Université de Paris-Saclay
Faculté des sciences d'Orsay
L3 E3A

Électronique numérique

UE-308

TD2 : Logique et circuits combinatoires

Exercice 1 : synthèse d'une fonction logique à partir d'un cahier des charges

On se propose d'étudier le problème propositionnel suivant : un employé est affecté à la surveillance d'une expérience. Il a alors à sa disposition 3 lampes (Verte, Orange et Rouge) lui indiquant l'état de cette expérience. Il devra immédiatement prévenir son responsable si l'une des lampes Orange ou Rouge s'allume. Si par contre la lampe verte s'allume avec l'une des deux autres, cela indique une fausse alerte et donc qu'il n'y a pas lieu de s'inquiéter. Il est impossible que les lampes Orange et Rouge s'allument en même temps. Nous souhaitons remplacer cet employé par un système automatique. A l'aide de la méthode de votre choix et en utilisant des notations claires, donnez la fonction logique permettant de résoudre ce problème.

Exercice 2 : simplifications/manipulations de fonctions logiques

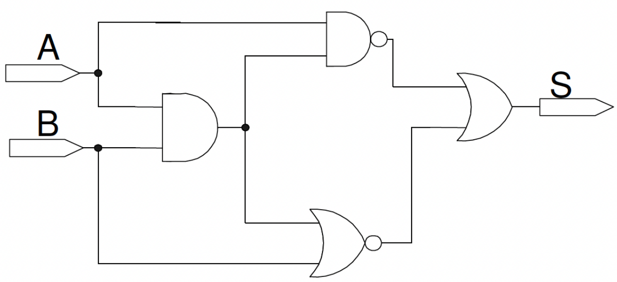
1. A partir d'une table de vérité, démontrer la relation $a.(b+c) = a.b + a.c$

2. A partir des règles de l'algèbre de Boole, démontrer le théorème du consensus $a.b + \bar{a}.c + b.c = a.b + \bar{a}.c$ en utilisant judicieusement le principe du tiers exclu sur la variable a.

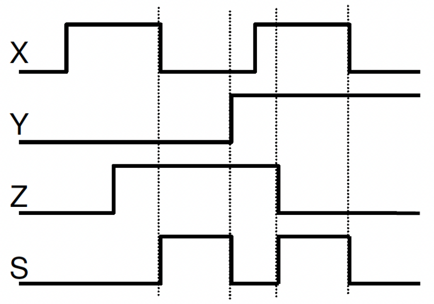
3. A partir des règles de l'algèbre de Boole, simplifier les relations suivantes : $\overline{\overline{a+b}+\overline{c+d}}$; $\overline{\overline{a.b}+c.d}$; $\overline{a.\overline{b}.(a+c)} + \overline{a}.b.\left(\overline{a+\overline{b}+\overline{c}}\right)$ et $(a \oplus b) \oplus b$.

4. A partir des règles de l'algèbre de Boole, écrire la fonction $a+\overline{b}.c$ sous forme de mintermes en utilisant judicieusement le principe du tiers exclu sur les variables a, b et c. Montrer que l'on peut écrire $a+\overline{b}.c = \left(a+\overline{b}\right).(a+c)$ puis écrire la fonction sous forme de maxtermes en utilisant judicieusement le principe de contradiction.

5. A partir du schéma logique suivant, déduire l'équation combinatoire de la sortie sous le forme la plus simple possible.



6. A partir du chronogramme suivant, déduire l'équation combinatoire de la sortie sous le forme la plus simple possible.



7. A partir d'un tableau de Karnaugh, simplifier les équations logiques suivantes : $s_1 = a.\overline{b} + \overline{a}.c + \overline{a}.\overline{b} + \overline{a}.b.\overline{c}$; $s_2 = \overline{\left(\overline{a+b}\right).c.d + a.\overline{b}.c.d}$.

8. A partir d'un tableau de Karnaugh, simplifier l'équation logique de la sortie donnée par la table de vérité suivante :

#	A	B	C	D	E	S	#	A	B	C	D	E	S
0	0	0	0	0	0	1	**16**	1	0	0	0	0	0
1	0	0	0	0	1	0	**17**	1	0	0	0	1	0
2	0	0	0	1	0	0	**18**	1	0	0	1	0	0
3	0	0	0	1	1	1	**19**	1	0	0	1	1	0
4	0	0	1	0	0	1	**20**	1	0	1	0	0	0
5	0	0	1	0	1	1	**21**	1	0	1	0	1	1
6	0	0	1	1	0	0	**22**	1	0	1	1	0	1
7	0	0	1	1	1	1	**23**	1	0	1	1	1	0
8	0	1	0	0	0	1	**24**	1	1	0	0	0	0
9	0	1	0	0	1	1	**25**	1	1	0	0	1	0
10	0	1	0	1	0	1	**26**	1	1	0	1	0	0
11	0	1	0	1	1	0	**27**	1	1	0	1	1	0
12	0	1	1	0	0	0	**28**	1	1	1	0	0	1
13	0	1	1	0	1	1	**29**	1	1	1	0	1	1
14	0	1	1	1	0	1	**30**	1	1	1	1	0	1
15	0	1	1	1	1	0	**31**	1	1	1	1	1	1

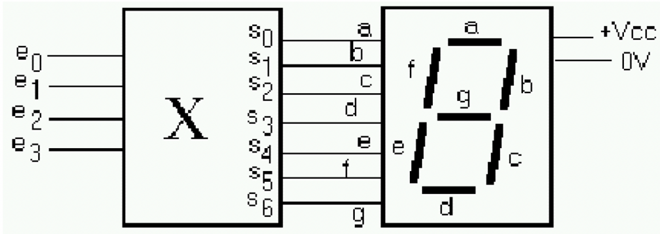
Exercice 3 : prédicat/proposition

On considère la fonction logique F égale à 1 si l'équation binaire $a.c + b = \overline{a} + b.\overline{c}$ est vraie pour un triplet (a, b, c) (0 sinon). Par le méthode de votre choix, donner et simplifier F en fonction des variables a, b et c.

Exercice 4 : circuit d'affichage binaire vers décimal

On souhaite utiliser un afficheur 7 segments pour afficher des caractères à partir d'un code binaire et en respectant une table dite de visualisation. Les $e_{i=0...3}$ sont les signaux d'entrées (4 bits) et les signaux a, b, c, d, e, f, g seront les 7 signaux de sortie qui commandent les 7 segments de l'afficheur (les segments sont actif à 1). La table de correspondance entre le code binaire et le caractère à afficher est la suivante :

e3,e2,e1,e0	a,b,c,d,e,f,g
0 0 0 0	A
0 0 0 1	b
0 0 1 0	c
0 0 1 1	d
0 1 0 0	E
0 1 0 1	F
0 1 1 0	t
0 1 1 1	u
1 0 0 0	0
1 0 0 1	1
1 0 1 0	2



Dire pourquoi cette fonction est incomplète. Donner la table de vérité du système. Simplifier les équations des sorties à l'aide de tableaux de Karnaugh.

Exercice 5 : soustracteur binaire

Pour fabriquer un soustracteur entre deux nombres binaires de même taille, il existe plusieurs méthodes. On peut considérer, par exemple, un additionneur classique mais avec un des deux nombres codé en complément à 2. Ici on va suivre la méthode (plus directe) qui a été proposée pour l'additionneur classique, à savoir, l'étude d'un demi-soustracteur (soustraction sur un bit) puis, par un raisonnement itératif, l'extension au cas de deux nombres de N bits chacun.

1. Coder en complément à 2 (sur 2 bits) les chiffres décimaux signés suivants : $(-1)_{10s}$, $(0)_{10s}$ et $(1)_{10s}$. En déduire la table de vérité du soustracteur $\alpha_0 - \beta_0$ où α_0 et β_0 sont deux « mots » de un bit chacun. Simplifier les équations des sorties et donner le circuit logique du demi-soustracteur.

2. Calculer avec cette méthode (en passant par le binaire naturel) $(17)_{10} - (3)_{10}$ puis $(17)_{10} - (20)_{10}$.

3. Par un raisonnement itératif similaire à la conception d'un circuit additionneur, proposer un schéma détaillé pour le soustracteur complet sur N bits $A - B$ où on posera $A = (\alpha_{N-1}\alpha_{N-2}\cdots\alpha_1\alpha_0)_2$ et $B = (\beta_{N-1}\beta_{N-2}\cdots\beta_1\beta_0)_2$.

Exercice 6 : transcodage binaire vers Gray

En notant $b_{N-1}b_{N-2}\cdots b_1b_0$ un entier écrit en binaire naturel, et de même en notant $g_{N-1}g_{N-2}\cdots g_1g_0$ le code de Gray correspondant, il est possible de montrer que $g_i = b_{i+1} \oplus b_i \ \forall i \in \{0, \ldots, N-2\}$.

1. Comment obtient t'on g_{N-1} ?

2. Montrer que ce résultat est vrai pour $N = 3$.

3. Proposer un circuit qui, pour un nombre en binaire naturel sur N bits en entrée fourni le code équivalent de Gray en sortie.

Exercice 7 : multiplexeurs

Avec l'aide exclusive d'un multiplexeur 8 vers 1, proposer à chaque fois un montage pour réaliser les trois fonctions logiques suivantes : $s_1 = a \oplus b$; $s_2 = a \oplus b \oplus c$ et $s_3 = (a + b) .c + a.b$

Exercice 8 : circuit comparateur binaire

En vous basant sur l'étude du demi additionneur et de l'additionneur complet vu en cours, proposer une mise en oeuvre itérative d'un comparateur de type « $A = B$ » où A et B sont des nombres en binaire naturel (non signé) de même taille N bits chacun, c'est-à-dire un circuit qui renvoi 1 en sortie ssi $A = B$. Construire le même circuit pour un comparateur de type « $A > B$ », c'est-à-dire un circuit qui renvoi 1 en sortie ssi $A > B$.

Exercice 9 : circuit de détection d'erreurs dans une transmission [1]

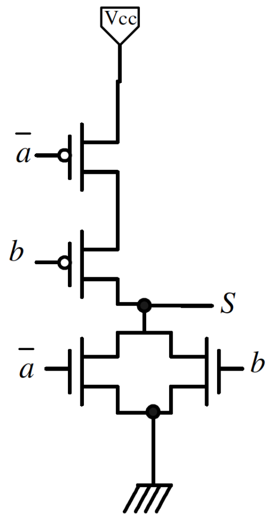
Dans un système de transmission binaire entre un émetteur et un récepteur (par câble par exemple), on sait que la probabilité d'avoir en moyenne une erreur ou plus par paquet de 8 bits est très faible mais malheureusement non nulle. Pour déceler ces erreurs de transmissions, on décide d'ajouter un bit P (pour chaque octet) qui représentera la fonction logique suivante : ce bit P est à 1 si le nombre de bits mis à 1 (c'est à dire parmi les 8 bits) est impair. P est à zéro dans le cas contraire.

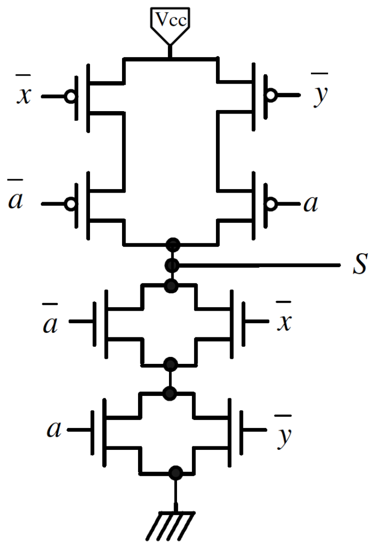
1. En raisonnant de manière itérative (paquets de 2 bits puis 3 bits, etc.), écrire l'équation logique permettant de générer le bit P à transmettre avec les 8 bits utiles.

2. On désire maintenant réaliser la partie détecteur d'erreur sur le récepteur une fois le message transmis. Le mot reçu par ce dernier sera donc composé de 9 bits (8 bits de données utiles et 1 bit de contrôle P_R envoyé par l'émetteur). Le récepteur doit alors produire un signal $ERREUR$ qui sera à 1 si le bit P_R n'est pas cohérent avec la définition de l'émetteur. Réalisez la fonction logique du signal $ERREUR$.

Exercice 10 : mise en oeuvre électronique

1. Donner l'équation logique S des schémas transistors suivants :





2. Donner le schéma transistor CMOS des l'équations logiques suivantes : $s_1 = a \oplus b$ et $s_2 = \overline{a.b} + c.d$

1. Il s'agit ici d'une forme simple de somme de contrôle (checksum) appelé bit de parité. Les sommes de contrôle sont fréquemment utilisées dans les systèmes de transmission de type USB ou encore dans le protocole IP.