Introduction à la formation

Exemple :

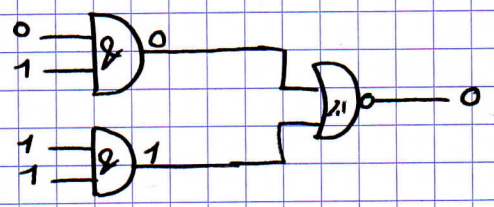
18	=>	0001	0010	=> 0x12
40	=>	0010	1000	=> 0x28
64	=>	0100	0000	=> 0x40

RTL => Register Transfert Level => Niveau le plus bas.

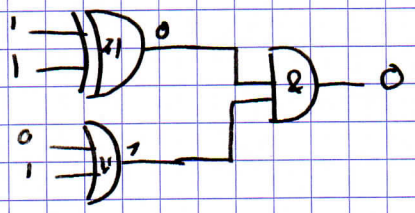
Exercice :

Not [(A & B) or (B & B)]

A = 0
B = 1



(B xor B) & (A or B)



TD : Additioneur 2 bits

A	B	S	R	b(10)
0	0	0	0	0
0	1	1	0	1
1	0	1	0	1
1	1	0	1	2

=> b(10) b(2)

Retenu =>
Resultat => LSB

R S

0 + 0 = 0 => "0 0"
0 + 1 = 1 => "0 1"
1 + 0 = 1 => "0 1"
1 + 1 = 2 => "1 0"

$$\boxed{\begin{aligned} S &= A \text{ xor } B \\ R &= A \,\&\, B \end{aligned}}$$

Cin	A	B	Cout	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S = A \text{ xor } B \text{ xor } Cin = Cin \text{ xor } (A \text{ xor } B)$$

$$Cout = (Cin . A) \text{ or } (Cin . B) \text{ or } (A . B)$$

$$= Cin . (A \text{ or } B) \text{ or } (A . B)$$

en passant par la TD V $\leftarrow$ $= Cin . (A \text{ xor } B) \text{ or } (A . B)$ => simplifier et gagner 1 porte logique que l'on a pas à créer.

Tableau de Karnaugh

On peut regrouper que en puissance de 2

S

Cin \ AB	00	01	11	10
0	0	1	0	1
1	1	0	1	0

XOR

$a \oplus b = \bar{a}b + \bar{a}b = (a+b)(\overline{a+b})$

A = 0
B = 0
A = 1
B = 1
A = 1
B = 0

S = 11 + 00 => 1

S = 00 + 11 => 1

S = 01 + 10 => 0

$$S = \overline{Cin} . \bar{A} B + \overline{Cin} A \bar{B} + Cin \bar{A} . \bar{B} + Cin A . B$$

$$= \overline{Cin} (\bar{A} B + A . \bar{B}) + Cin (\bar{A} \bar{B} + A . B)$$

$$= \overline{Cin} (A \oplus B) + Cin (\overline{A \oplus B})$$

$$= Cin \oplus A \oplus B$$

Cout

Cin \ AB	00	01	11	10
0	0	0	1	0
1	0	1	1	1

$$Cout = A . B +$$

$$= AB + Cin B + Cin A$$

$$= AB + Cin (A + B)$$

$$S = A\,\overline{B}\,\overline{C_{in}} \oplus \overline{A}\,\overline{B}\,C_{in} \oplus A\,B\,C_{in} \oplus \overline{A}\,B\,\overline{C_{in}}$$

$$S = \overline{C_{in}}\,(A\,\overline{B} \oplus \overline{A}\,B) \oplus C_{in}\,(\overline{A}\,\overline{B} \oplus A\,B)$$

TD : Circuit realisant une opération de multiplication par 2 (division par 2) d'une entrée sur 3 bits, sortie sur 4 bits (arrondir ~~supérieur~~ inférieur)

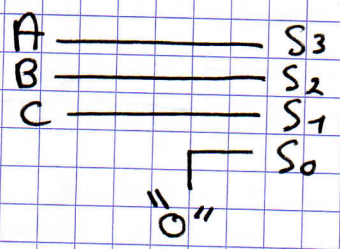
A B C						
0 0 0	0	×	2	⇒	0 0 0 0	0
0 0 1	1	×	2	⇒	0 0 1 0	2
0 1 0	2	×	2	⇒	0 1 0 0	4
0 1 1	3	×	2	⇒	0 1 1 0	6
1 0 0	4	×	2	⇒	1 0 0 0	8
1 0 1	5	×	2	⇒	1 0 1 0	10
1 1 0	6	×	2	⇒	1 1 0 0	12
1 1 1	7	×	2	⇒	1 1 1 0	14

décalage à gauche

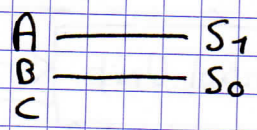
0 0 0	0	/	2	⇒	0 0 0 0	0
0 0 1	1	/	2	⇒	0 0 0 0	0
0 1 0	2	/	2	⇒	0 0 0 1	1
0 1 1	3	/	2	⇒	0 0 0 1	1
1 0 0	4	/	2	⇒	0 0 1 0	2
1 0 1	5	/	2	⇒	0 0 1 0	2
1 1 0	6	/	2	⇒	0 0 1 1	3
1 1 1	7	/	2	⇒	0 0 1 1	3

Décalage à Droite

→ Multiplication par "2"



→ Division par "2"



Analyse des composants dans les cartes

→ Base : - PU (Processeur Unit) } les 2 sont proches
- Mémoire

pour que le signal soit le moins corrompu possible. ⟸ augmenter la cadence de R/W

CPU => fonctionne de manière séquentiel

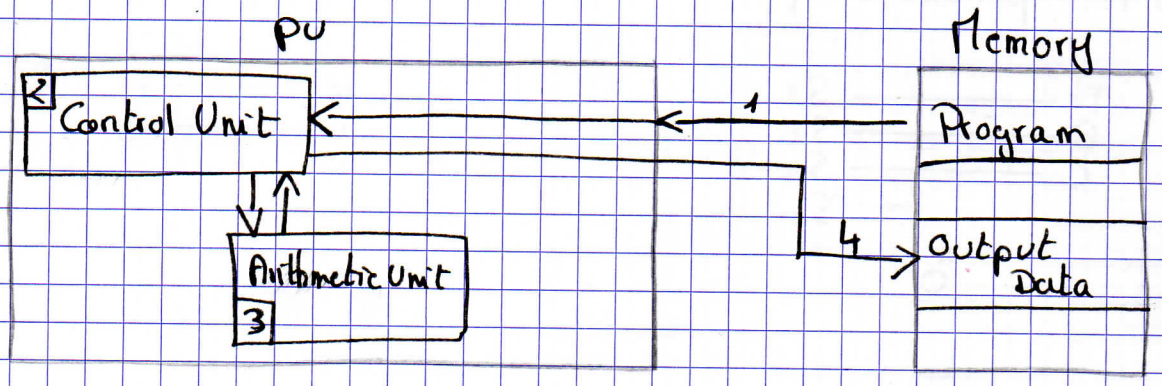
C'est le logiciel qui place les portes logiques pour être le plus efficace possible.

BRAM => Embeded RAM => Mémoire volatile
↳ perd sa mémorisation quand on coupe le courant

↳ interessant pour stocker de la mémoire de façon temporaire.

CPU => est un ensemble de cellule logique

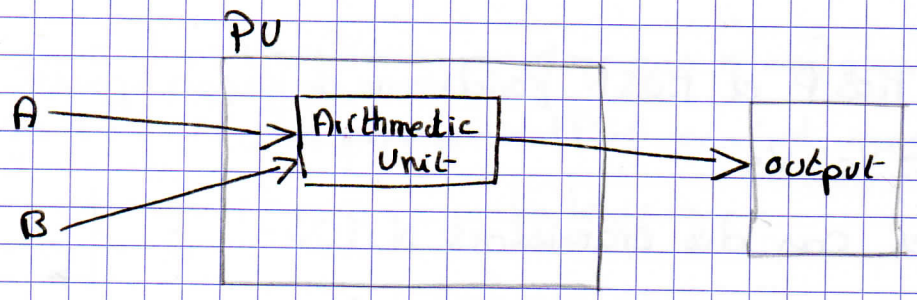
Opération d'un calculateur type CPU, GPU, NC



Si on a plusieurs programme qui font la même chose avec les même ressources, => on aura un problème de CPU

En FPGA, on est pas soumis à la logique sequentielle, on peut faire des calculs dit "à la volée"

Mode peu versatile, mais on a une faible consomation energétique. la latence est courte, car l'execution ne dépend pas de l'accès à la mémoire.



Chemin Critique ⇒ chemin le plus long

Generalement, on cherche à obtenir la latence la plus courte, d'ou l'effort de réduction d'équations logiques.

Dès qu'on le peut, il faut essayer de Paralleliser les traitements. la latence serai la même, mais on utiliserait plus de ressources dans le Process Unit.

Il y a 4 LUT dans un Slice
↳ look up table

Slice ? c'est un regroupement de composant bas niveau.

Chaque FPGA à sa propre architecture avec des perfs persos

LUT : permet de reproduire le comportement d'une porte logique.

Carry4 : optimiser les fonctions de comptage

Site : fpga4fun.com ⇒ site avec beaucoup d'exemples bien expliqués.

Le MOSFET

Transistor MOS est le composant la base de tout circuit numérique.

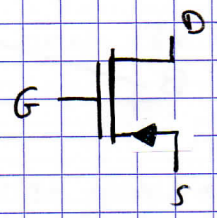
↳ CF Pdf pour plus d'info. (MOS)
↳ Slide p 109/110

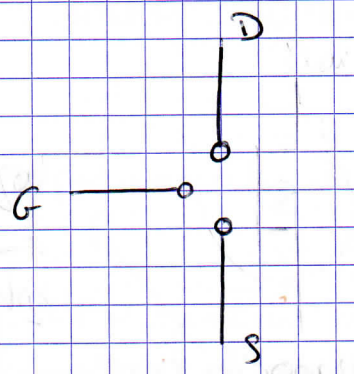
MOSFET a 3 broches, il y a 2 types : mode P : MOS P
mode N : MOS N

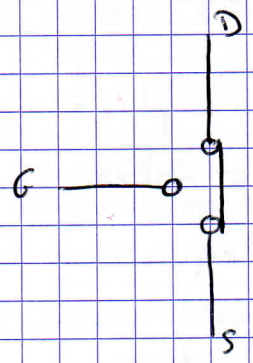
Les MOS P et MOS N fonctionne de façon opposé.

$\overrightarrow{V_{GS}} = V_G - V_S$

- Pour les cas des transistors NMOS :



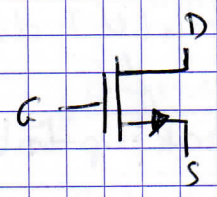


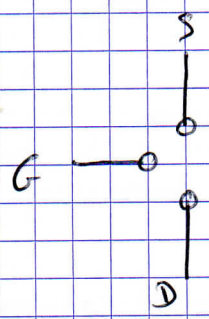


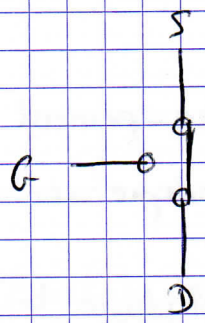
NMOS bloquant quand : $V_{GS} < V_{th}$

NMOS passant quand $V_{GS} > V_{th}$

- Pour les cas des transistors PMOS :







PMOS bloquant quand $V_{GS} > V_{th}$

PMOS passant quand $V_{GS} < -V_{th}$

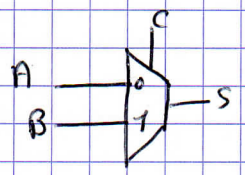
Ce sont les LUT qui permettent la reconfiguration du composant physiquement parlant. Une LUT est une nanomémoire qui contient la TdV d'un circuit.

La LUT, étant une mémoire, possède un bus d'adresse sur lequel on connecte nos entrées.

ASIC => gravé dans le silicium => donc non reconfigurable.

Avant de passer à l'ASIC, on passe par le FPGA car c'est reconfigurable et on peut définir vraiment le besoin en portes logiques.

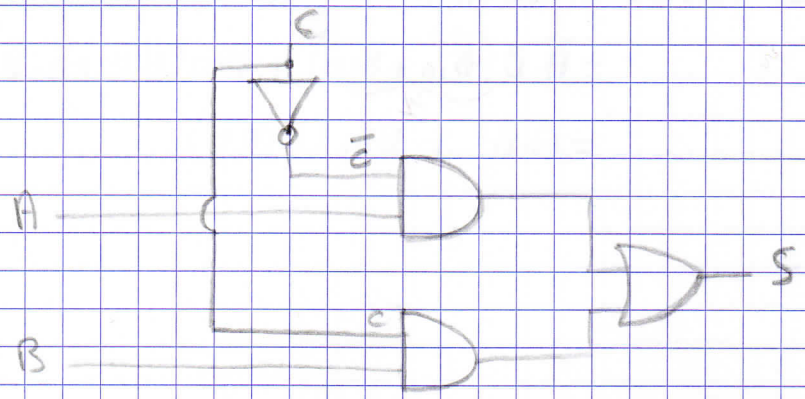
Comment Réaliser un mux à 2 entrées:



C	A	B	S
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

S	AB			
C	00	01	11	10
0	0	0	1	1
1	0	1	1	0

$$S = \bar{C} \cdot A + C B$$



Full Substractor

A	B	Bin	D	Bout
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

$$\begin{array}{r} -\,0 \\ 0 \\ \hline 0 \end{array} \qquad \begin{array}{r} 1 \\ -\,1 \\ \hline 11 \end{array}$$

Bout → D

$$\begin{array}{r} -\,0 \\ 1 \\ \hline 11 \end{array} \qquad \begin{array}{r} -\,1 \\ \hline 10 \end{array}$$

Bout → D

$$\begin{array}{r} 1 \\ -\,0 \\ \hline 1 \end{array} \qquad \begin{array}{r} -\,1 \\ \hline 00 \end{array}$$

$$\begin{array}{r} 1 \\ -\,1 \\ \hline 0 \end{array} \qquad \begin{array}{r} -\,0 \\ \hline 0 \end{array}$$

$$\begin{array}{r} 1 \\ -\,1 \\ \hline 0 \end{array} \qquad \begin{array}{r} -\,1 \\ \hline 11 \end{array}$$

D

Bin \ AB	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$D = A \oplus B \oplus Bin$$

Bout

Bin \ AB	00	01	11	10
0	0	1	0	0
1	1	1	1	0

$$Bout = \bar{A}B + Bin.\bar{A} + Bin.B$$

$$Bout = Bin.\bar{A}.\bar{B} + \bar{Bin}.\bar{A}.B + Bin\,\bar{A}.B + Bin.A.B$$

$$= \bar{A}.B(\underbrace{\overline{Bin} + Bin}_{1}) + Bin(A.B + \bar{A}.\bar{B})$$

$$= \bar{A}.B + Bin\,(\overline{A \oplus B})$$

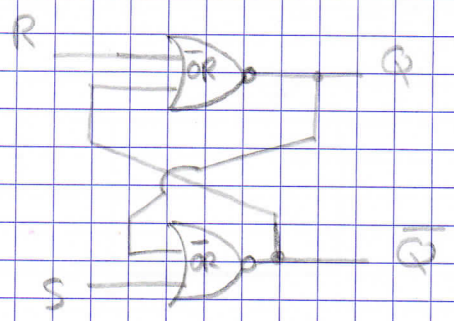
Logique Numérique Synchrone

① La bascule (latch) => Active sur l'état haut de l'horloge

↳ element a 2 états stables et peut être utilisé pour des stocker des informations.

↳ lorsque les 2 entrées sont à '0', la sortie conserve sa valeur d'état precedente.

S	R	Q	$\overline{Q}$
0	0	latch	latch
0	1	0	1
1	0	1	0
1	1	interdit => 0/0	



S_n	Q_n	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	interdit => 0

=> Cas non-synchrone, car on a pas le signal d'horloge.

② Le registre (Flip-Flop) => Triggered : actif au front montant de la CLK.

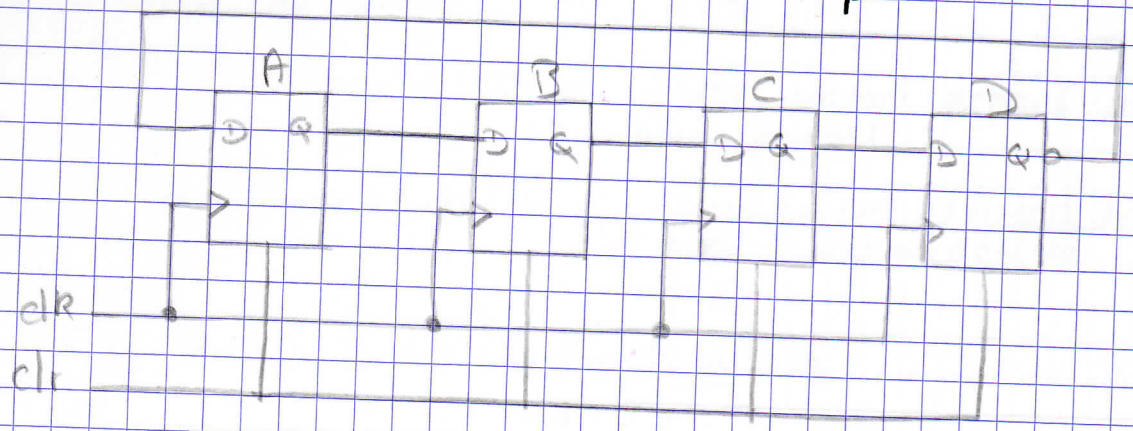
↳ dflop sous LT Spice

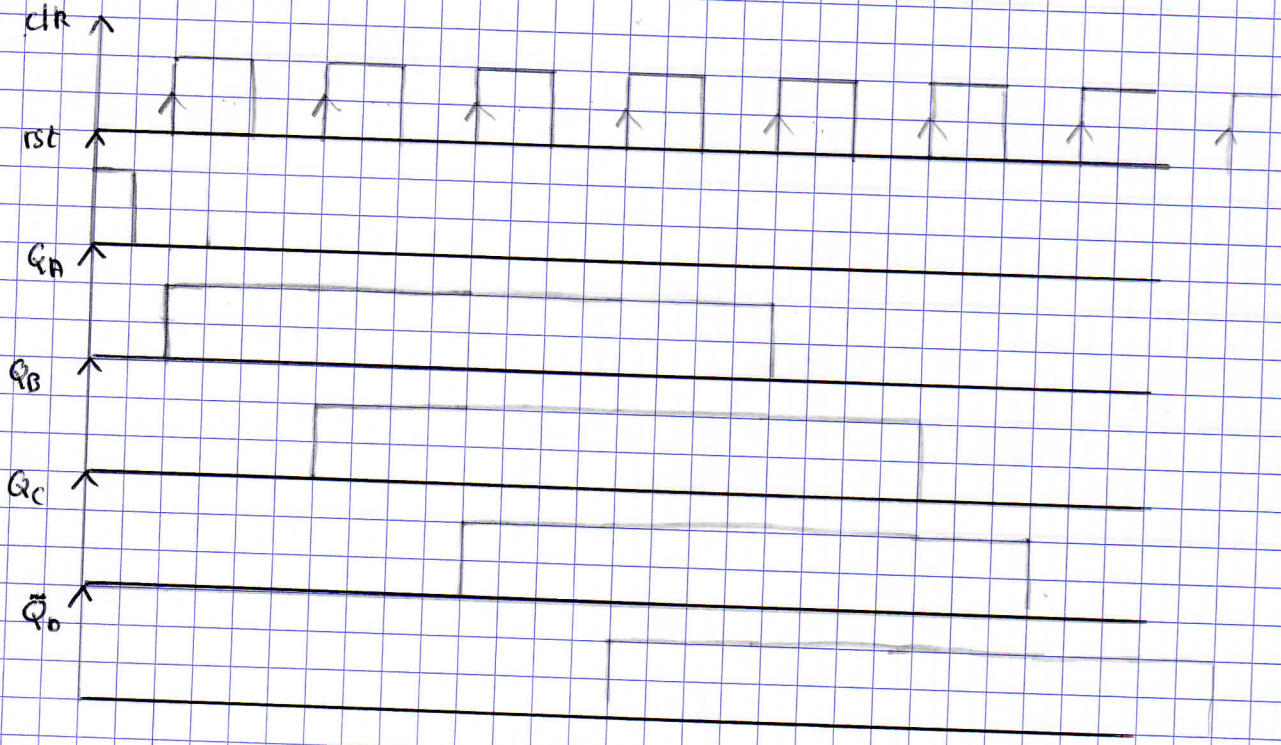
clk	D	Q
↗	d	D
↘ / 1 / 0	X	Q_{n-1} => mémorisation

③ Les compteurs

↳ permet d'interrompre un processus lorsque ce dernier ne reponds pas au bout d'un moment
↳ watchdog.

compteurs à 4 bits. avec un D. Flop

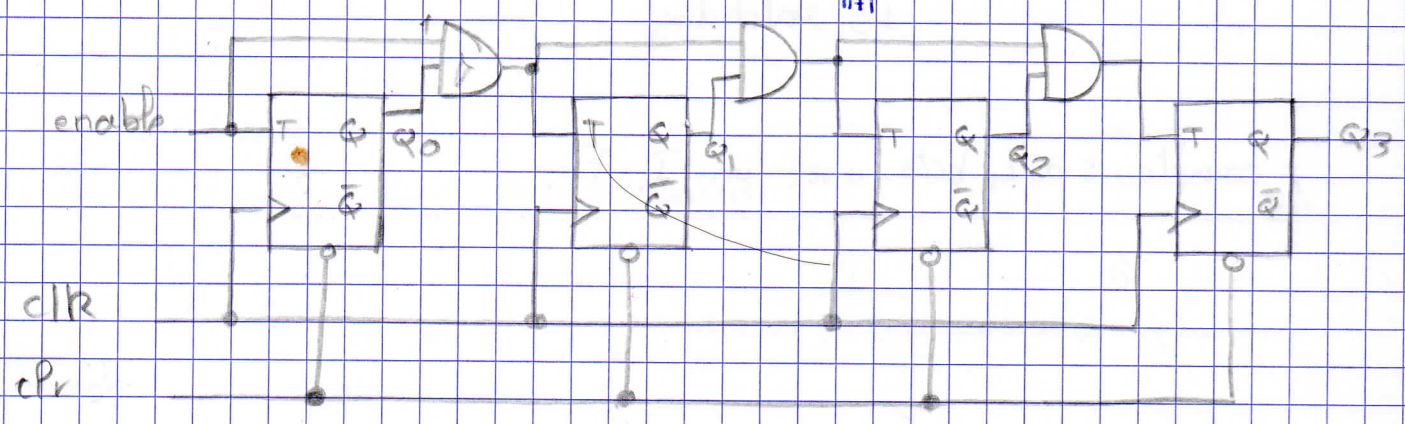


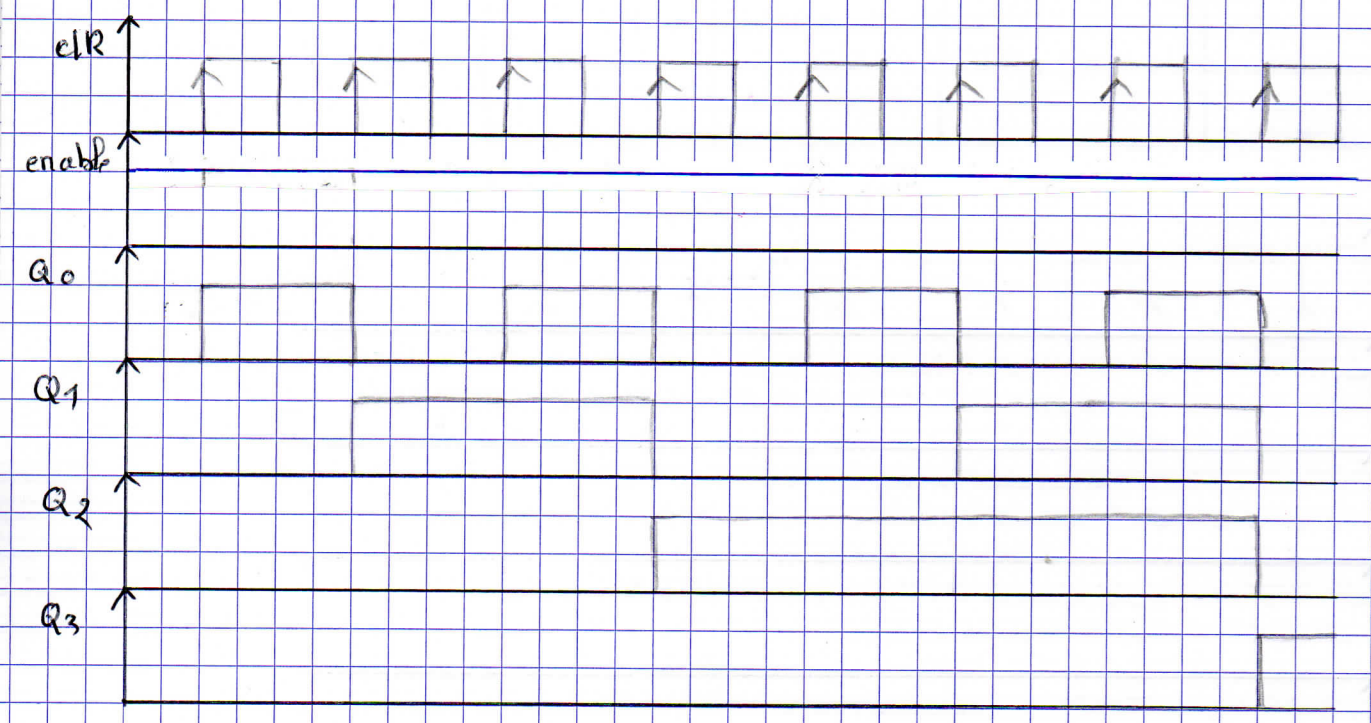


Compteur 4 bits. (Ripple Counter)

↳ réaliser le chronogramme du schéma

→ Sur une T-Flop, si T est actif alors $Q_{n+1} <= not(Q_n)$





④ La machine à état

↳ définie par un ensemble fini d'états possibles, ainsi que par les événements ou les conditions qui déclenchent la transition d'un état à un autre.

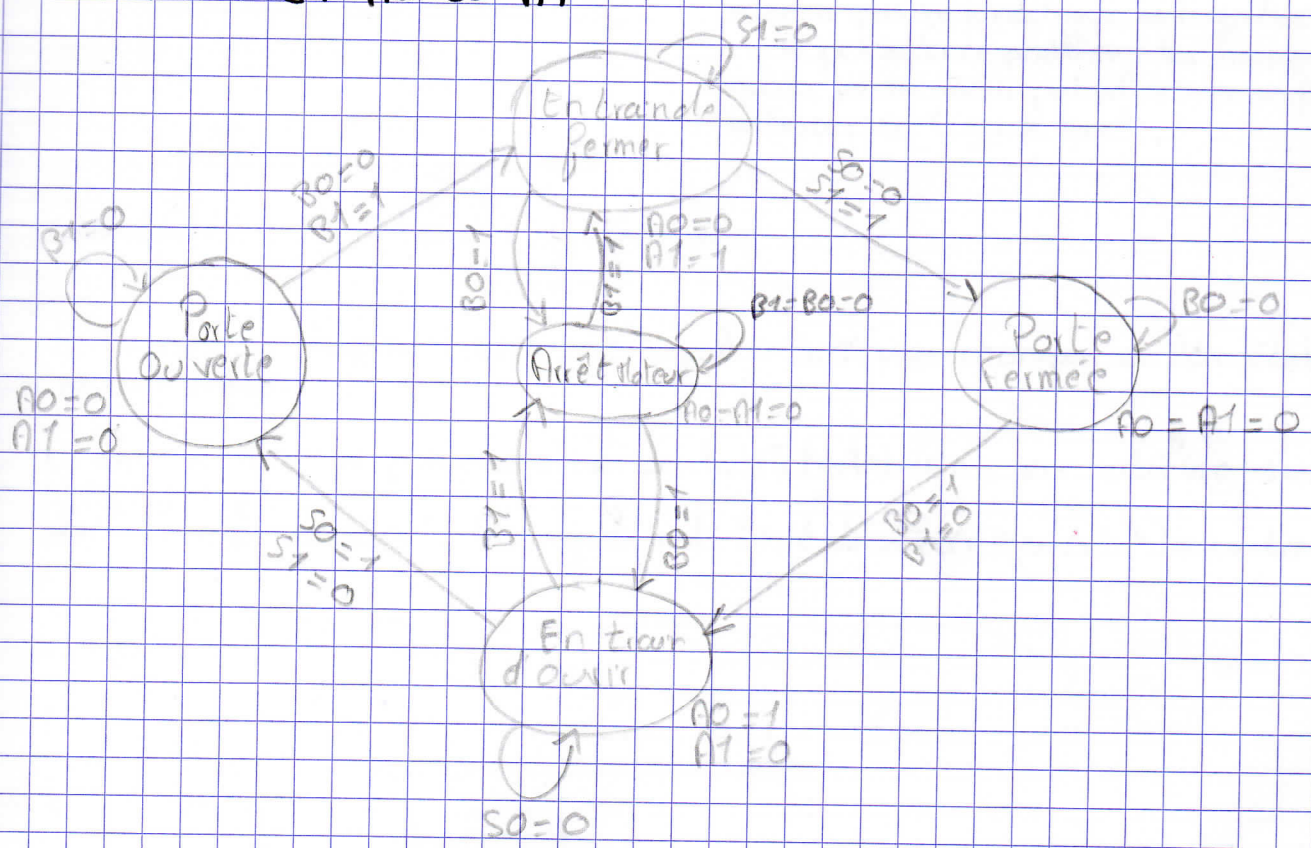
↳ FSM : Fine State Machine

Il existe 2 types :

- Machine de Moore : l'état suivant dépend uniquement de l'état courant.
- Machine de Mealy : l'état suivant dépend de l'état courant et aussi de signaux d'entrées.

exercice de la page 163 du PowerPoint

- entrées : B0 et B1 ⇒ bouton poussoir
- sorties : A0 et A1



Exercice de la page 167 du powerpoint

→ Etat de la machine :
- attente du 1er chiffre
- attente du 2ème chiffre
- attente du 3ème chiffre
- attente du 4ème chiffre
- Comparaison avec le code
- Ouverture porte
- fermeture porte

→ . Entrées : bouton Btn (0 à 9)

. Sorties : Ouverture Porte Op

. Signaux Internes : C_0, C_1, C_2 C_3 ⇒ Chiffre tapé par utilisateur (4 bits)
Tempo de 10sec ⇒ T_v
Lancement du compteur ⇒ Cpt

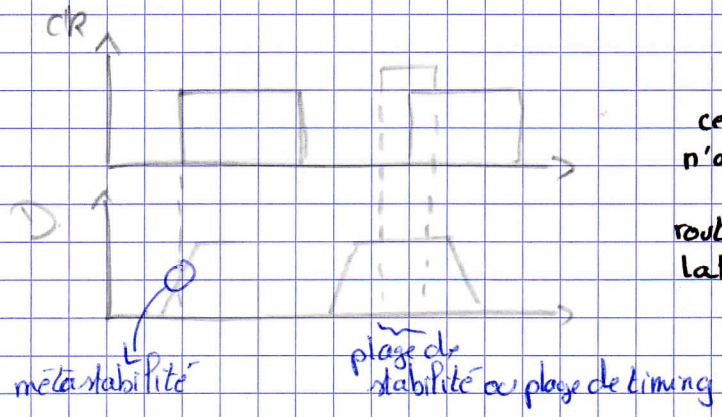
Gestion de timing

si on ignore ce phénomène, il est possible que le signal de sortie de la logique soit en état transitoire au moment où on effectue l'échantillonage.

La conséquence est un signal non stable dans le registre de sortie. On dit qu'il est Métastable.

Dans ce cas, les transistors sont "à moitié" ouvert ou fermé, le signal est dans un état non logique.

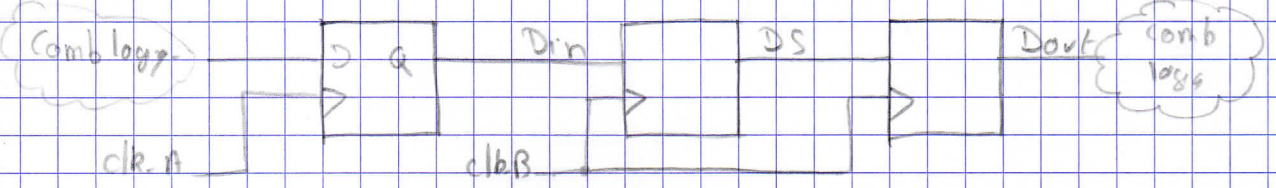
" Tout signal, en entrée d'un registre, doit être dans un état stable durant une plage de temps s'étalant avant et après le front montant d'horloge."

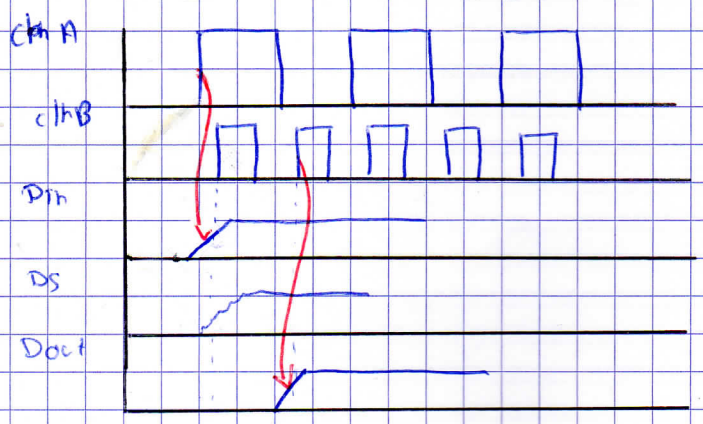


Clock Skew => horloge desynchronisé
cela se produit quand le signal d'horloge n'arrive pas en même temps aux registres
↳ pr limiter ce problème, les horloges sont routées sur un réseau particulier avec une faible latence. => " clock Tree"

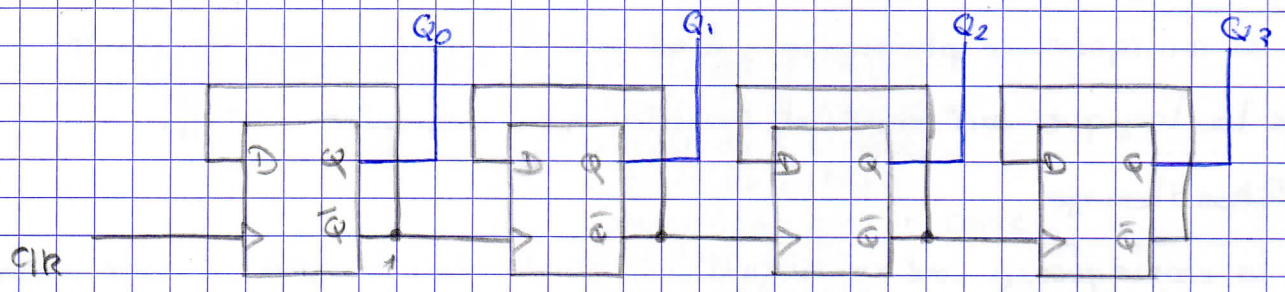
Durant la plage de timing, le signal ne doit pas changer et être établit à un état donné.

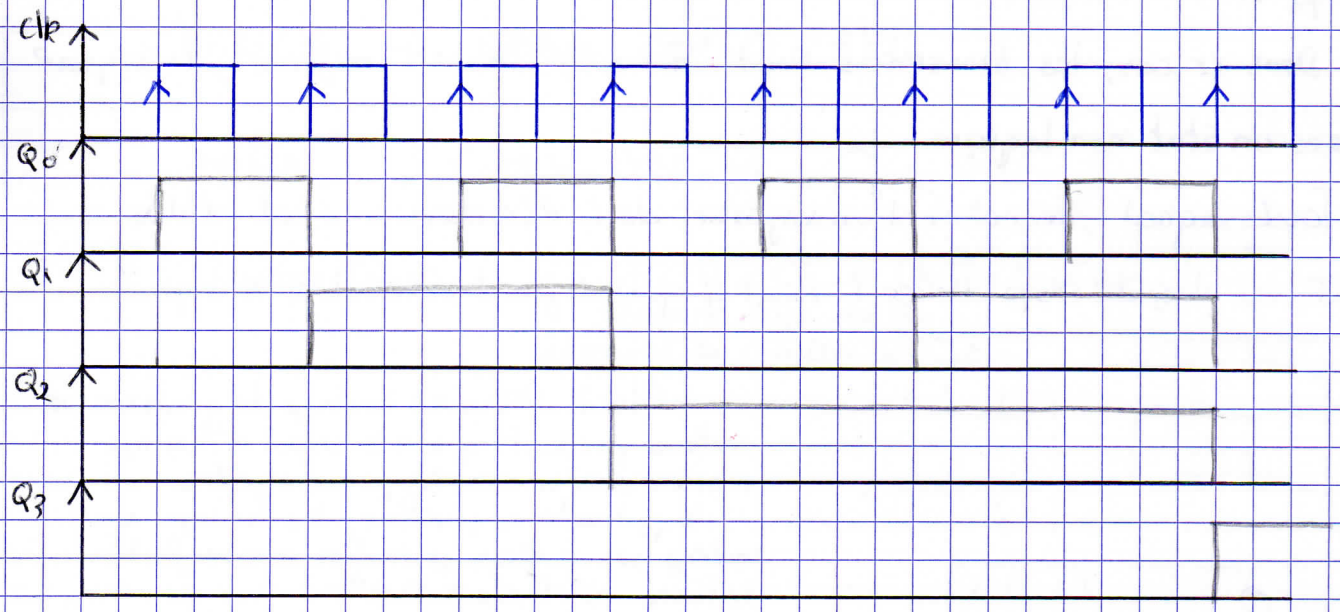
Pour éviter la propagation de la Metastabilité, on utilise souvent un " double rideaux". Un registre est tout de suite ajouté à la sortie.



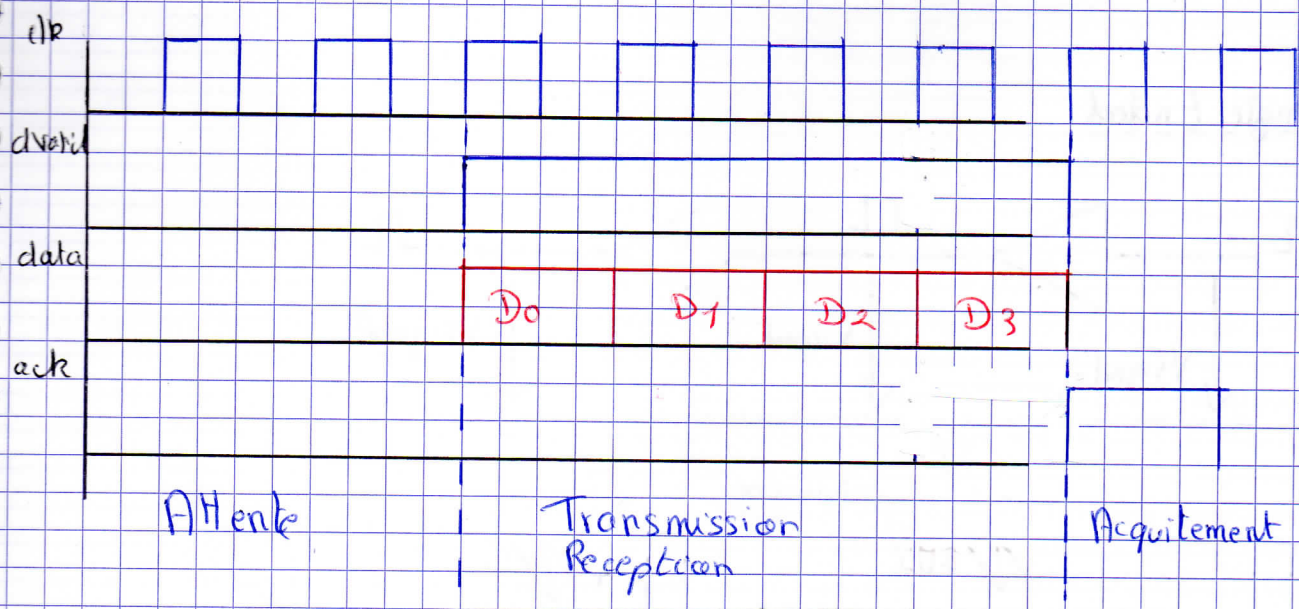


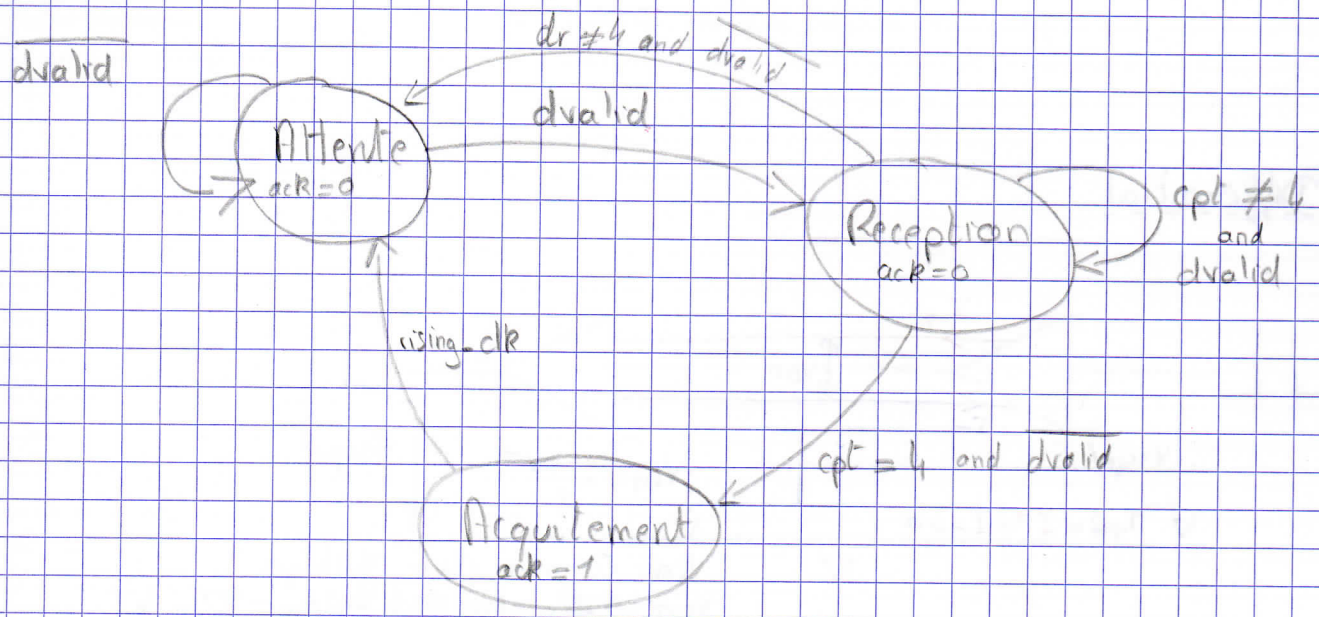
Compteur 4 bits, base de D-flip-flop schéma RTL : Ripple Counter

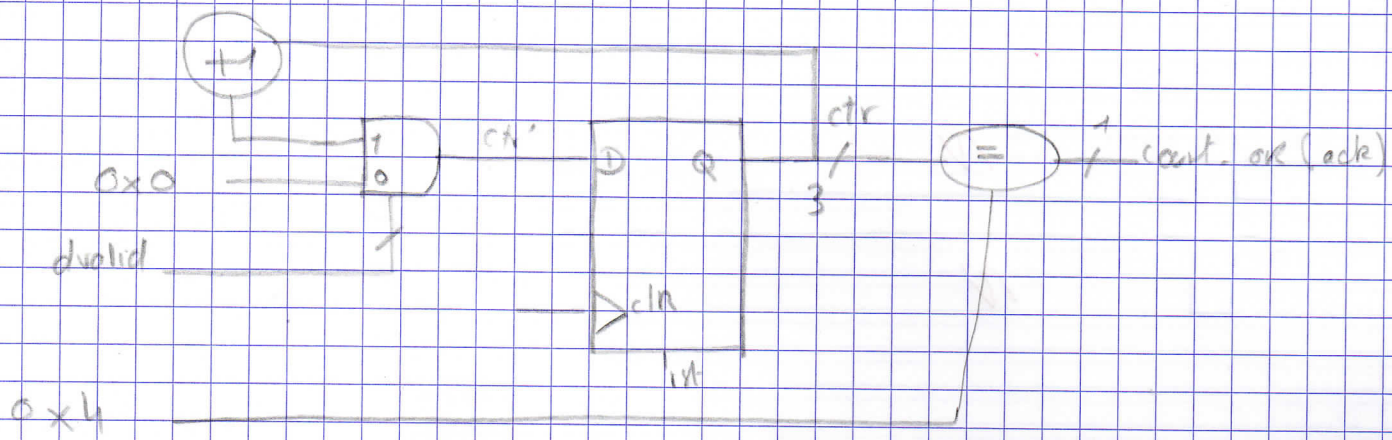




Machine à état. (slide p 177)

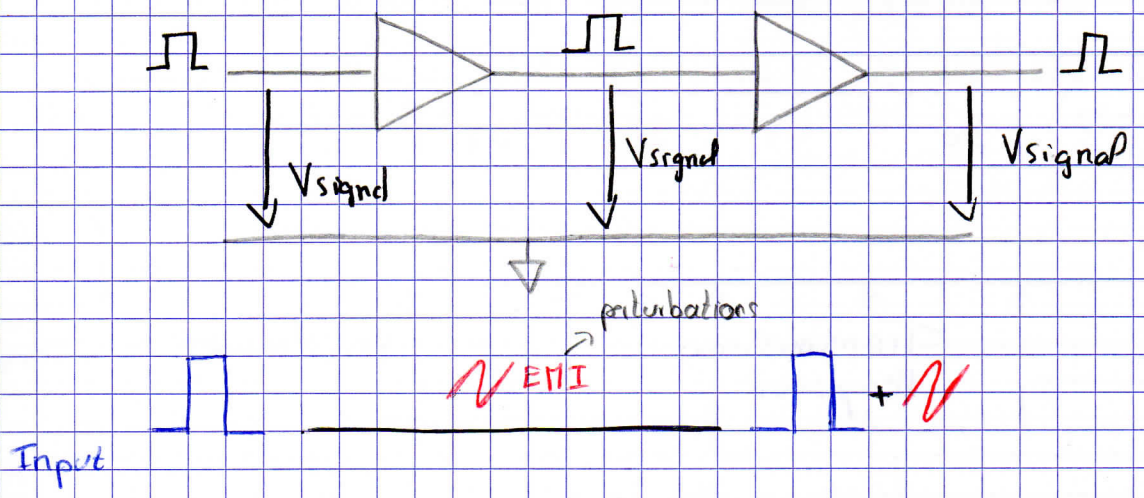
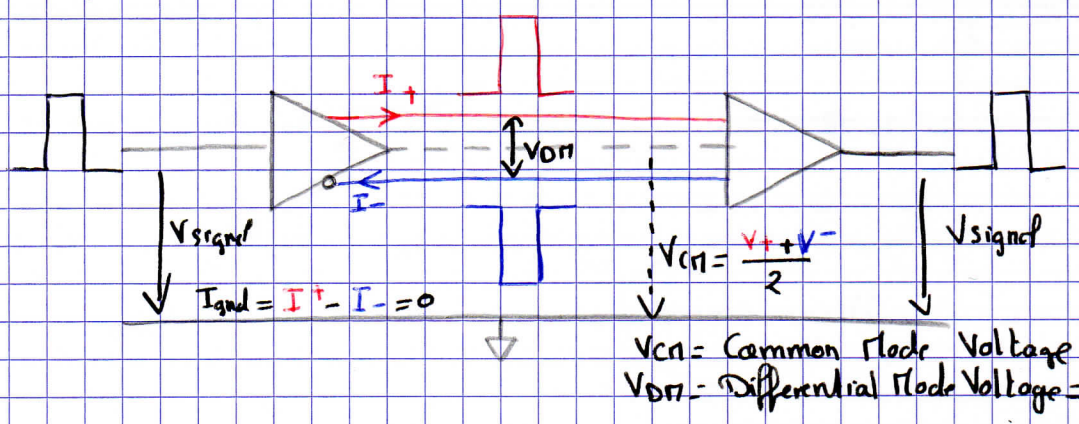






Interface De Transmission

① Single Ended



② Differential