Prova Finale

Progetto Reti Logiche 2021/2022

Xavier Wong (Codice Persona 10610479, Matricola 890686)

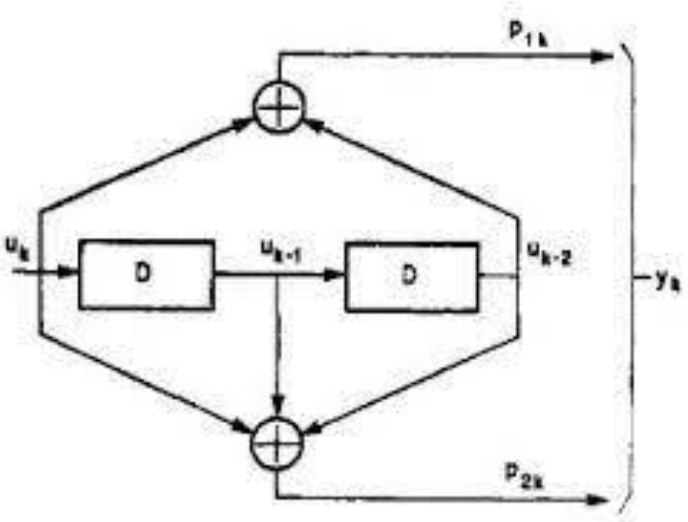
Indice:

b. Ottimizzazioni

1.Introduzione del progetto

Scopo del progetto

Lo scopo del progetto è di implementare un modulo hardware descritto in VHDL che, preso in ingresso una sequenza continua di W parole (ciascuno di 8 bit), restituisce un flusso di Z parole, generate dal codificatore convoluzionale con tasso di trasmissione ½ qua sotto descritto:



Specifiche del progetto

Il componente riceve in ingresso il numero di parole da codificare (il segnale “words” nel progetto) e scriverà i risultati a partire dall’indirizzo 1000 della RAM. Ciascuna parola viene serializzata per generare il flusso secondo il codificatore. Lo stato del codificatore viene resettato solo quando avrà elaborato ogni parola dell’input o se viene alzato il segnale di reset.

Interfaccia componente

Il componente da descrivere ha 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_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 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_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.

Dati e descrizione memoria

Ogni dato, di dimensione 8 bit, è memorizzato in una memoria con indirizzamento al byte:

- L'indirizzo 0 è riservato per la quantità di parole da leggere ed elaborare (con un valore massimo di 255 parole);
- Gli indirizzi da 1 a "WORDS" sono riservati per le parole da elaborare;

- Gli indirizzi da 1000 a 1000 +2*(WORDS) sono utilizzati per memorizzare il flusso d'uscita del codificatore.

2.Architettura del progetto

Quando il segnale i_start in ingresso viene portato a '1', il componente comincia l'elaborazione cambiando il proprio stato da IDLE a FETCH_W. Il componente termina la propria elaborazione portando a 1 il segnale o_done. Il test bench risponde portando in basso il segnale i_start che modifica lo stato della macchina a IDLE, in attesa del segnale i_start per ricominciare l'elaborazione.

Il componente dispone del segnale i_rst che, se portato al valore '1', riporta il componente allo stato iniziale asincronicamente.

Scelte progettuali

Il componente è dotato di 3 processi:

- Il primo processo rappresenta la parte sequenziale del progetto e gestisce i registri;
- Il secondo processo, denominato FSM, gestisce la lettura e la scrittura dei dati, contenendo uno stato di attesa per sincronizzarsi con il terzo processo;
- Il terzo e ultimo processo, denominato CON, è l'elaborazione dei dati secondo il codificatore convoluzionale, anch'essa una macchina a stati finiti.

Stati della FSM

1. IDLE: Stato iniziale in attesa del i_start = '1'. Ogni volta che il segnale i_rst viene alzato a '1' si ritorna in questo stato;

2. FETCH_W: Stato in cui viene chiesto la lettura della quantità di parole da elaborare leggendo il dato nell'indirizzo 0 della RAM;

3. WAITRAM: Stato in cui si attende la risposta della memoria dopo una richiesta di lettura;

4. GET_W: Stato in cui si memorizza la quantità di parole da codificare nel segnale "WORDS";

5. FETCHWORD: Stato in cui viene chiesto la lettura della parola da elaborare;
6. GETWORD: Stato in cui viene memorizzato la parola da elaborare nel registro cur_word per poi essere utilizzato nel processo CON;

7. CONV: Stato di attesa dell'elaborato del processo CON;

8. WRITE1: Stato di scrittura della prima metà dell'elaborato. Se ultime parole, alza il segnale di o_done dopo aver scritto la seconda metà dell'elaborato;

9. WRITE2: Stato di scrittura della seconda metà dell'elaborato della parola e aggiornamento dell'indirizzo di lettura;

10. DONE: Stato in cui si attende il segnale i_start basso per ricominciare l'elaborazione del flusso d'ingresso;

Stati del CON

Gli stati del processo CON sono ottenuti dal diagramma degli stati ottenuta dalla definizione del codificatore convoluzionale con tasso di trasmissione ½. In particolare, tale macchina torna allo stato 00 (l'equivalente di OO del progetto) quando lo stato della FSM è IDLE (che torna se inizia una nuova elaborazione o riceve in input i_rst = '1').

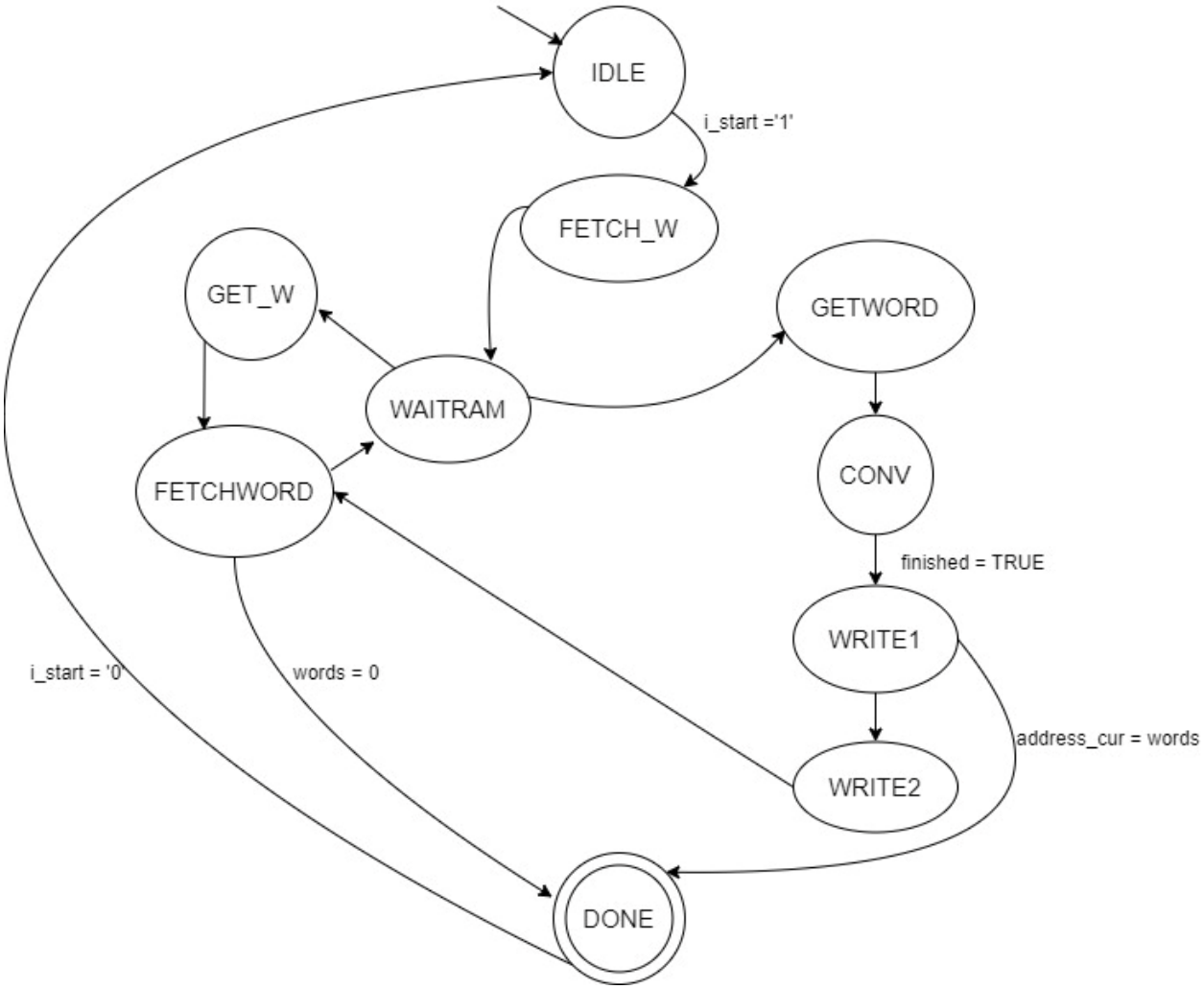


Diagramma degli stati di FSM

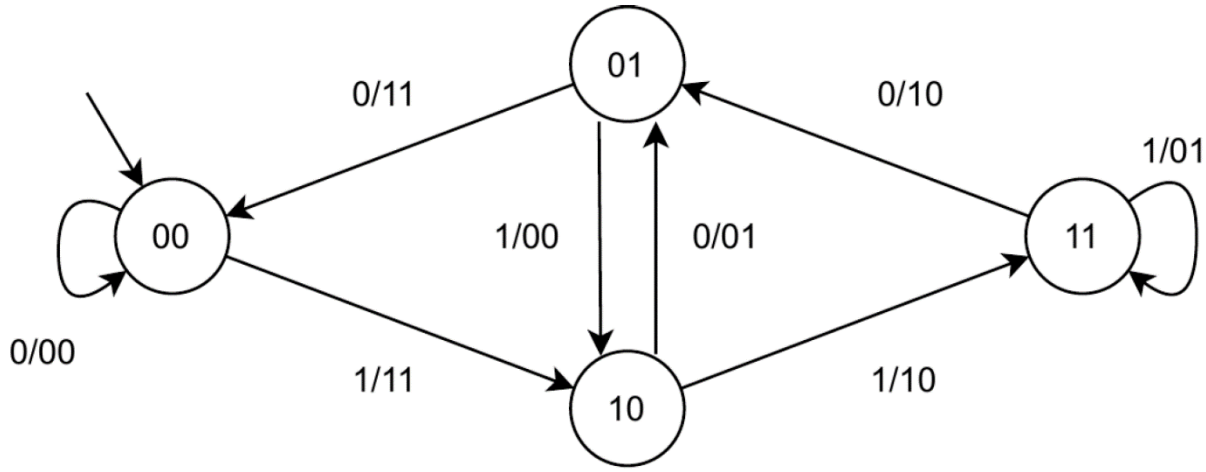
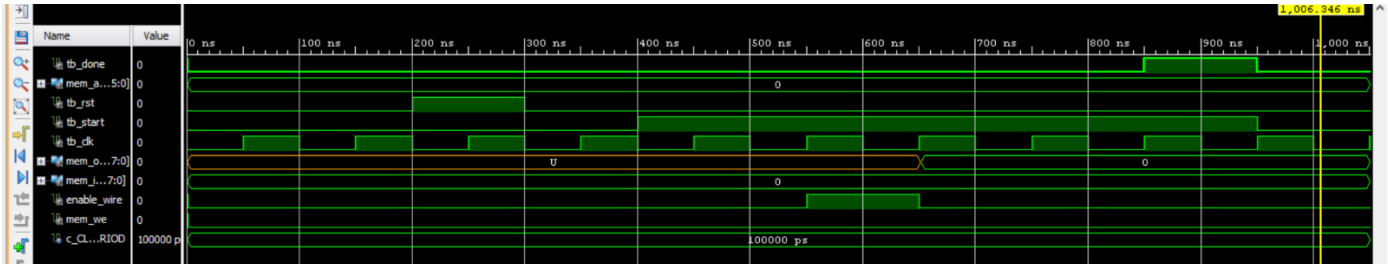


Diagramma degli stati di CON

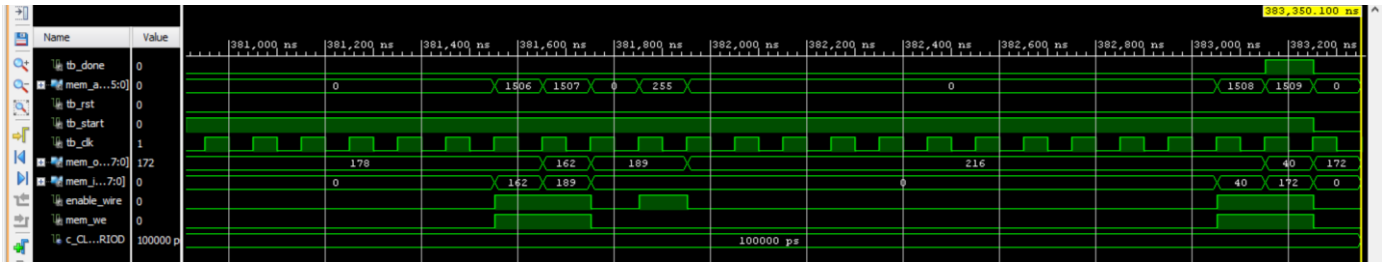
3.Risultati dei test

Per verificare la correttezza del componente sintetizzato, dopo averlo testato con i test bench d'esempio, ho definito altri test che verificano il comportamento del componente nei casi estremi per massimizzare i casi coperti dalla macchina a stati finiti:

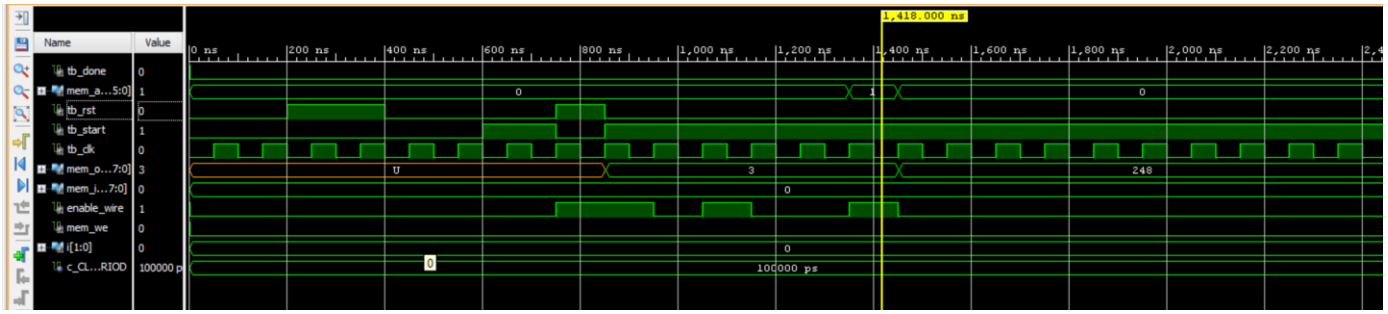
1. **Caso dimensione parole nulla (WORDS = 0).** Il test verifica che il componente non abbia scritto nulla in memoria;



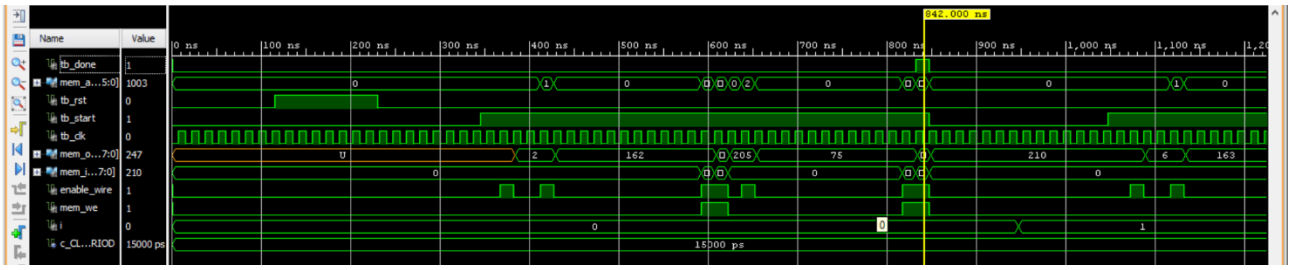
2. **Caso dimensione massima (WORDS = 255).** Il test verifica che l'elaborazione sia corretta;



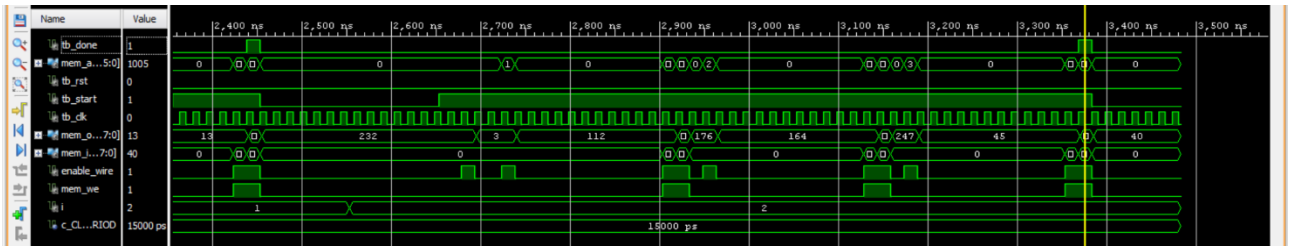
3. **Reset asincroni**. Il test verifica che lo stato della macchina vada correttamente a IDLE a ogni i_rst = '1';



4. **Caso generale** di elaborazione di flussi di dimensione e dati diversi. Il test verifica la correttezza di ogni input.



Primo done all'altezza della linea gialla



Gli ultimi 2 done del test con 3 flussi in ingresso

4.Conclusioni

Risultato delle sintesi

Il componente supera tutti i test nelle simulazioni Behavioral e Post-synthesis functional. In seguito vengono riportati i messaggi dei test elencati prceedentemente:

1. **Caso dimensione parole nulla:**

   Failure: Simulation Ended! TEST PASSATO

   Time: 1050100 ps Iteration: 0 Process: /project_tb/test

   File:D:/xavie/10610479.vhd/10610479.vhd.srcs/sim_1/imports/testbenches/tb_seq_min.vhd

   $finish called at time : 1050100 ps : File"D:/xavie/10610479.vhd/10610479.vhd.srcs/sim_1/imports/testbenches/tb_seq_min.vhd Line 107

2. **Caso dimensione parole massima:**

   Failure: Simulation Ended! TEST PASSATO

   Time: 383350100 ps Iteration: 0 Process: /project_tb/test

   File:D:/xavie/10610479.vhd/10610479.vhd.srcs/sim_1/imports/testbenches/tb_seq_max.vhd

$finish called at time : 383350100 ps :

File"D:/xavie/10610479.vhd/10610479.vhd.srcs/sim_1/imports/testbenches/tb_seq_max.vhd" Line 863

3. **Reset asincroni:**

Failure: Simulation Ended! TEST PASSATO

Time: 21150100 ps Iteration: 0 Process: /project_tb/test

File:D:/xavie/10610479.vhd/10610479.vhd.srcs/sim_1/imports/testbenches/tb_tre_reset.vhd

$finish called at time : 21150100 ps :

File"D:/xavie/10610479.vhd/10610479.vhd.srcs/sim_1/imports/testbenches/tb_tre_reset.vhd" Line 204

4. **Caso generale:**

Failure: Simulation Ended! TEST PASSATO

Time: 3482600 ps Iteration: 0 Process: /project_tb/test

File:D:/xavie/10610479.vhd/10610479.vhd.srcs/sim_1/imports/testbenches/tb_tre_bis.vhd

$finish called at time : 3482600 ps :

File"D:/xavie/10610479.vhd/10610479.vhd.srcs/sim_1/imports/testbenches/tb_tre_bis.vhd" Line 201

Ottimizzazioni

Per rendere più efficiente il componente nel caso di parole da leggere uguale a 0, lo stato FETCHWORD ha un ramo if che controlla il valore del segnale WORDS per portare immediatamente lo stato DONE. Ho separato i processi FSM e CON per maggiore leggibilità (nonostante si corri il rischio di inferred latch).