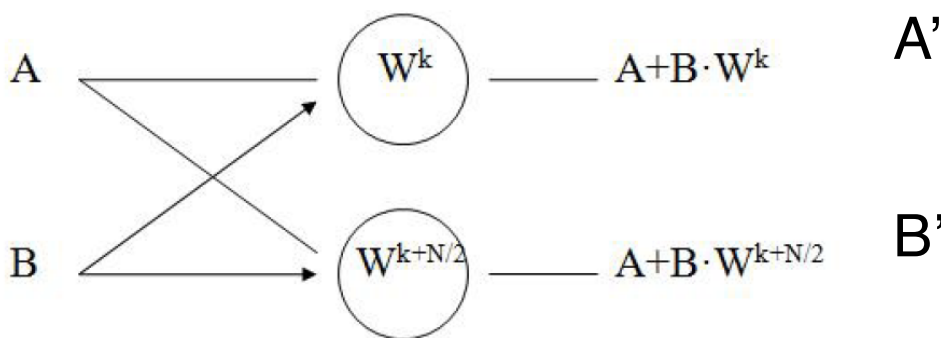


# Sistemi Digitali Integrati 2023-2024

## Esercitazione finale di progetto sulle architetture integrate

Si progetti un'unità di elaborazione che esegua la FFT basata sul processore Butterfly (elemento base per realizzare una FFT classica).

La Butterfly consiste nell'effettuare somme e moltiplicazioni sui numeri complessi  $A$ ,  $B$  e  $W^k$ , ( $A_r + jA_i$ ,  $B_r + jB_i$ ,  $W_r + jW_i$ ) sulla base dello schema riportato in figura:



Le operazioni da eseguire con la Butterfly sono:

$$A' = A + B \times W^k \quad B' = A - B \times W^k$$

Si supponga che i dati  $A$ ,  $B$ , e  $W^k$  (sia le parti reali sia quelle immaginarie) siano definiti in forma frazionaria ( $-1 < \text{dato} < 1$ ) in complemento a due (C2) su 24 bit e che si usi il metodo del “*Unconditional Block Floating Point Scaling*” che consiste nel non inserire obbligatoriamente bit di guardia in ingresso per evitare l’overflow dei dati (dato che si effettua una moltiplicazione, una somma e una sottrazione complesse si potrebbero avere fino a 2 bit di overflow) ma al termine della Butterfly si effettua uno scalamento sui dati di uscita per poter rientrare nell’intervallo  $-1 +1$  ed essere quindi compatibili con lo stadio successivo.

Ipotizzando che lo scalamento da implementare sia sempre di 1 bit, come occorre progettare il sistema nel caso in cui sia possibile un overflow di due bit nel primo stadio?

Si effettui il progetto della macchina, usando le tecniche della microprogrammazione viste a lezione (anche se un po’ sprecate data la semplicità dell’algoritmo), seguendo i seguenti passi:

1. Si derivi il DFD (Data Flow Diagram) dell’algoritmo e si studi il miglior scheduling (dalla soluzione ASAP a quella ALAP) che ottimizza il timing e l’uso degli operatori. Si utilizzino i passi e le ottimizzazioni viste a lezione/esercitazione.

A tal proposito si ipotizzi di avere un unico moltiplicatore in grado di effettuare sia la moltiplicazione di due dati sia la moltiplicazione di un dato per 2 agendo come uno shifter aritmetico. Un segnale di controllo permette di scegliere tra i due tipi di operazione.

Si supponga che il moltiplicatore abbia due livelli di PIPELINE, ovvero che ad ogni colpo di clock possa iniziare una nuova operazione ma il risultato sarà disponibile non nel ciclo corrente, né in quello successivo ma dopo due cicli di clock (ovviamente occorre tenerne conto nel Control DFD!!!) se si tratta di una moltiplicazione vera mentre invece in caso di shift il risultato sia disponibile dopo un colpo di clock (un livello di PIPE per gli shift); in pratica il blocco può iniziare una nuova operazione ad ogni ciclo in funzione del segnale di controllo, però ha due uscite: una per la moltiplicazione (con latenza due cicli) ed una per lo shift (con un ciclo di latenza).

Per le somme/sottrazioni si ipotizzi di avere separatamente un componente sommatore ed uno sottrattore. Si ipotizzi che entrambi abbiano un livello di PIPELINE interno.

**ATTENZIONE!** Quanto detto sopra vuol dire che voi avete un solo moltiplicatore, un sommatore e un sottrattore a disposizione, mentre nell'esercizio fatto in aula avevamo 2 moltiplicatori e 2 sommatore/sottrattori..

Dato che le moltiplicazioni frazionarie in C2 su  $n$  bit forniscono un'uscita su  $2n-1$  bit (essendo i numeri minori di 1), si studi il datapath del processore ipotizzando che tutte le operazioni interne vengano effettuate senza troncamento, ma si effettui l'arrotondamento solo sul dato di uscita alla butterfly con la tecnica del ROUNDING TO NEAREST EVEN per rientrare nei 24 bit previsti per la parte reale e per quella immaginaria di A' e B'. Data la necessità di inserire dei blocchi aritmetici per effettuare l'arrotondamento, si supponga che sia necessario uno step algoritmico (un colpo di clock) per generare il valore arrotondato a partire dal valore in ingresso da approssimare. Si supponga che per effettuare l'arrotondamento si abbiano a disposizione tutti i blocchi hardware necessari.

2. Utilizzando il tempo di vita delle variabili, si ottimizzi il datapath riducendo al minimo il numero di variabili temporanee senza introdurre nuovi step algoritmici.

3. Si derivi l'architettura, ipotizzando di inserire tutte le variabili di ingresso in una struttura di memorizzazione idonea (registri sparsi, registerfile a  $n$  porte, memoria locale a  $n$  porte...)

4. Si studi come inviare i dati in ingresso dal mondo esterno e come fornire i risultati in uscita (nell'ipotesi che l'unità di elaborazione possa eseguire una elaborazione singola o lavorare a pieno ritmo eseguendo sequenze continue di butterfly) e si proponga un progetto conseguente, motivando le scelte fatte. L'obiettivo del progetto deve essere di utilizzare la stessa sequenza dei dati che vengono inviati in ingresso anche per generare i dati di uscita rendendo interfacciabile l'uscita di un PE con l'ingresso di un altro identico.

**ATTENZIONE!** Si deve pensare a un meccanismo che permetta alla macchina di sapere se è un'esecuzione isolata o se invece è continua per cui occorre ottimizzare il CDFD evitando tempi morti tra una butterfly e l'altra. **Non è previsto alcun segnale esterno di ingresso per discriminare le due condizioni!!!!!!**

Si ipotizzi che i dati A e B vengano inviati su una/più porte diverse da quella/e in cui arrivano  $W_r$  e  $W_i$  che in una generica FFT totalmente parallela potrebbero essere sempre gli stessi per ogni butterfly e quindi costanti.

Per la sincronizzazione dei dati in ingresso e in uscita si studino dei segnali di controllo per far “partire” la macchina (START) e per segnalare in uscita che un’elaborazione è terminata e i dati sono validi (DONE).

5. Si ottimizzi il numero di bus globali (senza penalizzare le prestazioni) nonché il numero di connessioni tra gli elementi del datapath e i bus per minimizzare il costo.

6. Progettato completamente il datapath, si definisca l’unità di controllo mediante la tecnica della microprogrammazione utilizzando un sequenziatore con indirizzamento IMPLICITO. Si progetti il contenuto della microrom in grado di interfacciarsi correttamente con l’unità di esecuzione e con il mondo esterno.

7. Si descriva il progetto completo in VHDL (si consiglia l’ambiente di sviluppo basato su Quartus-Modelsim ma non mi spavento se utilizzate altri ambienti di sviluppo) descrivendo tutti i blocchi (*e’ sufficiente una descrizione a livello comportamentale di ogni singolo elemento*) in modo gerarchico utilizzando i “component” (come a Elettronica dei Sistemi Digitali). Fare attenzione ai package IEEE da utilizzare per il trattamento delle operazioni su numeri in complemento a 2.

8. Si testi il codice prodotto con un numero significativo di vettori di test che assicurino una copertura adeguata alla verifica del funzionamento (e che saranno ovviamente inclusi e giudicati nella relazione finale), dimostrando il corretto funzionamento del processing unit in tutti i casi ritenuti significativi.

9. Si usi la Butterfly progettata per realizzare una FFT  $16 \times 16$  e se ne verifichi il corretto funzionamento. Si facciano le opportune scelte sui dati in ingresso per tenere conto del fatto che nel primo stadio ci potrebbero essere due overflow.

## RISULTATI ATTESI E REGOLE

- E’ richiesta una relazione finale PER GRUPPO che descriva tutti i passi effettuati nel progetto, includendo il codice VHDL, le simulazioni e tutto quanto riterrete opportuno per descrivere il lavoro svolto, i risultati, eventuali criticità, etc.... **NON INCLUDETE IL PROGETTO QUARTUS!!!**

**Ogni gruppo deve essere di norma composto da tre studenti.**

- Sia nella relazione sia nel codice VHDL deve essere riportato il nominativo di tutti i componenti del gruppo.

- **La relazione (COMPRESI I LISTATI VHDL SVILUPPATI) deve essere inviata per mail** a me ([maurizio.zamboni@polito.it](mailto:maurizio.zamboni@polito.it)) ALMENO una settimana prima della data in cui prevedete di sostenere l’esame (e comunque NON OLTRE IL TERMINE DELLA SESSIONE DI ESAMI INVERNALI). **Al momento dell’esame discuteremo insieme sul progetto (oltre a verificare le vostre conoscenze sul VHDL sviluppato!)**

- La valutazione avverrà ovviamente sul progetto ma, in parte, anche sul modo e la completezza con cui si descrive il lavoro svolto (siate degli ottimi venditori!).

ATTENZIONE!!!! Per facilitarmi la ricerca delle mail, l'oggetto della mail deve essere:

**SDI\_2023\_COGNOME1\_COGNOME2\_COGNOME3**

(nell'ipotesi che il gruppo sia fatto da tre persone...)

- Il file inviato via mail (**SDI\_2023\_COGNOME1\_COGNOME2\_COGNOME3.pdf**) dovrà contenere la relazione con in appendice il codice sorgente vhdI (solo il codice!!! non includete il progetto!!). Se la dimensione del file e' non piccola potete zipparlo.
- E' ovviamente severamente vietato copiare il lavoro da altri. In caso di evidente copiatura il lavoro non sara' valutato (sia al "copiante" che al "copiato" ed entrambi saranno "memorizzati" in un database apposito.....).
- Per dubbi, consulenze, consigli sul progetto, etc.. cercatemi in aula o in ufficio tutte le volte che volete (probabilmente definiro' dei momenti in cui raggrupparvi per fare consulenza). Meglio vedersi piu' volte prima dell'esame che tante volte all'esame..... ;)