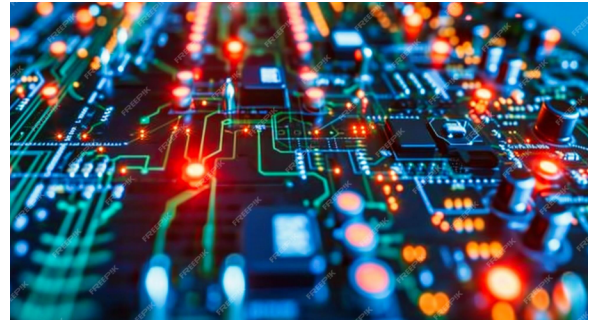


LAE1 **VHDL** – a.a. 2025-26

Simulazione PAR2 (12/12/2025)



Tempo a disposizione: 45 minuti

Struttura tema esame: **2 esercizi**. Il **primo** di **completamento** di DESCRIZIONE (no testbench) in VHDL di una rete logica sequenziale. Riceverete un sorgente VHDL contenente degli spazi in cui inserire il vostro codice VHDL. Le posizioni in cui inserire il codice sono identificate dal seguente pattern:

----- YOUR CODE HERE -----

Il vostro codice dovrà essere inserito **solo tra le due righe di commento**. Qualsiasi linea di codice inserita al di fuori delle aree indicate comporterà una **riduzione** dei punti assegnati alla soluzione dell'esercizio.

Il **secondo** esercizio consisterà nell'inserimento di risposte in un file di testo con formato prefissato. Riceverete un template risposte con un numero N di domande ed il seguente formato (**NON CAMBIATE IL NOME DEL FILE!**):

<1>

<2>

<3>

...

<N>

In esso le risposte vanno inserite :

- **SENZA** rimuovere i tag delle domande <x>

- **SENZA LASCIARE spazi** tra il question tag e la risposta (ad es. <x>A)
- Nel caso che la domanda riguardi quantità fisiche **NON inserite le unità di misura!** Ad esempio ... Alla domanda x "Quanti ns sono necessari perché il segnale A passi dal livello basso al livello altro?" potreste rispondere così :
<x>110

Relazioni di contenuto tra esercizio 1 e esercizio 2:

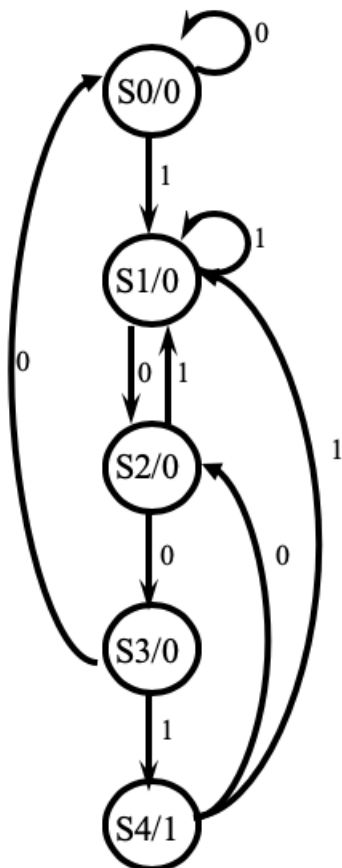
Posto che il primo esercizio (completamento codice) riguarderà **SEMPRE** una rete logica di tipo sequenziale da descrivere in VHDL, il contenuto di esercizio 2 (set di domande) è più vario. In esso potete (ed è anche probabile) trovare almeno una o due domande sul contenuto di domanda 1 ma le altre possono spaziare tra gli argomenti trattati nella seconda metà del corso (logica sequenziale in VHDL, registri, FSM, contatori). A corredo di esercizio 2 possono essere fornite delle immagini (analisi onda, circuiti rappresentati in formato black box, STD) che possono servire per rispondere a specifiche domande. Il numero domande a cui rispondere in esercizio 2 **è fissato a 5**.

Esercizio 1

Completare il codice VHDL (ese1.vhdl: parti – YOUR CODE HERE --) per implementare una Macchina a Stati Finiti (FSM) di Moore asincrona al reset, che riconosca la sequenza di bit in ingresso "1001" ed emetta, in caso di riconoscimento, un 1 sulla linea di output O.

- Ingressi :
 - o clk : clock di sistema.
 - o reset : segnale di reset asincrono. Quando RESET = '1', la macchina deve tornare immediatamente allo stato iniziale (S0).
 - o I : il bit di ingresso seriale.
- Uscita:
 - o O : segnale a '1' solo quando la macchina raggiunge lo stato che indica il completamento della sequenza "1001".

La macchina e' descritta dal seguente diagramma di transizione tra stati (STD) all'interno del quale, in ogni cerchio utilizzato per descrivere uno stato, trovate in quest'ordine due valori, il **nome dello stato** e il **valore sulla linea di output**, separati da un simbolo / . Sulle frecce che rappresentano le transizioni tra stati e' possibile osservare il valore sulla linea di input I . Segue STD della FSM :



ATTENZIONE! Come e' possibile notare da STD la FSM e' in grado di riconoscere occorrenze **SOVRAPPOSTE** del pattern da identificare (1001) ...

Esercizio 2 (set di domande)

<1> Qual è la sequenza di bit di input minima e più diretta che deve essere applicata al segnale sulla linea di input I (a partire dallo stato iniziale) per garantire che l'uscita O della FSM Moore passi a '1'? Specificare anche il valore dell'uscita **dopo che l'ultimo bit della sequenza** ha causato la transizione di stato.

A) Sequenza: 1001; Output dopo l'ultimo bit: '1'

B) Sequenza: 1001; Output dopo l'ultimo bit: '0'

C) Sequenza: 1010; Output dopo l'ultimo bit: '0'

D) Sequenza: 10010; Output dopo l'ultimo bit: '0'

<2> Quale delle seguenti sequenze di input è la più efficiente per ottenere un secondo match (sfruttando al meglio la logica di stato prossimo della FSM)?

A) 10011001

B) 1001001

C) 1000

D) 101001

<3> Quale tra le seguenti descrizioni VHDL nel processo **memory state** con sensitivity list (clk, reset) definisce una linea di **reset asincrono**?

```
A) if RESET = '1' then
    Current_State <= S0;
elseif rising_edge(CLK) then
    Current_State <= Next_State;
end if;
```

```
B) if rising_edge(CLK) then
    if RESET = '1' then
        Current_State <= S0;
    else
        Current_State <= Next_State;
    end if;
end if;
```

```
C) if RESET = '1' or rising_edge(CLK) then
    Current_State <= S0;
end if;
```

```
D) if CLK = '1' then
    if RESET = '1' then
        Current_State <= S0;
    end if;
end if;
```

<4> Partendo dallo stato iniziale S0 della FSM descritta in **Esercizio 1** , quale è il numero **minimo** di cicli di clock necessari affinché l'uscita O (output) del riconoscitore di sequenza '1001' (Macchina di Moore) diventi '1'?

- A) 3
- B) 4**
- C) 5
- D) 1

<5> Si consideri un **SINGOLO** processo VHDL destinato all'implementazione di un **contatore sincrono**. Quale tra i seguenti blocchi di codice composti da una o più righe descrive correttamente l'aggiornamento del **segnale CSI, interno** alla descrizione VHDL dell'architettura del contatore? Assumete che CSI sia un **segnale** avente tipo unsigned.

- A) **if(rising_edge(CLK)) then**
 Contatore <= Contatore + 1;

 end if;
- B) Contatore := Contatore + 1;
- C) if CLK = '1' then
 Contatore <= Contatore + 1;

 end if;
- D) if rising_edge(CLK) then
 Contatore := Contatore + 1;

 end if;