Implementation Security In Cryptography

Lecture 06: Finite Field and Hardware

Recap

- In the last lecture
  - Basics of Hardware Design

So Far — In case you are lost

- We learnt some basic notions of security — perfect secrecy, indistinguishability, importance of having a block cipher
- We saw a simple block cipher PRESENT, and learnt Verilog to code it
- We saw some hardware design principles and learnt about delay, area etc. to roughly estimate a design cost before deployment — helps to talk in terms of hardware

Next…

- We shall learn finite fields
- We shall see a glimpse of how to implement finite field arithmetic
    - Hardware, software
- And eventually we shall see how to implement AES — which is "totally" in a finite field..

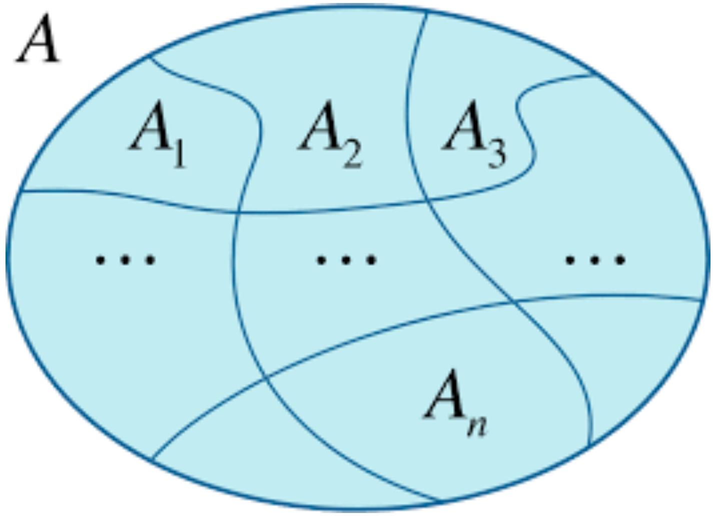
Today

- Finite Field — Mathematics and Hardware

Congruences

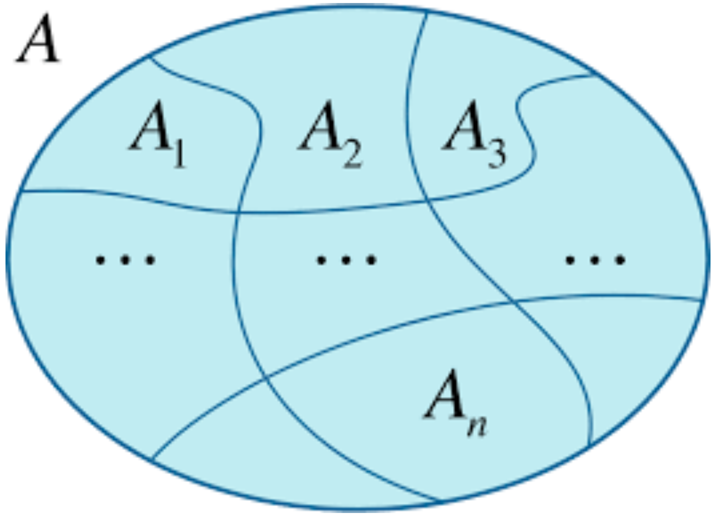
- What is $a \equiv b \ mod \ n$?

Congruences



- What is $a \equiv b \ mod \ n$?
  - $n \mid (b - a)$
- This is an equivalence relation
  - $a \equiv a \ mod \ n$ — reflexive
  - $a \equiv b \ mod \ n \implies b \equiv a \ mod \ n$ — symmetric
  - $a \equiv b \ mod \ n \wedge b \equiv c \ mod \ n \implies a \equiv c \ mod \ n$ — transitive
  - Therefore, this relation will create disjoint partitions over the set of integers.

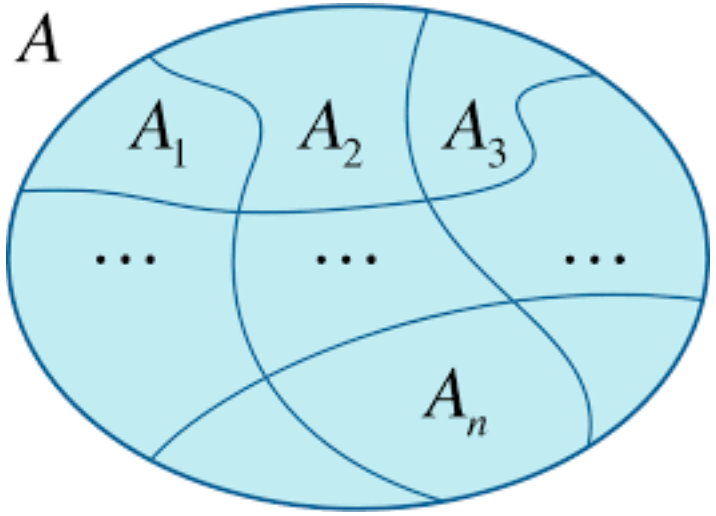
Residue Class



- $a \equiv b \ mod \ n$
  - $n \mid (b-a)$
  - $a = b + kn, k \in \mathbb{Z}$
  - The equivalent classes are as follows:
    - $a \ mod \ n$ consists of all integers that are obtain by adding (subtracting) kn with a.
  - Example: Let say n = 7
    - Residue class $1 \ mod \ 7 = \{1, 1 \pm 7, 1 \pm 2 * 7, \cdots\}$

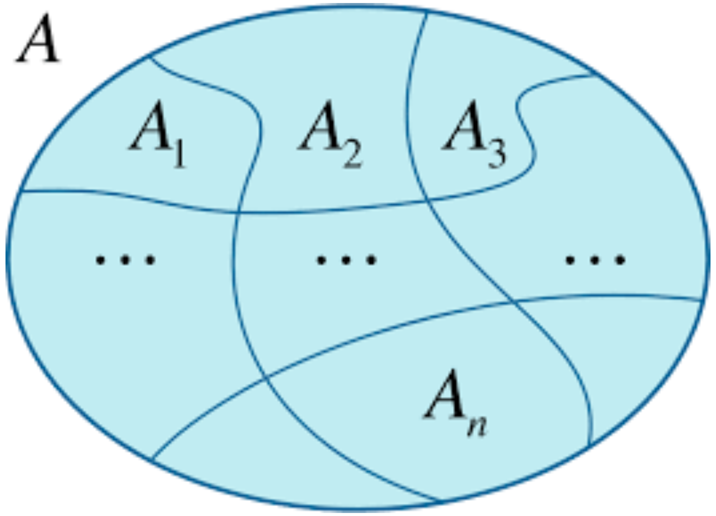
Residue Class

- Set of all residue classes mod n
  - Denoted as Z/nZ
  - How many elements does this set have?



Residue Class

- Set of all residue classes mod n
  - Denoted as Z/nZ
  - How many elements does this set have?
    - [0], [1], [2] ,…, [n-1]
  - Now let’s talk only in terms of these classes..



Some Important Theorems

- $a \equiv b \ mod \ n$, and $c \equiv d \ mod \ n$ implies
    - $-a \equiv -b \ mod \ n$
    - $a + c \equiv b + d \ mod \ n$
    - $ac \equiv bd \ mod \ n$
- Try the proofs by yourself

Group

- A group is a mathematical structure with a (nonempty) set and a (binary) operator $(G, +)$.
  - **Closure**: $a, b \in G \implies a + b \in G$
  - **Associativity**: $a, b, c \in G \implies a + (b + c) = (a + b) + c$
  - **Identity**: $\exists e \in G, \forall a \in G, a + e = e + a = a$
  - **Inverse**: $\forall a \in G, \exists a^{-1} \in G, a + a^{-1} = a^{-1} + a = e$
- A group is *abelian* or *commutative* if $\forall a, b \in G, a + b = b + a$

Examples

- The set of integers with +
  - The sums are also integers
  - a+(b+c) = (a+b)+c
  - 0 is the identity element
  - -a is the inverse of a.
- Does the set of integers form a group under multiplication?
- Set of rational numbers under multiplication?

What About the Residue Classes

- The set of residue classes form a group under “addition”
  - The addition is between classes: [a] + [b]
    - => a mod n + b mod n = (a + b) mod n
  - It is closed
  - It is associative
  - [0] is the identity
  - Inverse of [a] is basically [n - a]

$(\mathbb{Z}_3, +_3)$	$[\![0]\!]_3$	$[\![1]\!]_3$	$[\![2]\!]_3$
$[\![0]\!]_3$	$[\![0]\!]_3$	$[\![1]\!]_3$	$[\![2]\!]_3$
$[\![1]\!]_3$	$[\![1]\!]_3$	$[\![2]\!]_3$	$[\![0]\!]_0$
$[\![2]\!]_3$	$[\![2]\!]_3$	$[\![0]\!]_3$	$[\![1]\!]_3$

What About the Residue Classes

- The set of residue classes form a group under "addition"
  - The addition is between classes: [a] + [b]
    - => a mod n + b mod n = (a + b) mod n
  - It is closed
  - It is associative
  - [0] is the identity
  - Inverse of [a] is basically [n - a]
- The set of residue classes is also a group under multiplication **under certain conditions**

$(\mathbb{Z}_3, +_3)$	$[\![0]\!]_3$	$[\![1]\!]_3$	$[\![2]\!]_3$
$[\![0]\!]_3$	$[\![0]\!]_3$	$[\![1]\!]_3$	$[\![2]\!]_3$
$[\![1]\!]_3$	$[\![1]\!]_3$	$[\![2]\!]_3$	$[\![0]\!]_0$
$[\![2]\!]_3$	$[\![2]\!]_3$	$[\![0]\!]_3$	$[\![1]\!]_3$

Multiplicative Group Module n

- We denote it by (($\mathbb{Z}/ n\mathbb{Z}, \circ$) or (($\mathbb{Z}_n, \circ$)
  - The multiplication is between classes: [a] * [b]
    - => (a mod n) * (b mod n) = (a + k1*n)(b + k2*n) = ab + a*k2*n + b*k1*n + k1*k2*n^2 = ab + (a*k2 + b*k1 + k1*k2*n)*n = ab + k3*n = [ab]
  - It is closed
  - It is associative (prove it)
  - [1] is the identity
  - But where is the inverse?????
- **Turns out that inverse only exist for certain elements not for all**
- **Let's define this subset as $\mathbb{Z}_n^*$ — this indeed forms a group**

Multiplicative Group Module n

- What are the elements of $\mathbb{Z}_n^*$?

Multiplicative Group Module n

- What are the elements of $\mathbb{Z}_n^*$?
  - Elements that are co-prime to n
  - Another way: gcd(a,n) = 1
- What happens if n is a prime, say p?

Multiplicative Group Module n

- What are the elements of $\mathbb{Z}_n^*$?
  - Elements that are co-prime to n
  - Another way: gcd(a,n) = 1
- What happens if n is a prime, say p?
- $|\mathbb{Z}_n^*| = p - 1$
- How many elements in $\mathbb{Z}_n^*$ can be there if n is not a prime?
  - This number is called $\Phi(n)$ — **Euler's Totient Function**
  - **Example:** $\Phi(26) = 13$**,** $\Phi(p) = p - 1$**, if p is prime**

Fermat’s Little Theorem

- If gcd(a, n) = 1, then $a^{\Phi(n)} = 1\ mod\ n$
  - That means for any element of $\mathbb{Z}_n^*$, we can raise it to the power of $\Phi(n)$ and end up in the identity element!!!
- Proof: Will not be discussed for the sake of time!! I can tell you later if you are interested..
- $a^{p-1} = 1\ mod\ p$, if p is prime.
- <u>Interesting fact</u>: $a^{p-2} = a^{-1}\ mod\ p$

Small Examples

- Let’s consider $\mathbb{Z}_4^*$
- What are the elements? [1], [3] — let’s abuse the notation and denote as 1, 3
- $\Phi(4) = 2$
- So, let’s see $3^4\ mod\ 4$ = 81 mod 4 = 1 !!!
- Calculate: $2^{1000}\ mod\ 13$ = $2^{(83\times 12)+4}\ mod\ 13 = 16\ mod\ 13 = 3\ mod\ 13$

Ring

- Let's consider a (nonempty) set with two operations $(G, +, \circ)$
  - G is an abelian group under addition
  - G is closed under multiplication
  - Multiplication is associative
  - There is an identity element
  - **But not every element has a multiplicative inverse (other than 0)**.
  - Also, multiplication is **distributive** on addition
    - $a \cdot (b + c) = (a \cdot b) + (a \cdot c)$ for all a, b, c in R (left distributivity).
    - $(b + c) \cdot a = (b \cdot a) + (c \cdot a)$ for all a, b, c in R (right distributivity)
  - If the multiplication is commutative we call it **commutative ring**

Example

- Consider $\mathbb{Z}_n$ with + and $\circ$
  - G is a ring if n is composite, there is no inverse unless gcd(a,n) = 1.
- Consider $\mathbb{Z}_4$ — 2 does not have an inverse — no element can be multiplied with 2 giving 1. Rather 2*2 mods 4 = 0. 2 is, therefore, called a **zero divisor**.
- But then consider $\mathbb{Z}_5$ — It is also a ring, but you can see that every element has an inverse.

Field

- Let's consider a (nonempty) set with two operations $(G, +, \circ)$
  - G is commutative ring with every element except 0 having a multiplicative inverse.
- Example:
  - Set of real numbers with addition and multiplication
  - Set of rational numbers with addition and multiplication
  - Set of complex numbers with addition and multiplication
  - **Set of integers modulo a prime**

Finite Fields

- A finite field is a field with a finite number of elements.
  - **Integer modulo a prime**, but there are others too.
- The number of elements in the set is called the **order** of the field.
- A field with order m exists iff m is a prime power, i.e $m=p^n$ for some integer n and with p a prime integer.
- p is called the **characteristic** of the finite field.

But p^n is composite right?

- The representation of the field elements change
  - They are no more residue classes modulo an integer.
  - But what are they now?

Galois Fields

- GF(p): The elements of the fields can be represented by 0, 1, …, p-1
- For p^n, Elements are represented as polynomials over GF(p).

Binary Finite Fields

- **Binary Finite Fields:** The set G consists polynomials with coefficients in {0,1}
  - Also known as Galois field
  - Represented as $GF(2^m)$, where 2^m is the number of elements in S
  - Addition is XOR
  - For GF(2) — multiplication is AND
- AES is constructed using binary finite fields

Polynomials over a Field

A polynomial over a field F is an expression of the form:

$$b(x) = b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + \ldots + b_0$$

x being called indeterminate of the polynomial, and the $b_i \in F$ the coefficients.

The degree of a polynomial equals l if $b_j = 0,\ \forall j > l$, and l is the smallest number with this property. The set of polynomials over a field F is denoted by F[x]. The set of polynomials over a field F, which has a degree less than l, is denoted by $F[x]|_l$

Operations on Polynomials

- **Addition:**

$$\boxed{c(x) = a(x) + b(x) \Leftrightarrow c_i = a_i + b_i, 0 \le i \le n}$$

Example

Let F be the field in $GF(2)$. Compute the sum of the polynomials denoted by 87 and 131

In binary, 87 =01010111, and 131 =10000011.

In polynomial notations we have,

$$(x^6 + x^4 + x^2 + x + 1) \oplus (x^7 + x + 1)$$

$$= x^7 + x^6 + x^4 + x^2 + (1 \oplus 1)x + (1 \oplus 1)$$

$$= x^7 + x^6 + x^4 + x^2$$

The addition can be implemented with the bitwise XOR instruction.

Operations on Polynomials

Addition is closed

0 (polynomial with all coefficients 0) is the identity element.

The inverse of an element can be found by replacing each coefficient of the polynomial by its inverse in F.

$< F[x]_l, + >$ forms an Abelian group

Multiplication

- Associative
- Commutative
- Distributive wrt. addition of polynomials.

In order to make the multiplication closed over $F[x]|_l$
we select a polynomial m(x) of degree l, called the reduction polynomial.

The multiplication is then defined as follows:

$$c(x) = a(x).b(x) \Leftrightarrow c(x) \equiv a(x) \times b(x) \pmod{m(x)}$$

Hence, the structure $< F[x]|_l, +, . >$ is a commutative ring.

For special choices of the polynomial m(x), the structure becomes a field.

Irreducible Polynomial

- A polynomial d(x) is irreducible over the field GF(p) iff there exist no two polynomials a(x) and b(x) with coefficients in GF(p) such that d(x)=a(x)b(x), where a(x) and b(x) are of degree > 0.

Let F be the field GF(p). With suitable choice for the reduction polynomial, the structure $<F[x]|_n, +, .>$ is a field with p^n elements, usually denoted by $GF(p^n)$.

Example

Degree	Irreducible Polynomial
1	$(x+1), x$
2	(x^2+x+1)
3	(x^3+x^2+1), (x^3+x+1)
4	$(x^4+x^3+x^2+x+1)$, (x^4+x^3+1),(x^4+x+1)

Example of Multiplication

Compute the product of the elements 87 and 131 in GF(2^8)

87 =01010111, and 131 =10000011.

Example of Multiplication

Compute the product of the elements 87 and 131 in GF(2^8)

87 =01010111, and 131 =10000011.

In polynomial notations we have,

$(x^6 + x^4 + x^2 + x + 1) \times (x^7 + x + 1)$

$= (x^{13} + x^{11} + x^9 + x^8 + x^7) \oplus (x^7 + x^5 + x^3 + x^2 + x)$

$\oplus (x^6 + x^4 + x^2 + x + 1)$

$= x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1$

and,

$(x^{13} + x^{11} + x^9 + x^8 + x^6 + x^5 + x^4 + x^3 + 1)$

$\equiv x^7 + x^6 + 1 \pmod{x^8 + x^4 + x^3 + x + 1}$

Finite Field Multiplication

- Consider for example the field $GF(2^4)$ with irreducible polynomial $x^4 + x + 1$

$$
\begin{array}{cccccc}
 & & x^3 & +x^2 & +1 & \\
 & & x^2 & +x & +1 & \\
\hline
 & & x^3 & +x^2 & & +1 \\
 & x^4 & +x^3 & & +x & \\
x^5 & +x^4 & & +x^2 & & \\
\hline
x^5 & & & & +x & +1
\end{array}
$$

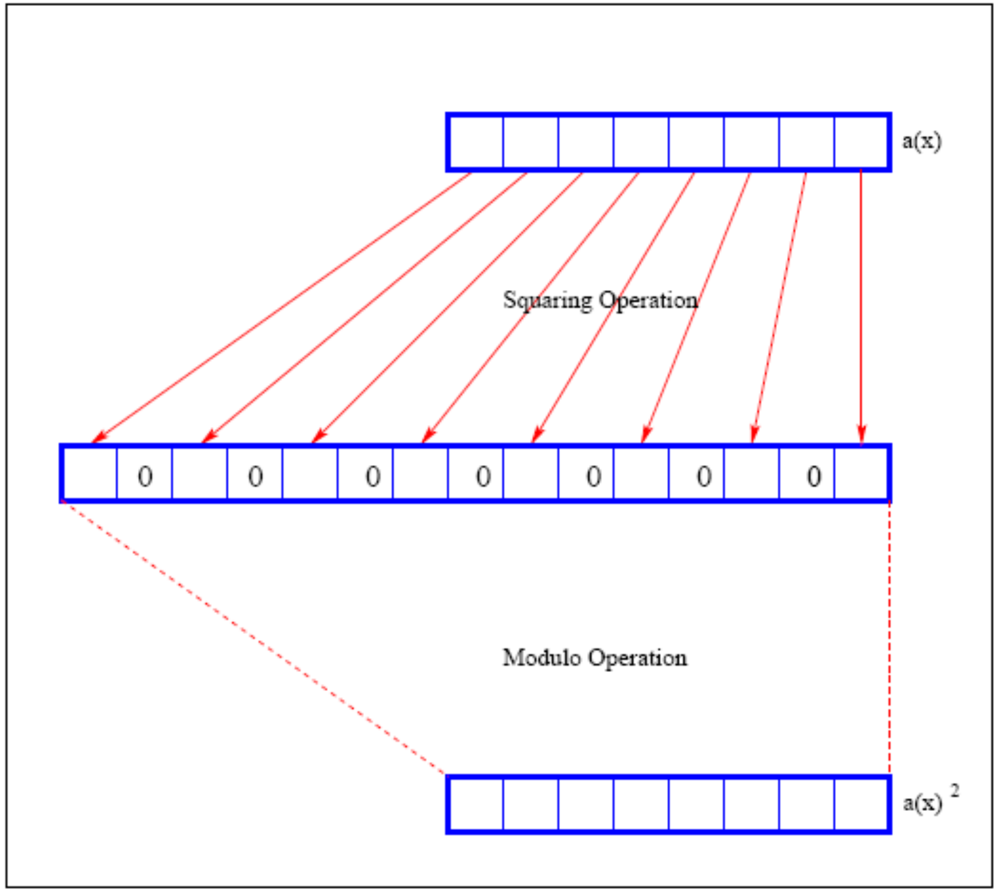
- $x^5 + x + 1$ is not in the field $GF(2^4)$
- So, modular reduction

$$(x^5 + x + 1) \bmod (x^4 + x + 1) = x^2 + 1$$

Main Points in Multiplication

- Do binary multiplication
  - Bitwise AND for partial products
  - XOR your partial products (with proper shifts)
- You can use any multiplier here — Karatsuba performs quite well.
  - Only you need to do XORs while combining the results of multiplications
- You will have a large polynomial — for two n-1 degree polys the result will be of (max) degree 2n-2.
- Now reduce with a the reduction poly of degree n.

Squaring



Squaring?

- Let’s do it…

Multiplication Algorithms

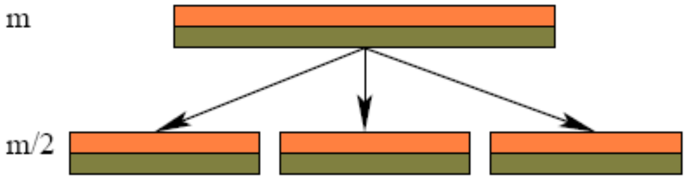
Multiplier	Space Complexity
Karatsuba	$O(n^{\log_2 3})$
Mastrovito	$O(n^2)$
Sunar-Koc	$O(n^2)$
Massey Omura	$O(n^2)$
Montgomery	$O(n^2)$

- The choice of multiplier is determined by the application.
  - Montgomery for example is suited for low resource environments.
  - If designed properly, the Karatsuba multiplier is the fastest.

Finite Field Multiplication

- There are several forms of : **Karatsuba multiplier**. We consider the combinational type which requires just a single clock cycle.
- Two common types of combinational Karatsuba implementations.
  - Simple Karatsuba Multiplier.
  - General Karatsuba Multiplier.

Simple Karatsuba Multiplier



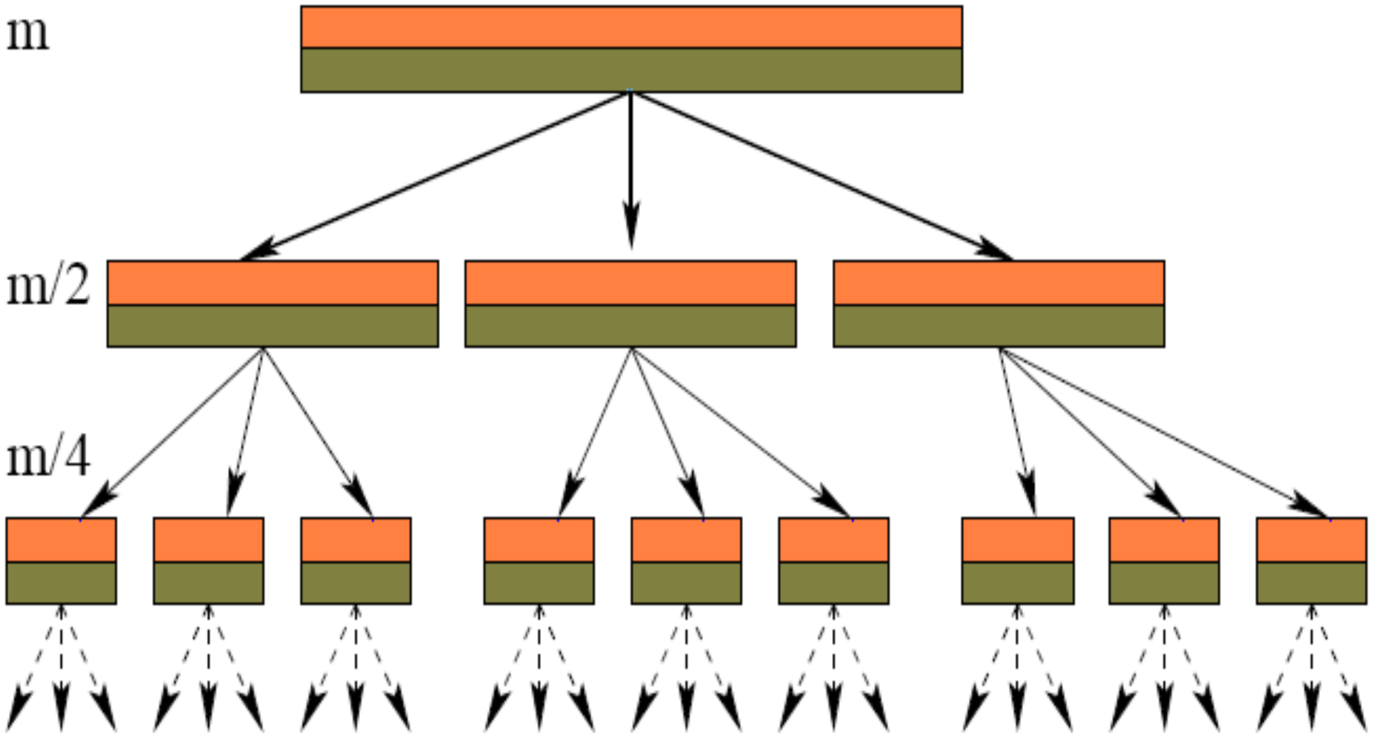
Split multiplicands into two

$$A(x) = A_h x^{m/2} + A_l$$
$$B(x) = B_h x^{m/2} + B_l$$

Use three $m/2$ bit multiplications

$$\begin{aligned}
C'(x) &= (A_h x^{m/2} + A_l)(B_h x^{m/2} + B_l) \\
&= A_h B_h x^m + (A_h B_l + A_l B_h) x^{m/2} + A_l B_l \\
&= A_h B_h x^m \\
&\quad + ((A_h + A_l)(B_h + B_l) + A_h B_h + A_l B_l) x^{m/2} \\
&\quad + A_l B_l
\end{aligned}$$

Recursive Simple Karatsuba Multiplier

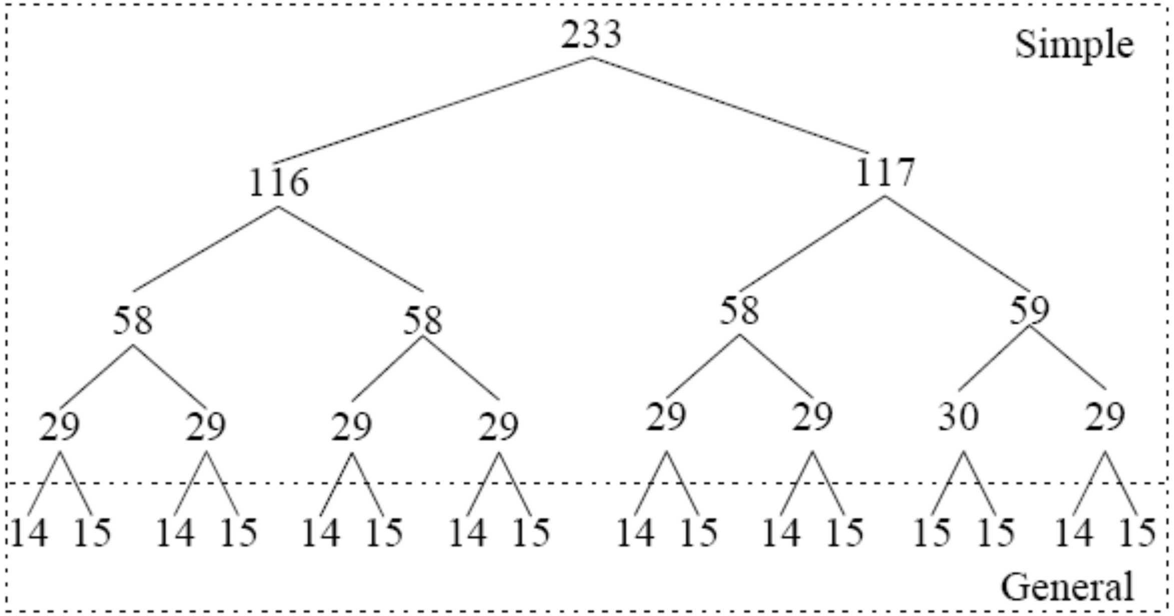


General Karatsuba Multiplier

- Instead of splitting into two, splits into more than two.
  - For example, an m bit multiplier is split into m different multiplications.

A. Weimerskirch, *Generalizations of the Karatsuba Algorithm for Efficient Implementations,* Cryptology ePrint Archive, 2006

Karatsuba Multiplier



The multiplier operates on 233 bit inputs and gives a 465 bit outputs.

The multiplier uses sub-multipliers, with operands as described in the figure.

The initial multipliers are Simple Karatsuba based, however after a threshold of 29, it was realized by Generalized Karatsuba blocks.

Module Multiplier in Verilog

```
module multiplier(a, b, d);
input wire [232:0] a;
input wire [232:0] b;
output wire [232:0] d;
wire [464:0] mout;

ks233 ks(a, b, mout);           (Karatsuba Multiplier)
mod   mod1(mout, d);            (Modulo Operation)

endmodule
```

Comparing the General and Simple

m	General			Simple		
	Gates	LUTs	LUTs Under Utilized	Gates	LUTs	LUTs Under Utilized
2	7	3	66.6%	7	3	66.6%
4	37	11	45.5%	33	16	68.7%
8	169	53	20.7%	127	63	66.6%
16	721	188	17.0%	441	220	65.0%
29	2437	670	10.7%	1339	669	65.4%
32	2977	799	11.3%	1447	723	63.9%

- **Hybrid Karatsuba Multiplier**
  - For all recursions less than 29 use the General Karatsuba Multiplier or school book.
  - For all recursions greater than 29 use the Simple Karatsuba multiplier

C. Rebeiro, *Power Attack Resistant Efficient FPGA Architecture for Karatsuba Multiplier , VLSID 2008*