TD 4

Circuits combinatoires

Exercice 1 : Comparaison d'entiers naturels

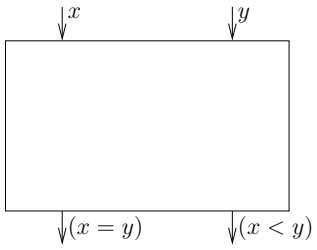
On souhaite mettre au point un circuit permettant de comparer deux entiers naturels codés en binaire, $a = (a_3 a_2 a_1 a_0)_2$ et $b = (b_3 b_2 b_1 b_0)_2$. La sortie du circuit devra prendre pour valeur 1 si $a \leq b$, 0 sinon. On va d'abord s'intéresser à la comparaison de deux bits x et y avant de passer à celle des entiers naturels : l'expression $(x = y)$ vaut 1 si x et y sont égaux, 0 sinon ; l'expression $(x \leq y)$ vaut 1 si $(x)_2 \leq (y)_2$, 0 sinon.

1) Complétez les tables de vérités suivantes, puis donnez des formules booléennes pour $(x = y)$ et pour $(x < y)$.

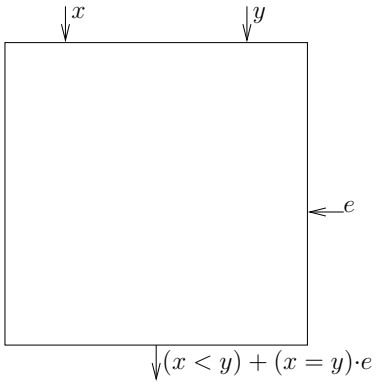
x	y	$(x = y)$
0	0	
0	1	
1	0	
1	1	

x	y	$(x < y)$
0	0	
0	1	
1	0	
1	1	

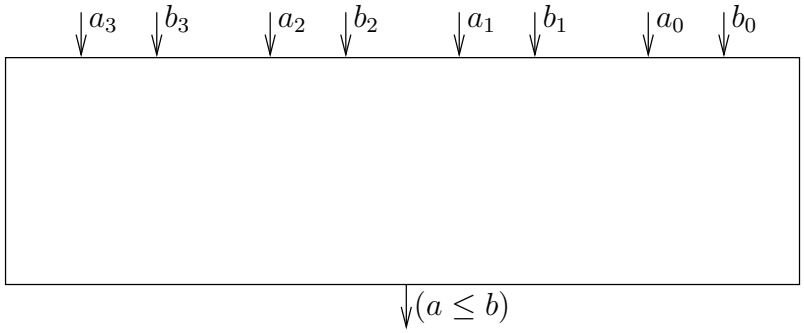
2) Complétez le circuit ci-dessous, de manière à ce que ses sorties produisent les valeurs de $(x = y)$ et $(x < y)$. Dans la suite, on appellera `HC` le composant obtenu.



3) Complétez le circuit ci-dessous, qui prend en entrée les deux bits x et y, ainsi qu'une entrée supplémentaire e, et dont la sortie prend la valeur $s = (x < y) + (x = y) \cdot e$. Dans la suite, on appellera `FC` le composant obtenu.



4) Complétez le circuit ci-dessous, de manière à réaliser la comparaison entre $a = (a_3 a_2 a_1 a_0)_2$ et $b = (b_3 b_2 b_1 b_0)_2$: la sortie prendra pour valeur 1 si $a \leq b$, 0 sinon. Expliquez votre raisonnement.



Exercice 2 : Décalage

On souhaite mettre au point un circuit permettant le décalage à droite ou à gauche d'un mot de 5 bits. Le circuit prend en entrée un mot $(e_4 e_3 e_2 e_1 e_0)$ et présente en sortie le résultat du décalage $(s_4 s_3 s_2 s_1 s_0)$. Une ligne de contrôle c permet de spécifier la direction du décalage.

— $c = 1$ correspond à un décalage à droite de 1 bit ; un 0 est alors inséré au niveau de s_4.

— $c = 0$ correspond à un décalage à gauche de 1 bit ; un 0 est alors inséré au niveau de s_0.

1) Pour $3 \leq i \leq 1$, proposez un circuit à base de portes NOT, AND et OR permettant de déterminer s_i en fonction de e_{i+1}, e_{i-1} et c. On voit que dans les deux cas chaque bit de la sortie s est calculé à l'aide des deux bits adjacents de l'entrée e. Par exemple, le bit s_2 vaut soit e_1, soit e_3 (cela dépend de la valeur de c).

2) Proposez un circuit permettant de déterminer s_0 en fonction de e_1, et c. Proposez un circuit pour déterminer s_4 en fonction de e_3, et c.

3) Représentez le décaleur 5 bits demandé.

Vous pouvez réfléchir à la manière d'associer les décaleurs, de manière par exemple à pouvoir fabriquer un décaleur 4 bits avec deux décaleurs 2 bits.

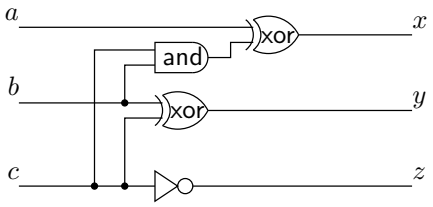
Exercice 3 : Générateur de parité impaire

Un générateur de parité impaire est une fonction qui retourne 1 si le nombre de bits à 1 parmi ses entrées est impair, 0 sinon. Définissez cette fonction pour un mot de 8 bits, et donnez un circuit logique implantant cette fonction. Vous pouvez commencer par donner une version naïve de votre circuit, mais vous essayerez ensuite de réduire sa latence.

Exercice 4 : Des circuits combinatoires

Partie 1 : Analyse d'un circuit combinatoire

On considère le circuit combinatoire suivant, dont les entrées sont a, b et c, et les sorties x, y et z :



1) Donnez des expressions booléennes pour x, y et z en fonction de a, b et c.

2) Complétez la table de vérité suivante :

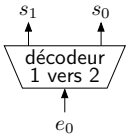
a	b	c	x	y	z
⋮	⋮	⋮	⋮	⋮	⋮

3) Quelle fonction réalise le circuit considéré ?

Partie 2 : Décodeurs

Un décodeur k bits est un circuits a k entrées $e_{k-1},\dots,e_0$ et 2^k sorties $s_{2^k-1},\dots,s_0$: la sortie $s_{(e_{k-1},\dots,e_0)_2}$ dont l'indice est indiqué par les entrées est activée et toutes les autres restent inactives.

1) On représente de la manière indiquée ci-dessous un décodeur 1 vers 2.



e_0	s_1	s_0
⋮	⋮	⋮

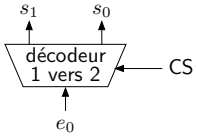
Complétez sa table de vérité, et donnez un logigramme pour un décodeur 1 vers 2.

On souhaite fabriquer un décodeur 2 vers 4 à partir de deux décodeurs 1 vers 2. Cela n'est pas facile à réaliser avec des décodeurs classiques, il faut leur ajouter une entrée supplémentaire CS (*Chip Select*). Le rôle de cette entrée est le suivant :

— quand CS=0, les sorties du décodeur restent à 0, quelles que soient les autres entrées ;

— quand CS=1, le décodeur se comporte comme un décodeur classique.

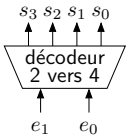
1) On représente de la manière indiquée ci-dessous un décodeur 1 vers 2 avec CS.



CS	e_0	s_1	s_0
⋮	⋮	⋮	⋮

Complétez sa table de vérité, et donnez un logigramme pour un décodeur 1 vers 2 avec CS.

2) On considère un décodeur 2 vers 4 classique, représenté ci-dessous : dressez sa table de vérité.



e_1	e_0	s_3	s_2	s_1	s_0
⋮	⋮	⋮	⋮	⋮	⋮

3) En utilisant deux décodeurs 1 vers 2 avec CS, ainsi qu'un décodeur 1 vers 2 classique, proposez un circuit réalisant un décodeurs 2 vers 4 classique. Justifiez brièvement votre réponse en vous basant sur la table de vérité de la question précédente.

Exercice 5 : Petits circuits combinatoires

1) Construire un circuit combinatoire à trois entrées x_0, x_1, x_2 capable de détecter si le nombre $(x_2x_1x_0)_2$ est divisible par 3.

2) Le but est de construire un circuit combinatoire à base de portes NOT, AND et OR pour la porte logique XOR à trois entrées a, b, c : commencez par écrire la table de vérité, puis proposez un circuit en vous basant sur la forme normale disjonctive. La porte XOR à trois entrées réalise la fonction dite d'imparité : pourquoi ce nom ?

Exercice 6 : Encodeur octal

Le but est de concevoir un encodeur octal : il s'agit d'un circuit à 8 entrées $e_0,\dots,e_7$ et 3 sorties s_0, s_1, s_2. Si l'entrée e_i est à 1 et que toutes les autres sont à 0, on veut que les sorties soient telles que $(s_2s_1s_0)_2 = i$.

1) Complétez un tableau de la forme suivante, en supposant qu'au plus l'une des entrées e_i peut être activée à la fois (dans les colonnes $e_0,\dots,e_7$, n'indiquez que les 1, pas les 0).

e_0	e_1	e_2	e_3	e_4	e_5	e_6	e_7	s_2	s_1	s_0
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

2) Exprimez les sorties s_2, s_1 et s_0 chacune sous la forme d'une somme de quatre littéraux.

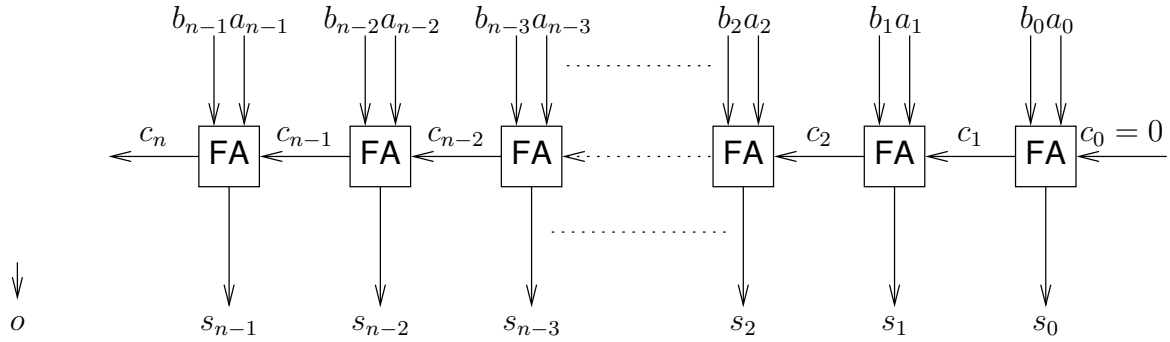
3) Représentez le circuit logique demandé (vous pouvez utiliser des portes OR à 4 entrées).

Exercice 7 : Addition en complément à 2 sur n bits

Soient a et b deux entiers relatifs, que l'on connait par leur représentation en complément à 2 sur n bits :

$$a = (a_{n-1} \dots a_0)_{\overline{2},n} = -a_{n-1}2^{n-1} + A' \quad \text{avec} \quad A' = (a_{n-2} \dots a_0)_2,$$
$$b = (b_{n-1} \dots b_0)_{\overline{2},n} = -b_{n-1}2^{n-1} + B' \quad \text{avec} \quad B' = (b_{n-2} \dots b_0)_2.$$

On note $A = (a_{n-1} \dots a_0)_2$ et $B = (a_{n-1} \dots a_0)_2$, et $S = (s_{n-1} \dots s_0)_2 = A + B \mod 2^n$; S est donc le résultat de l'addition des représentations de a et de b en tant qu'entiers naturels sur n bits. On étudie l'addition de a et de b à l'aide d'un additionneur à propagation de retenue classique, fabriqué à l'aide de *full-adders* (FA) :



Partie 1 : Dépassement de capacité

On souhaite ajouter à notre additionneur une sortie o (comme *overflow*) permettant de détecter si on se trouve en présence d'un cas de dépassement de capacité. On dit qu'il y a dépassement de capacité dans une opération en complément à 2 sur n bits **si le résultat de l'opération ne peut pas être représenté exactement** sur n bits (il est soit trop petit, soit trop grand).

Rappelons que le plus grand entier représentable en complément à 2 sur n bits est $(01\dots1)_{\overline{2},n} = 2^{n-1} - 1$, et que le plus petit est $(10\dots0)_{\overline{2},n} = -2^{n-1}$.

1) Posez, en complément à 2 sur 8 bits (ici, $n = 8$), les additions avec les opérandes a et b indiquées ci-dessous ; Donnez la valeur des opérandes et celle du résultat calculé (qui peut être différent du résultat exact) en décimal, en précisant bien le signe de ces valeurs. Indiquez clairement les opérations pour lesquelles il y a eu dépassement de capacité.

- $a = (10001111)_{\overline{2}}$, $b = (10010000)_{\overline{2}}$,
- $a = (01111111)_{\overline{2}}$, $b = (10000000)_{\overline{2}}$,
- $a = (01111111)_{\overline{2}}$, $b = (00000001)_{\overline{2}}$.

2) Montrez qu'en l'absence de dépassement de capacité, on a $a + b = (s_{n-1} \dots s_0)_{\overline{2},n}$, et qu'il y a dépassement de capacité ssi $a_{n-1} = b_{n-1} \neq s_{n-1}$. Pour cela, compléter le tableau suivant. Notez bien que, d'après les notations, $A' + B' = (\mathbf{c_{n-1}} s_{n-2} \dots s_0)_2$.

a, b	c_{n-1}	S	$a+b$	a_{n-1} $= b_{n-1}$ $\neq s_{n-1}$
$a \geq 0$ et $b \geq 0$	0	$\begin{array}{rccccc} & {\scriptstyle 0} & & & & \\ & 0 & a_{n-2} & \dots & a_0 & \\ + & 0 & b_{n-2} & \dots & b_0 & \\ \hline S = (& 0 & s_{n-2} & \dots & s_0 &)_2 \end{array}$	$\begin{aligned} a+b &= A'+B' \\ &= (0 s_{n-2} \dots s_0)_2 \\ &= (0 s_{n-2} \dots s_0)_{\overline{2},n} \end{aligned}$	faux
	1	$\begin{array}{rccccc} & {\scriptstyle 1} & & & & \\ & 0 & a_{n-2} & \dots & a_0 & \\ + & 0 & b_{n-2} & \dots & b_0 & \\ \hline S = (& 1 & s_{n-2} & \dots & s_0 &)_2 \end{array}$	$\begin{aligned} a+b &= A'+B' \\ &= (1 s_{n-2} \dots s_0)_2 \geq 2^{n-1} \end{aligned}$ Il y a dépassement de capacité.	vrai
$a \geq 0$ et $b < 0$	0	$\begin{array}{rccccc} & & a_{n-2} & \dots & a_0 & \\ + & & b_{n-2} & \dots & b_0 & \\ \hline S = (& & & & &)_2 \end{array}$		
	1	$\begin{array}{rccccc} & & a_{n-2} & \dots & a_0 & \\ + & & b_{n-2} & \dots & b_0 & \\ \hline S = (& & & & &)_2 \end{array}$		
$a < 0$ et $b < 0$	0	$\begin{array}{rccccc} & & a_{n-2} & \dots & a_0 & \\ + & & b_{n-2} & \dots & b_0 & \\ \hline S = (& & & & &)_2 \end{array}$		
	1	$\begin{array}{rccccc} & & a_{n-2} & \dots & a_0 & \\ + & & b_{n-2} & \dots & b_0 & \\ \hline S = (& & & & &)_2 \end{array}$		

3) Proposez une modification du circuit initial permettant de calculer la sortie o : elle doit prendre la valeur 1 en cas de dépassement de capacité, 0 sinon. Vous exprimerez pour cela o en fonction de c_{n-1} et c_n, en utilisant la question précédente.

Partie 2 : Additionneur-soustracteur

On souhaite maintenant compléter le circuit de manière à ce qu'il puisse aussi effectuer des soustractions en complément à 2 sur n bits. Pour cela, on ajoute une entrée e qui indiquera si le circuit doit calculer l'addition ($e = 1$) ou la soustraction ($e = 0$) des deux entiers placés sur ses entrées.

1) On note $\overline{B} = (\overline{b}_{n-1}\overline{b}_{n-2}\cdots\overline{b}_0)_2$ (complément à 1 de B) et $\overline{b} = (\overline{b}_{n-1}\overline{b}_{n-2}\cdots\overline{b}_0)_{\overline{2},n}$. On admet que, on l'absence de dépassement de capacité, si $S = (A + \overline{B}) + 1 \mod 2^n$, alors $(s_{n-1} \dots s_0)_{\overline{2},n} = a - b$. Testez, en complément à 2 sur 8 bits, la soustraction avec les opérandes a et b indiquées ci-dessous. Précisez à chaque fois s'il y a ou non dépassement de capacité.

- $a = (11111111)_{\overline{2}}$, $b = (10000000)_{\overline{2}}$,
- $a = (10000000)_{\overline{2}}$, $b = (00000001)_{\overline{2}}$,
- $a = (00000000)_{\overline{2}}$, $b = (10000000)_{\overline{2}}$.

2) Soit $D = (d_{n-1}d_{n-2}\cdots d_0)_2$ l'entier de n bits tel que

- si $e = 1$ alors $d = B$,
- si $e = 0$ alors $d = (\overline{b}_{n-1}\overline{b}_{n-2}\cdots\overline{b}_0)_2$.

Pour $0 \leq i < n$, complétez la table de vérité ci-dessous, et exprimez d_i par une fonction booléenne.

b_i	e	d_i
0	0	
0	1	
1	0	
1	1	

3) Donnez le circuit d'un additionneur-soustracteur en complément à 2 sur 8 bits (on admet que les cas de dépassement de capacité peuvent être détectés comme dans la première partie). Veillez à annoter votre schéma (il faut faire apparaître en particulier : les a_i, b_i, c_i, d_i, s_i, ainsi que e et o).