HOMEWORK ASSIGNMENT ON THE COMPUTER IMPLEMENTATION OF $GF(2^n)$ ARITHMETIC

SAMUEL J. LOMONACO, JR.

The objective of this assignment is to learn how to implement the arithmetic operations of the characteristic 2 finite field $GF(2^n)$ on a computer with fixed wordlength $L$, where $n < L-1$. On such a machine, a computer word $w$ consists of $L$ bits $w_{L-1}, \ldots, w_2, w_1, w_0$, listed from right to left with the indices $0, 1, 2, \ldots, L-1$, and diagrammatically represented as

<table>
<thead>
<tr>
<th>Bit Index</th>
<th>$L-1$</th>
<th>$\cdots$</th>
<th>2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word $w$</td>
<td>$w_{L-1}$</td>
<td>$\cdots$</td>
<td>$w_2$</td>
<td>$w_1$</td>
<td>$w_0$</td>
</tr>
</tbody>
</table>

We assume that the computer is equipped with the following inline functions:

- **Bitwise exclusive 'OR',** denoted by XOR, and defined as:
  \[
  \text{XOR}(w, u) = (w_{L-1} + u_{L-1}, \ldots, w_2 + u_2, w_1 + u_1, w_0 + u_0),
  \]
  where $w$ and $u$ are computer words, and where `+' denotes exclusive ‘OR’, a.k.a., as addition mod 2.

- **Left Shift**, denoted by LSHIFT, and defined as:
  \[
  \text{LSHIFT}(w) = (w_{L-2}, \ldots, w_2, w_1, w_0, 0),
  \]
  where $w$ denotes a computer word.

- **Bit**, denoted by BIT, and defined by
  \[
  \text{BIT}(j, w) = w_j,
  \]
  where again $w$ denotes a computer word.

- **Set Bit**, denoted by SETBIT$(j, w)$, when called sets the $j$-th bit $w_j$ of the the computer word $w$ to 1, i.e., sets $w_j = 1$.

Let
\[
\tilde{p} = x^n + p_{n-1}x^{n-1} + \cdots + p_2x^2 + p_1x + p_0
\]
be a degree $n$ primitive binary polynomial defining the Galois field $GF(2^n)$, i.e.,
\[
GF(2^n) = GF(2)[x]/(p),
\]
and let $\xi$ denote the primitive root
\[
\xi = x + (p).
\]

We represent the primitive polynomial $\tilde{p}$ as the computer word
\[
\tilde{p} = [L-1 \cdots n+2 \ n+1 \ n \ n-1 \cdots 2 \ 1 \ 0],
\]
and we represent each element $a = \sum_{j=0}^{n-1} a_j \xi^j$ of the field $GF(2^{n-1})$ as the computer word $a = \begin{bmatrix} L-1 & \cdots & n+1 & n & n-1 & \cdots & 2 & 1 & 0 \end{bmatrix}$.

**Caveate:** The reader should take care to note that we are using two different interpretations of computer words in this assignment, i.e., one as an element of the Galois Field $GF(2^n)$, the other as an element the polynomial ring $GF(2)[x]$. Which interpretation is chosen is determined by context. As an aid to the reader, we use $\alpha$ to denote the element of the finite field $GF(2^n)$, and $\alpha$ to denote the element of the polynomial ring $GF(2)[x]$. Thus, 

$$a = \sum_{j=0}^{n-1} a_j \xi^j \quad \text{and} \quad \alpha = \sum_{j=0}^{n-1} a_j x^j.$$ 

To avoid confusion, it is suggested that, at each step, the reader ask the question: "Where does the object 'live'?"

The procedure ADD($a, b$) for addition ‘+’ in $GF(2^n)$ is simply the inline routine XOR($a, b$)

We now construct an algorithmic procedure MULT($a, b$) for multiplication ‘·’ in $GF(2^n)$. We begin by noting that

$$ab = \left( \sum_{j=0}^{n-1} a_j \xi^j \right) b = \sum_{j=0}^{n-1} a_j \xi^j b.$$ 

So to implement MULT, we first need to construct a subroutine XiShift($b$) that computes and returns $\xi b$.

```
PROC XiShift(b)
LOCAL B
B = Lshift(b)
IF Bit(n, B) = 1 THEN
  B = XOR(B, p)
END IF
RETURN(B)
END PROC
```

We next create MULT by iteratively using XiShift to compute $\xi^j b$ and then XOR-ing each such term in whenever $a_j \neq 0$. Thus, we have:

```
PROC MULT(a, b)
LOCAL A, B, C, j
A = a
B = b
C = 0
LOOP j = 0..n DO
  IF Bit(j, A) = 1 THEN
    C = XOR(C, B)
  END IF
END LOOP
RETURN(C)
```

```
```
Our next objective is to construct an algorithmic procedure \textsc{Inverse}(a) that finds the inverse \( a^{-1} \) of each element \( a \) in \( GF(2^n) \), provided \( a \neq 0 \). We will do so as follows:

First note that, since the primitive polynomial \( p \) is irreducible, it follows that \( p \) is relatively prime to every non-zero polynomial \( \tilde{a} \) of degree less than \( n \), i.e.,
\[
\gcd(\tilde{a}, p) = 1.
\]

Next note that the extended Euclidean algorithm finds polynomials \( \tilde{s} \) and \( \tilde{t} \) such that
\[
\tilde{a}\tilde{s} + \tilde{p}\tilde{t} = 1.
\]
Thus, \( \tilde{s} \) is the inverse \( a^{-1} \) of \( a \) in \( GF(2^n) \).

Your homework assignment consists of the following two problems:

\textbf{Problem 1}. Let \( \tilde{a} \) and \( \tilde{b} \) be polynomials in \( GF(2)[x] \), and let \( \tilde{q} \) and \( \tilde{r} \) be the corresponding unique polynomials in \( GF(2)[x] \) such that
\[
\tilde{a} = \tilde{b}\tilde{q} + \tilde{r},
\]
where \( \tilde{r} = 0 \) or \( \deg(\tilde{r}) < \deg(\tilde{b}) \). Construct in pseudocode two algorithms \textsc{Quo}(\( \tilde{a}, \tilde{b} \)) and \textsc{Rem}(\( \tilde{a}, \tilde{b} \)) which respectively compute \( \tilde{q} \) and \( \tilde{r} \).

\textbf{Problem 2}. Using the above algorithmic procedures \textsc{Quo}(\( \tilde{a}, \tilde{b} \)) and \textsc{Rem}(\( \tilde{a}, \tilde{b} \)), construct in pseudocode an algorithmic procedure \textsc{Inverse}(a) which computes the inverse \( a^{-1} \) of \( a \) in \( GF(2^n) \), provided \( a \neq 0 \).