TD 1
Portes de base et minimisations de fonctions booléennes

Préparation
Ex. 1 : Codage des entiers naturels

Question 1 Remplir un tableau contenant les valeurs en base 2, en base 10 et en base 16 des entiers naturels entre 0 et 15 (inclus).

Question 2 Comment peut-on calculer facilement une valeur en hexadécimal à partir de sa valeur en binaire ? Convertir en hexadécimal la valeur binaire 101101011100_2.

Pour aller plus loin...
Question 3 Convertir en héxadécimal la valeur binaire 1110011010_2. À quoi faut-il faire attention ?

Question 4 Faire les additions suivantes en binaire : $44 + 21$ et $200 + 100$ sur 8 bits [1]. Que constate-t-on pour le résultat de la 2e addition ? Comment peut-on détecter ce phénomène ?

Question 5 Quel intervalle d'entiers naturels peut-on coder sur n bits ? Combien de bits faut-il pour coder m valeurs différentes (avec $m \geq 2$) ?

Préparation
Ex. 2 : Circuit comparateur d'entiers

Dans cet exercice, on travaille sur des entiers A et B codés sur 4 bits, ce qu'on notera $A = a_3a_2a_1a_0$ et $B = b_3b_2b_1b_0$. Pour construire les circuits demandés, on suppose qu'on dispose de portes avec au maximum 4 entrées.

Question 1 Quelle porte élémentaire produit 1 en sortie ssi ses deux entrées sont égales ?

Question 2 Construire un circuit prenant en entrée 2 entiers A et B et produisant une sortie eg valant 1 ssi ces 2 entiers sont égaux.

Question 3 Etendre ce circuit pour ajouter une sortie z valant 1 ssi A vaut 0.

Question 4 Ajouter une sortie imp valant 1 ssi le nombre de bits valant 1 dans B est impair. Par exemple, si $B = 5_{10} = 0101_2$ alors $imp = 0$ et si $B = 14_{10} = 1110_2$ alors $imp = 1$.

Méthodologie
Ex. 3 : Conception et minimisation de circuits combinatoires sur un exemple élémentaire

On travaille sur la fonction majorité de trois variables valant "vrai" ssi la majorité des paramètres de la fonction valent 1.

Table de vérité de la fonction Majorité

1. C'est à dire que les opérandes et le résultat de l'opération sont codés sur 8 bits.

a	b	c	$Maj(a,b,c)$	termes canoniques
0	0	0	0	
0	0	1	0	
0	1	0	0	
0	1	1	1	$\overline{a}.b.c$
1	0	0	0	
1	0	1	1	$a.\overline{b}.c$
1	1	0	1	$a.b.\overline{c}$
1	1	1	1	a.b.c

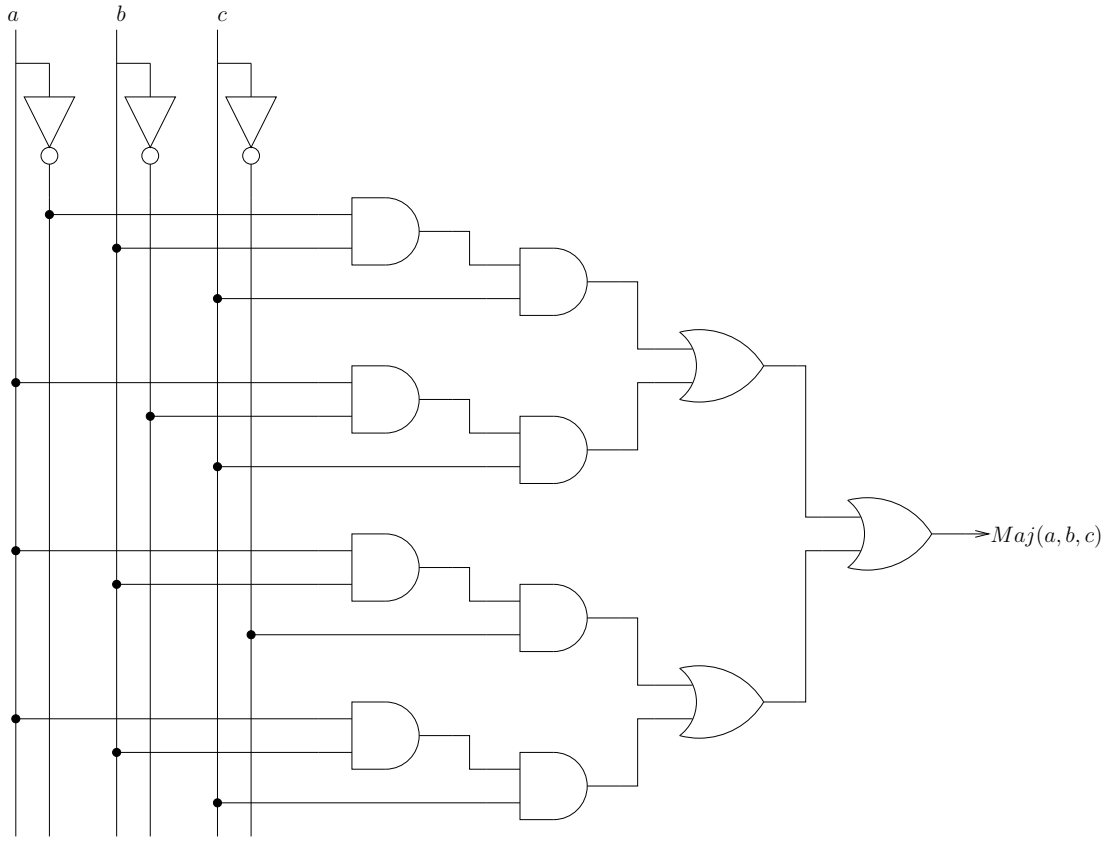
N.B : un "terme canonique" est le produit (AND) de toutes les variables de la fonction sous forme normale (a) ou complémentée ($\overline{a}$). On recherche une expression somme de produits donc on ne s'intéresse qu'aux termes correspondants aux 1 de la fonction.

Expression booléenne canonique de la fonction $Maj(a,b,c)$

C'est l'expression de la fonction $Maj(a,b,c)$ en fonction des paramètres a, b et c, sans chercher à minimiser cette expression. C'est donc la somme des termes canoniques déduits de la table de vérité :
$Maj(a,b,c) = \overline{a}.b.c + a.\overline{b}.c + a.b.\overline{c} + a.b.c.$

Circuit implantant cette fonction

Pour dessiner le circuit implantant cette fonction, on n'utilisera ici que des portes de bases AND, OR à 2 entrées ainsi que des portes INV.



Minimisation de l'expression de la fonction

On montre ici comment on peut minimiser l'expression d'une fonction en utilisant le calcul booléen. On part de :

$$Maj(a,b,c) = \overline{a}.b.c + a.\overline{b}.c + a.b.\overline{c} + a.b.c$$

On remarque que les simplifications suivantes sont possibles, grâce à la règle $\overline{x}.y + x.y = y$:

$$\overline{a}.b.c + a.b.c = b.c$$

$$a.\overline{b}.c + a.b.c = a.c$$

$$a.b.\overline{c} + a.b.c = a.b$$

Faut-il choisir une (et une seule) de ces simplifications ? Non, car on a aussi la règle $x + x = x$. On peut donc réécrire l'expression de la fonction Majorité comme ci-dessous :

$$\begin{aligned} Maj(a,b,c) &= \overline{a}.b.c + a.\overline{b}.c + a.b.\overline{c} + a.b.c + a.b.c + a.b.c \\ &= (\overline{a}.b.c + a.b.c) + (a.\overline{b}.c + a.b.c) + (a.b.\overline{c} + a.b.c) \\ &= b.c + a.c + a.b \end{aligned}$$

Tableaux de Karnaugh :

Une technique graphique pour la minimisation d'expressions booléennes

Un tableau de Karnaugh est une table de vérité, présentée de façon à ce que les adjacences du type $\overline{x}.y + x.y = y$ soient mises en évidence.

Le tableau de Karnaugh de la fonction $Maj(a,b,c)$ est donné ci-dessous. On remarque que l'ordre d'énumération des variables b et c n'est pas quelconque : une seule des variables change de valeur entre deux colonnes (y compris quand on rapproche la dernière colonne de la première : adjacence circulaire).

Remplir un tableau de Karnaugh revient à remplir une table de vérité. Par exemple, la troisième case de la première ligne correspond à $a = 0$ et $b = 1$ et $c = 1$: $Maj(0,1,1) = 1$ donc la case est remplie avec un 1.

a \ bc	**00**	**01**	**11**	**10**
0	0	0	1	0
1	0	1	1	1

Les groupements de points à 1 mettent en évidence les simplifications possibles. Il suffit alors de choisir un nombre minimal de regroupements qui couvrent tous les points à 1 de la fonction.

Pour chaque groupe de cases à 1, on extrait un terme construit à partir des variables qui ne changent pas de valeur pour l'ensemble des points regroupés. Par exemple, pour le groupement vertical de la troisième colonne : on remarque que la valeur de a change entre les deux cases regroupées, a n'apparaîtra donc pas dans le terme correspondant à ce groupe. Ce qui est commun dans ce groupe ce sont les valeurs de b et c qui valent 1 : on extrait donc le terme $b.c$

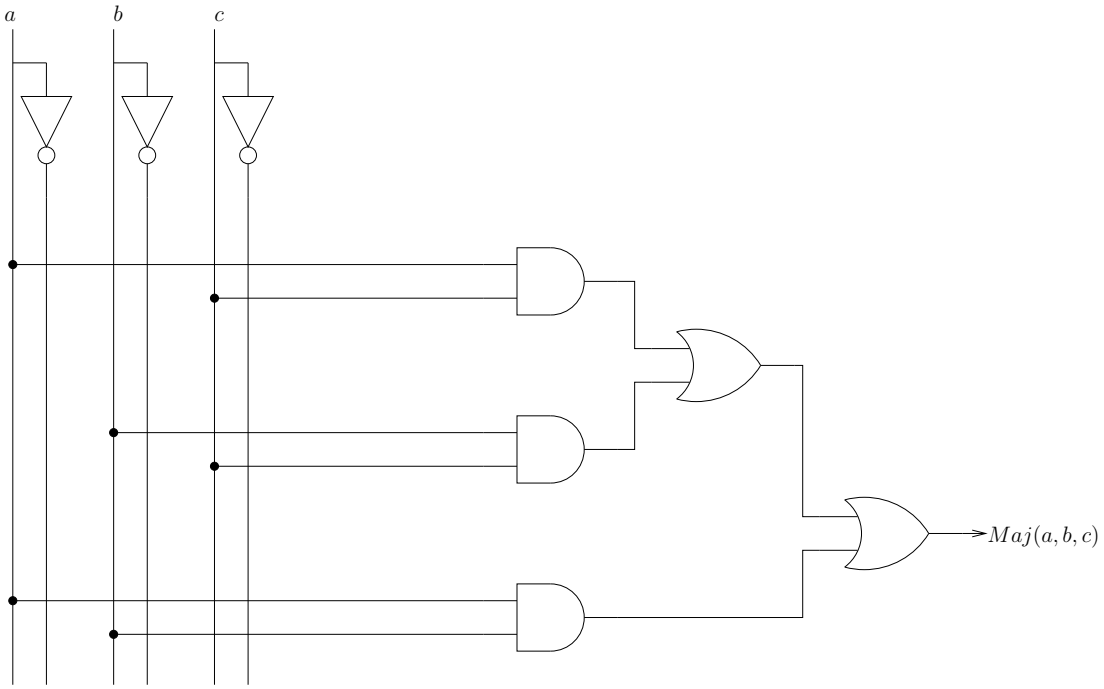
Sur cet exemple, il y a donc 3 termes qui correspondent aux trois groupements, on retrouve l'expression : $Maj(a,b,c) = b.c + a.c + a.b$.

N.B : Les groupements ne peuvent être que de 2, 4 ou 8 cases (puissances de 2). Par la suite, on utilisera souvent des tableaux avec 4 variables en entrées donc 16 cases. Le nombre de variables dans un terme correspond à la taille du groupement. Par exemple, dans un tableau de

16 cases (4 variables), un groupe de 8 cases correspond à un terme d'une variable, un groupe de 4 à un terme de 2 variables, un groupe de 2 à un terme de 3 variables et s'il reste un 1 isolé le terme correspondant utilise les 4 variables.

Circuit minimisé

A partir de l'expression minimisée, on peut dessiner le circuit suivant :



Pourquoi a t on besoin de minimiser ?

On remarque que le circuit obtenu à partir de la forme minimisée a deux avantages : moins de portes au total (5 au lieu de 11) et moins de portes traversées (entre l'entrée et la sortie, au plus 4 portes au lieu de 5). Ce circuit minimisé est donc moins coûteux en portes et plus rapide.

N.B : Chaque porte a un temps de propagation (différent suivant la technologie). Le temps de propagation à travers un circuit est déterminé par le nombre maximal de couches de portes à traverser entre une entrée et une sortie.

Remarque : dans la suite du semestre, nous n'utiliserons que les minimisations de fonctions booléennes à base de tableaux de Karnaugh.

Ex. 4 : Reste de la division entière

Soit un entier naturel E dans l'intervalle [0 .. 15], donc codé sur 4 bits e_3, e_2, e_1, e_0 ; on veut réaliser un circuit qui calcule le reste S de la division entière de E par 7.

Question 1 Sur combien de bits doit être codé S ?

Question 2 Représenter les fonctions de sortie du circuit à l'aide de tableaux de Karnaugh.

Question 3 Proposer des expressions minimisées des fonctions de sortie du circuit.

On considère maintenant que l'entier E en entrée est dans l'intervalle [0 .. 9]. Les fonctions en sortie sont donc des fonctions Φ - booléennes.

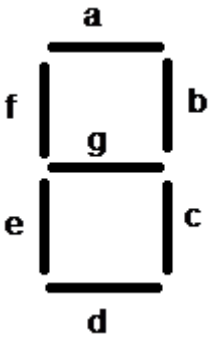
Méthodologie : Pour toutes les cases qui correspondent à une valeur d'entrées non utilisée (ici, les cases correspondant aux valeurs de 10 à 15), la valeur de la sortie est indifférente. On note Φ dans les cases correspondantes : Φ veut dire 0 ou 1, au choix du concepteur. On pourra donc utiliser les Φ pour simplifier davantage l'expression de la fonction (s'ils permettent d'obtenir de plus gros groupements de cases). On utilise le symbole Φ car il représente la superposition d'un 1 et d'un 0.

Question 4 Modifier les tableaux de Karnaugh obtenus en question 2 et proposer des expressions minimisées des fonctions de sortie du circuit.

Pour aller plus loin...
Question 5 Dessiner le circuit à base de portes AND, OR et INV à partir des équations de la question 3. Peut-on diminuer la complexité de ce circuit en utilisant un même groupement pour plusieurs sorties ?

Ex. 5 : Afficheur 7-segments

Un afficheur 7-segments est un dispositif d'affichage d'un chiffre décimal composé, comme son nom l'indique, de sept segments pouvant être allumés ou éteints indépendamment les uns des autres. Les segments sont disposés comme illustré ci-dessous :



On considère que l'affichage est réalisé comme suit :

Le but de cet exercice est de concevoir un circuit prenant en entrée un chiffre décimal et produisant en sortie les sept sorties booléennes correspondant aux segments de l'afficheur.

Question 1 Sur combien de bits doit être codée l'entrée ? Quelle politique adopter concernant les valeurs superflues ?

Question 2 En utilisant des tableaux de Karnaugh, donner les expressions minimisées des 7 segments.

Pour aller plus loin...
Question 3 Dessiner le circuit correspondant en utilisant des portes logiques de base.

Pour aller plus loin...

Ex. 6 : Transcodeur en code de Gray

On appelle code de Gray [2] (ou codage binaire réfléchi) un codage binaire ordonné dans lequel le codage ne change que d'un bit entre deux valeurs successives. Le code de Gray sur 2 ou 3 bits est utilisé dans les tableaux de Karnaugh. Soit donc le code de Gray sur 4 bits suivant :

Décimal	Gray
0	0000
1	0001
2	0011
3	0010

Décimal	Gray
4	0110
5	0111
6	0101
7	0100

Décimal	Gray
8	1100
9	1101
10	1111
11	1110

Décimal	Gray
12	1010
13	1011
14	1001
15	1000

On veut réaliser un circuit prenant en entrée un entier naturel codé sur 4 bits e_3, e_2, e_1, e_0 et produisant en sortie le code de Gray correspondant s_3, s_2, s_1, s_0.

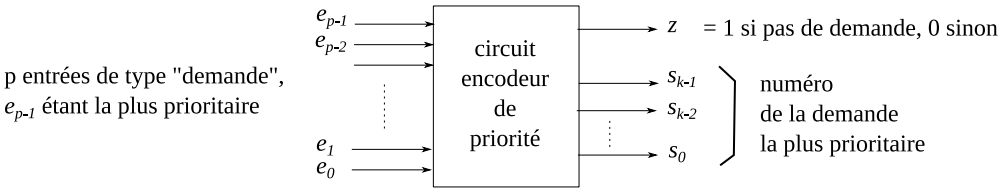
Question 1 Déterminer les équations minimisées des sorties à l'aide de tableaux de Karnaugh.

Question 2 Dessiner le circuit correspondant en utilisant des portes logiques de base.

Question 3 Pour apprendre à repérer les expressions à base de XOR, faire le tableau de Karnaugh de $e_3 \oplus e_2 \oplus e_1 \oplus e_0$; de $e_3.(e_2 \oplus e_1 \oplus e_0)$.

Pour aller plus loin...

Ex. 7 : Encodeur de priorité



Un encodeur de priorité est un circuit qui donne en sortie le numéro (codé en binaire) de l'entrée active la plus prioritaire. Par exemple, si seules les entrées e_5 et e_8 sont à 1, la sortie (bits s_{k-1} à s_0) codera 8 en binaire. Ce type de circuits est par exemple utilisé pour gérer les interruptions d'un processeur (cf. cours de CEP).

Question 1 Si $p = 2^n$, déterminer le nombre de bits en sortie (la valeur de k). Donner une description formelle du circuit.

Question 2 Dans le cas $n = 2$, déterminer les équations minimisées de z et $S = s_1 s_0$ en remplissant une table de vérité partielle (*i.e.* réfléchir sur les lignes pouvant être factorisées dans la table).

Question 3 Dessiner le circuit correspondant en utilisant des portes logiques de base.

2. Frank Gray, Bell Labs, 1953