



# VHDL

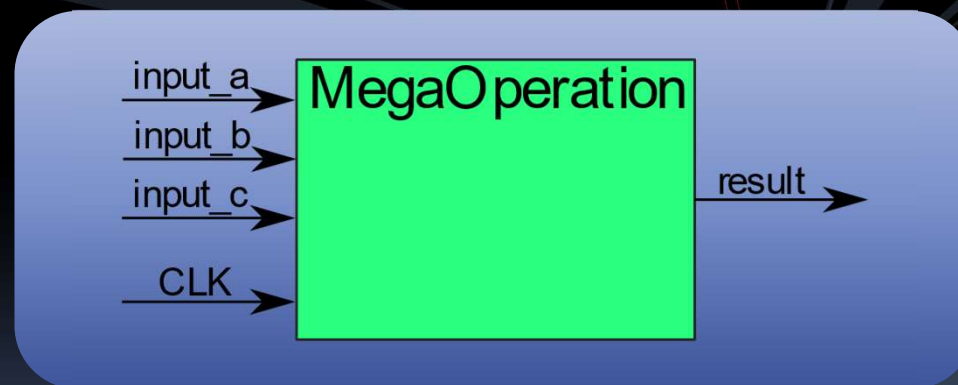
## PIPELINE - EXE

# Testo

Costruire una «entity» che esegua la seguente operazione:

$$result = (input\_a \times input\_b) + input\_c$$

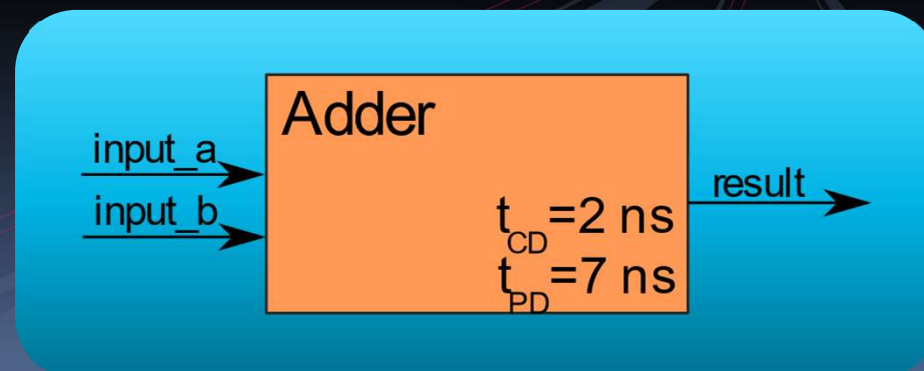
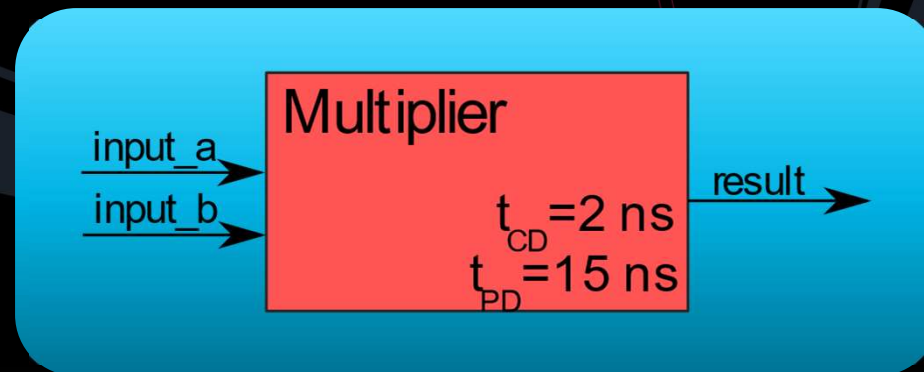
Tramite i moduli in allegato a questo file «multiplier» ed «adder».



Tutti gli ingressi «input\_» e «result» sono «std\_logic\_vector(31 DOWNT0 0)» da trattare come numeri in codifica «unsigned».

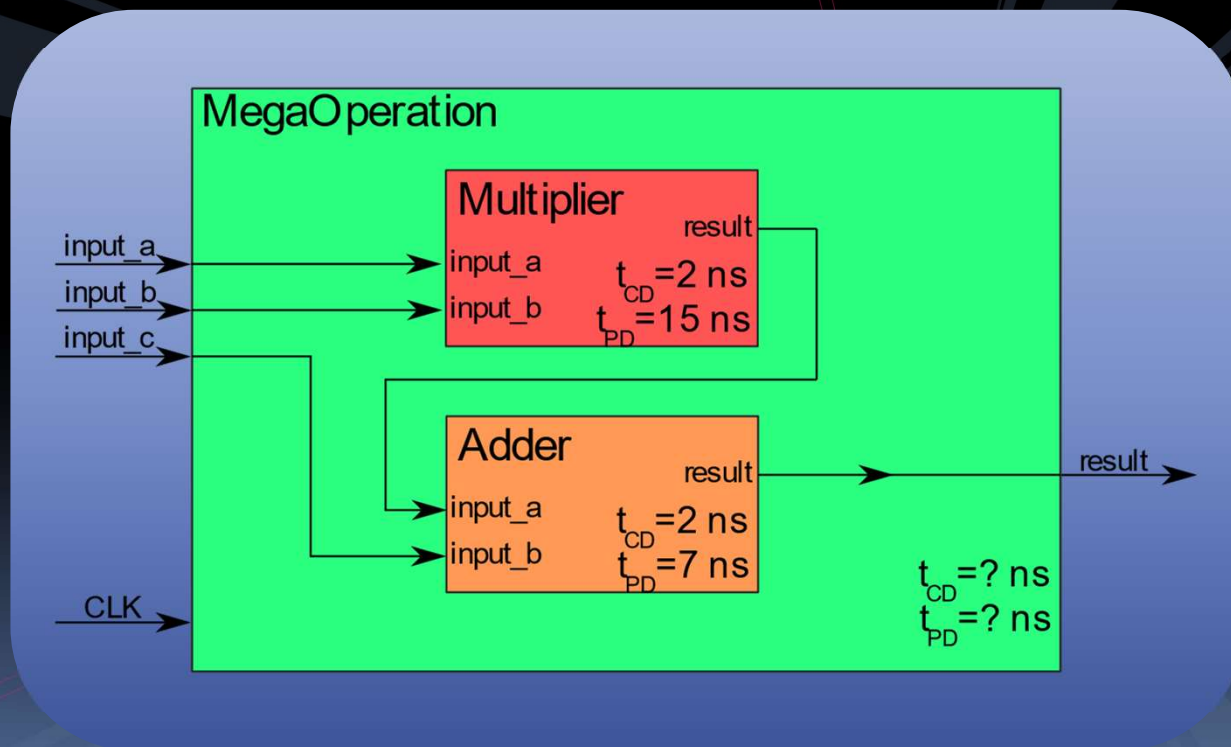
# Testo

Qui i moduli «multiplier» ed «adder» con i loro parametri di propagazione.



# Esercizio 1

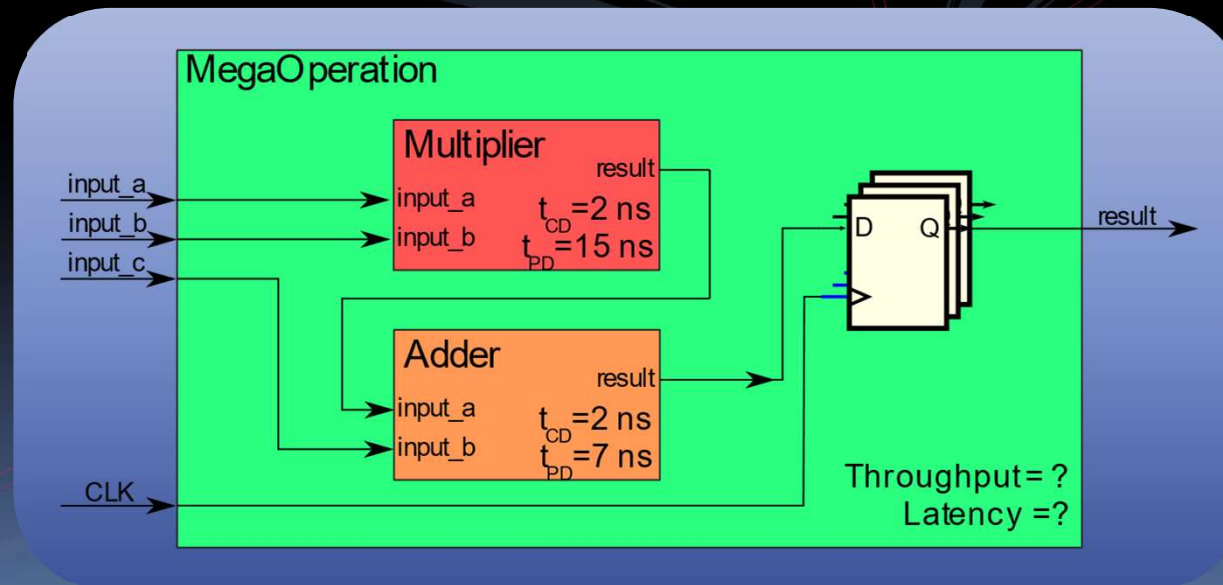
Come primo esercizio provare a simulare la struttura senza strutture di «**pipeline**» e confrontare se i risultati teorici di tempo di contaminazione e propagazione totali vengono rispettati.



## Esercizio 2

Visto che si vuole mantenere i dati nello stesso dominio di clock della sorgente viene chiesto di campionare il dato in uscita con un semplice registro (considerarlo come ideale).


- Qual è la **massima frequenza** del clock?
- Qual è la **latenza** del sistema?
- Qual è il **throughput** del sistema?





## Esercizio 3


Migliorare il sistema tramite **semplici** tecniche di pipeline che permettano di far funzionare il sistema ad una frequenza di clock di 50 MHz.

- Qual è il **massimo tempo di set-up** permesso dai registri che garantisce ancora il funzionamento del circuito?
  - Qual è la **massima frequenza** del clock considerando i registri ideale?
  - Qual è la **latenza** del sistema?
  - Qual è il **throughput** del sistema?
- 



## Esercizio 4

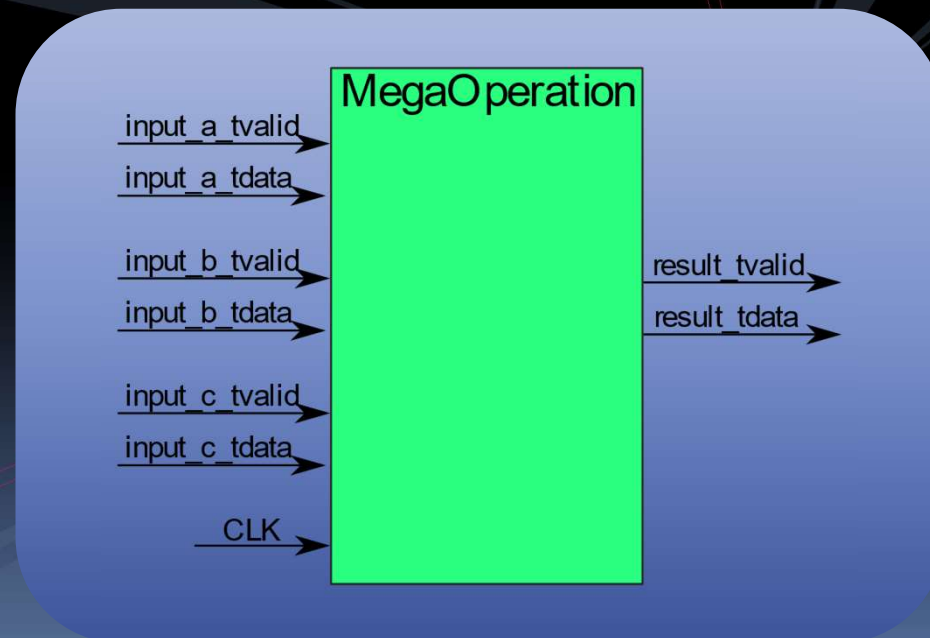
Si vuole spingere il sistema per lavorare a 100 Mhz. Visto che il modulo «**multiplier**» da solo non riuscirebbe a supportare questa frequenza introdurre la tecnica «**dell'interleaving**» utilizzando il minor numero di moduli «**adder**» e «**multiplier**».

- Qual è il **massimo tempo di set-up** permesso dai registri che garantisce ancora il funzionamento del circuito?
  - Qual è la **massima frequenza** del clock considerando i registri ideale?
  - Qual è la **latenza** del sistema?
  - Qual è il **throughput** del sistema?
- 

# Testo Esercizio aggiuntivo

Rifare tutti i punti precedenti considerando gli ingressi e l'uscita della «entity» «MegaOperation» come **BUS AXI Stream** composto dai seguenti segnali:

- \_tdata
- \_tvalid







# Tips

- Tutti i sistemi di pipeline tendono a scambiare cicli di latenza nella propagazione del segnale. Solitamente aggiungere stadi di pipeline nel percorso dei dati equivale aggiungere latenza.
- Può risultare utile durante la simulazione osservare i segnali interni ai moduli come si propagano. Per accederci in simulazione in Vivado seguire i seguenti passi:
  - Selezionare il modulo o sotto modulo di interesse nel menu «Scope»
  - Trascinare il segnale di interesse trovato nel menu «Objects» nella finestra delle «waveform»
  - Riavviare la simulazione (Non eseguirne una nuova)



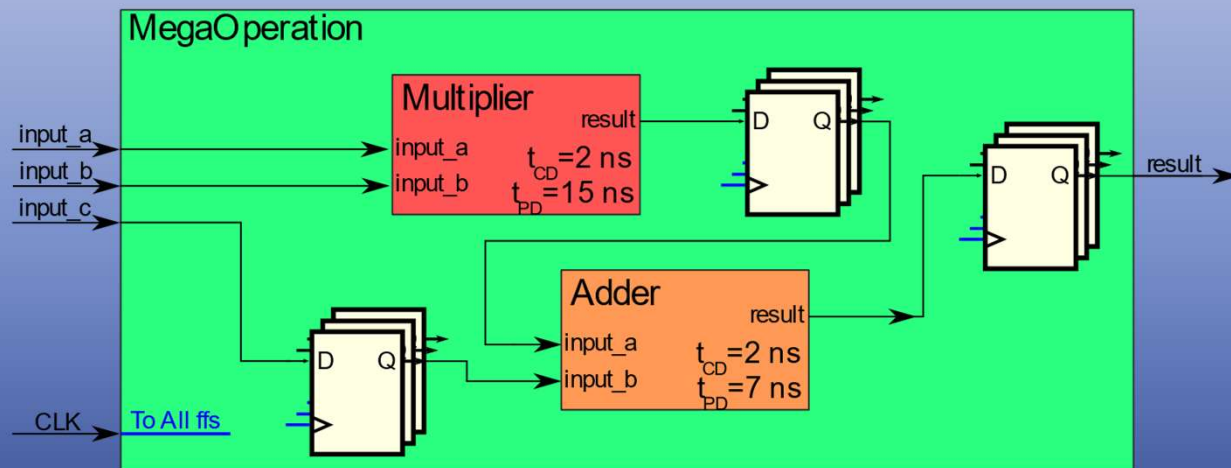
# Consigli

**Attenzione:** di seguito alcune linee guida per arrivare ad una soluzione dell'esercizio.

Consiglio di non leggerli prima di aver pensato autonomamente ad una soluzione.

# Consigli

Fare attenzione quando si creano dei sistemi in pipeline, tutti i dati devono essere opportunamente sincronizzati. In questo caso non è sufficiente aggiungere un registro tra il blocco «multiplier» ed «adder» ma bisogna aggiungerlo anche all'ingresso «dell'adder» in modo da sincronizzare i dati alle sue porte in ingresso.



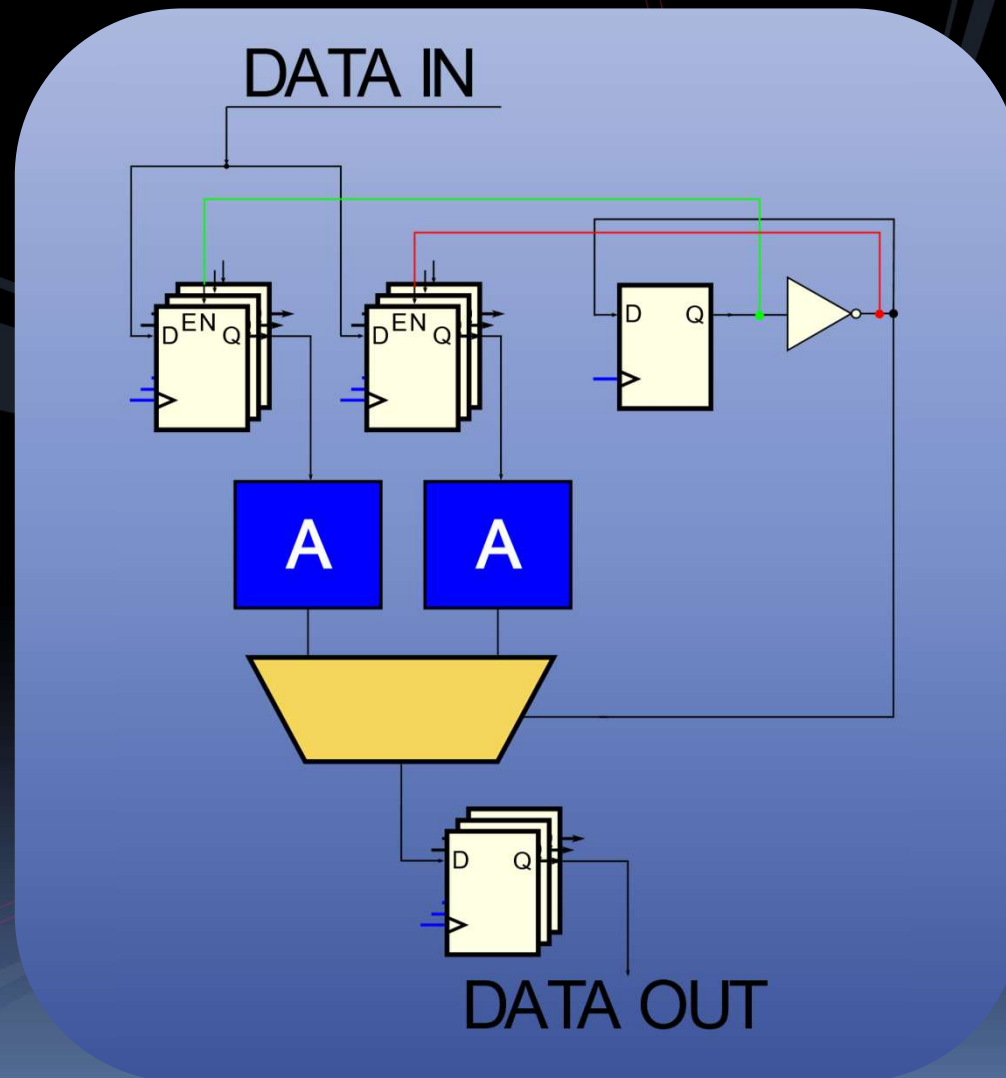


# Consigli

La tecnica «**dell'interleaving**» consiste nell'aggiungere più moduli «lenti» in modo da parallelizzare le operazioni, il risultato finale è un throughput più alto. **ATTENZIONE:** questa tecnica può essere utilizzata solo se le operazioni da eseguire non sono dipendenti l'una dalle altre.

Semplicemente tramite questa tecnica si allunga il tempo disponibile per l'elaborazione di un singolo blocco.

# Consigli





# Consigli

Utilizzare le porte «**AXI Stream**» per scambiarsi i dati vuol dire tenere in considerazione anche quando il dato è valido. Questa informazione va salvata nello stesso flusso dei dati e deve mantenere **la stessa latenza** per evitare che i dati in uscita ed il rispettivo «**\_tvalid**» siano disallineati temporalmente.