Seconde
Interrogation d'informatique (2° trimestre)
CORRECTION

PARTIE 1

I Compléter le tableau suivant de correspondance entre les systèmes décimal, binaire et hexadécimal :

Décimal	Binaire	Hexadécimal	Décimal	Binaire	Hexadécimal
O	***0***	***0***	8	*1000*	***8***
1	***1***	***1***	9	*1001*	***9***
2	***10***	***2***	10	*1010*	***A***
3	***11***	***3***	11	*1011*	***B***
4	***100***	***4***	12	*1100*	***C***
5	***101***	***5***	13	*1101*	***D***
6	***110***	***6***	14	*1110*	***E***
7	***111***	***7***	15	*1111*	***F***

II Etablir la correspondance entre les nombres exprimés dans les bases différentes en remplissant les cases vides du tableau :

Décimal	Binaire	hexadécimal
170	1010 1010	***AA***
3294	1100 1101 1110	***CDE***
4660	***1 0010 0011 0100***	1234
43 981	***1010 1011 1100 1101***	ABCD
257	***1 0000 0001***	***101***
32767	***111 1111 1111 1111***	***7FFF***

Pour faire les conversions ci-dessus, il suffit d'utiliser le tableau de conversion de la question I

Exemple $(1010\ 1010)_2 \Rightarrow (A\ A)_{16} = 16 \times A + A$ ***avec*** $A = 10 \Rightarrow (A\ A)_{16} = 170_{10}$

Ou encore $(1234)_{16} = (1\ 0010\quad 0011\quad 0100)_2 = 1 \times 16^3 + 2 \times 16^2 + 3 \times 16 + 6$

$4096 \quad + 512 \quad + 48 \quad + 6 \ = 4660$

Truc de calcul mental $16^3 = 16^2 \times 16 = 256 \times (20 - 4) = 5120 - 1024 = 4096$

Ou encore $257 = 256 + 1 = 16^2 + 1$ ***donc*** $(257)_{10} = (101)_{16} = (1\ 0000\ 0001)_2$

Etc.

III En système de base 8, peut-on rencontrer le nombre 678 ? Expliquer votre réponse.

Le 8 n'existe pas dans la base 8

IV Effectuer les opérations suivantes (si nécessaire, faire apparaître les retenues) :

En base 2 :

1010 1010	1 0000 0001	1 1010
+ 10	+ 1011	- 1
= ***1010 1100***	= ***1 0000 1100***	***= 1 1001***

en base 16

7FF	ABC	1998
+ 1	+ DEF	+ 2
= ***800***	= ***18AB***	= ***199A***

V En système binaire, comment savoir si un nombre est pair ou impair ?

Pair : dernier bit = 0

Impair dernier bit = 1

PARTIE 2

VI En base 10, avec quatre chiffres, on peut représenter tous les nombres de 0 à 9999 ;

a) en base 2 : avec huit bits :

- quel est le nombre le plus petit que l'on puisse représenter ? $X_{Min(2)}$ = ...*0000 0000*...........
- quel est le nombre le plus grand que l'on puisse représenter ? $X_{Max(2)}$ = ...*1111 1111*.......($= 2^8 - 1$)..
- En déduire l'éventail des nombres en base 2 que l'on puisse ainsi étudier, ramené en base 10 :

..................*0*..........<= $X_{(10)}$ <=........*255*... ....($= 2^8 - 1 = 256 - 1$)..

b) en base 16, avec quatre digits ("chiffres") :

- quel est le nombre le plus petit que l'on puisse représenter ? $X_{Min(16)}$ = **.......*0000***..................
- quel est le nombre le plus grand que l'on puisse représenter ? $X_{Max(16)}$ = **.......*FFFF***.......($= 16^4 - 1$)..
- En déduire l'éventail des nombres en base 16 que l'on puisse ainsi étudier, ramené en base 10 :

.................*0*........<= $X_{(10)}$ <=.....*65535*................

VII En écriture scientifique en base 10, un nombre peut être écrit sous la forme : mantisse x 10^{exposant}

Exemple le nombre 12,354 peut être écrit sous la forme 1,2354 x 10^1 ou 123,54 x 10^{-1} ou 123540 x 10^{-4} etc.

et il y a une infinité d'écriture suivant l'emplacement donné à la virgule. En informatique, on privilégie l'écriture avec un zéro avant la virgule soit l'écriture sur l'exemple ci-dessus : 0,12354 x 10^2 : cette convention permet de "coder" en machine le nombre en deux blocs un bloc pour la mantisse et un bloc pour l'exposant soit $\boxed{12354}$ $\boxed{+2}$

a) Supposons qu'il y ait cinq chiffres pour la mantisse et 2 chiffres (+ le signe) pour l'exposant pour coder un nombre <u>en base 10 :</u>

- quel est le nombre le plus petit que l'on puisse représenter ? $X_{Min(10)}$ = **....*0,10000 x10^-99*......................**
- quel est le nombre le plus grand que l'on puisse représenter ? $X_{Max(10)}$ = **....*0,99999 x10^+99*......................**

b) dans l'ordinateur, ce n'est pas la base 10 qui est utilisée mais la base 2 : supposons de même qu'un nombre réel soit représenté par cinq bits pour la mantisse et 2 bits pour l'exposant (+ 1 bit pour le signe) pour l'exposant :

- quel est le nombre le plus petit que l'on puisse représenter ? $X_{Min(2)}$ = ... ***0,10000 (mantisse) et -11 (exposant de 2)***
- quel est le nombre le plus grand que l'on puisse représenter ? $X_{Max(2)}$ = ***0,11111 (mantisse) et +11 (exposant de 2)***

En déduire l'éventail des nombres en base 10 que l'on puisse ainsi étudier:

Rappel pour la mantisse :

$(0,10000)_2 = 1 \times 2^{-1} + 0 \times 2^{-2} + 0 \times 2^{-3} + 0 \times 2^{-4} + 0 \times 2^{-5} = (0,5000)_{10}$

$(0,11111)_2 = 1 \times 2^{-1} + 1 \times 2^{-2} + 1 \times 2^{-3} + 1 \times 2^{-4} + 1 \times 2^{-5} = 0,5 + 0,25 + 0,125 + 0,0625 + 0,03125 = (0,96875)_{10}$

Rappel pour l'exposant :

$(-11)_2 = -(1 \times 2^1 + 1 \times 2^0) = (-3)_{10}$ ***or*** $2^{-3} = 8^{-1} = 0,125$

$(+11))_2 = +1 \times 2^1 + 1 \times 2^0 = (+3)_{10}$ ***or*** $2^3 = 8$

donc $X_{Min(10)} = 0,5 \times 0,125 = 0,0625$ ***et*** $X_{Max(10)} = 0,96875 \times 8 = 7,75$

.............0,0625.........<= $X_{(10)}$ <=......7,75................

On comprendra :

- que pour avoir un éventail plus large de nombres, il faut un plus grand nombre de bits : le plus souvent les nombres réels sont codés dans l'ordinateur sur 16 ou 32 bits

- deux nombres consécutifs (avec la représentation choisie) sont séparés d'environ 0,004. Exemple :

plus petit nombre : . 0,10000 (mantisse) et -11 (exposant de 2)

nombre suivant ... 0,10001 (mantisse) et -11 (exposant de 2) en base 2

nombre suivant ... 0,10010 (mantisse) et -11 (exposant de 2) en base 2 etc.

différence entre deux nombres consécutifs= 0,00001 (mantisse) et -11 (exposant de 2)

pour la mantisse $(0,00001)_2 = 0 \times 2^{-1} + 0 \times 2^{-2} + 0 \times 2^{-3} + 0 \times 2^{-4} + 1 \times 2^{-5} = 1/32 = (0,03125)_{10}$

pour l'exposant : $(-11)_2 = -(1 \times 2^1 + 1 \times 2^0) = (-3)_{10}$ ***or*** $2^{-3} = 8^{-1} = 0,125$

soit donc ε ***= plus petit intervalle entre deux nombres = 0,03125 x 0, 125 = 0,0039***

<u>donc l'ensemble des réels dans un ordinateur n'est pas continu mais discret (il y a des trous entre deux nombres consécutifs !)</u>

PARTIE 3

Portes logiques

Exercice I : Rappeler les tables de vérités des deux portes logiques suivantes

Porte NON-ET (ou NAND

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

Remarquer que =>

$S = a \oplus b$

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

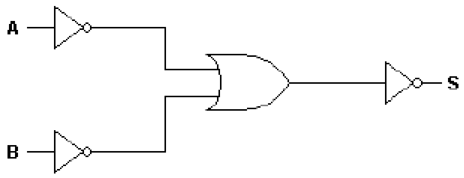
Porte OU exclusif (ou XOR)

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

Exercice II Remarque « déterminer l'équation » signifie écrire les opérations sous la forme de l'écriture en grisé du Rappel.

1)

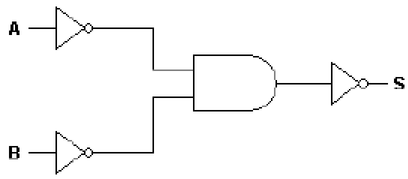
a. Déterminer l'équation du circuit de la figure suivante :



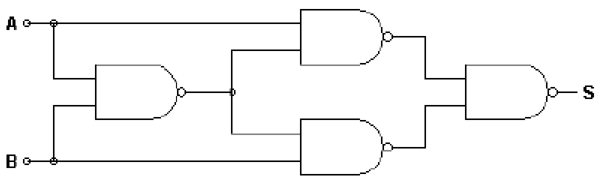
b. Dresser la table de vérité de ce circuit

c. Quelle est la fonction logique réalisée et quel est son symbole ?

2) Mêmes questions pour le circuit de la figure suivante :



3) Mêmes questions pour le circuit de la figure suivante :



Exercice 1

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

Exercice 2

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

Exercice 3

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

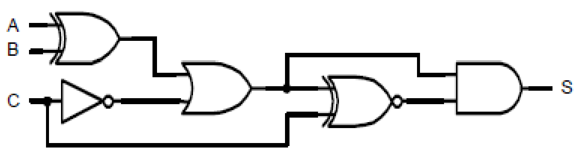
$S = \overline{\overline{a} + \overline{b}} = a.b$ $\qquad S = \overline{\overline{a} . \overline{b}} = a + b$ $\qquad S = \overline{\overline{a. \overline{a . b}} \;.\; \overline{\overline{a . b} . b}} = a \oplus b$

Fonction Exercice 1 :…***Porte ET***…… Fonction Exercice 2 :…… ***Porte OU*** …… Fonction Exercice 3 :…… ***Porte OU exclusif***

Le montage 3) permet de faire la porte XOR uniquement avec des portes NAND

Exercice III

1. Complétez la table de vérité correspondante au circuit logique suivant :



A	B	C	S
0	0	0	***0***
0	0	1	***0***
0	1	0	***0***
0	1	1	***1***
1	0	0	***0***
1	0	1	***1***
1	1	0	***0***
1	1	1	***0***

$S = \overline{((a \oplus b + \overline{c}) \oplus c)} \;.\; (a \oplus b + \overline{c}) = (\overline{a} . b + a . \overline{b}) . c = (a \oplus b) . c$