Progetto di Reti Logiche AA 2019-2020
Prof. PALERMO

Professore: Gianluca Palermo

Professore di supporto: Antonio Rosario Miele

Il seguente progetto è stato eseguito in gruppo.

I componenti del gruppo sono: - **Simeoni Raffaele** codice persona: 10527107

- **Vangi Gabriele** codice persona: 10567226

SPECIFICA DEL PROGETTO

Descrizione generale

La specifica del progetto è ispirata al metodo di codifica a bassa dissipazione di potenza denominato "Working Zone". Il metodo di codifica Working Zone si usa per trasformare il valore di un indirizzo quando questo viene trasmesso, se appartiene a certi intervalli (detti appunto working-zone).
Una working-zone è definita come un intervallo di indirizzi di dimensione fissa (Dwz) che parte da un indirizzo base. All'interno dello schema di codifica possono esistere multiple working-zone (Nwz).
Lo schema modificato di codifica da implementare è il seguente:

- se l'indirizzo da trasmettere (ADDR) non appartiene a nessuna Working Zone, esso viene trasmesso così come è, e un bit addizionale rispetto ai bit di indirizzamento (WZ_BIT) viene messo a 0. In pratica dato ADDR, verrà trasmesso WZ_BIT=0 concatenato ad ADDR (WZ_BIT & ADDR, dove & è il simbolo di concatenazione);
- se l'indirizzo da trasmettere (ADDR) appartiene ad una Working Zone, il bit addizionale WZ_BIT è posto a 1, mentre i bit di indirizzo vengono divisi in 2 sotto campi rappresentanti:
    - Il numero della working-zone al quale l'indirizzo appartiene WZ_NUM, che sarà codificato in binario
    - L'offset rispetto all'indirizzo di base della working zone WZ_OFFSET, codificato come one-hot

Codifica one hot per gli offset da 0 a 3:

- WZ_OFFSET = 0 è codificato one hot come 0001;
- WZ_OFFSET = 1 è codificato one hot come 0010;
- WZ_OFFSET = 2 è codificato one hot come 0100;
- WZ_OFFSET = 3 è codificato one hot come 1000;

Dati

I dati ciascuno di dimensione 8 bit sono memorizzati in una memoria con indirizzamento al Byte partendo dalla posizione 0. Anche l'indirizzo che è da specifica di 7 bit viene memorizzato su 8 bit. Il valore dell'ottavo bit sarà sempre zero.
- Le posizioni in memoria da 0 a 7 sono usate per memorizzare gli otto indirizzi base delle working-zone:
- 0 - Indirizzo Base WZ 0
- ...
- 7 - Indirizzo Base WZ 7
- La posizione in memoria 8 avrà al suo interno il valore (indirizzo) da codificare (ADDR);
- La posizione in memoria 9 è quella che deve essere usata per scrivere, alla fine, il valore codificato secondo le regole precedenti.

Note ulteriori sulla specifica

Il modulo partirà nella elaborazione quando un segnale START in ingresso verrà portato a 1. Il segnale di START rimarrà alto fino a che il segnale di DONE non verrà portato alto; Al termine della computazione (e una volta scritto il risultato in memoria), il modulo da progettare deve alzare (portare a 1) il segnale DONE che notifica la fine dell'elaborazione. Il segnale DONE deve rimanere alto fino a che il segnale di START non è riportato a 0. Un nuovo segnale start non può essere dato fin tanto che DONE non è stato riportato a zero. Se a questo punto viene rialzato il segnale di START, il modulo dovrà ripartire con la fase di codifica.

Interfaccia del Componente

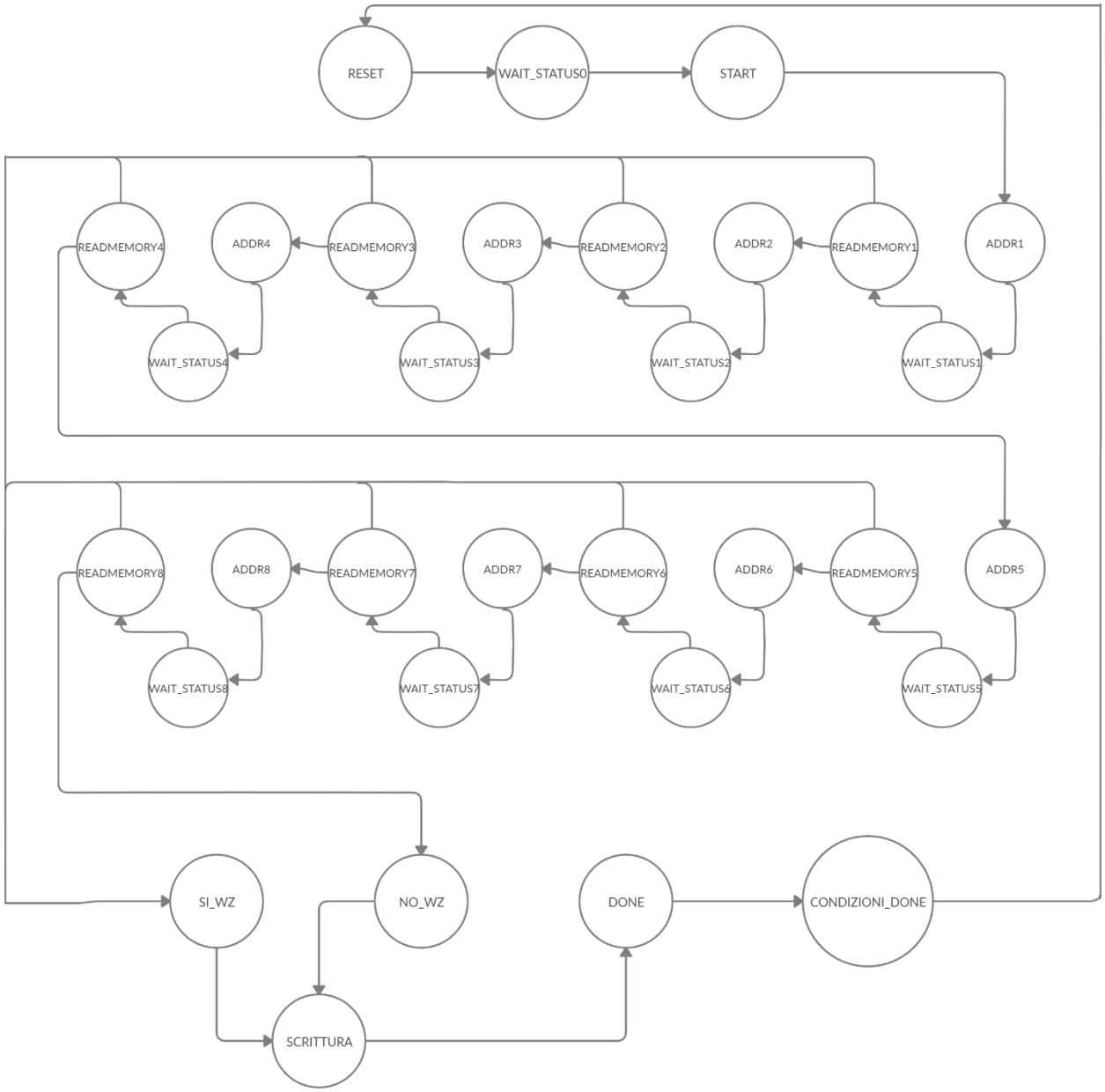
Il componente da descrivere deve avere la seguente interfaccia.

```
entity project_reti_logiche is
        port (
                i_clk : in std_logic;
                i_start : in std_logic;
                i_rst : in std_logic;
                i_data : in std_logic_vector(7 downto 0);
                o_address : out std_logic_vector(15 downto 0);
                o_done : out std_logic;
                o_en : out std_logic;
                o_we : out std_logic;
                o_data : out std_logic_vector (7 downto 0);
        );
end project_reti_logiche;
```

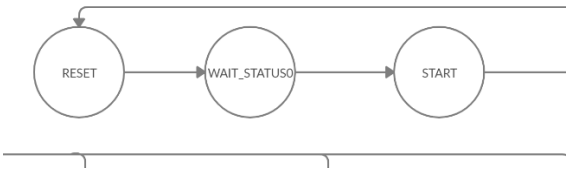
In particolare:

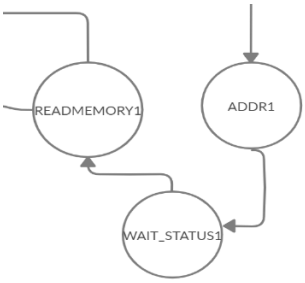
- i_clk è il segnale di CLOCK in ingresso generato dal TestBench;
- i_start è il segnale di START generato dal Test Bench;
- i_rst è il segnale di RESET che inizializza la macchina pronta per ricevere il primo segnale di START;
- i_data è il segnale (vettore) che arriva dalla memoria in seguito ad una richiesta di lettura;
- o_address è il segnale (vettore) di uscita che manda l'indirizzo alla memoria;
- o_done è il segnale di uscita che comunica la fine dell'elaborazione e il dato di uscita scritto in memoria;
- o_en è il segnale di ENABLE da dover mandare alla memoria per poter comunicare (sia in lettura che in scrittura);
- o_we è il segnale di WRITE ENABLE da dover mandare alla memoria (=1) per poter scriverci. Per leggere da memoria esso deve essere 0;
- o_data è il segnale (vettore) di uscita dal componente verso la memoria.

ARCHITETTURA e FUNZIONAMENTO



Inizialmente il modulo si trova nello stato RESET, nel quale resterà fino a quando il segnale START in ingresso verrà portato a 1, a quel punto i Il modulo partirà nella elaborazione leggendo l'indirizzo da codificare dall'indirizzo di memoria prescelto. Lo stato WAIT_STATUS0, nonostante non compia alcuna operazione, serve a far passare un ciclo di clock, permettendo al modulo la corretta lettura dalla memoria. Nello stato START l'indirizzo letto viene salvato in una variabile e si inizia con la codifica. Il segnale di START rimarrà alto fino a che il segnale di DONE non verrà portato alto.



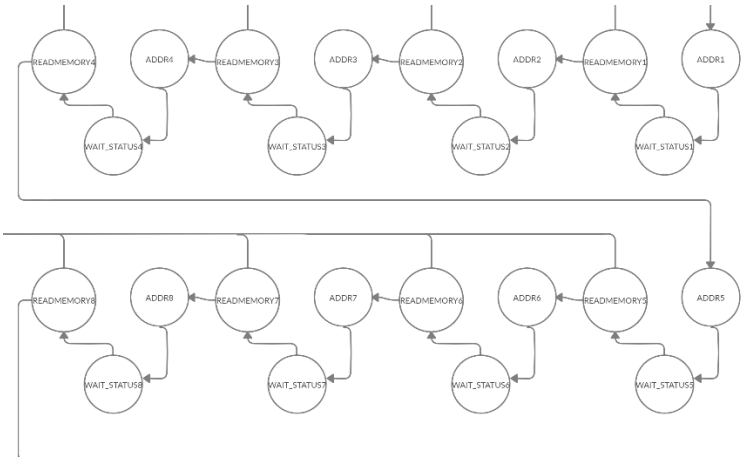


Sono presenti 8 blocchi composti da questi tre stati, ognuno per una Working Zone, i quali eseguono i seguenti compiti:

- ADDRn: legge l'indirizzo della n-esima Working Zone dalla memoria

-WAIT_STATUSn: fa passare un ciclo di clock per la corretta lettura dell'indirizzo

-READMEMORYn: confronta l'indirizzo da codificare (quello salvato durante lo stato START) con quello appena letto (e i suoi tre indirizzi successivi). Se trova un riscontro ricostruisce wz_NUM in base al numero n del blocco e wz_OFFSET in base a quale indirizzo ha ottenuto il riscontro, li salva in due variabili e salta allo stato SI_WZ. Se NON trova un riscontro passa al blocco successivo.



Alla fine di questo processo se l'indirizzo non è stato associato a nessuna Working Zone ci troveremo nello stato NO_WZ. Strutturare il modulo in questo modo gli permette di saltare subito allo stato SI_WZ in caso di riscontro, evitando di passare per stati inutili. Il cammino massimo si ha infatti solo nel caso in cui l'indirizzo da codificare non sia presente in nessuna Working Zone.

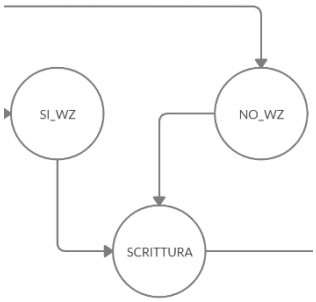
SI_WZ= In questo stato il bit addizionale WZ_BIT è posto a 1, e viene concatenato con le due variabili inizializzate nello stato READMEMORYn che ha ottenuto riscontro positivo, contenenti:

- Il numero della Working-zone al quale l'indirizzo appartiene WZ_NUM, codificato in binario.
- L'offset rispetto all'indirizzo di base della Working zone WZ_OFFSET, codificato come one-hot.

E viene poi inviato all'indirizzo di memoria indicato.

NO_WZ= In questo stato, l'indirizzo viene trasmesso così come è, e un bit addizionale rispetto ai bit di indirizzamento (WZ_BIT) viene messo a 0

Indipendentemente dal fatto che ci si trovi nell'uno o nell' altro stato, si passerà allo stato **Scrittura** nel quale si aspetta il clock affinché la scrittura sia completata.



Terminata la fase di scrittura si passa nello stato **Done** dove il segnale di done viene portato a 1.

Dopodichè si entra nello stato **Condizione_Done** dove si aspetta che il segnale di start scenda a 0 per poter riportare il segnale DONE a 0 e tornare nello stato di Reset così che la macchina sia pronta ad effettuare un'altra elaborazione.

RISULTATI SPERIMENTALI

- REPORT DI SINTESI:

```
---------------------------------------------------------------------------------------------------------------------------------------------------------------
Detailed RTL Component Info :
+---Adders :
          2 Input      9 Bit       Adders := 3
+---Registers :
                      16 Bit    Registers := 1
                       8 Bit    Registers := 2
                       4 Bit    Registers := 1
                       3 Bit    Registers := 1
                       1 Bit    Registers := 3
+---Muxes :
         32 Input      8 Bit        Muxes := 1
         32 Input      5 Bit        Muxes := 1
         66 Input      5 Bit        Muxes := 1
          4 Input      4 Bit        Muxes := 1
          2 Input      3 Bit        Muxes := 3
         32 Input      3 Bit        Muxes := 1
          2 Input      1 Bit        Muxes := 3
         32 Input      1 Bit        Muxes := 10
---------------------------------------------------------------------------------------------------------------------------------------------------------------
```

```
---------------------------------------------------------------------------------------------------------------------------------------------------------------
Report Check Netlist:
+------+------------------+-------+---------+-------+------------------+
|      |Item              |Errors |Warnings |Status |Description       |
+------+------------------+-------+---------+-------+------------------+
|1     |multi_driven_nets |      0|        0|Passed |Multi driven nets |
+------+------------------+-------+---------+-------+------------------+
---------------------------------------------------------------------------------------------------------------------------------------------------------------
```

Report Cell Usage:

	Cell	Count
1	BUFG	1
2	CARRY4	3
3	LUT1	3
4	LUT2	10
5	LUT3	14
6	LUT4	10
7	LUT5	4
8	LUT6	26
9	FDCE	5
10	FDRE	30
11	IBUF	11
12	OBUF	27

Report Instance Areas:

	Instance	Module	Cells
1	top		144

Slice Logic

Site Type	Used	Fixed	Available	Util%
Slice LUTs*	53	0	134600	0.04
LUT as Logic	53	0	134600	0.04
LUT as Memory	0	0	46200	0.00
Slice Registers	35	0	269200	0.01
Register as Flip Flop	35	0	269200	0.01
Register as Latch	0	0	269200	0.00
F7 Muxes	0	0	67300	0.00
F8 Muxes	0	0	33650	0.00

Memory

Site Type	Used	Fixed	Available	Util%
Block RAM Tile	0	0	365	0.00
RAMB36/FIFO*	0	0	365	0.00
RAMB18	0	0	730	0.00

DSP

Site Type	Used	Fixed	Available	Util%
DSPs	0	0	740	0.00

IO and GT Specific

Site Type	Used	Fixed	Available	Util%
Bonded IOB	38	0	285	13.33
Bonded IPADs	0	0	14	0.00
Bonded OPADs	0	0	8	0.00
PHY_CONTROL	0	0	10	0.00
PHASER_REF	0	0	10	0.00
OUT_FIFO	0	0	40	0.00
IN_FIFO	0	0	40	0.00
IDELAYCTRL	0	0	10	0.00
IBUFDS	0	0	274	0.00
GTPE2_CHANNEL	0	0	4	0.00
PHASER_OUT/PHASER_OUT_PHY	0	0	40	0.00
PHASER_IN/PHASER_IN_PHY	0	0	40	0.00
IDELAYE2/IDELAYE2_FINEDELAY	0	0	500	0.00
IBUFDS_GTE2	0	0	2	0.00
ILOGIC	0	0	285	0.00
OLOGIC	0	0	285	0.00

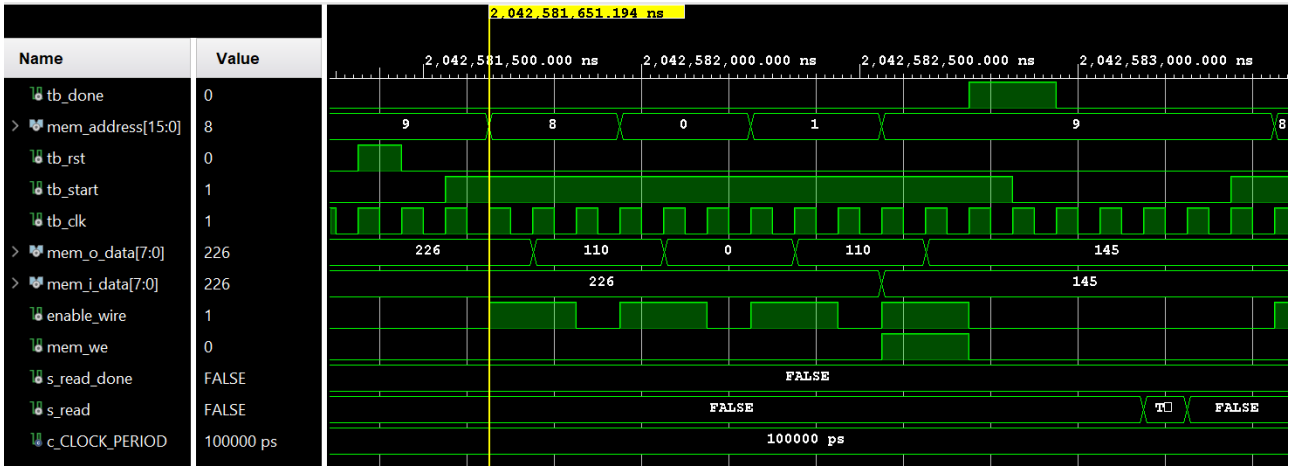
Clocking

Site Type	Used	Fixed	Available	Util%
BUFGCTRL	1	0	32	3.13
BUFIO	0	0	40	0.00
MMCME2_ADV	0	0	10	0.00
PLLE2_ADV	0	0	10	0.00
BUFMRCE	0	0	20	0.00
BUFHCE	0	0	120	0.00
BUFR	0	0	40	0.00

Primitives

Ref Name	Used	Functional Category
FDRE	30	Flop & Latch
OBUF	27	IO
LUT6	26	LUT
LUT3	14	LUT
IBUF	11	IO
LUT4	10	LUT
LUT2	10	LUT
FDCE	5	Flop & Latch
LUT5	4	LUT
LUT1	3	LUT
CARRY4	3	CarryLogic
BUFG	1	Clock

SIMULAZIONI E TEST BENCH



Sono stati eseguiti numerosi test bench per verificare che il modulo implementato soddisfasse le specifiche richieste, in particolare:

-MULTISTART: una volta codificato l'indirizzo, qualora il modulo riceva un altro segnale di START , sia in grado di ripartire in una nuova fase di codifica di un nuovo indirizzo.

-INDIRIZZO PRESENTE: nel caso in cui l'indirizzo sia contenuto in una Working Zone, questo sia codificato in modo corretto e il modulo non proceda inutilmente nelle successive WZ. (come l'esempio dell'immagine qui sopra riportata)

-INDIRIZZO NON PRESENTE: nel caso in cui l'indirizzo non sia contenuto in una Working Zone, questo sia codificato in modo corretto.

-MULTISTART CON RESET: verifica che il modulo funzioni correttamente anche nel caso in cui debba ripartire in una nuova codifica, ma con la presenza di un segnale di RESET. (come l'esempio dell'immagine qui sopra riportata. In corrispondenza della linea gialla riparte una nuova codifica)

-RESET ASINCRONO: verifica che il componente, nel caso in cui riceva un segnale RESET asincrono, continui a funzionare correttamente.

I test bench sono stati eseguiti sia pre che post-sintesi.

CONSLUSIONI:

Abbiamo deciso di implementare un modulo che tenesse in locale meno informazioni possibili e che dialogasse con la memoria chiedendole solo le informazioni necessarie per passare al prossimo stato;

cioè un modulo che evitasse di prelevare dalla memoria tutti gli indirizzi delle Working Zone senza che ce ne fosse effettivamente bisogno.