Indian Institute of Technology, Kharagpur
CS60004 - Hardware Security
Mid Semester Examination

Feb 25, 2019 Time Allowed: 2.0 hours

Instructions

1. Answer all the questions.
2. **Total Marks: 60**

---

1. Let us consider the computation of A^{δ_k}, where $\delta_k = \sum_{i=1}^{2^k} q^i = q + q^2 + \cdots + q^{2^k}$. Let $M(k)$ denote the number of multiplications required for computing $M(k)$. Prove that $M(k) = M(k-1) + 1$.
Hint: Try to write down A^{δ_k} in terms of $A^{\delta_{k-1}}$ and $q^{2^{k-1}}$ and then count number of multiplications in this expression.

(5 marks)

2. *[Overlap-free karatsuba Multiplier]* This multiplier uses a nice trick to reduce XOR gate delay. In this question we shall explore this overlap-free Karatsuba multiplier. We shall mainly focus on multiplications in $GF(2)[x]$ here, i.e for $A = \sum_{i=0}^{n-1} a_i x^i$ and $B = \sum_{i=0}^{n-1} b_i x^i$ we shall compute AB. In standard Karatsuba the recursive formula for AB is:

$$A_H B_H x^{2m} + [(A_H + A_L)(B_H + B_L)] - [A_H B_H + A_L B_L]x^m + A_L B_L$$

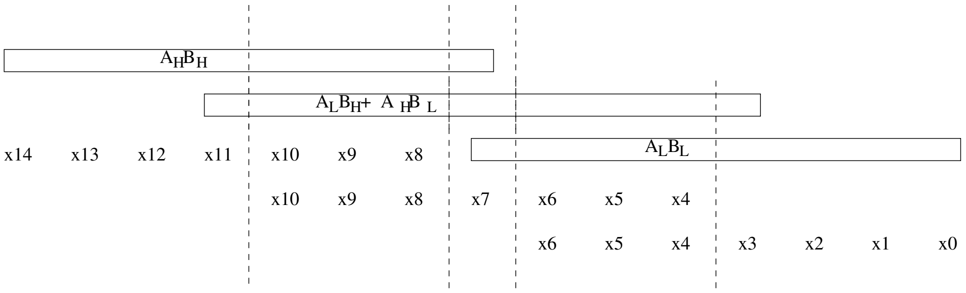

Figure 1: Idea of overlapping.

where $m = n/2$. More specifically we divide the polynomial A and B in "most significant half" and "less significant half" as: $A = x^m A_H + A_L$ and so for B. Apart from the XOR delays for the components within the $\{\}$ (which requires delay of 2 XOR computation) we have another XOR gate delay for adding the ovelapped coefficients of the partial products. For example if $m = 4$ and $n = 8$, the overlap can be represented as shown in Fig. 1.

In overlap-free Karatsuba multiplication we try to get rid of these XOR gate delay corresponding to the overlaps. Your task is to find an expression for overlap-free Karatsuba multiplier.
Hint: Split A and B according to even and odd exponents.

(10 marks)

3. This is question 2. It consists of two parts:

   (a) *[Advanced Encryption Standard (AES)]* The ShiftRow operation in AES is based on the rotation of rows of the state array. The rotation of a row by one position is represented by 4×4 permutation matrix $\widehat{R}$ over $F = GF(2^8)$ where

$$\widehat{R} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix} \tag{1}$$

   Remember AES state is a element in the vector space F^{16}. We can represent an AES state matrix as a vector

$$[a_0, a_4, a_8, a_{12}, a_1, a_5, a_9, a_{13}, a_2, a_6, a_{10}, a_{14}, a_3, a_7, a_{11}, a_{15}] \tag{2}$$

   which can be obtained by changing the basis of actual AES state matrix represented as:

$$\begin{bmatrix} a_0 & a_4 & a_8 & a_{12} \\ a_1 & a_5 & a_9 & a_{13} \\ a_2 & a_6 & a_{10} & a_{14} \\ a_3 & a_7 & a_{11} & a_{15} \end{bmatrix} \tag{3}$$

Now your task is to construct a matrix S which will perform ShiftRow operation over the vector representation of AES state. Construct S as a block diagonal matrix using power of $\widehat{R}$ as its entries. It will be a 16×16 matrix which can be represented as a block diagonal matrix of size 4×4.
Hint: A block diagonal matrix is a square matrix, having main diagonal blocks as square matrices, and the off-diagonal blocks as zero matrices.

(5 marks)

(b) Consider one single round of AES. Let us consider that an input differential of Δ is applied as the input to this round. Δ is defined as follows:

$$\begin{bmatrix} \delta & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{4}$$

where $\delta \neq 0$. It is also given that the output differential of the S-Box with δ as its input differential is $\delta^{'}$. Compute the output differential after one complete round of AES.
Hint: Try to use AES sub-operations (i.e. Sub-Bytes, ShiftRows etc. on the differential).

(5 marks)

4. *[Hybrid Quad-Quad-root Itoh-Tsujii Algorithm (ITA)]* In the class test you were asked about hybrid square-square-root ITA (SSRITA). In this question we extend this to a hybrid quad-quad-root algorithm (QQRITA), which is very similar to the SSRITA. The only difference is that it requires an addition chain of length $\frac{m-1}{4}$ for a field $GF(2^m)$. Let us consider an element $a \in GF(2^m)$. Prove the following statements:

(a) If $\alpha(a) = a^{4^i-1}$ and $\delta(a) = a^{1-4^{-i}}$ then,

$$a^{-1} = \begin{cases} \{\alpha_{\frac{m-1}{4}}(a)\}^2 \delta_{\frac{m-1}{4}}(a), & \text{if } 4 \mid (m-1) \\ \{\alpha_q(a)\}^{2^{r+1}} \delta_q(a) a^{2^{r+1}-2}, & \text{if } 4 \nmid (m-1) \end{cases}$$

Algorithm quad-quadroot-parallel-ita (Quad-Quad-root based parallel ITA)

Require: An element $a \in GF(2^m)$ and addition chain $U = \{u_i\}$ for $\frac{m-1}{4}$

$$U = \{1, 2, \ldots, \frac{m-1}{4}\}$$

Ensure: $a^{-1} \in GF(2^m)$ such that $a^{-1} \cdot a = 1$

```
begin
$l = length\ of\ U$
$\alpha_{u_1} = a^3$
$\delta_{u_1} = (a^3)^{-4}$
for $i = 2$ to $l$ do
    $p = u_{i-1}$
    $q = u_i - u_{i-1}$
    $\delta_{u_i} = \delta_q * \delta_p^{4^{-q}}$; $\alpha_{u_i} = \alpha_q * \alpha_p^{4^q}$
        //Computation of $\alpha_{u_i}$ in parallel with $\delta_{u_i}$
end for
$a^{-1} = [\alpha_{u_l}]^2 \cdot \delta_{u_l}$
return $a^{-1}$
end
```

Figure 2: Quad-quad-root parallel ITA.

Hint: Try to use Fermat's Little theorem while proving the second part.

(7 marks)

(b) Consider the following algorithm for parallel QQRITA in Fig. 2. How many clock cycles are required for this algorithm?
Hint: Both the quad and quad-root blocks have the cascaded structures as taught in the class. The expression for the number of clock cycles will be similar to that of the quad ITA discussed in class (For quad ITA $\#Cl = l_q + 1 + \sum_{i=2}^{l_q-1} \left\lceil \frac{u_i - u_{i-1}}{u_s} \right\rceil$, where l_q is the length of the addition chain $(\frac{m-1}{2})$) .

(3 marks)

(c) A simple trick here will lead to significant reduction in hardware overhead. The main observation here is that for any element u_i in the addition chain α_{u_i} can be computed from δ_{u_i} and vice versa. Establish the relation between α_{u_i} and δ_{u_i}. In other words, express one of them in terms of the other one.

(3 marks)

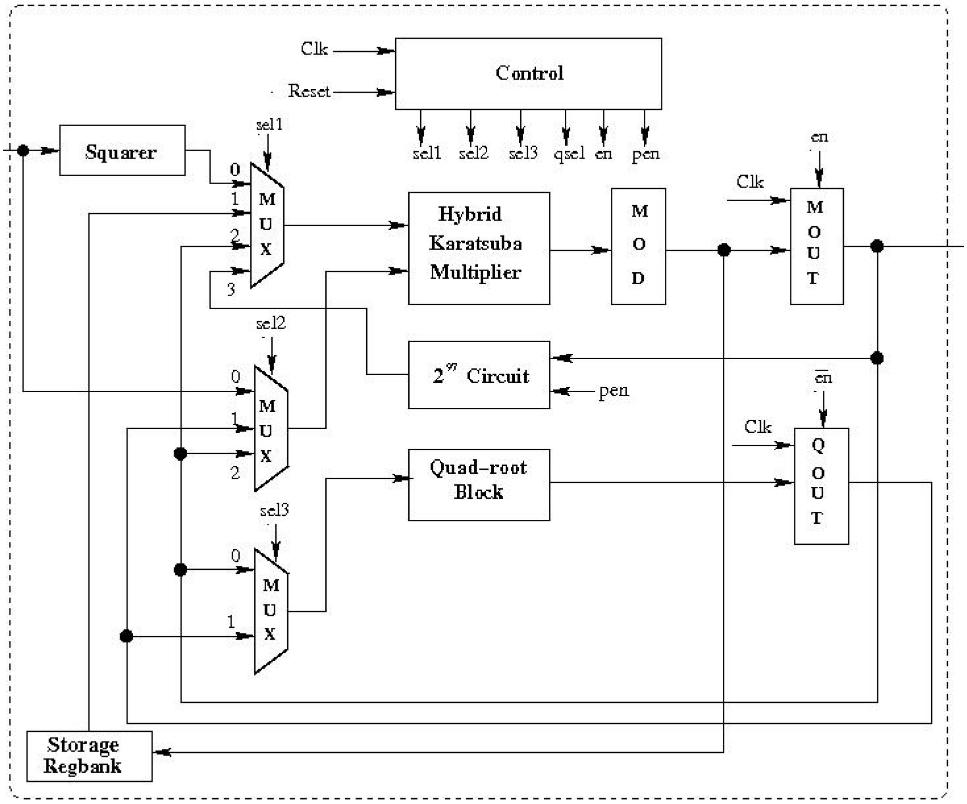

Figure 3: The new ITA architecture.

(d) Now consider the following addition chain

$$[1, 2, 3, 6, 12, 24, 48, 96, 192]$$

which is an addition chain for $m = 192$ and utilized for computing inverse in $GF(2^{193})$. With the above algorithm inverse can be computed by utilizing this addition chain only up to the length $\frac{m-1}{4} = 48$. Also, the consequence of the trick in part (c) is that we only require hardware for either the quad or the quad-root operation. Now let us consider that we use the quad-root circuit. We would require an extra circuit for computing 2^{97}. Why this circuit is required? Try to reason for this from the expression you have derived in the previous question.

Hint: Try to write down the computation steps for computing a^{-1} with quad-root and then apply the above mentioned trick. You will understand.

(2 marks)

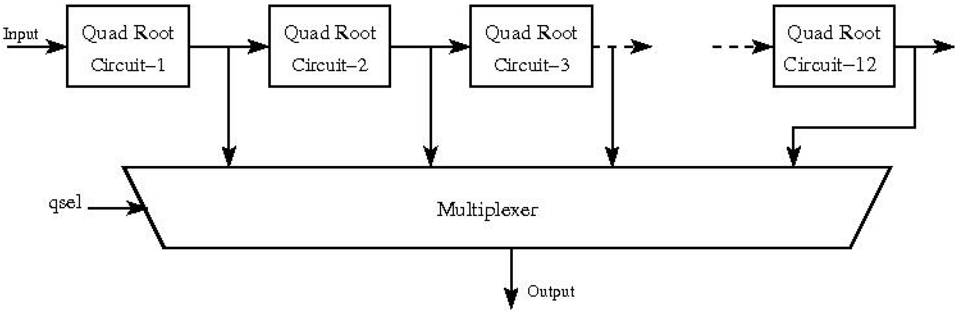

Figure 4: Cascaded quad-root block.

(e) Now consider the following circuit for computing inverses in $GF(2^{193})$ using the above mentioned tricks (Fig. 3). It consists of a quad-root circuit, a multiplier and a circuit for computing 2^{97}. Your task here is to generate the control signals for this circuit. How many clock cycles will be required?

Hint: The *qsel* signal is an input to the quad-root block which is used to select what power of quad-root we are computing. More precisely, we do $\delta^{4^{-qsel}}$ in the cascaded quad root circuit shown in Fig. 4.

(10 marks)

5. Enumerate the field $GF(2^4)$ where the irreducible polynomial is given as $x^4 + x + 1$. Find out the number of primitive elements in the multiplicative group of this field. Also write down all the primitive elements. You also need to show why these elements are primitive.

(2 + 3 + 5 marks)

End of Paper