando il segnale `I_START` viene portato a 1, il componente sviluppato inizierà l'elaborazione: l'indirizzo base, preso dalla RAM, verrà memorizzato in un registro e quindi dato a tre sommatori. Tale indirizzo e i suoi successivi incrementi verranno confrontati con l'indirizzo da codificare, mediante una batteria di xnor, ognuno dei quali restituirà 0 se sono uguali, 1 altrimenti. Leggendo, dal basso verso l'alto, gli output degli xnor si ottiene `WZ_OFFSET`
Questo verrà dato in input alla porta or, la quale fornirà l'esito circa l'appartenenza o meno dell'indirizzo ad una WZ.
Il contatore serve per tenere traccia di un eventuale `WZ_NUMBER` e per passare in rassegna tutti gli indirizzi della RAM, esso infatti viene incrementato ad ogni ciclo mediante un segnale di controllo. Il contatore parte da "1000", tale scelta implementativa è dovuta all'estrazione di "ADDR" dall'ottavo indirizzo di memoria.
Infine, il segnale `CONTROLLO_RISORSA` permette al multiplexer di effettuare una scelta tra l'indirizzo codificato e quello "inalterato".
Gli altri xnor sono stati introdotti per fornire alla macchina a stati importanti segnali per il controllo del circuito, così da prevenire eventuali errori dovuti al delay introdotto dai componenti di memoria.

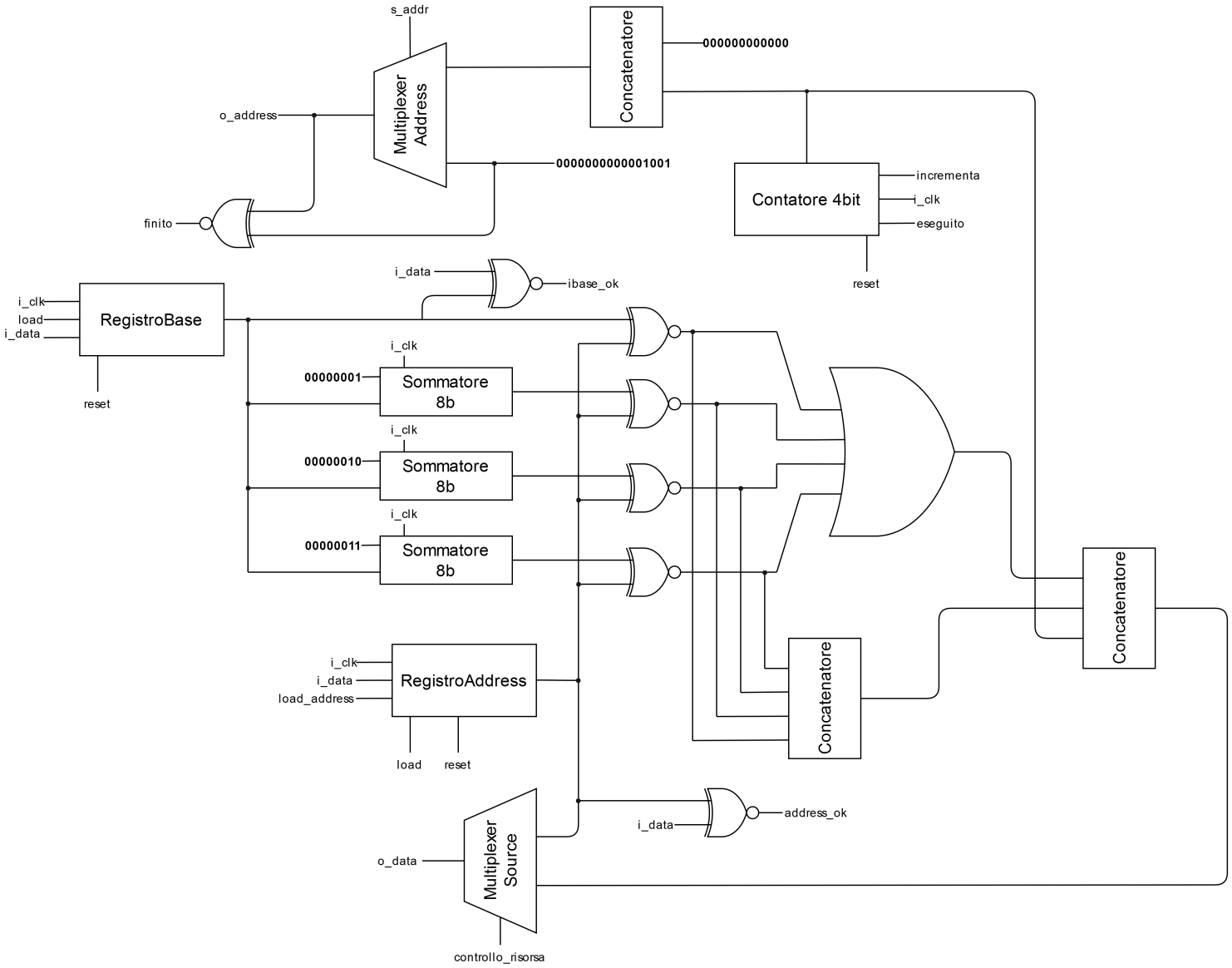

Figura 2.1: Schema circuitale

2.2 Macchina a stati

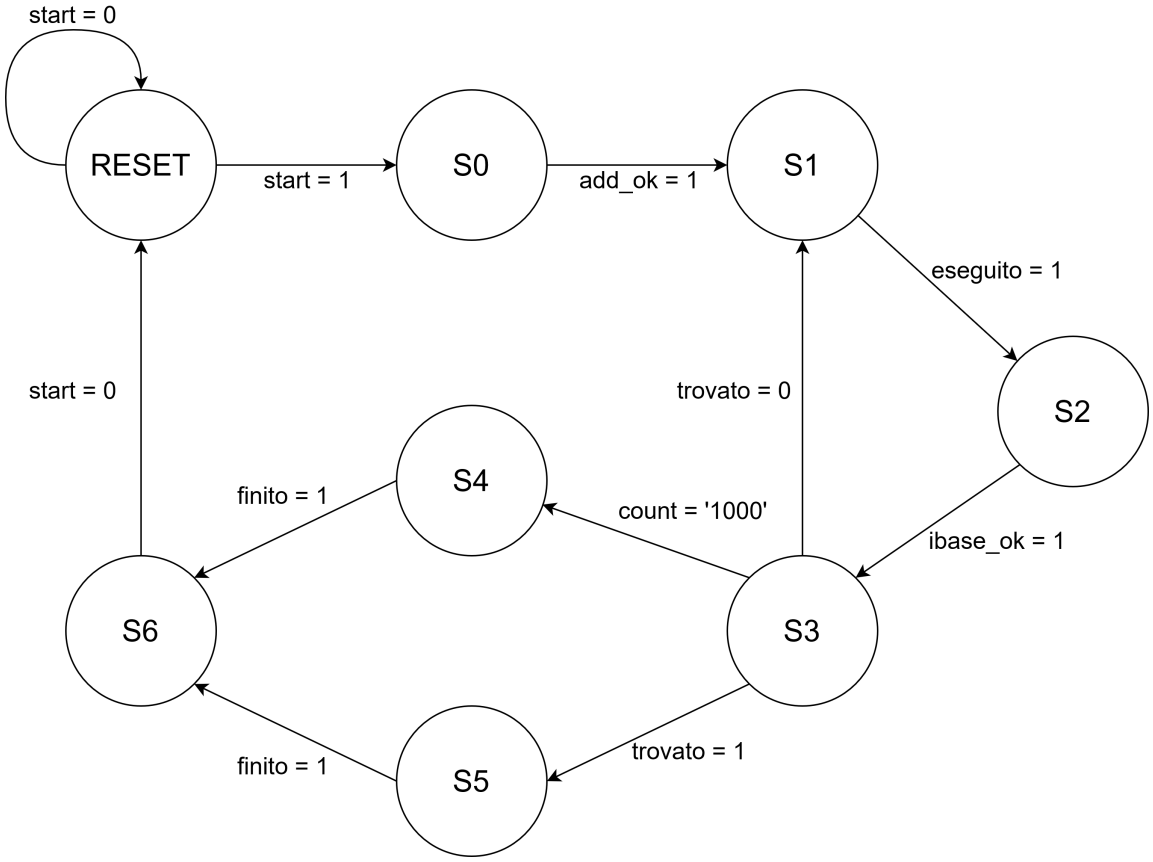

Figura 2.2: Macchina a stati

La macchina, atta a controllare il circuito, risulta composta da 8 stati. Al seguito si fornisce una breve descrizione per ciascuno di essi:

- STATO RESET: Stato iniziale in cui si attende l'evento `I_START=1`.
  Tutti i segnali di controllo sono sottoposti ad un ripristino al loro stato iniziale.
- STATO S0: Stato in cui l'indirizzo da codificare viene caricato in un registro.
  Si attivano i seguenti segnali di controllo:
  - ☐ `LOAD_ADD=1`;
  - ☐ `EN=1`;
- STATO S1: Stato in cui viene incrementato il contatore.
  Si attivano i seguenti segnali di controllo:

- ☐ INCREMENTA=1;
- ☐ LOAD_ADD=0;

- STATO S2: Stato in cui si carica nel registro il valore dell'indirizzo base preso in considerazione.

  Si attivano i seguenti segnali di controllo:

  - ☐ LOAD=1;
  - ☐ INCREMENTA=0;

- STATO S3: Stato in cui l'indirizzo da codificare viene confrontato con l'indirizzo base e con i rispettivi incrementi, l'esito viene fornito dal segnale TROVATO:

  -TROVATO=0 implica un ritorno in S1;

  -TROVATO=1 esegue lo stato S5.

  In tale stato si tiene, inoltre, in considerazione il contatore: se quest'ultimo è giunto all'ottavo ciclo di conteggio vuol dire, infatti, che l'indirizzo da codificare non appartiene a nessuna WZ.

  Si attivano i seguenti segnali di controllo:

  - ☐ LOAD=0;
  - ☐ EN=0;

- STATO S4: L'indirizzo da codificare, non appartenendo a nessuna WZ, viene memorizzato inalterato nel nono spazio della RAM.

  Si attivano i seguenti segnali di controllo:

  - ☐ WE=1;
  - ☐ EN=1;
  - ☐ S_ADD=1;
  - ☐ CONTROLLO_RISORSA=0;

- STATO S5: L'indirizzo, appartenendo ad una WZ, viene opportunatamente codificato e memorizzato in RAM.

  Si attivano i seguenti segnali di controllo:

  - ☐ WE=1;
  - ☐ EN=1;
  - ☐ S_ADD=1;
  - ☐ CONTROLLO_RISORSA=1;

- STATO S6: Il segnale FINITO segnala la fine della computazione e si pone O_DONE=0.

  Qualora I_START=0, si tornerebbe in RESET e la macchina sarebbe pronta ad effettuare un'eventuale e ulteriore elaborazione.

  Si attiva il seguente segnale di controllo:

  - ☐ O_DONE=1;

2.3 Scelte progettuali

La principale scelta progettuale effettuata è stata la minimizzazione del numero di registri necessari mediante un approccio ciclico.
Il componente, così strutturato, impiega infatti un minor numero di risorse e offre maggiore scalabilità.
Infatti, qualora si volesse aumentare il numero di WZ, mantenendo inalterata la loro dimensione, basterebbe aumentare il valore massimo del ciclo di conteggio del contatore.

Capitolo 3

Risultati sperimentali

3.1 Report di sintesi

In seguito alla fase di sintesi è stato automaticamente generato un report che fornisce utili informazioni circa il modulo hardware creato.
Rivestono un ruolo particolare le informazioni sottoelencate:

- Utilizzo risorse in termini di BRAM, DSP, FF, LUTs.

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	64	0	134600	0.05
LUT as Logic	64	0	134600	0.05
LUT as Memory	0	0	46200	0.00
Slice Registers	52	0	269200	0.02
Register as Flip Flop	52	0	269200	0.02
Register as Latch	0	0	269200	0.00
F7 Muxes	0	0	67300	0.00
F8 Muxes	0	0	33650	0.00

Figura 3.1: Report risorse

- Utilizzo dei componenti.

```
Detailed RTL Component Info :
+---Adders :
       2 Input      8 Bit       Adders := 3
       2 Input      4 Bit       Adders := 1
+---XORs :
       2 Input     16 Bit         XORs := 1
       2 Input      8 Bit         XORs := 6
+---Registers :
                    8 Bit    Registers := 5
                    4 Bit    Registers := 1
                    1 Bit    Registers := 1
+---Muxes :
       2 Input     16 Bit        Muxes := 1
       2 Input      8 Bit        Muxes := 10
       8 Input      8 Bit        Muxes := 1
       2 Input      4 Bit        Muxes := 1
       8 Input      1 Bit        Muxes := 3
```

Figura 3.2: Report componenti

Capitolo 4

Simulazioni

Per verificare il corretto funzionamento del componente sono stati effettuati le seguenti simulazioni:

- Test bench che verifica l'appartenenza di un indirizzo ad una WZ e relativa codifica.

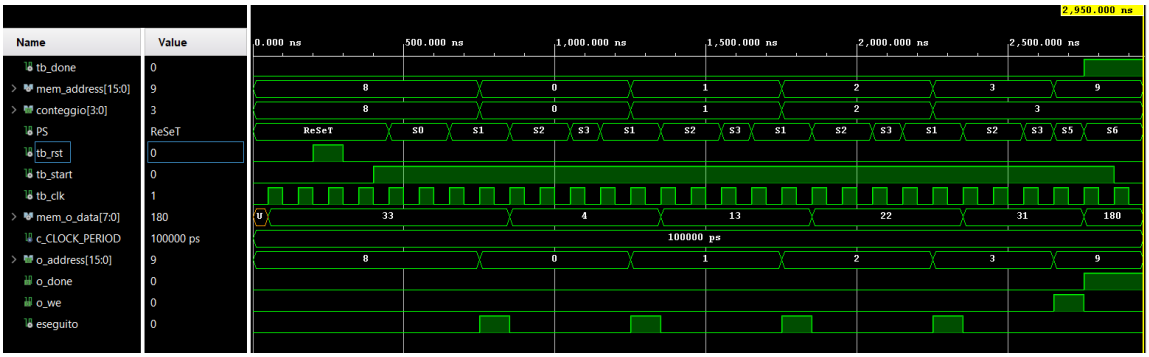

Figura 4.1: L'indirizzo appartiene alla WZ

- Test bench che verifica la non appartenenza di un indirizzo ad alcuna WZ. Il suddetto indirizzo viene memorizzato inalterato alla RAM.

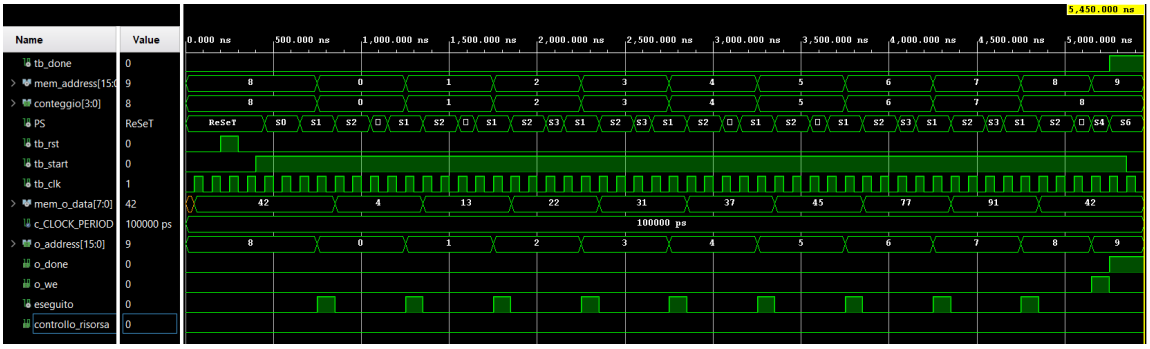

Figura 4.2: L'indirizzo non appartiene alla WZ

Inoltre sono stati effettuati, in modo totalmente automatizzato, altri 200 000 test generati casualmente attraverso un software scritto in linguaggio C.
Alcuni di essi analizzano la casistica in cui vengono forniti al modello diversi indirizzi da codificare senza effettuare un reset completo dell'intero circuito tra una computazione e l'altra.
Infine, per assicurare un'ulteriore copertura, sono stati effettuati dei test che indirizzano la simulazione verso i corner case.
Al seguito si riportano i casi più significativi:

- Test bench per cui l'indirizzo da codificare è "00000000".

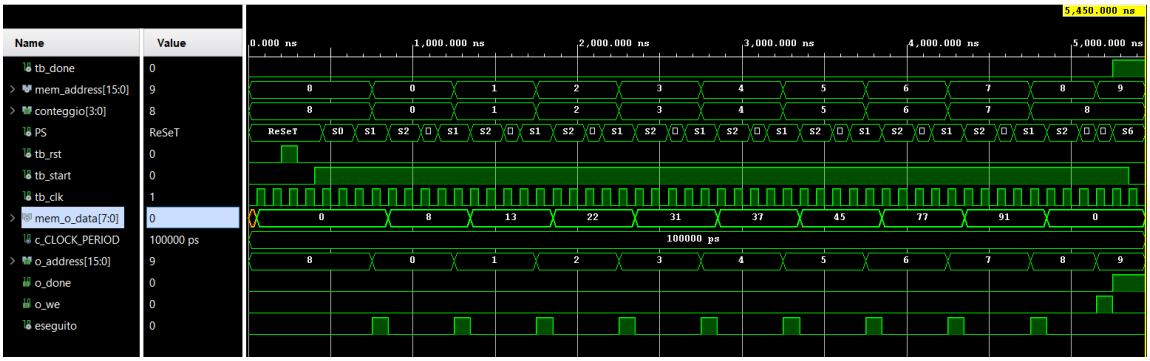

Figura 4.3: L'indirizzo 00000000 non appartiene alla WZ

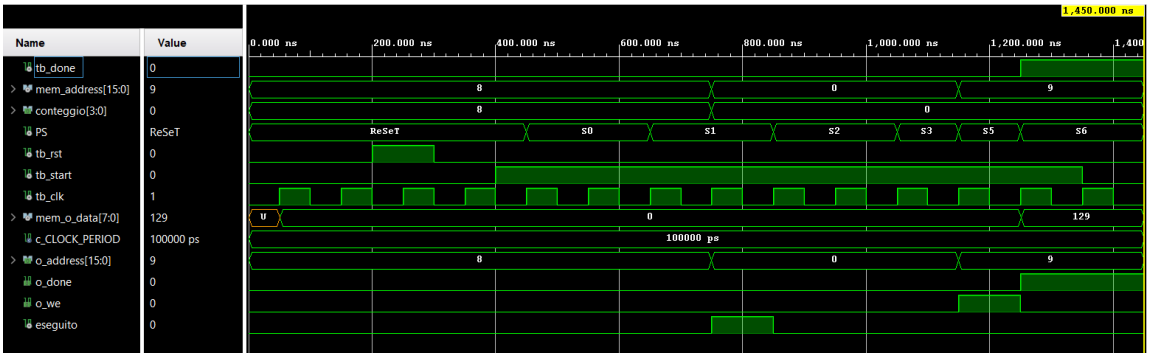

Figura 4.4: L'indirizzo 00000000 appartiene alla WZ

- Test bench per cui l'indirizzo da codificare è 11111111

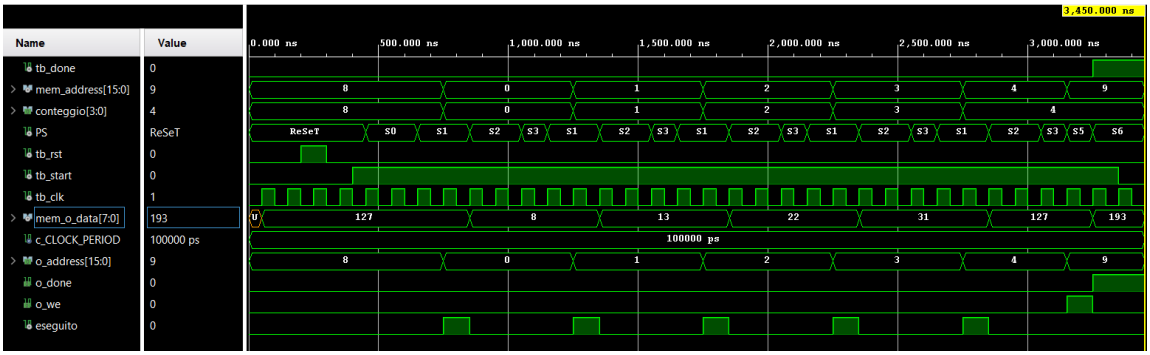

Figura 4.5: L'indirizzo 127 appartiene alla quarta WZ

- Test bench per cui l'indirizzo da codificare appartiene all'ultima WZ.

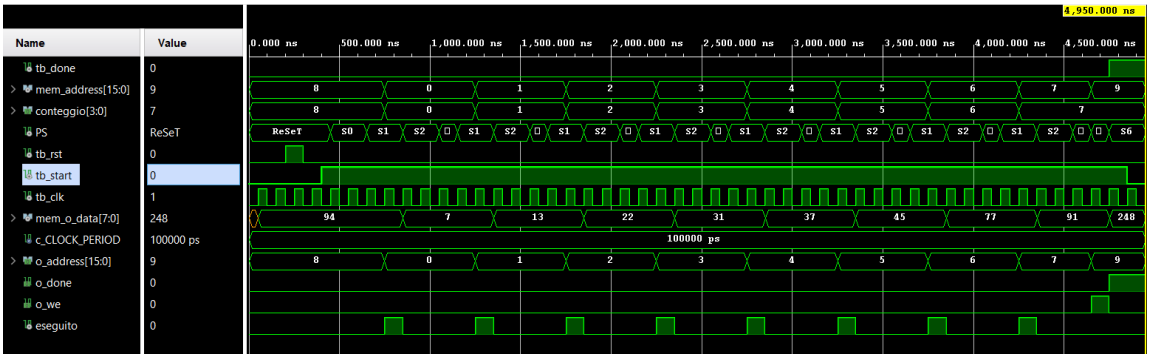

Figura 4.6: L'indirizzo appartiene all'ultima WZ

Capitolo 5

Considerazioni finali

I test bench sovraelencati sono stati effettuati e correttamente simulati dal componente in: *Behaviural* e *Post-Synthesis Fuctional*.
Inoltre, per maggiore completezza, verrano riportati al seguito i tempi di simulazione nelle quattro principali casistiche :

- Tempo di simulazione nel caso in cui l'indirizzo appartenga alla prima WZ: 1.45 µs.
- Tempo di simulazione nel caso in cui l'indirizzo appartenga ad una WZ (la quarta, nel suddetto caso): 2.95 µs.
- Tempo di simulazione nel caso in cui l'indirizzo appartenga all'ultima WZ: 4.95 µs.
- Tempo di simulazione nel caso in cui l'indirizzo non appartenga ad una WZ: 5.45 µs.

Dalle tempistiche sopra riportate emerge che la velocità del componente nella traduzione dell'indirizzo dipende dal numero della WZ di appartenenza: il tempo minimo è di 1.45 µs quello massimo di 4.95 µs.