TECHNISCHE
UNIVERSITÄT
DARMSTADT

INTEGRIERTE
SCHALTUNGEN
UND SYSTEME

A Reconfigurable System on Chip Implementation for Elliptic Curve Cryptography over GF(2^n)

Michael Jung[1], M. Ernst[1], F. Madlener[1],
S. Huss[1], R. Blümel[2]

[1] Integrated Circuits and Systems Lab
Computer Science Department
Darmstadt University of Technology
Germany

[2] cv cryptovision GmbH
Gelsenkirchen
Germany

ECC over GF(2^n)

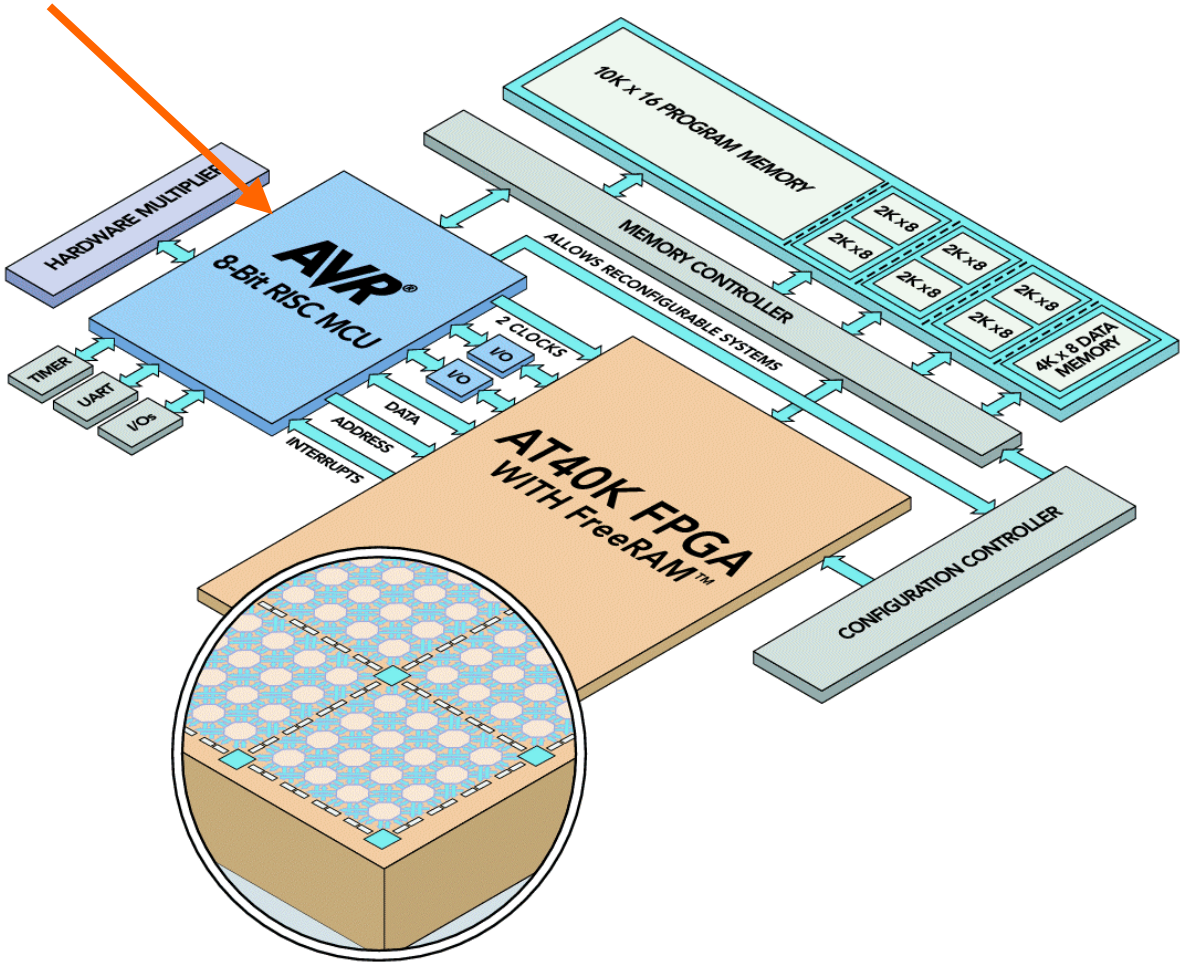
- GF(2^n) operations implemented in HW
  - Square, Multiply, Add
  - Inversion with *Fermat's little Theorem*
  - Polynomial Base
  - Multiplier based on a novel, generalized version of the *Karatsuba Ofman Algorithm*[1]
- EC level algorithms implemented in SW
  - *2P Algorithm*[2] for EC point multiplication
  - Projective Point Coordinates
- Algorithmic flexibility combined with performance

[1] Karatsuba and Ofman, *Multiplication of multidigit numbers on automata,* 1963
[2] Lopez/Dahab, *Fast multiplication on elliptic curves over GF(2m) without precomputations,* 1999

Atmel FPSLIC

Reconfigurable System on Chip

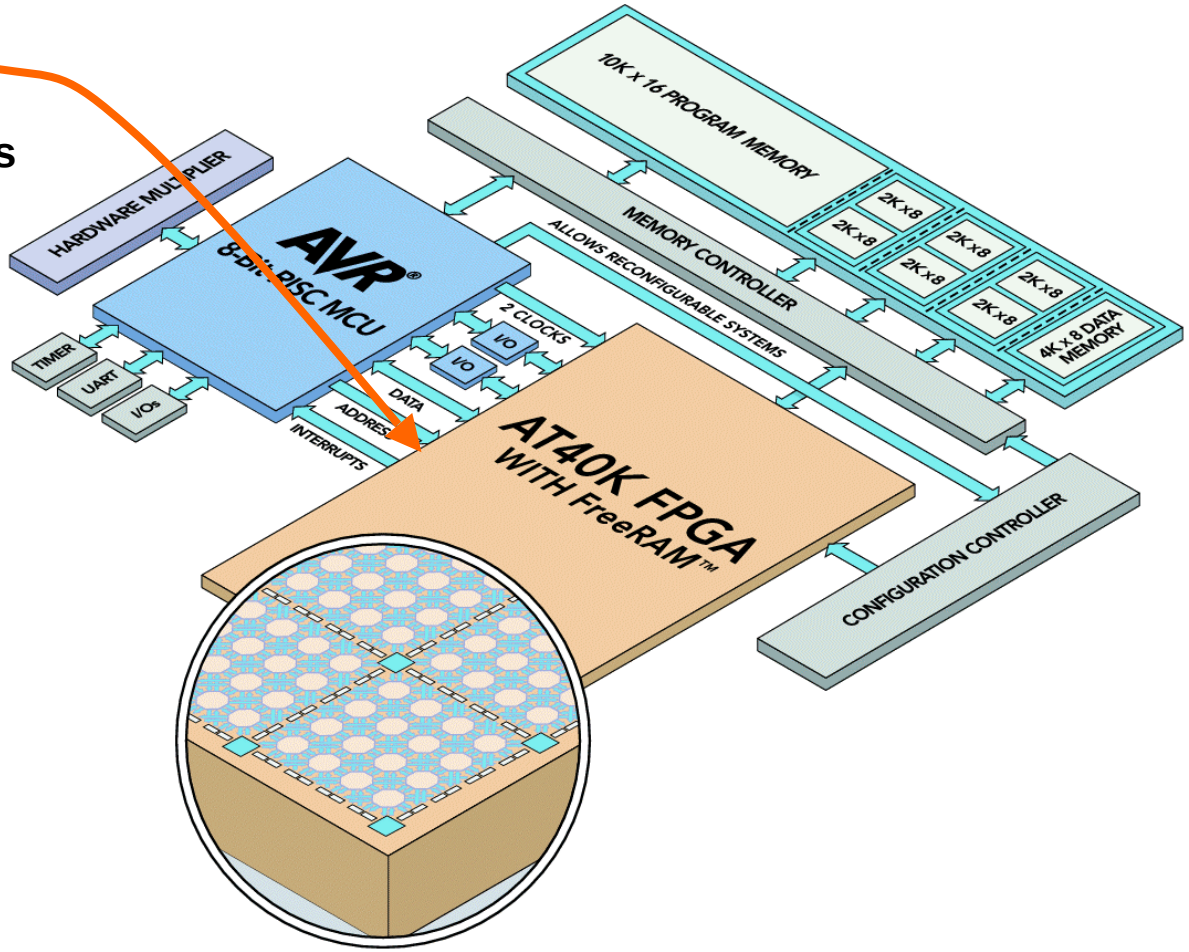
- 8-Bit RISC Microcontroller
  - 20+ MIPS @ 25 MHz



Atmel FPSLIC

Reconfigurable System on Chip

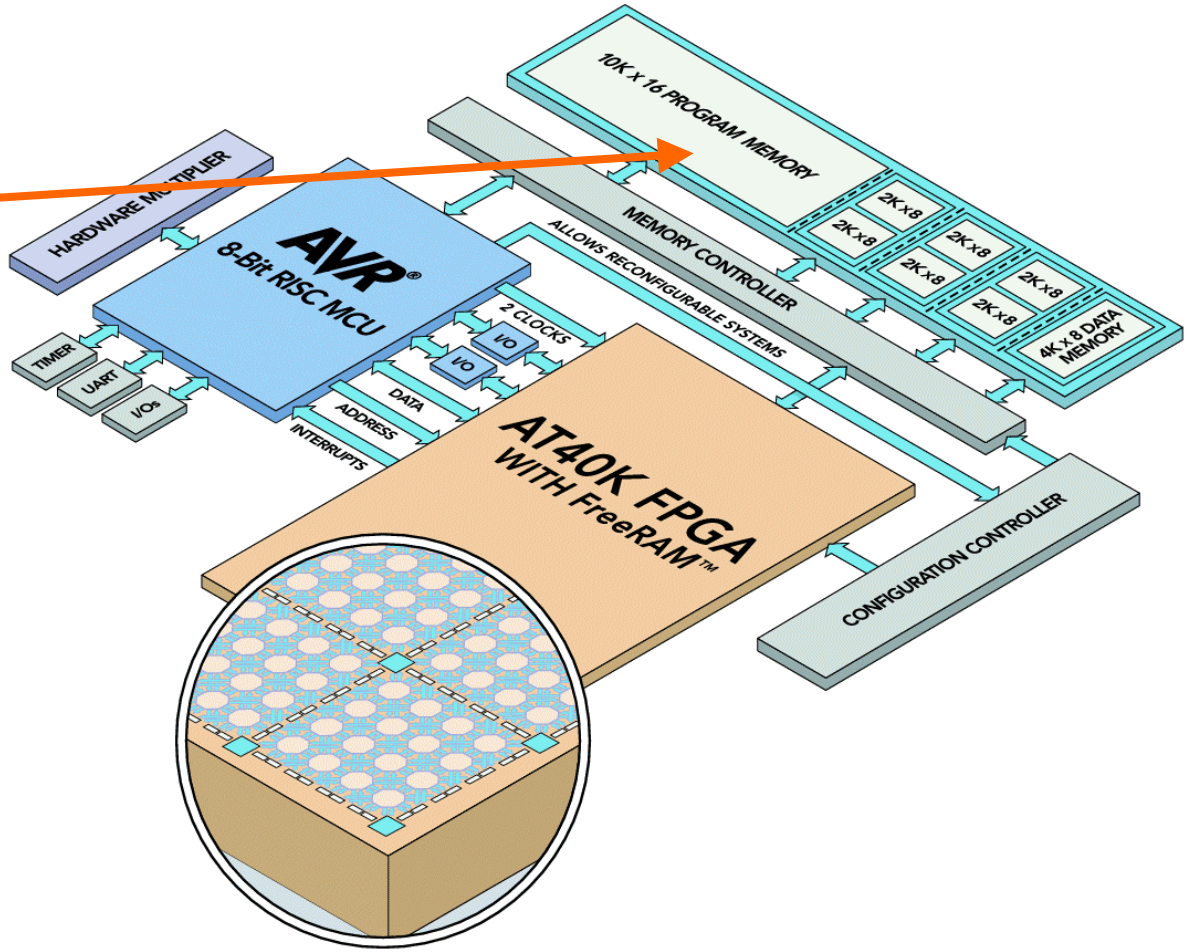
- 8-Bit RISC Microcontroller
- FPGA
  - 40k Gate Equivalents
  - Distributed FreeRAM™ Cells



Atmel FPSLIC

Reconfigurable System on Chip

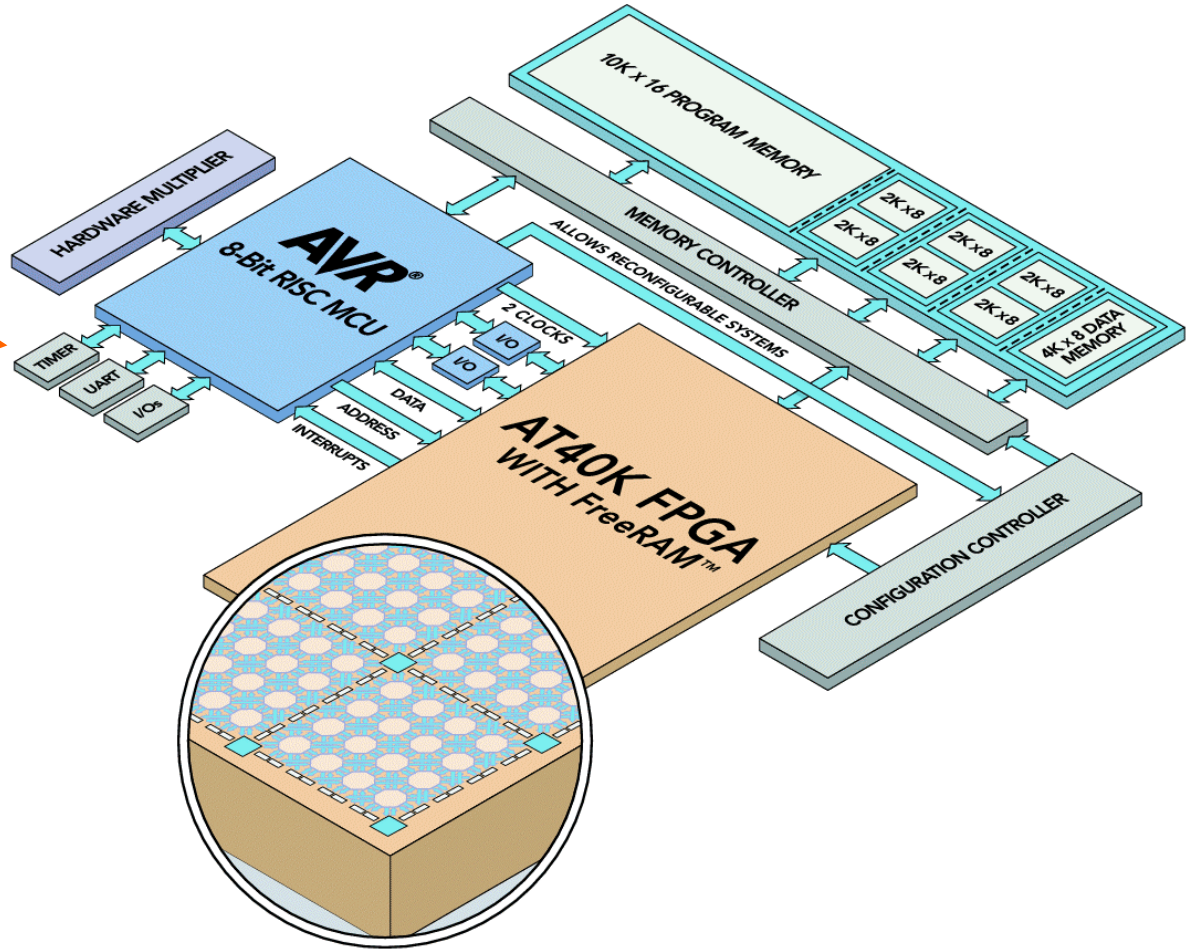
- 8-Bit RISC Microcontroller
- FPGA
- 36 kByte RAM
  - Dual Ported
  - Simultaneously accessible by MCU and FPGA



Atmel FPSLIC

Reconfigurable System on Chip

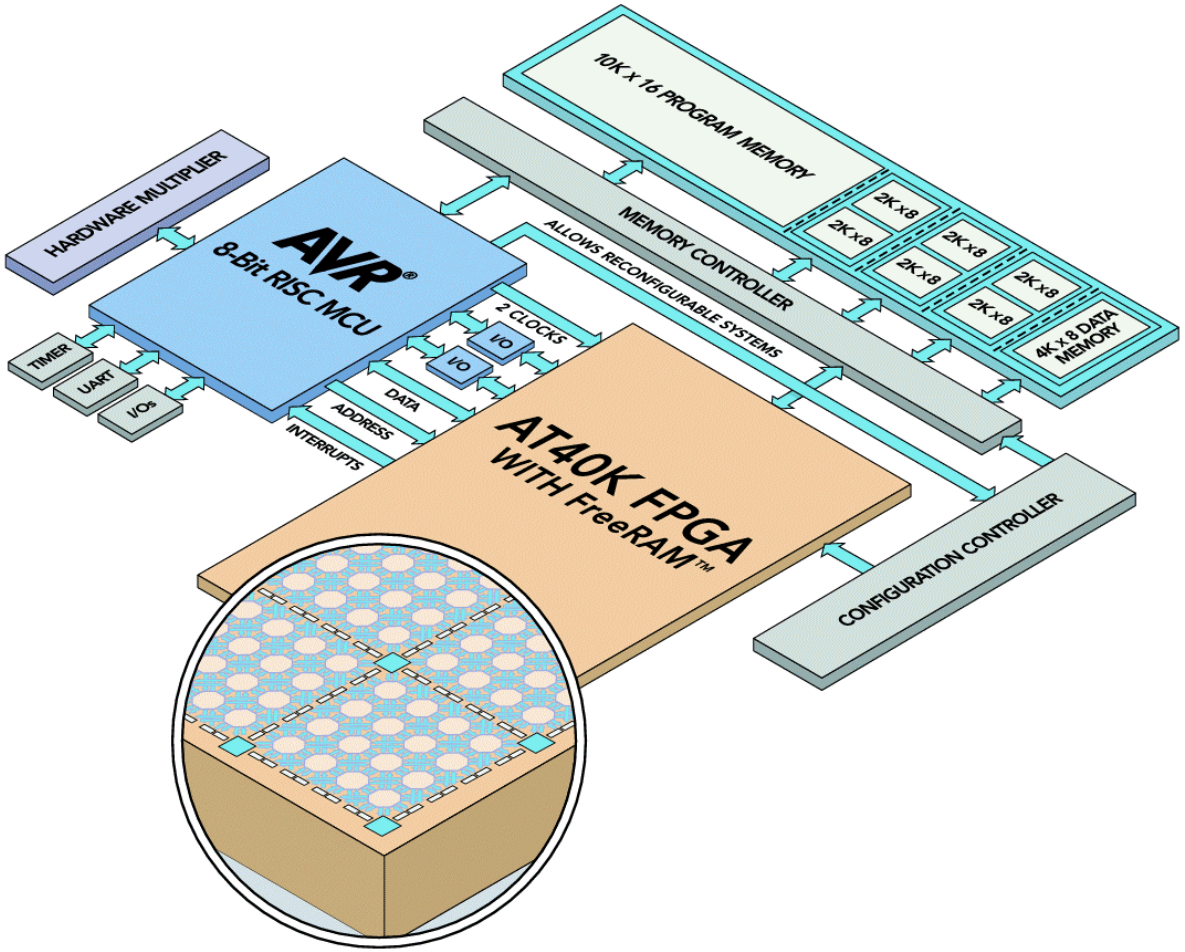
- 8-Bit RISC Microcontroller
- FPGA
- 36 kByte RAM
- Peripherals
  - UARTs
  - Timers
  - etc.



Atmel FPSLIC

Reconfigurable System on Chip

- 8-Bit RISC Microcontroller
- FPGA
- 36 kByte RAM
- Peripherals
- Low cost, low complexity device



Polynomial Karatsuba Multiplication

$$A = \bigoplus_{i=0}^{n-1} a_i x^i \qquad B = \bigoplus_{i=0}^{n-1} b_i x^i$$

$$A \cdot B = (A_1 \hat{x} \oplus A_0) \cdot (B_1 \hat{x} \oplus B_0) \qquad \text{with} \qquad \hat{x} = x^{n/2}$$

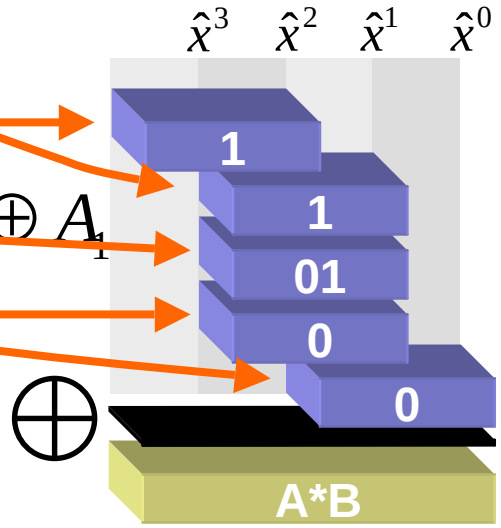
$$= (A_1 \odot B_1)\hat{x}^2 \oplus (A_1 \odot B_0 \oplus A_0 \odot B_1)\hat{x} \oplus (A_0 \odot B_0)$$

$$T_1 = (A_1 \odot B_1)$$

$$T_2 = (A_1 \oplus A_0) \odot (B_1 \oplus B_0) = A_1 B_0 \oplus A_0 B_1 \oplus A_1$$

$$T_3 = (A_0 \odot B_0)$$

$$A \cdot B = T_1 \hat{x}^2 \oplus (T_2 \oplus T_1 \oplus T_3)\hat{x} \oplus T_3$$



Multi-Segment Karatsuba (MSK)

- Generalization of the Karatsuba Multiplication Algorithm
- Polynomials are split into an arbitrary number of segments

$$MSK_k(A,B) = \left(\bigoplus_{i=1}^{k} S_{i,0}(A,B) \cdot \hat{x}^{i-1} \right) \oplus \left(\bigoplus_{i=1}^{k-1} S_{k-i,i}(A,B) \cdot \hat{x}^{i-1+k} \right)$$

with

$$S_{m,l}(A,B) = \left(\bigoplus_{i=1}^{m-1} S_{i,l}(A,B) \right) \oplus \left(\bigoplus_{i=1}^{m-1} S_{i,l+m-i}(A,B) \right) \oplus M_{m,l}(A,B),$$

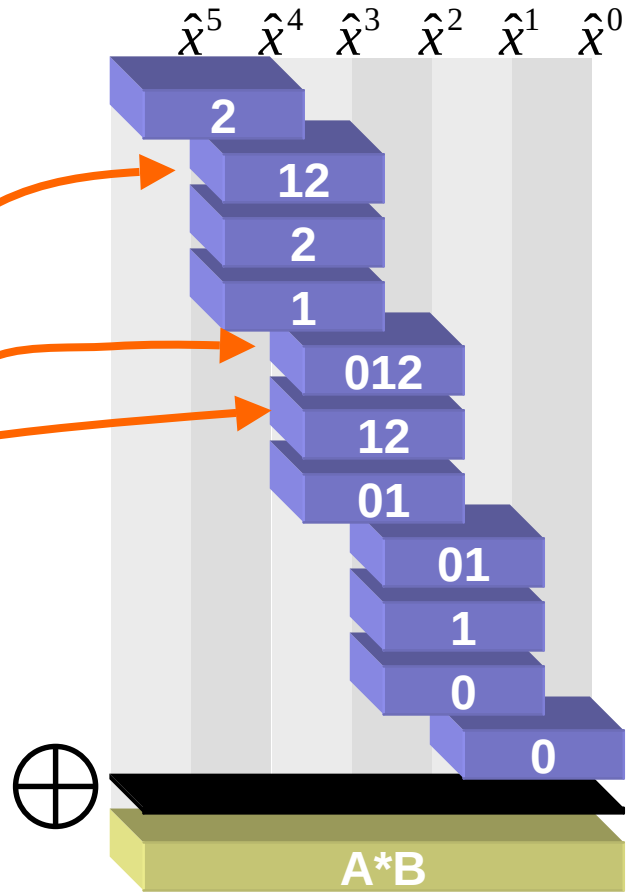
$$S_{1,l}(A,B) = M_{1,l}(A,B) \quad \text{and} \quad M_{m,l}(A,B) = \left(\bigoplus_{i=l}^{l+m-1} A_i \right) \cdot \left(\bigoplus_{i=l}^{l+m-1} B_i \right)$$

MSK_k for k=3

$$MSK_3 = (M_{1,2}) \cdot \hat{x}^4 \oplus$$
$$(M_{1,1} \oplus M_{1,2} \oplus M_{2,1}) \cdot \hat{x}^3 \oplus$$
$$(M_{2,0} \oplus M_{2,1} \oplus M_{3,0}) \cdot \hat{x}^2 \oplus$$
$$(M_{1,0} \oplus M_{1,1} \oplus M_{2,0}) \cdot \hat{x}^1 \oplus$$
$$(M_{1,0}) \cdot \hat{x}^0$$

with

$$M_{1,0} = A_0 \cdot B_0$$
$$M_{1,1} = A_1 \cdot B_1$$
$$M_{1,2} = A_2 \cdot B_2$$
$$M_{2,0} = (A_0 \oplus A_1) \cdot (B_0 \oplus B_1)$$
$$M_{2,1} = (A_1 \oplus A_2) \cdot (B_1 \oplus B_2)$$
$$M_{3,0} = (A_0 \oplus A_1 \oplus A_2) \cdot (B_0 \oplus B_1 \oplus B_2)$$

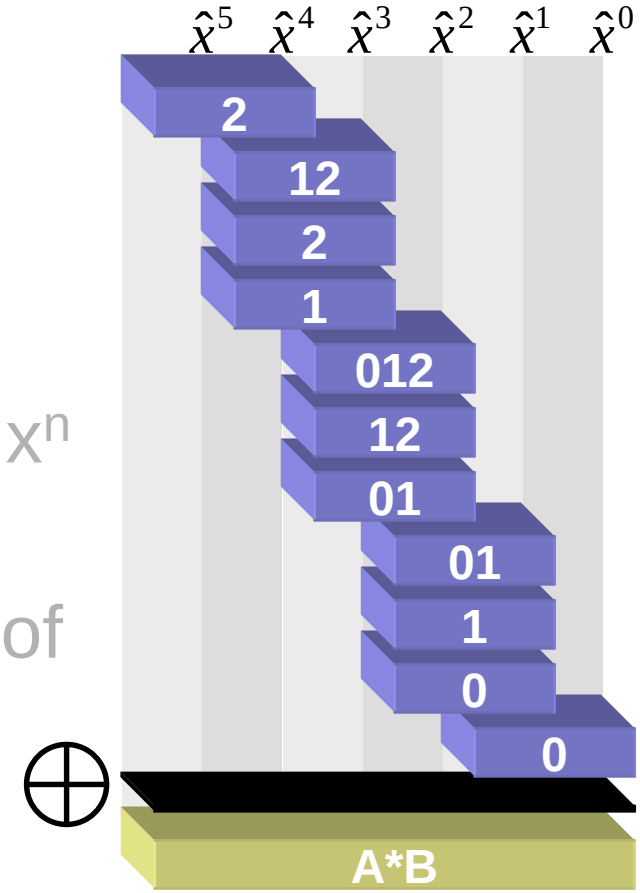


Reordering of Partial Products

1) Pattern Grouping

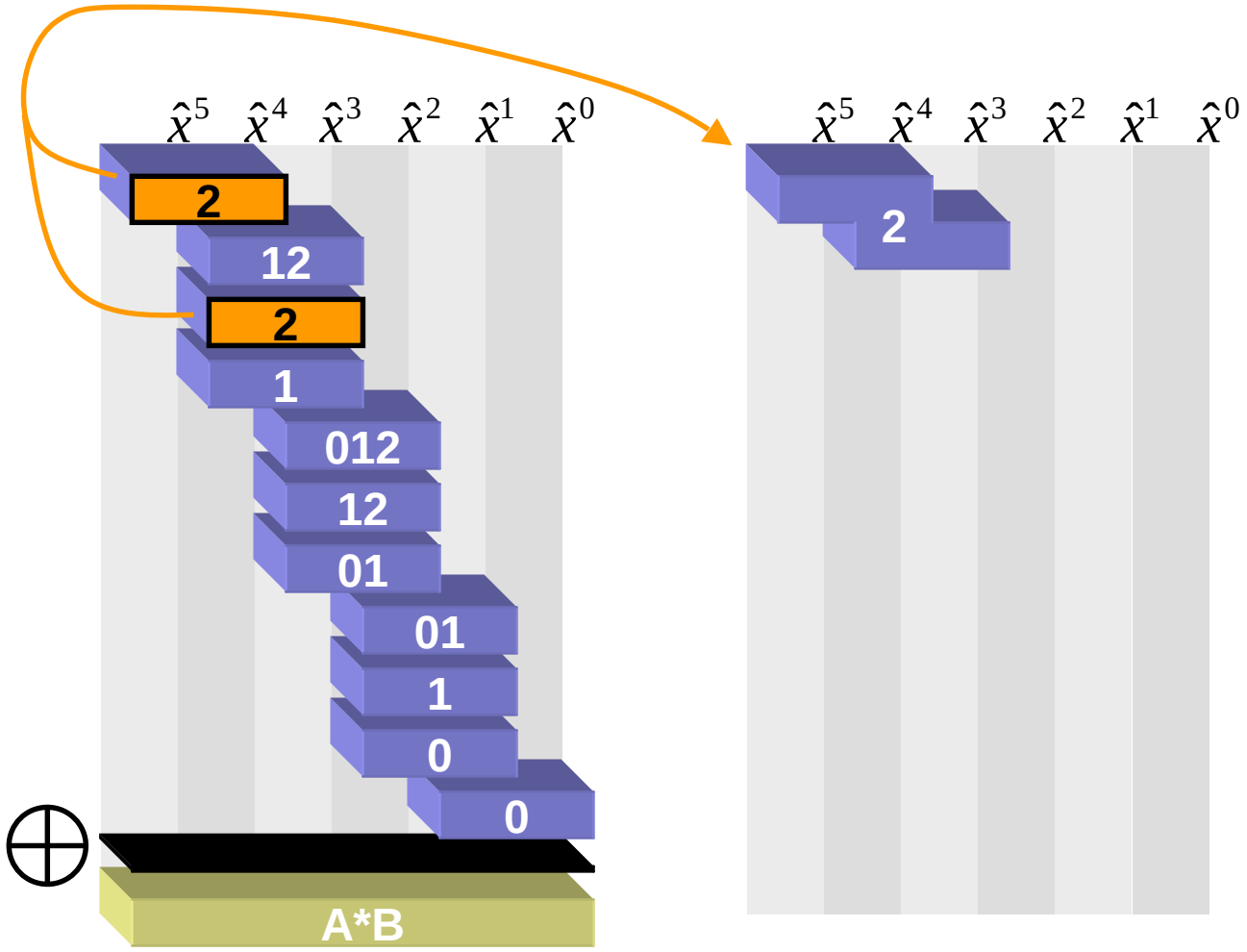
2) Reorder pattern by decreasing x^n

3) Minimize differences in the set of indices of adjacent pattern



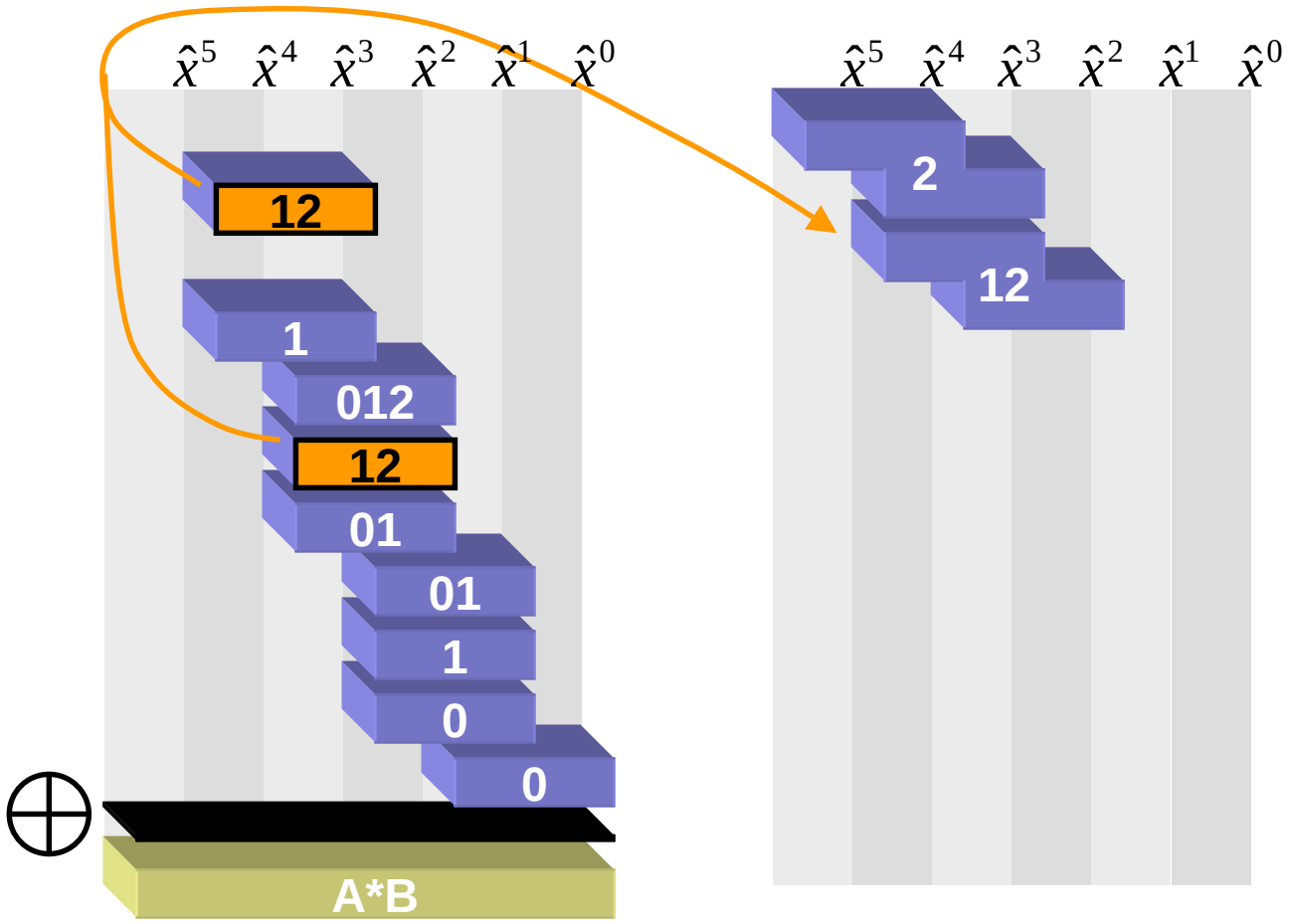
Pattern Grouping

1.) Grouping the partial products to one of three possible pattern



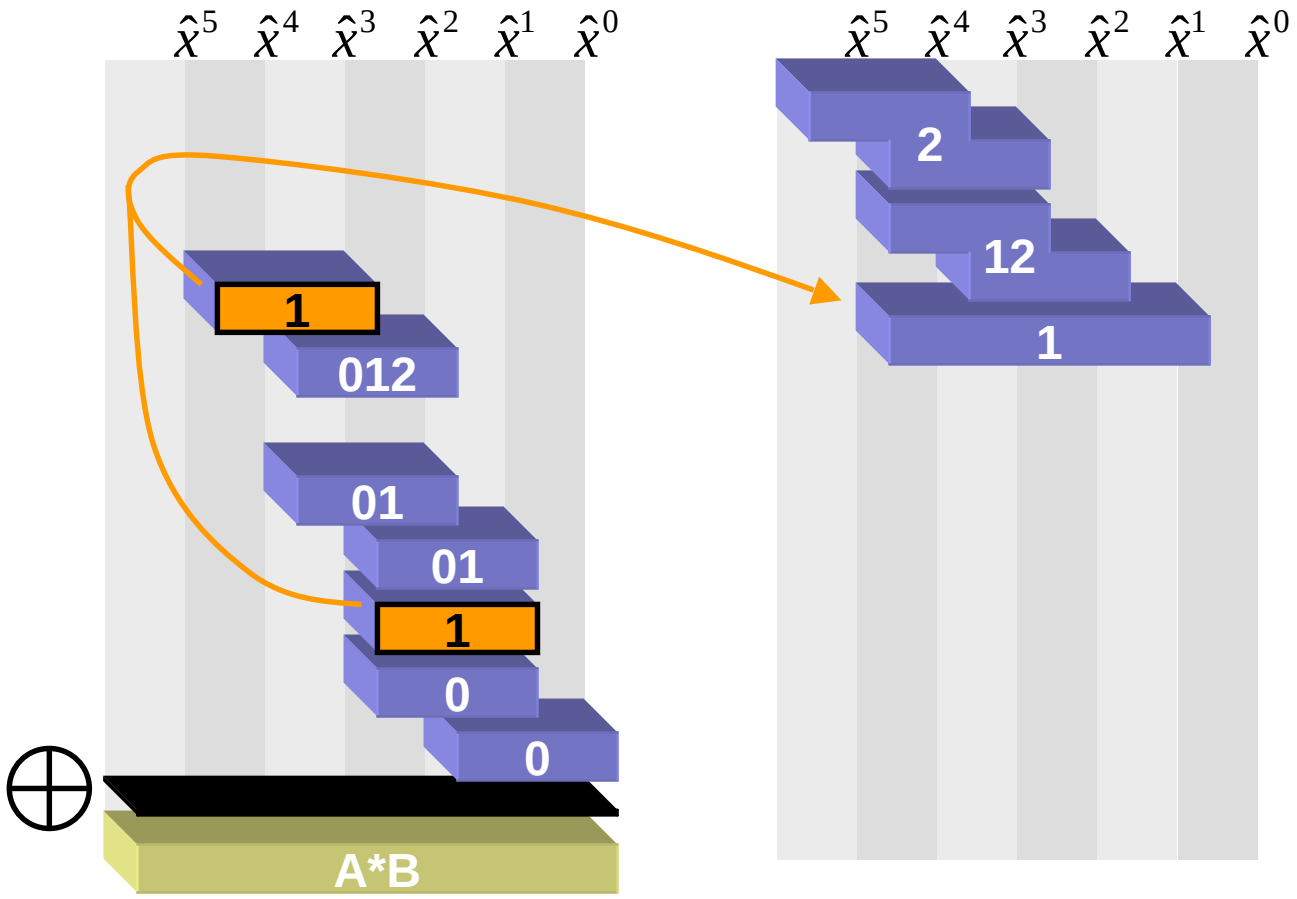
Pattern Grouping

1.) Grouping the partial products to one of three possible pattern



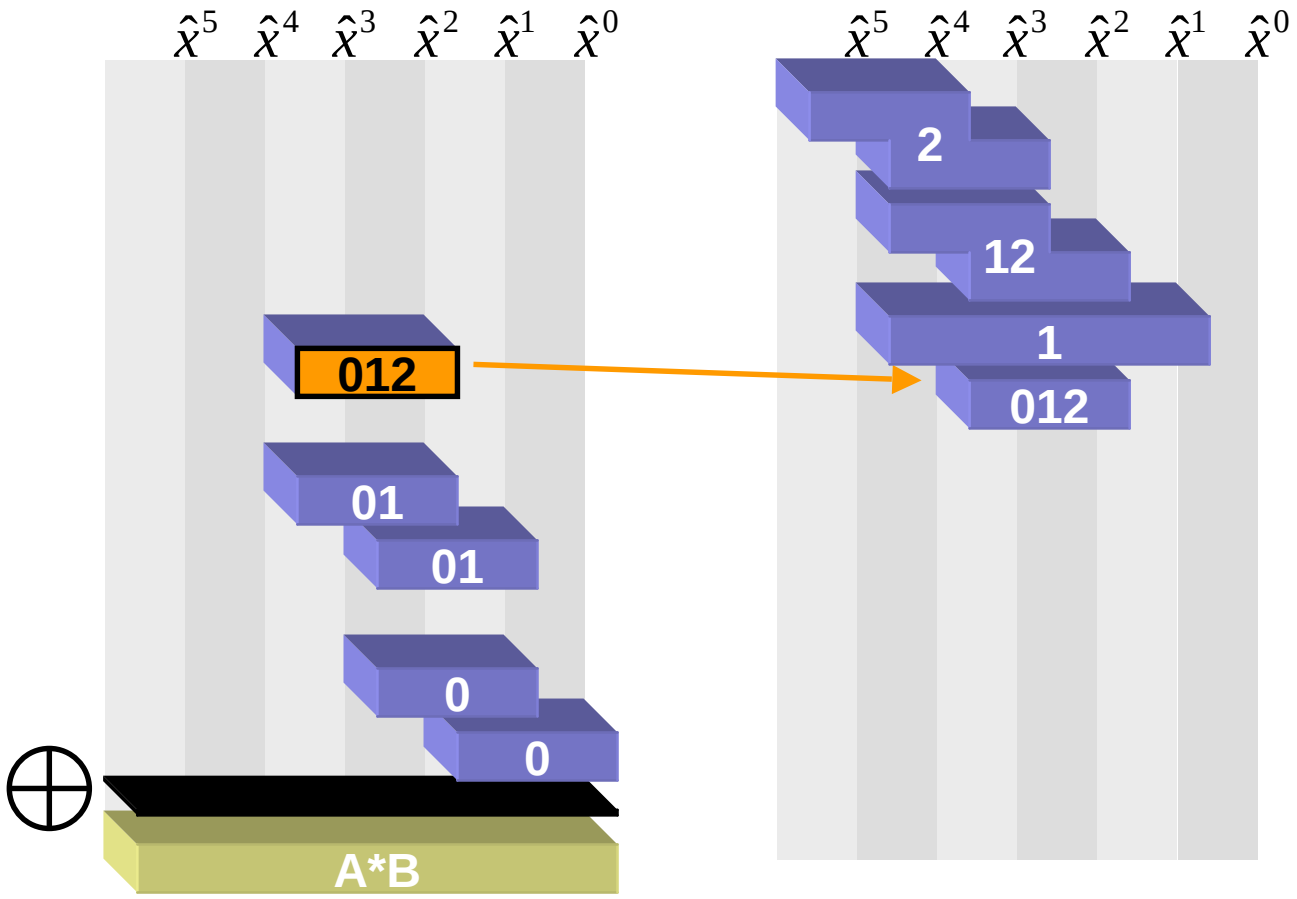
Pattern Grouping

1.) Grouping the partial products to one of three possible pattern



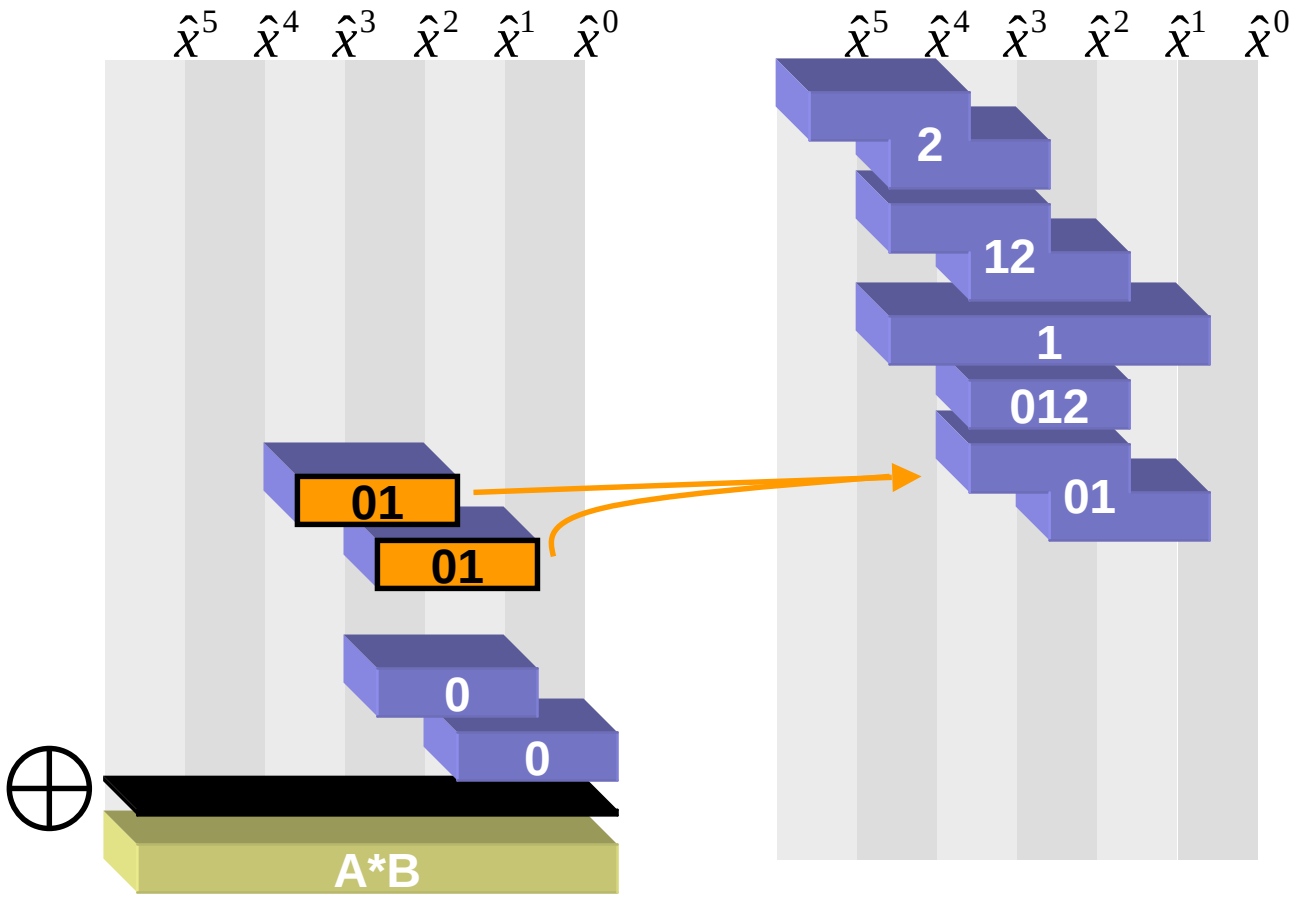
Pattern Grouping

1.) Grouping the partial products to one of three possible pattern



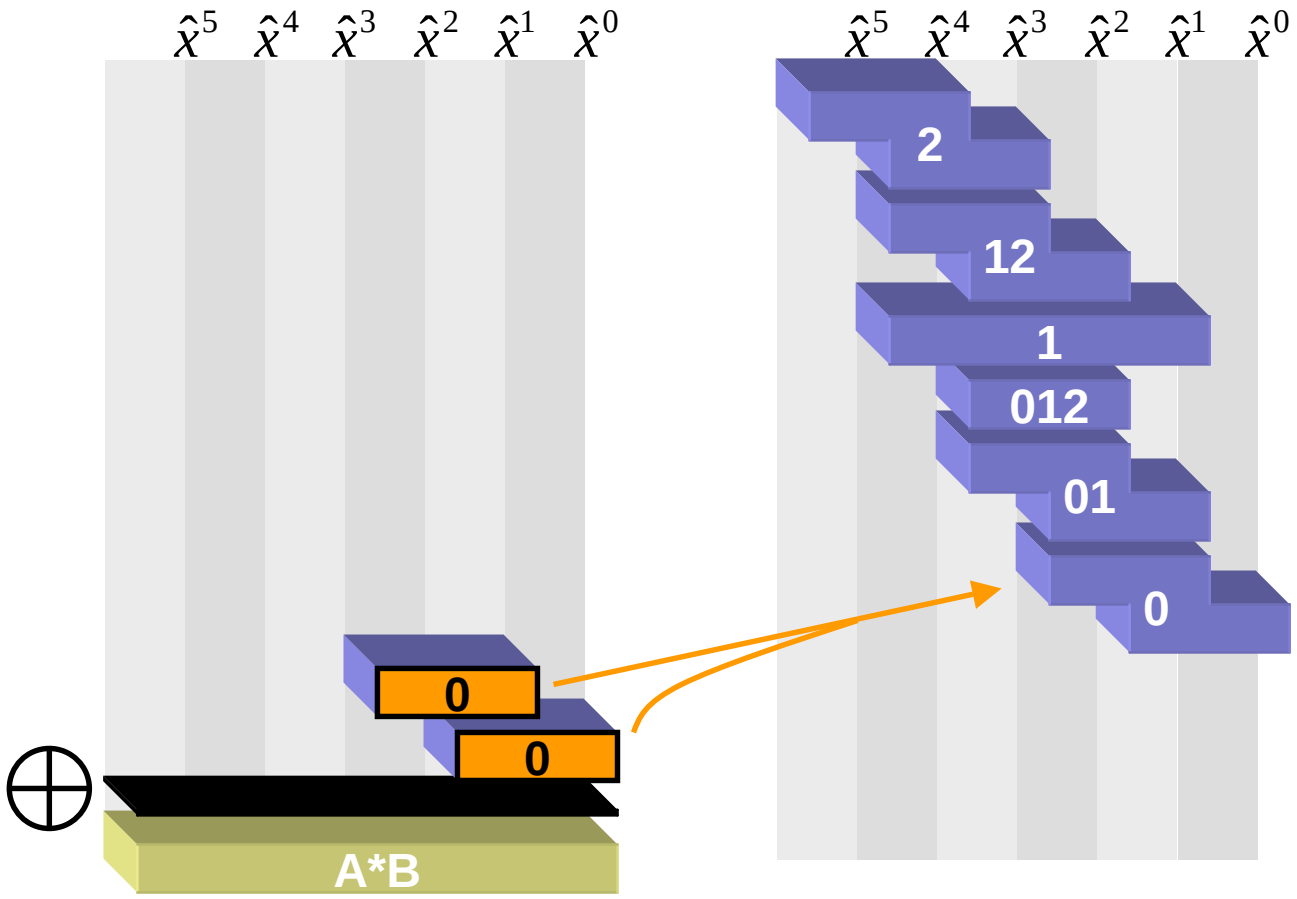
Pattern Grouping

1.) Grouping the partial products to one of three possible pattern



Pattern Grouping

1.) Grouping the partial products to one of three possible pattern

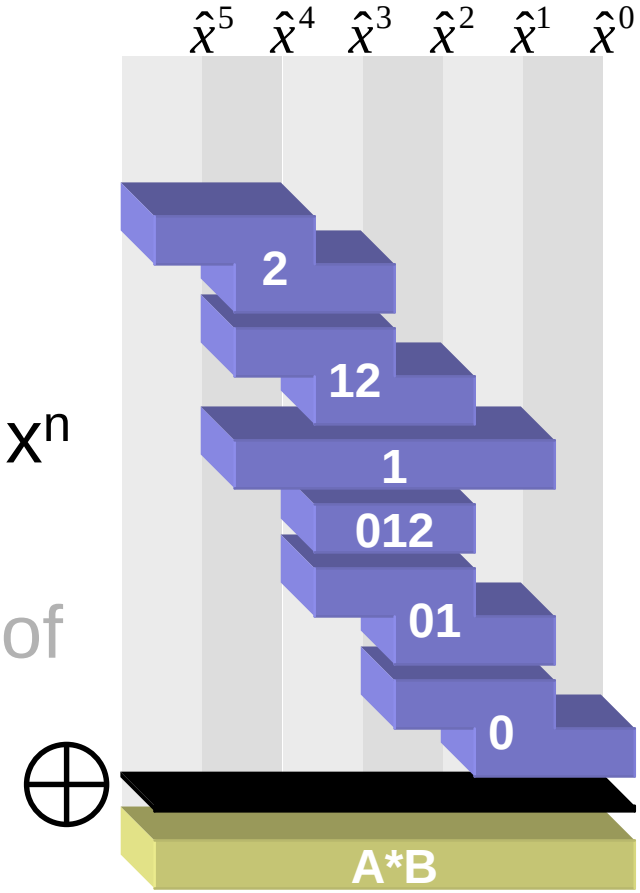


Reordering of Partial Products

1) Pattern Grouping

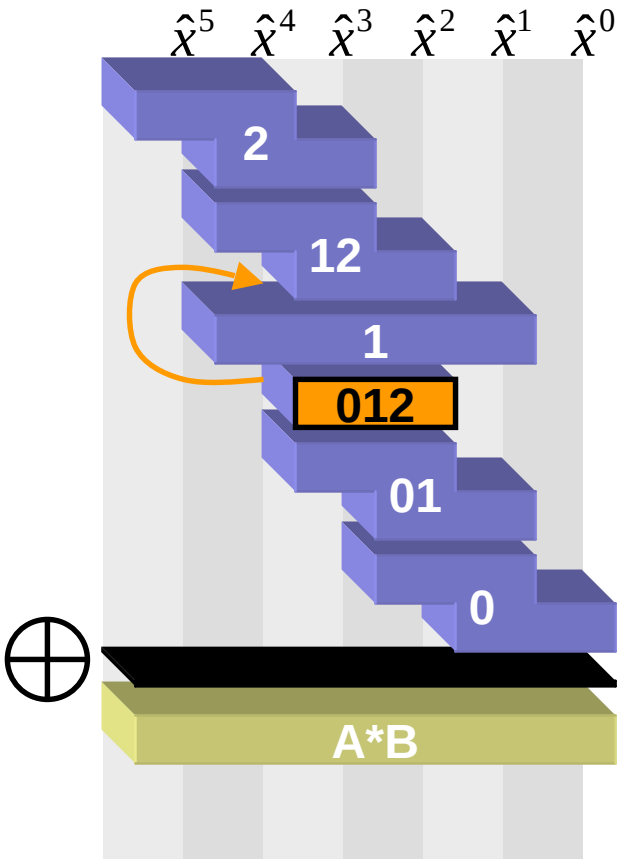
2) Reorder pattern by decreasing x^n

3) Minimize differences in the set of indices of adjacent pattern



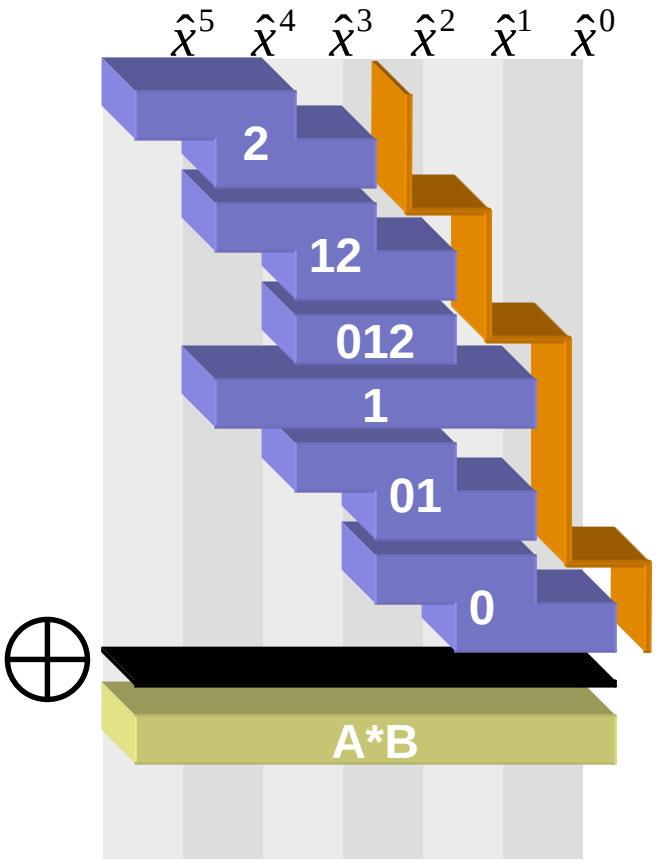
Reordering of Partial Products

2.) Ordering of the pattern in a top-left to bottom-right fashion



Reordering of Partial Products

2.) Ordering of the pattern in a top-left to bottom-right fashion

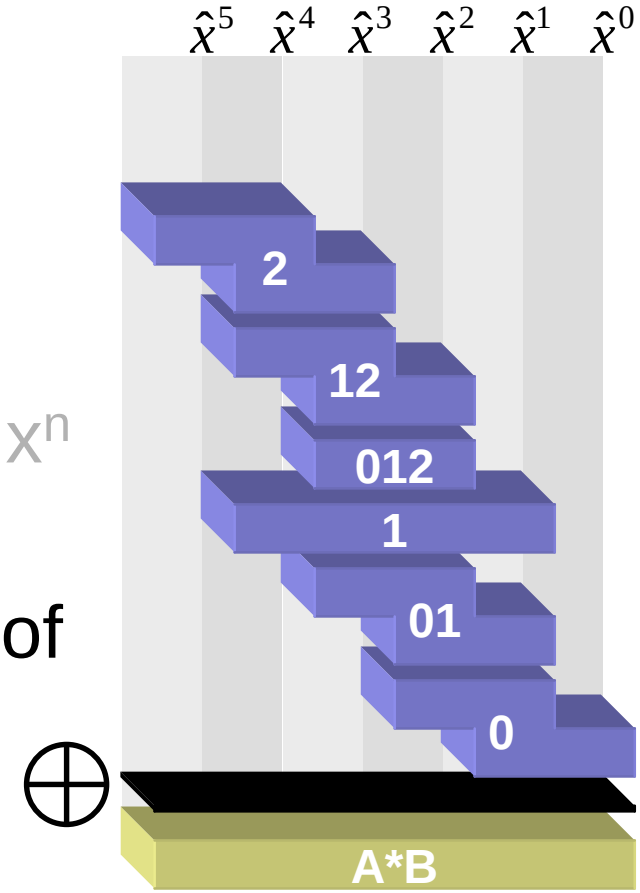


Reordering of Partial Products

1) Pattern Grouping

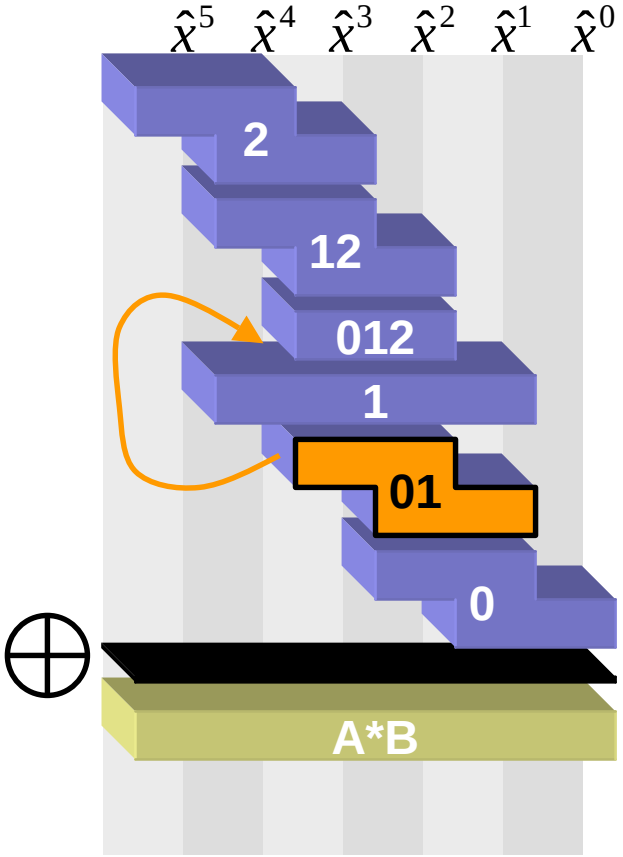
2) Reorder pattern by decreasing x^n

3) Minimize differences in the set of indices of adjacent pattern



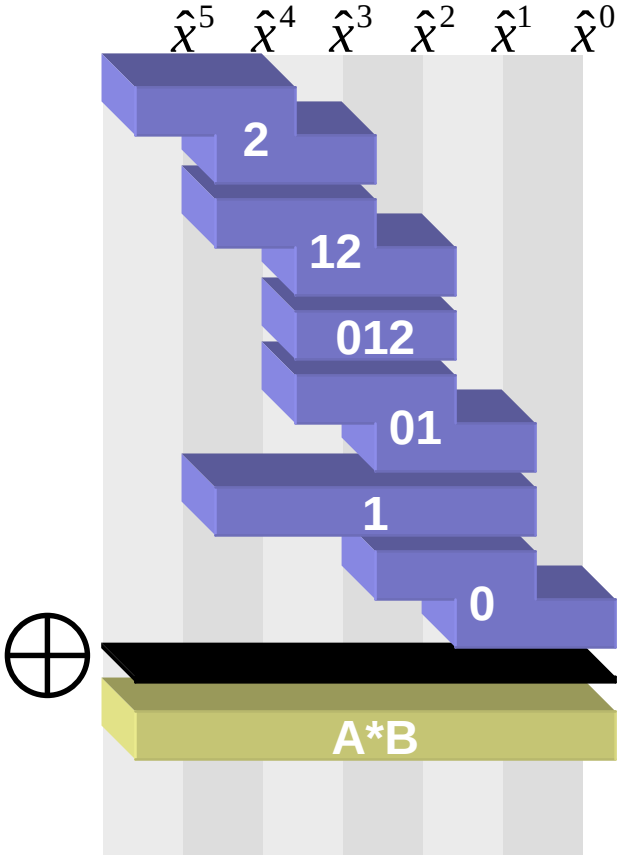
Reordering of Partial Products

3.) Ensuring that the set of added up segments in the partial products differs only by one element between two adjacent patterns

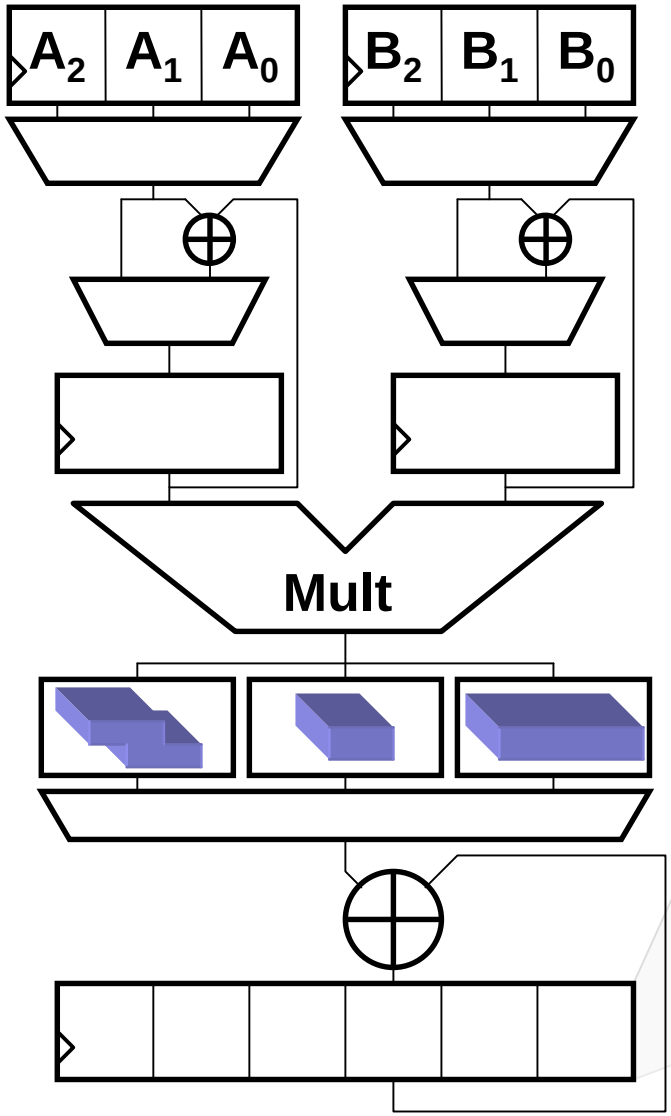


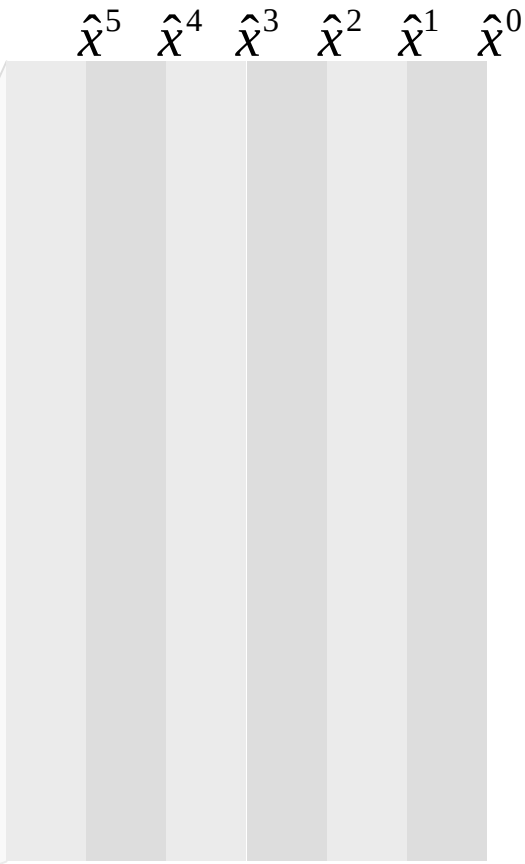
Reordering of Partial Products

3.) Ensuring that the set of added up segments in the partial products differs only by one element between two adjacent patterns

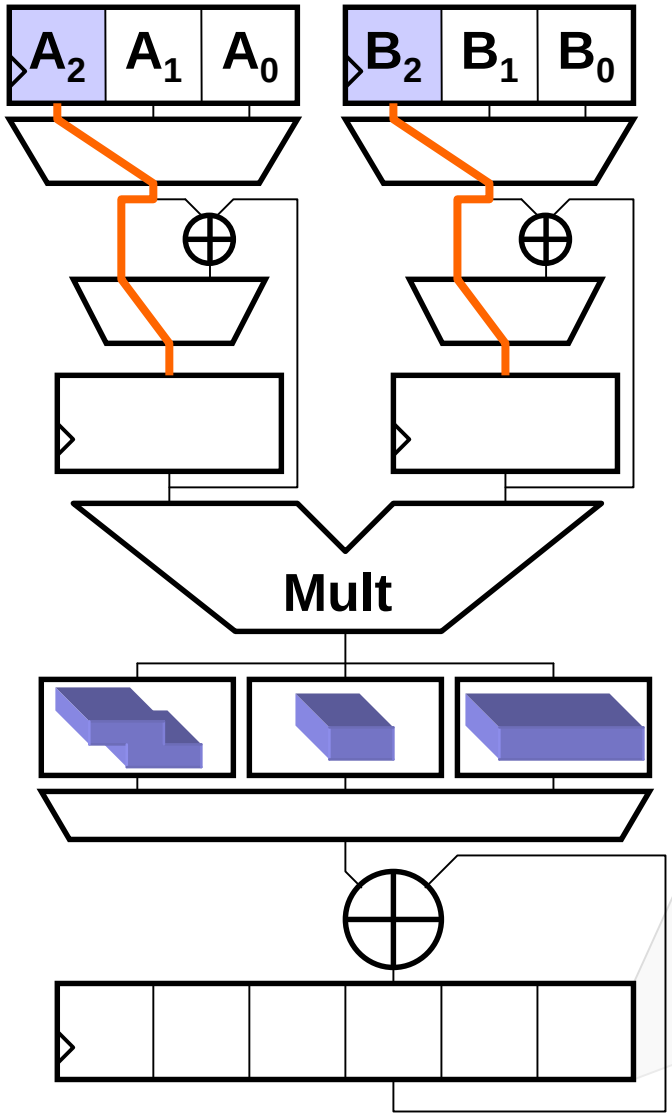


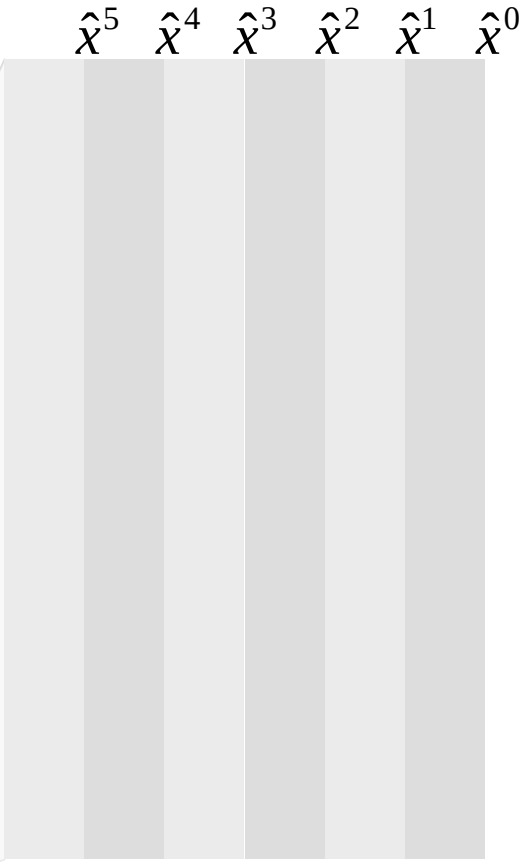
Multiplication Sequence





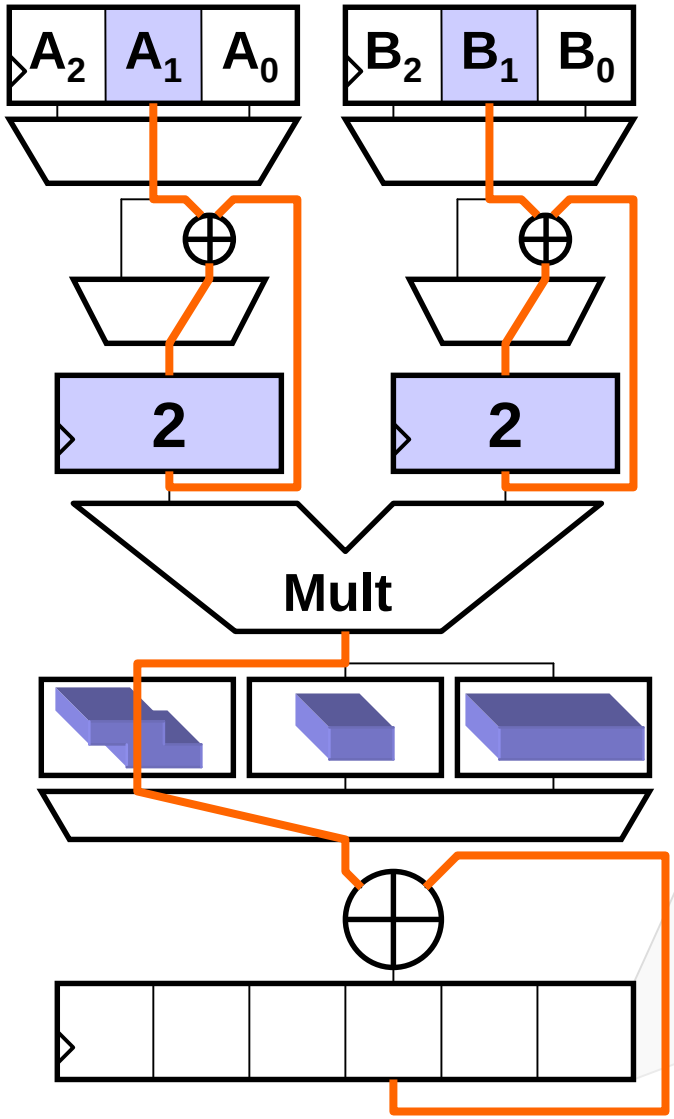
Multiplication Sequence

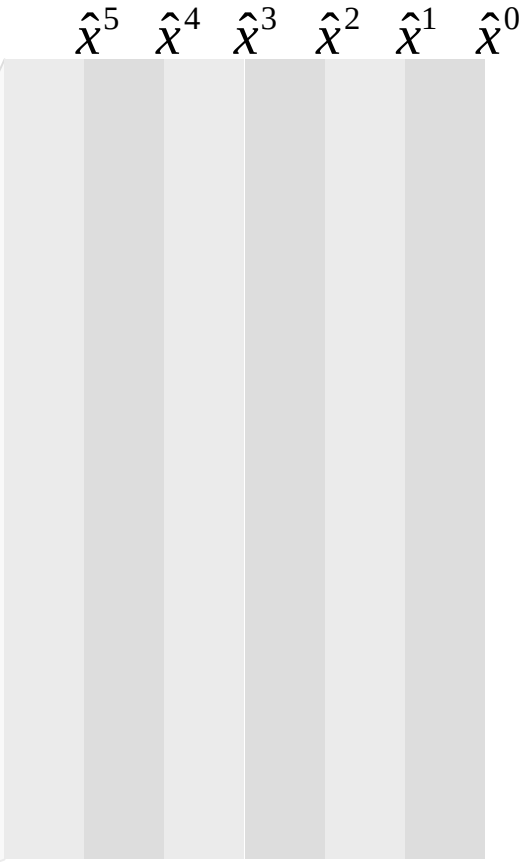




Clock Cycle: 1

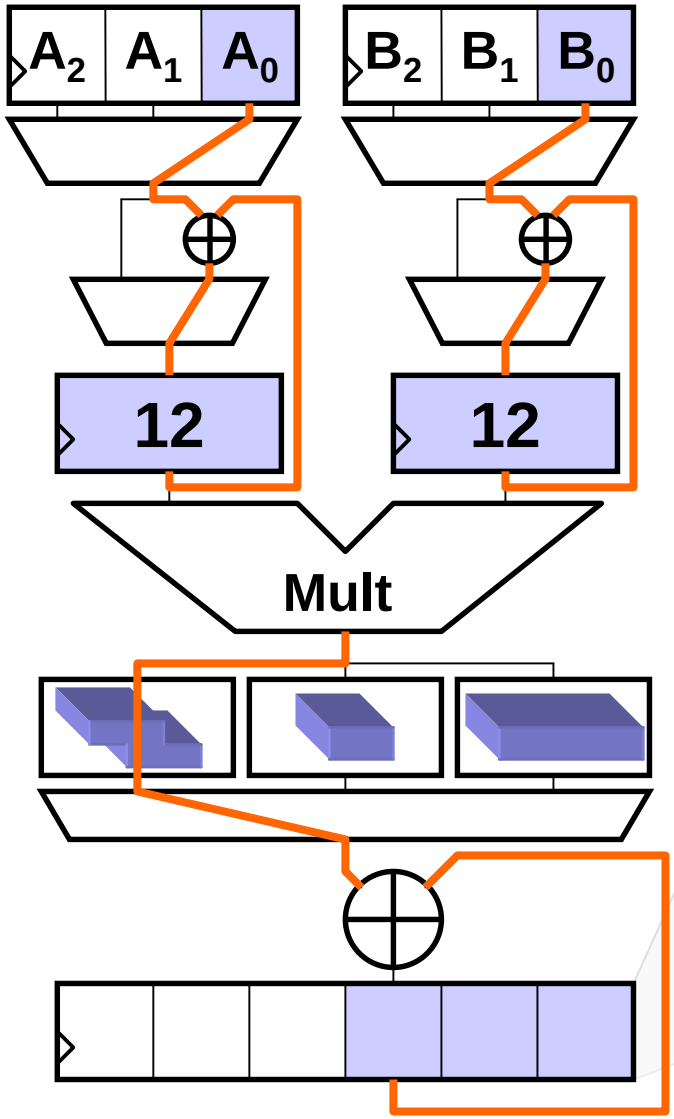
Multiplication Sequence

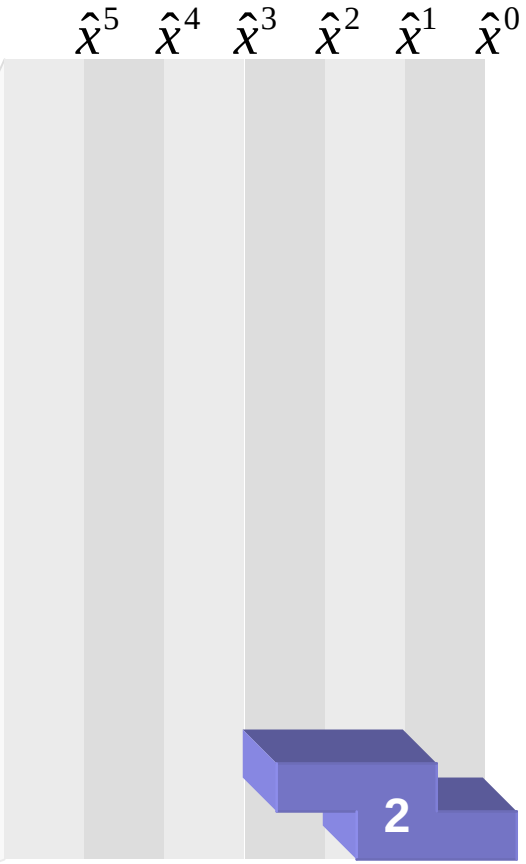




Clock Cycle: 2

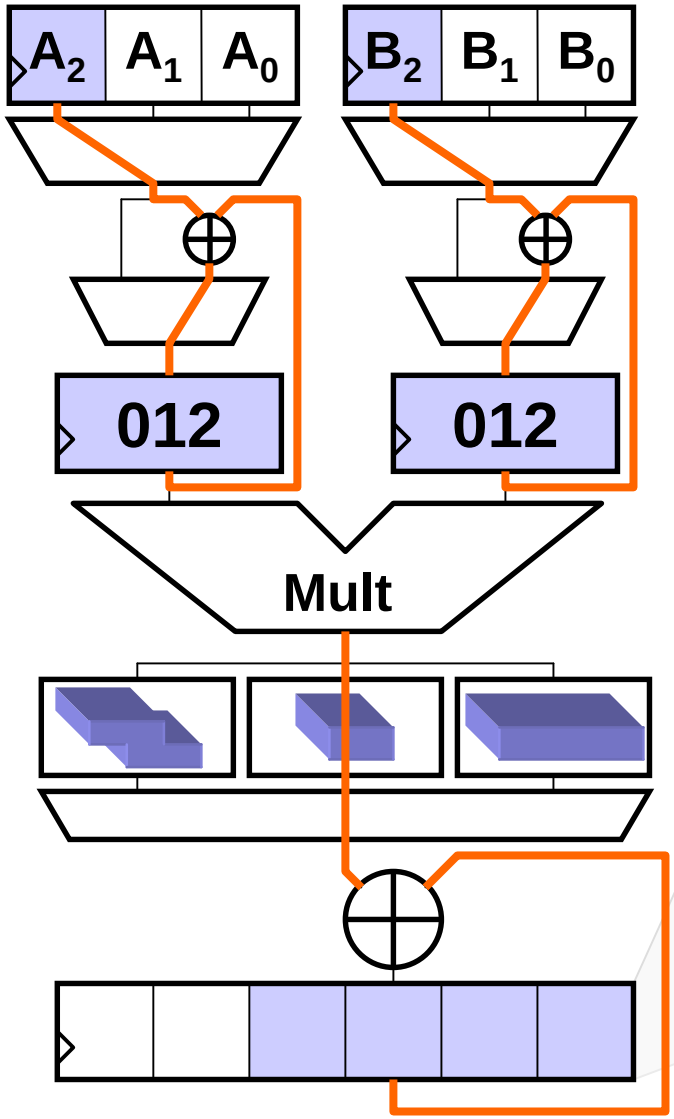
Multiplication Sequence

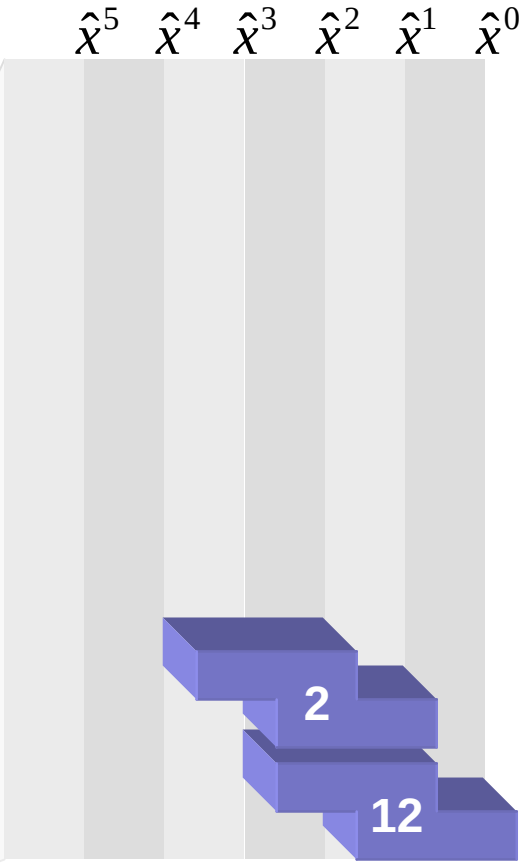




Clock Cycle: 3

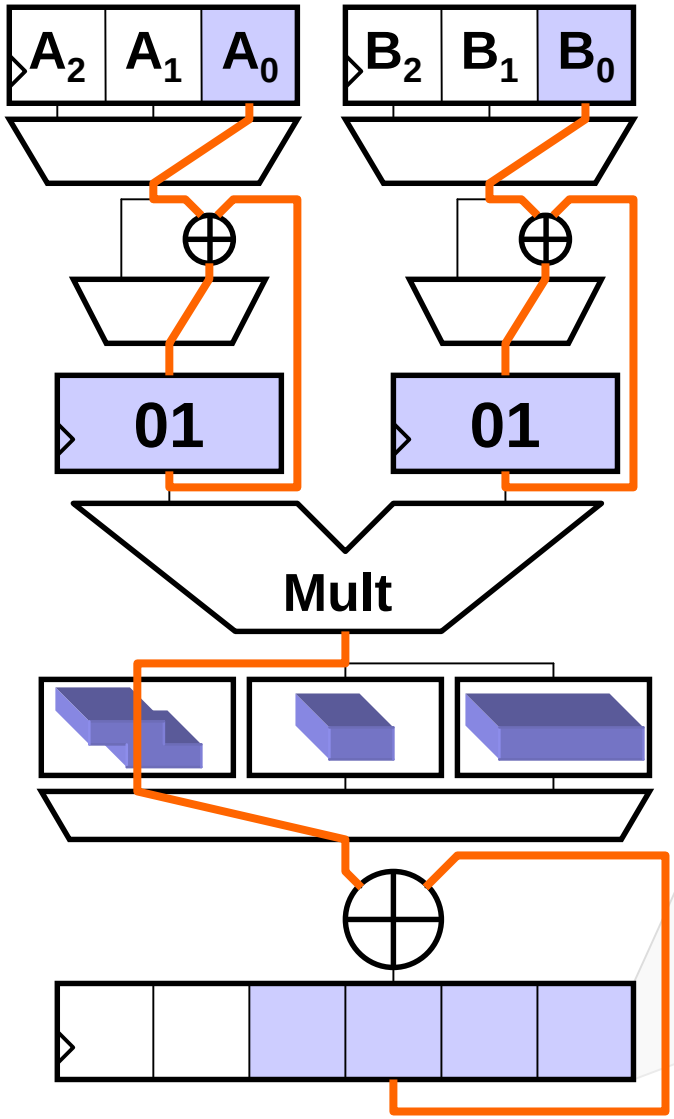
Multiplication Sequence

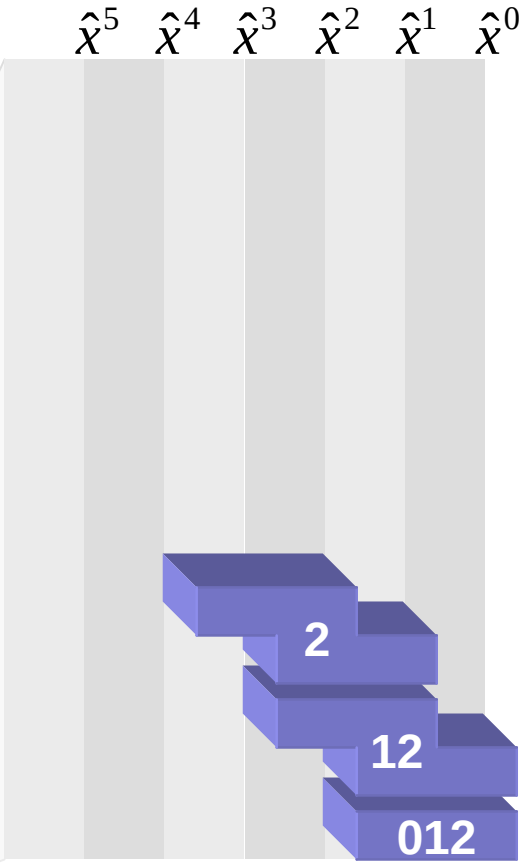




Clock Cycle: 4

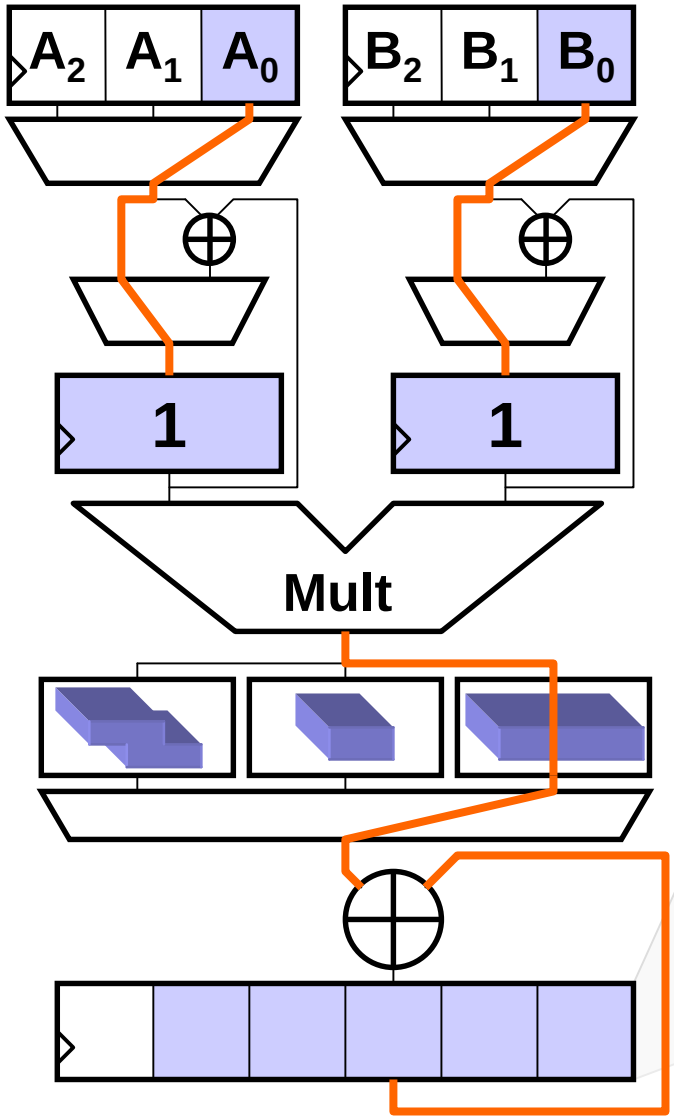
Multiplication Sequence

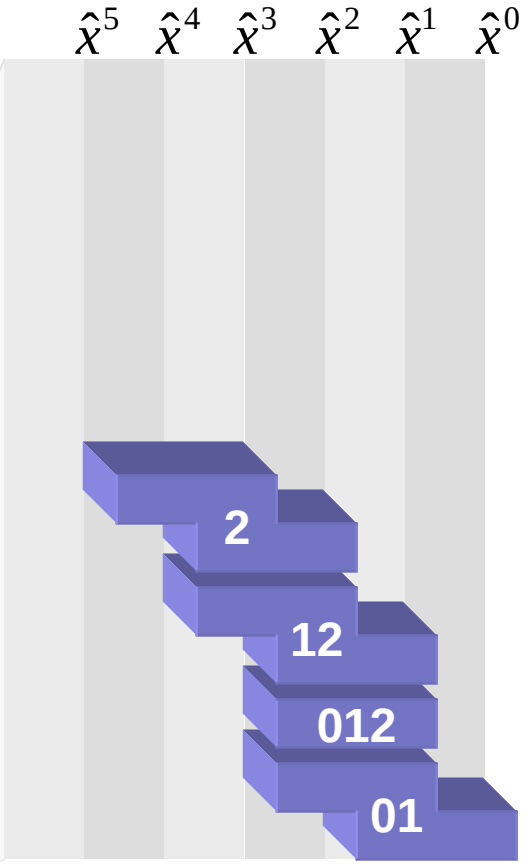




Clock Cycle: 5

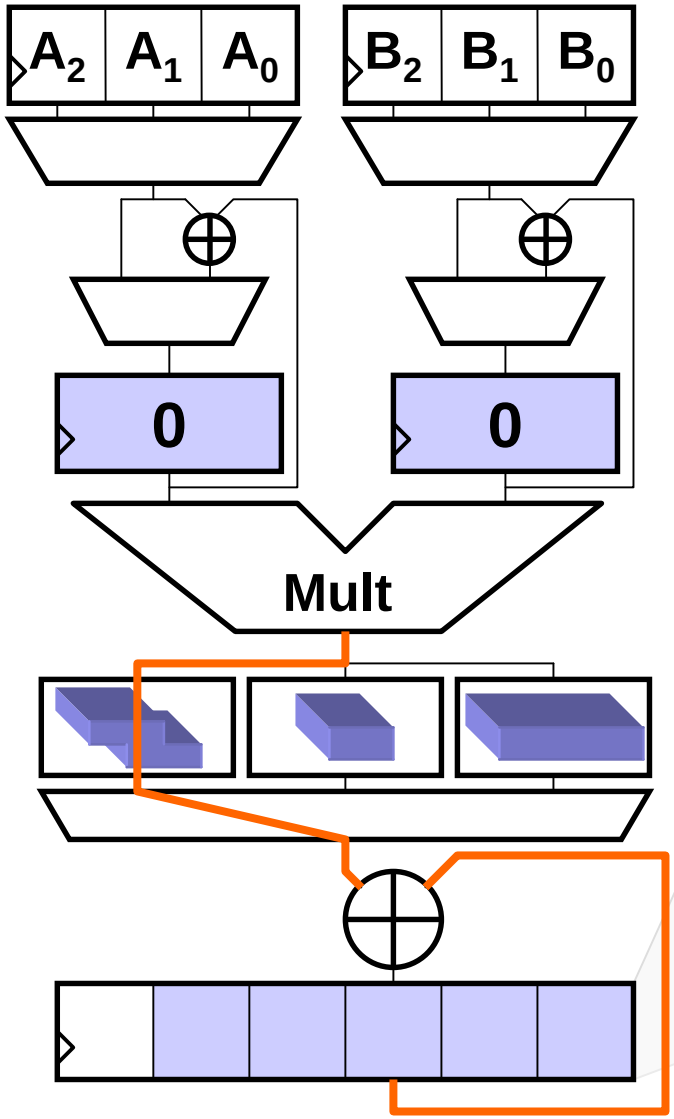
Multiplication Sequence

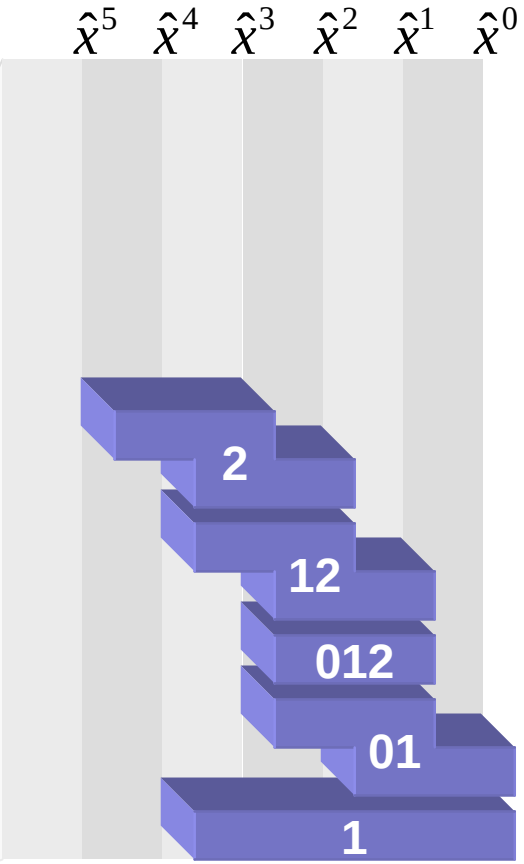




Clock Cycle: 6

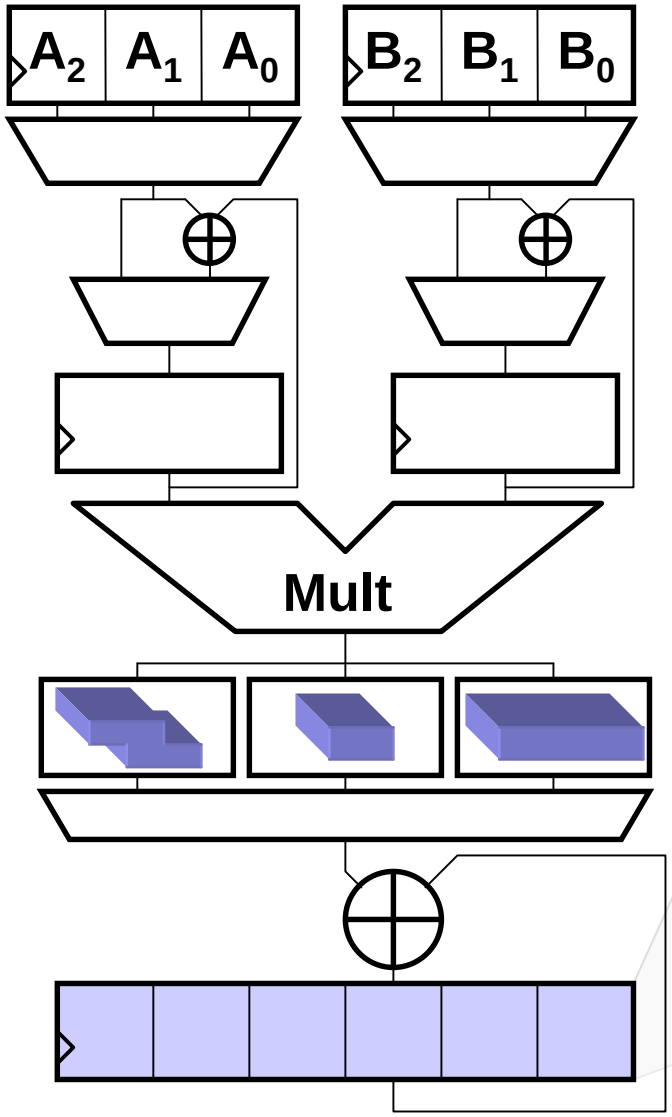
Multiplication Sequence

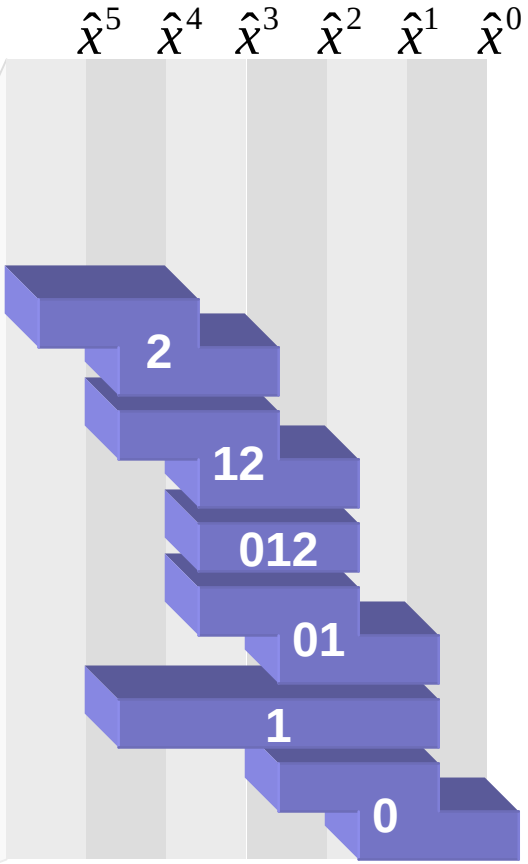




Clock Cycle: 7

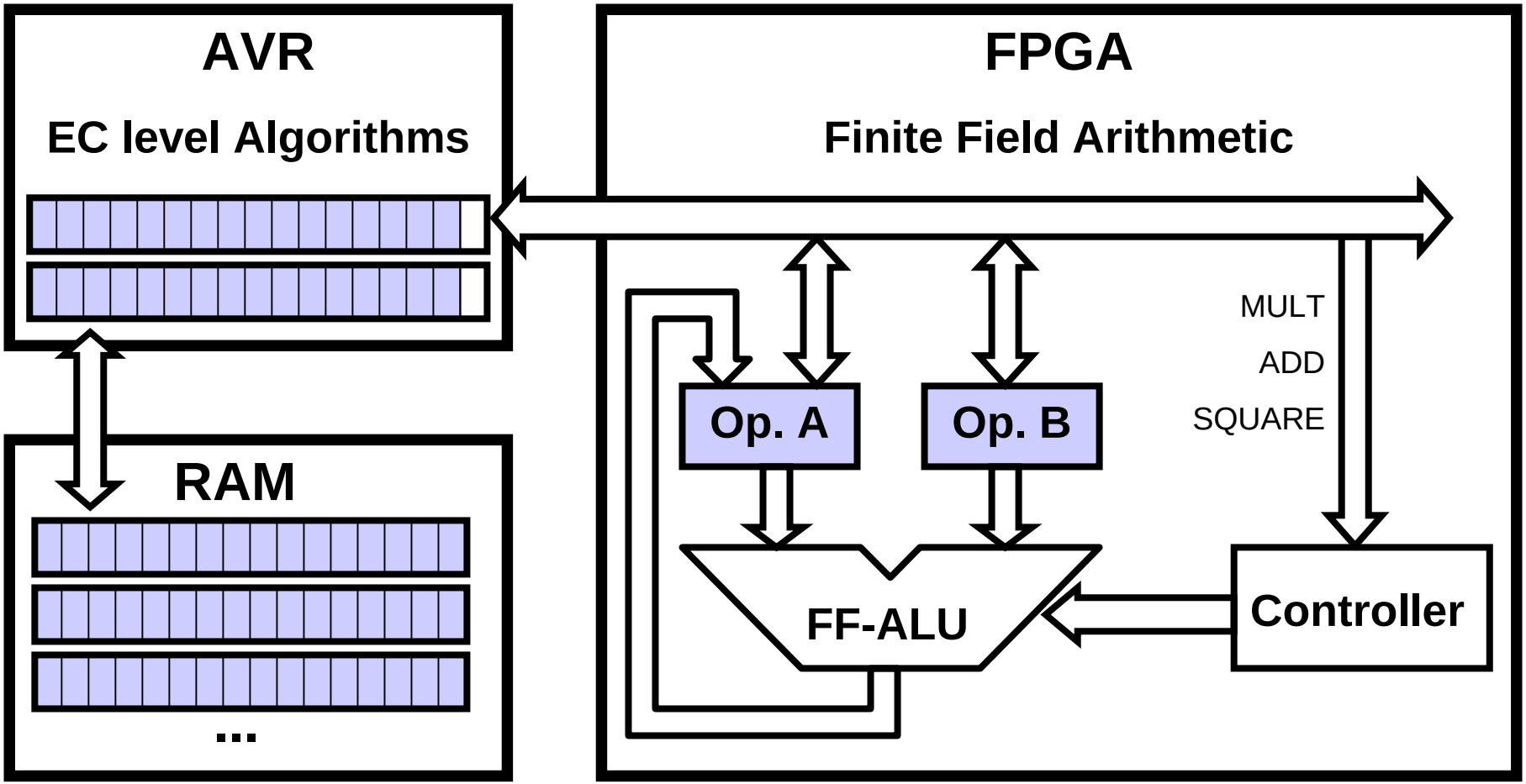
Multiplication Sequence





Clock Cycle: 8

Coprocessor Interface



Results

Prototype Implementation:

- **5-Segment Karatsuba (MSK_5)**
- **23-Bit combinational Multiplier**
- **GF(2^{113})**

FF-Level Operation	Clock Cycles best case	Clock Cycles worst case
FF-Mult	32	152
FF-Add	16	136
FF-Square	1	91

Operation	Clock Cycles
EC-Double	493
EC-Add	615
k·P	130,200

- **One EC point multiplication takes 10.9 ms @ 12 MHz**
- **Speed-up factor of about 40 compared to an assembler optimized software implementation**