Ingegneria Informatica
Dipartimento di Elettronica, Informazione e Bioingegneria
Politecnico di Milano

Progetto Finale di Reti Logiche

Riccardo Paltrinieri
Matricola: 10626923
Professore: Gianluca Palermo

01/04/2020

Indice

1. Specifiche di progetto

La specifica del Progetto di Reti Logiche (Prova finale) 2019 è ispirata al metodo di codifica a bassa dissipazione di potenza denominato "Working Zone"[1].
Il metodo di codifica Working Zone è un metodo pensato per il Bus Indirizzi che si usa per trasformare il valore di un indirizzo quando questo viene trasmesso, se appartiene a certi intervalli (detti appunto working-zone). In questo caso il componente progettato invia al Bus solo l'identificativo della working-zone a cui appartiene e il valore dell'offset codificato come one-hot.

1.1 Richieste

Nella versione da implementare il numero di bit da considerare per l'indirizzo da codificare è 7. Il che definisce come indirizzi validi quelli da 0 a 127. Il numero di working-zone è 8 (Nwz=8) mentre la dimensione della working-zone è 4 indirizzi incluso quello base (Dwz=4).
Questo comporta che l'indirizzo codificato sarà composto da 8 bit: 1 bit per WZ_BIT + 7 bit per ADDR, oppure 1 bit per WZ_BIT, 3 bit per codificare in binario a quale tra le 8 working zone l'indirizzo appartiene, e 4 bit per codificare one hot il valore dell'offset di ADDR rispetto all'indirizzo base.
Il modulo da implementare leggerà l'indirizzo da codificare e gli 8 indirizzi base delle working-zone e dovrà produrre l'indirizzo opportunamente codificato.

1.2 Specifica dei componenti

Il componente descritto ha 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;
```

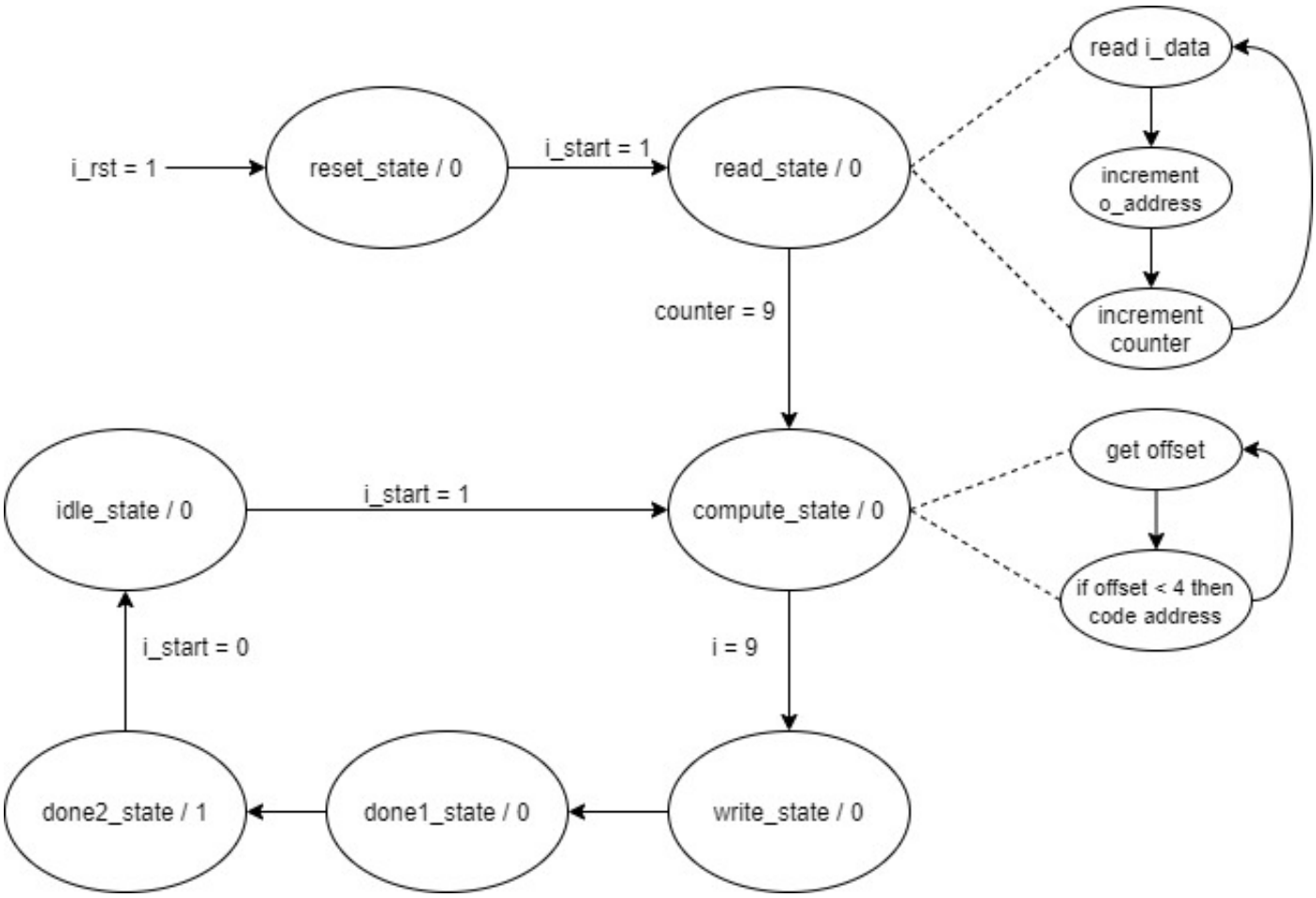
La memoria e il suo protocollo sono descritti nel seguente modo seguendo una specifica derivata dalla User Guide di Vivado:

```
-- Single-Port Block RAM Write-First Mode (recommended template)
--
-- File: rams_02.vhd
--
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
entity rams_sp_wf is
port(
  clk  : in  std_logic;
  we   : in  std_logic;
  en   : in  std_logic;
  addr : in  std_logic_vector(15 downto 0);
  di   : in  std_logic_vector(7 downto 0);
  do   : out std_logic_vector(7 downto 0)
);
end rams_sp_wf;

architecture syn of rams_sp_wf is
type ram_type is array (65535 downto 0) of std_logic_vector(7 downto 0);
signal RAM : ram_type;
begin
  process(clk)
    begin
    if clk'event and clk = '1' then
      if en = '1' then
        if we = '1' then
          RAM(conv_integer(addr)) <= di;
          do                      <= di;
        else
          do <= RAM(conv_integer(addr));
        end if;
      end if;
    end if;
  end process;
end syn;
```

2. Scelte Progettuali

Per l'implementazione ho deciso di utilizzare una Macchina a Stati Finiti (FSM), in particolare una macchina di Moore con l'uscita che corrisponde al segnale o_done:



2.1 Reset state

Si tratta dello stato di partenza della macchina ma viene raggiunto ogni volta che il segnale asincrono i_rst viene posto a '1'. Tutte le uscite vengono impostate al proprio valore di default, dopodichè la macchina rimane in attesa del segnale di start.

2.2 Read state

In questo stato viene attivata la memoria del testbench impostando il valore di o_en=1 e viene attivato un ciclo attraverso un contatore a 4 bit e un case statement, per leggere dalla memoria un valore per ogni periodo di clock. Questo valore viene salvato in una memoria temporanea e vengono incrementati il contatore e l'indirizzo in output.

2.3 Compute state

Una volta che sono stati letti tutti gli indirizzi dalla memoria parte un'altro ciclo che per ogni periodo di clock calcola la differenza tra l'indirizzo da codificare e l'indirizzo dell'i-esima working-zone e assegna il risultato alla variabile offset. Se questa differenza è < 4 viene codificato l'indirizzo secondo la specifica, in particolare:

- Il bit wz_bit viene posto a 1.
- Il numero (i) dell'i-esima working-zone viene convertito in 3 bit e assegnato a wz_num.
- L'offset precedentemente calcolato viene codificato one-hot in wz_offset attraverso un demultiplexer implementato attraverso un case-statement.

2.4 Write state

2.5 First done state

2.6 Second done state

2.7 Idle state

3. Risultati dei Test

4. Risultati della Sintesi

4.1 Test

4.1.1 Grid convergence

5. Conclusions

Bibliography

[1] T. Lang E. Musoll and J. Cortadella. Working-zone encoding for reducing the energy in microprocessor address buses. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 6(5):568–572, 1998.