Polynômes Binaires II : Opérations [1]

Notation hexadécimale.
La numération hexadécimale remplace avantageusement la numération binaire. Quatre chiffres binaires sont regroupés pour former un unique chiffre hexadécimal.

Dcimal	Hexadcimal	Binaire
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

On rappelle qu'en langage C, pour indiquer la numération hexadécimale on fait précéder les chiffres par `0x`. Par exemple `0x12` $= 0001\ 0010_2 = 1 \times 16 + 2 = 18_{10}$.
Dans la plupart des compilateurs, le type `long unsigned int` représente des entiers codes avec 32 bits (chiffres binaires) : $(b_0, b_1, \ldots, b_{31})$ représente l'entier $b_0 + 2b_1 + \cdots + 2^{31}b_{31}$. Les entiers représentables sont donc compris entre 0 et $2^{32} - 1$. On peut noter avec 32 chiffres binaires ou 8 chiffres hexadécimaux.

1. ADDITION ET SOUSTRACTION.

Le corps $\mathbb{F}_2$ tant de caractéristique 2, *i.e.* $1 + 1 = 0$, l'addition et la soustraction des polynômes binaires sont la m opération qui consiste effectuer une addition modulo 2 terme rme.
L'addition modulo 2 est un *ou exclusif* et est réalise par l'opérateur ^ en C.
Par exemple

1. Ce TP utilise des fonctions et des procedure programmees dans le TP "Polynômes Binaires I : Representation"

$$\begin{array}{rcll} A & = & 1+X^2+X^5+X^6 & = \texttt{0110 0101} = \texttt{0x65} \\ B & = & 1+X^+X^2+X^4+X^6+X^7 & = \texttt{1101 0111} = \texttt{0xD7} \\ A+B & = & X+X^4+X^5+X^7 & = \texttt{1011 0010} = \texttt{0xB2} \end{array}$$

2. MULTIPLICATION.

Tout d'abord, remarquons que multiplier un polynôme par X revient décaler les termes vers la gauche. Cela correspond l'expression `x << 1`.
Par exemple :

$$\begin{array}{rcllll} A & = & X+X^5+X^7 & = & \texttt{1010 0010} & = \texttt{0xa2} \\ AX & = & X^2+X^6+X^8 & = & \texttt{1 0100 0100} & = \texttt{0x144} = (\texttt{0xa2} << 1) \end{array}$$

On veut calculer le produit de $A = a_0 + a_1X + a_2X^2 + \ldots + a_nX^n$ par $B = b_0 + b_1X + b_2X^2 + \ldots + B_mX^m$. L'application des règles de distributivité, en développant par rapport B, montre que le produit $C = A \times B$ vaut

$$\begin{array}{rcl} A \times B = & & b_0A \\ & + & b_1XA \\ & & \ldots \\ & + & b_mX^mA \end{array}$$

Un algorithme de multiplication de `a` par `b` consiste donc parcourir les bits de `b` du poids faible vers le poids fort, puis, chaque itération, si le bit vaut 1, ajouter `a` (*ou exclusif*) dans une variable qui contiendra le produit, et multiplier `a` par X (décalage gauche).

Exercice (sur machine) 1.
Écrire une fonction `long unsigned PolMul(long unsigned a,long unsigned b)` qui renvoie le produit de deux polynômes binaires. Combien vaut $(X+X^5+X^7)(1+X^2+X^5+X^9)$?

3. ÉLVATION AU CARRÉE.

L'élévation au carrée de polynômes binaires est une opération linaire : $(A+B)^2 = A^2+B^2$. Cette opération peut par conséquent e réalisée de fa bien plus efficace qu'en multipliant un polynôme par lui-m.
Par exemple :

$$\begin{array}{rcllll} A & = & X+X^3+X^4 & = & \texttt{0001 1010} & = \texttt{0x1a} \\ A^2 & = & X^2+X^6+X^8 & = & \texttt{1 0100 0100} & = \texttt{0x144} \end{array}$$

Remarquez que le carré du polynôme A est obtenu en *dilatant* la représentation binaire de A, c'est–dire en insérant un zéro entre chaque chiffre. Cette remarque est la base du calcul efficace du carré.

Exercice (sur machine) 2.
Écrire une fonction `long unsigned PolSqr(long unsigned a)` qui renvoie le carré d'un polynôme binaire `a`. Les bits de `a` seront parcourus du poids faible vers le poids fort.

4. DIVISION EUCLIDIENNE.

Utilisons l'algorithme usuel de division euclidienne pour diviser par exemple $A = X + X^3 + X^4 + X^5 + X^8 + X^9$ par $B = 1 + X + X^3$, mais en utilisant les représentations binaires :

$A =$	`1100111010`	q_i
$B =$	`1011`	1
	`0111111010`	
	`1011`	1
	`0010011010`	
	`1011`	1
	`0000101010`	
	`1011`	01
$R =$	`0000000110`	00
$Q =$	`1110100`	

Le résultat est

$$\begin{aligned} Q &= X^2 + X^4 + X^5 + X^6 \\ R &= X + X^2 \end{aligned}$$

Le nombre de zéros supplémentaire dans le quotient correspond au nombre de décalages effectuer sans soustraire le diviseur.
Le premier travail est d'aligner les termes dominants de A et B en décalant B vers la gauche. Ce n'est possible que si $\deg(B) < \deg(A)$, mais dans le cas contraire, il n'y a rien calculer puisque le quotient est 0 et le reste est A.
Ensuite, B est décalé d'un rang vers la droite. Si les termes dominants de A et B sont égaux, alors il faut soustraire B de A (c'est encore un *ou exclusif*).

Exercice (sur machine) 3.
Écrire une fonction `long unsigned PolModDiv(long unsigned*r,long unsigned a,long unsigned b)` qui renvoie le quotient euclidien de `a` par `b` et qui écrit le reste dans une variable `r` dont on passe l'adresse.

5. DIVISION SUIVANT LES PUISSANCES CROISSANTES.

La division suivant les puissance croissantes est utile pour déterminer les suites produites par un LFSR. Le principe est exactement similaire la division euclidienne, mais avec un traitement dans l'ordre inverse.
Pour diviser A par B, il faut supposer que B a un terme constant.
Par exemple, divisons $A = X + X^4 + X^6 + X^9 + X^{12}$ par $B = 1 + X^3 + X^4$ suivant les puissances croissantes en utilisant leur représentation binaire :

$A =$	1001001010010	q_i
$B =$	11001	01
	10011001100000	
	11001	0001
	10010101000000	
	11001	1
	10001100000000	
	11001	01
	11101000000000	
	$\cdots$	
$Q =$	$\cdots$101100010	

Le nombre de zéros additionnels du quotient correspond au nombre de décalages faire faire `b` avant de coider sur le premier terme non nul de `a`. Les termes du quotient sont trouvés partir du poids faible.
Le résultat est

$$Q = X + X^5 + X^6 + X^8 + \cdots$$

Exercice (sur machine) 4.
Écrire une fonction `long unsigned PowerSerial(long unsigned a,long unsigned b)` qui renvoie les 32 premiers termes du résultat de la division suivant les puissances croissantes de `a` par `b`. Ces 32 termes sont représentés comme un polynôme binaire. Quels sont les termes suivants de Q ?

Exercice (sur machine) 5.
Écrire une fonction d'élévation d'un polynôme binaire la puissance n modulo P `long unsigned PolPowerMod(long unsigned a,long unsigned n,long unsigned p)`. Vérifier le petit thorme de FERMAT dans le corps $\mathbb{F}_{256}$ en vérifiant que $X^{255} \equiv 1 \bmod (1 + X + X^3 + X^4 + X^8)$.

On admettra que le polynôme $1 + X + X^3 + X^4 + X^8$ est irréductible sur F_2 et définit bien un corps.