Semaine 08 – Contrôle d'erreur – TD

Exercice 1 Le langage Shadok contient quatre mots de base : *ga*, *bu*, *zo* et *meu*.

1. Proposer un codage binaire de longueur fixe minimal permettant de coder ces mots. Donner ses propriétés détectrices et correctrices.
2. Soit le codage binaire $\chi(ga) = \texttt{0000}$, $\chi(bu) = \texttt{0110}$, $\chi(zo) = \texttt{1001}$ et $\chi(meu) = \texttt{1111}$.
   (a) Combien d'erreurs peut-il détecter ?
   (b) Combien d'erreurs peut-il corriger ?
   (c) Si on reçoit `0000 0110 0100 1001` puis `0000 0110 0110 1001`, de quoi peut-on être sûr ?
3. Proposer un codage binaire 2-correcteur.

Exercice 2 Pouvoir corriger toute erreur simple pour m bits de données demande r bits de contrôle avec $m + r < 2^r$. Le codage de Hamming $(2^r - 1, 2^r - r - 1)$ prend $2^r - r - 1$ bits de données et ajoute r bits de parité aux positions $2^0, 2^1, 2^2, \ldots, 2^{r-1}$. Le bit de donnée en position $\sum_{j=0}^{r-1} a_j 2^j$ est vérifié par les bits de parité aux positions j avec $a_j = \texttt{1}$. Si un bit est erroné, sa position est obtenue comme la somme des positions des bits de parité non conformes.

0. Justifier la première phrase de l'énoncé.
1. Compléter le mot $\boxed{}\boxed{}\boxed{\texttt{1}}\boxed{}\boxed{\texttt{0}}\boxed{\texttt{1}}\boxed{\texttt{0}}$ en un mot du code de Hamming (7,4).
2. Les mots `1010110` et `0110011` se présentent au destinataire pour le code de Hamming (7,4). En supposant qu'il n'y ait pas plus d'un bit erroné par mot, retrouver les mots d'origine.
3. Même question avec les mots `0x3fa4`, `0x5d4b` et `0x046d` pour le code de Hamming (15,11).
4. Écrire les méthodes de codage (d'un mot binaire de longueur $2^r - r - 1$) et de décodage (d'un mot binaire de longueur $2^r - 1$).

Exercice 3 On propose de coder un coup au jeu du morpion avec l'alphabet S = {A, B, C, 1, 2, 3, X, O} (A, B, C représentent les colonnes, 1, 2, 3 les lignes). Il n'y a pas d'ordre préétabli : par exemple *marquer d'une croix la case à l'intersection de la colonne 1 et de la ligne 3* peut s'écrire A3X, X3A, XA3, etc.

1. Proposer un codage binaire 1-correcteur pour l'alphabet S.

2a. On propose maintenant le codage suivant :

A = `1111100000010`	B = `0001111100001`	C = `1110000110001`	X = `0000000010111`
1 = `0011000001100`	2 = `1000011001011`	3 = `1000100100100`	O = `1111111111000`

Discuter des propriétés détectrices et correctrices de ce codage.

2b. On reçoit les trois coups suivants :

```
0001111100001 1000011001011 1111111111000
1111100000011 0011000000101 0011000010101
1110000110001 1000100100100 1000011110100
```

Pour ces trois coups reçus :
(i) indiquer s'il y a détection d'erreur ;
(ii) si ce n'est pas le cas, décoder ;
(iii) si c'est le cas, indiquer si une correction est possible et dans ce cas corriger et décoder.

3. De quel autre moyen dispose-t-on pour éviter d'interpréter un mauvais coup (*i.e.* ce n'est pas le coup que le joueur a envoyé mais son altération donne un coup plausible) ?

Exercice 4 Un code de Hamming a une distance de Hamming (minimale) égale à 3 : il est donc 1-correcteur.

1. Que se passe-t-il si un mot de code (de Hamming) subit non pas une, mais deux erreurs simples ?
2. Pour remédier à ce défaut, tout code de Hamming peut être *étendu* en utilisant un bit de parité supplémentaire (*overall parity-check bit*). Quelle est la distance de Hamming d'un code de Hamming étendu ? Quelles sont ses propriétés détectrices et correctrices ? En quoi est-ce un remède ?
3. En pratique, on place ce bit de parité globale en position `0` et on garde ceux du code de Hamming original aux positions 2^k. Pour le décodage, notons s_P la parité globale du mot reçu et s_H la somme des positions 2^k des bits de parité non-conformes.
   (a) Expliciter la méthode de décodage selon que s_P et/ou s_H sont nuls ou non.
   (b) L'appliquer sur chacun des six mots reçus suivants :

1	1	1	1	
1	0	0	0	$s_P =$
1	1	0	1	$s_H =$
0	0	1	0	

$\Rightarrow$

1	0	0	0	
0	0	0	1	$s_P =$
1	0	0	0	$s_H =$
1	1	0	1	

$\Rightarrow$

1	1	0	1	
1	1	0	1	$s_P =$
1	0	0	0	$s_H =$
1	0	0	0	

$\Rightarrow$

1	1	1	1	
0	0	0	0	$s_P =$
1	0	1	0	$s_H =$
1	0	0	0	

$\Rightarrow$

1	1	1	0	0	0	0	0
0	0	0	0	0	1	1	0
0	0	1	0	1	0	1	0
0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0
0	0	0	0	1	1	1	0
0	0	1	0	0	0	1	0
1	0	0	0	0	1	0	0

$s_P =$, $s_H =$

$\Rightarrow$

0	1	1	0	0	0	0	0
0	1	0	0	0	1	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1
1	1	1	0	0	0	0	1
0	0	0	0	0	0	1	0
0	0	1	0	0	0	1	0
0	0	0	0	0	0	1	0

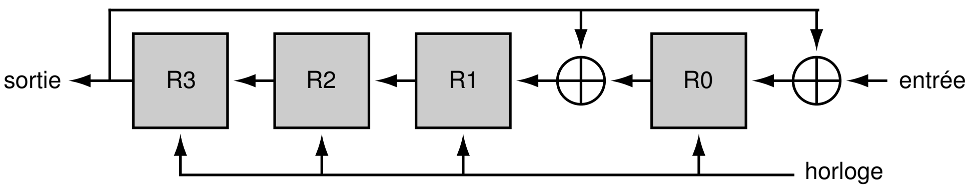
$s_P =$, $s_H =$

$\Rightarrow$

   (c) Écrire les versions *étendues* des méthodes de la question 4 de l'exercice **2**.

Exercice 5 On considère un code polynomial défini par le polynôme $G(x) = x^4 + x + 1$.

1. Rappeler le principe des codes polynomiaux à redondance cyclique (CRC).
2. Quel est le nombre r de bits de redondance qui seront ajoutés par G ?
3. On veut calculer les bits de redondance du mot $M =$ `1101101011`. Donner le polynôme $D(x)$ correspondant à M. Calculer le reste $R(x)$ de la division euclidienne de $D(x).x^r$ par $G(x)$. En déduire la valeur des bits de redondance.
4. Vérifier le mot de code reçu $N =$ `1010100100100010101 01` avec le polynôme G.
5. Les bits de redondance sont en général fabriqués par un dispositif matériel utilisant des portes logiques et des registres à décalage. Pour cet exemple, le circuit a le schéma suivant :



   Donner le contenu des registres à chaque coup d'horloge pour $D(x).x^4$.
6. Dessiner le circuit correspondant au polynôme générateur $H(x) = x^6 + x^5 + x^3 + x + 1$.
7. Calculer le CRC du mot $W =$ `101010010010001010101` avec le polynôme générateur $H(x)$.