

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 $w_{L-1}, \dots, w_2, w_1, w_0$, listed from right to left with the indices $0, 1, 2, \dots, L-1$, and diagrammatically represented as

$$\begin{array}{r} \text{Bit Index} \\ \text{Word } w = \end{array} \begin{array}{cccccc} L-1 & \cdots & 2 & 1 & 0 \\ \boxed{w_{L-1}} & \cdots & \boxed{w_2} & \boxed{w_1} & \boxed{w_0} \end{array}$$

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

- **Bitwise exclusive ‘OR’**, denoted by XOR, and defined as:

$$\text{XOR}(w, u) = (w_{L-1} + u_{L-1}, \dots, w_2 + u_2, w_1 + u_1, w_0 + u_0) ,$$

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:

$$\text{LSHIFT}(w) = (w_{L-2}, \dots, w_2, w_1, w_0, 0) ,$$

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 ξ denote the primitive root

$$\xi = x + (p) .$$

We represent the primitive polynomial \tilde{p} as the computer word

$$\tilde{p} = \begin{array}{cccccccccccc} L-1 & \cdots & n+2 & n+1 & n & n-1 & \cdots & 2 & 1 & 0 \\ \boxed{0} & \cdots & \boxed{0} & \boxed{0} & \boxed{1} & \boxed{p_{n-1}} & \cdots & \boxed{p_2} & \boxed{p_1} & \boxed{p_0} \end{array} ,$$

Date: November 2, 2011.

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

$$a = \begin{array}{cccccccc} L-1 & \cdots & n+1 & n & n-1 & \cdots & 2 & 1 & 0 \\ \hline 0 & \cdots & 0 & 0 & a_{n-1} & \cdots & a_2 & a_1 & a_0 \end{array} .$$

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 a to denote the element of the finite field $GF(2^n)$, and \tilde{a} to denote the element of the polynomial ring $GF(2)[x]$. Thus,*

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

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

The procedure $\text{ADD}(a, b)$ for addition '+' in $GF(2^n)$ is simply the inline routine $\text{XOR}(a, b)$

We now construct an algorithmic procedure $\text{MULT}(a, b)$ 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 $\text{XiSHIFT}(b)$ that computes and returns ξb .

```

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 iterately 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)
END PROC

```

```

END IF
B = XiSHIFT(B)
END LOOP
RETURN(C)
END PROC

```

Our next objective is to construct an algorithmic procedure $\text{INVERSE}(a)$ 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 \tilde{p} is irreducible, it follows that \tilde{p} is relatively prime to every non-zero polynomial \tilde{a} of degree less than n , i.e.,

$$\text{gcd}(\tilde{a}, \tilde{p}) = 1 .$$

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, s is the inverse a^{-1} of a in $GF(2^n)$.

Your homework assignment consists of the following two problems:

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 $\text{deg}(\tilde{r}) < \text{deg}(\tilde{b})$. Construct in pseudocode two algorithms $\text{QUO}(\tilde{a}, \tilde{b})$ and $\text{REM}(\tilde{a}, \tilde{b})$ which respectively compute \tilde{q} and \tilde{r} .

Problem 2. Using the above algorithmic procedures $\text{QUO}(\tilde{a}, \tilde{b})$ and $\text{REM}(\tilde{a}, \tilde{b})$, construct in pseudocode an algorithmic procedure $\text{INVERSE}(a)$ which computes the inverse a^{-1} of a in $GF(2^n)$, provided $a \neq 0$.

UNIVERSITY OF MARLAND BALTIMORE COUNTY
E-mail address: Lomonaco@umbc.edu
URL: www.csee.umbc.edu/~lomonaco