Università degli Studi di Napoli Federico II

Scuola Politecnica e delle Scienze di Base

Dipartimento di Ingegneria Elettrica e Tecnologie dell’Informazione

Corso di Laurea Magistrale in Ingegneria Informatica

Immagine che contiene cerchio, simbolo, schizzo

Descrizione generata automaticamente

Elaborato di Architettura dei Sistemi Digitali

*Prof.ssa Alessandra De Benedictis*

a.a. 2024-25

Studenti:

Alessandro Campanella M63001697

Cesare Pulcrano M63001697

Sommario

[**Capitolo 1: Reti combinatorie elementari** 5](#_Toc197506266)

[Esercizio 1: Multiplexer 16:1 5](#_Toc197506267)

[Esercizio 1.1 5](#_Toc197506268)

[Progetto e architettura 5](#_Toc197506269)

[Implementazione 5](#_Toc197506270)

[Simulazione 5](#_Toc197506271)

[Esercizio 1.2 5](#_Toc197506272)

[Progetto e architettura 5](#_Toc197506273)

[Implementazione 5](#_Toc197506274)

[Simulazione 5](#_Toc197506275)

[Esercizio 1.3 5](#_Toc197506276)

[Sintesi su board di sviluppo 5](#_Toc197506277)

[Esercizio 2: Sistema ROM + M 6](#_Toc197506278)

[Esercizio 2.1 6](#_Toc197506279)

[Progetto e architettura 6](#_Toc197506280)

[Implementazione 6](#_Toc197506281)

[Simulazione 6](#_Toc197506282)

[Esercizio 2.2 6](#_Toc197506283)

[Sintesi su board di sviluppo 6](#_Toc197506284)

[**Capitolo 2: Reti sequenziali elementari** 7](#_Toc197506285)

[Esercizio 3: Riconoscitore di sequenze 7](#_Toc197506286)

[Esercizio 3.1 7](#_Toc197506287)

[Progetto e architettura 7](#_Toc197506288)

[Implementazione 7](#_Toc197506289)

[Simulazione 7](#_Toc197506290)

[Esercizio 3.2 7](#_Toc197506291)

[Sintesi su board di sviluppo 7](#_Toc197506292)

[Esercizio 4: Shift register 7](#_Toc197506293)

[Esercizio 4.1 7](#_Toc197506294)

[Progetto e architettura 8](#_Toc197506295)

[Implementazione 8](#_Toc197506296)

[Simulazione 8](#_Toc197506297)

[Esercizio 5: Cronometro 8](#_Toc197506298)

[Esercizio 5.1 8](#_Toc197506299)

[Progetto e architettura 8](#_Toc197506300)

[Implementazione 8](#_Toc197506301)

[Simulazione 8](#_Toc197506302)

[Esercizio 5.2 8](#_Toc197506303)

[Sintesi su board di sviluppo 8](#_Toc197506304)

[Esercizio 5.3 (solo 9 CFU) 9](#_Toc197506305)

[Sintesi su board di sviluppo 9](#_Toc197506306)

[Esercizio 6: Sistema di lettura-elaborazione-scrittura PO\_PC 9](#_Toc197506307)

[Esercizio 6.1 9](#_Toc197506308)

[Progetto e architettura 9](#_Toc197506309)

[Implementazione 9](#_Toc197506310)

[Simulazione 9](#_Toc197506311)

[Esercizio 6.2 9](#_Toc197506312)

[Sintesi su board di sviluppo 9](#_Toc197506313)

[**Capitolo 3: Macchine aritmetiche** 10](#_Toc197506314)

[Esercizio 7: Moltiplicatore di Booth 10](#_Toc197506315)

[Esercizio 7.1 10](#_Toc197506316)

[Progetto e architettura 10](#_Toc197506317)

[Implementazione 10](#_Toc197506318)

[Simulazione 10](#_Toc197506319)

[Esercizio 7.2 10](#_Toc197506320)

[Sintesi su board di sviluppo 10](#_Toc197506321)

[Esercizio 7BIS: Divisore Non-Restoring (solo 9 CFU) 10](#_Toc197506322)

[Esercizio 7BIS.1 10](#_Toc197506323)

[Progetto e architettura 10](#_Toc197506324)

[Implementazione 10](#_Toc197506325)

[Simulazione 10](#_Toc197506326)

[Esercizio 7BIS.2 11](#_Toc197506327)

[Sintesi su board di sviluppo 11](#_Toc197506328)

[**Capitolo 4: Comunicazione con handshaking** 11](#_Toc197506329)

[Esercizio 8: Comunicazione con handshaking 11](#_Toc197506330)

[Esercizio 8.1 11](#_Toc197506331)

[Progetto e architettura 11](#_Toc197506332)

[Implementazione 11](#_Toc197506333)

[Simulazione 11](#_Toc197506334)

[**Capitolo 5: Processore** 12](#_Toc197506335)

[Esercizio 9: Processore IJVM 12](#_Toc197506336)

[Progetto e architettura 12](#_Toc197506337)

[Implementazione 12](#_Toc197506338)

[Simulazione 12](#_Toc197506339)

[**Capitolo 6: Interfaccia seriale** 12](#_Toc197506340)

[Esercizio 10: Interfaccia UART 12](#_Toc197506341)

[Progetto e architettura 12](#_Toc197506342)

[Implementazione 12](#_Toc197506343)

[Simulazione 12](#_Toc197506344)

[Esercizio 10BIS: Interfaccia UART (solo 9 CFU) 13](#_Toc197506345)

[Sintesi su board di sviluppo 13](#_Toc197506346)

[**Capitolo 7: Switch multistadio** 13](#_Toc197506347)

[Esercizio 11: Switch multistadio 13](#_Toc197506348)

[Esercizio 11.1 13](#_Toc197506349)

[Progetto e architettura 13](#_Toc197506350)

[Implementazione 13](#_Toc197506351)

[Simulazione (?) 13](#_Toc197506352)

[Esercizio 11.2 (solo 9 CFU) 13](#_Toc197506353)

[Progetto e architettura 13](#_Toc197506354)

[Implementazione 13](#_Toc197506355)

[Simulazione (?) 14](#_Toc197506356)

[Esercizio 11.3 (solo 9 CFU) 14](#_Toc197506357)

[Progetto e architettura 14](#_Toc197506358)

[Implementazione 14](#_Toc197506359)

[Simulazione (?) 14](#_Toc197506360)

[**Capitolo 8: Esercizio prova di esame dicembre 2024** 14](#_Toc197506361)

[Esercizio 12: Prova di esame del 19 dicembre 2024 14](#_Toc197506362)

[Progetto e architettura 14](#_Toc197506363)

[Implementazione 14](#_Toc197506364)

[Simulazione (?) 15](#_Toc197506365)

[Appendice 16](#_Toc197506366)

[Button Debouncer 16](#_Toc197506367)

[Progetto e architettura 16](#_Toc197506368)

[Implementazione 16](#_Toc197506369)

[Divisore di frequenza 16](#_Toc197506370)

[Progetto e architettura 16](#_Toc197506371)

[Implementazione 16](#_Toc197506372)

# **Capitolo 1: Reti combinatorie elementari**

## Esercizio 1: Multiplexer 16:1

### Esercizio 1.1

Progettare, implementare in VHDL e testare mediante simulazione un **multiplexer indirizzabile 16:1**, utilizzando un approccio di progettazione per composizione a partire da **multiplexer 4:1.**

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 1.2

Utilizzando il componente sviluppato al punto precedente, progettare, implementare in VHDL e testare mediante simulazione una **rete di interconnessione a 16 sorgenti e 4 destinazioni**.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 1.3

Sintetizzare ed implementare su board il progetto della rete di interconnessione sviluppato al punto 1.2, utilizzando gli switch per fornire gli input di selezione e i led per visualizzare i 4 bit di uscita. Per quanto riguarda i 16 bit dato in input, essi devono essere immessi mediante switch, 8 bit alla volta, sviluppando un’apposita “rete di controllo” per l’acquisizione che utilizzi due bottoni della board per caricare rispettivamente la prima e la seconda metà del dato in ingresso.

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

## Esercizio 2: Sistema ROM + M

### Esercizio 2.1

Progettare, implementare in VHDL e testare mediante simulazione un **sistema S** composto da una **ROM** puramente combinatoria di 16 locazioni da 8 bit ciascuna e da una macchina combinatoria **M** che opera come segue: fornito al sistema un indirizzo A di 4 bit, il sistema restituisce il valore contenuto nella ROM all’indirizzo A opportunamente “trasformato” attraverso la macchina M. Il comportamento della macchina M è totalmente a scelta dello studente, l’unico vincolo è che essa prenda in ingresso 8 bit e ne fornisca in uscita 4.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 2.2

Sintetizzare ed implementare su board il progetto del sistema ROM+M sviluppato al punto 2.1, utilizzando gli switch per fornire l’indirizzo della ROM da cui leggere i valori da trasformare e i led per visualizzare i 4 bit di uscita.

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

# **Capitolo 2: Reti sequenziali elementari**

## Esercizio 3: Riconoscitore di sequenze

### Esercizio 3.1

Progettare, implementare in VHDL e testare mediante simulazione una macchina in grado di riconoscere la sequenza **101**. La macchina prende in ingresso un segnale binario i che rappresenta il dato, un segnale A di tempificazione e un segnale M di modo, che ne disciplina il funzionamento, e fornisce un’uscita Y alta quando la sequenza viene riconosciuta. In particolare,

* se M=0, la macchina valuta i bit seriali in ingresso a gruppi di 3 (sequenze non sovrapposte),
* se M=1, la macchina valuta i bit seriali in ingresso uno alla volta, tornando allo stato iniziale ogni volta che la sequenza viene correttamente riconosciuta (sequenze parzialmente sovrapposte).

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 3.2

Sintetizzare e implementare su board la rete sviluppata al punto precedente, utilizzando uno switch S1 per codificare l’input i e uno switch S2 per codificare il modo M, in combinazione con due bottoni B1 e B2 utilizzati rispettivamente per acquisire l’input da S1 e S2 in sincronismo con il segnale di tempificazione A, che deve essere ottenuto a partire dal clock della board. Infine, l’uscita Y può essere codificata utilizzando un led.

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

## Esercizio 4: Shift register

### Esercizio 4.1

Progettare, implementare in VHDL e testare mediante simulazione un registro a scorrimento di N bit in grado di shiftare a destra o a sinistra di un numero Y variabile di posizioni a seconda di una opportuna selezione. In particolare, i valori possibili di Y sono 1 e 2. L’utente tramite selezione deve scegliere di quante posizioni shiftare. Il componente deve essere realizzato utilizzando sia un a) approccio comportamentale sia un b) approccio strutturale.

Nota: il numero di bit del registro deve essere implementato come un generic, e dall’esterno deve poter essere scelta la modalità di funzionamento mediante opportuni segnali di selezione.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

## Esercizio 5: Cronometro

### Esercizio 5.1

Progettare, implementare in VHDL e testare mediante simulazione un cronometro, in grado di scandire secondi, minuti e ore a partire da una base dei tempi prefissata (es. si consideri il clock a disposizione sulla board). Il progetto deve prevedere la possibilità di inizializzare il cronometro con un valore iniziale, sempre espresso in termini di ore, minuti e secondi, mediante un opportuno ingresso di set, e deve prevedere un ingresso di reset per azzerare il tempo. Il componente deve essere realizzato utilizzando un approccio strutturale, collegando opportunamente dei contatori secondo uno schema a scelta.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 5.2

Sintetizzare ed implementare su board il componente sviluppato al punto precedente, utilizzando i display a 7 segmenti per la visualizzazione dell’orario (o una combinazione di display e led nel caso in cui i display a disposizione siano in numero inferiore a quello necessario), gli switch per l’immissione dell’orario iniziale e due bottoni, uno per il set dell’orario e uno per il reset. Si utilizzi una codifica a scelta dello studente per la visualizzazione dell’orario sui display (esadecimale o decimale).

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

### Esercizio 5.3 (solo 9 CFU)

Estendere il componente sviluppato ai punti precedenti in modo che sia in grado di acquisire e memorizzare internamente fino ad un numero N di intertempi in corrispondenza di un ingresso di stop. Opzionalmente, il componente può prevedere una modalità di visualizzazione in cui, alla pressione di un bottone, vengano visualizzati sui display gli intertempi memorizzati (uno per ogni pressione).

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

## Esercizio 6: Sistema di lettura-elaborazione-scrittura PO\_PC

### Esercizio 6.1

Progettare, implementare in VHDL e verificare mediante simulazione un sistema dotato di una memoria ROM di N locazioni da 8 bit ciascuna, una macchina combinatoria M in grado di trasformare (secondo una funzione a scelta dello studente) la stringa di 8 bit letta dalla ROM in una stringa di 4 bit, e una memoria MEM di N locazioni che memorizza la stringa in output da M.

Il sistema si avvia in corrispondenza di un segnale di START che viene fornito esternamente. Una volta avviato, tramite un’apposita unità di controllo che gestisce la tempificazione del sistema, viene scandita una locazione alla volta della ROM e viene scritta la corrispondente locazione di MEM. Gli indirizzi di memoria sono forniti da un contatore. Le memorie ROM e MEM hanno rispettivamente un read e un write sincrono.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 6.2

Sintetizzare ed implementare su board il componente sviluppato al punto precedente, utilizzando due bottoni per i segnali di read e reset rispettivamente e i led per la visualizzazione delle uscite della macchina istante per istante.

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

# **Capitolo 3: Macchine aritmetiche**

## Esercizio 7: Moltiplicatore di Booth

### Esercizio 7.1

Progettare, implementare in VHDL e simulare una macchina moltiplicatore di Booth in grado di effettuare il prodotto di 2 stringhe A e B da 8 bit ciascuna.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 7.2

Sintetizzare il moltiplicatore implementato al punto 7.1 su FPGA e testarlo mediante l’utilizzo dei dispositivi di input/output (switch, bottoni, led, display) presenti sulla board di sviluppo in dotazione. La modalità di utilizzo degli stessi è a completa discrezione degli studenti.

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

## Esercizio 7BIS: Divisore Non-Restoring (solo 9 CFU)

### Esercizio 7BIS.1

Progettare, implementare in VHDL e simulare una macchina divisore (modalità non-restoring) in grado di effettuare la divisione intera fra due stringhe A e B di 4 bit ciascuna.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 7BIS.2

Sintetizzare il divisore implementato al punto 7BIS.1 su FPGA e testarlo mediante l’utilizzo dei dispositivi di input/output (switch, bottoni, led, display) presenti sulla board di sviluppo in dotazione. La modalità di utilizzo degli stessi è a completa discrezione degli studenti.

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

# **Capitolo 4: Comunicazione con handshaking**

## Esercizio 8: Comunicazione con handshaking

### Esercizio 8.1

Progettare, implementare in VHDL e testare mediante simulazione un sistema composto da 2 nodi, A e B, che comunicano mediante un protocollo di handshaking. Il nodo A e il nodo B possiedono entrambi una memoria interna in cui sono memorizzate N stringhe di M bit, denominate X(i) e Y(i) rispettivamente (i=0,..,N-1). Il nodo A trasmette a B ciascuna stringa X(i) utilizzando un protocollo di handshaking; B, ricevuta la stringa X(i), calcola S(i)=X(i)+Y(i) e immagazzina la somma in opportune locazioni della propria memoria interna.

Per il progetto è possibile considerare una implementazione di tipo comportamentale per effettuare la somma, mentre è necessario prevedere esplicitamente un componente contatore sia nel sistema A sia nel sistema B per scandire la trasmissione/ricezione delle stringhe e per terminare la comunicazione.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

# **Capitolo 5: Processore**

## Esercizio 9: Processore IJVM

A partire dall’implementazione fornita del processore operante secondo il modello IJVM,

1. si proceda all’analisi dell’architettura mediante simulazione e si approfondisca lo studio del suo funzionamento per due istruzioni a scelta,
2. si modifichi un codice operativo a scelta, documentando tutte le modifiche effettuate.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

# **Capitolo 6: Interfaccia seriale**

## Esercizio 10: Interfaccia UART

Partendo dall’implementazione fornita dalla Digilent di un dispositivo UART-RS232 (componente RS232RefComp.vhd), progettare, implementare e simulare in VHDL un sistema composto da 2 unità A e B che condividono lo stesso segnale di clock e comunicano tra loro mediante interfaccia seriale. Il sistema A contiene una ROM di 8 locazioni da 1 byte ciascuno, un contatore CONT\_A per scandire le locazioni della ROM e una UART\_A, mentre il sistema B contiene una memoria MEM di 8 locazioni da 1 byte ciascuno, un contatore CONT\_B per scandire le locazioni della MEM e una UART\_B. Quando un segnale WR viene asserito nell’unità A, viene prelevato un byte dalla ROM e inviato all’unità B, che dovrà riceverlo e salvarlo in MEM.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

## Esercizio 10BIS: Interfaccia UART (solo 9 CFU)

Dopo aver simulato il comportamento del sistema, lo si implementi su board usando un bottone per il segnale di WR in A, un bottone per il segnale di RD in B, e i display per la visualizzazione del dato correntemente trasmesso e memorizzato in MEM. Si testi l’errore di overrun.

#### Sintesi su board di sviluppo

<se richiesto: descrizione dell’architettura complessiva necessaria per la sintesi su board di sviluppo (nel caso ci siano eventuali componenti aggiuntivi per la gestione dell’I/O); file di constraint utilizzato per il progetto>

# **Capitolo 7: Switch multistadio**

## Esercizio 11: Switch multistadio

### Esercizio 11.1

Progettare ed implementare in VHDL uno switch multistadio secondo il modello omega network. Lo switch deve consentire lo scambio di messaggi di 2 bit ciascuno da un nodo sorgente a un nodo destinazione in una rete con 4 nodi, implementando uno schema a priorità fissa fra i nodi (es. nodo 1 più prioritario, con priorità decrescenti fino al nodo 4).

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione (?)

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 11.2 (solo 9 CFU)

Rimuovendo l’ipotesi di lavorare secondo uno schema a priorità fissa fra i nodi e considerando una rete di 8 nodi, lo switch deve gestire eventuali conflitti generati da collisioni con un meccanismo a scelta dello studente.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione (?)

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

### Esercizio 11.3 (solo 9 CFU)

Si implementi un protocollo di handshaking semplice regolato da una coppia di segnali (pronto a inviare/pronto a ricevere) per l’invio di ciascun messaggio fra due nodi.

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione (?)

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

# **Capitolo 8: Esercizio prova di esame dicembre 2024**

## Esercizio 12: Prova di esame del 19 dicembre 2024

Un sistema è composto da 2 nodi, A e B. A include una ROM (progettata come macchina sequenziale con READ sincrono) di 8 locazioni da 4 bit, mentre B include un sommatore parallelo in grado di effettuare la somma di 2 stringhe di 4 bit ciascuna e un registro R di 4 bit. Il sistema opera come segue: all’arrivo di un segnale di start, A inizia a prelevare gli elementi ROM[i] dalla propria memoria e li invia, uno alla volta, a B mediante handshaking. B somma progressivamente le stringhe ricevute utilizzando il sommatore e alla fine inserisce il risultato nel registro R.

1. Si disegni l’architettura complessiva del sistema tramite un diagramma a blocchi, identificando parte operativa e parte di controllo di ciascun nodo. Ogni nodo deve essere progettato seguendo un approccio strutturale, individuando tutti i componenti, le loro interfacce e le loro interconnessioni.
2. Si progettino le unità di controllo di A e B evidenziando gli stati, gli ingressi e le uscite negli automi risultanti. È obbligatorio specificare la tempificazione che si intende dare alle macchine (fronte attivo del clock, tempificazione dei segnali di READ/WRITE su registri e memorie).
3. Si progetti il sommatore secondo un’architettura di tipo carry look ahead.
4. Si fornisca l’implementazione in VHDL dell’intero sistema e si proceda alla simulazione nel caso in cui il clock del sistema A e del sistema B siano diversi (A più lento e A più veloce).

#### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

#### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

#### Simulazione (?)

<descrizione dei testbench di maggiore rilevanza utilizzati per testare il sistema e i suoi componenti e discussione dei principali risultati in simulazione>

# Appendice

## Button Debouncer

### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>

## Divisore di frequenza

### Progetto e architettura

<descrizione dell’approccio di progetto utilizzato, disegno architetturale del sistema e dei suoi componenti, descrizione delle funzionalità>

### Implementazione

<codice VHDL dei componenti significativi: componenti elementari riutilizzati in diversi progetti vanno inseriti in un’appendice>