Chapitre 3 :
Caractéristiques des composants logiques de base

Justine Philippe

ISEN | école d'ingénieurs

Sommaire

- Transcodage
- Multiplexeur et démultiplexeur
- Comparateur
- Additionneur et soustracteur
- Bascules

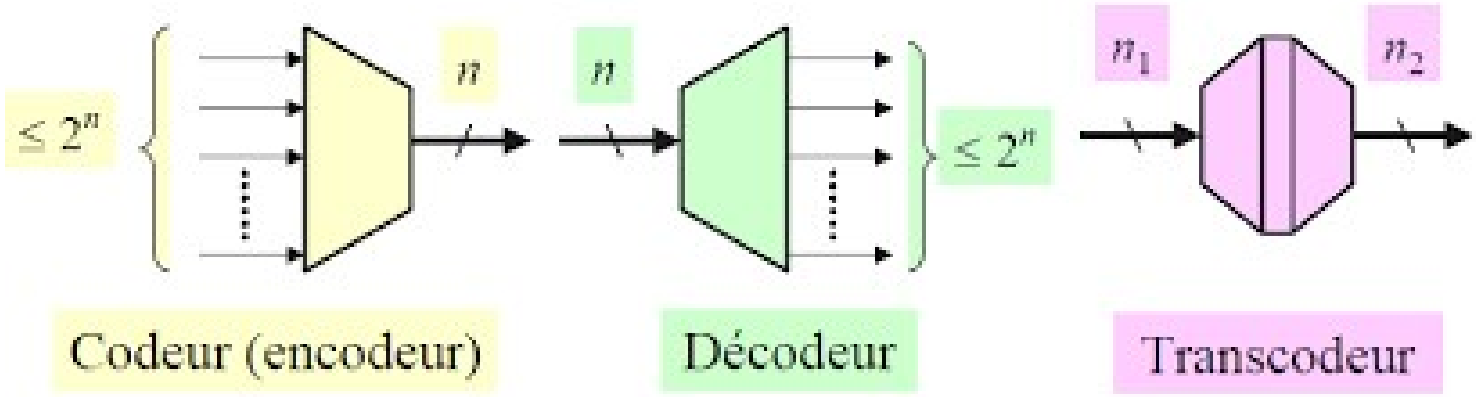
Sommaire

- Transcodage
- Multiplexeur et démultiplexeur
- Comparateur
- Additionneur et soustracteur
- Bascules

Définitions

- Le transcodage permet de transformer une information disponible en entrée sous une forme donnée (généralement un code) en la même information, mais sous une autre forme (généralement un autre code)

- Trois types de transcodeur :

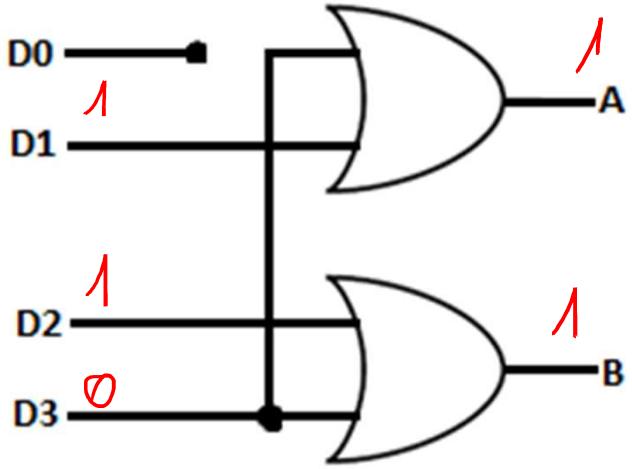


Codeur

- Le codeur (ou encodeur) possède plusieurs entrées, dont une seule est activée à la fois. Il fournit en sortie le code binaire correspondant.
- En général, il possède 2^n entrées et n sorties
- Exemple : codeur élémentaire à 2 bits

$B = D2 + D3$

$A = D1 + D3$

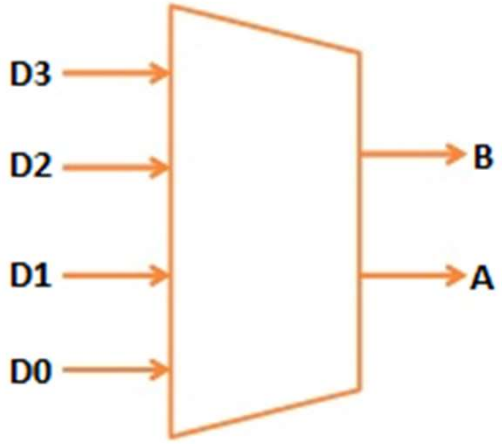


D3	D2	D1	D0	B	A
0	0	0	1	0	0
0	0	1	0	0	1
0	1	0	0	1	0
1	0	0	0	1	1

Codeur

- Pour éviter les conflits, les codeurs fixent généralement un ordre de priorité parmi les entrées. La priorité est habituellement donnée au bit de poids le plus élevé, on parle alors de **codeur prioritaire**

- Exemple : codeur à 2 bits

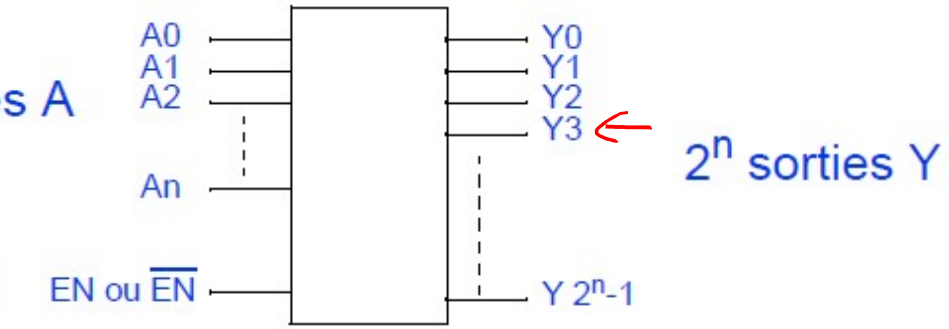


Codeur non prioritaire/prioritaire

D3	D2	D1	D0	B	A
0	0	0	1	0	0
0	0	1	0 X	0	1
0	1	0 X	0 X	1	0
1	0 X	0 X	0 X	1	1

Décodeur simple n vers 2^n

(011)
$A_2 A_1 A_0$



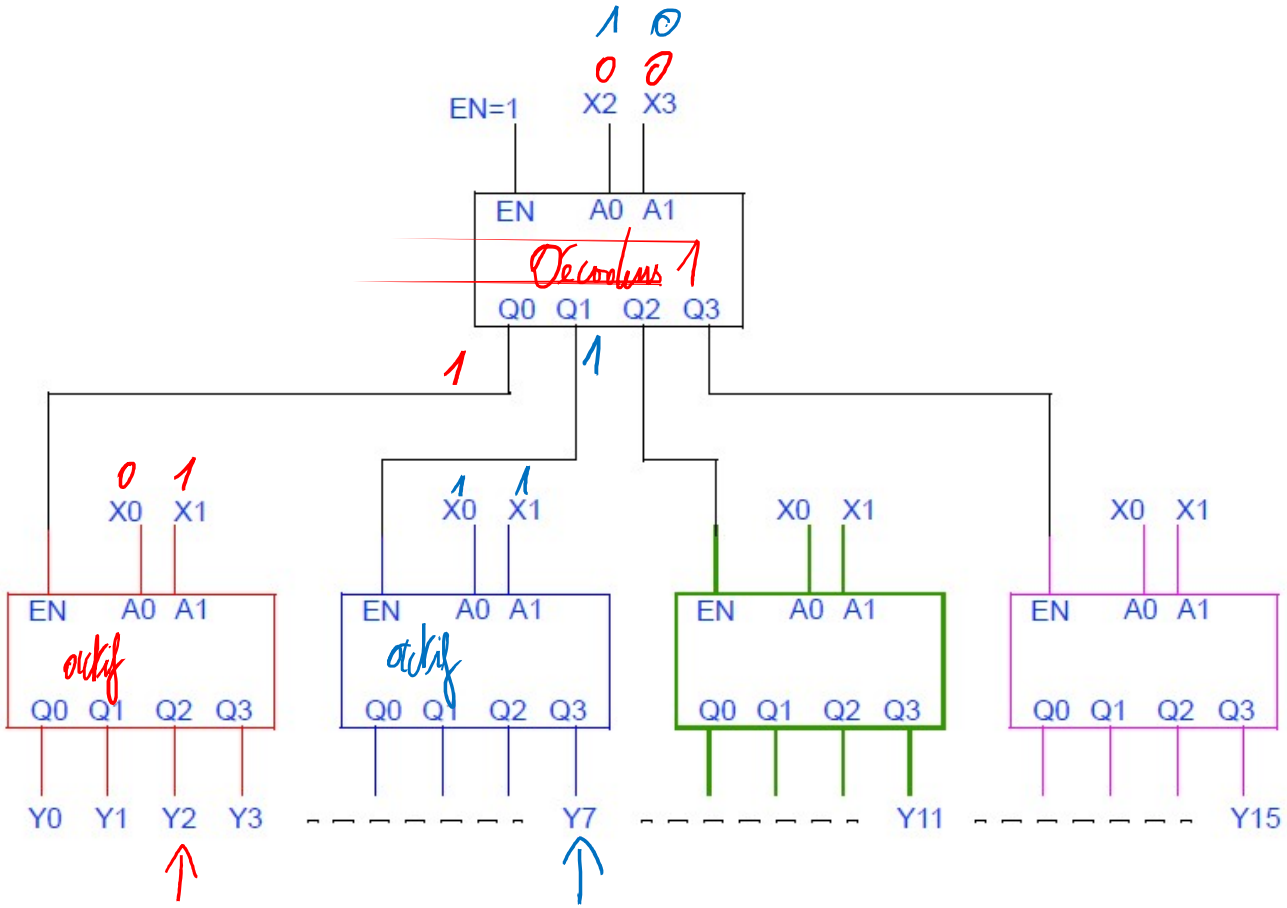
entrée de validation (Enable)

Exemple d'un décodeur 2 vers 4

A0	A1	EN	Y0	Y1	Y2	Y3
X	X	0	0	0	0	0
0	0	1	1	0	0	0
0	1	1	0	1	0	0
1	0	1	0	0	1	0
1	1	1	0	0	0	1

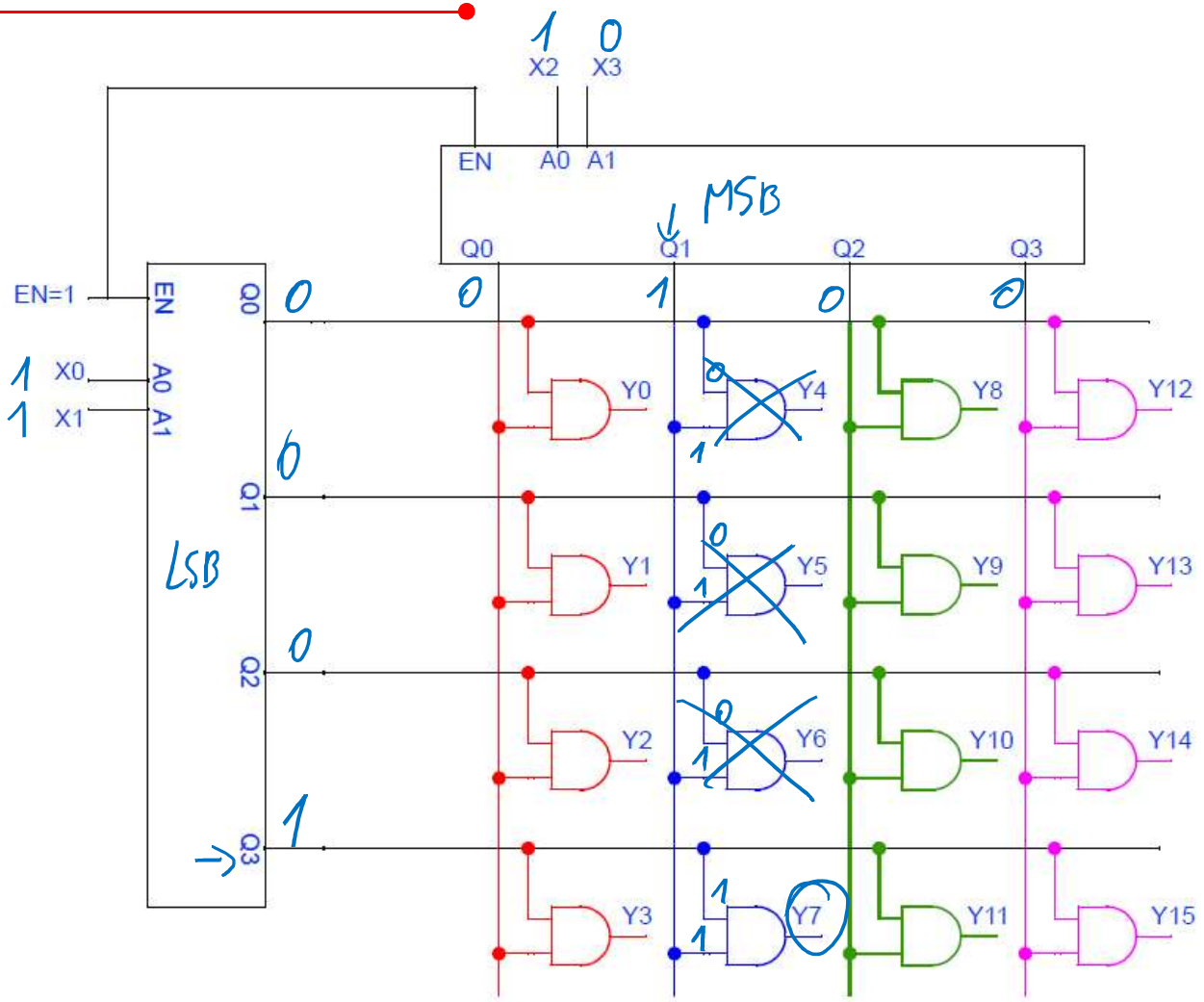
- La sortie Yi concernée est sélectionnée (activée) si le code présent sur les entrées A0..An lui correspond ET si l'entrée de validation (Enable) est active
- Fonctionnement d'une entrée de validation dans le cas général :
  - **Enable activé :** la fonction logique est réalisée, les sorties dépendent de l'état des entrées
  - **Enable non activé :** la fonction logique n'est pas réalisée, les sorties se placent dans un état particulier qui **dépend du type de fonction**. Par exemple, pour un décodeur, cet état particulier est celui pour lequel aucune sortie n'est sélectionnée.

Expansion : décodeur arborescent



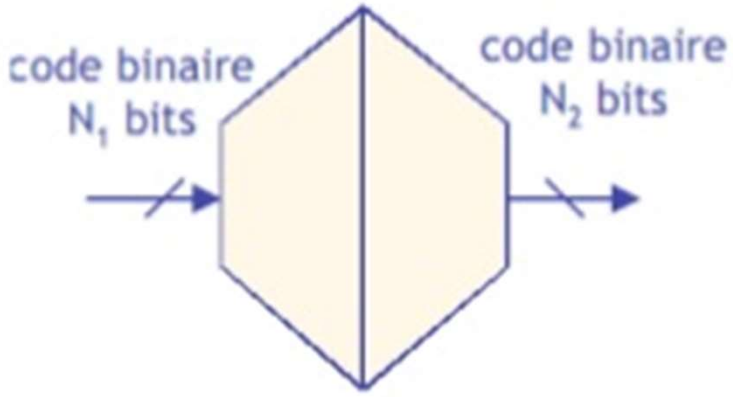
X3	X2	X1	X0	S
0	0	0	0	Y0
0	0	0	1	Y1
0	0	1	0	Y2
0	0	1	1	Y3
0	1	0	0	Y4
0	1	0	1	Y5
0	1	1	0	Y6
0	1	1	1	Y7
1	0	0	0	Y8
1	0	0	1	Y9
1	0	1	0	Y10
1	0	1	1	Y11
1	1	0	0	Y12
1	1	0	1	Y13
1	1	1	0	Y14
1	1	1	1	Y15

Expansion : décodeur coïncident



Transcodeur

- Le transcodeur convertit un code en un autre. On parle aussi de convertisseur de codes. Leurs utilisations en nombre relativement limités expliquent qu'on ne les trouvent pas tous sous forme de circuits intégrés : il faut alors les réaliser à l'aide de portes logiques NON ET, NON OU, etc… La réalisation pratique d'un transcodeur passe par l'écriture de sa table de vérité, puis par la recherche des équations de sorties avec les tableaux de Karnaugh.

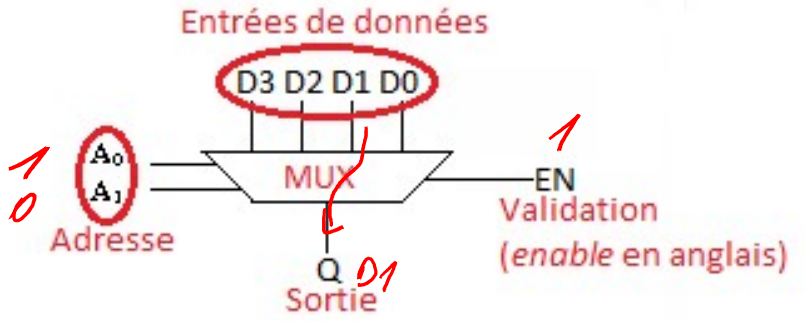


Sommaire

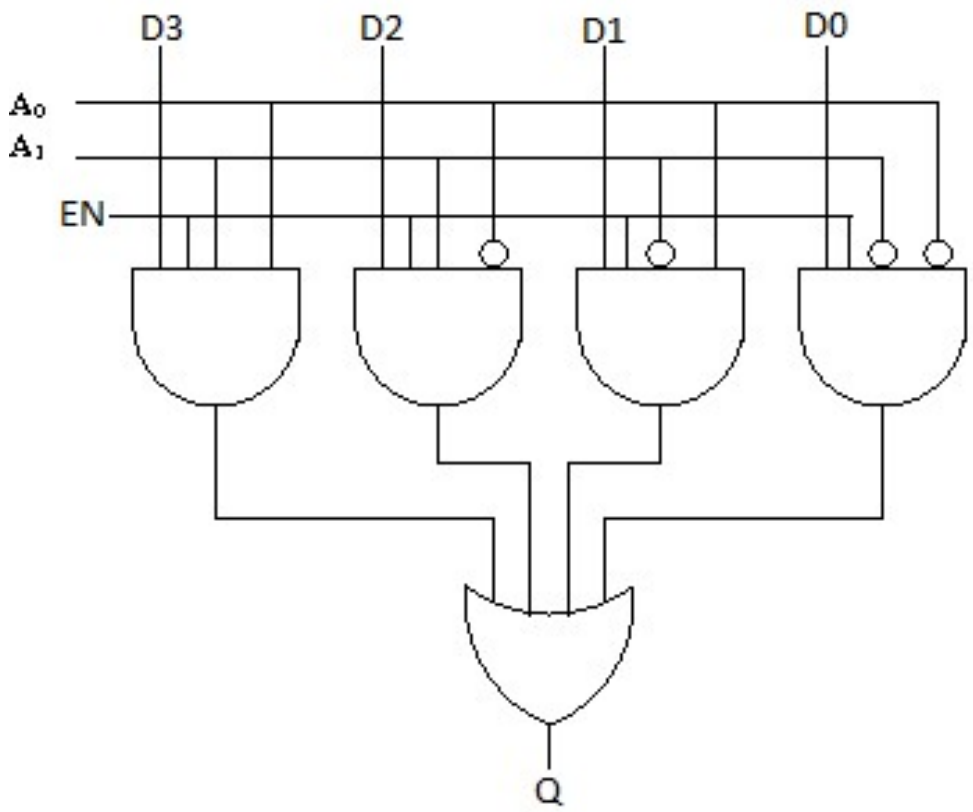
- Transcodage
- Multiplexeur et démultiplexeur
- Comparateur
- Additionneur et soustracteur
- Bascules

Multiplexeur

- Les multiplexeurs, appelés aussi sélecteurs de données, sont des systèmes logiques qui possèdent :
  - 2^n entrées de données
  - une seule sortie
  - n entrées supplémentaires formant **l'adresse binaire** sélectionnant l'une des entrées à recopier en sortie
  - une entrée Enable
- Ces systèmes transmettent ainsi plusieurs données à partir d'un seul dispositif
- Exemple : multiplexeur à 4 entrées, aussi appelé multiplexeur 4 vers 1



Multiplexeur



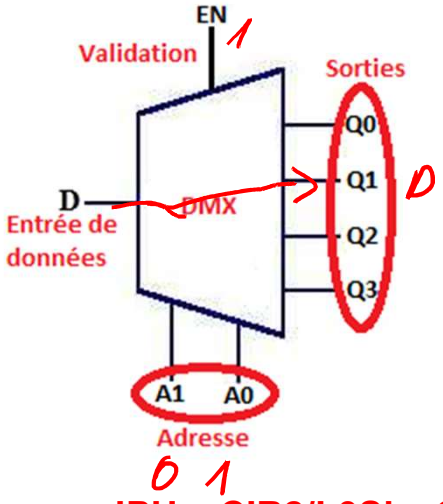
EN	A1	A0	Q
0	X	X	0
1	0	0	D0
1	0	1	D1
1	1	0	D2
1	1	1	D3

$$Q = EN.\overline{A_1}.\overline{A_0}.D_0 + EN.\overline{A_1}.A_0.D_1 + EN.A_1.\overline{A_0}.D_2 + EN.A_1.A_0.D_3$$

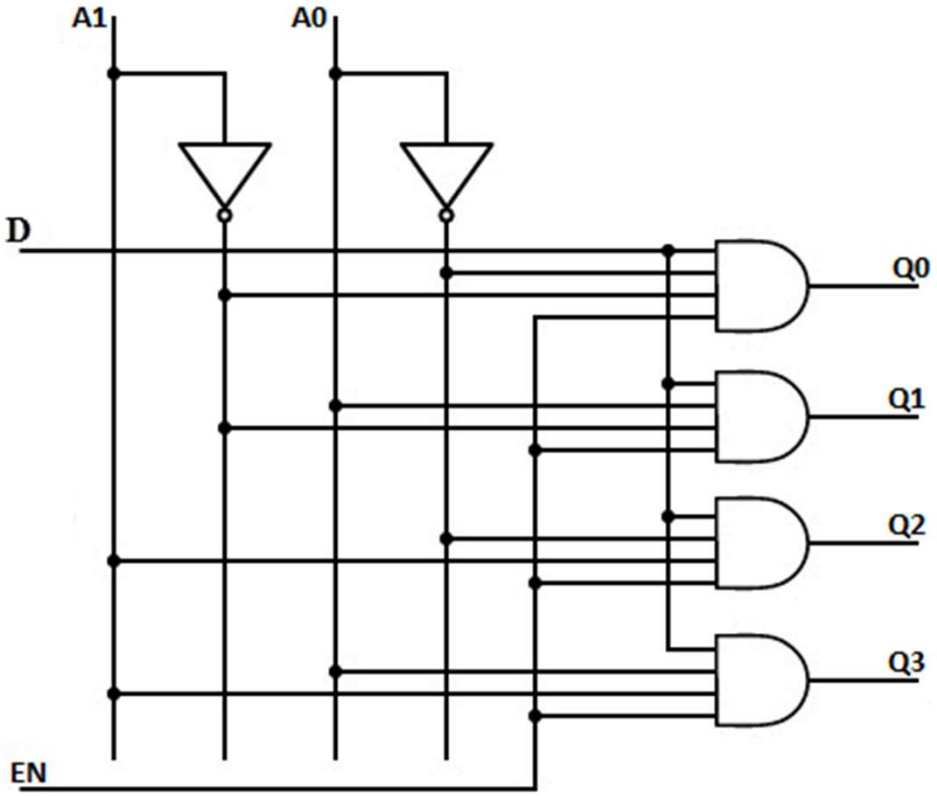
Démultiplexeur

- Les démultiplexeurs, ou distributeur de données, permettent de réaliser l'opération inverse des multiplexeurs : on envoie un signal d'entrée vers l'une des sorties possibles, en fonction d'une adresse codée en binaire
- L'utilisation essentielle des multiplexeurs et des démultiplexeurs est la gestion de l'écriture et de la lecture des registres qui stockent les données

- Exemple : Démultiplexeur à 4 sorties, ou démultiplexeur 1 vers 4



Démultiplexeur



EN	A1	A0	Q0	Q1	Q2	Q3
0	X	X	0	0	0	0
1	0	0	D	0	0	0
1	0	1	0	D	0	0
1	1	0	0	0	D	0
1	1	1	0	0	0	D

$$\begin{cases} Q_0 = EN.\overline{A_1}.\overline{A_0}.D \\ Q_1 = EN.\overline{A_1}.A_0.D \\ Q_2 = EN.A_1.\overline{A_0}.D \\ Q_3 = EN.A_1.A_0.D \end{cases}$$

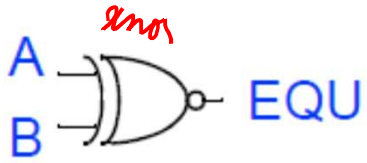
Sommaire

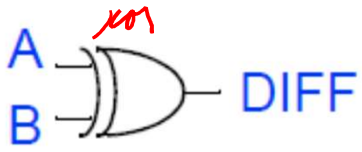
- Transcodage
- Multiplexeur et démultiplexeur
- Comparateur
- Additionneur et soustracteur
- Bascules

Traitement parallèle

Exemple : la comparaison de deux nombres

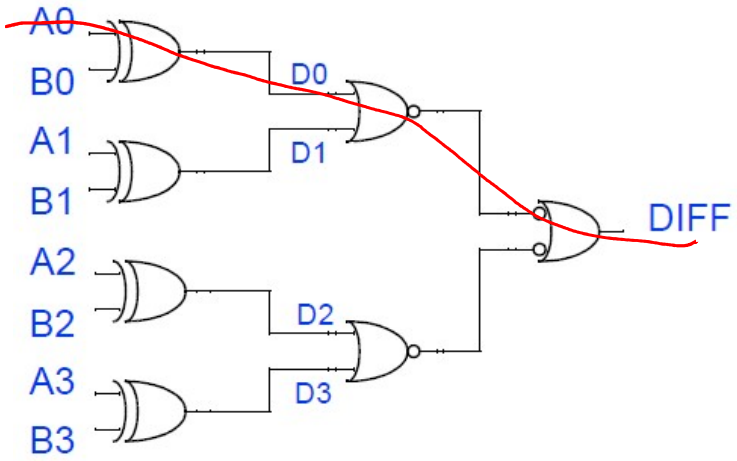
- Comparateur à 1 bit





EQU = 1 si A = B DIFF = 1 si A ≠ B

- Extension à N bits



Délai ?

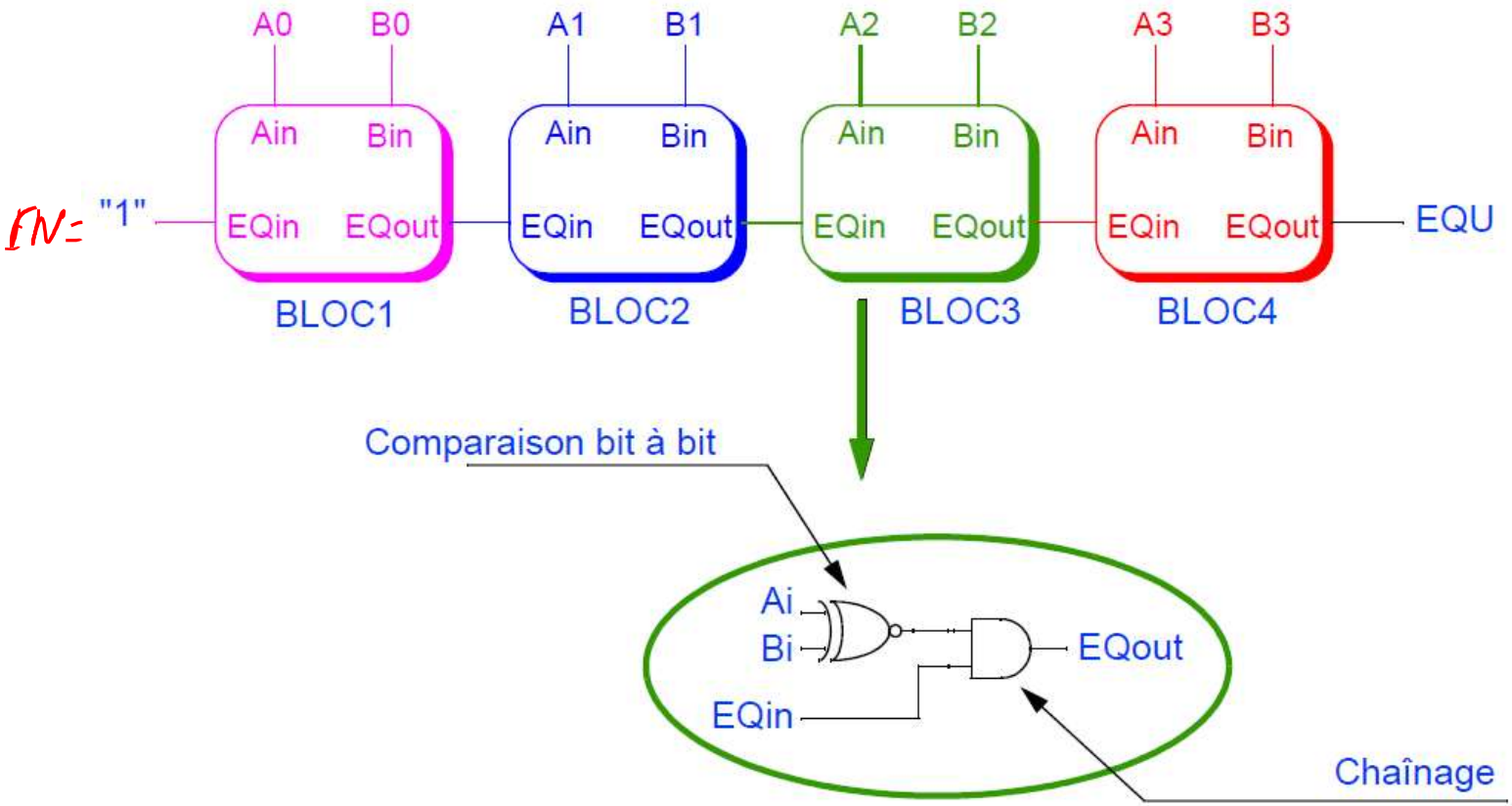
1 retard de valeur τ par porte :

délai sur DIFF : 3τ

Traitement itératif

Exemple : la comparaison de deux nombres

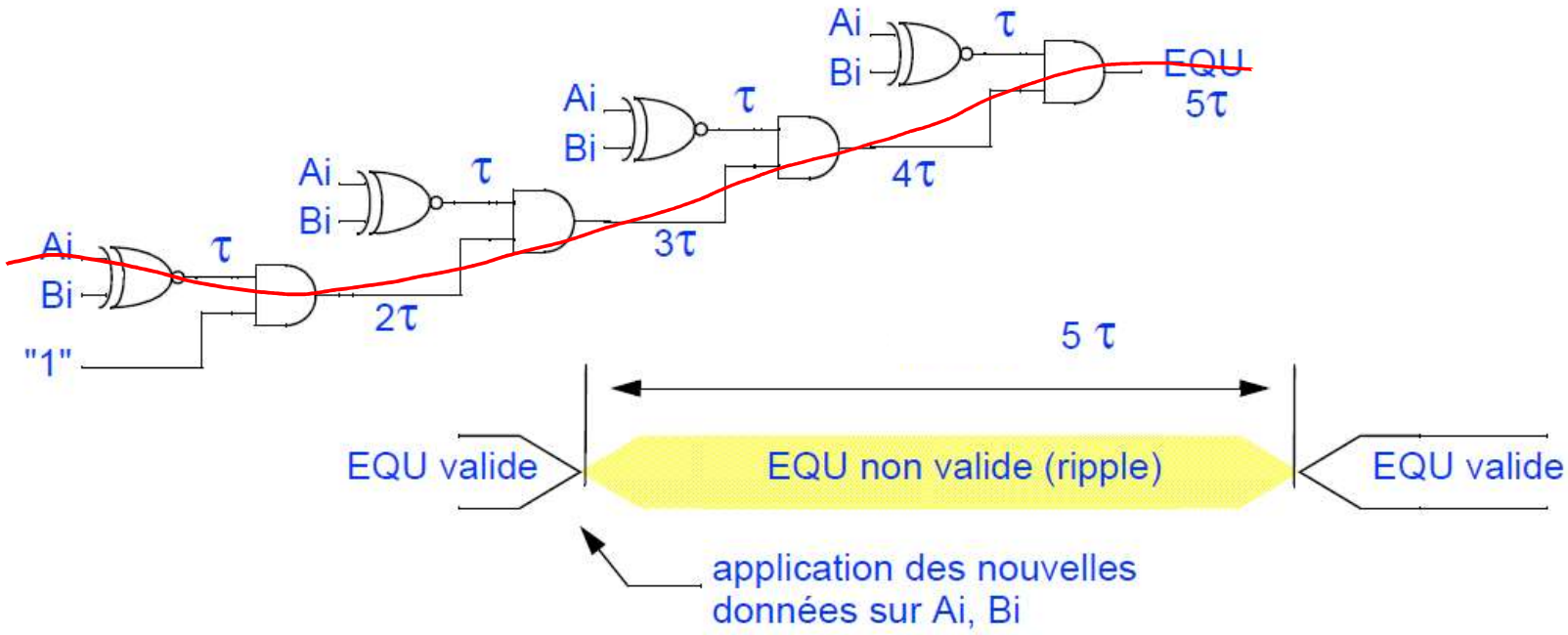
- Structure générale



Traitement itératif

Exemple : la comparaison de deux nombres

- Délai : Un retard de valeur τ par porte

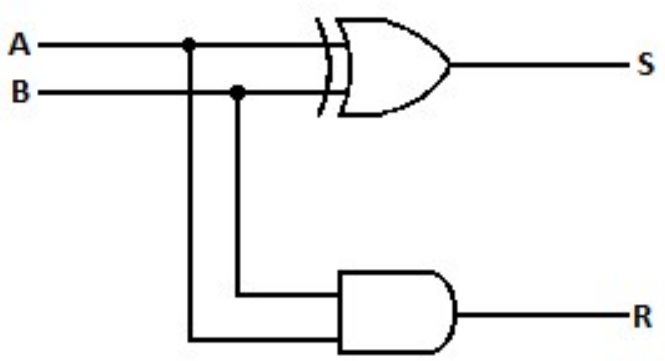


Sommaire

- Transcodage
- Multiplexeur et démultiplexeur
- Comparateur
- Additionneur et soustracteur
- Bascules

Additionneur

- Un additionneur est un circuit capable de faire l'addition de deux nombres de n bits. Une addition génère deux résultats : la somme et la retenue
- Exemple : Demi-additionneur 1 bit (half adder)



A	B	S	R
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

$$Somme\ S = A \oplus B$$
$$Retenue\ R = A.B$$

Limité à des mots de 1 bit car ne permet pas le chaînage

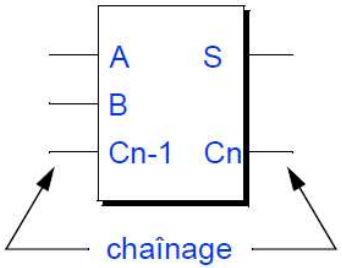
Additionneur

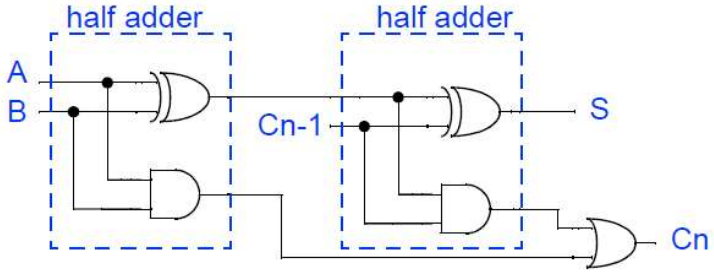
- Additionneur complet 1 bit (full adder)

$$S = \bar{A}.\bar{B}.C_{n-1} + A.\bar{B}.\overline{C_{n-1}} + \bar{A}.B.\overline{C_{n-1}} + A.B.C_{n-1}$$

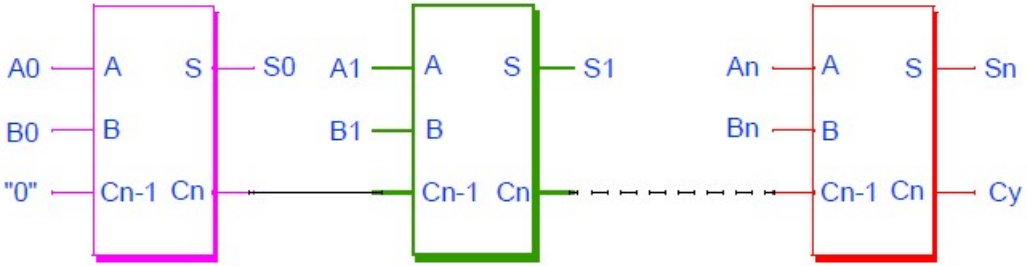
$$C_n = A.B + A.C_{n-1} + B.C_{n-1} = A.B + C_{n-1}.(A + B)$$

$$équivalent\ à\ C_n = A.B + C_{n-1}.(A \oplus B)$$



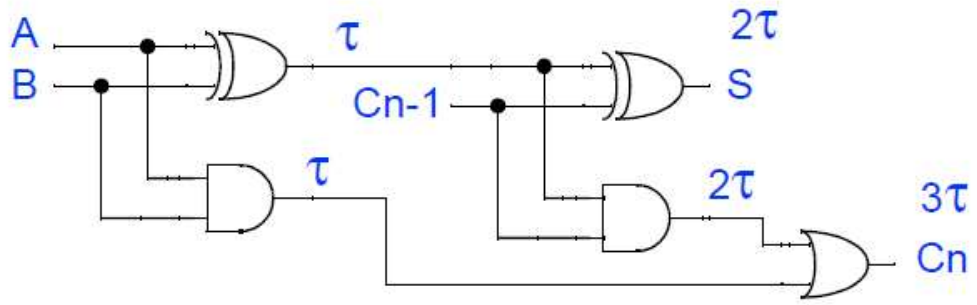


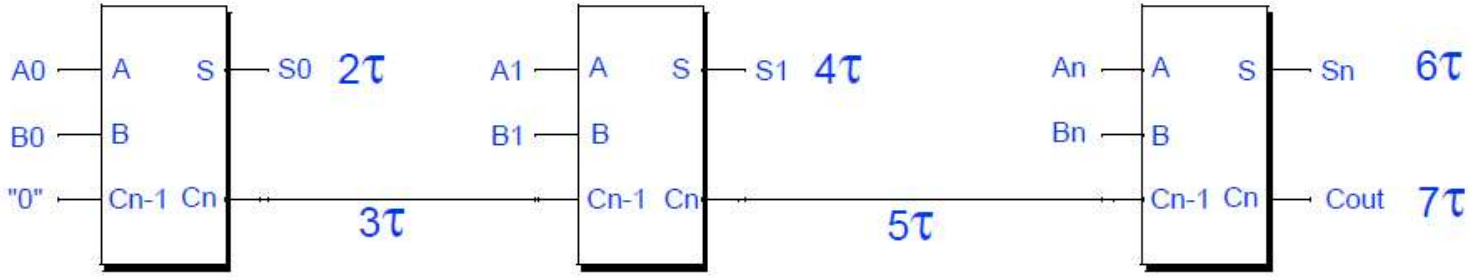
- Additionneur n bits itératif (ripple adder)



Additionneur

- Délai





Pour n bits : le délai sur Cout vaut (2n + 1) délai de porte

le délai sur Sn vaut 2n délai de porte

Extension à n bits : traitement parallèle

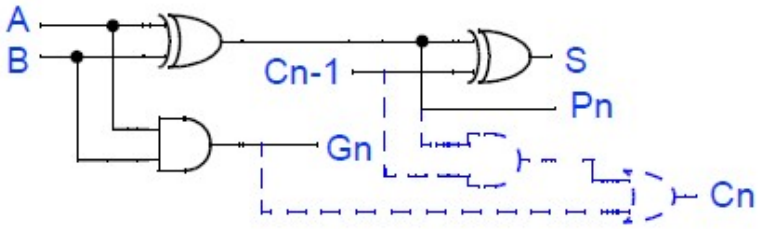
- Anticipation de la retenue (carry look-ahead)

$$C_n = A.B + C_{n-1}.(A \oplus B) = G_n + C_{n-1}.P_n$$

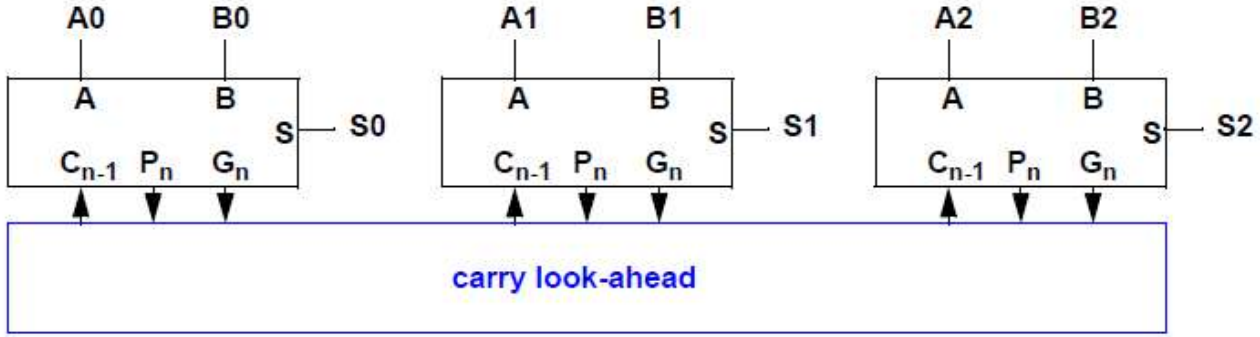
Fonction de génération — G_n

Fonction de propagation — P_n

  - Additionneur partiel :

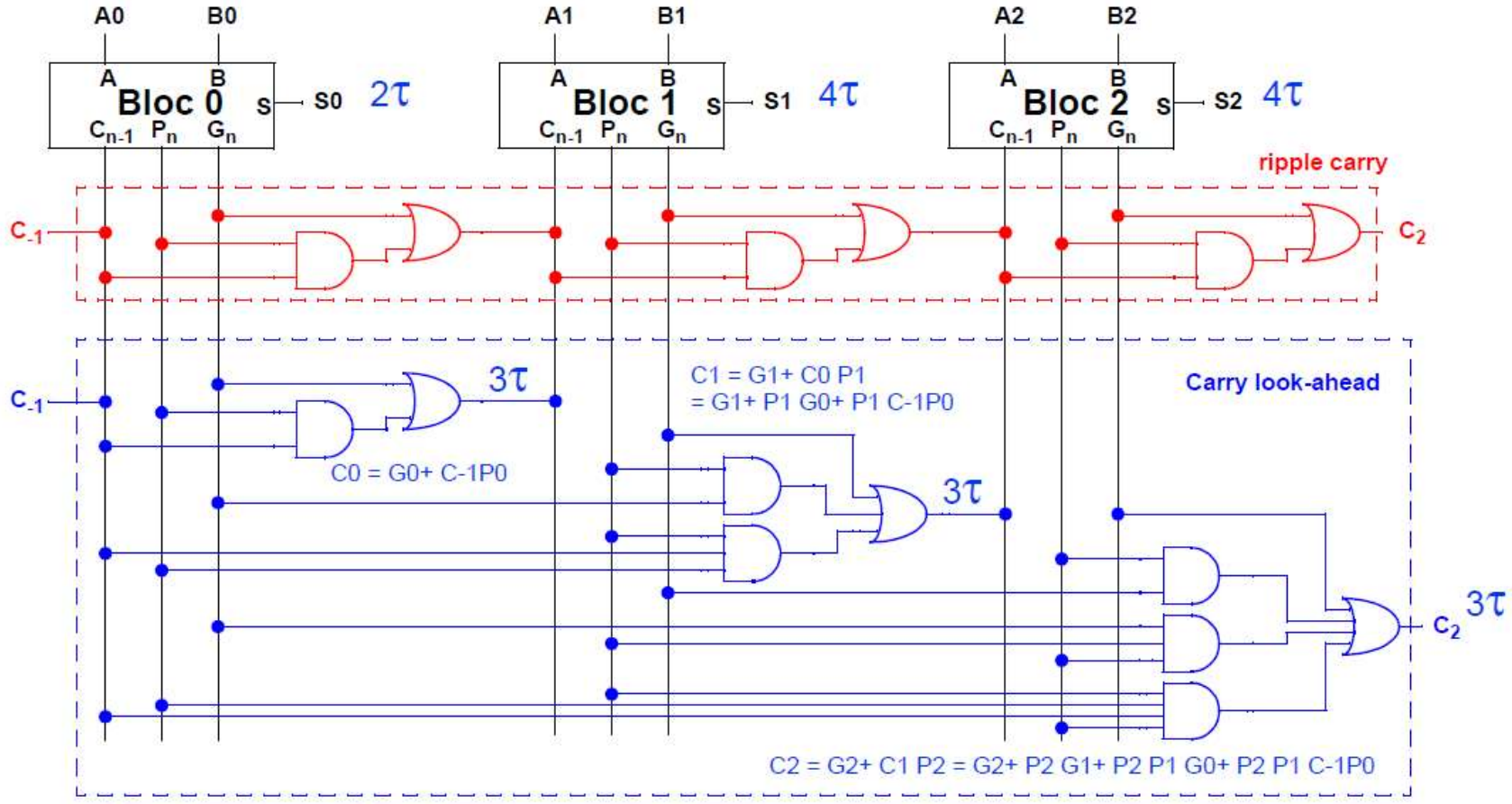


  - Additionneur = additionneur partiel + bloc de calcul de retenue



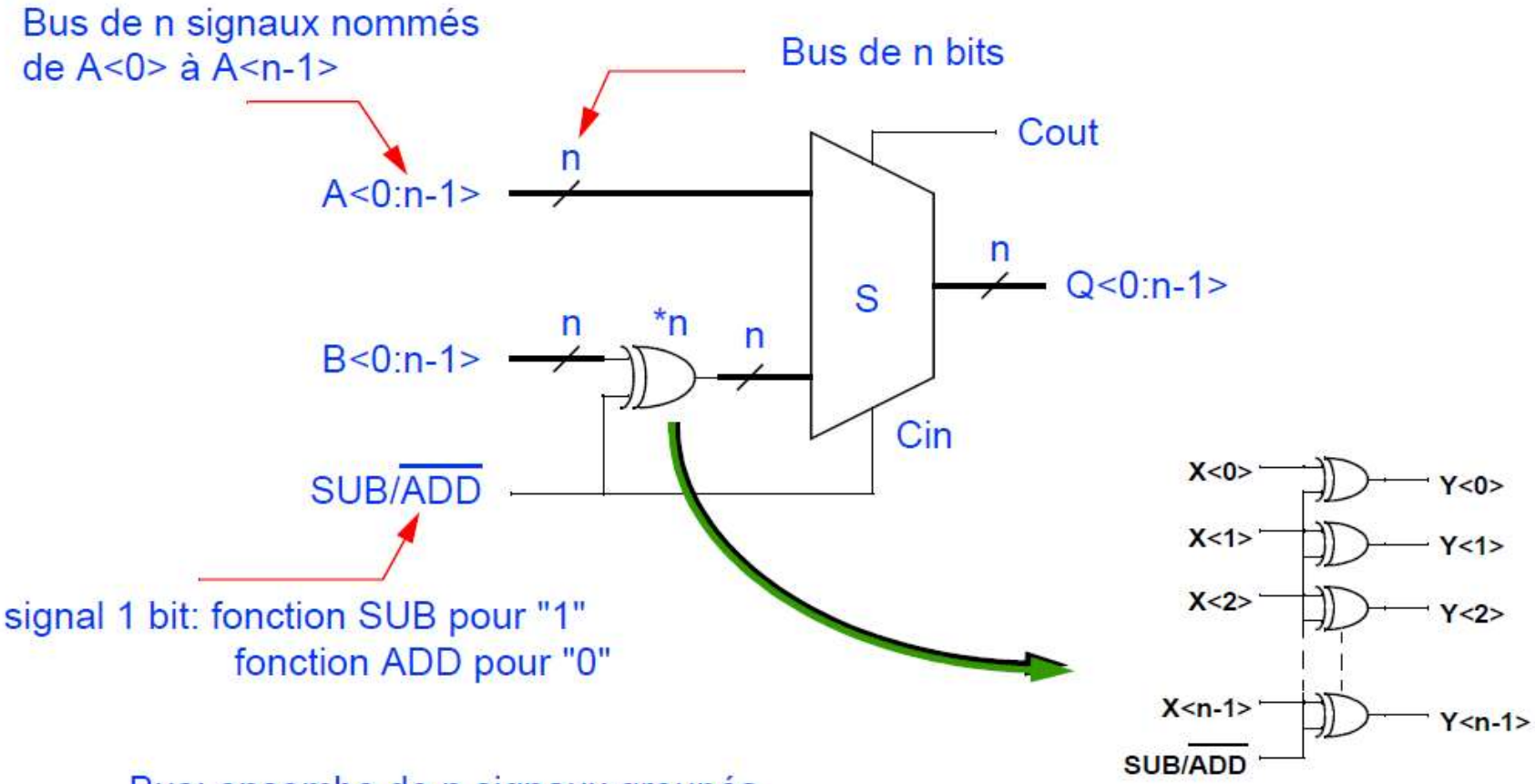
Extension à n bits : traitement parallèle

- Délai sur Gn et Pn : τ



Soustracteur

- Principe : $A - B = A + (-B) \Longrightarrow A + \bar{B} + 1$



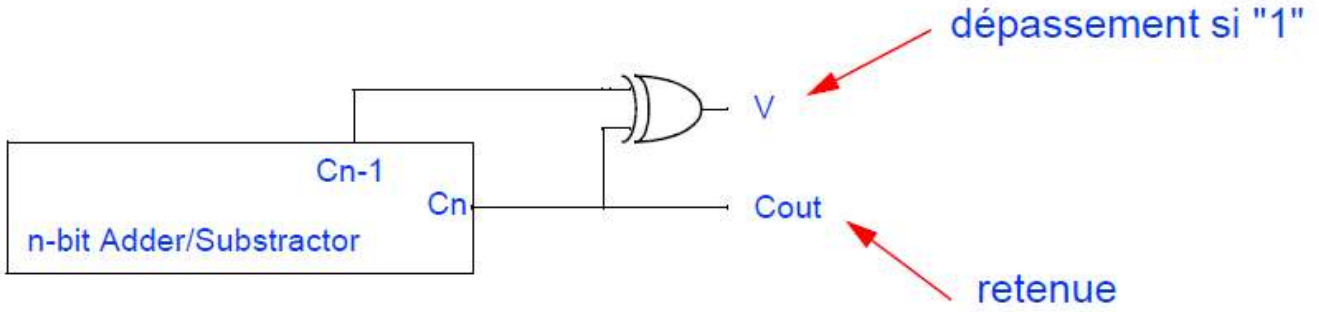
Dépassement de dynamique

- Exemple sur 8 bits en complément à 2 : dynamique comprise entre -128 et +127

Retenues 0 1		Retenues 0 1	
+70	0 1000110	-70	1 0111010
+80	0 1010000	-80	1 0110000
+150	1 0010110	-150	0 1101010
	⚠ **-106**		⚠ **+106**

Le résultat devrait être sur 9 bit (la dernière retenue est le 9ème bit)

Il y a dépassement (overflow) lorsque les deux dernières retenues sont différentes

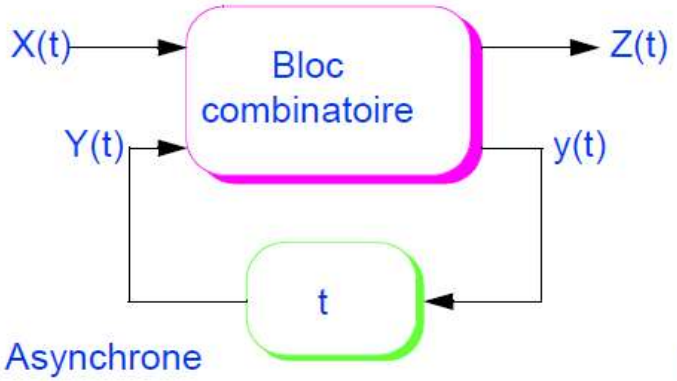


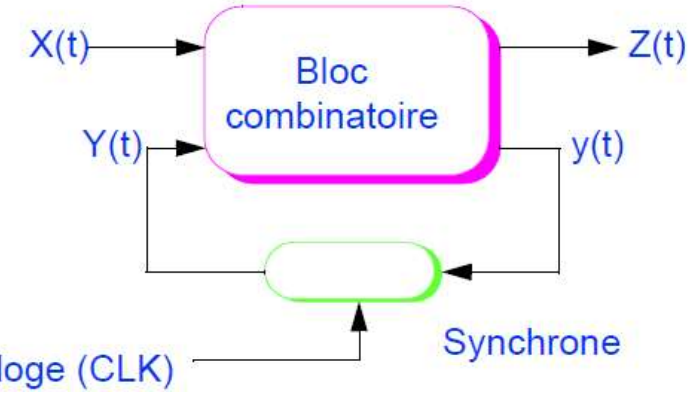
Sommaire

- Transcodage
- Multiplexeur et démultiplexeur
- Comparateur
- Additionneur et soustracteur
- Bascules

Définitions

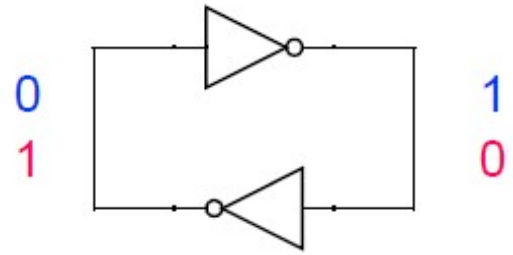
- Logique séquentielle : La combinaison des sorties ne dépend pas seulement de celle des entrées mais aussi de l'état précédent des sorties
- Types de circuits :
  - Astable: le circuit ne possède pas d'état stable (oscillateur)
  - Monostable: le circuit possède un état stable et un état fugitif de durée déterminée déclenché par un évènement particulier (trigger)
  - Bistable: le circuit possède deux états stables (mémoire)
- Circuits synchrones et asynchrones :





Bascule élémentaire : la bascule RS

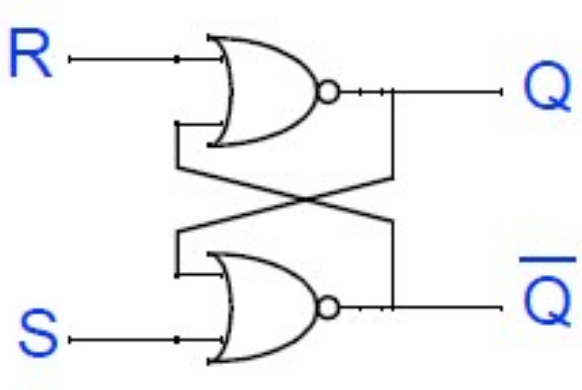
- Circuit minimal :



Bascule : circuit bistable

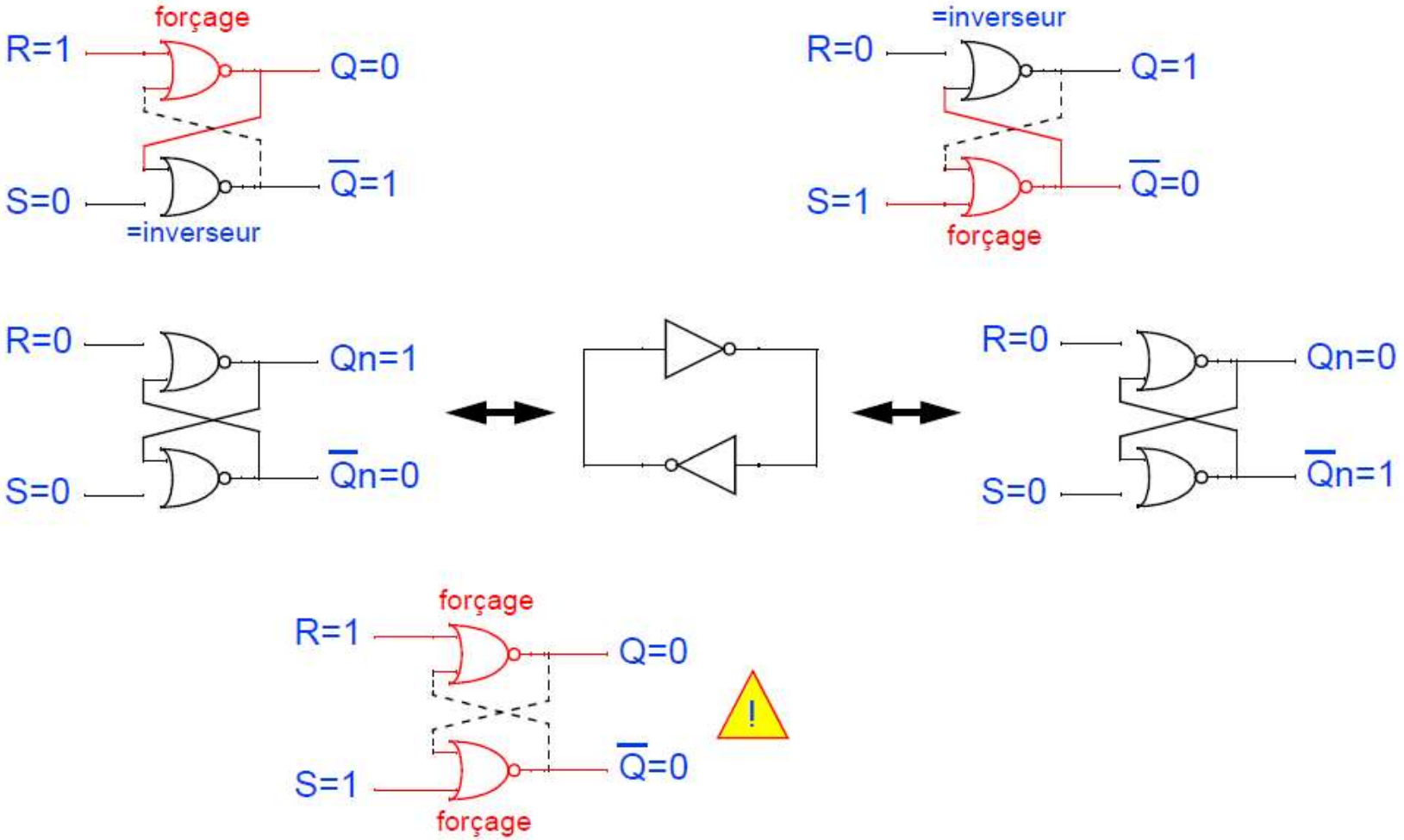
⚠ Il faut un moyen de fixer l'état désiré

- Réalisation pratique :

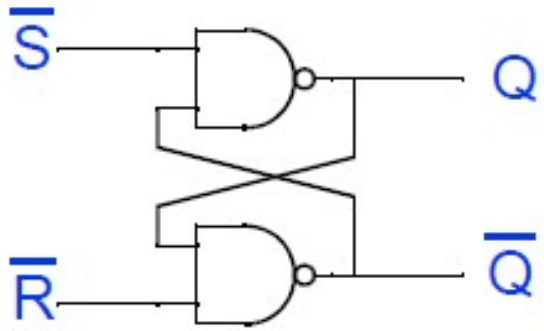


R	S	Q	$\overline{Q}$	Remarques
0	0	Q	$\overline{Q}$	Etat mémoire
0	1	1	0	Mise à 1 (S = SET)
1	0	0	1	Mise à 0 (R = RESET)
1	1	-	-	Etat interdit

Bascule élémentaire : la bascule RS

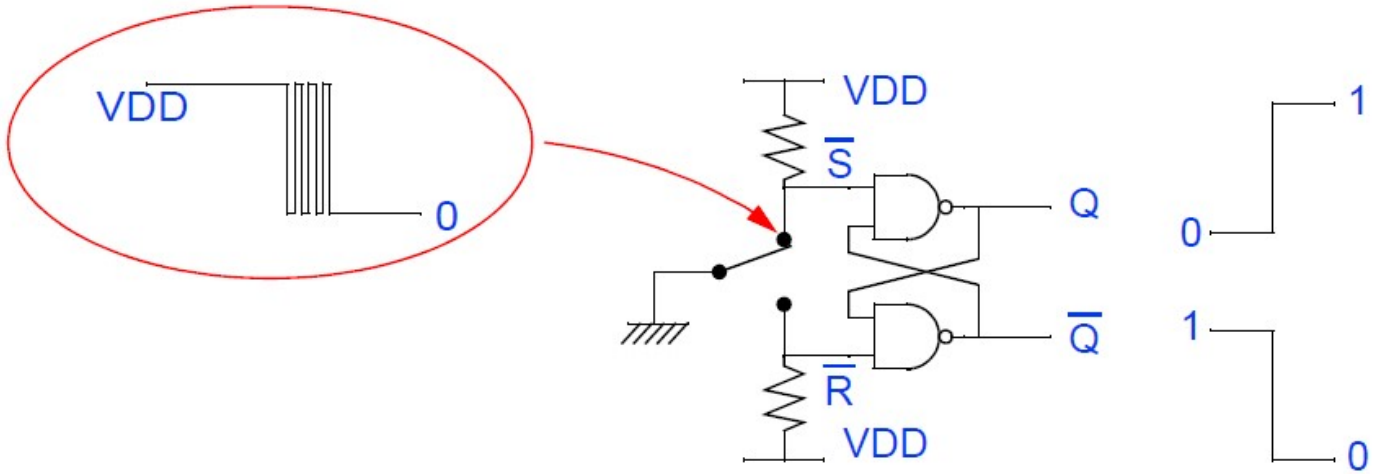


La bascule $\overline{R}\overline{S}$

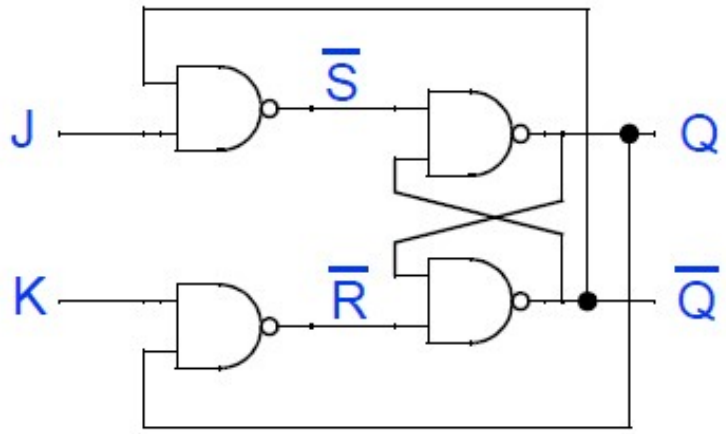


$\overline{R}$	$\overline{S}$	Q	$\overline{Q}$	Remarques
0	0	-	-	Etat interdit
0	1	0	1	Mise à 0
1	0	1	0	Mise à 1
1	1	Q	$\bar{Q}$	Etat mémoire

- Application : circuit anti-rebond



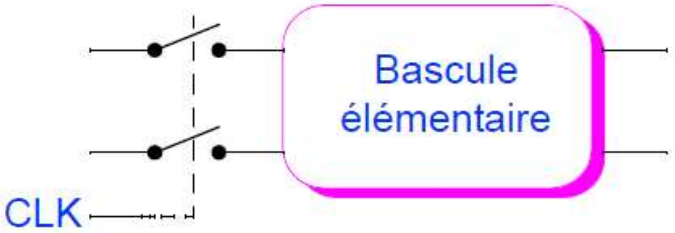
La bascule JK (peu utilisée actuellement)



J	K	Q	$\bar{Q}$	Remarques
0	0	Q	$\bar{Q}$	Etat mémoire
0	1	0	1	Mise à 0
1	0	1	0	Mise à 1
1	1	$\bar{Q}$	Q	Changement d'état

- Limitations des bascules asynchrones :
  - Fonctionnement asynchrone: la sortie réagit "immédiatement" à l'entrée
  - Les entrées doivent donc rester stables

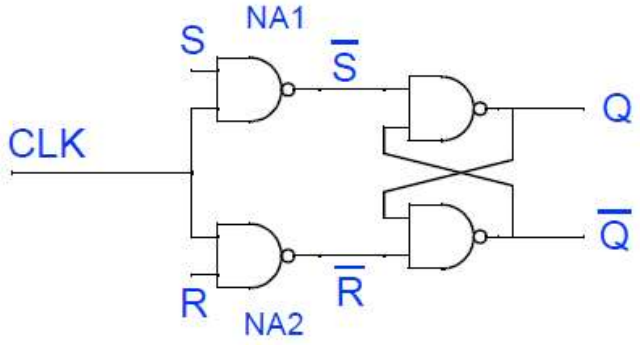
La bascule RS synchrone



CLK inactif: la bascule est isolée (il faut s'assurer qu'elle se trouve en configuration mémoire)

CLK actif: la bascule fonctionne normalement !

« Actif » ne signifie pas nécessairement au niveau haut (1)



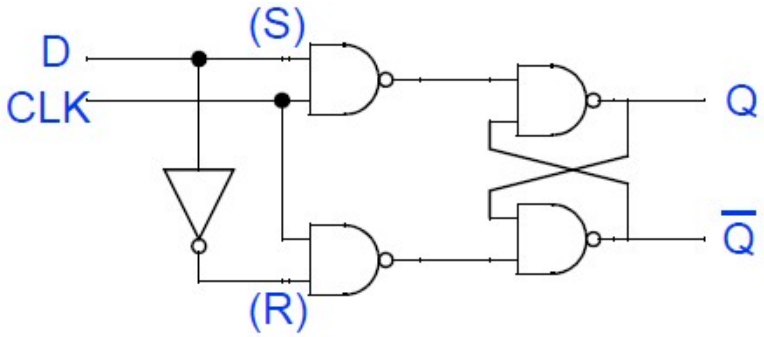
CLK = 0, R et S indifférents car NA1 et NA2 imposent un état $\bar{R} = \bar{S} = 1$ et par conséquent : Qn = Qn-1 (état mémoire)

CLK = 1: fonctionnement classique de la bascule car NA1 et NA2 se comportent en inverseurs

La bascule est transparente: la sortie réagit "immédiatement" à l'entrée lorsque CLK=1

ISEN école d'ingénieurs

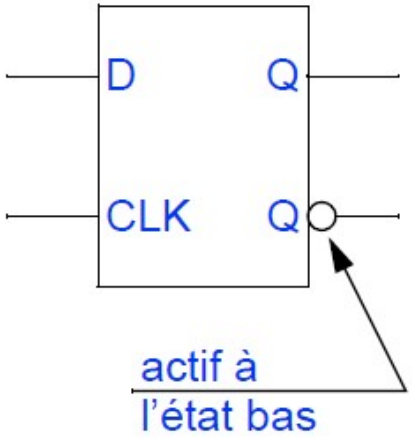
La bascule D



D (Data) = donnée

CLK	D	Q
0	X	Q
1	0	0
1	1	1

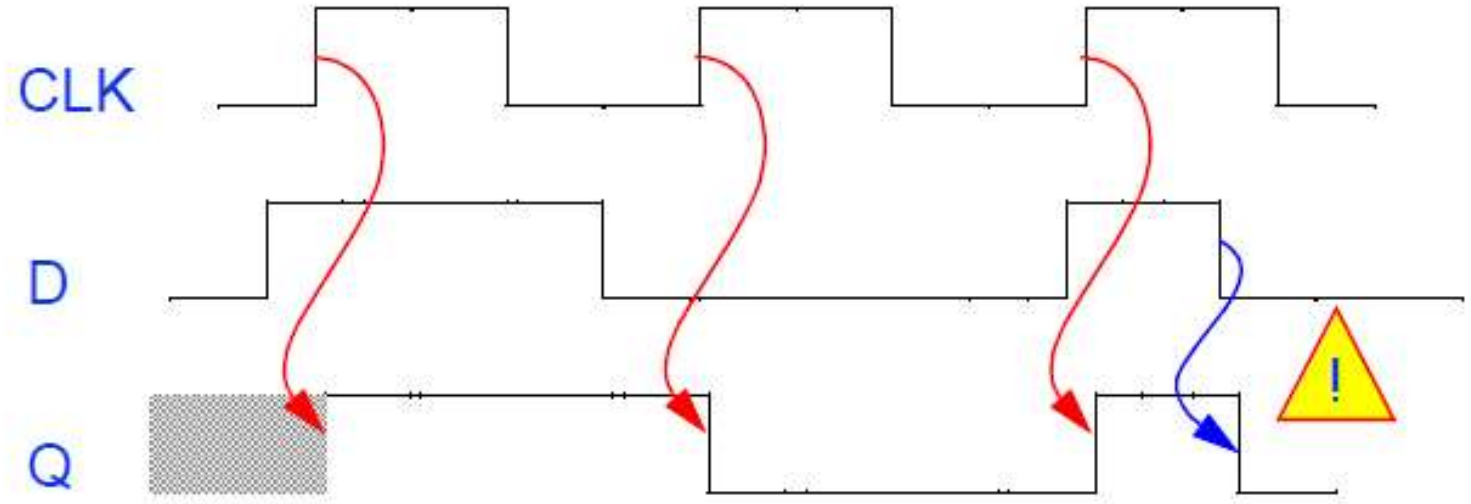
CLK = 1 : La bascule est transparente



Bascule D : bloc de base incontournable

ISEN école d'ingénieurs

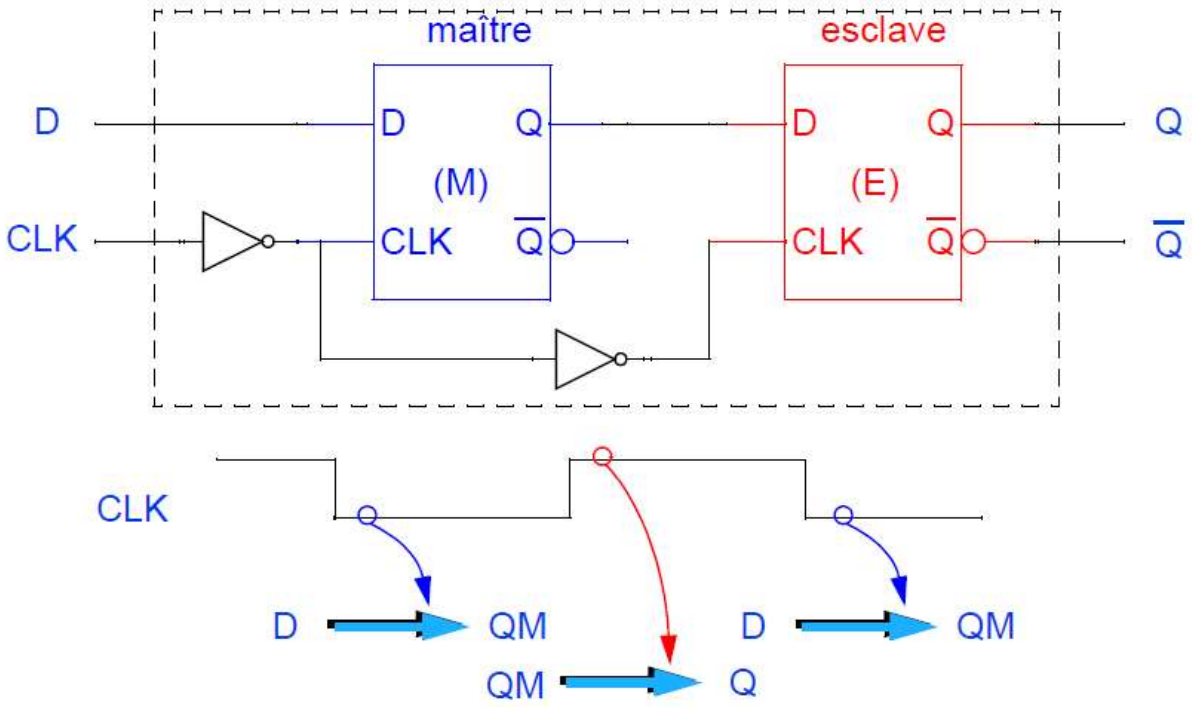
La bascule D



- Limitations pratiques :
  - La transition en sortie **semble** déclenchée par un front d'horloge
  - La sortie **peut évoluer** pendant le temps où CLK = 1
    - ⚠ Les entrées doivent **rester stables** lorsque CLK = 1

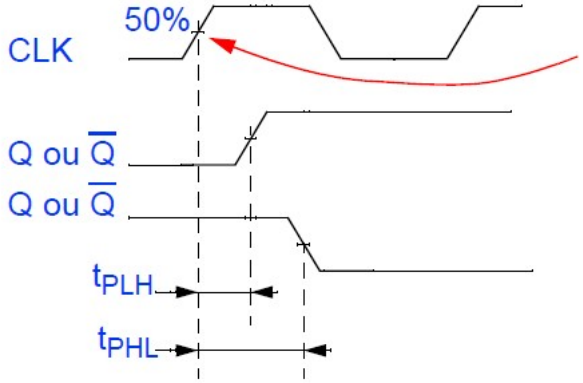
La bascule D

- Structure Maître-Esclave (Master-Slave) :
  - Principe: maître et esclave travaillent en alternance
  - Elimine la nécessité d'avoir des entrées stables lorsque CLK est actif
  - Dans cet exemple, la donnée apparaît en Q (et $\bar{Q}$) lors du front montant de CLK



Propagation et signal d'horloge

- Temps de propagation :

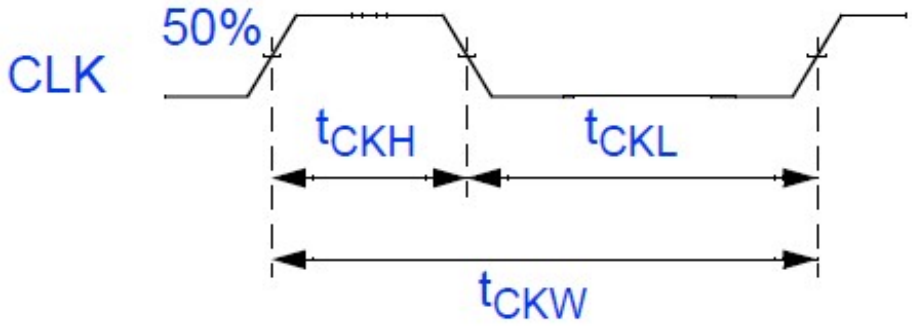


Transition active de l'horloge

Temps écoulé entre la transition active sur l'horloge (CLK) et un changement d'état sur une sortie (Q ou $\bar{Q}$)

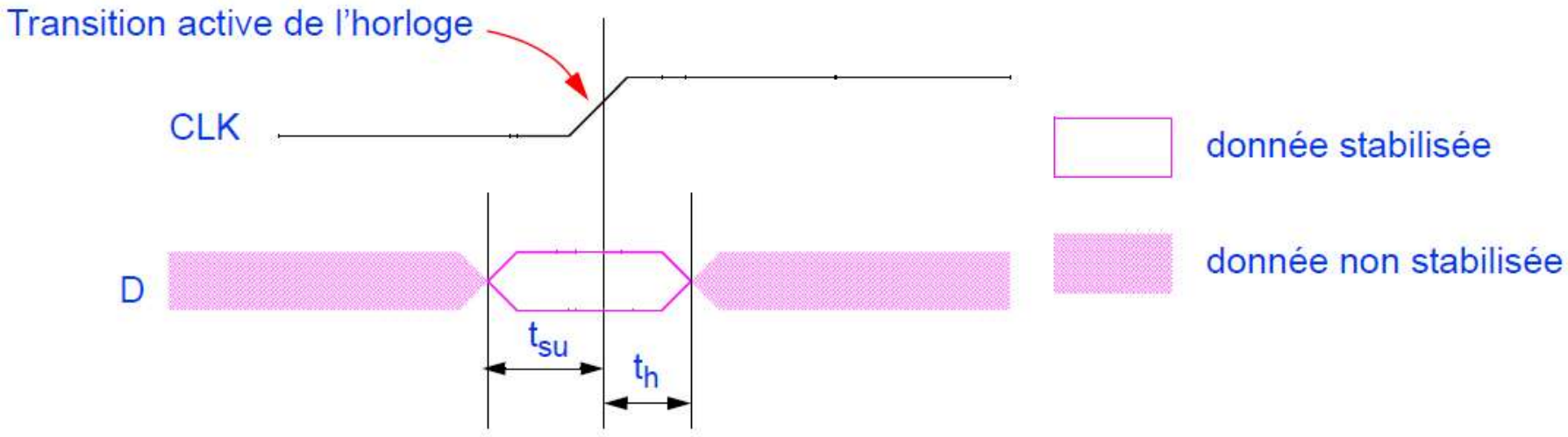
- Durée d'impulsion (pulse duration) : t_{CKL} et t_{CKH} doivent permettre aux bascules maître et esclave de se positionner correctement

$$t_{CKW} = \frac{1}{f_{max}}$$



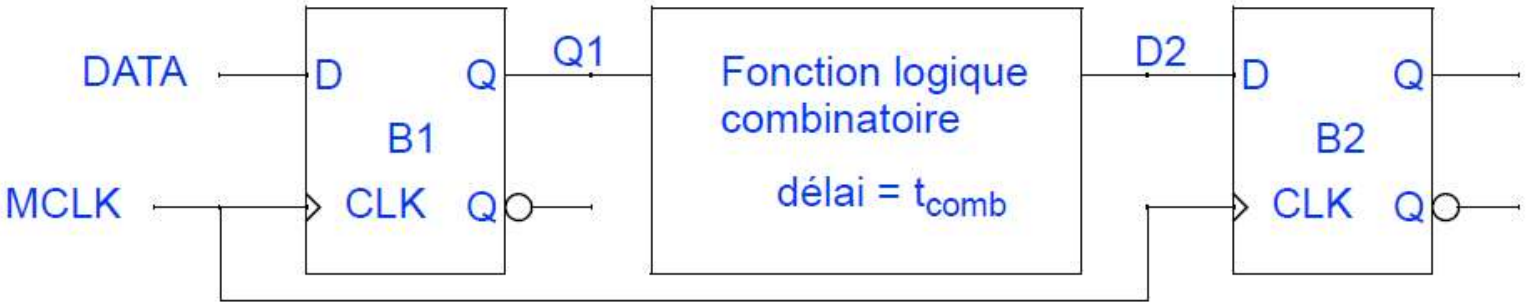
Setup et Hold

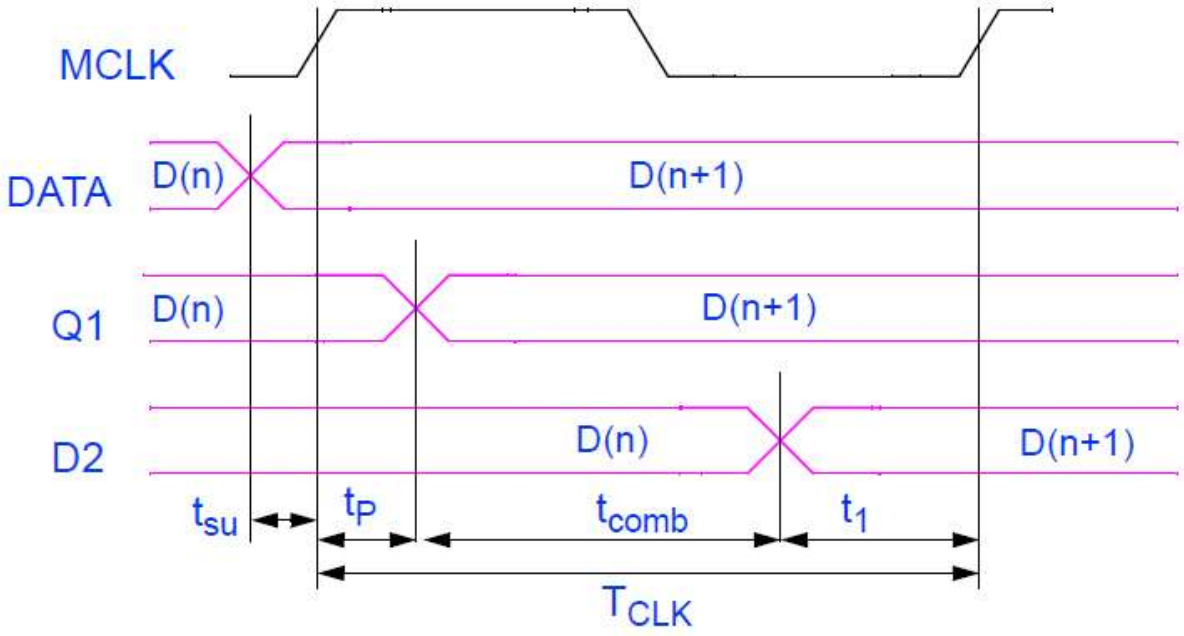
- Temps de prépositionnement (setup time, t_{su}) et de maintien (hold time, t_h)



  - t_{su} et t_h permettent à la bascule maître d'effectuer sa transition vers un état mémoire avec des données d'entrée stables
  - t_{su} et t_h ne sont pas nécessairement égaux, t_h peut être nul
  - La transition active de l'horloge **n'est pas obligatoirement** un front montant

Fréquence maximale de fonctionnement



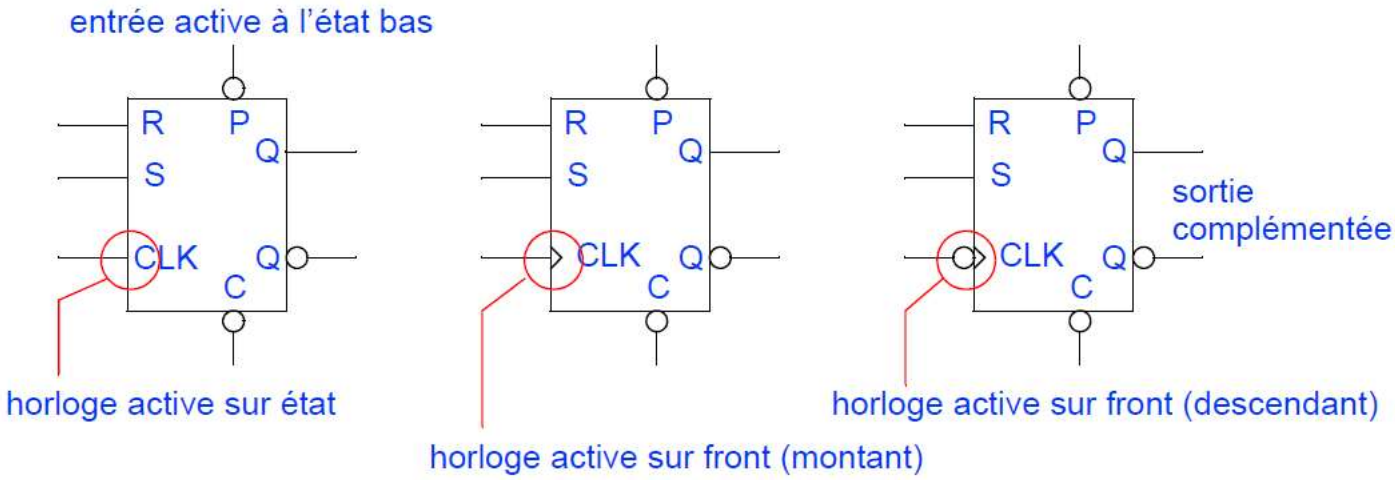


Au minimum: $t_1 = t_{su}$

$T_{CLK\ min} = t_P + t_{comb} + t_{su}$

$f_{max} = (T_{CLK\ min})^{-1}$

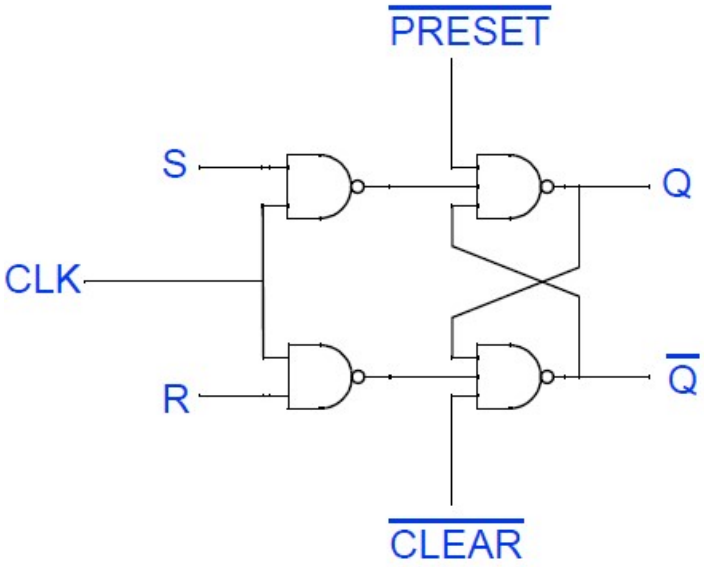
Symboles et définitions

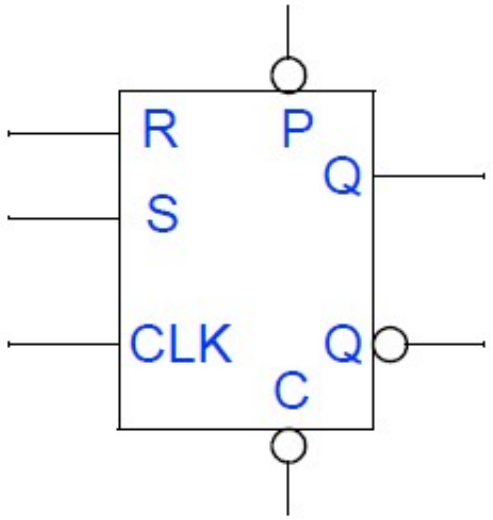


- Bascule avec horloge active sur état = latch (verrou)
- Bascule avec horloge active sur front = Flip-Flop
- Dans une bascule, il n'y a pas de relation temporelle directe entre les entrées (R, S, J, K ou D) et la sortie Q
- Les temps caractéristiques sont définis par rapport à l'horloge CLK:
  - t_{su}, t_h entre CLK et les entrées de données (R S, J, K ou D)
  - t_P entre CLK et les sorties Q et $\bar{Q}$

Entrées d'initialisation

- Exemple avec une bascule RS :
  - $\overline{PRESET} = 0, \overline{CLEAR} = 1 \Rightarrow Q = 1$
  - $\overline{PRESET} = 1, \overline{CLEAR} = 0 \Rightarrow Q = 0$
  - $\overline{PRESET} = 1, \overline{CLEAR} = 1 \Rightarrow Bascule\ normale$
  - Les entrées d'initialisation sont ici **asynchrones**





Récapitulatif (A savoir)

- Codeurs et décodeurs
- Multiplexage et démultiplexage
- Composants arithmétiques (comparateurs, additionneurs, soustracteurs)
- Bascules

Fin du chapitre 3