

Chapitre 1

De l'analogique au numérique ...

Systeme numérique ??

- Elements électroniques permettant de traiter des informations representees sous forme numerique :
 - calculatrices, montres, ordinateurs ...
- Donnée numerique : formée de grandeurs binaires (0 et 1)
- Matérialisée par un conducteur (fil) dont le niveau de tension represente la valeur binaire :
 - si 1 => Vdd (logique positive)
 - si 0 => Vdd (logique négative)

Analogique ?? Numérique ??

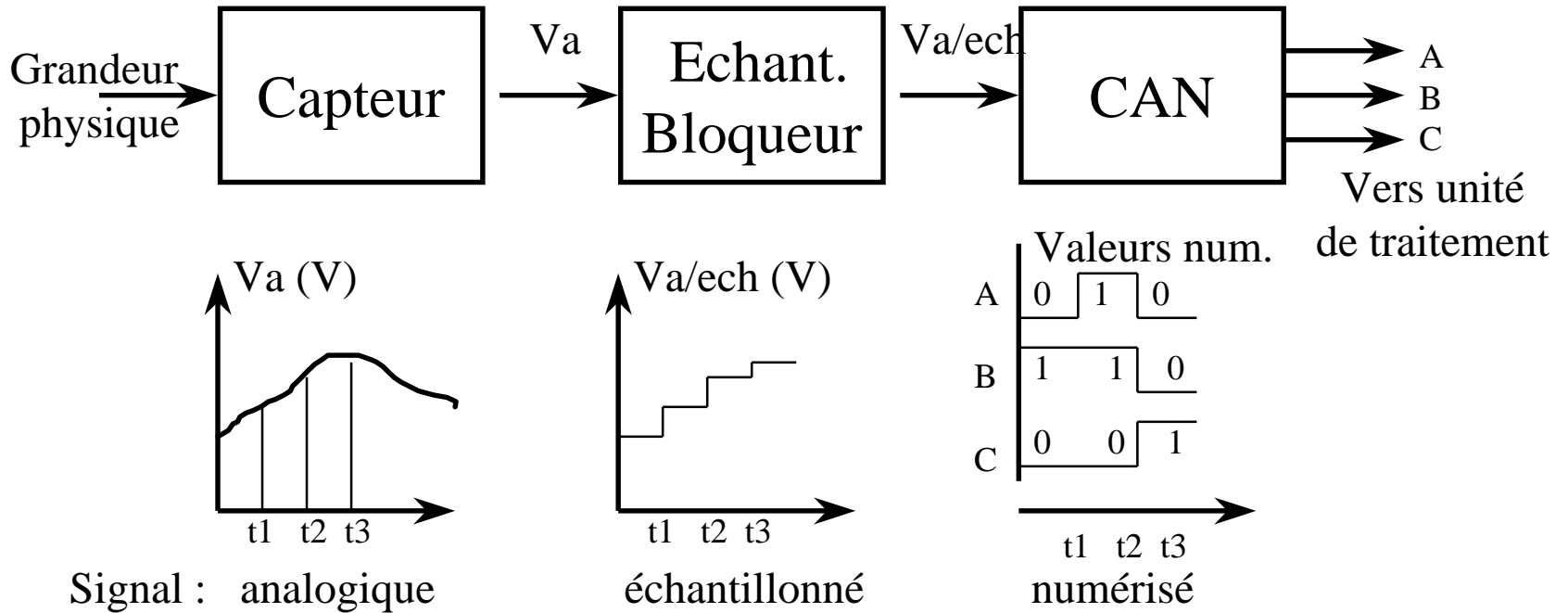
- Donnée numérique :
 - valeur discrète de la grandeur physique considérée
 - représente sa valeur à un instant donné
- Donnée analogique :
 - valeur continue de la grandeur physique considérée
 - varie continûment et proportionnellement avec celle-ci

Avantages ...

- Simplicité de conception
- bonne robustesse au bruit
- précision élevée et prévisible
- stockage de l'information aisée
- flexibilité d'un système numérique (programmation)
- possibilité d'effectuer des calculs complexes sur une donnée
- niveau d'intégration élevée sur silicium

Le problème : Le monde réel est analogique !!

Génération d'un signal numérique



Chaîne d'acquisition d'un signal analogique

Conversion de données : 3 étapes

- Acquisition du signal analogique
 - Echantillonnage du signal
 - Numérisation du signal

L'échantillonnage

- fréquence d'échantillonnage $> 2 * \text{fréquence max du signal}$

Théorème de Shannon

- valeur de l'échantillon conservé jusqu'à la suivante

échantillonneur-bloqueur

- conversion numérique entre 2 prises d'échantillon

limite technologique !!

- attention au rapport signal/bruit

choix du CAN crucial

Codage d'une donnée numérique

- bit, nombre binaire :
 - l'élément binaire : le **bit** (binary digit) ou chiffre binaire
 - 2 valeurs possibles : 0 ou 1
- élément de base du nombre (ou mot) binaire :
 - base 2
 - différentes représentations binaires d'un nombre décimal : codes

Nombre binaire

- A chaque chiffre du nombre binaire correspond un poids exprimé comme une puissance de 2 :

$$2^2 \ 2^1 \ 2^0, \ 2^{-1} \ 2^{-2}$$

$$1 \ 0 \ 1, \ 1 \ 0$$

MSB

LSB

Conversion décimal/binaire

- décomposition du nombre décimal en somme de puissance de 2

$$\begin{aligned}13_{10} &\Rightarrow 8+4+1 \\ &\Rightarrow 1x2^3 + 1x2^2 + 0x2^1 + 1x2^0 \\ &\Rightarrow 1101_2\end{aligned}$$

- divisions successives par 2, les restes successifs étant les chiffres binaires en partant du poids faible...

Code binaire naturel (1)

- sur 3 bits :

binaire	naturel
$2^2 2^1 2^0$	10^2
0 0 0	0
0 0 1	1
0 1 0	2
0 1 1	3
1 0 0	4
1 0 1	5
1 1 0	6
1 1 1	7

- format des nbres binaires = poids en puissance de 2 (2, 4...)
 - mots de 4, 8, 16, 32 bits, etc ...
 - calculs et électroniques simplifiés

Code binaire naturel (2)

- Exercice : Donner l'équivalent en code binaire naturel des nombres de 0 à 15 ...

Code octal

- base 8 (2^3)
- 8 symboles possibles : 0 1 2 3 4 5 6 7
- ex : $372_8 \Rightarrow 3 \times 8^2 + 7 \times 8^1 + 2 \times 8^0 = 250_{10}$
- conversion décimal-octal : division par 8
- conversion octal-binaire : immédiate
 - conversion de chaque chiffre en base 8 en binaire sur 3 bits
- conversion binaire-octal :
 - on regroupe par 3 bits et on remplace par le chiffre décimal correspondant
- utilité : façon abrégée de représenter de grands chiffres binaires

Code hexadécimal

- base 16
- 16 symboles : 0 1 2 3 4 5 6 7 8 9 A B C D E F
- même principe de conversion que le code octal
- ex : $9F2_{16} \Rightarrow 1001\ 1111\ 0101_2$
- utilisé lorsqu'on programme en assembleur (langage machine) par exemple.

Code DCB (Décimal Codé Binaire)

- BCD en anglais...
- On code chaque chiffre par son équivalent en binaire
ex : $390_{10} \Rightarrow 0011\ 1001\ 0000_{\text{BCD}}$
- Codage sur 4 bits obligatoire (0 à 9)
- plus facile et moins coûteux à coder que le binaire pur
- plus coûteux en nombre de bits requis
- Utilisé notamment pour l'affichage de donnée

Code Gray (1)

- code dit "à distance minimale"
- Un nombre de diffère du précédent ou du suivant que par un seul bit
- Code non pondéré : pas de poids affecté sur la position des chiffres dans le mot
- ne convient pas aux opérations arithmétiques
- utilisé dans les convertisseurs par exemple

Code Gray (2)

Sur 3 bits :

0	000
1	<u>001</u>
2	011
3	<u>010</u>
4	110
5	111
6	101
7	100

- Méthode du miroir
- Code robuste :
Permet d'éviter des erreurs dû à de nombreux changements d'un nombre binaire à un autre
- ex : 0111 à 1000 en binaire pur
- Ce passage peut entraîner plusieurs états intermédiaires
=> risque de mauvais fonctionnements

Code Gray (3)

- Exercice : Donner l'équivalent en code Gray des nombres de 0 à 15 ...

Code alphanumérique (1)

- Le plus connu : le code ASCII
(American Standard for Interchange of information)
- Transcrit en 1 mot binaire de 7 bits, soit 128 codes possibles dont
 - Les 26 lettres de l'alphabet
 - les 10 chiffres
 - 7 signes de ponctuation
 - entre 20 et 40 caractères spéciaux (+ / - * # % ...)
- Utilisé dans les périphériques des ordinateurs (clavier, écran, imprimantes...)

Code alphanumérique (2)

- exemple : une ligne de programme tapée au clavier

GOTO 25

G 1000111

O 1001111

T 1010100

O 1001111

(espace) 0100000

2 0110010

5 0110101

Chapitre 2

Circuits logiques combinatoires

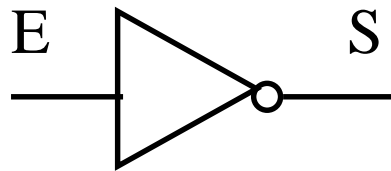
Circuit combinatoire ??

- A tout moment :
 - Le circuit réalise sur sa sortie une fonction de ses entrées
 - La valeur de sortie dépend **uniquement** des valeurs présentes en entrées
- Pas d'effet mémoire, aucune notion de valeur antérieure
- Si une entrée évolue, la sortie sera éventuellement affectée, mais seulement après un temps donné
 - => Contraintes temporelles fortes

Quelques circuits combinatoires : Les portes logiques élémentaires (1)

Porte inverseuse

$$S = \bar{E}$$



Symbole

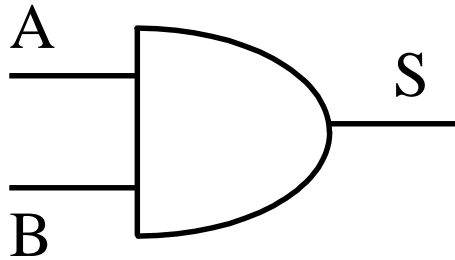
<u>E</u>	<u>S</u>
0	1
1	0

Table de vérité

Les portes logiques élémentaires (2)

Porte ET

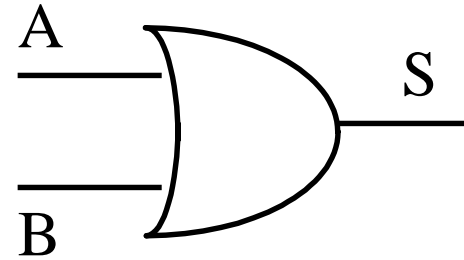
$$S = A.B$$



<u>A</u>	<u>B</u>	<u>S</u>
0	0	0
0	1	0
1	0	0
1	1	1

Porte OU

$$S = A+B$$

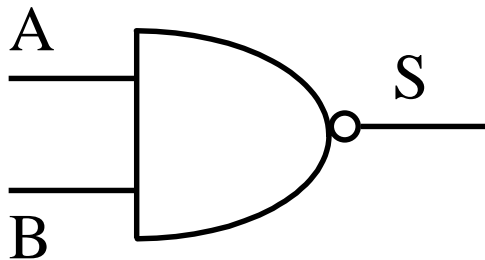


<u>A</u>	<u>B</u>	<u>S</u>
0	0	0
0	1	1
1	0	1
1	1	1

Les portes logiques élémentaires (3)

Porte NAND

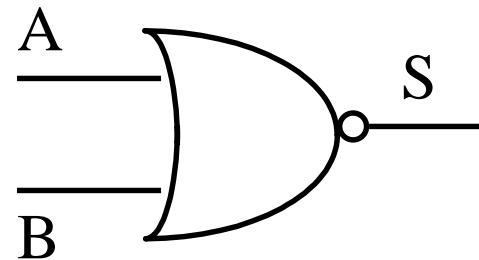
$$S = \overline{A.B}$$



<u>A</u>	<u>B</u>	<u>S</u>
0	0	1
0	1	1
1	0	1
1	1	0

Porte NOR

$$S = \overline{A+B}$$



<u>A</u>	<u>B</u>	<u>S</u>
0	0	1
0	1	0
1	0	0
1	1	0

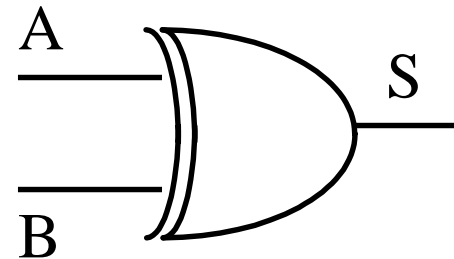
Les classiques ... (1)

OU exclusif ou XOR

- Très utilisé : permet de savoir si des données sont différentes

$$S = \bar{A}.B + A.\bar{B}$$
$$S = A \oplus B$$

<u>A</u>	<u>B</u>	<u>S</u>
0	0	0
0	1	1
1	0	1
1	1	0



Les classiques ... (2)

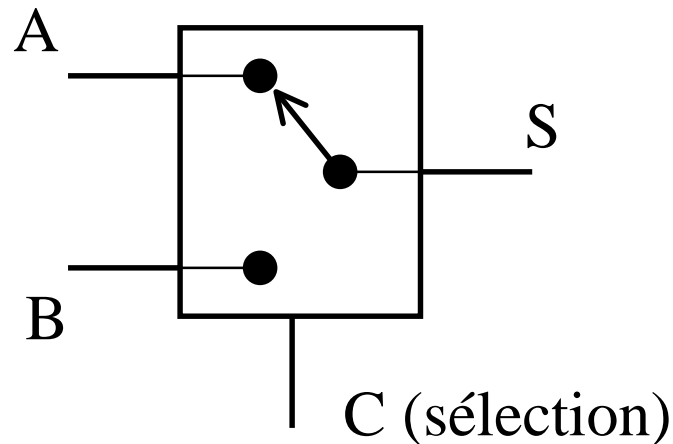
OU exclusif (2)

- Exercice : Donner le schéma en portes logiques (inv, ou, et) d'un OU exclusif à 2 entrées.

Les classiques ... (3)

Multiplexeurs (1)

- "Connecte" la sortie à une des entrées
- Cette sélection se fait par un système d'adressage
- ex : Mux à 2 entrées



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

Les classiques ... (4)

Multiplexeurs (2)

- Si l'adresse des entrées est sur n bits, il y aura 2^n entrées
- Il existe des Mux de 4, 8 (par exemple le 74-151), 16 entrées...
- Utilisé pour : Aiguillage de données, conversion série/parallèle ...

Les classiques ... (5)

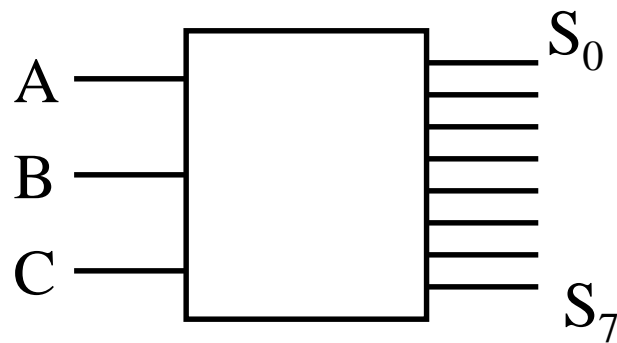
Multiplexeurs (3)

- Exercice 1 : Donner le schéma en portes logiques (inv, ou, et) d'un multiplexeur à 4 entrées.
- Exercice 2 : Comment utiliser le Mux ci-dessus pour avoir la fonctionnalité d'un OU exclusif à 2 entrées.

Les classiques ... (6)

Décodeurs (1)

- Établir une correspondance entre entrées et sorties
- ex : un décodeur "1 parmi 8" (74-138) : pour un mot de 3 bits en entrée donne 1 sortie parmi les 8



Les classiques ... (7)

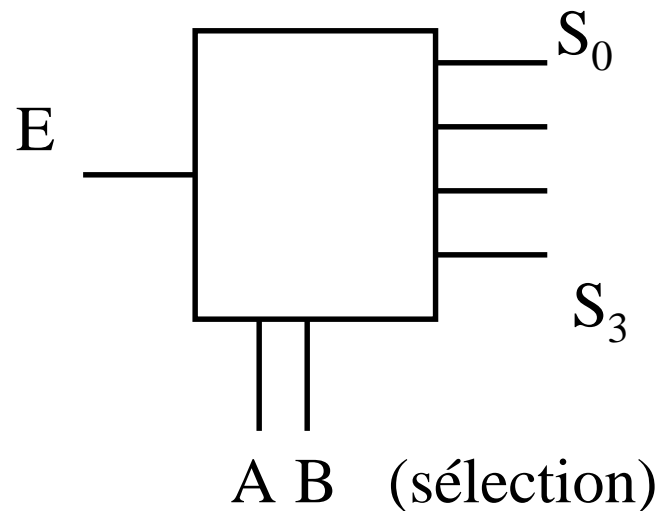
Décodeurs (2)

- un circuit qui effectue une transformation code binaire pur/code Gray, par exemple, est un décodeur
- Possèdent en général d'une entrée "validation" (chip enable) qui autorise le décodeur à fonctionner
- Utilisation : Conversion, système d'adressage, séquençement de jeu d'instruction

Les classiques ... (8)

Décodeurs (3)

- **Le Démultiplexeur** : décodeur qui pour un mot de n bits (adresse) fait correspondre l'entrée à une des sorties



Conception de circuits combinatoires

Algèbre de Boole (1)

- Proche de l'algèbre classique
- Algèbre de boole permet :
 - d'exprimer la fonctionnalité logique d'un circuit (équation booléenne)
 - de manipuler les variables logiques en vue d'optimiser l'implantation matérielle d'une fonction logique

Algèbre de Boole (2)

- Variables booléennes :
 - 2 valeurs possibles : 0 et 1
 - 3 opérations élémentaires :
 - addition logique (OU)
 - multiplication logique (ET)
 - complémentation logique (NON ou inverseur)
- Equations booléennes
 - présentées sous forme de :
 - somme de produit : $(A.B)+(C.D)$ (la plus courante)
 - produit de somme

Algèbre de Boole (3)

- Théorèmes de Boole
 - Simplification des équations booléennes
 - 8 primaires :
 - $x \cdot 0 = 0$
 - $x \cdot 1 = x$
 - $x \cdot x = x$
 - $x \cdot \neg x = 0$
 - $x + 0 = x$
 - $x + 1 = 1$
 - $x + x = x$
 - $x + \neg x = 1$

Algèbre de Boole (4)

- Théorèmes de Boole (suite)
 - Associativité
 - $x + (y + z) = (x + y) + z = x + y + z$
 - $x \cdot (y \cdot z) = (x \cdot y) \cdot z = x \cdot y \cdot z$
 - Commutativité
 - $x + y = y + x$
 - $x \cdot y = y \cdot x$
 - Distributivité
 - $x \cdot (y + z) = x \cdot y + x \cdot z$
 - $(w + x) \cdot (y + z) = w \cdot y + w \cdot z + x \cdot y + x \cdot z$
 - Divers (à démontrer)
 - $x + x \cdot y = x$
 - $x + /x \cdot y = x + y$

Algèbre de Boole (5)

- Théorèmes de Morgan :

$$\overline{(x + y)} = \bar{x} \cdot \bar{y}$$

$$\overline{(x \cdot y)} = \bar{x} + \bar{y}$$

- Permettent d'implanter à partir de portes logiques imposées n'importe quelle équation booléenne.
- Exemple : Exprimer l'expression suivante qu'avec des Nands à 2 entrées: $S = A/BC + B/D$

Algèbre de Boole (6)

- Exercice 1 : Simplifier l'expression booléenne suivante :

$$S = ABC + AB/C + A/BC$$

- Implanter la fonction XOR en utilisant des portes Nand à 2 entrées
- Exprimer l'exemple du transparent n°38 qu'avec des NOR à 2 entrées

Tableau de Karnaugh (1)

- Outil graphique
- permet de simplifier de manière méthodique une équation booléenne
- optimisation du passage d'une table de vérité à une implantation en portes logiques

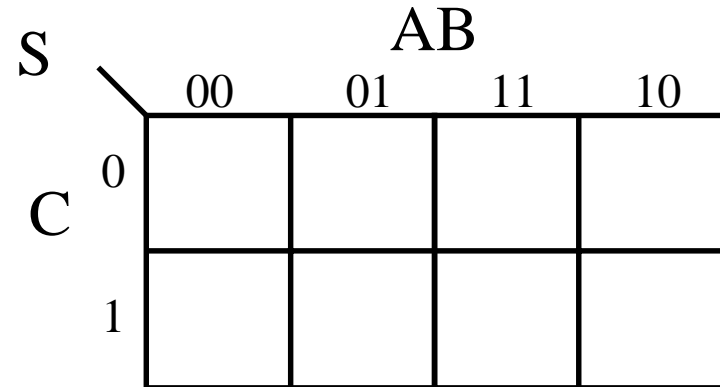
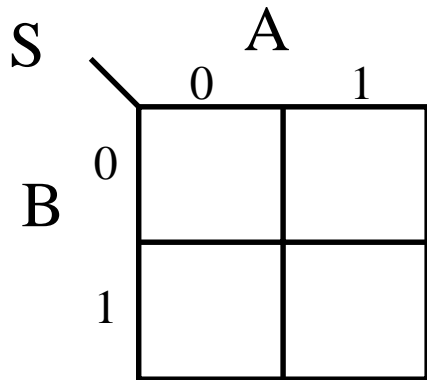


Tableau de Karnaugh (2)

- Méthode :
 - On remplit le tableau en fonction de la table de vérité
 - On réunit tout les "1" adjacents par doublet, quartet, etc ... en essayant d'en réunir le plus possible à la fois
 - On effectue un OU entre les correspondances de chaque "1" réunis et on en déduit l'équation simplifiée de cette réunion :

		AB			
		00	01	11	10
C	0		1	1	
	1				

$$S = \neg AB/C + AB/C$$

$$S = B/C(\neg A + A)$$

$$S = B/C$$

Tableau de Karnaugh (3)

- Exercice : Dédurre l'équation booléenne simplifiée du tableau de karnaugh suivant :

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

Vérifier qu'il n'est pas possible de simplifier encore l'équation.

Tableau de Karnaugh (4)

- Solution : $S = \neg DA + D/C + \neg C/B + D/AB$

Diagram of a 4-variable Karnaugh map for function S. The vertical axis is labeled 'DC' and the horizontal axis is labeled 'AB'. The map contains 1s in cells (00,00), (00,10), (01,10), (10,00), (10,01), (10,10), and (10,11). Shaded regions indicate prime implicants: a vertical bar for $\neg C$, a horizontal bar for D , a vertical bar for $\neg B$, and a horizontal bar for A .

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

Contraintes temporelles (1)

- Propriétés dynamiques d'une porte logique
 - Une porte logique est composée de transistors
 - Ces transistors fonctionnent en "saturé-bloqué"
 - Ces transistor mettent un "certain temps" pour passer d'un état passant à un état bloqué et inversement
 - => Temps de propagation !
 - Temps que met une sortie pour être éventuellement modifiée par le changement d'une entrée
 - Il faudra en tenir compte lors de la conception d'un montage numérique

Contraintes temporelles (2)

- Aléas de commutation
 - Aléas de propagation
 - dû à des temps de propagation différents (chemins différents) dans un même bloc logique
 - engendre des changements intempestifs et non prévus de la sortie du bloc logique
 - Solution : prendre en compte tout les "1" adjacents même s'il y a redondance
 - Aléas de simultanéité
 - Modification simultanée de 2 entrées, qui théoriquement n'engendre pas de changement à la sortie, mais qui crée une pulse (pic, spike, glitch) en sortie

Exercices

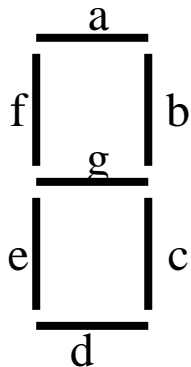
- Décodeur binaire-DCB

- Donner les équations logiques de conversion de nombres binaires de 4 bits "dcba" en un nombre DCB de 8 bits "HGFE DCBA"

$$5_{10} \rightarrow 0101_2 \rightarrow 0000\ 0101_{\text{dcb}}$$

$$12_{10} \rightarrow 1100_2 \rightarrow 0001\ 0010_{\text{dcb}}$$

- Afficheur 7 segments à cathode commune :



- Quel niveau logique activera l'afficheur ?
- Donner les équations logiques de tout les segments
- Dessiner le schéma électrique du segment "g"
- en tenant compte d'un temps de propagation dans les portes logiques, dessiner le chronogramme lorsque les entrées DCBA = 0110 et que C passe de 1 à 0. Y'a t-il un aléa sur g ??
- Comment l'éliminer ??

Chapitre 3

Arithmétique binaire et transmissions de données

Arithmétique binaire :

Addition binaire (1)

- Analogue à l'addition de 2 nombres décimaux
 - $0 + 0 = 0$
 - $1 + 0 = 1$
 - $1 + 1 = \underline{1} 0$ (retenue reportée toujours sur le rang de gauche)
 - $1 + 1 + 1 = \underline{1} 1$ (retenue)
 - ex : $011 (3)$
 $\quad + 110 (6)$
 $\quad \underline{\quad}$
 $1001 (9)$
- toutes les opérations arithmétiques de base sont dérivées de l'addition.

Addition binaire (2)

- Exercice : Additionneur 2 bits
 - Donner la table de vérité d'un additionneur 2 bits (avec génération de retenue) appelé communément "half adder" (demi-additionneur)
 - Donner un schéma d'implantation en utilisant Karnaugh
 - Cet additionneur ne tiens pas compte d'une retenue entrante. Que faut-il faire pour effectuer cela en utilisant une cellule identique pour obtenir un additionneur complet.

Représentation des nombres (1)

- nombres entiers
 - binaire pur, DCB, gray ...
- nombres entiers relatifs
 - Soit on utilise un bit de signe, donc un bit de plus... (0 => positif)
 - Notation difficile à implanter
- Soit on utilise la représentation en **complément à 2** (la plus utilisée)

Représentation des nombres (2)

- On prend le complément à 1 du nombre (c.a.d. on inverse les 1 et les 0) et on ajoute 1.
- Ex : le complément à 2 de 0011_2 (3_{10}) est : 1101_{2c}
- Permet de faire une addition ou une soustraction avec un seul et même additionneur !! En effet, 1101 représente en fait -3 et donc :

$$\begin{array}{r} 0011 \quad (3) \\ + \underline{1101} \quad (-3) \\ \hline 1\ 0000 \quad (0) ! \end{array}$$

Addition en complément à 2 (1)

- Le nombre positif sera précédé d'un 0
- Le nombre négatif sera complémenté à 2 et précédé d'un 1
- Pour les additionner, on prendra tel quel le nombre positif et le complément à deux du nombre négatif
- Attention :
 - le bit de signe fait partie intégrante de l'addition
 - si le résultat de l'addition est négatif, le nombre obtenue est le complément à 2 du résultat final.
 - Il faut toujours s'assurer que le résultat de la somme aura le même nombre de bit que les opérandes, sinon il y a **dépassement**

Addition en complément à 2 (2)

- Ex1 : on veut additionner (+9) a (+5)

$$\begin{array}{r} 0\ 1001\ (+9) \\ +\ 0\ 0101\ (+5) \\ \hline 0\ 1110\ (+14) \end{array}$$

- Ex2 : on veut additionner (-11) a (+5)

$$\begin{array}{r} 1\ 0101\ (\text{complément à 2 de } 1\ 1011 \text{ soit } -11) \\ +\ 0\ 0101\ (+5) \\ \hline 1\ 1010 \end{array}$$

- Nous obtenons un nombre négatif donc complémenté à 2, soit en fait la représentation de -6 (1010 est le complément à 2 de 0110)

Soustraction binaire

- Il suffit de faire une addition avec le complément à deux du nombre que l'on veut soustraire (y compris son bit de signe si c'est un nombre signé)
- Ex : la soustraction de (+9) et (+4) s'écrit donc :

$$\begin{array}{r} 0\ 1001 \quad (+9) \\ + \underline{1\ 1100} \quad (-4 \text{ en complément à } 2) \\ \hline 1\ 00101 \quad (+5, \text{ la retenue étant rejetée}) \end{array}$$

Exercices

- Effectuer les opérations suivantes avec la notation en complément à 2. Utiliser pour chaque nombre 8 bits (bit de signe compris).
 - Additionner +8 et +6 ; +19 et -24, -48 et -70
 - Soustraire +16 et +17; +21 et -13 ; -36 et -15

Multiplication binaire (1)

- Même principe que la multiplication décimale !!
- Ex: multiplier 9 et 11

$$\begin{array}{r} 1001 \text{ (9)} \\ + \underline{1011 \text{ (11)}} \\ 1001 \\ 1001 \\ 0000 \\ \underline{1001} \\ 1100011 \text{ (99)} \end{array}$$

- **Attention :**
 - Les nombres sont représentés en notation binaire **exacte**
 - on ne tient pas compte dans la multiplication du bit de signe.

Multiplication binaire (2)

- Les machines numériques n'additionnent que 2 nombres à la fois. La somme des produits partiels se fera donc 2 par 2, en prenant à chaque fois le résultat de la somme précédente
- **Une multiplication d'un nombre par 2 revient simplement à faire un décalage à gauche.**
- Si la multiplication utilise le complément à deux
 - Si les 2 opérandes sont positives, pas de problème
 - Si les 2 opérandes sont négatives (donc représentés en complément à 2) on les re-complémente à 2 et on effectue la multiplication (on rajoute le bit de signe = 0).

Multiplication binaire (3)

- Si la multiplication utilise le complément à deux (suite)
 - Si l'une des 2 opérandes est négative (donc représentés en complément à 2), on la re-complémente à 2 et on effectue la multiplication. Le résultat final sera le complément à 2 du résultat de la multiplication (on rajoute le bit de signe = 1).
- **Exercice :**
 - Multiplier les chiffres 4 et -6

Division binaire (1)

- Même principe que la division décimale !!
- Ex : diviser 12 par 2 :

$$\begin{array}{r|l} \overline{1100} & 10. \\ 10 & 110 \\ 00 & \\ 0 & \end{array}$$

- **Une division d'un nombre par 2 revient simplement à faire un décalage à droite.**
- **Attention :**
 - Les nombres sont représentés en notation binaire **exacte**
 - on ne tient pas compte dans la multiplication du bit de signe.

Division binaire (2)

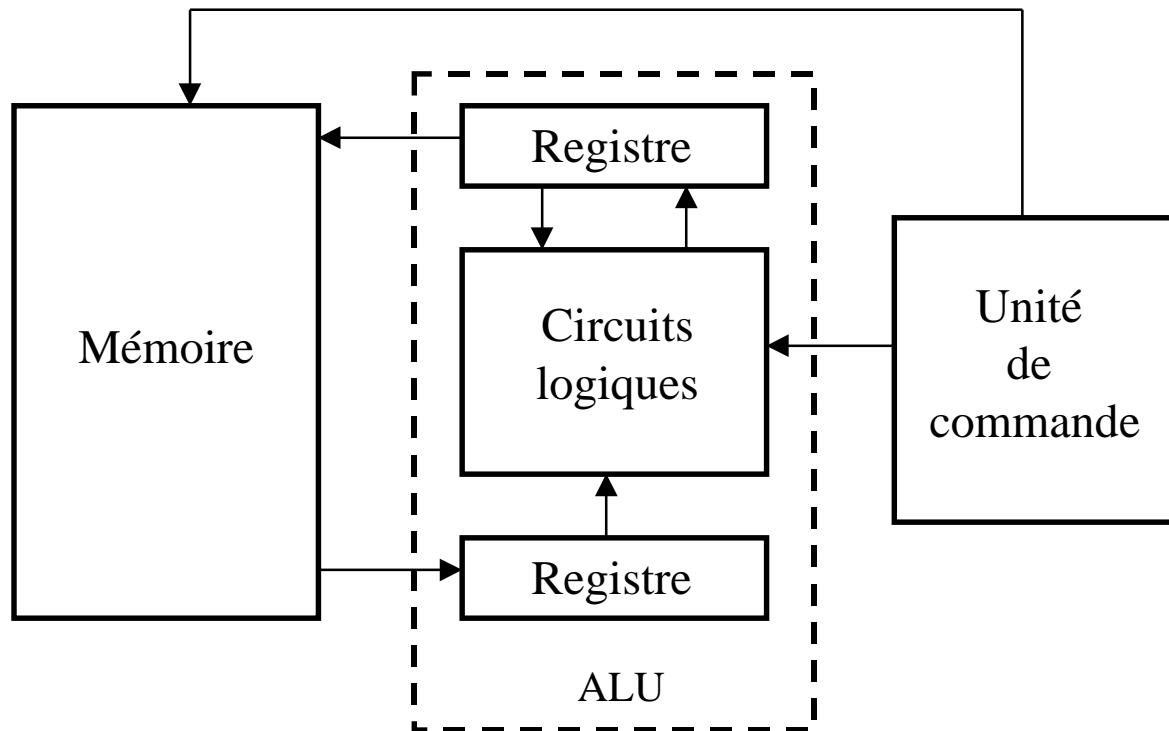
- Si la division utilise le complément à deux :
 - Si les 2 opérandes sont négatives (donc représentés en complément à 2) on les re-complémente à 2 et on effectue la division.
 - Si les 2 opérandes ont le même signe, le quotient est laissé sous sa forme positive (on rajoute un bit de signe =0)
 - Si les 2 opérandes sont de signes opposés, le quotient est complémente à 2 pour obtenir un nombre négatif (on rajoute un bit de signe =1)
- **Exercice :**
 - Diviser 9 et -3 ; 10 et 4

Unité Arithmétique et Logique (ALU) (1)

- Élément effectuant des opérations arithmétiques et logiques
- Un des éléments principaux d'un processeur avec l'unité de commande qui la contrôle et la mémoire de laquelle elle reçoit des données et à qui elle envoie le résultat des calculs.

Unité Arithmétique et Logique (2)

- Constituer essentiellement de circuits logiques (additionneur , ...) et de 2 registres à décalage



Unité Arithmétique et Logique (3)

- Ex : le circuit 74–181
 - 2 entrées et une sortie sur 4 bits
 - 16 opérations d'arithmétiques possibles (sélection par un mot de 4 bits)
 - 16 fonctions logiques possibles.

Transmission de données

Modes de transmission

- parallèle : plus rapide mais plus coûteux (connectique)
- série : plus lent, conversion // - série obligatoire, mais moins coûteux
- Dans tous les cas :
 - nécessité d'utiliser un protocole de communication entre les objets communicants (début et fin de mots)
 - nécessité d'augmenter la tolérance au bruit du système

=> détection d'erreur

Contrôle de parité (1)

- Rajouter au mot un bit supplémentaire (bit de parité) qui prendra la valeur 1 ou 0 suivant la méthode de contrôle de parité choisi :
 - **Parité paire** : on fixe le bit de parité pour que le nombre total de 1 dans la représentation codée (y compris le bit de parité) soit un nombre pair.
 - **Parité impaire** : l'inverse
 - Ex : le système émetteur rajoute au nombre 1000011 un 1 pour respecter la parité paire. Si le système récepteur détecte un nombre impair de 1 alors que le bit de parité est à 1, c'est que le mot reçu est faux. Génération d'un signal d'erreur.
 - Le circuit 74–280 détermine la parité d'un mot binaire.
 - Limitation : on ne connaît pas quel bit est faux

Contrôle de parité (2)

- Des systèmes utilisant le **code de Hamming** par exemple, qui rajoute plusieurs bits au mot transmis, permettent, à la réception, de savoir quel bit est faux et de le corriger.
- **Exercice :**
 - Etablir le circuit qui génère le bit de parité pour la transmission d'un mot de 2 bits. On considèrera le cas du codage en parité paire (even) ou impaire (odd)
 - Etendre à un mot de 8 bits en utilisant le circuit de parité 2 bits
 - Comparer le résultat au schéma fonctionnel du circuit 74-280. Vérifier que les bits de parité fournis sont identiques.

Liaison RS 232

- Problème des transmissions parallèles : beaucoup de fils !! et un câble donnée devient rapidement obsolète.
- Solution simple et très utilisée : la liaison RS232
 - conversion //, série, transmission sur 2 fils : un de référence et un de signal (paire torsadée pour limitée les parasites par induction électromagnétique)
 - "codage" de la donnée (protocole de début et de fin de mot)
 - liaison utilisée dans beaucoup de petits appareils domotiques (calculatrice, organizer, portable), présente dans chaque ordinateur.
 - Faible débit : 1200 à 9600 bauds

Chapitre 4

Circuits logiques séquentiels

Circuits séquentiels ??

- Un système séquentiel fait intervenir une **variable interne** en plus des entrées pour faire évoluer la sortie.
- Cette variable est appelée **l'état interne**.
- L'évolution d'un circuit séquentiel est cadencée par un signal périodique appelé "**horloge**".
- Il y a 2 types de circuit séquentiel :
 - Les circuits **asynchrones**
 - Les circuits **synchrones**

Spécificités d'un circuit asynchrone

- Chaque coups d'horloge déclenche un processus combinatoire dans le bloc logique asynchrone
- Assez simple à implanter
- Attention à la synchronisation entre différents blocs travaillant en parallèle !!

Spécificités d'un circuit synchrone

- La prise en compte des entrées et la validation des sorties ne sont effectuées qu'à un instant donné (ex : le front montant de l'horloge).
- Tout le système est cadencé par la même horloge ce qui garanti une synchronisation parfaite de tout les calculs effectués en //.
- Les systèmes synchrones sont plus simples à concevoir et à dépanner parce qu'on est sur qu'au coup d'horloge les données et les états internes sont prêts !!

Contraintes liées à l'horloge

- Les circuits réagissent à un front d'horloge (montant en général) ou un niveau d'horloge (1 ou 0)
- Pour avoir un fonctionnement correct dans un circuit synchrone, il faut respecter certaines contraintes temporelles, liées aux propagations dans les circuits :
 - Temps de **pré positionnement** (**setup** time) : Il faut garantir que les données sont prêtes au coup d'horloge.
 - Temps de **cycle** : Il faut garantir que le temps de calcul des variables internes et des sorties est inférieur à la période de l'horloge.
 - Temps de **maintien** (**hold** time) : Il faut garantir que les données soient présentes suffisamment longtemps après le coups d'horloge de manière à ce qu'elles soient prises correctement en compte par le système.

Reset et set

- Dans la plupart des circuits synchrones, il y aura des commandes asynchrones qui permettront à n'importe quel moment de positionner le système dans un état donné au niveau des variables internes.
- Les plus utilisées sont :
 - La remise à zéro (**reset**)
 - La remise à un (**set**)
- Très utilisé lors de la mise sous tension d'un circuit:
 - on effectue souvent un Reset automatique pour être sur de démarrer le système à partir d'un état stable !!

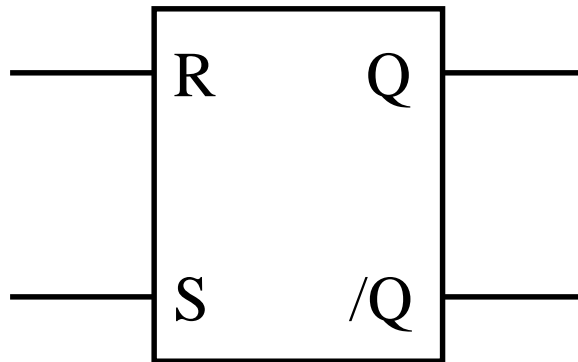
Circuits séquentiels usuels

Éléments de mémorisation (1)

- Ces éléments sont nommés "bascule" (Flip-flop)
- Ces bascules sont en général synchrones (états des sorties re-évaluées à chaque coups d'horloge)
- Possèdent 2 sorties : Q et /Q
- Disponibles en plusieurs versions : avec ou sans Set et/ou Reset

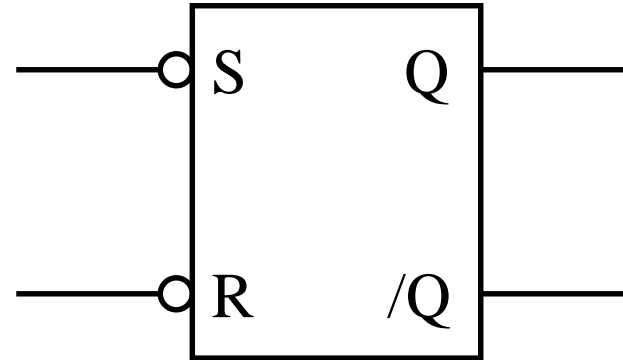
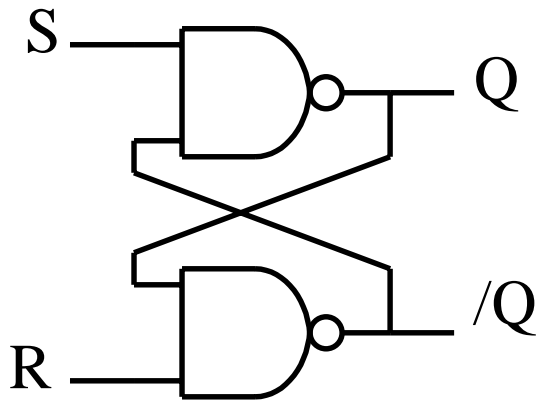
Eléments de mémorisation (2)

- **Bascule RS :**
 - La bascule la plus élémentaire en terme de porte logique!!
 - Bascule asynchrone
 - Se construit à l'aide de 2 portes NAND ou NOR et fournit une sortie et son complément qui est l'image du code de ses 2 entrées



Eléments de mémorisation (3)

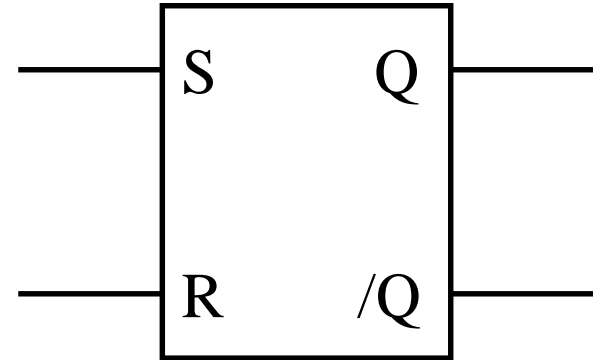
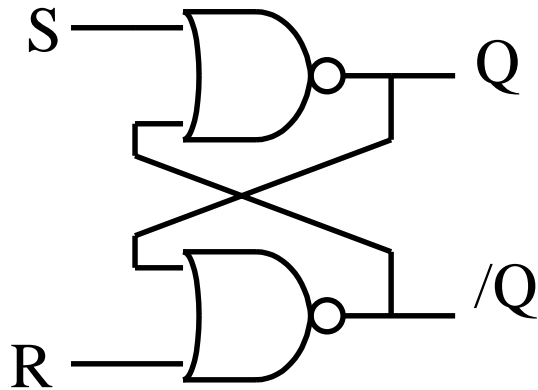
- **Bascule RS** : en portes Nand



S	R	sortie
0	0	ambiguë
0	1	Q = 1
1	0	Q = 0
1	1	inchangée

Éléments de mémorisation (4)

- **Bascule RS** : en porte Nor



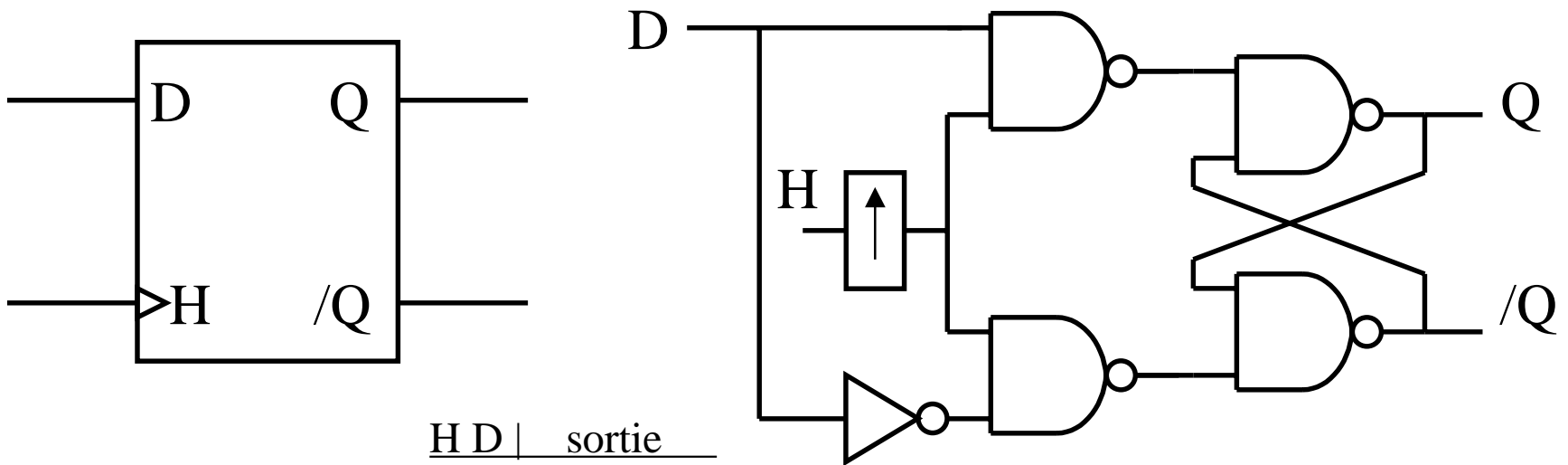
S	R	sortie
0	0	inchangée
0	1	Q = 0
1	0	Q = 1
1	1	ambiguë

Eléments de mémorisation (5)

- **Exercice :**
 - Modifier la bascule RS "Nand" de manière à la rendre synchrone à un niveau haut d'un signal d'horloge (pour rendre la bascule sensible à un front, on ajoutera un détecteur de front)

Éléments de mémorisation (6)

- Bascule D :
 - La plus utilisée car la plus simple !!
 - Bascule synchrone.

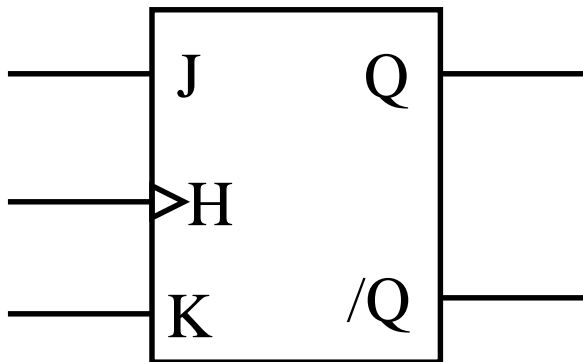


H	D	sortie
0	X	inchangée
1	0	Q = 0
1	1	Q = 1

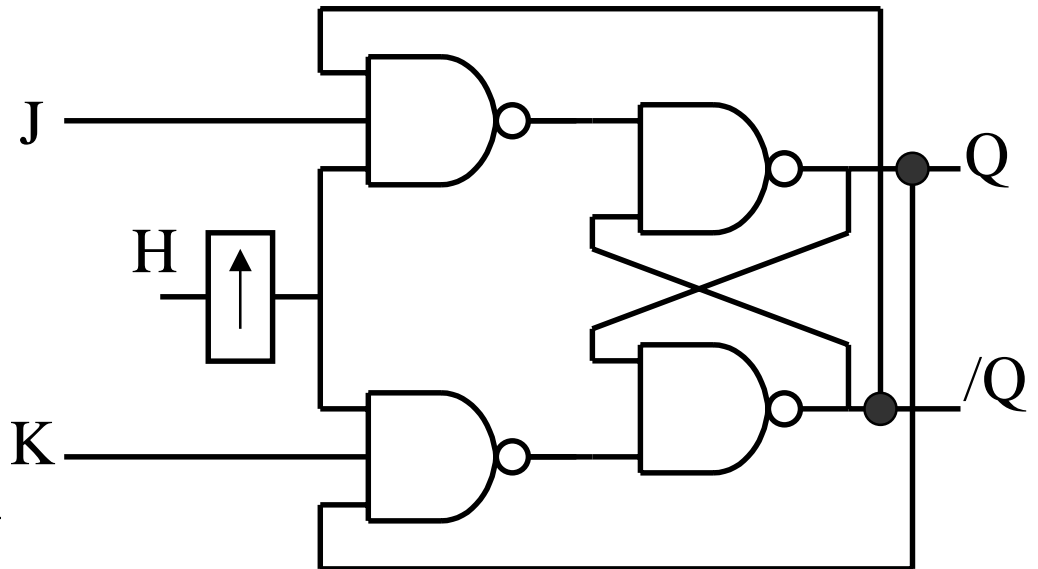
Eléments de mémorisation (7)

- **Bascule JK :**

- Proche de la bascule RS mais plus de combinaison interdite
- J= set, K= reset

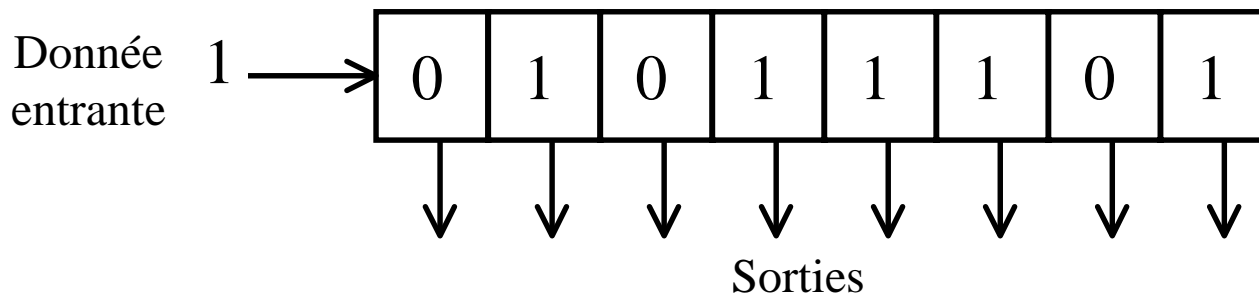


J	K	Q
0	0	inchangée
0	1	Q = 0
1	0	Q = 1
1	1	/Q

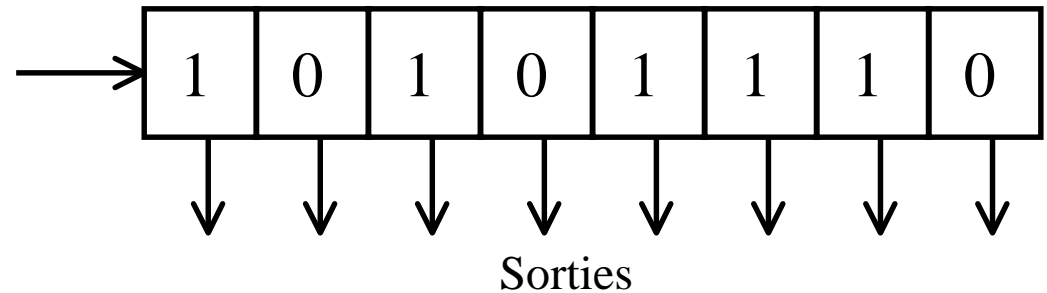


Les registres à décalage (shift register)

- C'est un ensemble de bascules qui permettent de décaler à chaque coup d'horloge le mot binaire qu'elles représentent à un instant donné.

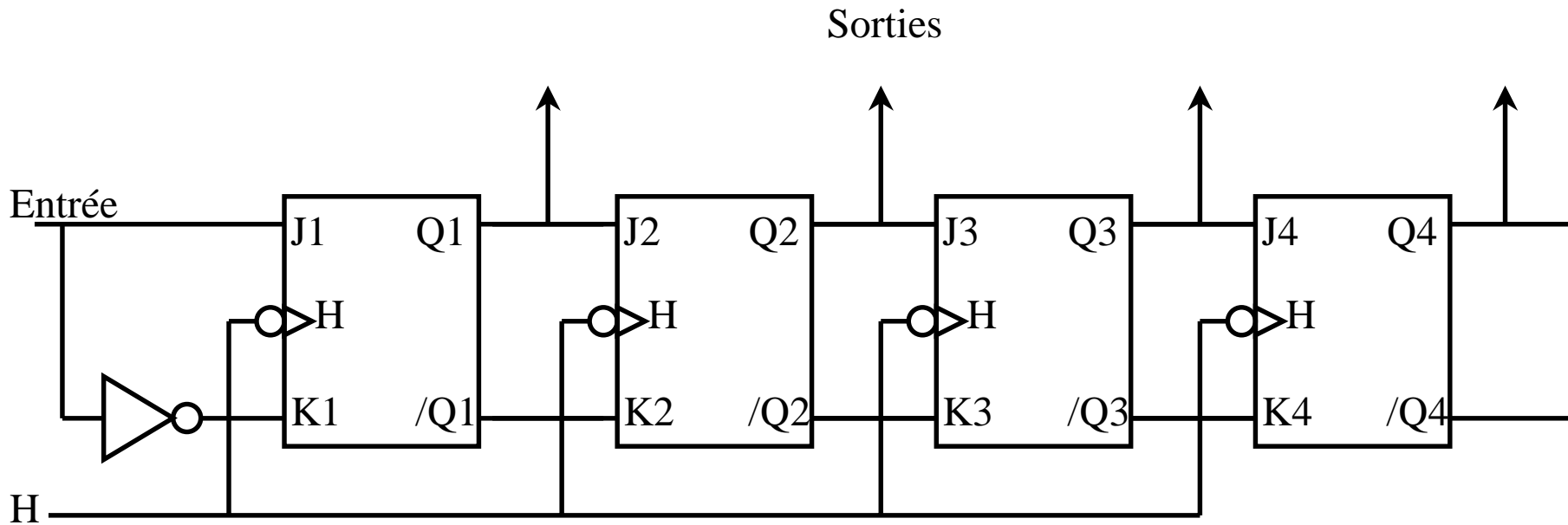


Décalage à droite



Les registres à décalage (2)

- En bascule JK :



Les registres à décalage (3)

- Il existe de nombreux types de circuits "registre à décalage" :
 - A écriture et lecture // : 74174 (6 bits) et 74178 (4 bits et une entrée série également)
 - A écriture série, lecture série : 4731B (64 bits)
 - A écriture //, lecture série : 74165 (entrée série ou //, 8 bits)
 - A écriture série, lecture // : 74164 (8 bits)
 - Universel : 74299
 - Lecture/écriture série ou //
 - Décalage bidirectionnel
 - mémorisation

Les compteurs

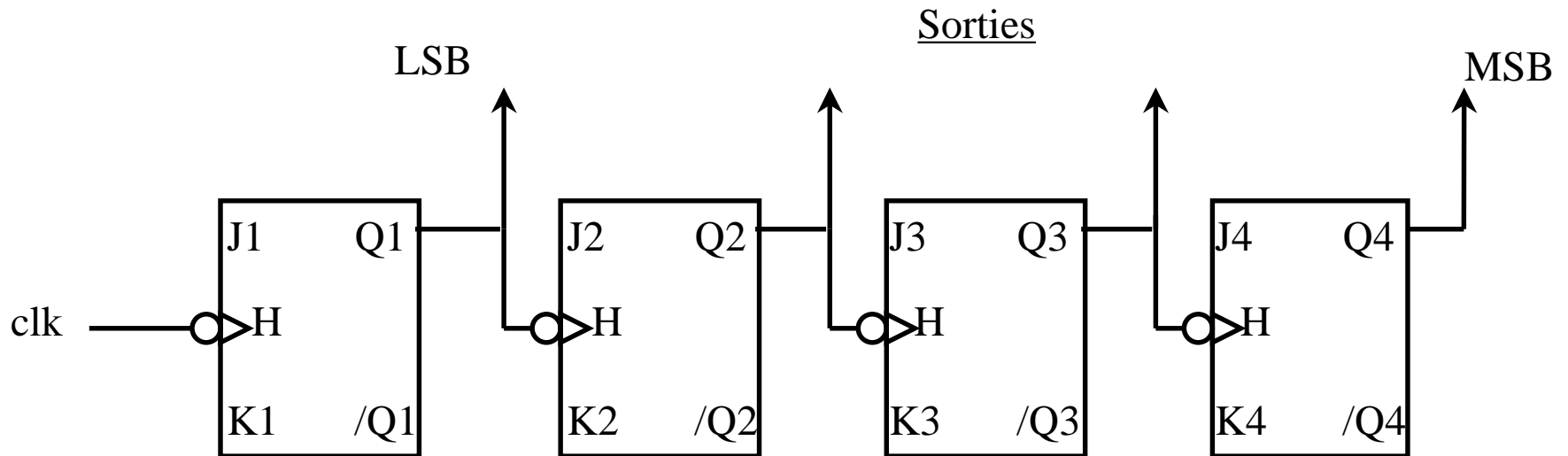
- Très utilisés en numérique !!
- Constitués de Bascule D ou JK
- Permettent de générer une séquence de nombres binaires
- Avec possibilité de :
 - remise à zéro
 - chargement d'une valeur donnée
 - compteur/décompteur
- 2 types de compteurs :
 - synchrones
 - asynchrones

Compteur asynchrones (1)

- Dits "à propagation", mais cadencés par une horloge.
- Les plus simples à mettre en œuvre.
- Chaque bascule correspond à un poids binaire.
- Ex : 4 bascules implique compteur par $2^4 = 16$
=> On dit que l'on a un compteur **modulo 16**
- Inconvénients :
 - Il y a propagation des signaux dans le compteur, il est donc lent à fournir la bonne valeur en sortie après le coups d'horloge, donc limitation en fréquence
 - attention à la synchronisation avec d'éventuels autres blocs logiques.

Compteur asynchrones (2)

- Compteur modulo 16 à bascules JK :



Nota : Toutes les entrées sont à 1 (mode "toggle")

Compteur asynchrones (3)

- Décompteur : (si la clk est active au front descendant), on utilisera la sortie complémentée d'une bascule pour piloter la clk de la bascule suivante
- avec des bascules D, on reliera D à /Q
- Exemple de compteur : 74293
 - Modulo 16 (4 JK)
 - entrées clock des 2 premières bascules externes (possibilité d'un compteur 3 bits)
 - le reset se fait par le Nand de 2 entrées appelées Master Reset. Le reset est effectué quand les deux MR sont à 1

Compteur asynchrones (4)

- Compteur modulo $X < 2^n$
 - on détecte le nombre X par un système à portes logiques
 - on remet à zéro le compteur
 - entrée reset obligatoire
- **Inconvénient** : il y a un aléa car on passe momentanément à l'état supérieur au modulo.

Compteur asynchrones (5)

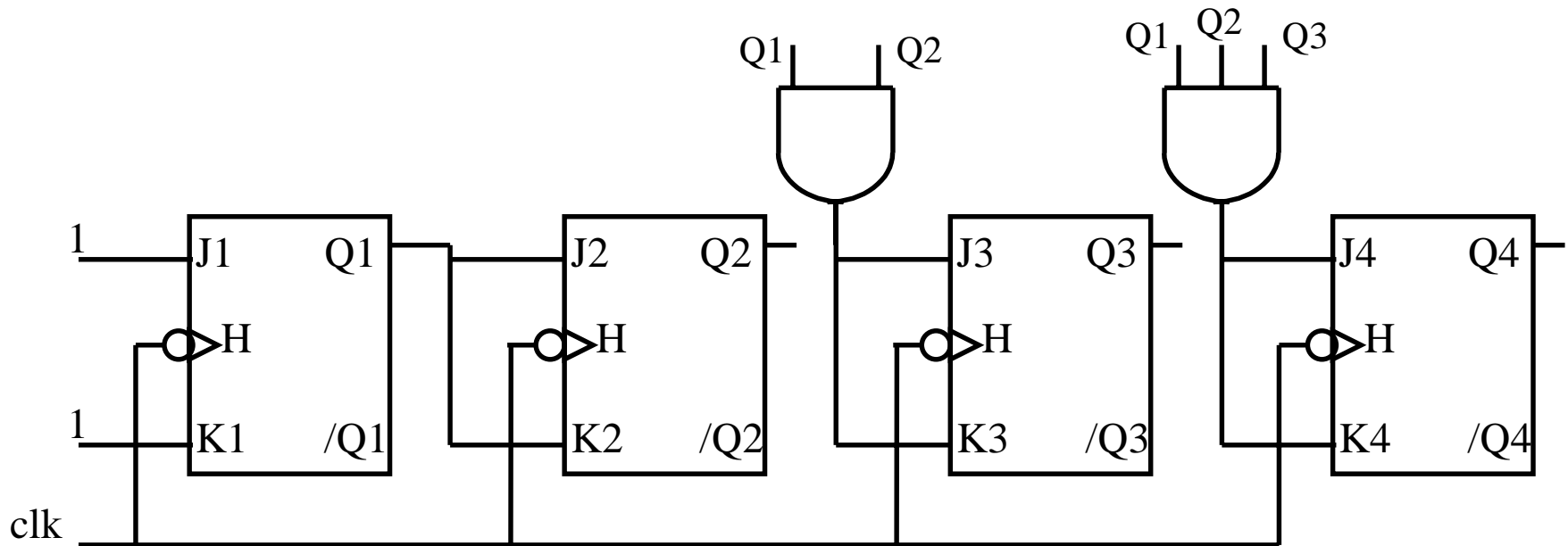
- **Exercice 1** : compteur modulo 6
 - Combien faut-il de bascules JK ??
 - Dessiner le schéma complet du compteur
 - Faire un chronogramme.
- **Exercice 2** : Utiliser le circuit 74293 pour effectuer un compteur modulo 14

Compteur synchrones (1)

- Dits "parallèles" car toutes les bascules sont activées au même coups d'horloge.
- Il faut donc un système logique qui permet de savoir à chaque coups d'horloge ce que doit faire une sortie de bascule, cad commuter ou non...
- Exemple de compteur synchrone : 74160
- Plus complexe : le 74193
 - Compteur-décompteur modulo 16
 - Précharge d'un nombre de début de comptage possible

Compteur synchrones (2)

- Compteur modulo 16 :



- Pour un décompteur, on prendra les sorties complémentées.

Compteur synchrones (3)

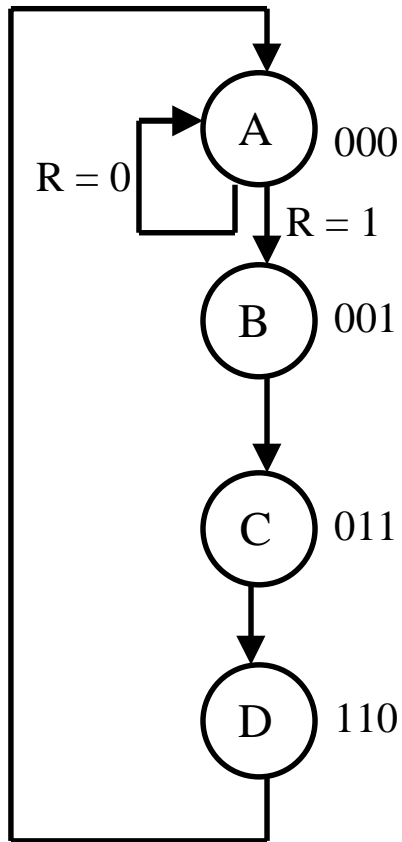
- pour un compteur synchrone modulo N , les entrées d'une bascule JK seront commandées par un ET de **toutes** les sorties précédentes.
- Exemple d'application : Le diviseur de fréquence
 - Si l'on fait un compteur modulo n , on peut obtenir un signal qui a une période n fois plus grande que le signal d'horloge !!

Conception de compteurs synchrones :

Les machines à état

- La conception d'un compteur synchrone quelconque revient à faire **la synthèse d'une machine à état**
- Une machine à état contient n états, qu'il faudra obtenir dans un ordre donné, en fonction éventuellement d'entrées extérieures à la machine
- On commence par établir le **graphe des états** de la machine, donc la séquence de nombres à obtenir en sortie de la machine

Graphe des états



- Chaque état est représenté par une lettre dans une bulle.
- Les liens entre bulles représentent les transitions entre états lors d'un coup d'horloge
- Une transition peut dépendre d'éventuelles conditions (alarme , ...), on parle alors de transition conditionnelle.
- On mettra à côté de chaque bulle, l'état logique des sorties.

Synthèse d'une machine à état

- Faire un tableau dans lequel on met de gauche à droite :
 - Les états "présents" des sorties ainsi que les éventuelles variables
 - Les états "suivants" des sorties (au coups de clock suivant)
 - L'état des entrées JK ou D des bascules pour passer de l'état présent à suivant.
- Déterminer l'équation logique (karnaugh) de chaque entrées de bascule en fonction des sorties, puis câbler.
- Les variables externes peuvent être, soit synchronisées avec l'horloge (machine à état de Moore), soit prises en compte dans l'instant (machine de Mealy)

Synthèse d'une machine à état (2)

- **Exercice :**
 - Faire le graphe d'état de 2 feux tricolores, l'un étant prioritaire lorsqu'une voiture se présente devant, l'autre étant vert sinon.
- On peut implanter ce système de plusieurs manières :
 - 6 bascules fournissant les 6 signaux R1, R2, V1, V2, O1, O2
 - un compteur par 6 et un décodeur 3 vers 6 ou une mémoire
- Cette dernière architecture s'appelle **un séquenceur**.

Séquenceur

- Un séquenceur est un système séquentiel autonome qui exécute un programme (une série d'instruction)
- peut être vu comme un processeur rudimentaire
- L'architecture classique d'un séquenceur est basé sur :
 - un compteur programmable qui cadence la séquence et adresse...
 - une mémoire qui pour une position donnée du compteur fournit un jeu d'instruction (sous forme d'un mot binaire) à ...
 - un décodeur qui reçoit aussi les entrées extérieures et qui prend la décision de continuer le comptage ou de sauter à une position donné du compteur.

Exercice

- Concevoir le système synchrone dont les caractéristiques sont les suivantes :
 - comptage impair (0, 1, 3, 5, 7)
 - bascule D
 - Reset
 - Une entrée asynchrone E force le système à effectuer la séquence 3, 5, 7.

Chapitre 5

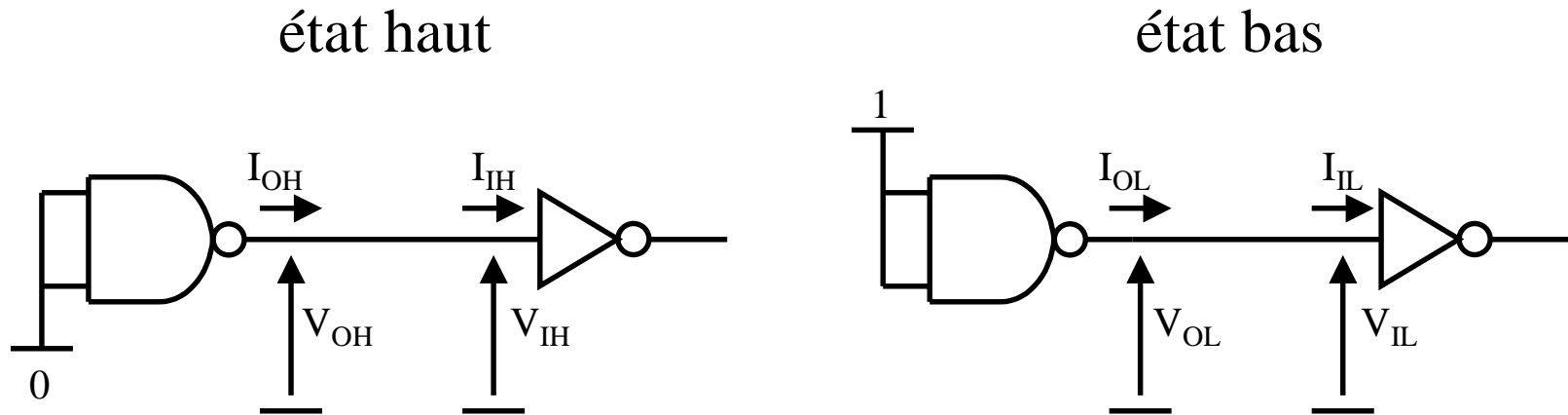
**Circuits logiques :
Paramètres électroniques, circuits,
états haute impédance, bus de données**

Circuits logiques ??

- Constitués de transistors bipolaire ou MOS
 - fabriqués sur un substrat de silicium
 - intégration de plusieurs millions de transistor MOS (VLSI)
- Plus l'intégration est forte, plus la performance atteinte est grande pour une fonctionnalité visée :
 - vitesse
 - consommation
 - dispersion
 - fiabilité
 - coût

Paramètres électriques (1)

- Niveaux de tensions et de courants



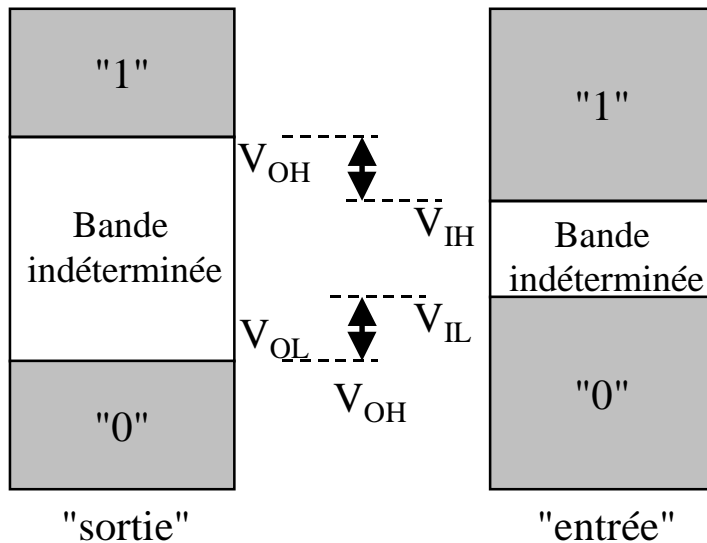
=> Définition des courants d'entrées à l'état haut et bas

=> Définition des niveaux logiques

Paramètres électriques (2)

- **Immunité aux bruits**

- définit l'aptitude d'un circuit à tolérer des tensions parasites sur son entrée.
- Se traduit par la **marge de sensibilité aux bruits** haute (M_{NH}) et basse (M_{NL})



=> Marge de sensibilité à l'état haut :

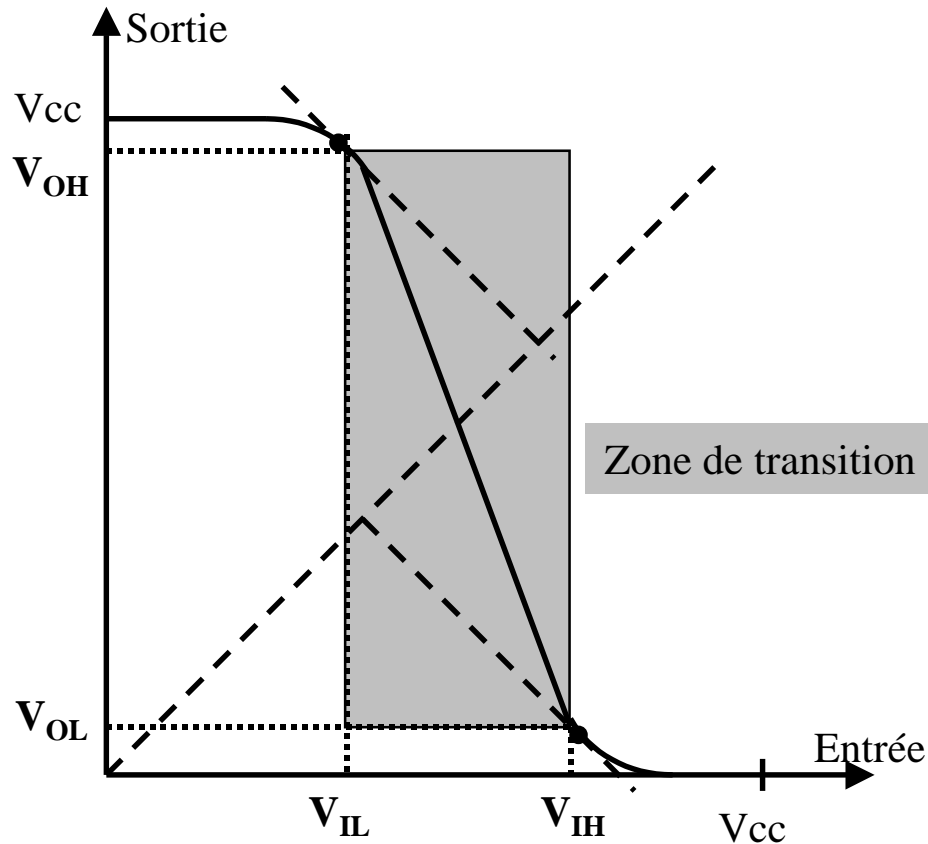
$$M_{NH} = V_{OH} - V_{IH}$$

=> Marge de sensibilité à l'état bas :

$$M_{NL} = V_{IL} - V_{OL}$$

Paramètres électriques (3)

- Caractéristique de transfert d'une porte logique



Paramètres électriques (4)

- Retards dus aux **temps de propagation**
 - T_{plh} : retard pour passer du niveau bas au niveau haut
 - T_{phl} : retard pour passer du niveau haut au niveau bas
 - se mesure de 50% du signal d'entrée à 50% du signal de sortie

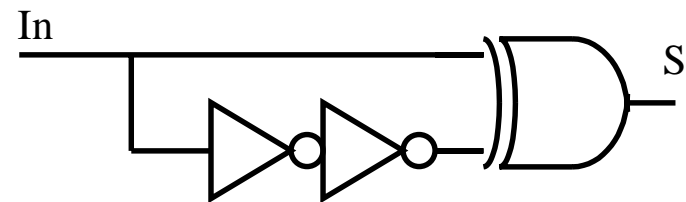
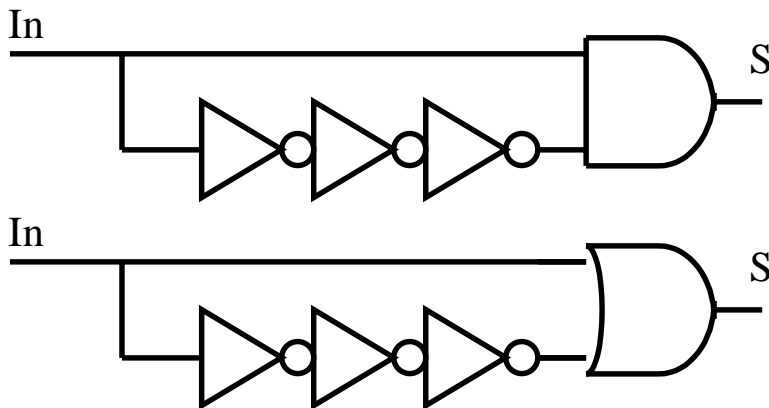
- **Temps de commutation**
 - t_r : temps de montée
 - t_f : temps de descente
 - se mesure entre 10% et 90% du signal

Paramètres électriques (5)

- **Sortance (Fanout)**
 - Nombres de portes logiques standards qui peuvent être pilotées correctement (au niveau temporel) par une seule sortie
 - en général, on prend un Fanout de 10
 - rapport entre le courant $I_{o(h,l)}$ et le courant $I_{i(h,l)}$ d'une porte
 - existe donc une sortance pour l'état bas et haut !!
 - On prendra la plus basse des deux
- **Consommation**
 - $P_{D(moy)} = I_{CC(moy)} \times V_{CC}$
 - $I_{CC(moy)} = (I_{CCH} + I_{CCL})/2$

Paramètres électriques (6)

- **Exercice : Détecteurs de front**
 - Etudier les circuits suivants en prenant en compte les temps de propagation pour les portes inv, OU, ET, Xor : $T_{\text{phl}} = 15\text{ns}$; $T_{\text{plh}} = 20\text{ns}$
 - Déterminer la durée du signal de sortie lorsque l'entrée passe de 0 à 1 et de 1 à 0
 - Quelle est la fréquence de travail maximum de ces circuits



Famille logique TTL (1)

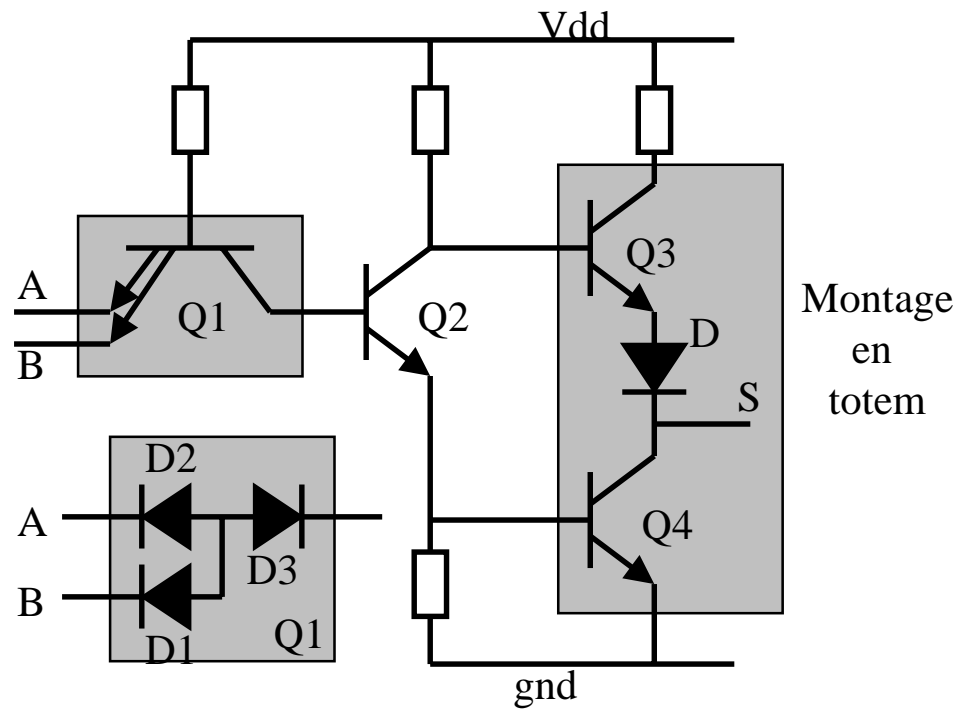
- TTL : Transistor Transistor Logic (1964, Texas Instrument)
- Composée de transistors bipolaires NPN fonctionnant en saturé/bloqué
- Propriétés :
 - Les circuits TTL commencent par "74-(classe)(numéros)"
 - une entrée non connectée équivaut à une entrée à "1"
 - il vaudra mieux faire une connexion à "1"

Famille logique TTL (2)

- Caractéristiques du circuit 74-00 (4 portes Nand):
 - $V_{cc} = 5 \text{ Volts } \pm 0.25\text{V}$
 - Température d'utilisation : $0-79^{\circ}\text{C}$
 - $V_{oh \min} = 2.4\text{V}$; $V_{ol \max} = 0.4\text{V}$
 - $V_{il \max} = 0.8\text{V}$; $V_{ih \min} = 2\text{V}$
 - $I_{ih} = 40\mu\text{A}$; $I_{il} = 1.6\text{mA}$
 - $I_{oh} = 0.4\text{mA}$; $I_{ol} = 16\text{mA}$
 - retard de propagation : $\sim 10\text{ns}$
 - puissance dissipée : $P_d \sim 10\text{mW}$ pour une porte Nand à 2 entrées

Famille logique TTL (3)

- Implantation de la porte Nand du circuit 74-00 :



Famille logique TTL (4)

- Exemple d'implantation : la porte Nand (suite)
 - Le montage Totem de sortie permet de limiter la consommation
 - Lorsque la sortie est à "0", Q4 absorbe le courant I_{IL} qui est fourni par le transistor Q1 de l'étage bipolaire suivant.
 - Lorsque la sortie est à "1", Q3 fournit le courant I_{IH} qui est absorbé par le transistor Q1 de l'étage bipolaire suivant.
 - Il y a un pic de courant consommé quand la sortie Totem passe de l'état bas à haut : Q3 commence à conduire avant que Q4 ne soit bloqué
 - => Découplage des alimentations (10 à 100nF)

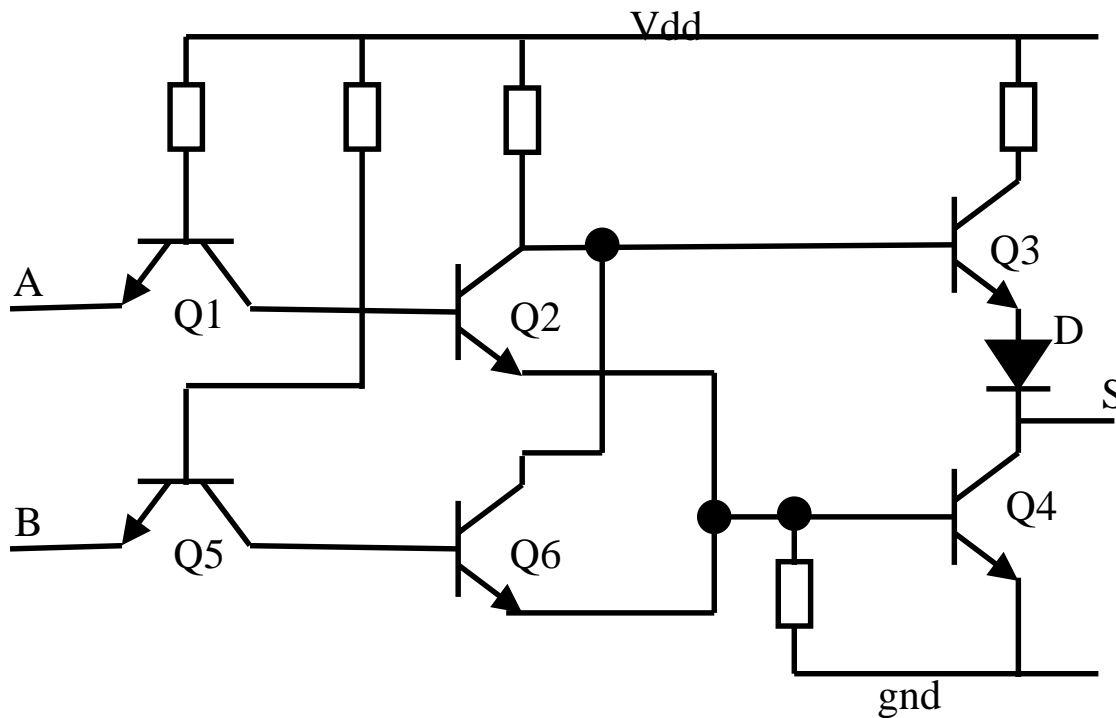
Famille logique TTL (5)

- **Exercice 1 :**
 - Quelles sont les marges de bruit de la porte Nand?
 - Calculer les sortances haute et basse ??
 - A partir du schéma TTL de la porte Nand, dessiner la structure à transistor d'un inverseur.
 - Le circuit 74-04 (8 inverseurs) possède les mêmes caractéristiques que le 74-00. combien une porte Nand peut-elle charger d'inverseurs ?

Famille logique TTL (6)

- **Exercice 2 :**

- Quelle est la fonctionnalité du circuit TTL ci-dessous ??
- Expliquer.



Famille logique TTL (7)

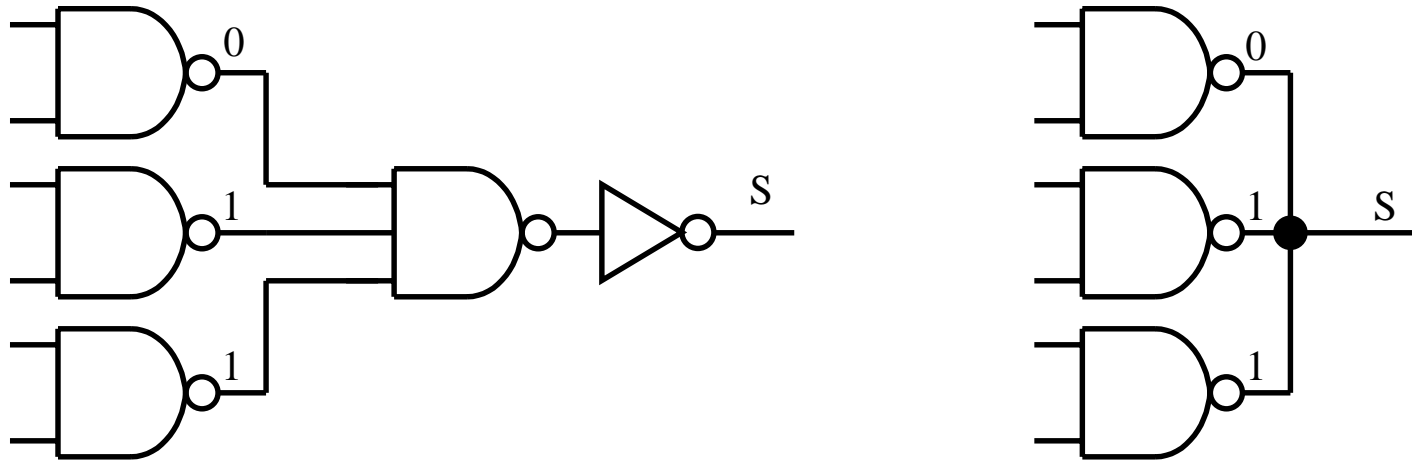
- 74-xx : l'originale (9ns, 35Mhz, 10mW)
- les obsolètes :
 - 74L-xx : faible consommation (33ns, 3Mhz, 1mW)
 - 74H-xx : rapide (6ns, 50Mhz, 23mW)
- les "Schottky":
 - utilise des diodes du même nom afin d'améliorer la commutation en limitant la saturation des transistors.

=> 74S-xx 2 fois plus rapide que 74H, même consommation

 - les obsolètes : 74S-xx et 74LS-xx (moins rapide et moins gourmande)
 - **74AS-xx** la plus rapide (1.7ns, 200Mhz, 8mW)
 - **74ALS-xx**, la nouvelle classique (4ns, 70Mhz, 1.2mW)

Circuits TTL à collecteur ouvert (1)

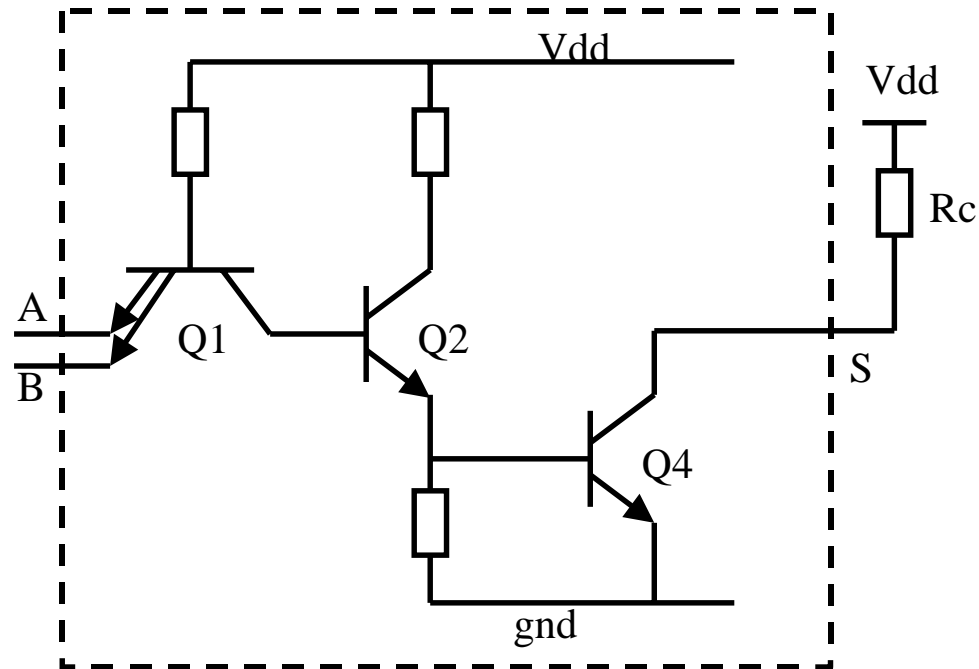
- Simplification des montages par l'utilisation de "ET câblé"



- Si une des 3 Nand a sa sortie à zéro (Q4 passant), cela force le potentiel à zéro.
- Problème : très forte consommation car "court-circuit" éventuel avec les 2 autres Nand si elles fournissent un "1"

Circuits TTL à collecteur ouvert (2)

- **Solution** : circuit à collecteur "ouvert"

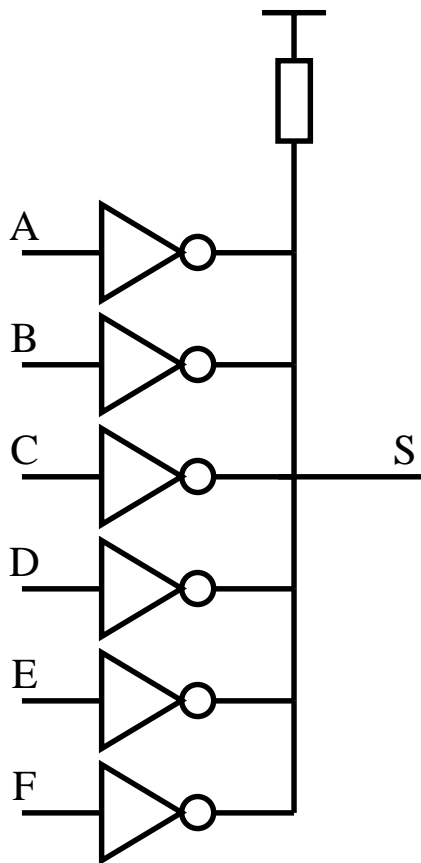


- plus de sortie "totem"
- R_C est appelée "résistance de rappel" (pull up)
- R_C détermine la consommation et la vitesse du circuit

Circuits TTL à collecteur ouvert (3)

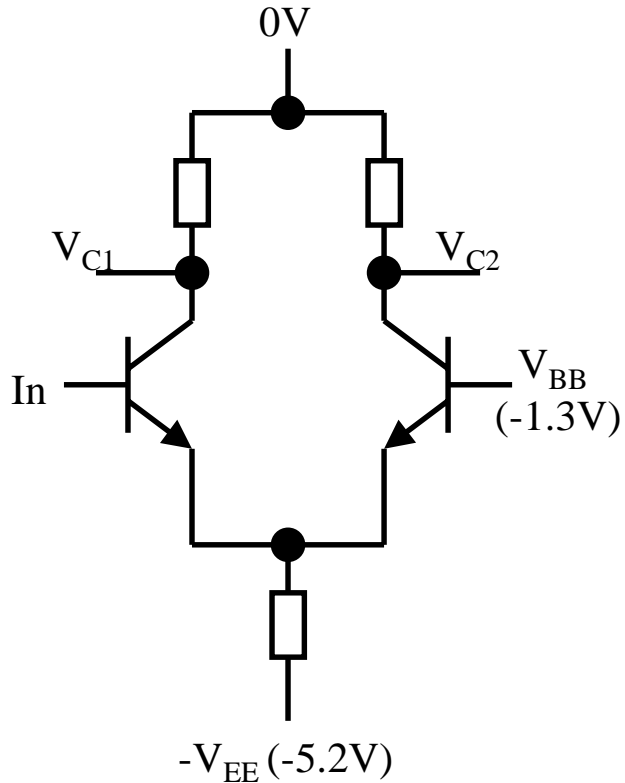
- **Précaution** : attention à ne pas dépasser I_{ol} !!
- Applications :
 - **simplification de circuit** dans le cas où vitesse et consommation ne sont pas une contrainte
 - **circuit tampon (74-05)**: fort courant I_{ol} ,
 - => permet de piloter une Led par exemple
 - **translation de niveaux logiques**
 - => permet de transformer un signal binaire 0-5V en un signal 0-3.3V par exemple.

Circuits TTL à collecteur ouvert (4)



- **Exercice** : soit le circuit à collecteur ouvert 7405 (6 inverseurs) monté comme suit :
 - Déterminer l'expression logique de S
 - Calculer la résistance de rappel du montage sachant que ce circuit à comme charge 5 portes TTL
 - Rappel :
 - $V_{oh\ min} = 2.4V$; $V_{ol\ max} = 0.4V$
 - $V_{il\ max} = 0.8V$; $V_{ih\ min} = 2V$
 - $I_{ih} = 40\mu A$; $I_{il} = 1.6mA$
 - $I_{oh} = 0.4mA$; $I_{ol} = 16mA$

Circuit ECL



- "Logique à couplage par l'émetteur"
- circuit rapide (600Mhz, 40mW, 1ns)
- pas de saturation des transistors
- "0" => -1.7V; "1" => -0.8V
- marge de bruit pire cas de 250mV
- Attention : les niveaux de sortie sont différents des niveaux d'entrées !!
- V_{C2} est le complément de V_{C1} (= /in)
=> porte OU et NOR avec un seul circuit!
- Pas de bruit de commutation
- Technologie incompatible avec les autres

Familles logiques MOS

- Technologie la plus utilisée
- Tous les circuits numériques MSI et VLSI sont en technologie MOS
- Trois familles :
 - PMOS
 - NMOS
 - CMOS (la plus utilisée)

Circuits CMOS (1)

- Dits "complémentaires" : constitués d'une partie NMOS et d'une partie PMOS.
- plus rapides et moins gourmands que les autres familles MOS
- surface d'implantation très très inférieure à une technologie TTL.
- Les MOS fonctionnent en saturé/bloqué
 - => considérés comme des interrupteurs parfaits
 - NMOS : "1" => passant (saturé), "0" => ouvert (bloqué)
 - PMOS : "1" => ouvert (bloqué), "0" => passant (saturé)

Circuits CMOS (2)

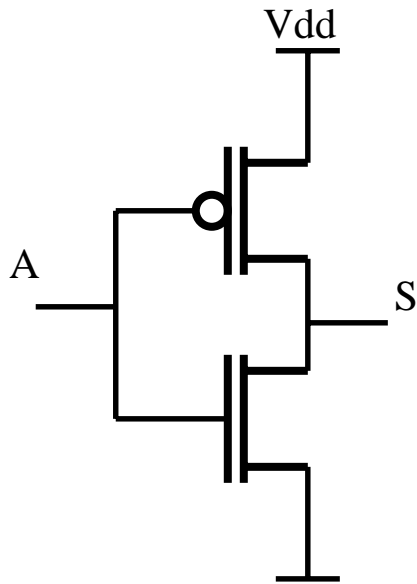
- Série 4000/14000 : la plus ancienne (50ns, 12Mhz, 0.1mW à 100Khz)
- 74C-xx : perf. Quasi Identiques à la série 4000 mais compatibilité broche à broche avec les circuits TTL
- 74HC-xx : plus rapide (8ns, 40Mhz, 0.17mW)
- **74HCT-xx** : totalement compatible avec les circuits TTL (échange standard)

Circuits CMOS (3)

- Caractéristiques série 4000 :
 - alimentation : 3-15V (2-6V pour HC et HCT)
 - $V_{oh\ min} = V_{dd}$; $V_{ol\ max} = 0V$
 - $V_{il\ max} = 30\%$ de V_{dd} ; $V_{ih\ min} = 70\%$ de V_{dd}
 - marge de bruit : 1.5V sous 5V
 - consommation : quasi nulle en statique
 - => consomme lors des commutations !
 - Consommation augmente avec la fréquence d'utilisation
 - Fanout : 50 si $f < 1Mhz$
- Précaution :
 - **pas d'entrée non connectée** car risque de détérioration par décharge électrostatiques

Circuits CMOS (4)

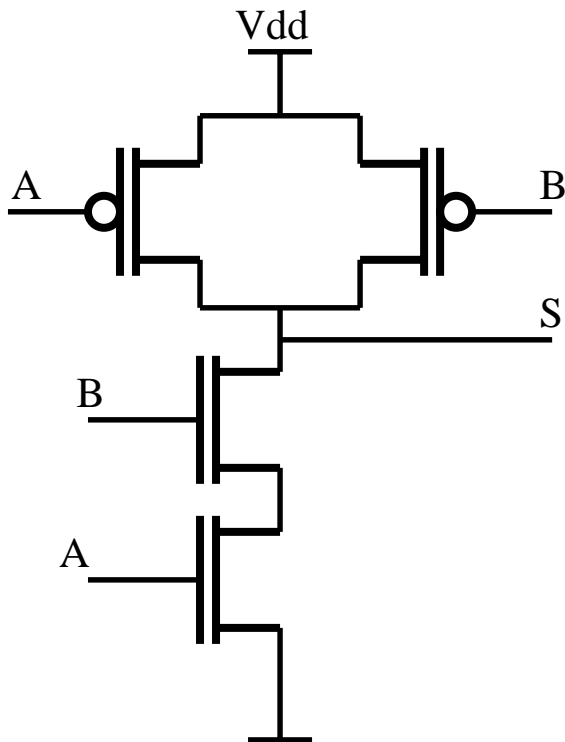
- Exemple d'implantation : la porte inverseuse



- Si $A = 0$, le PMOS est passant et le NMOS bloqué
 $\Rightarrow S = Vdd$
- Si $A = 1$, le PMOS est bloqué et le NMOS passant
 $\Rightarrow S = 0$

Circuits CMOS (5)

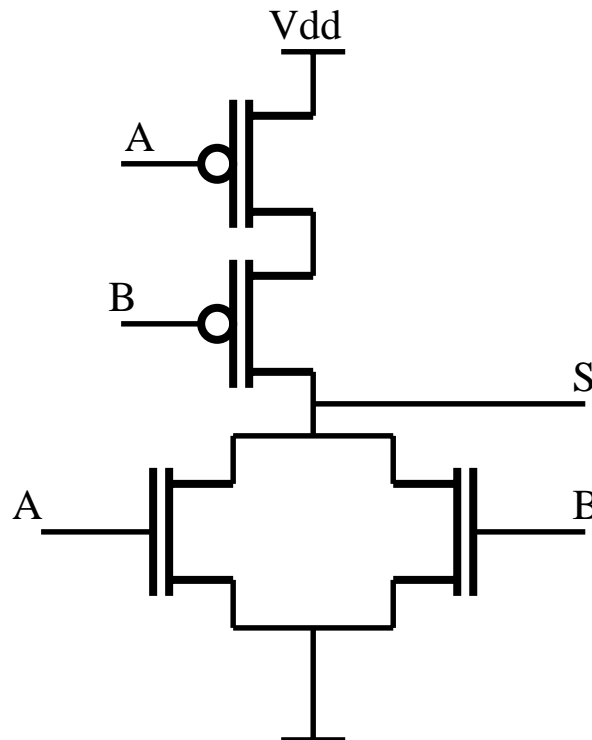
- Exemple d'implantation : la porte Nand
 - L'implantation PMOS est la duale des NMOS



- Si $A = B = "1"$, les NMOS sont passants, les PMOS bloqués,
 $\Rightarrow S = "0"$
- Si une des entrées est à "0", la branche NMOS est ouverte et un des PMOS est passant,
 $\Rightarrow S = "1"$

Circuits CMOS (6)

- Exercice :
 - Quelle est la fonctionnalité du circuit ci-dessous??
 - Explications

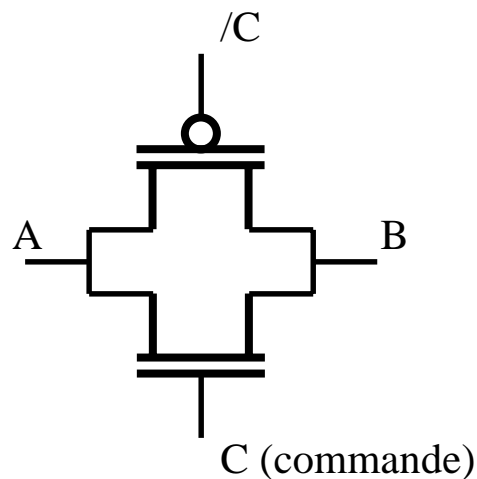


Autres familles MOS

- **Circuit NMOS :**
 - constitué uniquement de NMOS
 - la partie PMOS est remplacée par un NMOS connecté à Vdd donc toujours passant (résistance)
 - plus lent et plus consommant que les CMOS
 - plus simple et moins gourmand en surface
- **Circuit PMOS :**
 - idem NMOS
 - encore plus lent que les NMOS (mobilité des porteurs)
- **Circuit à drain ouvert**
 - mêmes intérêts que les circuits TTL à collecteur ouvert

Interrupteur bilatéral

- Ou "porte de transmission"
 - exclusivité CMOS
 - interrupteur imparfait ($R_{on} = 200\text{ohms}$, $R_{off} = 10^{12}\text{ohms}$)
 - Signal numérique ou analogique (limité à $0\text{V} - V_{dd}$)
 - circuit 4016 ou 74HC4016



Interfaçage CMOS-TTL (1)

Récapitulatif des caractéristiques CMOS et TTL

Interfaçage CMOS-TTL (2)

- sorties CMOS - entrées TTL
 - à l'état haut :
 - $V_{OH(CMOS)} = 4.9V > V_{IH(TTL)} = 2V \Rightarrow \text{correct}$
 - $I_{IH(TTL)} \ll I_{OH(CMOS)} \Rightarrow \text{correct}$
 - à l'état bas :
 - $V_{OL(CMOS)} = 0.1V < V_{IL(TTL)} = 0.8V \Rightarrow \text{correct}$
 - $I_{IL(TTL)}$ varie de $100\mu A$ à $2mA$
 - $I_{OH(CMOS)}$ varie de $0.4mA$ à $4mA \Rightarrow \text{correct sauf pour la CMOS 4000 } (I_{OL}=0.4mA)$ qui ne peut piloté que la TTL 74ALS ($I_{IL} = 100\mu A$)

\Rightarrow ATTENTION AU FANOUT !!

Interfaçage CMOS-TTL (3)

- sorties TTL - entrées CMOS

- **TTL vers CMOS 74HCT : pas de problème**

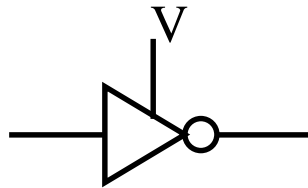
- problème avec les CMOS 4000 et 74HC :

$$V_{IH(\text{CMOS})} > V_{OH(\text{TTL})} \Rightarrow \text{ne marche pas !!}$$

- Solution : résistance de rappel à 5V
- Si le circuit CMOS est alimenté par une tension $> 5V$, il reste la possibilité d'utiliser un circuit tampon (collecteur ouvert 7407)

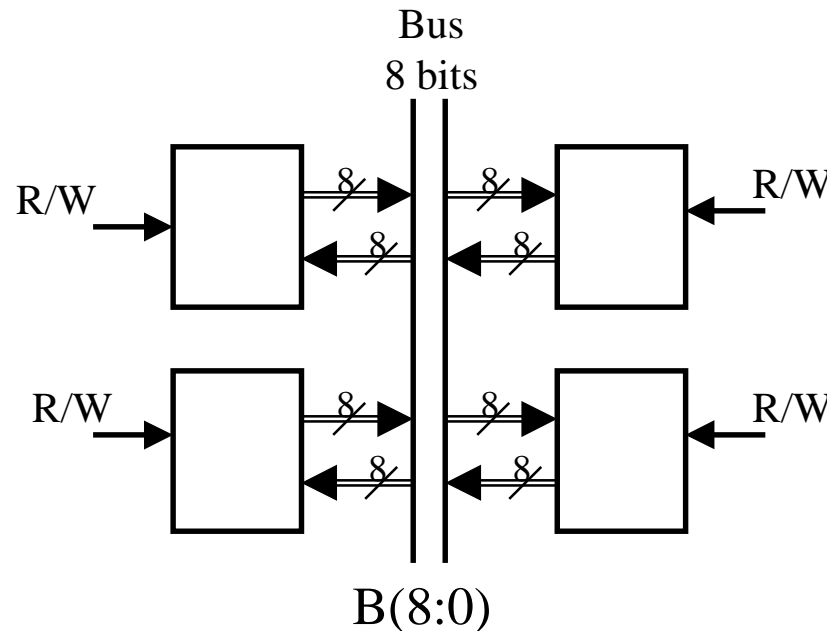
Sortie haute impédance

- Problème en TTL et CMOS : on ne peut pas connecter des sorties ensembles.
- Solution : un troisième état de la sortie :
=> état haute impédance
- La sortie est équivalente à un circuit ouvert (ou flottante)
- Cet état est activé par une nouvelle entrée sur le circuit.
- Une seule sortie parmi les n connectées ensembles ne doit pas être à l'état HI à un instant donné.



Bus de données

- Application principale des circuits 3 états !!
 - Constitué d'un certain nombre de fils (4, 8, ...64) permettant le transfert de mots binaires en parallèle
 - Un seul élément est autorisé à "écrire" sur le bus et cette donnée peut être envoyée sur plusieurs autres blocs logiques.

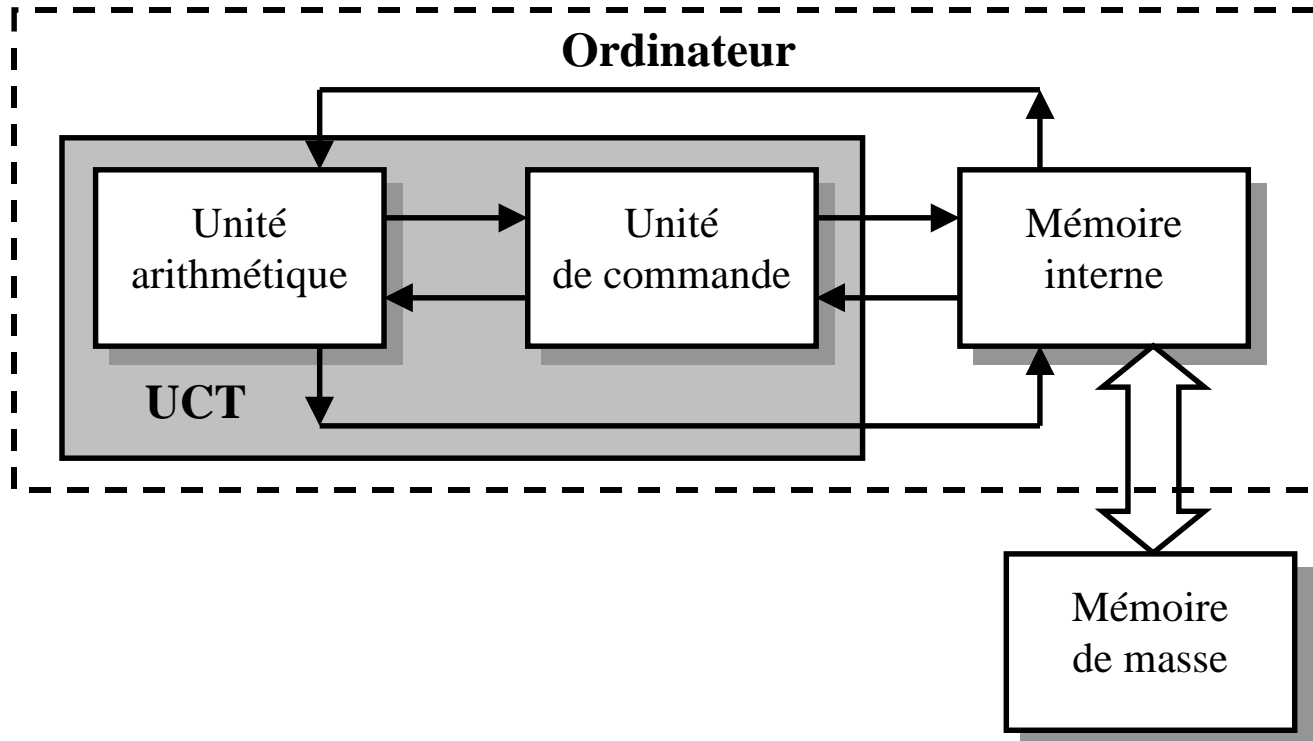


Chapitre 6

Circuits mémoires - Circuits programmables

Circuits mémoires : Généralités

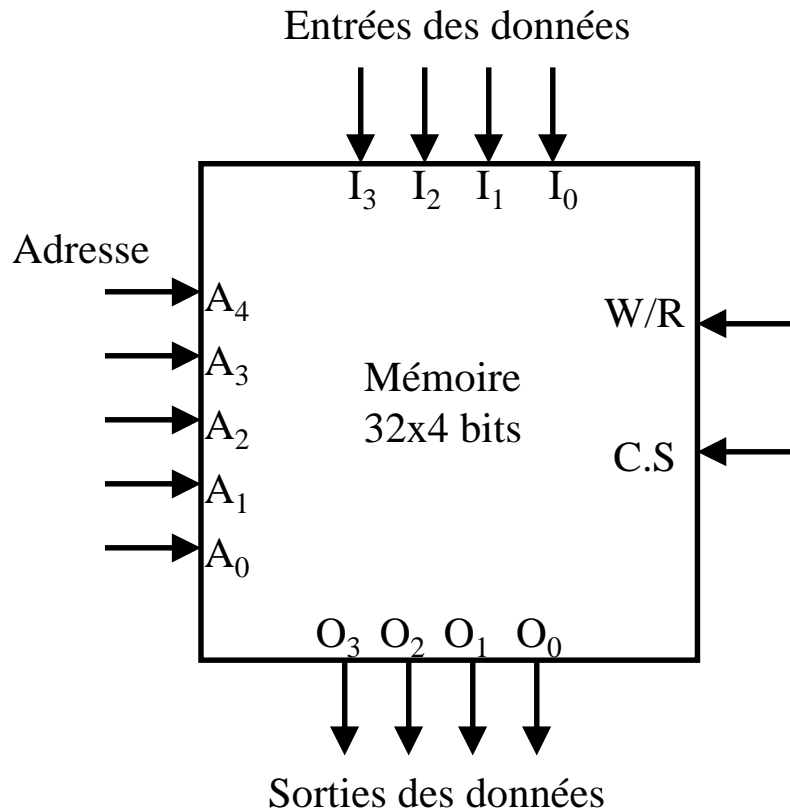
- Capacité de mémorisation => polyvalence d'un système numérique
- plusieurs types de mémoires :
 - Les registres : très rapides, pour opérations internes d'une UCT
 - Mémoires de masse : lentes, grande capacité de stockage,
 - Mémoires à semi conducteur : rapides, "faible" capacité de stockage,



Circuits mémoires : Terminologie

- **Cellule mémoire** : élément qui stocke 1 bit (bascule, condensateur)
- **Mot mémoire** : groupe de cellules mémoires qui représente un mot binaire stocké
- **Capacité** : quantité de bits stockés (en "Kilo bits" => 1024 bits) ou quantité de mots de 8 bits stockés (en "Méga octets" => 2^{20} octets)
- **Adresse** : nombre binaire qui représente la localisation physique d'un mot en mémoire
- **Lecture (écriture)** : Opération au cours de laquelle un mot binaire est extrait de (stocké dans) la mémoire à une adresse donnée.
- **Temps d'accès** (T_{acc}): durée nécessaire à une opération de lecture

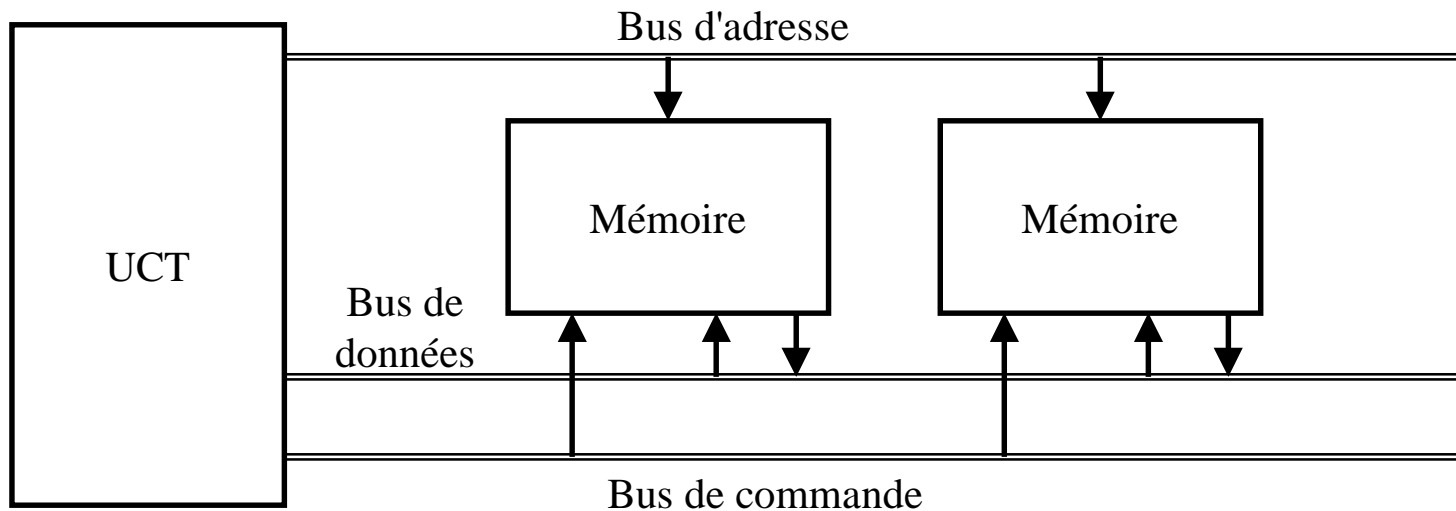
Circuits mémoires : Fonctionnement



- Etablir une adresse
- Choisir entre le mode lecture ou écriture
- Valider la mémoire pour prendre en compte l'adresse et effectuer l'opération de W/R
- Prise en compte des valeurs présentes sur le bus d'entrée ou de sortie.
- Dans tout les cas : se référer à la documentation de la mémoire !

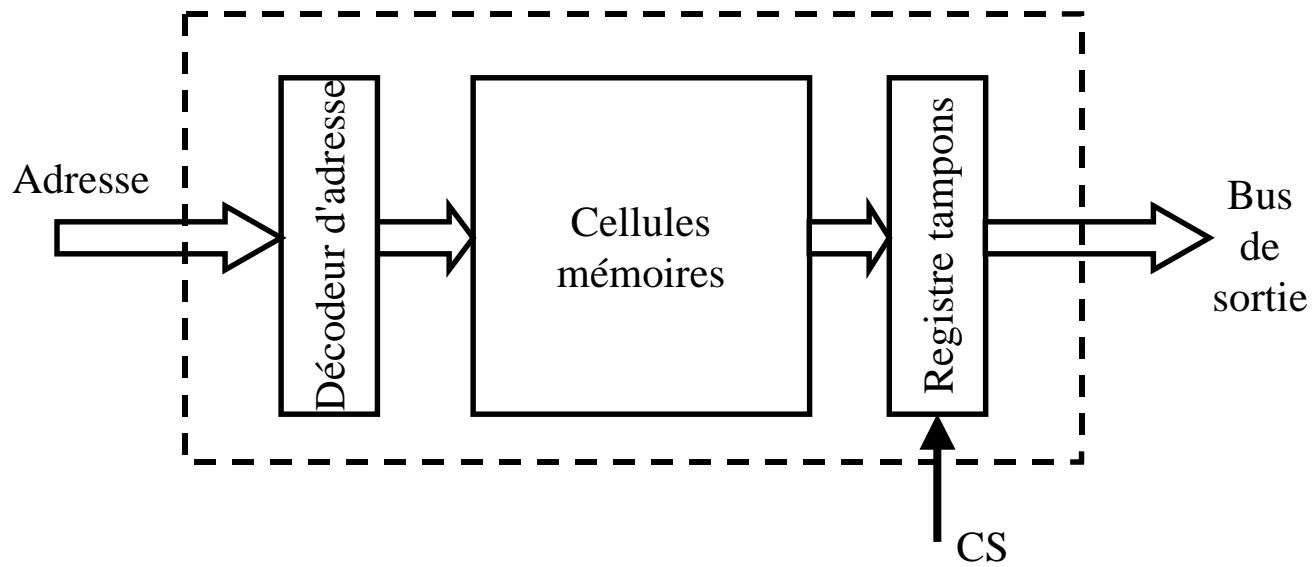
Connexions UCT-Mémoire

- 3 bus de données:
 - bus de commande (unidirectionnel)
 - bus de données (bidirectionnel)
 - bus d'adresse (unidirectionnel)



Mémoire morte (ROM)

- Mémoire **non volatile**
- Données écrites **une seule fois**.
- Opération de lecture seulement (Read Only Memory)



Mémoire morte (2)

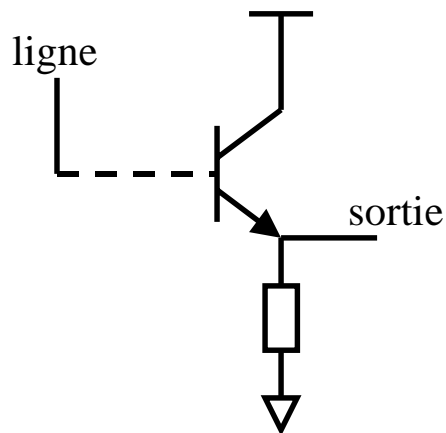
- Synchronisation d'une ROM :
 - 2 contraintes temporelles à respecter:
 - **temps d'accès** (t_{acc}): temps minimum entre le moment où une adresse est envoyée à la ROM et le moment où la donnée correspondante est effectivement sur le bus de sortie
 - **temps de validation de sortie** (t_{oe}): temps minimum entre l'instant de validation du registre tampon de sortie et le moment où la donnée est effectivement sur le bus de sortie (cad en sortie du registre tampon)

Mémoire morte (3)

- Utilisations :
 - mémoire d'amorçage (BIOS) : démarrage d'un ordinateur
 - microprogrammation : systèmes d'exploitation (DOS), interpréteurs de langage, jeux électroniques, cartes à injection pour automobile ...
 - tables de données (tables trigonométriques par exemple)
 - convertisseur de données
 - 74185 : convertisseur 6 bits binaire-DCB
 - générateurs de fonction

Types de mémoire ROM (1)

- Mémoire morte programmée par masque (**MROM**)
 - programmée à la fabrication (masque)
 - programmation effectuée par le concepteur de la ROM suivant des spécifications précises
 - économiques que sur de très gros volume de production.
 - L'élément mémoire est un transistor bipolaire NPN à collecteur ouvert



- => si la base est connectée : "1" mémorisé
- => si la base n'est pas connectée : "0"
- => si la ligne n'est pas validée, il n'y a pas de consommation dans cette ligne

Types de mémoire ROM (2)

- Mémoire morte programmable (**PROM**)
 - programmation par l'utilisateur
 - Structure à **liaison fusible** (grillé => 0)
 - une seule programmation possible
 - une PROM bipolaire classique : 74186
 - 64 mots de 8 bits
 - temps d'accès : 50ns
 - Les PROM à transistor MOS offrent beaucoup plus de mémoire

Types de mémoire ROM (3)

- Mémoire morte effaçable programmable (**EPROM**)
 - Le point mémoire est un transistor à effet de champs
 - programmation par utilisation de tensions spécifiques (10-25V)
 - => accumulation de charge sur la grille du FET qui sature le transistor
 - Effacement par exposition sous rayon ultra-violet.
 - => photocourant qui permet l'évacuation des charges de grille
 - => 15 à 30 min d'exposition
 - => obligation d'enlever la mémoire de la carte pour l'effacer et la reprogrammer
 - Large gamme disponible sur le marché : de 4Ko (EEPROM 2732) jusqu'à 128Ko

Types de mémoire ROM (4)

- Mémoire morte effaçable et programmable électriquement (**EEPROM**)
 - procédé de programmation identique à l'EPR0M
 - procédé du même genre pour l'effacement
 - effacement en 10ms
 - 5 fois plus rapide à programmer que l'EPR0M
 - possibilité de programmer et d'effacer sur le circuit imprimé
 - exemple : EEPROM 2864 (8Ko)

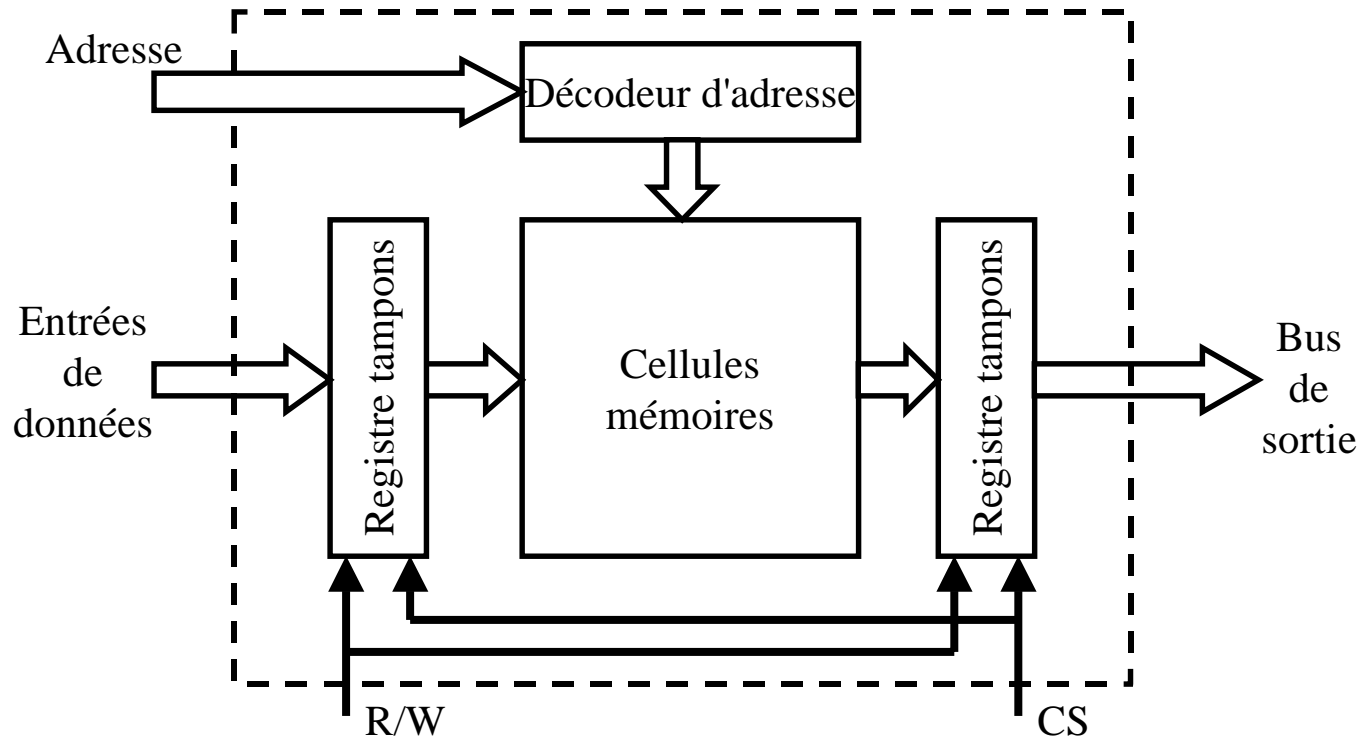
Mémoire ROM (5)

<u>DCB</u> => <u>2 parmi 5</u>	
ABCD	XYZTU
0000	10100
0001	00011
0010	00101
0011	00110
0100	01001
0101	01010
0110	01100
0111	11000
1000	10001
1001	10010

- **Exercice** : Transcodeur programmé
Utiliser une mémoire PROM pour réaliser un convertisseur de code DCB (0-9) en code "2 parmi 5".
Quelle capacité minimum doit avoir la PROM ?
Prévoir un bit de sortie "**CV**" (Code Valide, a mettre en MSB), Pourquoi ?
Donner la liste des valeurs à mémoriser en **octal**

Mémoires vives (RAM)

- Possibilités de lire et d'écrire rapidement (Random Access Memory) et à volonté
- Données perdues à la coupure de l'alimentation : mémoire volatile



Types de mémoire RAM (1)

- Mémoire vive statique (**SRAM**)
 - stockage par cellule type bascule
 - durée de stockage indéfinie (sans coupure d'alimentation)
 - bipolaire (rapide) ou MOS (capacité de stockage plus grande)
 - Synchronisation :
 - Attention au temps de maintien des données sur le bus de sortie une fois le circuit désactivé (voir doc)
 - Exemple : CMOS 6264 :
 - 8Ko,
 - cycle R/W de 100ns,
 - consommation en veille de 0.1mW

Types de mémoire RAM (2)

- Mémoire vive dynamique (**DRAM**)
 - Donnée stockée dans un condensateur
 - contraintes :
 - "rafraîchissement" obligatoire de la donnée (2 à 10ms) pour ne pas perdre l'information
 - moins rapide
 - Avantages :
 - très faible encombrement (facteur 4)
=> grande capacité de stockage
 - très faible consommation (moitié)

Types de mémoire RAM (3)

- **Rafraîchissement des DRAM**
 - principe : si une ligne est activée pour une opération, toute la ligne est rafraîchie (on reboucle la sortie de l'élément mémoire sur la capacité)
 - Un circuit "contrôleur de mémoire dynamique" active périodiquement (compteur) chaque ligne pour rafraîchir les données en tenant compte des accès mémoires.
- **Utilisation :**
 - partout où on a besoin de grande capacité de stockage (ordinateur)
 - les SRAM seront utilisées quand la vitesse de travail primera sur l'encombrement et la consommation

Types de mémoire RAM (4)

- Mémoire vive rémanente (**NVRAM**)
 - incluent une SRAM et une EEPROM de mêmes tailles
 - Si une coupure de courant est détectée, la SRAM sauvegarde ses données dans l'EEPROM !!
 - Inconvénient : grande complexité et donc faible capacité
 - Avantage de la SRAM avec possibilité de sauvegarde des données sans avoir recours à une pile de secours

Types de mémoire RAM (5)

- Mémoires séquentielles :
 - l'ordre avec lequel on lit les données est directement liés à celui dans lequel elles ont été enregistrées
 - il en existe 2 types :
 - les "piles" (accès **LIFO**, Last In First Out)
 - les "files" (accès **FIFO**, First In First Out)
 - Conçues à partir de registres à décalage ou mémoire classique (et un registre à décalage pour l'adressage)
 - Utilisation :
 - Dès qu'il faut des données séquentielles et répétitives (ex : régénération d'un écran vidéo).
 - Tampon entre un système rapide et un lent

Circuits programmables

- Circuits VLSI composés d'éléments logiques interconnectables par programmation de manière à implanter l'équation booléenne ou la fonction logique souhaitée.
- 2 familles :
 - les tableaux de portes logiques (PAL, PLA)
 - les tableaux de cellules logiques (EPLD, LCA)
- 2 types :
 - Programmable 1 seule fois (fusible)
 - Reconfigurable

les tableaux de portes logiques

- Composé d'une matrice de porte ET connectée à une matrice de porte OU :
 - => équations booléenne sous forme de somme de produit
- 2 types :
 - PAL : Seule la matrice "ET" est programmable (limitation de la fonction booléenne à x sommes de produit)
 - PLA : les 2 matrices sont programmables
- Programmation en général unique (fusible)
- Une PROM peut être utilisée si on utilise toutes les combinaisons possibles des "entrées" (adresses) car seule l'équivalent de la matrice "OU" est programmable

les tableaux de portes logiques

- **Exercice** : Trouver les équations booléennes câblées dans cette PAL
- Nota : à un croisement de 2 fils
 - X fusible intact (connexion)
 - ° connexion permanente
 - par défaut la matrice est est remplie de X

les tableaux de cellules logiques

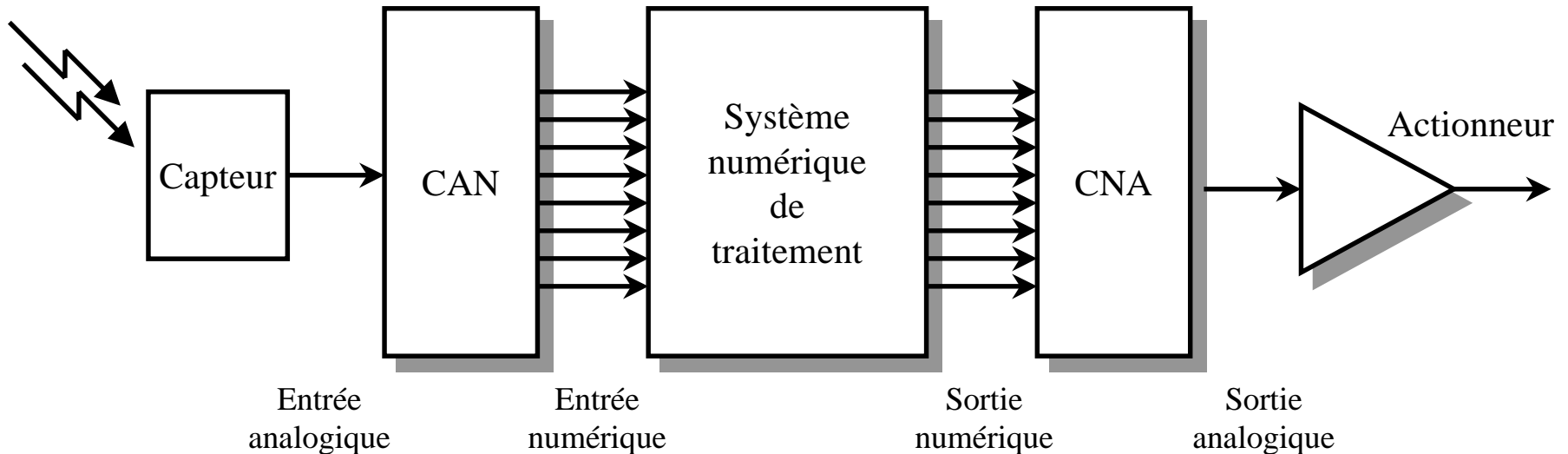
- Apparus récemment du fait de leur complexité nécessitant une forte intégration :
 - grand nombre de broches d'entrées/sorties
 - réalisation de fonctions logiques complexes grâce à une architecture cellulaire puissante et riche en interconnexion
 - les tableaux de cellules sont reprogrammables en général
 - bibliothèque de fonctions à disposition
- Utilisation d'un logiciel spécifique pour la programmation
- 2 grandes marques : Altera (EPLD) et Xilinx (LCA)

Chapitre 7

Conversion Numérique-Analogique Analogique-Numérique

Convertisseur N/A, A/N

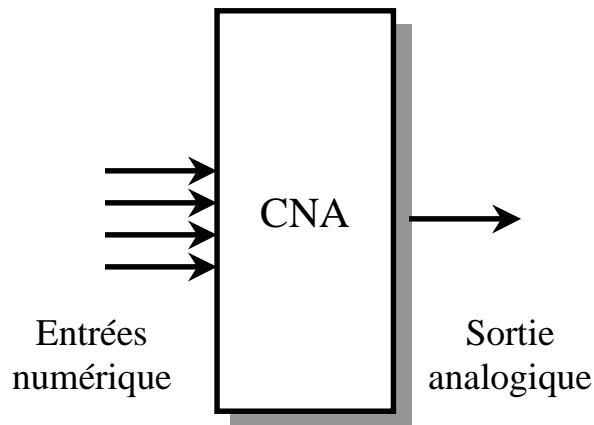
- interface entre le monde réel (analogique) et les systèmes de traitement (à 99% numérique)
- exemple : Système de régulation



Conversion Numérique-Analogique

- Mot binaire d'entrée en binaire pur ou DCB
- La tension (ou courant) de sortie est proportionnelle à la valeur numérique :

$$\Rightarrow \text{Sortie "analogique"} = K \times (\text{entrée numérique})$$



D	C	B	A	V _{sortie} (V)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

CNA : Spécifications (1)

- **Résolution** : (ou pas de progression) :
 - Plus petite variation en entrée qui entraîne une modification de la sortie
 - Directement liée au nombre de bits
 - => 4 bits impliquent 16 valeurs analogiques en sortie
 - **Exprimée en nombre de bits de l'entrée**
 - en % de la pleine échelle : rapport entre le pas de progression et la pleine échelle :
 - => exemple précédent : $1V/15V = 6.67\%$ de résolution
 - plus le nombre de bits est grand et plus la résolution est fine

CNA : Spécifications (2)

- **Précision :**
 - Exprimée selon 2 paramètres :
 - **Erreur pleine échelle** (écart maximal en %PE entre la sortie du CNA et sa valeur idéale)
 - **Erreur de linéarité** (écart maximal en % entre le pas de progression réel et sa valeur théorique)
 - 0.01 à 0.1 % pour des CNA universels
- Attention : résolution et précision doivent être compatibles

CNA : Spécifications (3)

- **Temps d'établissement :**

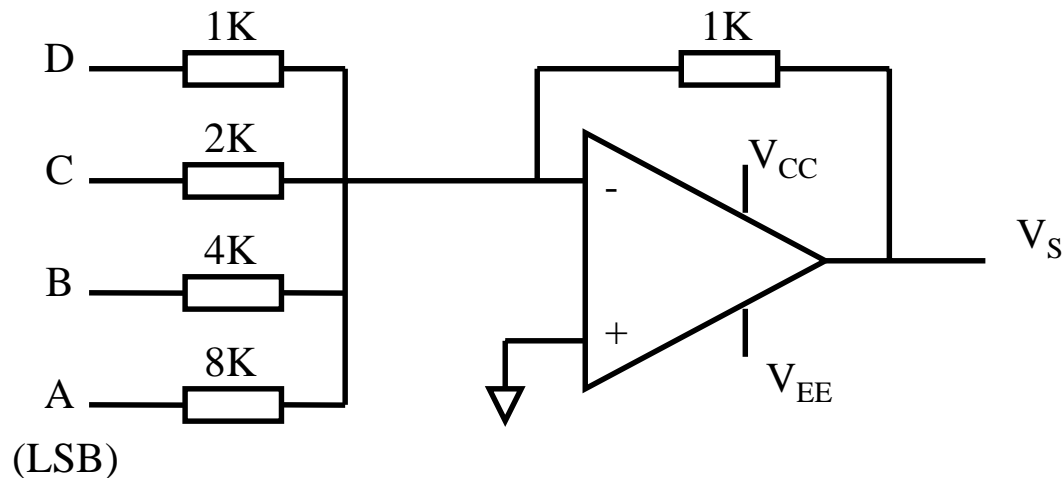
- Temps nécessaire à la sortie pour passer de la valeur zéro à la valeur pleine échelle (50ns à 10 μ s)
- Définit à 1/2 pas de progression de la valeur finale
ex : résolution de 10mV, le T_e est la mesure du temps qui s'écoule jusqu'à ce que la sortie se stabilise à 5mV près de sa valeur pleine échelle

- **Tension de décalage :**

- tension d'offset de sortie
- S'ajoute à toutes les valeurs que fournit le CNA
- possibilité de compensation de cette erreur de décalage par un potentiomètre de compensation d'offset.

Types de Convertisseur N-A (1)

- Amplificateur sommateur :



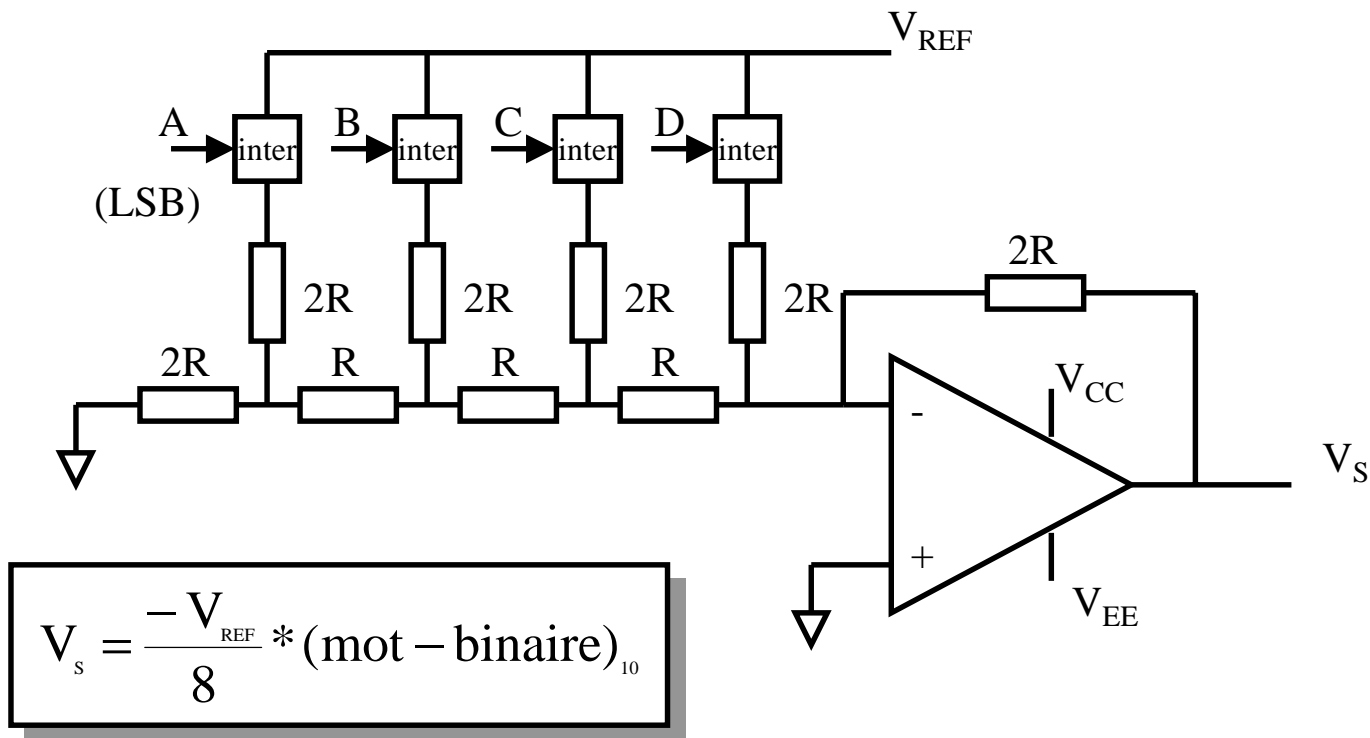
$$V_S = -(V_D + 0.5V_C + 0.25 V_B + 0.125V_A)$$

Types de Convertisseur N-A (2)

- Amplificateur sommateur : (suite)
 - Les entrées binaires commandent des interrupteurs qui assurent des tensions d'entrées (0 ou 1) du CNA précises
 - Les valeurs des résistances permettent de donner à chaque entrées binaires leurs poids respectifs
 - => Grande précision sur les résistances obligatoire sinon la précision se dégrade rapidement
 - problème d'intégration : difficulté de faire de grande valeur de résistance précisément
 - possibilité de sortir en courant

Types de Convertisseur N-A (3)

- Réseau R/2R :



Types de Convertisseur N-A (4)

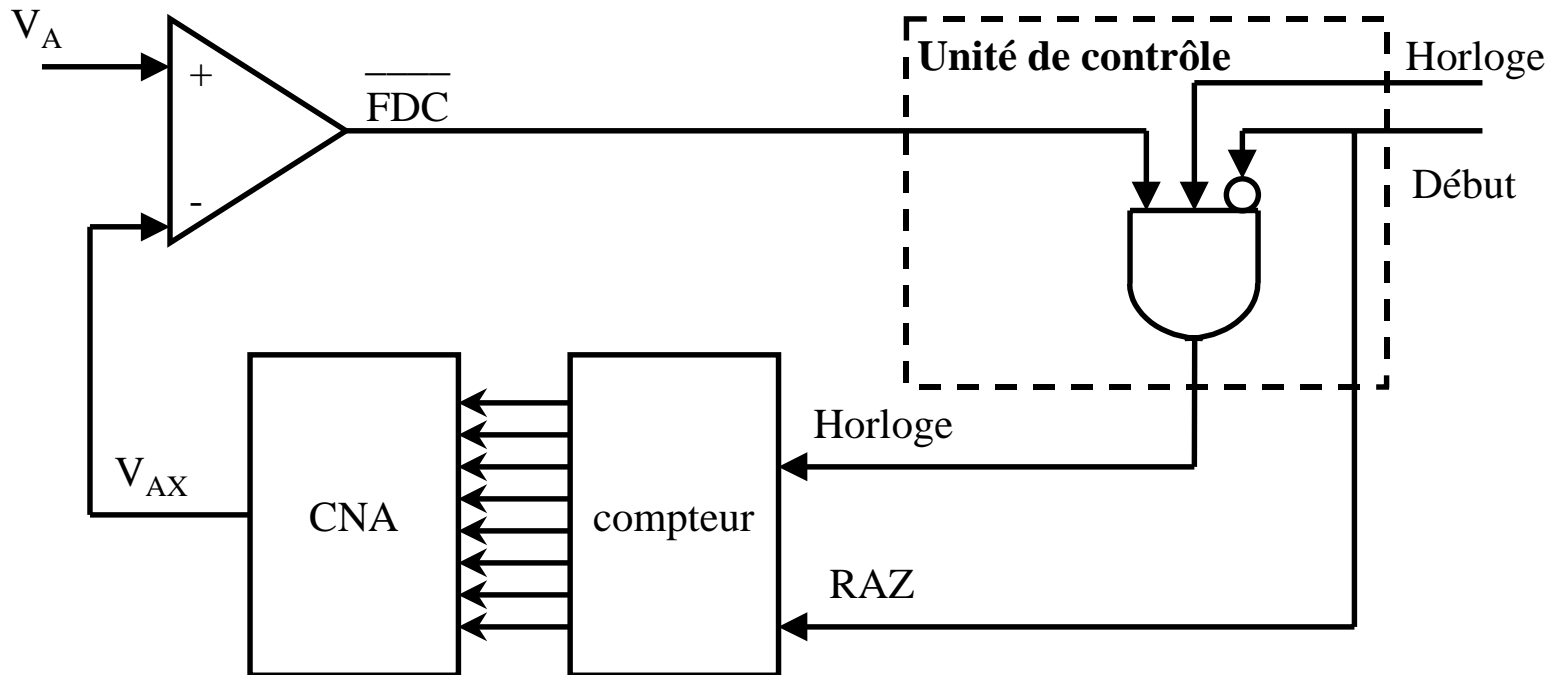
- Réseau R/2R : (suite)
 - Résistances de valeurs voisines, donc intégration plus facile
- **Exercice :**
 - Si $V_{ref} = 5 \text{ V}$, quelle est la résolution et quelle est la tension pleine échelle du convertisseur R/2R ?

Conversion Analogique - Numérique

- Entrée analogique, sortie numérique codée binaire pur
- Conversion plus longue et plus complexe qu'un CNA
- Contient en général un CNA ...
- Conversion en général cadencée par un signal d'horloge
- Une unité de contrôle gère la conversion, la génération de signaux (fin de conversion par exemple), la mise en mémoire de la valeur convertie (registre)...

Types de Convertisseur A-N (1)

- CAN à rampe
 - on met à zéro le compteur et on l'incrémente jusqu'à ce que la valeur de sortie de celui-ci soit égale à la valeur analogique d'entrée ($V_{AX} > V_A$).



Types de Convertisseur A-N (2)

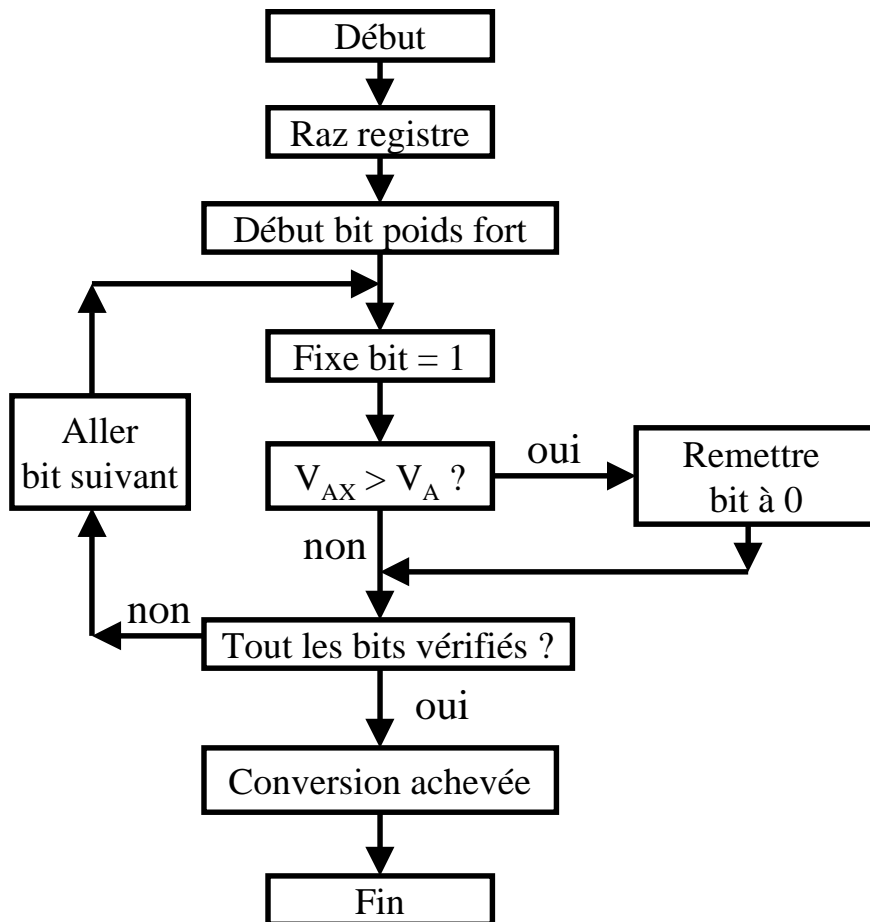
- CAN à rampe (suite)
 - **Résolution** : égale à la résolution du CNA utilisé !
 - => Représente la plus petite variation que doit subir l'entrée analogique pour modifier le mot binaire en sortie : on parlera également de **Quantum**
 - Précision : du même genre que pour le CNA
 - **Temps de conversion** (T_c) : correspond à la conversion de la valeur maximum convertible : Si le convertisseur contient N bits, alors $T_{C(\max)} = 2^N - 1$ cycles d'horloge
 - **Inconvénient** : conversion lente (elle double à chaque fois que l'on rajoute un bit)
 - **Utilisation** : appareil de mesure (voltmètre)

Types de Convertisseur A-N (3)

- **Exercice :**
 - Soit les valeurs suivantes pour le CNA à rampe : Horloge = 1Mhz; $V_t = 0.1\text{mV}$; Tension PE du CNA = 10.23V; sortie sur 10 bits
 - Déterminer :
 - le pas de progression
 - la résolution de ce convertisseur (en %PE)
 - l'équivalent binaire de $V_A=3.728\text{V}$
 - la durée de conversion de cet échantillon

Types de Convertisseur A-N (4)

- CAN à approximations successives :



- plus complexe au niveau du système de contrôle
- on remplace le compteur par un registre dont on contrôle chaque bit
- temps de conversion plus court et fixe (N coups d'horloge)
- Exemple de circuit : ADC0804
 - sorties 8 bits, 3 états
 - 2 entrées (différentielle)
 - possibilité d'horloge interne
 - 2 masses

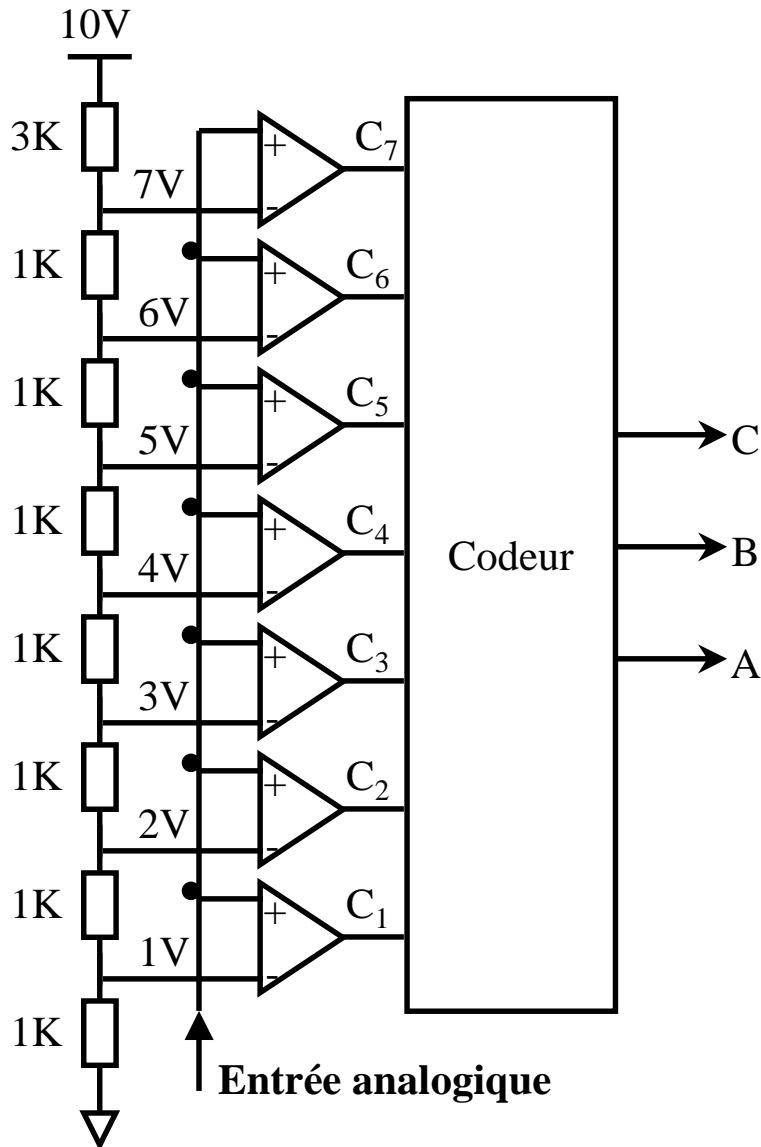
Types de Convertisseur A-N (5)

- **Exercice :**
 - CAN 4 bits, pas de progression de 1V
 - Quelle est sa tension pleine échelle ?
 - A partir de l'organigramme précédent, décrire le fonctionnement du convertisseur pour convertir l'échantillon $V_A = 10.4V$
 - Dessiner l'évolution du signal V_{AX}

Types de Convertisseur A-N (6)

- CAN parallèle (flash)
 - le plus rapide et le plus complexe en terme de circuit
 - Un CAN de N bits (2^N grandeurs numériques possibles) contiendra N-1 comparateurs !
 - Tous les comparateurs fonctionnent en parallèle
 - Pas de CNA, pas d'horloge ni d'unité de contrôle
 - On convertit ensuite la sortie binaire des N-1 comparateurs (code température) en code binaire pur
 - En général limité à 8 bits (AD9002 par exemple)

Types de Convertisseur A-N (7)



- CAN parallèle (flash) (suite)

Exemple sur 3 bits :

VA	C7	C6	C5	C4	C3	C2	C1	CBA
$VA < 1V$	0	0	0	0	0	0	0	000
$1 < VA < 2V$	0	0	0	0	0	0	1	001
$2 < VA < 3V$	0	0	0	0	0	1	1	010
$3 < VA < 4V$	0	0	0	0	1	1	1	011
$4 < VA < 5V$	0	0	0	1	1	1	1	100
$5 < VA < 6V$	0	0	1	1	1	1	1	101
$6 < VA < 7V$	0	1	1	1	1	1	1	110
$VA > 7V$	1	1	1	1	1	1	1	111

Types de Convertisseur A-N (8)

- CAN de poursuite :
 - très proche du CAN à rampe mais le compteur est un compteur/décompteur
 - si l'entrée augmente, on incrémente le compteur et vice-versa
 - Beaucoup plus rapide en moyenne que le CAN à rampe
- CAN double rampe
 - le plus lent, le moins cher et le moins sensible au bruit et aux dispersion des composants => **mesure de précision**
 - pas de CNA (par rapport au CAN à rampe)
 - on charge un condensateur avec un courant constant proportionnel à l'entrée pendant un temps donné.
 - On décharge ce condensateur avec un courant de référence et en même temps on fait évoluer un compteur
 - Le comptage s'arrête quand le condensateur est déchargé