Progetto finale di Reti Logiche

Prof. Fornaciari, Prof. Palermo e Prof. Salice
Anno Accademico 2022 - 2023

(AGGIORNATO AL 15 Dicembre 2022)

Descrizione generale

La specifica della "*Prova Finale (Progetto di Reti Logiche)*" per l'Anno Accademico 2022/2023 chiede di implementare un modulo HW (descritto in VHDL) che si interfacci con una memoria e che rispetti le indicazioni riportate nella seguente specifica.

Ad elevato livello di astrazione, il sistema riceve indicazioni circa una locazione di memoria, il cui contenuto deve essere indirizzato verso un canale di uscita fra i quattro disponibili.
Le indicazioni circa il canale da utilizzare e l'indirizzo di memoria a cui accedere vengono forniti mediante un ingresso seriale da un bit, mentre le uscite del sistema, ovvero i succitati canali, forniscono tutti i bit della parola di memoria in parallelo.

Interfacce

Il modulo da implementare ha **due ingressi primari da 1 bit** (**W** e **START**) e **5 uscite primarie**. Le uscite sono le seguenti: quattro da 8 bit (**Z0, Z1, Z2, Z3**) e una da 1 bit (**DONE**). Inoltre, il modulo ha un segnale di clock **CLK**, unico per tutto il sistema e un segnale di reset **RESET** anch'esso unico.

Funzionamento

All'istante iniziale, quello relativo al **reset** del sistema, le uscite hanno i seguenti valori:

Z0, Z1, Z2 e Z3 sono 0000 0000, DONE è 0.

I dati in ingresso, ottenuti come sequenze sull'**ingresso primario seriale W** lette sul fronte di salita del clock, sono organizzati nel seguente modo:

- 2 bit di intestazione (i primi della sequenza) seguiti da
- N bit di indirizzo della memoria.

Gli N bit permettono di costruire un indirizzo di memoria (si legga qui di seguito la specifica per questi N bit). All'indirizzo di memoria è memorizzato il **messaggio da 8 bit** che deve essere indirizzato verso un canale di **uscita**.

I due bit di **intestazione** identificano il **canale d'uscita** (Z0, Z1, Z2 o Z3) sul quale deve essere indirizzato il messaggio. Il primo bit è il bit più significativo del canale di uscita, il secondo quello meno significativo, più in dettaglio:

00 identifica Z0, 01 identifica Z1, 10 identifica Z2 e, infine, 11 identifica Z3.

Gli **N** bit di indirizzo possono variare da 0 fino ad un **massimo di 16 bit**. Gli indirizzi di memoria sono tutti di 16 bit.

Se il numero di bit di N è inferiore a 16, l'indirizzo viene **esteso** con **0 sui bit più significativi**. Ad esempio:

```
(N = 7)    1010111          -> 0000000001010111
(N = 16)   1110000001010111 -> 1110000001010111
(N = 0)    0000000000000000 -> 0000000000000000
```

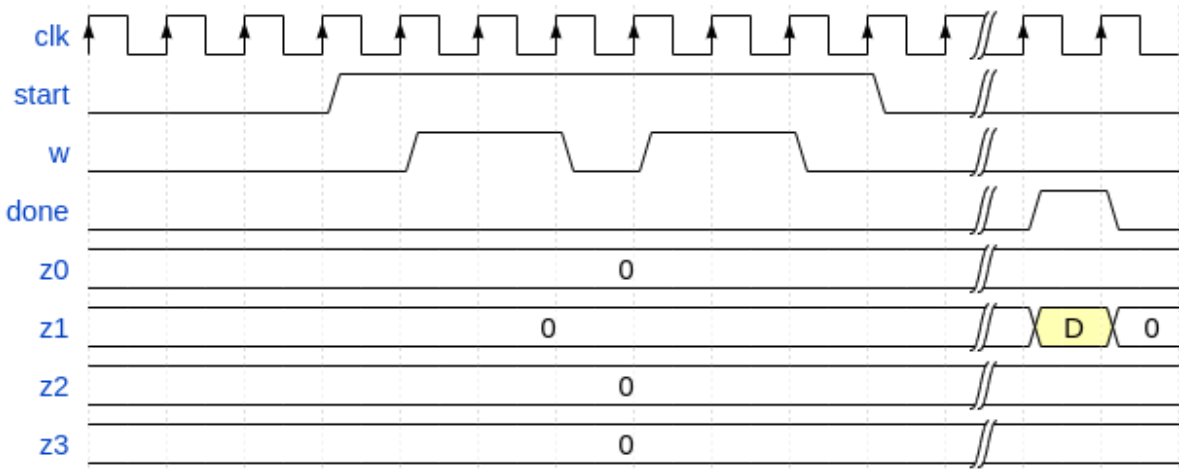
Tutti i bit su W devono essere letti sul fronte di salita del clock.

La sequenza di ingresso è valida quando il segnale START è alto (=1) e termina quando il segnale START è basso (=0).
Il segnale START rimane alto per almeno di 2 cicli di clock e non più di 18 cicli di clock (2 bit del canale e 16 bit per il massimo numero di bit per indirizzare la memoria). Si assuma questa condizione sempre verificata (non è necessario gestire il caso in cui il segnale di START rimanga attivo meno di 2 cicli di clock o più di 18).

Le uscite Z0, Z1, Z2 e Z3 sono inizialmente 0. I valori rimangano inalterati eccetto il canale sul quale viene mandato il messaggio letto in memoria; i valori sono visibili solo quando il valore di DONE è 1.
Quando il segnale DONE è 0 tutti i canali Z0, Z1, Z2 e Z3 devono essere a zero (32 bit a 0).
Contemporaneamente alla scrittura del messaggio sul canale, il segnale DONE passa da 0 passa a 1 e rimane attivo per un solo ciclo di clock (dopo 1 ciclo di clock DONE passa da 1 a 0). In pratica quando DONE=1 il canale associato al messaggio cambierà il suo valore, mentre gli altri canali mostreranno l'ultimo valore trasmesso derivato dai messaggi ad essi associati.
Il segnale START è garantito rimanere a 0 fino a che il segnale DONE non è tornato a 0.
Il tempo massimo per produrre il risultato (ovvero il tempo trascorso tra START=0 e DONE=1) deve essere inferiore a 20 cicli di clock.

Il modulo deve essere progettato considerando che prima del primo START=1 (e prima di richiedere il corretto funzionamento del modulo) verrà *sempre* dato il RESET (RESET=1). Una seconda (o successiva) elaborazione con START=1 non dovrà invece attendere il reset del modulo. Ogni qual volta viene dato il segnale di RESET (RESET=1), il modulo viene re-inizializzato.

Esempio (diagrammi temporali)
Lettura di un dato "D" dall'indirizzo di memoria 0000000000010110.
La scrittura del dato "D" avviene sull'uscita specificata Z1 (bit di intestazione "01")



Interfaccia del Componente
Il componente da descrivere deve avere la seguente interfaccia.

```
entity project_reti_logiche is
    port (
        i_clk   : in std_logic;
        i_rst   : in std_logic;
        i_start : in std_logic;
        i_w     : in std_logic;

        o_z0    : out std_logic_vector(7 downto 0);
        o_z1    : out std_logic_vector(7 downto 0);
        o_z2    : out std_logic_vector(7 downto 0);
        o_z3    : out std_logic_vector(7 downto 0);
        o_done  : out std_logic;

        o_mem_addr : out std_logic_vector(15 downto 0);
        i_mem_data : in std_logic_vector(7 downto 0);
        o_mem_we   : out std_logic;
        o_mem_en   : out std_logic
    );
end project_reti_logiche;
```

In particolare:

- il nome del modulo **deve essere** `project_reti_logiche` e deve essere presente **una sola architettura** per ogni entità; la violazione di queste indicazioni comporta l'impossibilità di eseguire il Test Bench e una conseguente valutazione di zero;
- i_clk è il segnale di CLOCK in ingresso generato dal Test Bench;
- i_rst è il segnale di RESET che inizializza la macchina pronta per ricevere il primo segnale di START;

- i_start è il segnale di START generato dal Test Bench;
- i_w è il segnale W precedentemente descritto e generato dal Test Bench;
- o_z0, o_z1, o_z2, o_z3 sono i quattro canali di uscita;
- o_done è il segnale di uscita che comunica la fine dell'elaborazione;
- o_mem_addr è il segnale (vettore) di uscita che manda l'indirizzo alla memoria;
- i_mem_data è il segnale (vettore) che arriva dalla memoria in seguito ad una richiesta di lettura;
- o_mem_en è il segnale di ENABLE da dover mandare alla memoria per poter comunicare (sia in lettura che in scrittura);
- o_mem_we è il segnale di WRITE ENABLE da dover mandare alla memoria (=1) per poter scriverci. Per leggere da memoria esso deve essere 0.

APPENDICE: Descrizione Memoria
NOTA: **La memoria è già istanziata all'interno del Test Bench e non va sintetizzata**

La memoria e il suo protocollo può essere estratto dalla seguente descrizione VHDL che fa parte del test bench e che è derivata dalla User guide di VIVADO disponibile al seguente link:
https://www.xilinx.com/support/documentation/sw_manuals/xilinx2017_3/ug901-vivado-synthesis.pdf

```
-- 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 after 2 ns;
        else
          do <= RAM(conv_integer(addr)) after 2 ns;
        end if;
      end if;
    end if;
  end process;
end syn;
```