Circuits Logiques

Michel Billaud

vers 2000

Table des matières

1 Eléments de Technologie

Nous avons vu comment coder l'information par des suites de valeurs binaires. Nous allons voir maintenant comment représenter ces valeurs binaires par des signaux électriques afin de les manipuler par des circuits électroniques.

1.1 Représentations de la logique binaire

1.1.1 Représentation des signaux

Il existe plusieurs façons de coder les valeurs 0 et 1. La plus simple, que nous adopterons dans la suite, est appelée *logique positive* et consiste à représenter 1 par la présence d'une tension (par exemple +5v) et 0 par une tension nulle.[1]

1.1.2 Génération d'un signal

Il est facile de fournir de tels signaux à un circuit: il suffit d'un interrupteur à deux positions (utilisé pour les va-et-vient); dans une position la sortie de l'interrupteur transmettra la tension +V, dans l'autre 0v (voir figure [fig1])

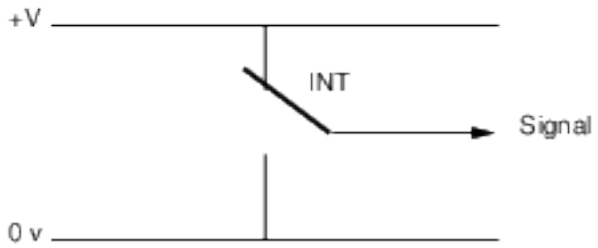


Figure 1: Génération d'un signal d'entrée

En pratique, on utilise plutôt (figure [fig2]) un interrupteur simple dont la sortie est reliée à une *résistance de rappel* R assez forte. Lorsque l'interrupteur est fermé (il fat contact) la sortie est à +V, lorsqu'il est ouvert la résistance ramène la tension de sortie vers 0. De plus ceci évite d'avoir une entrée "en l'air" quand l'interrupteur est entre deux positions.

1.1.3 Observation d'un signal

Pour observer la sortie d'un circuit on peut utiliser (figure [ampoule]) une ampoule du voltage voulu (5v = lampe de poche).

On préfèrera généralement employer des diodes électro-luminescentes (LED = Light Emitting Diode), qui coûtent moins cher et nécessitent un courant moindre (voir figure [temoin-led]). 0

[1] Une logique *négative* aura des signaux inversés: 0v représente la valeur 1, +5v la valeur 0.

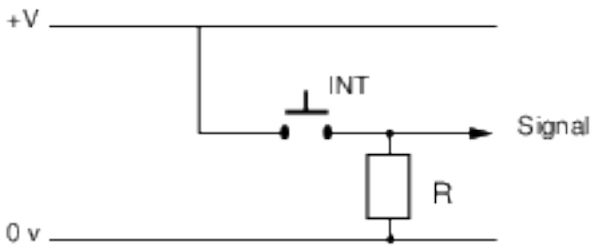


Figure 2: Génération d'un signal d'entrée (2)

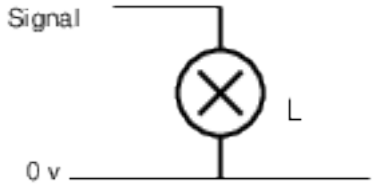


Figure 3: Ampoule témoin

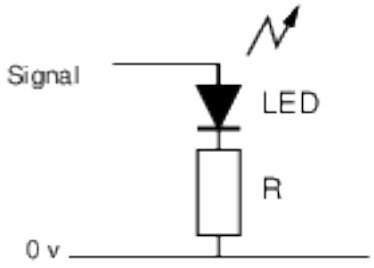


Figure 4: LED témoin

Encore mieux, pour éviter de trop “tirer” sur la sortie du circuit observé, on pourra utiliser un transistor en amplification (voir plus loin [trans-ampli]).

1.2 Notions rudimentaires d’électronique

Quelques notions sommaires d’électronique sont nécessaires pour la compréhension des circuits logiques de base. Le lecteur est supposé connaître la loi d’Ohm et des rudiments d’électricité.

1.2.1 La diode

La diode à semi-conducteurs est un composant électronique muni de deux bornes: l’*anode* et la *cathode* (voir [diode]).

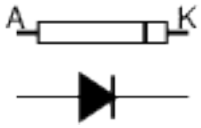


Figure 5: La diode: apparence physique et symbole

La propriété fondamentale de la diode est d’opposer qu’une résistance très faible lorsqu’elle est traversée par un courant de l’anode vers la cathode (sens passant), et une résistance très forte dans le sens inverse. Voir figure [diode-on-off].

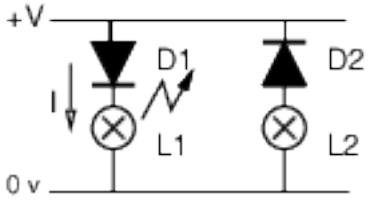


Figure 6: Diodes passante (D1) et bloquée (D2)

Pour analyser les circuits logiques nous considérerons que les diodes sont parfaites, c’est-à-dire ayant une résistance nulle dans l’état passant, et infinie dans l’état bloqué.

> **Remarque importante.** Lorsque nous construirons des circuits nous aurons soin d’éviter de faire traverser les diodes par des courants trop forts (risque de claquage). Il faudra donc de les monter en série avec des résistances pour limiter le courant.

1.2.2 La diode électro-luminescente

Les diodes électro-luminescentes (LED = Light Emitting Diod, voir fig. [fig9]) émettent de la lumière quand elles sont traversées par un courant de l’ordre de 10 à 20 mA dans le sens passant. Ces diodes offrant une résistance très faible, il conviendra de les monter en série avec une *résistance limitatrice de courant* d’une valeur suffisante ($R = U/I = 5v/10mA = 500\Omega$).

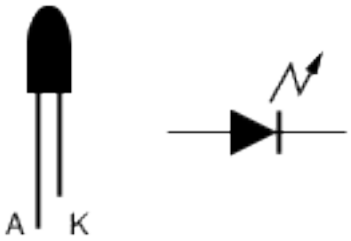


Figure 7: LED: apparence physique et schéma

1.2.3 Le transistor

Le transistor[2] est un composant à 3 pattes: l’émetteur, le collecteur et la base (voir figure [transistor]).

[2]Pour simplifier l’exposé nous n’évoquerons que les transistors NPN.

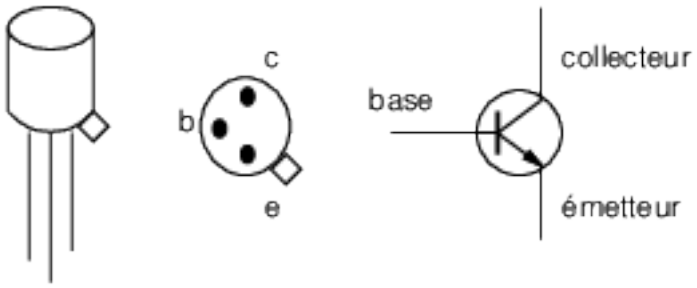


Figure 8: Transistor NPN: boitier, broches (vues de dessous), schéma

Dans les montages usuels, le collecteur est relié à la tension d'alimentation $+V$ au travers d'une résistance R_c, l'émetteur étant relié à la masse (0v). Le courant I_b qui traverse la base permet de contrôler l'intensité de celui qui traverse l'émetteur et le collecteur.

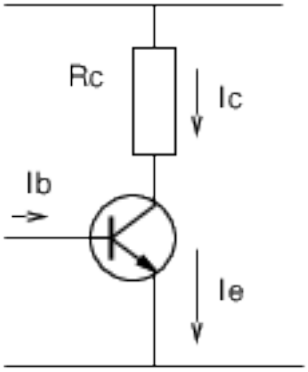


Figure 9: Polarisation d'un transistor

Lorsque le transistor est utilisé en amplification, les courants sont liés par les équations

- $I_e = I_c + I_b$
- $I_c = \beta.I_b + I_{ce}$

Les deux constantes β et I_{ce} dépendent du transistor: le gain β est de l'ordre de 100, le courant I_{ce} vaut quelques micro-ampères pour les transistors au silicium, et quelques nano-ampères pour les transistors au germanium. En général on le néglige pour les calculs.

Les montages logiques utilisent le mode de fonctionnement *bloqué-saturé*, en débordant largement de la plage de valeurs où les formules ci-dessus sont valides: si on applique une tension assez forte (proche de +V) à la base, l'intensité I_b est maximum et le transistor n'oppose qu'une résistance très faible entre émetteur et collecteur: le transistor est saturé. Si on applique une tension nulle à la base, le courant I_c sera négligeable : le transistor est alors dit bloqué.

[trans-ampli]

Ceci justifie le montage de la figure [ampli-diode] : lorsque l'entrée est à +V, le transistor est saturé, donc il laisse passer le courant à travers la LED qui s'éclaire. Lorsque l'entrée est au niveau bas (0v), le transistor est bloqué et la LED reste éteinte. En raison des propriétés amplificatrices du transistor, le courant de base n'a pas besoin d'être très élevé (de l'ordre de I_c/β), par conséquent on peut mettre une résistance limitatrice sur la base. Cette résistance évite de trop "tirer de courant" du signal que l'on observe, ce qui risquerait de perturber son fonctionnement.

1.3 Portes logiques

En assemblant ces composants on obtient des *portes logiques* qui combinent les signaux.

1.3.1 Porte OU

1.3.1.1 Montage à diodes Deux diodes et une résistance suffisent pour réaliser une *porte OU* selon le schéma de la figure [porte-ou].

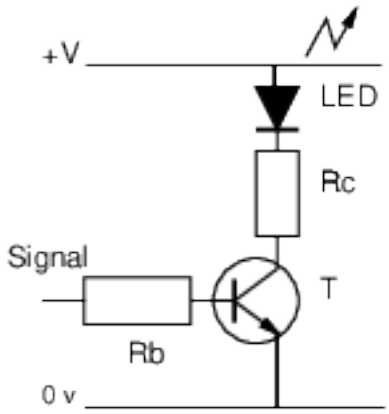


Figure 10: LED amplifiée par un transistor

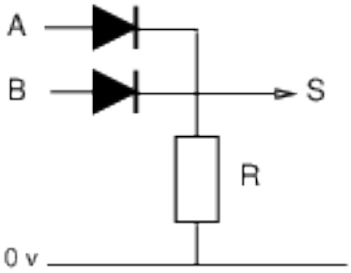


Figure 11: Porte logique OU à diodes

1.3.1.2 Analyse du montage Supposons que les deux entrées A et B soient portées au potentiel +V. Les deux diodes sont dans le sens passant, et laissent donc passer le courant. Tout se passe alors comme si la sortie S était reliée à la même tension +V.

Si A et B sont reliées à la masse (0v), les diodes ne conduisent pas le courant. La sortie S est donc ramené à 0v par la résistance de rappel.

Si une des entrées est à +V et l'autre à la masse, la diode passante suffit, comme dans le premier cas, à amener la tension +V sur S.

1.3.1.3 Table de vérité de l'opérateur OU Si nous prenons la convention de logique positive, la tension +V correspond à la valeur logique "vrai" (notée 1) et 0v à "faux" (0). La table de vérité de l'opération "+" ainsi obtenue est celle de l'opérateur "OU logique": $

A	B	$A+B$
0	0	0
0	1	1
1	0	1
1	1	1

La sortie est à 1 si au moins une des entrées est à 1.

1.3.1.4 Porte OU multiple En reliant plusieurs entrées de la même façon on obtient une porte OU multiple. Son fonctionnement se résume en une phrase: la sortie est à 1 si au moins une des entrées est à 1.

Dans les schémas logiques, on représente les portes logiques par un symbole (figure [log2])

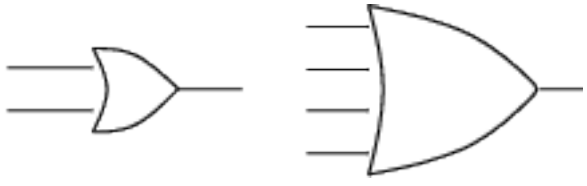


Figure 12: Symboles des portes OU

1.3.1.5 Montage à transistors Le montage de la figure [log8] remplit la même fonction. L'analyse en est laissée au lecteur.

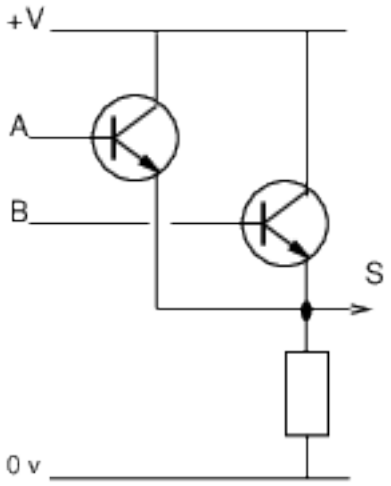


Figure 13: Porte OU à transistors en parallèle

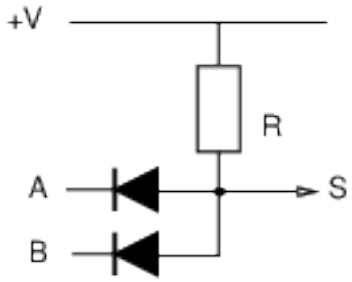


Figure 14: Porte ET à diodes

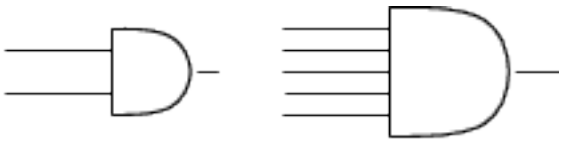


Figure 15: Symboles des portes ET

1.3.2.4 Montage à transistors Le montage de la figure [log7] remplit la même fonction. L'analyse en est laissée au lecteur.

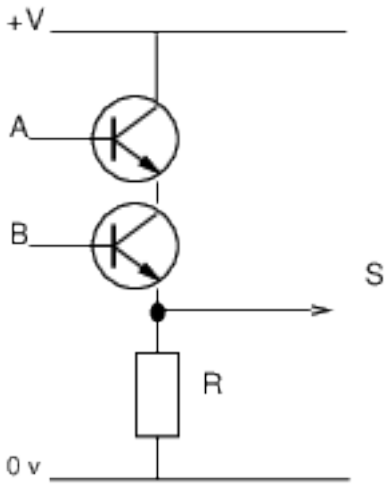


Figure 16: Porte ET à transistors en série

1.3.3 Porte NON

1.3.3.1 Montage Dans la figure [log5] on utilise un transistor en mode bloqué/saturé.

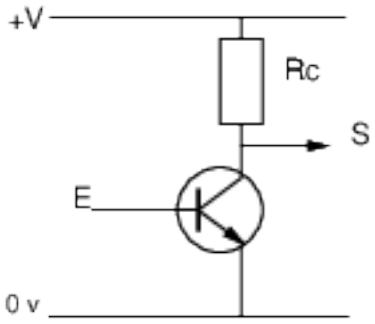


Figure 17: Porte NON à transistor

1.3.3.2 Analyse du montage Lorsque l'entrée est au niveau haut, le transistor est saturé. La sortie est alors reliée à la masse par l'intermédiaire de la résistance interne très faible du transistor : la sortie S est au niveau bas.

Lorsque l'entrée est au niveau bas, le transistor est bloqué. La résistance de rappel ramène donc la sortie au niveau haut.

1.3.3.3 Table de vérité de l'opérateur NON La table de l'opérateur NON "¬" ainsi obtenu est

A	$\neg A$
0	1
1	0

La sortie est à 1 si et seulement si l'entrée est à 0.

On symbolise cet opérateur (figure [log10]) par un triangle (indiquant par convention l'amplification) suivi d'un rond (négation).

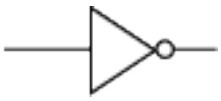


Figure 18: Symbole de la Porte NON

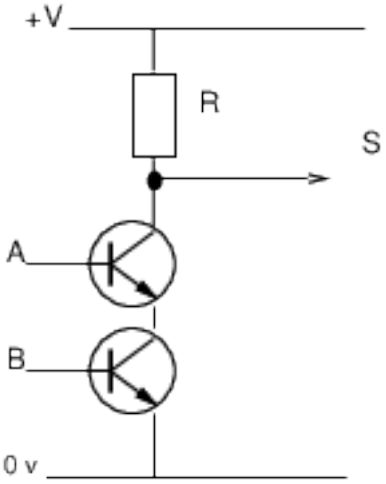


Figure 19: Porte NAND

1.3.4 Porte NON-ET (NAND)

Le montage de la figure [log9] réalise une fonction dont la table de vérité est:

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

On remarque aisément que l'on a $S = \neg(A.B)$, et on appelle cet opérateur le NON-ET (ou "nand").

Exercice Donnez un schéma de cet opérateur utilisant 2 diodes, une résistance et un transistor.

On le représente par un symbole "ET" suivi du rond qui indique la négation (figure [log11]).

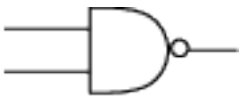


Figure 20: Porte NON-ET

La porte NAND a un intérêt pratique évident: elle permet de reconstituer tous les autres types de portes.

- La porte NON, puisque $\neg A = A \; nand \; 1$
- La porte ET, puisque $A.B = \neg(A \; nand \; B) = (A \; nand \; B) \; nand \; 1$
- La porte OU, puisque $A + B = \neg(\neg A \cdot \neg B)$ et donc $A + B = ((A \; nand \; 1) \; nand \; (B \; nand \; 1)) \; nand \; 1$.

1.3.5 Porte NON-OU (NOR)

La fonction "non-ou" (symbolisée fig. [log14]) est définie de la même façon, par l'équation

$$nor(A, B) = \neg(A + B)$$

□

Exercice Proposez une porte NOR à transistors.

1.3.6 Porte OU-exclusif (XOR)

La fonction XOR "ou-exclusif" (fig. [log15]) est souvent notée $\oplus$. On la définit par:

$$A \oplus B = (\neg A \cdot B) + (A \cdot \neg B)$$

Figure 21: Symbole de la porte XOR (ou-exclusif)

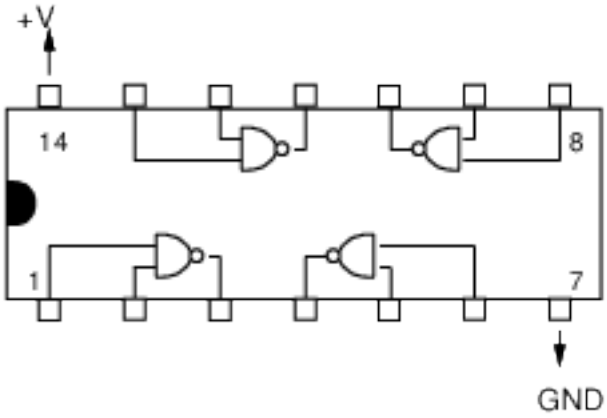


Figure 22: Circuit CMOS 4011 vu de dessus

- porte 2: entrées 8 et 9, sortie 10;
- porte 3: entrées 1 et 2, sortie 3;
- porte 4: entrées 5 et 6, sortie 4;

La figure [log13] montre comment réaliser une porte OU avec ce circuit.

Exercice Fonctions binaires usuelles Montrez comment réaliser les expressions $A.B$, $A \oplus B$, $A + \overline{B}$ à l'aide d'un circuit 4011.

Exercice Fonction majorité Réalisez la fonction $maj(A, B, C)$, dont le résultat est 1 si au moins deux entrées sont à 1, à l'aide d'un circuit 4011 (ou plusieurs).

Exercice Réalisez la fonction \$f(A,B,C)= si $A = 1$ alors $B.C$ sinon $B + C$\$.

1.3.7.2 Conditions d'emploi des circuits CMOS Les circuits de type CMOS présentent certains avantages pour les montages expérimentaux:

- ils acceptent une tension d'alimentation entre 3 et 15 V;
- leur consommation est très faible (de l'ordre de 0,1 mW par porte);
- les entrées des circuits CMOS ont une impédance très élevée : on peut relier de nombreuses entrées sur la sortie d'une porte sans craindre de "tirer" trop de courant de celle-ci.

Par contre, ces circuit sont sensibles à l'électricité statique[3]. De plus, la propagation des signaux de l'entrée à la sortie d'une porte est plus lente (20-40 ns) qu'avec d'autres familles de circuits, comme les TTL.

1.3.7.3 Emploi des circuits TTL Les circuits TTL (transistor-transistor-logic) sont très utilisés pour les réalisations professionnelles. A titre d'exemple, les caractéristiques de la série SN74 sont:

- tension nominale d'alimentation de $5v \pm 0.5V$, risque de claquage à partir de 7V
- fonctionnement entre 0 et 70^oC
- puissance par porte de l'ordre de 10 mW
- courant d'entrée de l'ordre de $1.5mA$
- temps de propagation de l'ordre de 10 à 20 ns

[3]En particulier "effet d'antenne" lorsqu'on approche la main d'un circuit dont une des entrées est restée "en l'air"

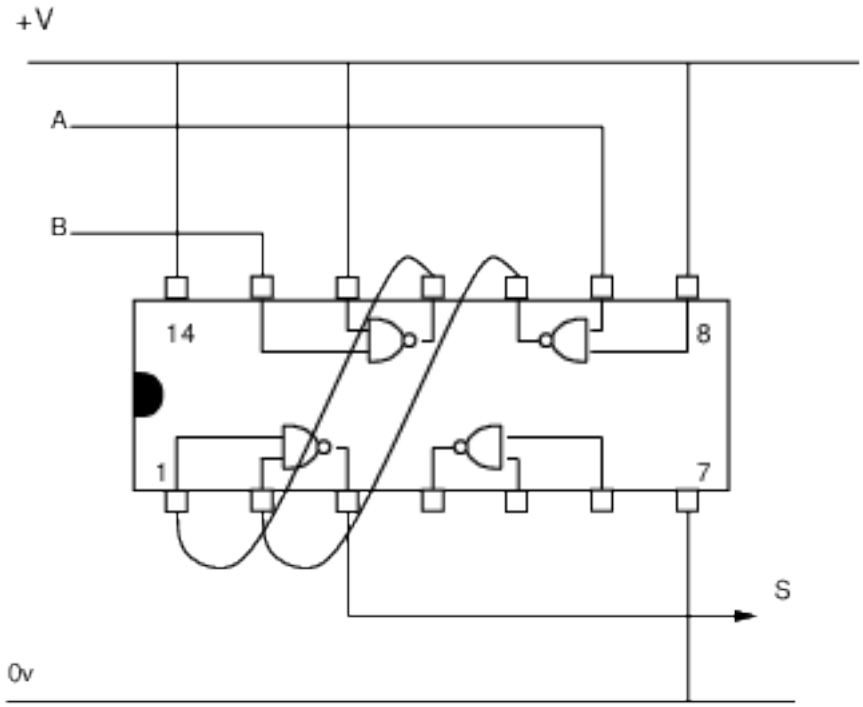


Figure 23: Porte OU avec CMOS 4011

2 Algèbre de Boole et circuits logiques

En combinant les portes logiques, et en prenant garde de ne pas faire de boucles dans les circuits[4], on réalise des circuits dont les sorties ne dépendent que des valeurs des entrées. On pourra alors décrire la fonction réalisée par ce circuit par sa table de vérité.

L'étude des "tables de vérité" a été entreprise il y a plus d'un siècle, et constitue ce qu'on appelle l'algèbre de Boole[5].

Les propriétés de cette algèbre nous permettront de *raisonner* sur les circuits, et donc de les construire rigoureusement, plutôt que par la (coûteuse) méthode des essais et erreurs. En particulier nous verrons des méthodes systématiques pour simplifier les circuits.

2.1 Définitions

On appelle *fonction booléenne* toute fonction de $\{0,1\}^n$ dans $\{0,1\}$.

Il y a 2^{2^n} fonctions booléennes à n variables : en effet la table de vérité pour décrire d'une fonction booléenne à n variables contient 2^n cases, qui contiennent chacune 0 ou 1.

Une *expression booléenne* est un terme construit à partir de variables (prises dans un ensemble V), de constantes 0 ou 1, et d'opérateurs ET, OU, et NON. Exemple : $a + \neg(a \cdot \neg b) + b$.

Une expression booléenne *représente* une fonction booléenne ; nous dirons que deux expressions sont équivalentes si elles dénotent la même fonction.

Exemple Si nous construisons la table de vérité de l'expression ci-dessus, et de l'expression $\neg a + b$:

a	b	$\neg b$	$a \cdot \neg b$	$\neg(a \cdot \neg b)$	$(\neg a \cdot \neg b) + b$	$\neg a$	$\neg a + b$
0	0	1	0	1	1	1	1
0	1	0	0	1	1	1	1
1	0	1	1	0	0	0	0

[4] les circuits combinatoires (sans boucles) et séquentiels (avec boucles) seront étudiés dans les chapitres suivants

[5] L'oeuvre la plus célèbre de George Boole (1815-1864) est intitulée *Une investigation dans les lois de la pensée, sur lesquelles sont fondées les théories mathématiques de la logique et des probabilités* (1854), plus connue sous le titre abrégé *Les lois de la pensée*

a	b	$\neg b$	$a \cdot \neg b$	$\neg(a \cdot \neg b)$	$(\neg a \cdot \neg b) + b$	$\neg a$	$\neg a + b$
1	1	0	0	1	1	0	1

nous constatons qu'elles coïncident : les deux expressions représentent la même fonction.

Par commodité, on appelle *somme* toute expression de la forme $t_1 + t_2 + \ldots + t_n$. Elle peut se réduire à un seul terme, ou aucun (dans ce cas c'est 0). Un *produit* est de la forme $t_1.t_2 \ldots t_n$. Le produit vide est 1.

Un monôme est un produit de variables ou de négations de variables, chaque variable n'apparaissant qu'au plus une fois.

Exemple de monômes: $x, \neg y, x \cdot \neg y$.

L'expression 1 est également un monome, en tant que produit vide.

Remarque À partir d'un ensemble de n variables on peut construire $C_n^k.2^k$ monômes distincts à k variables : pour choisir un tel monôme il suffit de sélectionner k variables parmi n (d'où le coefficient binomial), et d'attribuer ou non à chacune d'entre elles une barre (2^k possibilités).

Il y a en tout 3^n monômes, puisqu'il suffit de décider, pour chacune des variables, de la faire figurer telle quelle, avec une barre, ou pas du tout. Ceci prouve l'égalité:

$$C_n^0.2^0 + C_n^1.2^1 + \ldots + C_n^n.2^n$$

que l'on peut d'ailleurs retrouver en développant $(1+2)^n$ à l'aide des formules connues.

Un monôme est dit *complet* par rapport à un ensemble donné de variables si toutes les variables de cet ensemble apparaissent une fois (avec ou sans négation).

Il y a 2^n monômes complets sur un ensemble de n variables, chaque monôme complet correspondant à une des cases d'une table de vérité à n variables.

Une expression est *en forme canonique* si elle est écrite sous forme d'une somme sans répétition de monômes complets.

2.1.1 Proposition

Toute fonction booléenne peut être construite par composition d'opérations OU, ET et NON.

Preuve

Toute fonction booléenne $f(a, b, c \ldots)$ peut être décrite par sa table de vérité. Chaque case de cette table correspond à un *monôme complet*. Une expression de la fonction f s'obtient en faisant la somme des monômes complets dont la valeur est 1.

Exemple Considérons une fonction maj (majorité) à trois variables a, b, c, qui vaut 1 quand au moins deux des entrées sont à un. On dresse la table de vérité de maj, en faisant figurer dans la marge les monômes correspondants:

a	b	c	$maj(a,b,c)$	monôme
0	0	0	0	$\neg a \cdot \neg b \cdot \neg c$
0	0	1	0	$\neg a \cdot \neg b \cdot c$
0	1	0	0	$\neg a \cdot b \cdot \neg c$
0	1	1	1	$\neg a \cdot b \cdot c$
1	0	0	0	$a \neg b \cdot \neg c$
1	0	1	1	$a \cdot \neg b \cdot c$
1	1	0	1	$a \cdot b \cdot \neg c$
1	1	1	1	$a \cdot b \cdot c$

et on obtient l'expression en forme canonique (en omettant les points pour le "et") :

$$maj(a, b, c) = (\neg a)bc + a(\neg b)c + ab(\neg c) + abc$$

2.2 Propriétés des algèbres de Boole

Les opérations ET, OU, NON définies sur l'ensemble $\{0, 1\}$ possèdent les propriétés suivantes:

$\neg(\neg A) = A$		double négation
$\neg 0 = 1$	$\neg 1 = 0$	constantes
$\neg A + B = \neg A \cdot \neg B$	$\neg(A.B) = \neg A + \neg B$	dualité
(A+B)+C = A+(B+C)	$(AB)C = A(BC)$	associativité
$A + B = B + A$	$AB = BA$	commutativité
$(A + B)C = AC + BC$	$AB + C = (A + C)(B + C)$	distributivité
$A + A = A$	$AA = A$	idempotence
$A + 0 = A$	$A \cdot 1 = A$	éléments neutres
$A + 1 = 1$	$A \cdot 0 = 0$	absorption
$A + \neg A = 1$	$A \cdot \neg A = 0$	complémentaires

Remarques

- Chaque égalité peut-être vérifiée en construisant les tables de vérité de ses parties gauche et droite.
- La colonne de droite contient des équations *duales* de celles de la colonne de gauche, obtenues en intervertissant les opérateurs + et ·, et les constantes 0 et 1. On peut passer d'une équation à l'équation duale en utilisant seulement $\neg(\neg A) = A$, $\neg 0 = 1$ et $\neg(A + B) = \neg A \cdot \neg B$.
- On voit assez facilement que ces équations permettent de développer n'importe quel terme sous forme d'une expression canonique. L'expression canonique d'une fonction étant unique (à la commutation de + et . près), ces équations suffisent donc pour montrer l'équivalence de deux expressions.
- Les lois de dualité sont appelées aussi lois de De Morgan.[6]

2.2.1 Algèbre de Boole

On appelle *algèbre de Boole* tout ensemble qui possède deux constantes 0 et 1, et des opérations $+, \cdot, \neg$ qui satisfont les équations ci-dessus.

Exercice Montrez qu'il n'y a pas d'algèbre de Boole à 3 éléments $\{0, 1, 2\}$.

Indication : que peut bien valoir $\neg 2$?

On voit facilement que le produit $A \times B$ de deux algèbres de Boole est également une algèbre de Boole, dont les membres sont des couples. Les opérations sur les couples sont définies par

- $\neg(a, b) = (\neg a, \neg b)$,
- $(a, b) + (a', b') = (a + a', b + b')$,
- $(a, b) \cdot (a', b') = (a \cdot a', b \cdot b')$.

Dans le sens inverse, un résultat mathématique (théorème de Stone) dit que toute algèbre de Boole se ramène en fait à un produit d'algèbres de Boole à 2 éléments[7].

2.3 Simplification des expressions

À partir d'une expression booléenne, on pourra facilement fabriquer un circuit. Pour réaliser une même fonction, on aura tout intérêt à avoir des circuits qui utilisent le moins possible de portes logiques, pour des raisons de simplicité, de coût, de taille du circuit et de consommation de courant.

2.3.1 Approche algébrique

On peut essayer de simplifier les circuits en utilisant les équations vues plus haut. Par exemple, pour la fonction maj :

$maj(a, b, c) = \neg abc + a\neg bc + ab\neg c + abc$

$= \neg abc + a\neg bc + ab\neg c + abc + abc + abc$ (idempotence)

[6] Les travaux d'Augustus de Morgan (1806-1871) influencèrent fortement George Boole, et ses vives recommandations permirent à ce dernier, bien qu'autodidacte, d'obtenir la Chaire de Mathématiques du Queen's College de Cork.

[7] Ce produit peut être infini, mais c'est une autre histoire

$= \neg abc + abc + a\neg bc + abc + ab\neg c + abc$ (commutativité)

$= (\neg a + a)bc + a(\neg b + b)c + ab(\neg c + c)$ (distributivité)

$= 1.bc + a.1.c + ab.1$ (complémentarité)

$= bc + ac + ab$ (éléments neutres)

Mais dans ce genre de travail, la difficulté est de mener le calcul vers une expression simple ... que l'on ne connaît pas a priori.

Un petit problème mathématique ? Il existe $2^{2^2} = 16$ fonctions à deux variables.

1. Donnez une expression aussi simple que possible de chacune d'elles.
2. Déterminez celles qui sont *croissantes*, c'est-à-dire telles que $x \leq x'$ et $y \leq y'$ entraîne $f(x, y) \leq f(x', y')$. Pour chacune d'elles, montrez qu'elle peut s'exprimer (sans utiliser de négation) par une somme de produits de variables.
3. Réciproquement, montrez que toute fonction dont l'expression contient des ET et des OU (mais pas de NON) est nécessairement croissante.
4. Généralisez au cas des fonctions ayant un nombre quelconque de variables.

2.3.2 Méthode de Karnaugh

La méthode de Karnaugh est une méthode visuelle pour trouver une expression simple d'une fonction booléenne de quelques variables (jusqu'à 6).[8].

Elle repose sur une présentation particulière de la table de vérité de la fonction étudiée. Voici la disposition adoptée pour les fonctions de 3 variable a, b et c :

a b : c	0	1
0 0		
0 1		
1 1		
1 0		

et pour 4 variables :

a b : c d	0 0	0 1	1 1	1 0
0 0				
0 1				
1 1				
1 0				

La disposition est telle que les "coordonnées" de deux cases adjacentes ne diffèrent que par une seule variable. Comme chaque case correspond à un monôme complet, un groupement de 2 cases adjacentes représentera un monôme à $n - 1$ variables.

Exemple : dans la première table, - la case en bas à gauche correspond à $a = 1, b = 0, c = 0$, c'est-à-dire au monôme $a \cdot \neg b \cdot \neg c$. - pour sa voisine de droite, c'est le monôme $a \cdot \neg b \cdot c$. - en les regroupant, on obtient $a \cdot \neg b \cdot \neg c + a \cdot \neg b \cdot \neg c$ qui est égal à $a \cdot \neg b \cdot (c + \neg c)$, qui se simplifie en $a \cdot \neg b$.

Attention : il faut également considérer les bords opposés comme étant adjacents, en regroupant les cases en haut à gauche et en bas à gauche, on obtient $\neg b \cdot \neg c$.

Les monômes à $n - 2$ variables sont visualisés sous forme de carrés, ou de rectangles 1×4.

Exemples

- On a $bd = (a + \neg a)b(c + \neg c)d$. En développant cette expression, on trouve une somme de 4 monômes complets qui occupent le milieu de la seconde table.
- Les 4 coins de cette même table forment (virtuellement) un carré dont l'expression est $> \neg a \cdot \neg d$.

[8] Nous ne montrerons ici que la méthode pour 3 et 4 variables, le passage à 5 ou 6 variables nécessite une bonne vision dans l'espace

- La colonne de droite correspond à $> c \cdot \neg d$, etc.
- chaque monôme d'une seule variable (sur 4) occupe 8 cases disposées en rectangles 2×4.

La méthode de Karnaugh consiste à

1) écrire la table de vérité de la fonction dans la table
2) procéder à des regroupements que l'on entoure
3) écrire la somme des monomes correspondants.

2.3.2.0.1 Algorithme L'algorithme est le suivant:

- Si tous les "1" ont été entourés : arrêter.
- trouver, visuellement, le plus gros regroupement possible contenant au moins un 1 non entouré.
- l'entourer sur le tableau, et écrire son expression
- recommencer

La figure [boole1] illustre le déroulement possible de la méthode sur l'exemple de la fonction majorité. Le résultat obtenu est $BC + AB + AC$.

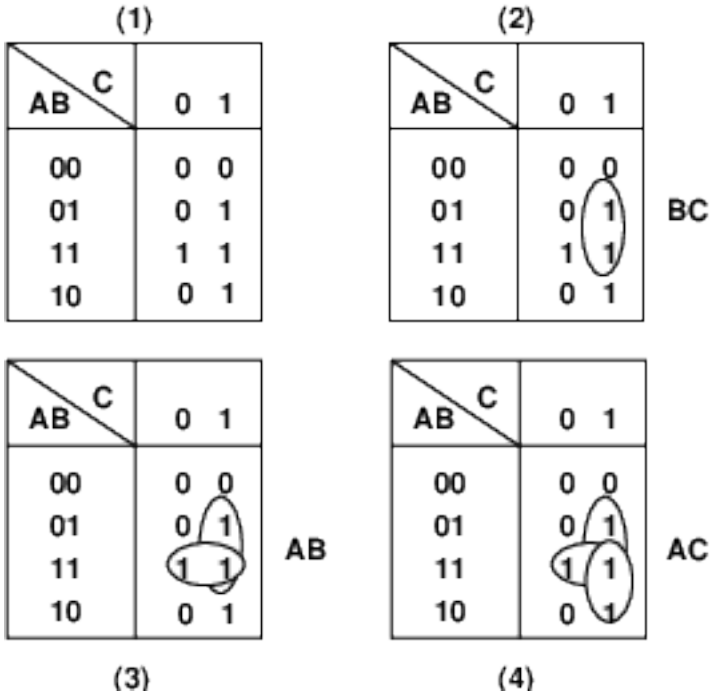


Figure 24: Méthode de Karnaugh : exemple

Exercice. Le vote : Une commission est composé d'un président P et trois membres A,B,C. Cette commission doit se prononcer sur un vote par oui (1) ou non (0) à la majorité. Personne ne peut s'abstenir. En cas d'égalité entre les oui et les non, la voix du président est prépondérante. En utilisant la méthode de Karnaugh, donnez une expression simple de la fonction $vote(A, B, C, P)$.

Exercice Décodage 7 segments hexadécimal Un afficheur 7 segments est composé de 7 diodes électroluminescentes notés a,b,... g (voir figure [boole2]). Un nombre est fourni, codé en binaire sur 4 bits $x_3x_2x_1x_0$, qui devra être affiché. Donnez l'expression des 7 fonctions $a(x_3, x_2, x_1, x_0), b(x_3, x_2, x_1, x_0), \ldots$

La méthode de Karnaugh est également applicable dans le cas des fonctions incomplètement spécifiées, c'est-à-dire qui ont des "cases vides" dans lesquelles on peut mettre ce que l'on veut, parce que cela correspond à des situations qui ne peuvent pas se produire.

La méthode est alors la suivante: former les plus gros paquets possibles contenant au moins un 1 non entouré et aucun 0.

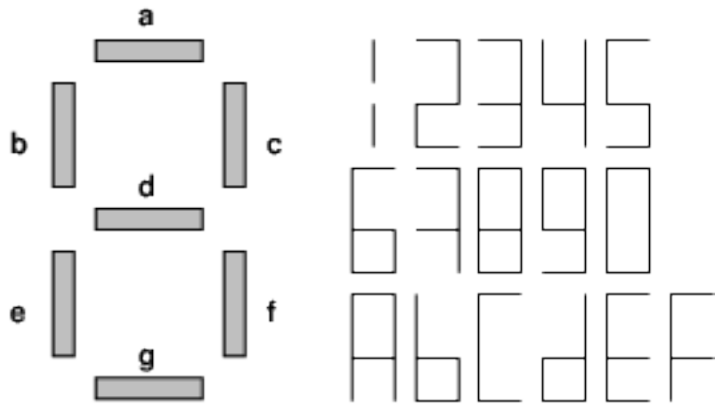


Figure 25: Afficheur 7 segments, chiffres

Exercice Le dé électronique (fig. [boole3]) possède trois entrées, par lesquelles arrivent des nombres de 1 à 6 codés en binaire. Donnez une fonction pour chacune des lampes qui indiquera si elle doit être allumée ou éteinte.

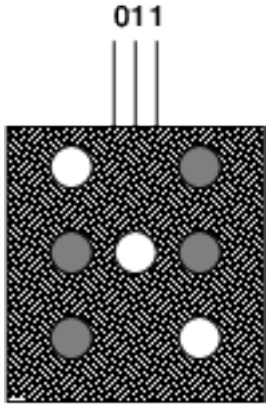


Figure 26: Le dé électronique

Exercice Décodage 7 segments décimal Reprendre l'exercice de l'afficheur 7 segments, en supposant que les valeurs d'entrée vont toujours de 0000 à 1001.

Exercice Multiples de 3 Proposez un circuit qui prendra en entrée un nombre N entre 0 et 15 (codé en binaire évidemment) et dont la sortie indiquera si N est un multiple de 3.

3 Quelques circuits combinatoires

Dans ce chapitre nous étudions brièvement quelques circuits combinatoires qui interviennent dans la réalisation des composants logiques d'un ordinateur.

Nous présenterons l'étude de ces circuits sous une forme commune : la *spécification* énonce rapidement le rôle du circuit, la *fonction de tranfert* indique précisément (en général par des tables de vérité) la valeur des sorties pour toutes les entrées possibles, la *réalisation* montre un des circuits possibles.

3.1 Demi-additionneur

3.1.1 Spécification

C'est un circuit (voir fig. [combi1]) à deux entrées a, b et deux sorties r, s. Les entrées représentent les nombres 0 et 1, et rs est l'expression en binaire de leur somme. (s est la somme, r la retenue).

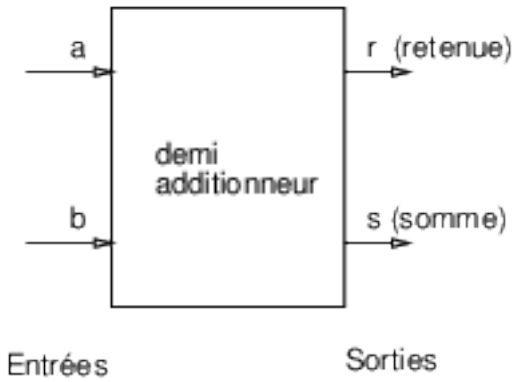


Figure 27: Demi-additionneur

3.1.2 Fonction de transfert

Table

a	b	r	s
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Équations

- $r = a \cdot b$
- $s = a \cdot \neg b + b \cdot \neg a = a \oplus b$

3.1.3 Réalisation:

Le schéma de la figure [combi2] se déduit immédiatement des équations.

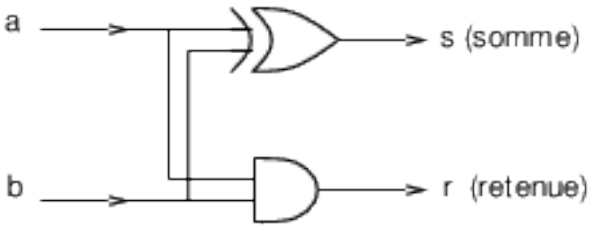


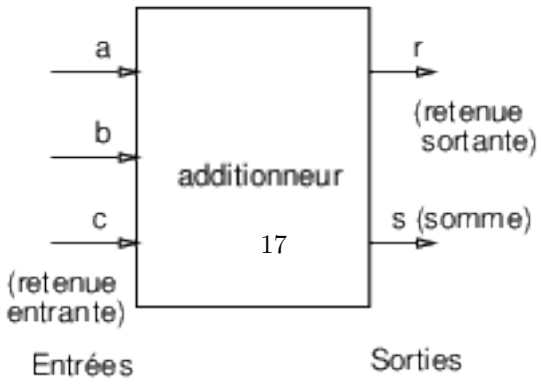
Figure 28: Schéma d'un demi-additionneur

3.2 Additionneur élémentaire

Le demi-additionneur que nous venons de voir ne suffit pas pour réaliser des additions "chiffre par chiffre" parce qu'il produit des retenues, mais ne sait pas les utiliser.

3.2.1 Spécification

L'*additionneur élémentaire* (Fig. [combi3]) est un circuit à 3 entrées a, b, c et deux sorties r, s. Le nombre rs indique, en binaire, le nombre d'entrées qui sont à 1.



a b c	r s
0 0 0	0 0
0 0 1	0 1
0 1 0	0 1
0 1 1	1 0
1 0 0	0 1
1 0 1	1 0
1 1 0	1 0
1 1 1	1 1

Équations :

- $r = a \cdot b + b \cdot c + a \cdot c$
- $s = a \oplus b \oplus c = a \cdot b \cdot c + \neg a \cdot b \cdot c + a \cdot \neg b \cdot c + a \cdot b \cdot \neg c$

Exercice Montrez qu'un additionneur peut être réalisé à partir de trois demi-additionneurs

Exercice Montrez qu'un additionneur peut être réalisé à partir de deux demi-additionneurs et une porte OU.

3.3 Circuit additionneur $2 \times n$ bits

En mettant côte à côte n additionneurs on peut réaliser un additionneur n bits.

3.3.1 Spécification

Circuit à $2n+1$ entrées $(a_0 \dots a_{n-1}, b_0 \dots b_{n-1}, c)$ et $n+1$ sorties $(s_0 \dots s_{n-1}, r)$. Les entrées $a_{n-1}a_{n-2} \dots a_1a_0$ représentent un nombre A codé en binaire (et de même pour B sur les entrées b_j), et c est la retenue entrante. Les sorties $rs_{n-1}sn-2 \dots s_1s_0$ expriment la valeur de $A + B + c$.

3.3.2 Réalisation

Une réalisation directe (par étude des tables de vérité et recherche d'expressions minimales) n'est envisageable que pour de très petites valeurs de n (si n = 3 il y a 7 entrées, donc des tables de vérité à 128 cases ...).

On procède donc par décomposition du problème : il suffit de mettre en parallèle n additionneurs élémentaires identiques. La figure [combi4] montre un additionneur 2×4 bits.

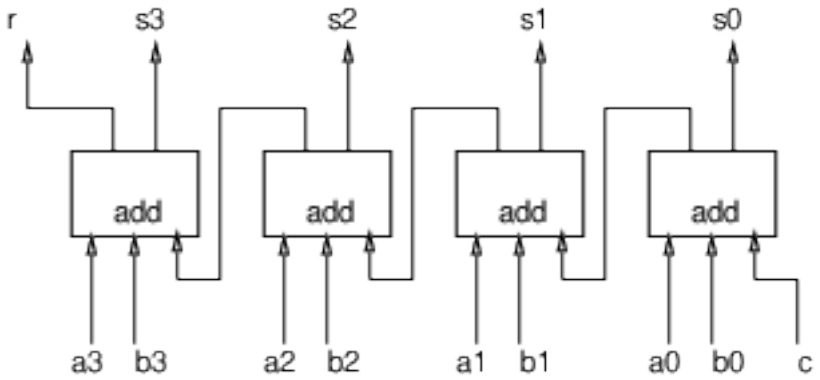


Figure 30: Additionneur en tranches

Remarque Ces additionneurs "en tranches" peuvent eux-mêmes être juxtaposés pour former des additionneurs capables de traiter des nombres plus grands : il suffit de relier la retenue sortante de chaque circuit à la retenue entrante du suivant.

3.4 Décodeur simple

3.4.1 Spécification

Ce circuit possède n entrées $a_0 \dots a_{n-1}$ et 2^n sorties $s_0 \dots s_{2^n-1}$. Les entrées a_i codent un nombre A en binaire. La sortie s_A est mise à 1, les autres à 0.

3.4.1.1 Fonction de transfert Pour un circuit à 2 entrées, $n = 2$, on a la table :

a_1 a_0	s_0 s_1 s_2 s_3
0 0	1 0 0 0
0 1	0 1 0 0
1 0	0 0 1 0
1 1	0 0 0 1

3.4.1.2 Réalisation On voit facilement qu'à chaque sortie correspond un monôme complet sur $a_0, a_1 \dots$, et réciproquement. Voir figure [combi5] pour le schéma d'un décodeur "2 vers 4".

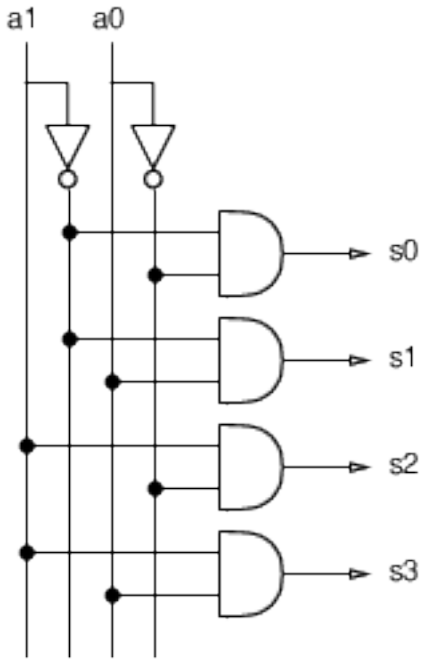


Figure 31: Décodeur "2 vers 4"

Exercice Dessinez un décodeur "3 vers 8"

3.5 Décodeur avec validation

Ce décodeur possède une entrée supplémentaire V, qui sert à "activer" le circuit. Si cette entrée est à 1, le décodeur fonctionne comme précédemment. Si $V = 0$, toutes les sorties sont à 0.

On obtient (fig. [combi6]) ce circuit par une simple modification du schéma précédent.

Remarque Les décodeurs avec validation peuvent être assemblés entre eux pour former des décodeurs plus gros. Le montage de la figure [combi7] montre un décodeur "maître" pilotant 4 décodeurs esclaves, le tout formant un décodeur "4 vers 16" avec validation.

Exercice Quelles doivent être les valeurs des entrées pour avoir $s_{13} = 1$?

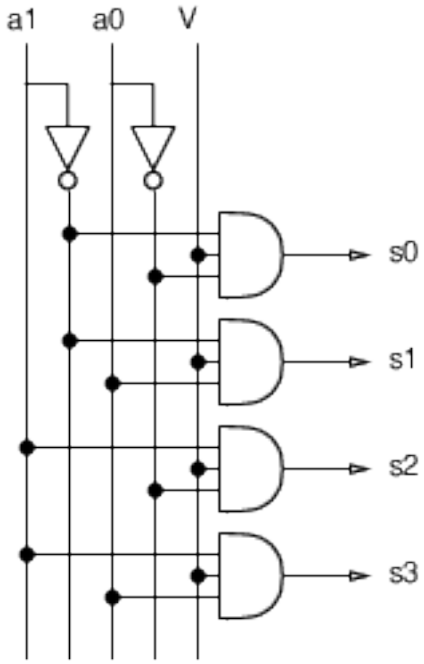


Figure 32: Décodeur “2 vers 4” avec validation

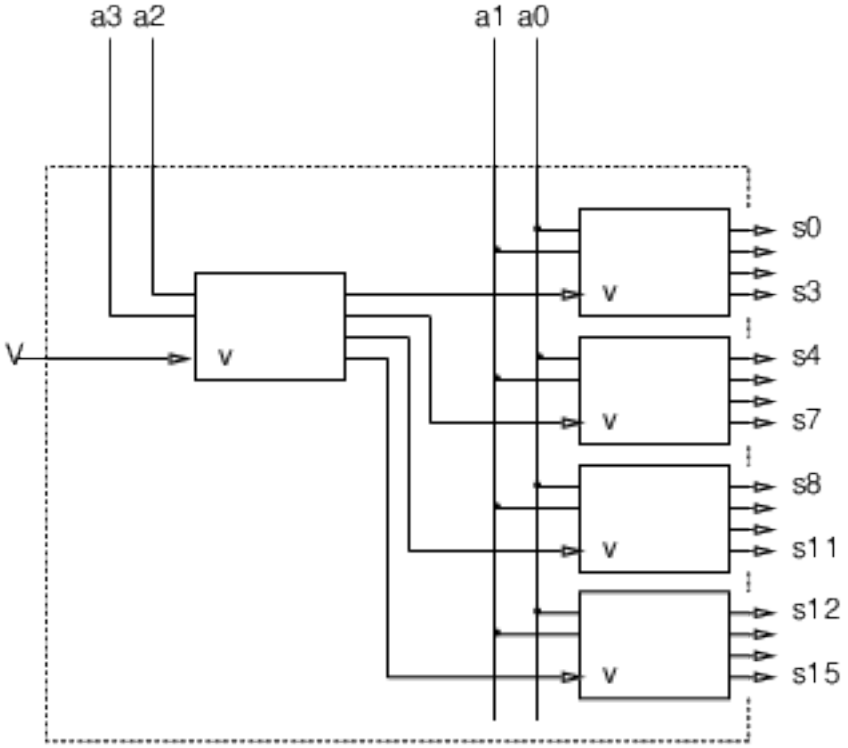


Figure 33: Montage de décodeurs en maître-esclaves

4 Multiplexeur

Autre circuit très utile, le multiplexeur permet de sélectionner une *entrée*, pour la reporter sur la sortie. C'est un circuit à une seule sortie S. Il prend comme entrées un nombre N sur k bits, et 2^k entrées e_i. La valeur de la N-ième entrée est copiée sur la sortie S.

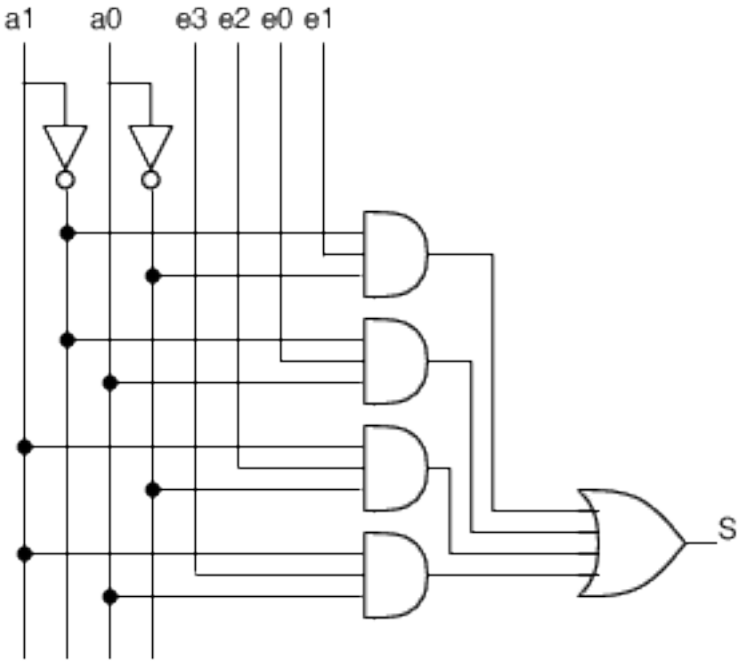


Figure 34: Multiplexeur 4 voies

La figure [combi8] montre un multiplexeur 4 voies. On remarquera que le multiplexeur contient un décodeur 2 vers 4.

> **Exercice Multiplexeur 16 voies** Montrez comment réaliser un multiplexeur 16 voies par un montage maître-esclaves.

4.1 Exercices

4.1.1 Test d'égalité

Concevoir un circuit qui prendra en entrée deux nombres binaires A et B de 4 chiffres, et indiquera si ces deux nombres sont égaux.

4.1.2 Comparateur

Concevoir un circuit qui prendra en entrée deux nombres binaires A et B de 4 chiffres, et indiquera si $A < B$, $A = B$ ou $A > B$. Essayez d'obtenir un circuit "cascadable" pour comparer des nombres plus grands.

4.1.3 Additionneur à retenue anticipée

Dans l'additionneur classique la retenue se propage de chiffre en chiffre, ce qui induit un délai de calcul proportionnel au nombre N de bits des nombres à traiter. Evaluez ce délai en fonction de N et du temps de traversée des portes ET, NON, OU.

La première retenue sortante r_0 vaut $a_0b_0 + a_0c_0 + b_0c_0$ (c_0 est la première retenue entrante). Pour la seconde retenue on a $r_1 = a_1b_1 + a_1c_1 + b_1c_1$. Comme $c_1 = r = 0$, on peut donc exprimer r_1 en fonction de a_0, a_1, b_0, b_1, c_0 (donnez la formule). C'est ce qu'on appelle le *pré-calcul* de r_1. Or cette formule s'exprime avec deux étages de portes (faites le schéma).

En suivant la même technique, donnez une formule pour r_2, puis r_3. Que dire du temps de traversée d'un tel additionneur N bits ? Quelle est la contrepartie de ce gain de temps ?

4.1.4 Compteur

Réaliser un circuit à 3 entrées e_1, e_2, e_3 et 2 sorties s_0, s_1. Le nombre binaire $s_1 s_0$ indiquera le nombre d'entrées qui sont à 1.

4.1.5 Encodeur de priorités

Réaliser un circuit à 3 entrées e_1, e_2, e_3 et 2 sorties s_0, s_1. Le nombre binaire $s_1 s_0$ indiquera le plus grand indice des entrées qui sont à 1, ou 00 si toutes les entrées sont à 0.

5 Circuits séquentiels

5.1 Définition

Un circuit séquentiel possède des entrées E et des sorties S, mais aussi un état interne Q qui sert à mémoriser de l'information. On peut résumer un tel circuit par deux fonctions. L'une (fonction de sortie f) indique la valeur des sorties en fonction des entrées et de l'état actuel :

$$S = f(E, Q)$$

L'autre (fonction de transition g) indique l'état que le circuit prendra à l'instant suivant:

$$Q' = g(E, Q)$$

5.2 Du bistable à la bascule RS

Une expérience simple consiste à monter (fig. [seq1]) deux portes NON tête-bêche, la sortie de l'une servant d'entrée à l'autre. Soient Q_1 et Q_2 les deux sorties.

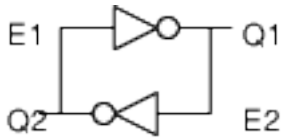


Figure 35: Bistable à deux portes NON

On constate que ce circuit prend l'un des deux eétats suivants:

- état (0) : $Q_1 = 0$ et $Q_2 = 1$
- état (1) : $Q_1 = 1$ et $Q_2 = 0$

et qu'il y reste indéfiniment: on dit que le circuit est *stable*. (C'est un *bistable*, car il y a deux *positions de stabilité*).

En revanche si on boucle l'entrée avec la sortie d'une seule porte NON, (fig. [seq2]) c'est un circuit *astable* qui oscille continuellement (et très rapidement) entre 0 et 1.

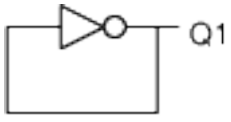


Figure 36: Circuit astable

Reprenons notre bistable à 2 portes NON. Supposons qu'il soit dans l'état (0), c'est-à-dire $Q_1 = 0, Q_2 = 1$. Pour le faire basculer dans l'autre état, il faudrait forcer l'entrée E2 à 1. Mais pour l'instant l'entrée E2 n'est reliée qu'à Q_1. On intercale donc (fig. [seq3]) une porte OU avec une entrée de commande S (Set).

Dans l'état (0) avec S=0, la situation est stable. L'arrivée d'une impulsion sur S provoque le passage de E2 à 1, puis Q_2 prend la valeur 0 et Q_1 passe à 0. Le bistable est donc dans la situation (1) qui est stable, et il y reste désormais quel que soit le signal d'entrée sur S. Ceci est résumé par le chronogramme de la figure[seq4].

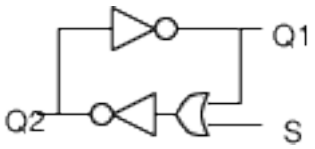


Figure 37: Bistable avec commande S (set)

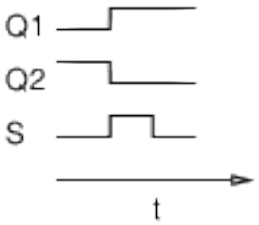


Figure 38: Chronogramme

Pour permettre le retour à l'état (0) on peut ajouter - par symétrie - une autre entrée de commande R (reset) et une autre porte OU (fig. [seq5]).

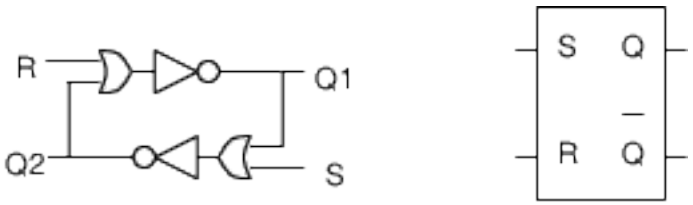


Figure 39: Bascule RS: schéma et symbole

Le circuit résultant (que l'on réalise avec des portes NOR) est appelé *bascule RS*. Son fonctionnement normal se résume comme suit:

- 2 entrées R et S
- 2 sorties Q_1 et Q_2 (appelées aussi Q et $\neg Q$)
- R et S ne doivent pas être à 1 simultanément.
- une impulsion sur S met Q_1 à 1 et Q_2 à 0
- une impulsion sur R met Q_1 à 0 et Q_2 à 1

La figure [seq6] montre, sous forme de chronogramme, l'effet d'une séquence de "tops" envoyés successivement sur les entrées de commande d'une bascule RS.

Remarque

Si R et S sont simultanément à 1 les deux sorties sont à 1. Si S et R repassent simultanément à 0 le résultat est alors imprévisible (dépend des vitesses de réaction des différentes portes).

5.3 Bascules dérivées

5.3.1 Bascule RS à portes NAND

En mettant des portes NAND au lieu des portes NOR, on obtient une bascule RS à signaux de commande inversés. On désigne alors les entrées de commande par $\neg R$ et $\neg S$ (voir figure [seq7]).

5.3.2 Bascule RS avec horloge

Les commandes R et S ne sont actives que lorsque l'entrée d'horloge est à 1. Il suffit (fig. [seq8]) de valider les signaux R et S par l'entrée H au moyen de deux portes ET.

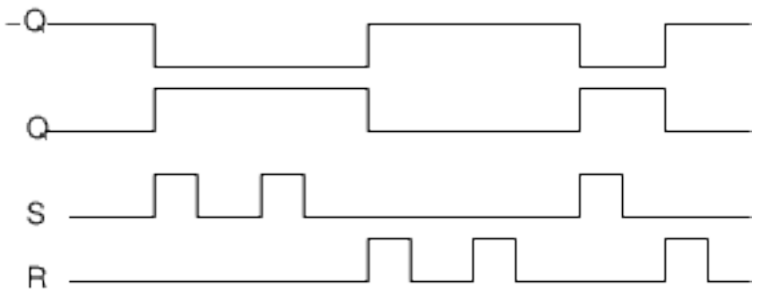


Figure 40: Chronogramme d'une bascule RS

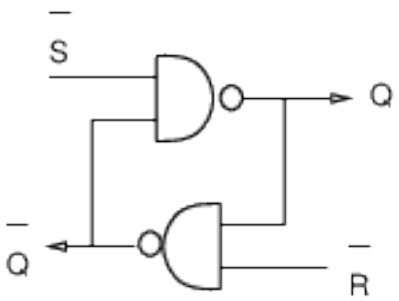


Figure 41: Bascule RS à portes NAND

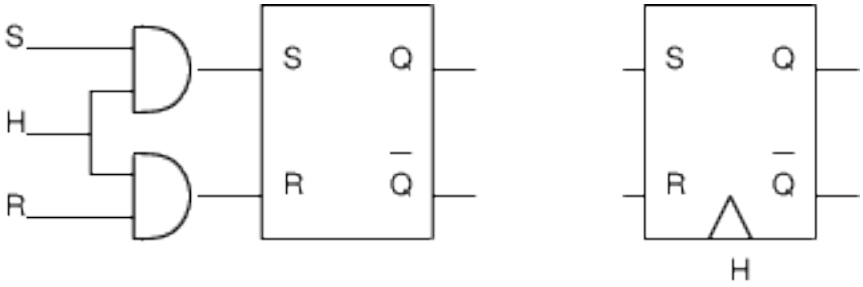


Figure 42: Bascule RSH: schéma et symbole

5.3.3 Bascule D

On la réalise à partir de la bascule RSH (figure [seq9]) Lorsque son entrée H est à 1, elle mémorise la valeur de son entrée D. Quand H=0, elle reste insensible aux changements sur D (voir chronogramme de la figure [seq10])

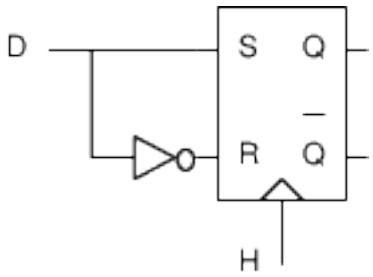


Figure 43: Bascule D

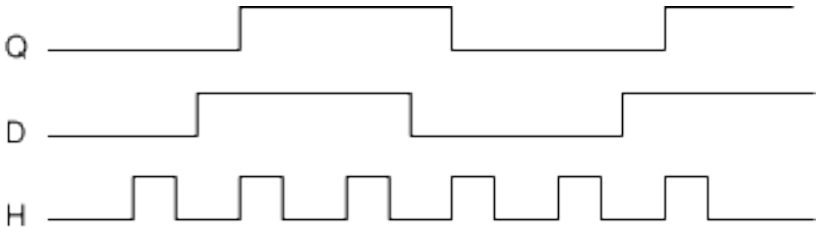


Figure 44: Chronogramme d'une Bascule D

La bascule D est utilisée pour la constitution des *mémoires*.

5.4 La conception de circuits séquentiels

Considérons par exemple une commande d'éclairage par bouton-poussoir. Il y a une entrée B (l'état du bouton : appuyé ou pas), une sortie L (la lampe allumée ou éteinte), mais il est clair que la sortie ne dépend pas seulement de l'entrée, mais aussi de ce qui s'est passé avant, que l'on représente par un état Q.

Ici on pourra décider que Q mémorise simplement l'état de la lampe. Si Q=0, la lampe est éteinte. Si Q=1, la lampe est allumée

La sortie est fonction de l'entrée et de l'état

$$L = f(B, Q)$$

Ici on prendra par exemple $L = Q$.

Une autre fonction indique comment l'état interne évolue au cours du temps. Pour simplifier le raisonnement on découpe le temps en "instants". L'état Q' "à l'instant suivant" dépend des entrées ainsi que de l'ancien état:

$$Q' = g(B, Q)$$

Dans l'exemple on dira que si B=0 l'état reste stable et si B=1 l'état s'inverse, d'où la table:

bouton B	état Q	état suivant $Q' = g(B, Q)$
0	0	0
0	1	1
1	0	1
1	1	0

A partir de là on pourrait en conclure qu'il suffit (fig. [seq11]) d'une bascule D pour représenter l'état, qui serait activée par B et dont la sortie serait rebouclée sur l'entrée à travers un NOT.

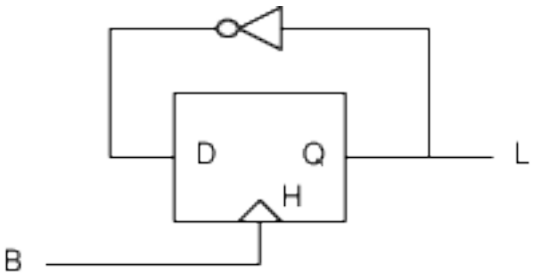


Figure 45: Un montage naïf

Mais attention ça ne marche pas ! En effet l'impulsion sur le bouton dure "plusieurs instants", ce qui fait clignoter (trop vite pour que l'oeil le perçoive) la sortie pendant l'appui du bouton : c'est un état instable. La vitesse élevée d'oscillation fait alors qu'à la fin de l'impulsion le résultat est imprévisible.

Pour s'en sortir on emploiera le montage de la figure [seq12], (appelé maître-esclave).

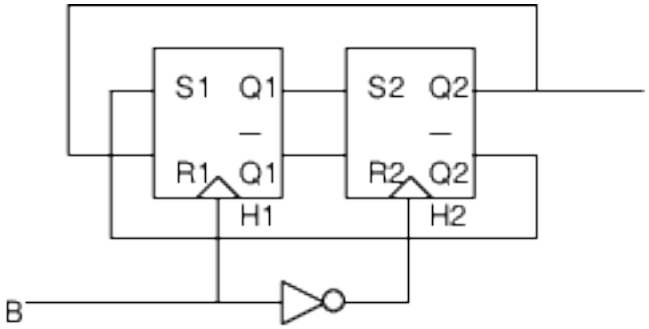


Figure 46: Bascule RSH maître-esclave

la bascule 1 (maître) est validée par $H_1 = B$ et la 2 (esclave) par $H_2 = \neg B$, mais on utilise (et c'est là toute l'astuce) une porte NON dont le seuil de déclenchement est plus bas que la normale, ce qui fait qu'une impulsion sur B correspond à deux impulsions décalées sur H_1 et H_2, comme le montre le chronogramme de la figure [seq13].

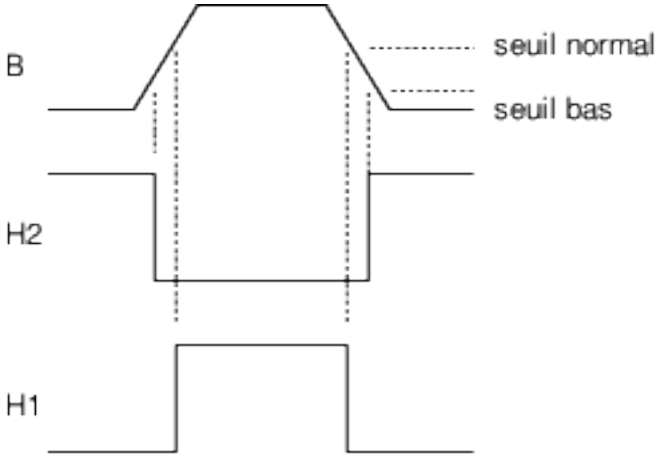


Figure 47: Seuils de déclenchement

Donc à l'arrivée d'une impulsion (front montant) la bascule esclave se verrouille en mémorisant l'état du maître, puis le maître change d'état. À la fin de l'impulsion (front descendant de l'impulsion sur B) le maître se verrouille, et l'esclave change d'état.

> **Exercice** : vérifier le fonctionnement de ce circuit à l'aide d'un chronogramme (dessiner l'arrivée de 2 impulsions sur B).

Problème Bascule JK En utilisant le même principe (maître-esclave) concevoir une bascule JK. Comme son nom l'indique une bascule JK possède 3 entrées J,K et H et deux sorties Q et $\neg Q$. La bascule ne peut changer d'état que pendant une impulsion sur H. Si J=K=0 l'état reste inchangé. Si J=1 K=0 l'état devient 1. Si J=0 et K=1 l'état devient 0. Si J=K=1 l'état s'inverse.

5.5 Circuits Synchrones

On appelle *circuit synchrone* un circuit séquentiel dont l'état interne ne change qu'à l'arrivée d'une impulsion sur une entrée spéciale appelée "Horloge". On suppose en général que les autres signaux ne varient pas pendant l'arrivée d'un top d'horloge.

L'intérêt de cette horloge est de donner une définition précise de "l'instant suivant". C'est l'horloge qui rythme les transitions d'état.

5.6 Application à la synthèse de compteurs

On appelle *compteur* un dispositif séquentiel qui passe périodiquement par une suite déterminée d'états.

Par exemple le *compteur binaire* sur deux bits suivra la séquence $00, 01, 10, 11, 00, 01, 10, 11, 00$, etc. à chaque top d'horloge.

Nous allons voir comment réaliser, *de manière systématique* (et donc automatisable !), n'importe quel compteur, d'abord avec des bascules D maître-esclave, puis avec des bascules JK qui permettent d'obtenir des circuits plus simples.

5.6.1 Réalisation à l'aide de bascules D

5.6.1.1 Exemple compteur binaire 2 bits

Les deux bits de l'état sont stockés dans deux bascules D.

- il n'y a pas de signal en entrée
- les sorties Q_1, Q_0 sont issues directement des bascules D
- pour réaliser la fonction de transition, il suffit de présenter sur les entrées D1 et D0 la valeur du prochain état que l'on calculera à partir de Q_1 et Q_0. Pour cela on construit la table:

$Q_1 Q_0$	$D_1 D_0$
0 0	0 1
0 1	1 0
1 0	1 1
1 1	0 0

D'où on tire facilement :

- $D_1 = Q_1 \oplus Q_0$
- $D_0 = \neg Q_0$

5.6.2 Réalisation à l'aide de bascules JK

Avec les bascules D, nous exprimions le nouvel état en fonction de l'ancient. Avec des bascules JK, le principe est légèrement différent : nous déterminons les *commandes* à envoyer aux bascules (sur les entrées J et K) pour *modifier* l'état.

Avant de reprendre l'exemple du compteur 2 bits, quelques remarques sur les bascules JK. Regardons la table de transition qui montre l'état Q' obtenu quand, dans l'état Q, la bascule reçoit des commandes J et K :

Q	JK	Q'
0	0 0	0
0	0 1	0
0	1 0	1
0	1 1	1
1	0 0	1
1	0 1	0
1	1 0	1

Q	JK	Q'
1	1 1	1

On voit qu'il y a deux façons de passer d'un état à un autre. Par exemple pour passer de 0 à 1, on a J=1 et K=0 (forcer à 1), et aussi J=1 et K=1 (inverser).

On en tire des conditions nécessaires et suffisantes pour passer d'un état à un autre

de	à	si et seulement si
0	0	J = 0 (ne pas monter)
0	1	J = 1 (monter)
1	0	K = 1 (descendre)
1	1	K = 0 (ne pas descendre)

On s'en sert pour construire le compteur 2 bits à partir de 2 bascules JK.

Q_1Q_0	$Q_1'Q_0'$	commandes
0 0	0 1	$J_1 = 0, J_0 = 1$
0 1	1 0	$J_1 = 1, K_0 = 1$
1 0	1 1	$K_1 = 0, J_0 = 1$
1 1	1 0	$K_1 = 0, K_0 = 1$

Il ne reste plus qu'à trouver des expressions convenables pour J_0, K_0, J_1 et K_1 en fonction de Q_1 et Q_0.

En reportant dans un tableaux de Karnaugh pour J_1

$J_1(Q_1, Q_0)$	$Q_0 = 0$	$Q_0 = 1$
$Q_1 = 0$	0	1
$Q_1 = 1$	.	.

on obtient $J_1 = Q_1$, par regroupement sur la colonne de droite.

Pour J_0 :

$J_0(Q_1, Q_0)$	$Q_0 = 0$	$Q_0 = 1$
$Q_1 = 0$	1	.
$Q_1 = 1$	1	.

on voit qu'il suffit donc de prendre $J_0 = 1$.

De même, l'abondance de cas indéterminés conduit à des expressions très simples pour les commandes K :

- $K_1 = Q_1$
- $K_0 = 1$

5.7 Exercices

Compteur modulo 3 La séquence à effectuer est $00, 01, 10, 00, 01, 10$, etc.

compteur binaire sur 3 bits : séquence $000, 001, 010, 011, ...111, 000, ...$

Compteur modulo 10 : 0000,0001,. . . .,1000,1001,0000. . .

Compteur/décompteur 3 bits Une entrée supplémentaire C > indique s'il faut compter (C=1) ou décompter (C=0).

**Chenillard sur 4 bits : 0000 1000 0100 0010 0001 0000 ....

Compteurs rampants 0000, 1000, 1100, 1110, 1111, 0111, 0011, 0001, 0000, ...

Compteurs de Gray

- sur 1 bit: 0 1 0 1...
- sur 2 bits: 00 01 11 10 00 ....
- sur 3 bits: 000 001 011 010 110 111 101 100 ...