Circuits logiques de base

Adrian Daerr ⟨adrian.daerr@univ-paris-diderot.fr⟩

2018 / 2019
Électronique numérique L3 EIDD 1A

Récapitulatif du premier cours

- Électronique numérique : 2 niveaux (0/1) uniquement (robuste, rapide)
  - Représentation en général par la tension : Circuits TTL : 0 V / 5 V ; circuits CMOS : 0 V / 15 V
  - D'autres choix possibles : courant, fréquence, phase (p.ex. transmission acoustique, radio), ...
- Conception d'un circuit commence par :
  - Cahier des charges
  - Fonction logique (tableau de vérité)
  - Simplification de la fonction
    - $\exists$ solutions equivalentes en complexité
    - Choix peut dépendre de : coût, vitesse, encombrement, fiabilité, ...
  - Choix des composants et construction du circuit

Aujourd'hui : présentation de circuits logiques combinatoires de base, exemples de réalisations disponibles en circuit intégré

Circuits combinatoires vs. circuits séquentiels

Circuit combinatoire Fonction logique ne dépend que de l'état logique des entrées :

$$S = F(a, b, \dots)$$

Circuit séquentiel Fonction logique dépend aussi de sa valeur antérieure :

$$S_{n+1} = F(a, b, \dots, S_n)$$

Exemple : un ordinateur est un circuit séquentiel.

Plan du cours

Circuits combinatoires de base

- Les portes logiques (récapitulatif)
- Le demi-additionneur
- L'additionneur complet
- L'additionneur-soustracteur – l'ALU
- Le multiplexeur
- Le démultiplexeur
- Les décodeurs & transcodeurs

Un circuit simple non combinatoire

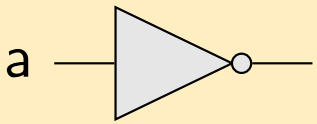
- Le multivibrateur

Circuits programmables : composants logiques universels

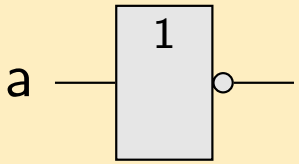
- Mémoires, registres, compteurs

Briques élémentaires : portes logiques

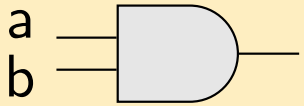
Opérateur NOT (non, complément, inverseur)



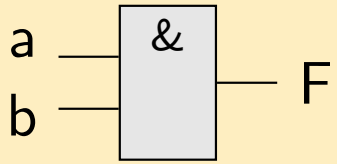
a	$F = \overline{a}$
0	1
1	0



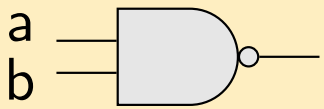
Opérateur AND (et, conjonction, intersection)



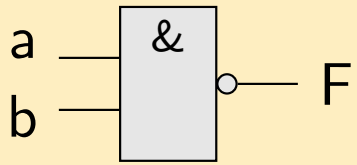
a	b	$F = a \cdot b$
0	0	0
0	1	0
1	0	0
1	1	1



Opérateur NAND (et-non)

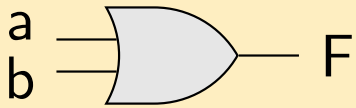


a	b	$F = \overline{a \cdot b}$
0	0	1
0	1	1
1	0	1
1	1	0

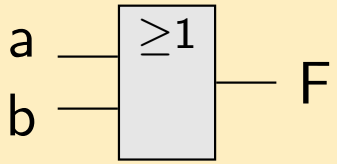


Briques élémentaires : portes logiques (suite)

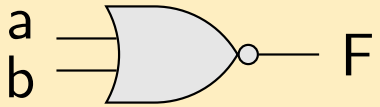
Opérateur OR (ou, disjonction, réunion)



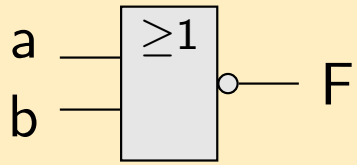
a	b	$F = a + b$
0	0	0
0	1	1
1	0	1
1	1	1



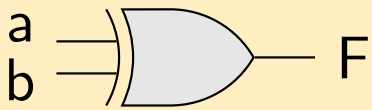
Opérateur NOR (ou-non)



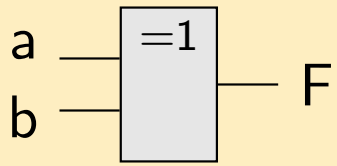
a	b	$F = \overline{a + b}$
0	0	1
0	1	0
1	0	0
1	1	0



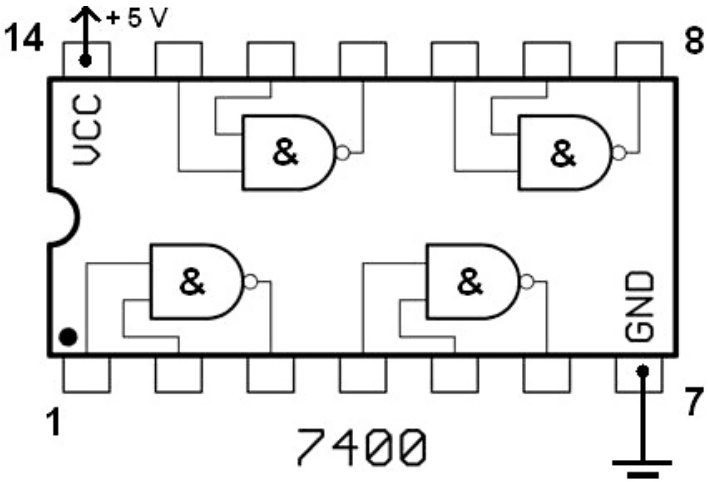
Opérateur XOR (ou exclusif)

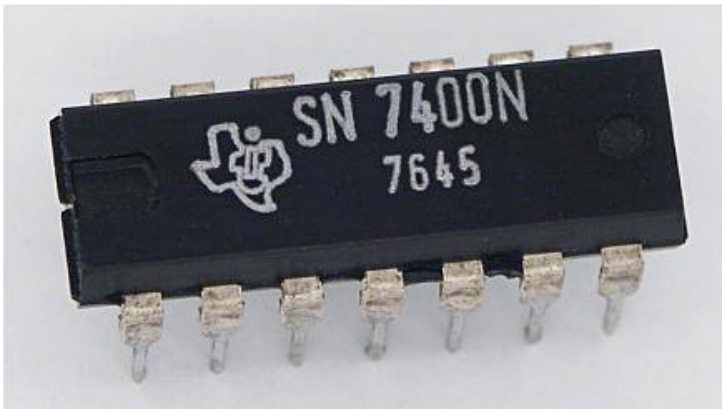


a	b	$F = a \cdot \overline{b} + \overline{a} \cdot b$
0	0	0
0	1	1
1	0	1
1	1	0

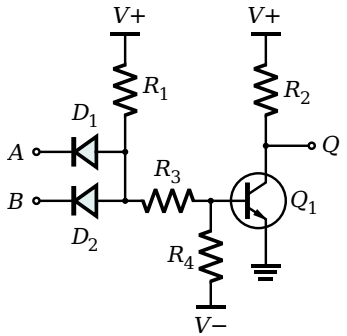


Exemples de circuits réalisant des portes logiques NAND

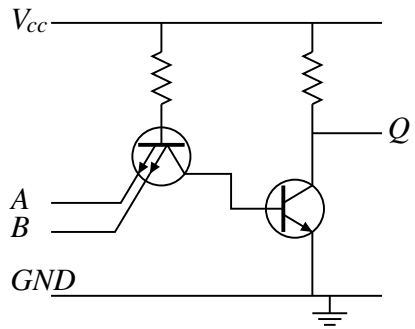




- Relativement facile à réaliser
  ⇝ la plus fabriquée



logique DTL (perimée)



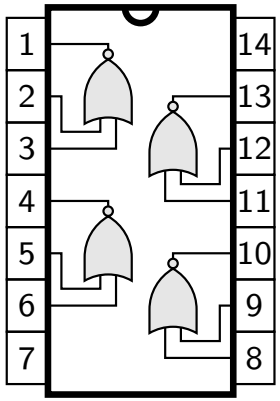
logique TTL

- Exemple de circuits intégrés :

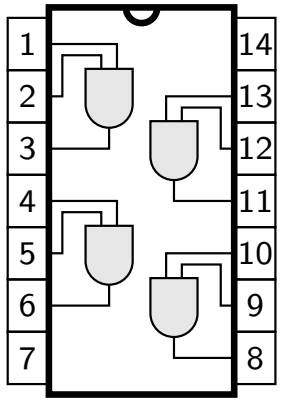
Fonction	TTL	CMOS
4 NAND à 2 entrées	74 00	4011
3 NAND à 3 entrées	74 10	4023
2 NAND à 4 entrées	74 20	4012
1 NAND à 8 entrées	74 30	4068

Exemples de circuits réalisant des portes logiques

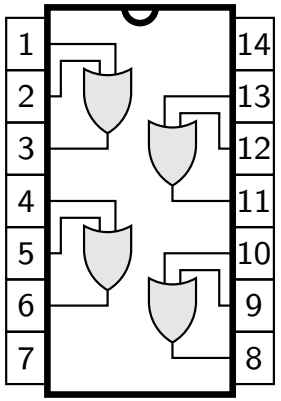
Autres portes :



4*NOR2 : 74 02

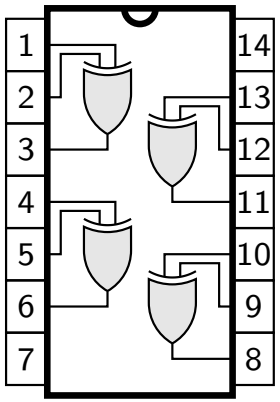


4*AND2 : 74 08

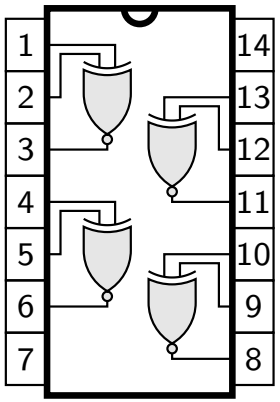


4*OR2 : 74 32

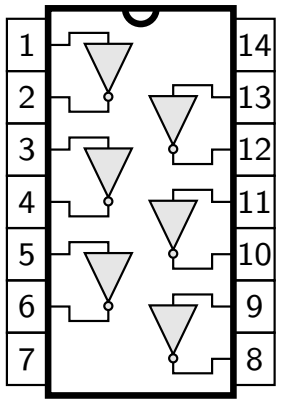
Numéro en 74 xx : circuit dans la famille TTL



4*XOR2 : 74 86



4*XNOR2 : 74 266



6*NOT : 74 04

. . .

Le demi-additionneur

a	b	$a +^\star b$	Σ	R
0	0	$0_{10} = 00_2$	0	0
0	1	$1_{10} = 01_2$	1	0
1	0	$1_{10} = 01_2$	1	0
1	1	$2_{10} = 10_2$	0	1

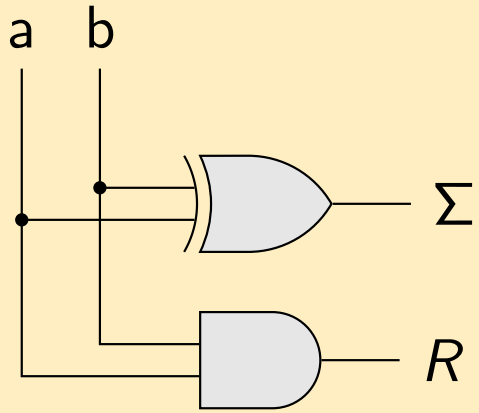
Σ . . . Somme (un seul chiffre !)
R . . . Retenue

En base 2 : retenue dès que $\Sigma > 1$ (!)

$\Sigma = \overline{a} \cdot b + a \cdot \overline{b}$
$R = a \cdot b$

* Attention : dans ce tableau $+$ est l'opération arithmétique «addition», pas l'opération Booléenne «ou».

Réalisation



L'additionneur complet

Tenir compte d'une évenutelle retenue r au chiffre précédent :

r	a	b	$a +^\star b +^\star r$	Σ	R
0	0	0	$0_{10} = 00_2$	0	0
0	0	1	$1_{10} = 01_2$	1	0
0	1	0	$1_{10} = 01_2$	1	0
0	1	1	$2_{10} = 10_2$	0	1
1	0	0	$1_{10} = 01_2$	1	0
1	0	1	$2_{10} = 10_2$	0	1
1	1	0	$2_{10} = 10_2$	0	1
1	1	1	$3_{10} = 11_2$	1	1

* Attention : dans cette colonne $+$ est l'opération arithmétique «addition», pas l'opération Booléenne «ou».

Σ ... Somme

$$\Sigma = \overline{r} \cdot (\overline{a} \cdot b + a \cdot \overline{b}) + r \cdot (\overline{a} \cdot \overline{b} + a \cdot b)$$

R ... Retenue

$$R = \overline{r} \cdot a \cdot b + r \cdot (a + b)$$

L'additionneur complet

Tenir compte d'une évenutelle retenue r au chiffre précédent :

r	a	b	minterme	Σ	R
0	0	0	$\overline{r}\,\overline{a}\,\overline{b}$	0	0
0	0	1	$\overline{r}\,\overline{a}\,b$	1	0
0	1	0	$\overline{r}\,a\,\overline{b}$	1	0
0	1	1	$\overline{r}\,a\,b$	0	1
1	0	0	$r\,\overline{a}\,\overline{b}$	1	0
1	0	1	$r\,\overline{a}\,b$	0	1
1	1	0	$r\,a\,\overline{b}$	0	1
1	1	1	$r\,a\,b$	1	1

Σ ... Somme

$\Sigma = \overline{r}\cdot(\overline{a}\cdot b + a\cdot\overline{b}) + r\cdot(\overline{a}\cdot\overline{b} + a\cdot b)$

R ... Retenue

$R = \overline{r}\cdot a\cdot b + r\cdot(a+b)$

Forme disjonctive canonique :

$\Sigma = \overline{r}\cdot\overline{a}\cdot b + \overline{r}\cdot a\cdot\overline{b} + r\cdot\overline{a}\cdot\overline{b} + r\cdot a\cdot b$

$R = \overline{r}\cdot a\cdot b + r\cdot\overline{a}\cdot b + r\cdot a\cdot\overline{b} + r\cdot a\cdot b$

Σ	$a\overline{b}$	ab	$\overline{a}b$	$\overline{a}\overline{b}$
r		1		1
$\overline{r}$	1		1	

R	$a\overline{b}$	ab	$\overline{a}b$	$\overline{a}\overline{b}$
r	1	1	1	
$\overline{r}$		1		

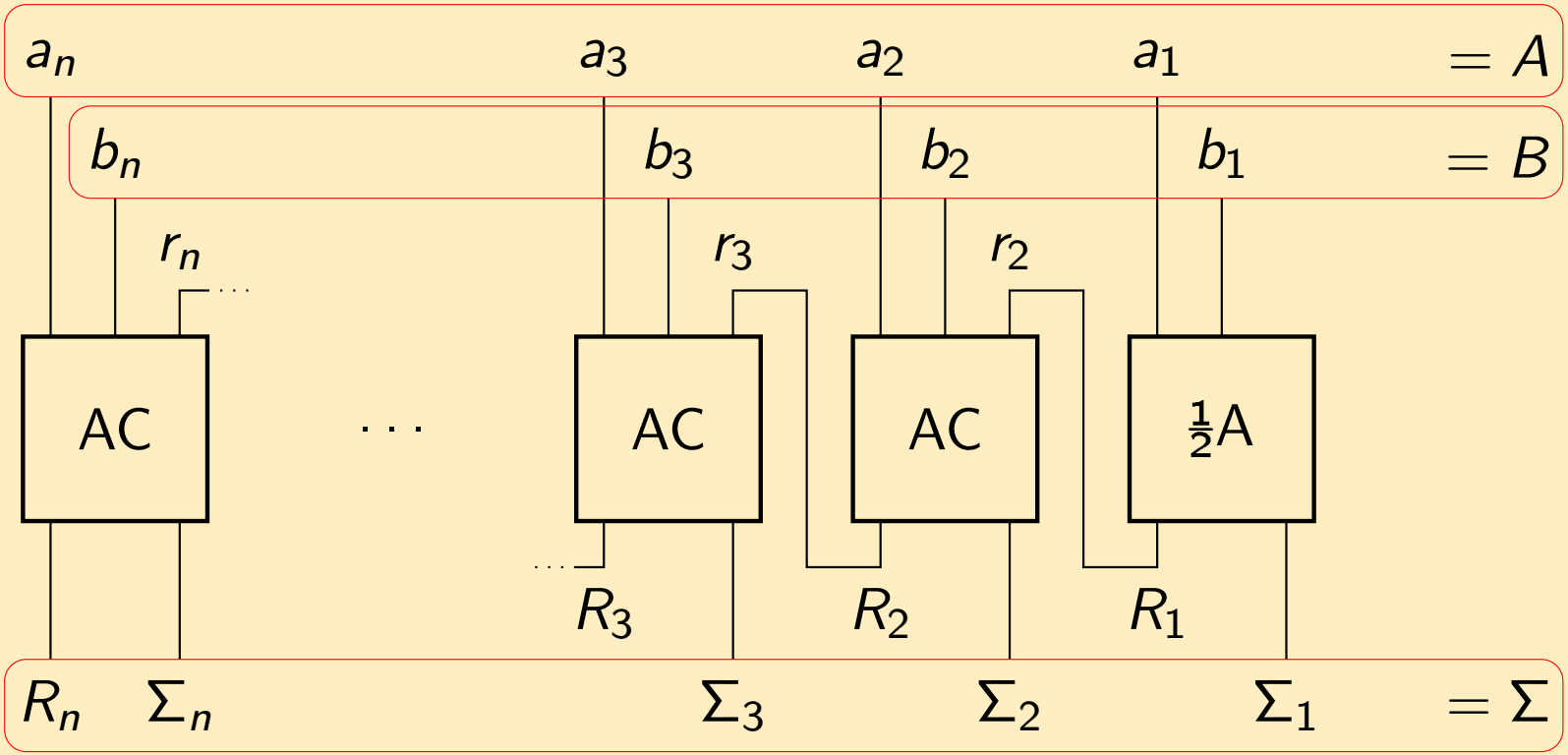
Σ : pas de simplifications !

$R = a\cdot b + r\cdot(a+b)$

Circuit intégré TTL 74 80 : additionneur complet

Cascader les additionneurs

Pour sommer deux nombres A, B de n bits



a_1 et b_1 sont les bits de poid le plus faible

$\frac{1}{2}$A est un demi-additionneur, AC un additionneur complet

L'additionneur-soustracteur

Addition

r	a	b	Σ	R_a
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Σ ... Somme, R_a ... Retenue

$$\Sigma = \overline{r} \cdot (\overline{a} \cdot b + a \cdot \overline{b}) + r \cdot (\overline{a} \cdot \overline{b} + a \cdot b)$$

$$R_a(r, a, b) = a \cdot b + r \cdot (a + b)$$

Soustraction

r	a	b	D	R_s
0	0	0	0	0
0	0	1	1	1
0	1	0	1	0
0	1	1	0	0
1	0	0	1	1
1	0	1	0	1
1	1	0	0	0
1	1	1	1	1

D ... Différence, R_s ... Retenue

$$D(r, a, b) = \Sigma(r, a, b)$$

$$R_s(r, a, b) = R_a(r, \overline{a}, b)$$

Facile de transformer un additionneur en soustracteur

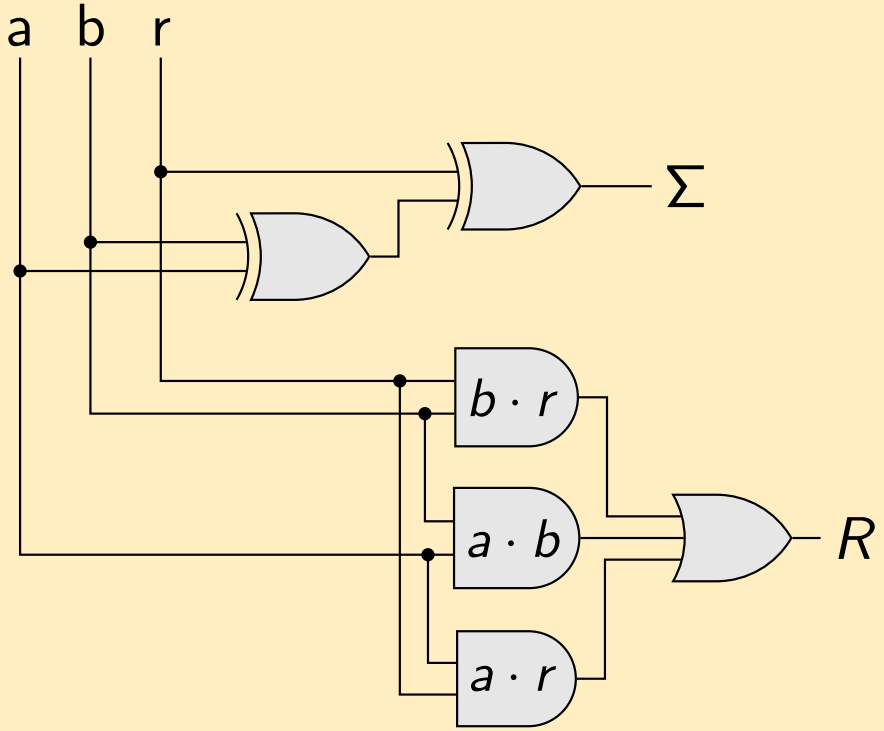
L'additionneur-soustracteur

Facile de transformer un additionneur en soustracteur

Addition

$$\Sigma = (\overline{a} \cdot b + a \cdot \overline{b}) \cdot \overline{r} + (\overline{a} \cdot \overline{b} + a \cdot b) \cdot r$$

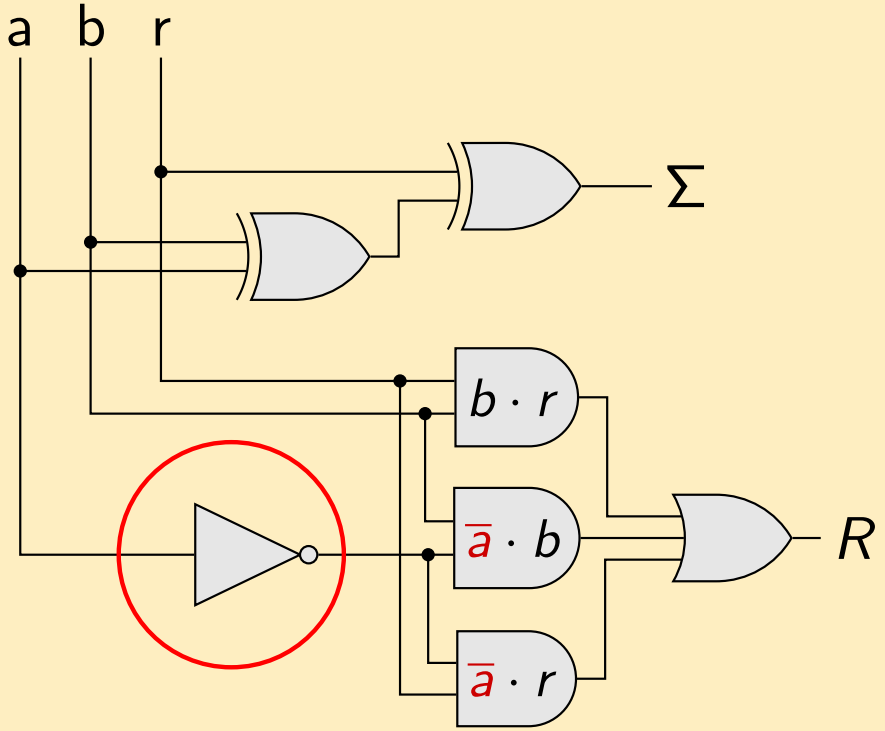
$$R(r, a, b) = a \cdot b + a \cdot r + b \cdot r$$



Soustraction

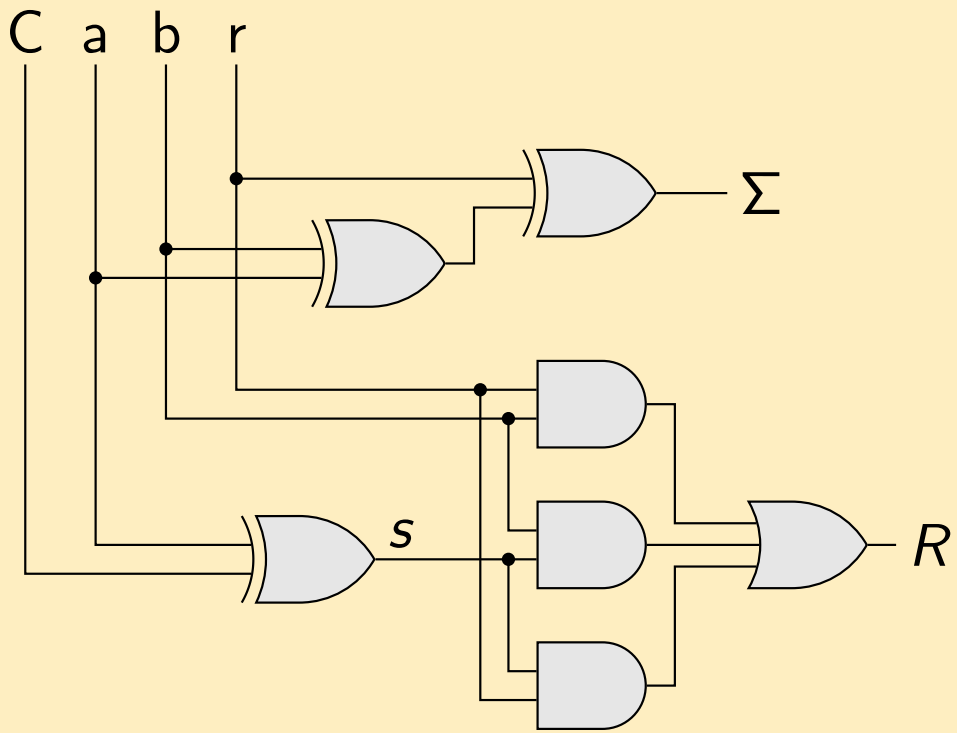
$$\Sigma = (\overline{a} \cdot b + a \cdot \overline{b}) \cdot \overline{r} + (\overline{a} \cdot \overline{b} + a \cdot b) \cdot r$$

$$R(r, a, b) = \overline{a} \cdot b + \overline{a} \cdot r + b \cdot r$$



L'additionneur-soustracteur

Mieux : choix à la volée entre addition et soustraction



C contrôle le comportement du circuit :

- $C = 0 \Rightarrow$ addition
- $C = 1 \Rightarrow$ soustraction

C	a	s	
0	0	0	$= a$
0	1	1	
1	0	1	$= \overline{a}$
1	1	0	

Exemple simple d'une unité arithmétique logique (ALU)

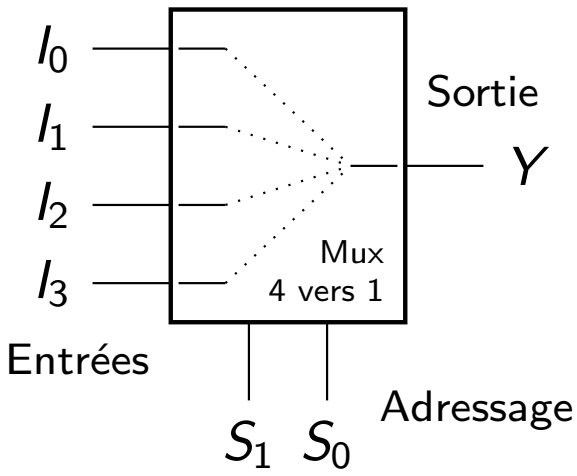
Le multiplexeur 2^n vers 1

- Aiguillage d'une parmi 2^n entrées I_k vers une unique sortie Y.
- Le numéro d'entrée k à connecter est indiqué en représentation binaire aux n lignes d'*adressage* :

$$(S_{n-1} \dots S_1 S_0) = k \Rightarrow Y = I_k$$

- Forme normale disjonctive ($n = 2$) :

$$\begin{aligned} Y = & I_0 \overline{S_1}\,\overline{S_0} + I_1 \overline{S_1} S_0 \\ & + I_2 S_1 \overline{S_0} + I_3 S_1 S_0 \end{aligned}$$



k	S_1	S_0	I_0	I_1	I_2	I_3	Y
0	0	0	0	X	X	X	0
			1	X	X	X	1
1	0	1	X	0	X	X	0
			X	1	X	X	1
2	1	0	X	X	0	X	0
			X	X	1	X	1
3	1	1	X	X	X	0	0
			X	X	X	1	1

Le multiplexeur 2^n vers 1

- Aiguillage d'une parmi 2^n entrées I_k vers une unique sortie Y.
- Le numéro d'entrée k à connecter est indiqué en représentation binaire aux n lignes d'*adressage* :

$$(S_{n-1} \dots S_1 S_0) = k \Rightarrow Y = I_k$$

- Forme normale disjonctive ($n = 3$) :

$$Y = I_0 \overline{S_2 S_1 S_0} + I_1 \overline{S_2 S_1} S_0 + \dots + I_7 S_2 S_1 S_0$$

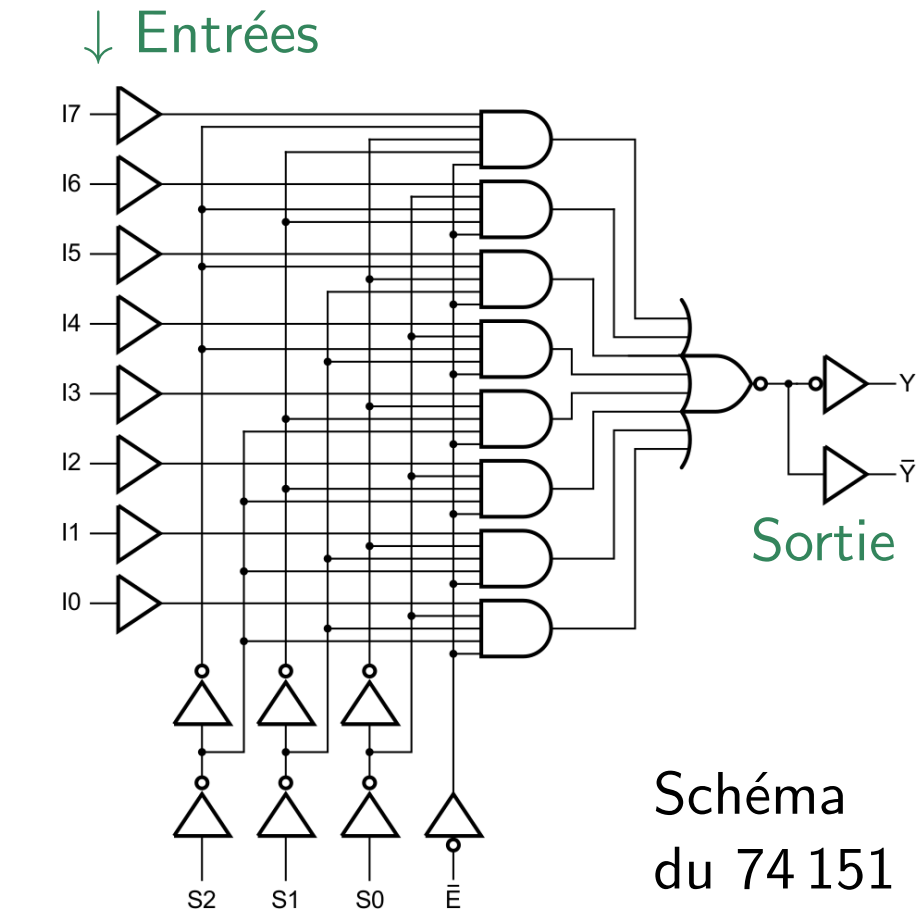


Schéma du 74 151

Réalisation de fonctions logiques à l'aide d'un multiplexeur

Fonction de n variables :

1. Connecter les variables aux lignes de sélection
2. Connecter les entrées du mux à 1 ou 0 selon la table de vérité

Exemple : addition

r	a	b	Σ	R
S_2	S_1	S_0	$\downarrow I_k$	
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

- $S_2 = r, \quad S_1 = a, \quad S_0 = b,$
- $(I_0 I_1 I_2 I_3 I_4 I_5 I_6 I_7) = (01101001)$ selon la colonne Σ.

(Deuxième mux requis pour R)

Réalisation de fonctions logiques à l'aide d'un multiplexeur

Fonction de $n+1$ variables :

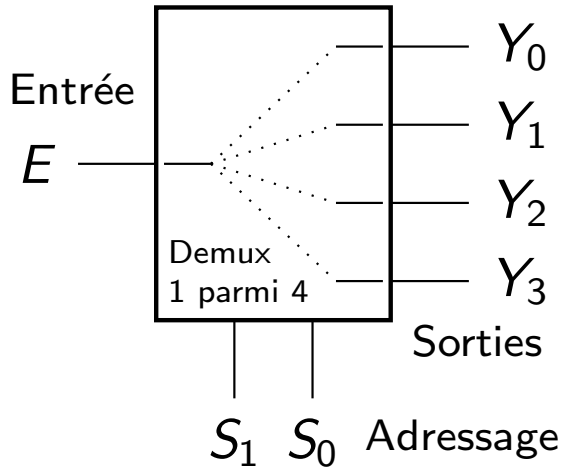
1. Connecter les n premières variables aux lignes de sélection (appelons z la variable restante)
2. Connecter les entrées du mux à 1 ou 0, z ou $\overline{z}$ selon la table de vérité

Le démultiplexeur

Opération inverse du multiplexeur :

- Aiguillage de l'entrée unique vers l'une des 2^n sorties
- Les autres sorties deviennent constantes $= 1$
- Sélection se fait à l'aide de n bits d'adressage

$$Y_k = \begin{cases} E & \text{si} \quad k = (S_1 S_0)_2 \\ 1 & \text{sinon} \end{cases}$$



Le démultiplexeur

Opération inverse du multiplexeur :

- Aiguillage de l'entrée unique vers l'une des 2^n sorties
- Les autres sorties deviennent constantes $= 1$
- Sélection se fait à l'aide de n bits d'adressage

E	S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	0	0	1	1	1	1
1	0	1	1	1	1	1
1	1	0	1	1	1	1
1	1	1	1	1	1	1

Exemple d'application : activer un dispositif parmi 2^n (en *logique négative* : $0 \Rightarrow$ activation)

Le démultiplexeur

Opération inverse du multiplexeur :

- Aiguillage de l'entrée unique vers l'une des 2^n sorties
- Les autres sorties deviennent constantes $= 1$
- Sélection se fait à l'aide de n bits d'adressage

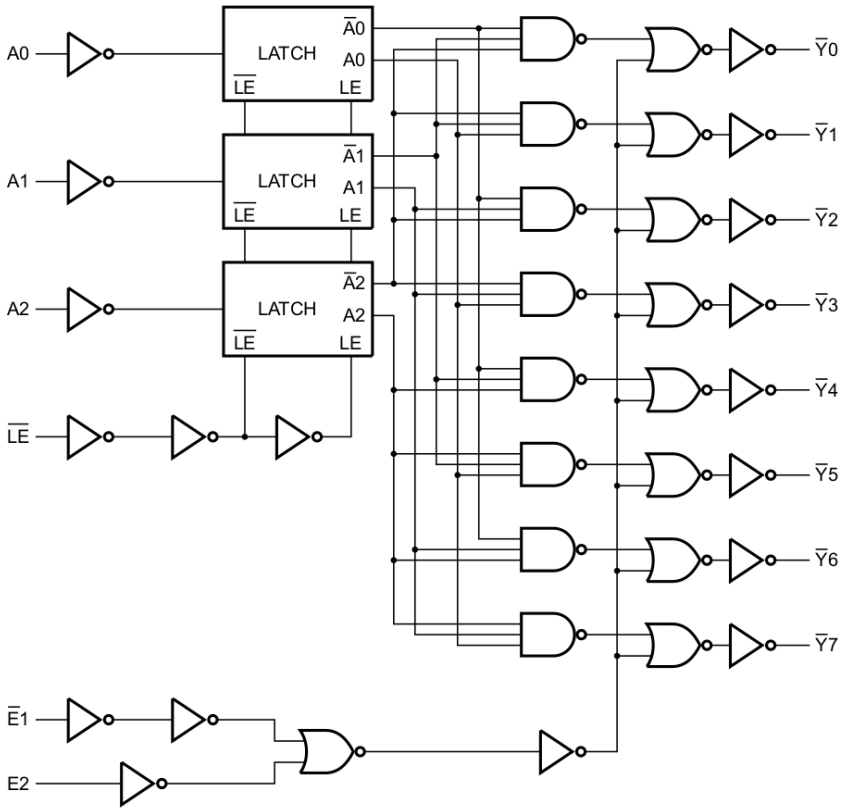


Schéma du 74 137

Codeur (décodeur, transcodeur)

Traduction entre un mot sur n bits et un mot sur m bits.

Exemple : encodeur prioritaire $n = 2^m$ vers m :

Position du premier L en comptant de la droite (si aucun $L \Rightarrow \mathrm{EO} = L$)

Exemple d'application : scanner un clavier, ordre de priorité d'interruption.

FUNCTION TABLE – '148, 'LS148

INPUTS									OUTPUTS				
EI	0	1	2	3	4	5	6	7	A2	A1	A0	GS	EO
H	X	X	X	X	X	X	X	X	H	H	H	H	H
L	H	H	H	H	H	H	H	H	H	H	H	H	L
L	X	X	X	X	X	X	X	L	L	L	L	L	H
L	X	X	X	X	X	X	L	H	L	L	H	L	H
L	X	X	X	X	X	L	H	H	L	H	L	L	H
L	X	X	X	X	L	H	H	H	L	H	H	L	H
L	X	X	X	L	H	H	H	H	H	L	L	L	H
L	X	X	L	H	H	H	H	H	H	L	H	L	H
L	X	L	H	H	H	H	H	H	H	H	L	L	H
L	L	H	H	H	H	H	H	H	H	H	H	L	H

H = high logic level, L = low logic level, X = irrelevant

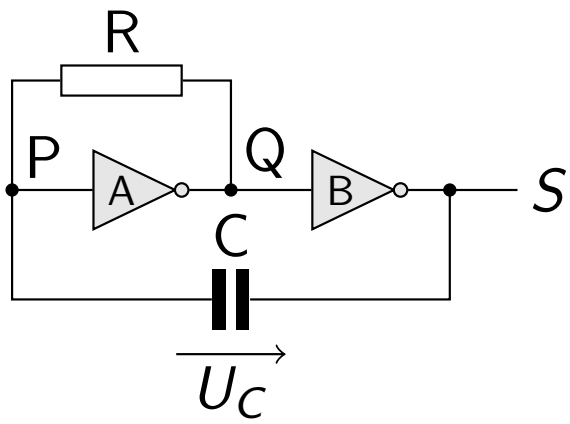
Tableau de vérité condensé du 74148 (non-condensé : 512 lignes !)

Codeur (décodeur, transcodeur) — suite

Autres exemples de transcodeurs

- Adressage d'éléments d'affichage : segments (TD : 7 segments) ou pixels
- Calcul de codes correcteurs
- Mot avec code correcteur $\rightarrow$ mot corrigé
- ...

Circuit astable : le multivibrateur



$$f \simeq \frac{1}{(2 \log 3) RC}$$

Supposons : $1 = [2.5 \ldots 5]\,\mathrm{V}$, $0 = [0 \ldots 2.5]\,\mathrm{V}$.
Si au départ $U_P = U_S = 5\,\mathrm{V}$ et $U_Q = 0\,\mathrm{V}$, alors :

1. C se charge lentement à travers $R \rightsquigarrow U_P \searrow$
2. Quand $U_P < 2.5\,\mathrm{V}$, A voit un 0 logique à l'entrée et bascule sa sortie brutalement vers $U_Q = 5\,\mathrm{V}$ (1 logique).
3. Suite au changement de son l'entrée Q, B bascule également sa sortie S : $5\,\mathrm{V} \rightarrow 0\,\mathrm{V}$.
4. À cause de la capacité, U_P chute également de $5\,\mathrm{V}$. $U_C \simeq +2.5\,\mathrm{V}$ au moment du basculement, donc $U_P \simeq$ -2.5 V. A voit toujours un 0 logique.
5. Le condensateur se décharge à tavers R, puis se charge à polarité opposée.
6. Quand $U_P \geq 2.5\,\mathrm{V}$, A voit un 1 logique à l'entrée et bascule sa sortie Q à $0\,\mathrm{V}$ ...

Mémoire

- n'est pas combinatoire
- opération de lecture et d'écriture unité par unité (bit, mot)
  - adressage d'une cellule (bit, ou mot de m bits)
  - lecture ou modification du contenu
- $\exists$ différents types de mémoires :
  - persistante (contenu conservé lors d'une coupure de courant)
    - lecture seule (ROM : Read Only Memory)
    - ecriture une fois, ensuite que lecture (PROM : Programmable ROM)
    - écriture lente, lecture rapide (Mémoire Flash, (E)EPROM : (Electronically) Erasable PROM)
  - non persistante (p.ex. RAM d'un PC)
    - écriture/lecture rapide (RAM : Random Access Memory)
    - statique ou dynamique (doit être raffraichie périodiquement)

Mémoires pour la réalisation de fonctions logiques

Terme technique : *Look Up Table* (LUT)

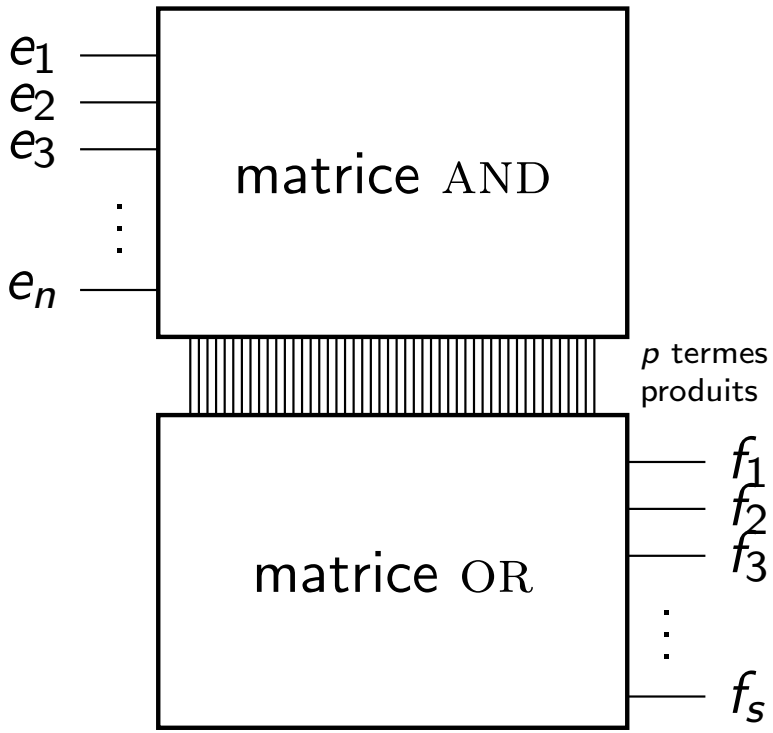
- Utilisation des variables comme lignes d'adressage
- Écriture : stockage de la table de vérité dans la mémoire
- Lecture : réalisation de la fonction

- Méthode souple, très performante en vitesse/surface
- Utilisée dans les FPGA

Composants logiques à architecture programmable

Programmable Logic Array (PLA)

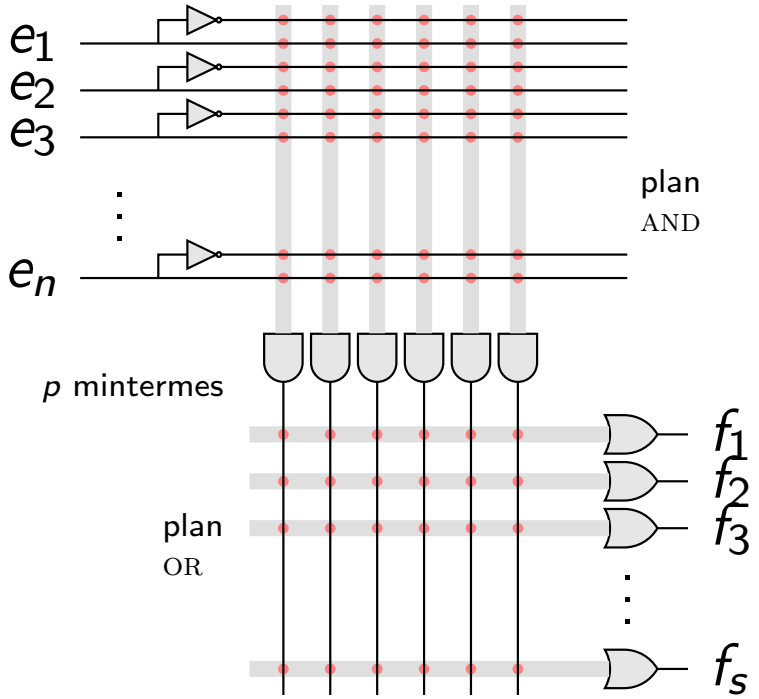
- Connexions programmables (une seule fois ou plusieurs)
- Exemple : $n = 16$ entrées/variables, $p = 48$ termes conjonctifs, $s = 8$ disjonctions/fonctions arbitraires de ces termes.



Composants logiques à architecture programmable

Programmable Logic Array (PLA)

- Connexions programmables (une seule fois ou plusieurs)
  au niveau des points rouges



Fin du deuxième cours

TD en deux groupes à partir de 11 h.

À la semaine prochaine !