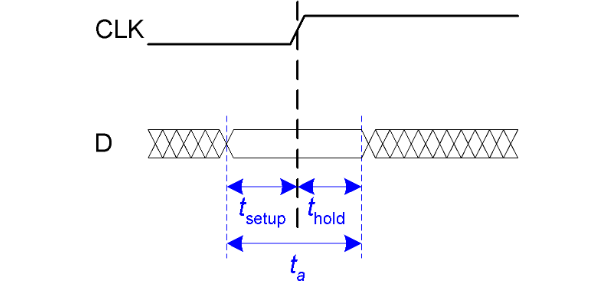
**Timing**

Anche i flip-flop come le porte logiche impiegano del tempo per svolgere il loro lavoro. C’è un istante di tempo in cui il segnale D non è stabile. Per far si che funzioni, il segnale D deve essere stabile sul fronte del CLOCK. Se questo vincolo non viene rispettato, può causarsi uno stato di metastabilità. La finestra di apertura è centrata sul fronte del CLOCK.

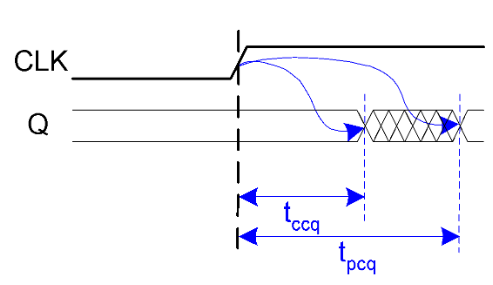
Identifichiamo 3 tempi :

* **Setup time** (): tempo prima del fronte di salita, dove il valore è stabile
* **Hold time** (): tempo dopo il fronte di salita, dove il valore è stabile
* **Aperture time** (): tempo intorno il fronte di apertura

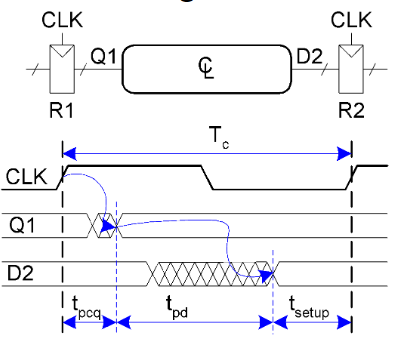


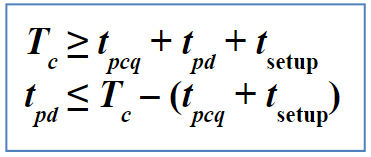
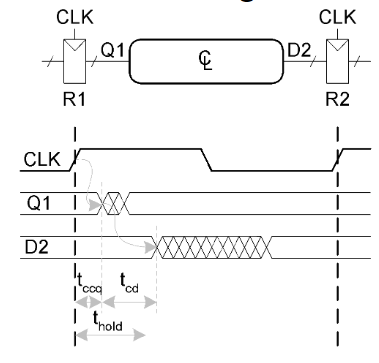
Propagation delay () : Tempo dopo il fronte di salita dove l’output Q è stabile

Contamination delay() : Tempo dopo il fronte di salita dove l’output Q è instabile

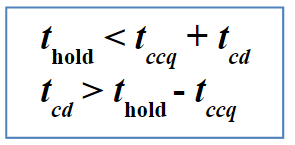


Fra i delay delle registrazioni c’è un minimo ed un massimo, dipende dai delay di ogni elemento del circuito.

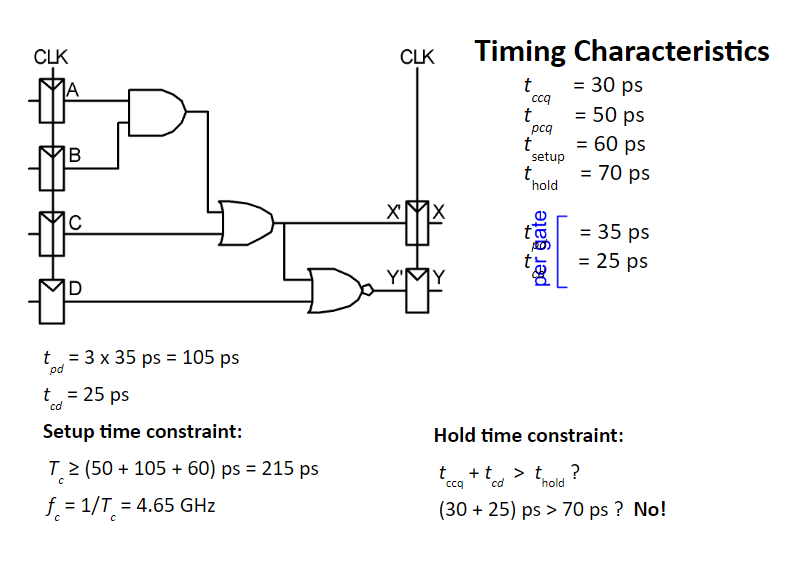
Costrizioni del tempo di setup : dipende dal delay massimo dal registro R1 alla logica combinatoria di R2. L’input al registro R2 deve essere stabile almeno prima del fronte di salita.



Costrizioni del tempo di hold : Dipende dal delay minimo dal registro R1 alla logica combinatoria di R2. L’input al registro R2 deve essere stabile almeno prima del fronte di salita.

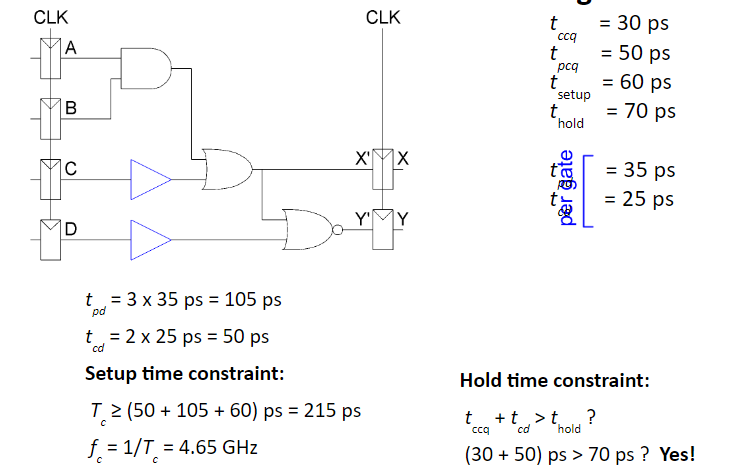


**Analisi del tempo**



Come vediamo il circuito non soddisfa il vincolo del tempo di hold. Per questo dobbiamo fare in modo che la somma del Contamination delay e del siano maggiori del tempo di hold.

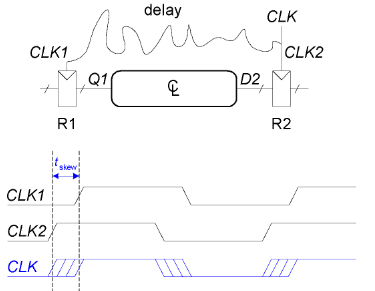
Aggiungiamo quindi al nostro circuito due buffer nei percorsi che partono da C e D :



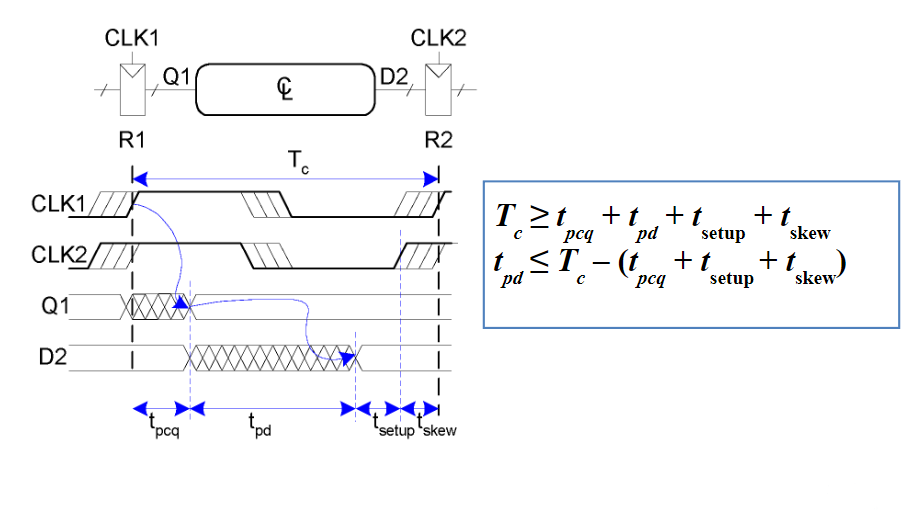
Facendo così il tempo è aumentato, adesso il vincolo è soddisfatto.

**Clock Skew**

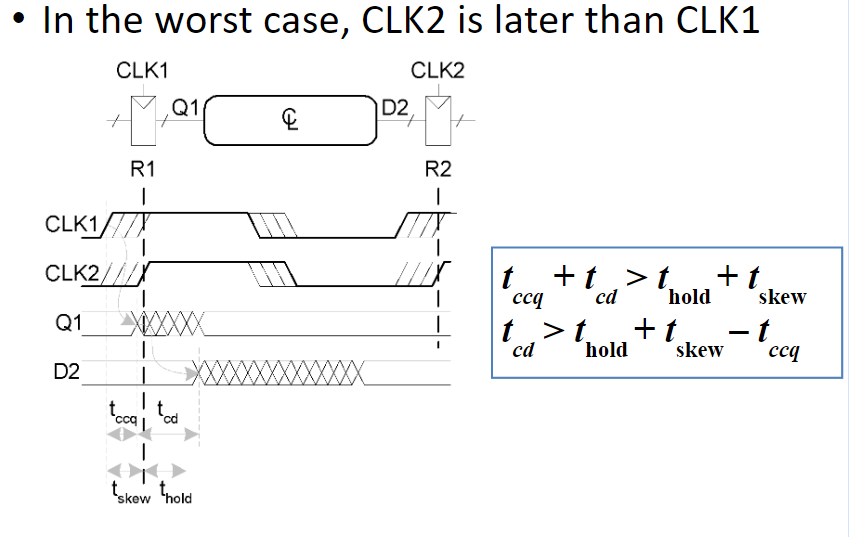
Il CLOCK non raggiunge tutte le registrazioni nello stesso momento, con la parola *Skew* si intende la differenza tra due fronti del CLOCK. Dobbiamo considerare il caso nella quale il clock arriva in istanti diversi:



Parlando di tempo di setup, consideriamo il peggior caso in cui CLK2 arrivi prima di CLK1:



Vediamo la stessa cosa per il tempo di Hold :



**Parallelismo**

Ci sono due tipi di parallelismo, quello spaziale (più hardware eseguono gli stessi compiti una volta) e quello temporale (i compiti sono suddivisi in più sotto-compiti, anche chiamato pipeling)

Definizioni:

* Token : Gruppo di input usati per processare gli output
* Latency : Il tempo di un token per passare dall’inizio alla fine
* Troughput : numeri di token prodotti per ogni unità di tempo

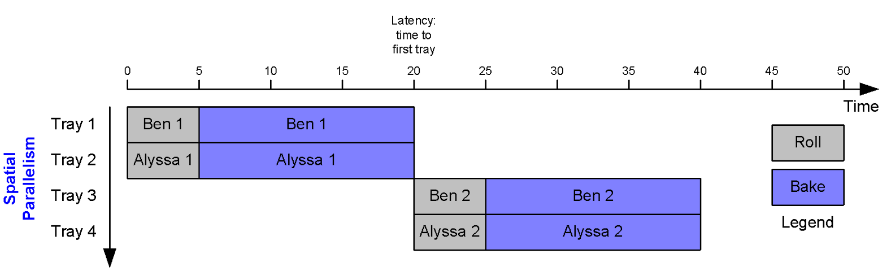
Esempio di parallelismo :

Ben prepara dei biscotti, ci vogliono 5 minuti per preparare l’impasto e 15 minuti per cuocerli, senza parallelismo, la Latency sarà uguale a 20 minuti (5+15), cioè 1/3 di ora. Il Troughput sarà di una teglia di biscotti ogni terzo di ora, quindi 3 teglie all’ora.

Utilizziamo adesso il parallelismo, supponiamo che Ben chieda ad Alyssa una mano, facendo si che 2 persone cucinino insieme i biscotti.

***Caso 1 : Parallelismo spaziale***

Ben e Alyssa preparano e cucinano i biscotti nello stesso momento, ognuno prepara una teglia.



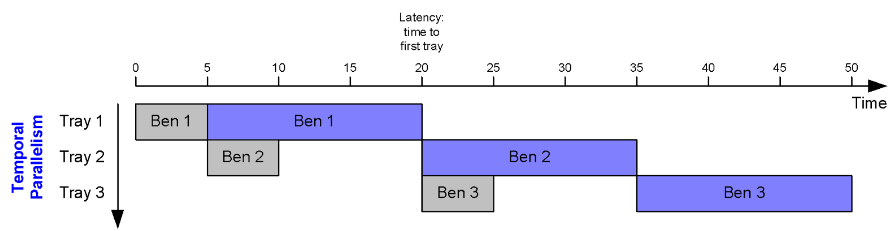
Latency = 5+15 = 20 minuti

Troughput = 2 teglie ogni 20 minuti

Cioè 6 teglie ogni ora

***Caso 2 : Parallelismo temporale***

Si divide il compito in : “*preparare i biscotti*” e “*cucinare i biscotti*” e si usano due teglie, mentre il primo cucina i biscotti, il secondo prepara la seconda teglia.

  
Latency = 5+(10)+15 = 30 minuti

Troughput = 1 teglia ogni quarto d’ora  
Cioè 8 teglie ogni ora