

Progetto finale di Reti Logiche

Prof. Fornaciari, Prof. Palermo e Prof. Salice

(AGGIORNATO AL 28 Novembre 2018)

Descrizione generale

Sia dato uno spazio bidimensionale definito in termini di dimensione orizzontale e verticale, e siano date le posizioni di N punti, detti “centroidi”, appartenenti a tale spazio. Si vuole implementare un componente HW descritto in VHDL che, una volta fornite le coordinate di un punto appartenente a tale spazio, sia in grado di valutare a quale/i dei centroidi risulti più vicino (Manhattan distance).

Lo spazio in questione è un quadrato (256x256) e le coordinate nello spazio dei centroidi e del punto da valutare sono memorizzati in una memoria (la cui implementazione non è parte del progetto). La vicinanza al centroide viene espressa tramite una maschera di bit (maschera di uscita) dove ogni suo bit corrisponde ad un centroide: il bit viene posto a 1 se il centroide è il più vicino al punto fornito, 0 negli altri casi. Nel caso il punto considerato risulti equidistante da 2 (o più) centroidi, i bit della maschera d'uscita relativi a tali centroidi saranno tutti impostati ad 1.

Degli N centroidi $K \leq N$ sono quelli su cui calcolare la distanza dal punto dato. I K centroidi sono indicati da una maschera di ingresso a N bit: il bit a 1 indica che il centroide è valido (punto dal quale calcolare la distanza) mentre il bit a 0 indica che il centroide non deve essere esaminato. Si noti che la maschera di uscita è sempre a N bit e che i bit a 1 saranno non più di K.

Per il progetto, si consideri che il numero di centroidi sia pari a N=8.

Dati

I dati ciascuno di dimensione 8 bit sono memorizzati in una memoria con indirizzamento al Byte partendo dalla posizione 0.

- L'indirizzo 0 è usata per memorizzare il numero di centroidi (Maschera di ingresso: definisce quale centroide deve essere esaminato);
- Gli indirizzi dal 1 al 16 sono usati per memorizzare le coordinate a coppie X e Y dei centroidi:
 - 1 - Coordinata X 1° Centroide
 - 2 - Coordinata Y 1° Centroide
 - 3 - Coordinata X 2° Centroide
 - 4 - Coordinata Y 2° Centroide
 - ...
 - 15 - Coordinata X 8° Centroide
 - 16 - Coordinata Y 8° Centroide
- Gli indirizzi 17 e 18 sono usati per memorizzare le Coordinate X e Y del punto da valutare
- L'indirizzo 19 è usato per scrivere, alla fine, il valore della maschera di uscita.

Note ulteriori sulla specifica

1. I centroidi nella maschera vengono elencati dal bit meno significativo -posizione più a destra - (Centroide 1) al più significativo (Centroide 8);

2. Il valore della maschera risultante dall'identificazione del/dei centroide/i più vicino/i segue la stessa regola vista al punto precedente. Il bit meno significativo rappresenta il Centroide 1 mentre quello più significativo il Centroide 8;
3. 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.

ESEMPIO:

La seguente sequenza di numeri mostra un esempio del contenuto della memoria al termine di una elaborazione. I dati dall'indirizzo 0 a 18 sono ingressi per l'elaborazione mentre il dato in posizione 19 è il risultato dell'elaborazione. I valori che qui sono rappresentati in decimale, sono memorizzati in memoria con l'equivalente codifica binaria su 8 bit senza segno.

Indirizzo Memoria	Valore	Commento
0	185	/* Maschera 10111001
1	75	// X centroide 1 (da considerare vedi Maschera)
2	32	// Y centroide 1 (da considerare vedi Maschera)
3	111	// X centroide 2 (da NON considerare vedi Maschera)
4	213	// Y centroide 2 (da NON considerare vedi Maschera)
5	79	// X centroide 3 (da NON considerare vedi Maschera)
6	33	// Y centroide 3 (da NON considerare vedi Maschera)
7	1	// X centroide 4 (da considerare vedi Maschera)
8	33	// Y centroide 4 (da considerare vedi Maschera)
9	80	// X centroide 5 (da considerare vedi Maschera)
10	35	// Y centroide 5 (da considerare vedi Maschera)
11	12	// X centroide 6 (da considerare vedi Maschera)
12	254	// Y centroide 6 (da considerare vedi Maschera)
13	215	// X centroide 7 (da NON considerare vedi Maschera)
14	78	// Y centroide 7 (da NON considerare vedi Maschera)
15	211	// X centroide 8 (da considerare vedi Maschera)
16	121	// Y centroide 8 (da considerare vedi Maschera)
17	78	// X del punto da valutare
18	33	// Y del punto da valutare
19	17	// Maschera di uscita 00010001 (OUTPUT)

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.

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;
            else
                do <= RAM(conv_integer(addr));
            end if;
        end if;
    end if;
end process;
end syn;

```