New Efficient Hardware Architectures for Montgomery Modular Multiplication

Miaoqing Huang[1], Kris Gaj[2],
Marcin Rogawski[2]

[1] The George Washington University, Washington, D.C., U.S.A.
[2] George Mason University, Fairfax, VA, U.S.A.

Outline

- Motivation
- Classical Hardware Architecture for Montgomery Multiplication by Tenca and Koc from CHES 1999
- Our Optimized Hardware Architecture
- Conceptual Comparison
- Implementation Results
- Possible Extensions
- Conclusions

Motivation

- Fast modular multiplication required in multiple cryptographic transformations
    - RSA, DSA, Diffie-Hellman
    - Elliptic Curve Cryptosystems
    - ECM, p-1, Pollard's rho methods of factoring, etc.

- Montgomery Multiplication invented by Peter L. Montgomery in 1985 is most frequently used to implement repetitive sequence of modular multiplications in both software and hardware

- Montgomery Multiplication in hardware replaces division by a sequence of simple logic operations, conditional additions and right shifts

Montgomery Modular Multiplication (1)

$Z = X \cdot Y \bmod M$ X, Y, M – n-bit numbers

Integer domain **Montgomery domain**

$X \longrightarrow X' = X \cdot 2^n \bmod M$

$Y \longrightarrow Y' = Y \cdot 2^n \bmod M$

$$Z' = MP(X', Y', M) =$$
$$= X' \cdot Y' \cdot 2^{-n} \bmod M =$$
$$= (X \cdot 2^n) \cdot (Y \cdot 2^n) \cdot 2^{-n} \bmod M =$$
$$= X \cdot Y \cdot 2^n \bmod M$$

$Z = X \cdot Y \bmod M \longleftarrow Z' = Z \cdot 2^n \bmod M$

Montgomery Modular Multiplication (2)

$X \longrightarrow X'$

$$X' = MP(X, 2^{2n} \bmod M, M) =$$
$$= X \cdot 2^{2n} \cdot 2^{-n} \bmod M = X \cdot 2^{n} \bmod M$$

$Z \longleftarrow Z'$

$$Z = MP(Z', 1, M) =$$
$$= (Z \cdot 2^{n}) \cdot 1 \cdot 2^{-n} \bmod M = Z \bmod M = Z$$

Montgomery Product

$S[0] = 0$

for i=0 to n-1

$$S[i+1] = \begin{cases} \dfrac{S[i]+x_i \cdot Y}{2} & \text{if} \quad q_i = S[i] + x_i \cdot Y \bmod 2 = 0 \\ \dfrac{S[i]+x_i \cdot Y + M}{2} & \text{if} \quad q_i = S[i] + x_i \cdot Y \bmod 2 = 1 \end{cases}$$

$Z = S[n]$

M assumed to be odd

Classical Design by Tenca & Koc CHES 1999

Multiple Word Radix-2 Montgomery Multiplication algorithm (MWR2MM)

Main ideas:

Use of short precision words (w-bit each):

- Reduces broadcast problem in circuit implementation
- Word-oriented algorithm provides the support needed to develop scalable hardware units.

Operand Y(multiplicand) is scanned word-by-word,
operand X(multiplier) is scanned bit-by-bit.

Classical Design by Tenca & Koc CHES 1999

Each word has w bits

Each operand has

n bits

e words

$$e = \left\lceil \frac{n+1}{w} \right\rceil$$

$$X = (x_{n-1}, \ldots, x_1, x_0)$$

$$Y = (Y^{(e-1)}, \ldots, Y^{(1)}, Y^{(0)})$$

$$M = (M^{(e-1)}, \ldots, M^{(1)}, M^{(0)})$$

The bits are marked with subscripts, and the words are marked with superscripts.

MWR2MM
Multiple Word Radix-2 Montgomery Multiplication algorithm by Tenca and Koc

Algorithm 2 The Multiple-Word Radix-2 Montgomery Multiplication Algorithm

Require: odd $M, n = \lfloor \log_2 M \rfloor + 1$, word size w, $e = \lceil \frac{n+1}{w} \rceil$, $X = \sum_{i=0}^{n-1} x_i \cdot 2^i$, $Y = \sum_{j=0}^{e-1} Y^{(j)} \cdot 2^{w \cdot j}$, $M = \sum_{j=0}^{e-1} M^{(j)} \cdot 2^{w \cdot j}$, with $0 \leq X, Y < M$

Ensure: $Z = \sum_{j=0}^{e-1} S^{(j)} \cdot 2^{w \cdot j} = MP(X, Y, M) \equiv X \cdot Y \cdot 2^{-n} \pmod{M}, 0 \leq Z < 2M$

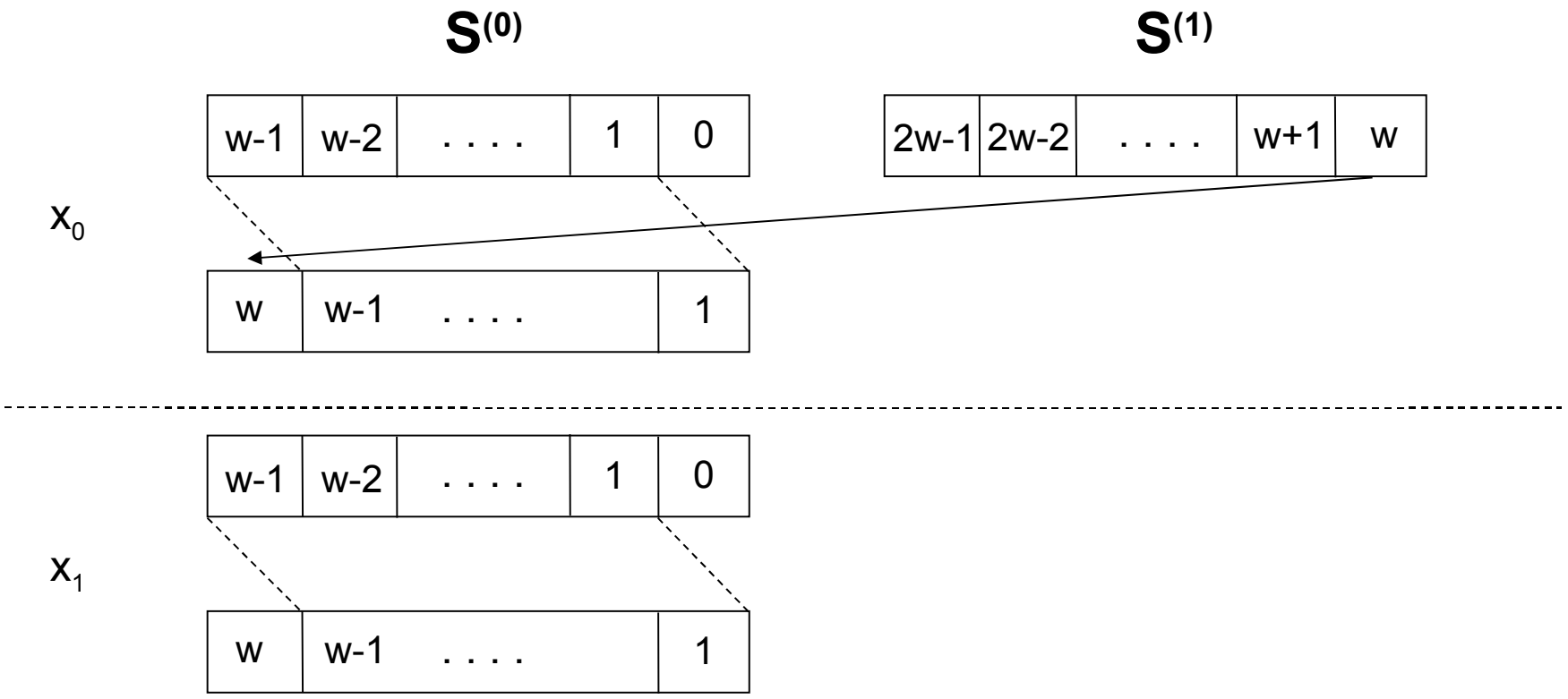
1: $S = 0$ — *initialize all words of S*
2: **for** $i = 0$ to $n - 1$ step 1 **do**
3: $q_i = (x_i \cdot Y_0^{(0)}) \oplus S_0^{(0)}$
4: $(C^{(1)}, S^{(0)}) = x_i \cdot Y^{(0)} + q_i \cdot M^{(0)} + S^{(0)}$
5: **for** $j = 1$ to $e - 1$ step 1 **do**
6: $(C^{(j+1)}, S^{(j)}) = C^{(j)} + x_i \cdot Y^{(j)} + q_i \cdot M^{(j)} + S^{(j)}$
7: $S^{(j-1)} = (S_0^{(j)}, S_{w-1..1}^{(j-1)})$
8: **end for**
9: $S^{(e-1)} = (C_0^{(e)}, S_{w-1..1}^{(e-1)})$
10: **end for**
11: return $Z = S$

Task A

Task B **e-1 times**

Task C

Problem in Parallelizing Computations



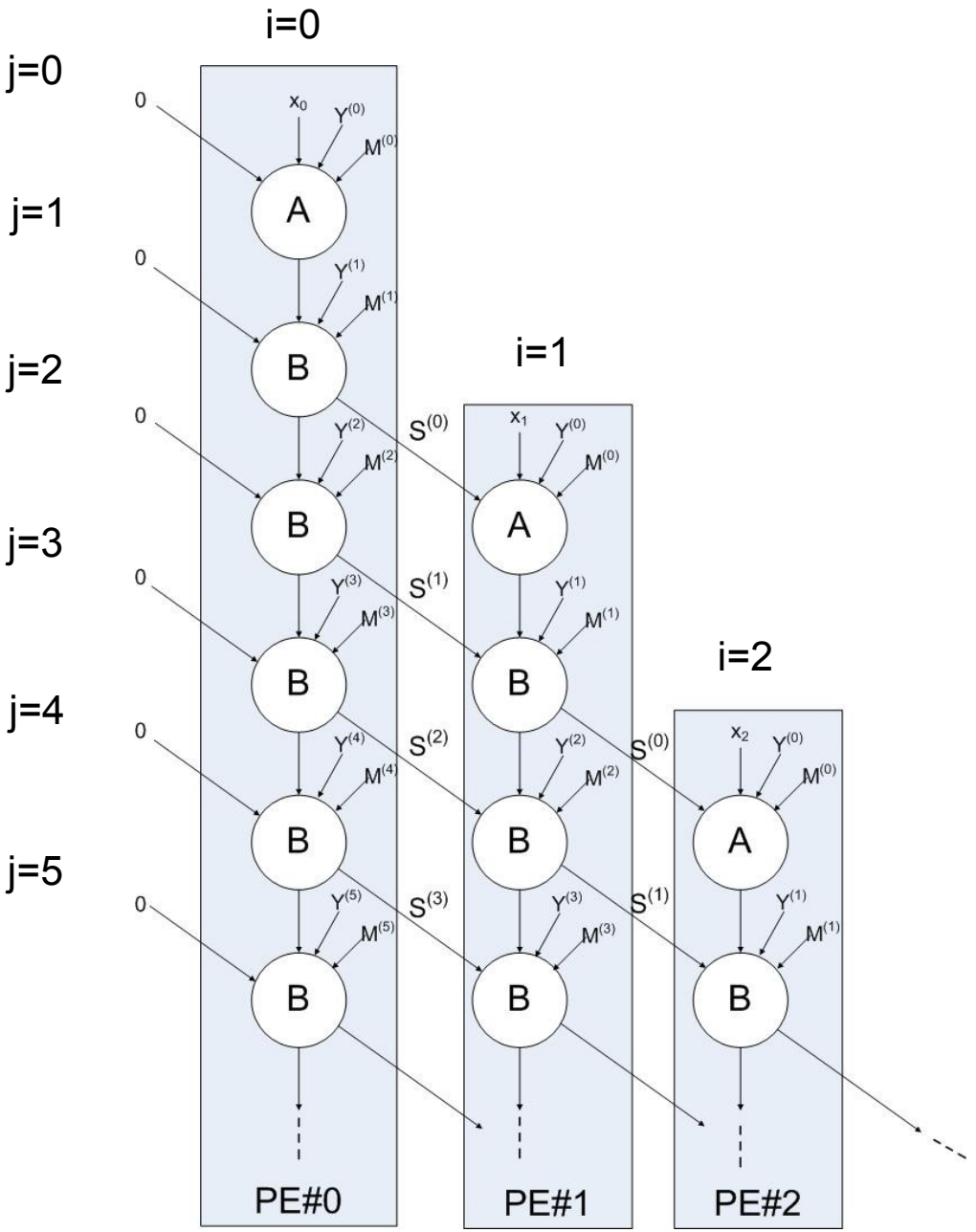
Data Dependency Graph by Tenca & Koc

- One PE is in charge of the computation of one column that corresponds to the updating of *S* with respect to one single bit x_i.
- The delay between two adjacent PEs is 2 clock cycles.
- The minimum computation time is

  $2 \cdot n + e - 1$ clock cycles
- given

  $(e+1)/2$ PEs

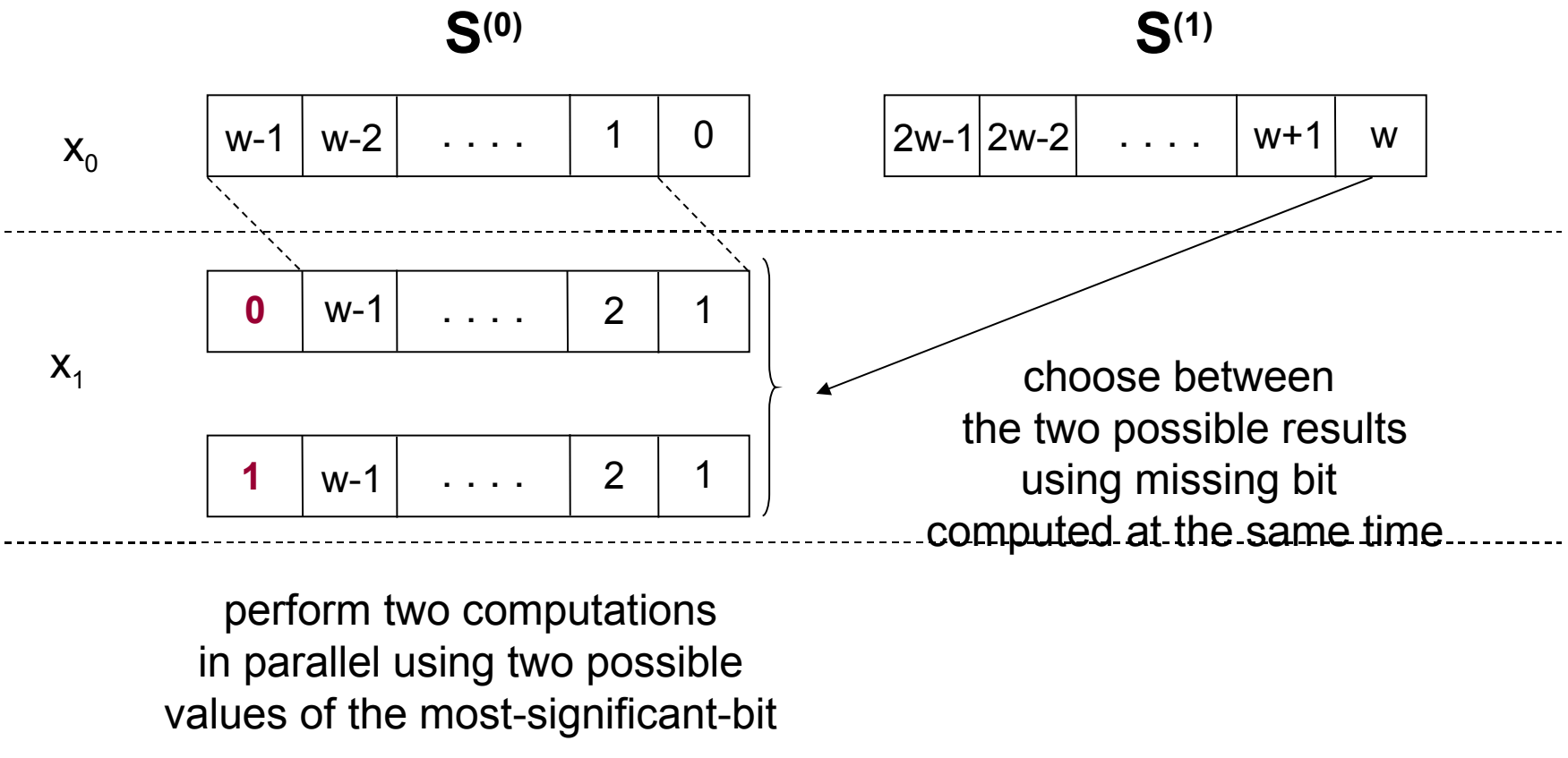
  working in parallel.



Main Idea of the New Architecture

- In the architecture of Tenca & Koc
  - w-1 least significant bits of partial results $S^{(j)}$ are available one clock cycle before they are used
  - only one (most significant) bit is missing
- Let us compute a new partial result under two assumptions regarding the value of the most significant bit of $S^{(j)}$ and choose the correct value one clock cycle later

Idea for a Speed-up



Primary Advantage of the New Approach

- Reduction in the number of clock cycles

  from

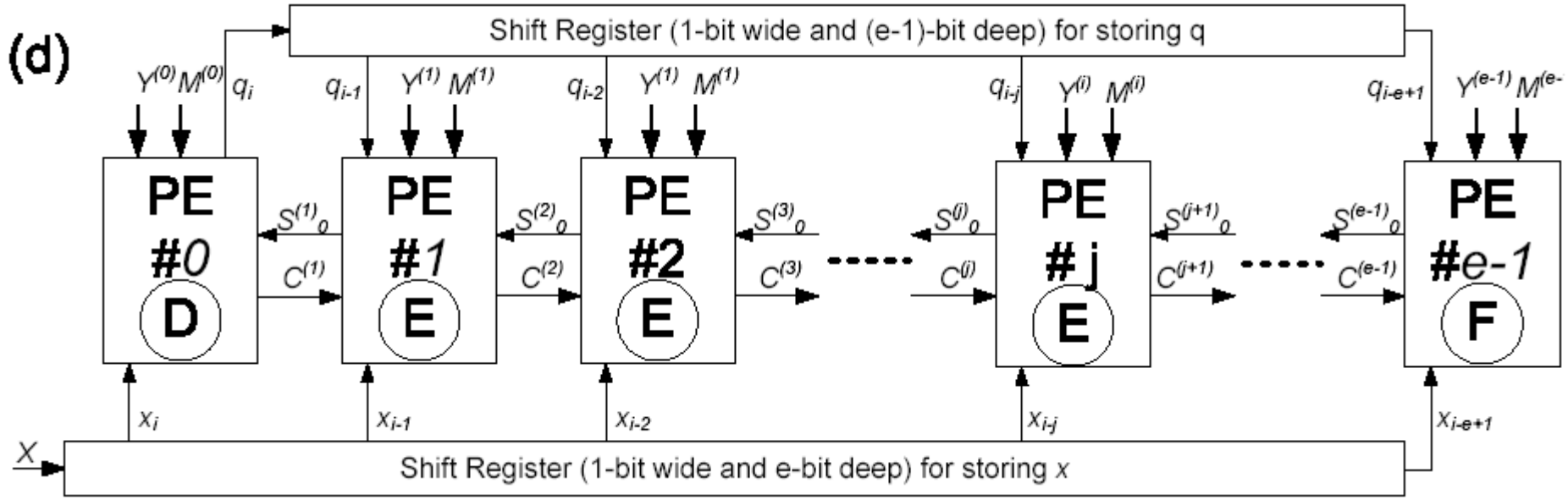
  2 n + e - 1

  to

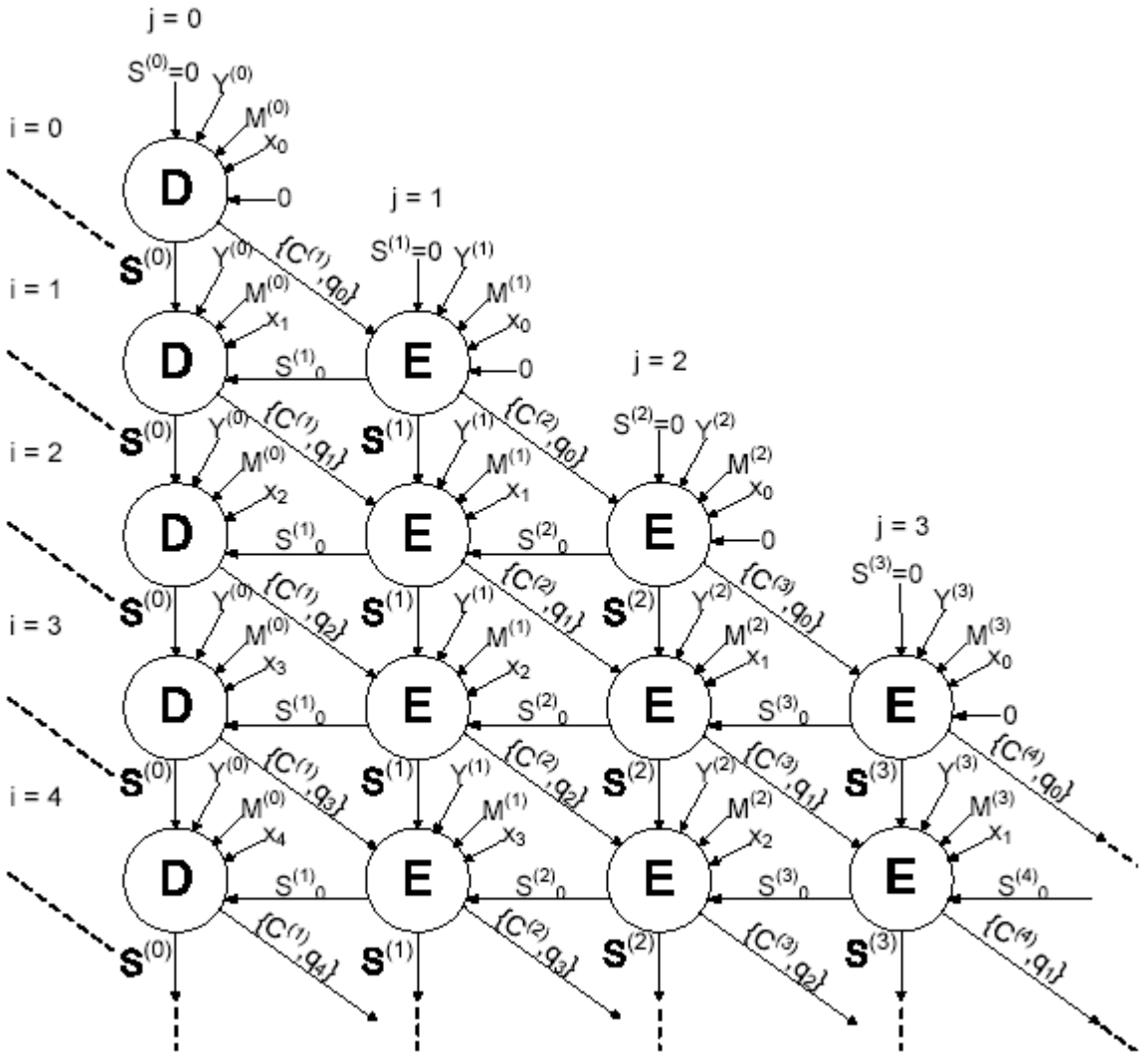
  n + e – 1

- Minimum penalty in terms of the area and clock period

The Proposed Optimized Hardware Architecture

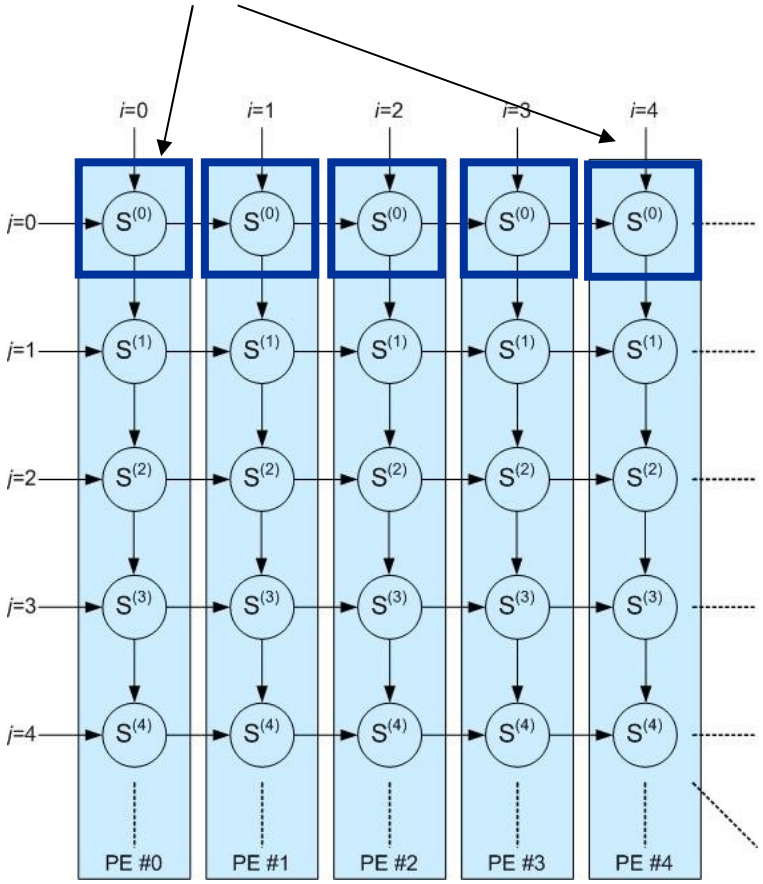


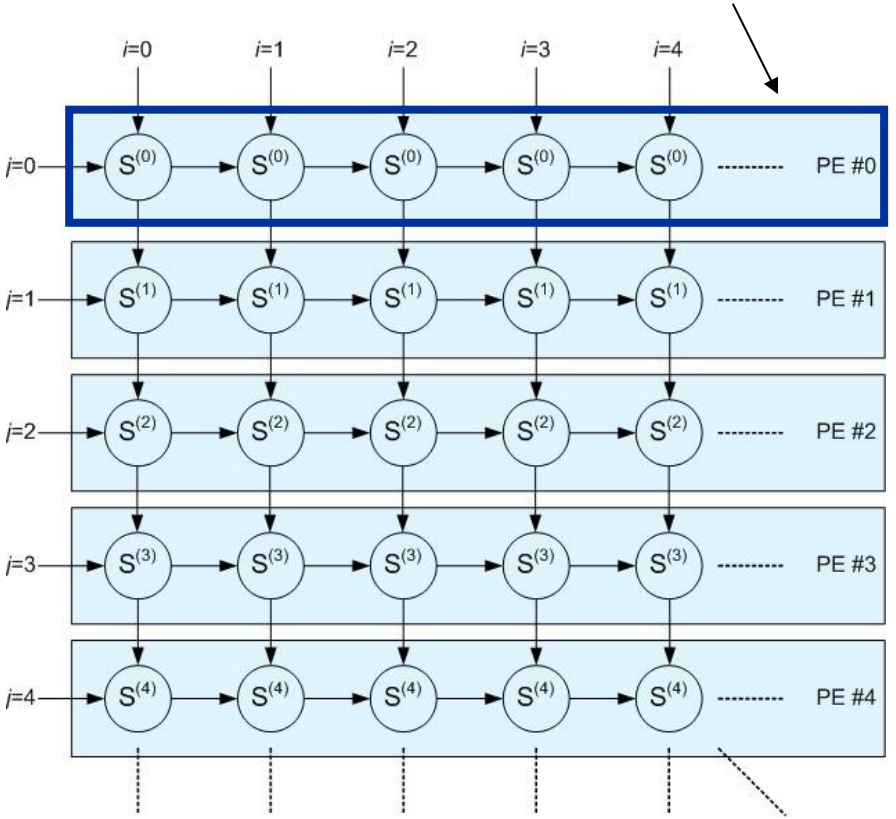
Data Dependency Graph of the Proposed New Architecture



The Overall Computation Pattern

Special state of each PE **vs.** **One special PE type**
simpler structure of each PE



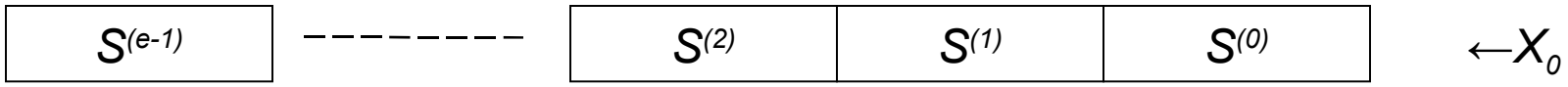


Tenca & Koc, CHES 1999

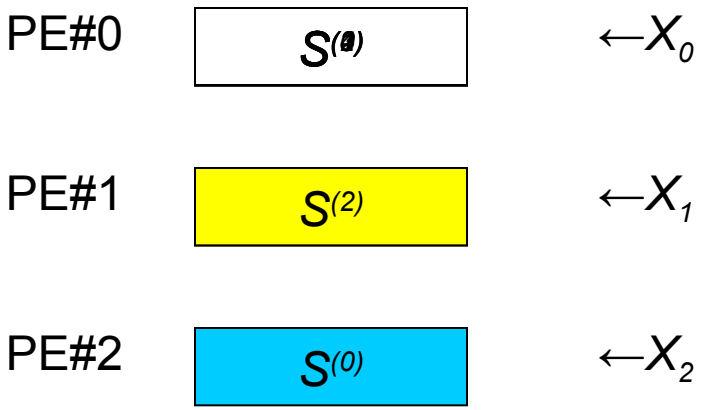
Our new proposed architecture

Demonstration of Computations

- Sequential

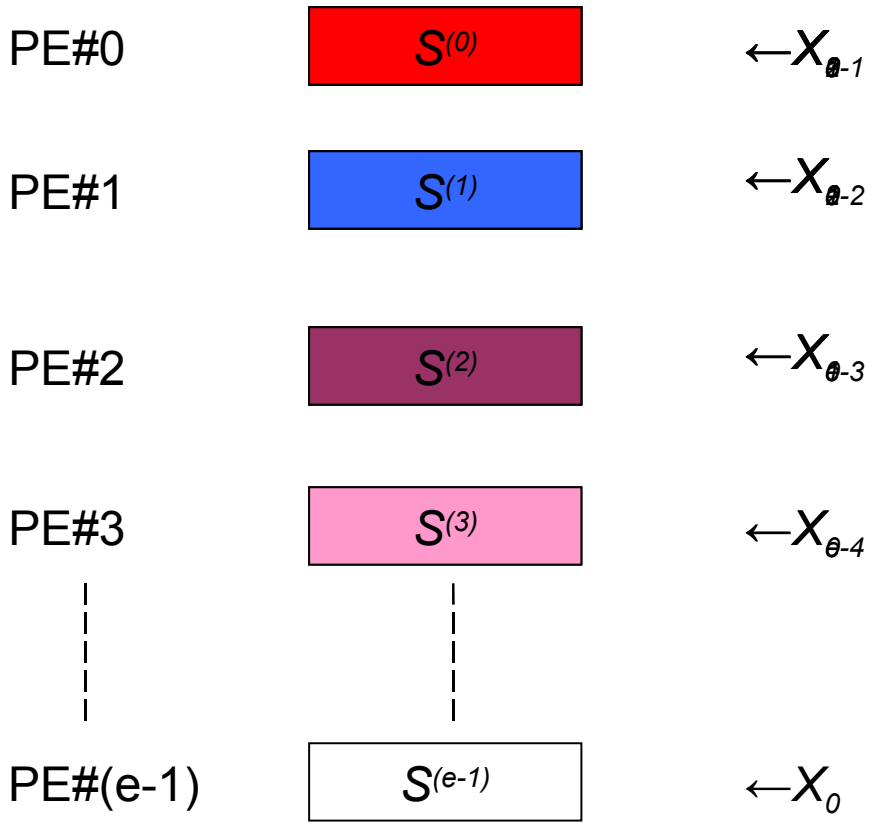


- Tenca & Koç's proposal



Demonstration of Computations *(cont.)*

- **The proposed optimized architecture**



Related Work

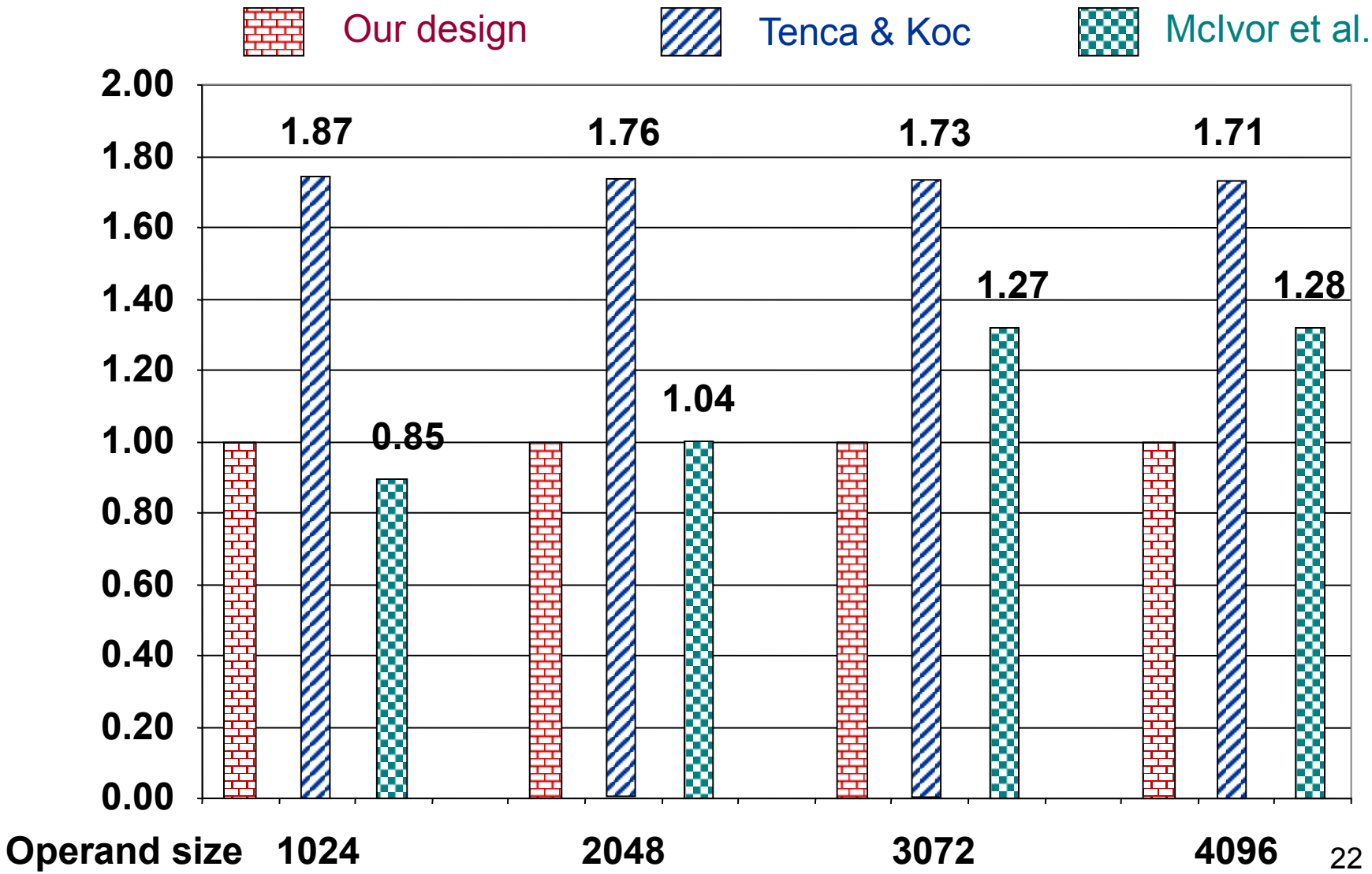
- Tenca, A. and Koç, Ç.K.: A scalable architecture for Montgomery multiplication, CHES 1999, LNCS, vol.1717, pp.94--108, Springer, Heidelberg, 1999
  - The original paper proposing the MWR2MM algorithm
- Tenca, A., Todorov, G., and Koç. Ç.K.: High-radix design of a scalable modular multiplier, CHES 2001, LNCS, vol.2162, pp.185--201, Springer, Heidelberg, 2001
  - Scan several bits of *X* one time instead of one bit
- Harris, D., Krishnamurthy, R., Anders, M., Mathew, S. and Hsu, S.: An Improved Unified Scalable Radix-2 Montgomery Multiplier, ARITH 17, pp.172-178, 2005
  - Left shift *Y* and *M* instead of right shift $S^{(i)}$
- Michalski, E. A. and Buell, D. A.: A scalable architecture for RSA cryptography on large FPGAs, FPL 2006, pp.145--152, 2006
  - FPGA-specific, assumes the use of of built-in multipliers
- McIvor, C., McLoone, M. and McCanny, J.V.: Modified Montgomery Modular Multiplication and RSA Exponentiation Techniques, IEE Proceedings – Computers & Digital Techniques, vol.151, no.6, pp.402-408, 2004
  - Use carry-save addition on full-size n-bit operands

Implementation, Verification, and Experimental Testing

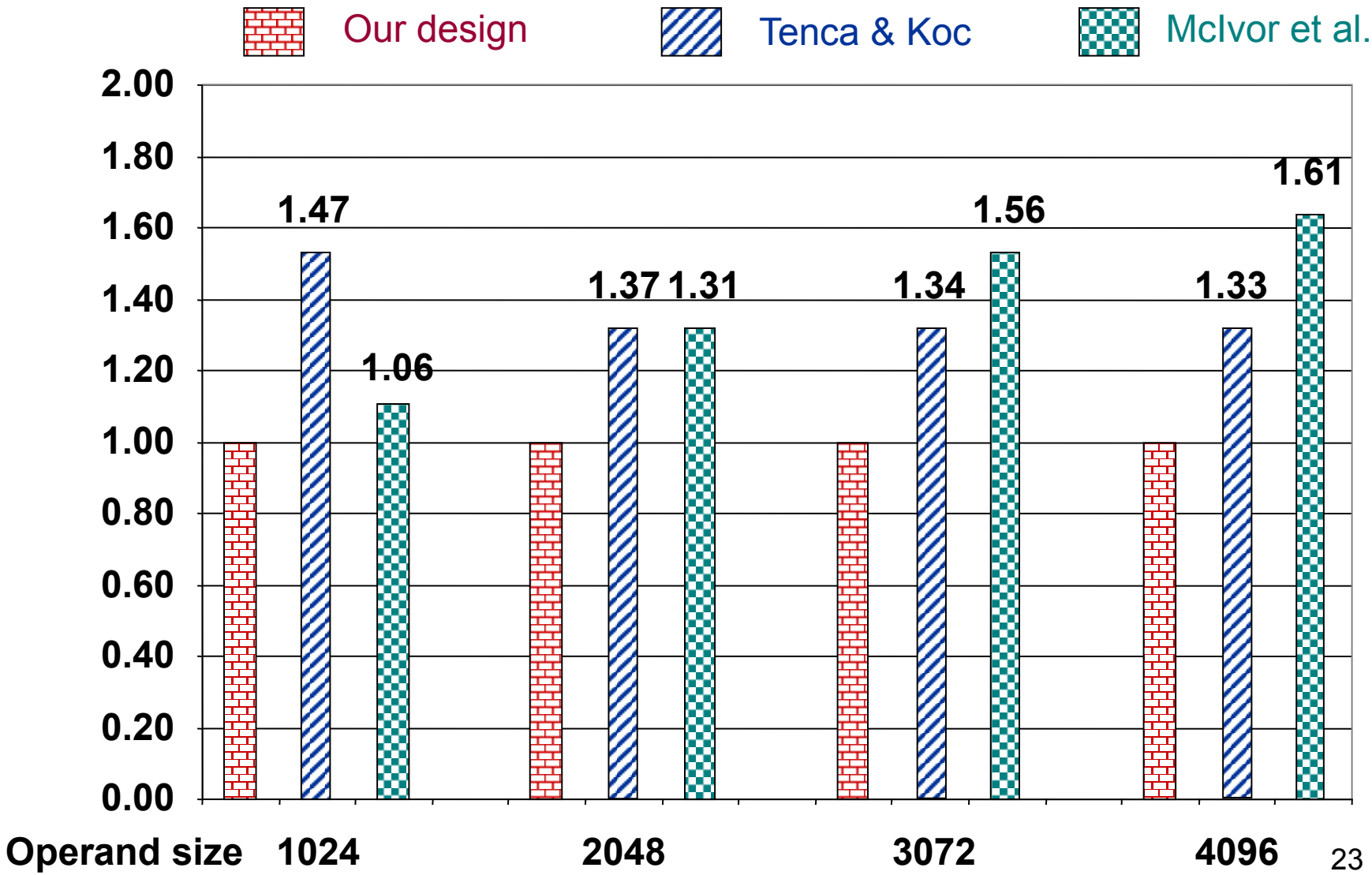
New architecture and two previous architectures:

- Modeled in Verilog HDL and/or VHDL
- Functionally verified by comparison with a reference software implementation of the Montgomery Multiplication
- Implemented using Xilinx Virtex-II 6000-4 FPGA
- Experimentally tested using SRC 6 reconfigurable computer based on microprocessors and FPGAs with the 100 MHz maximum clock frequency for the FPGA part

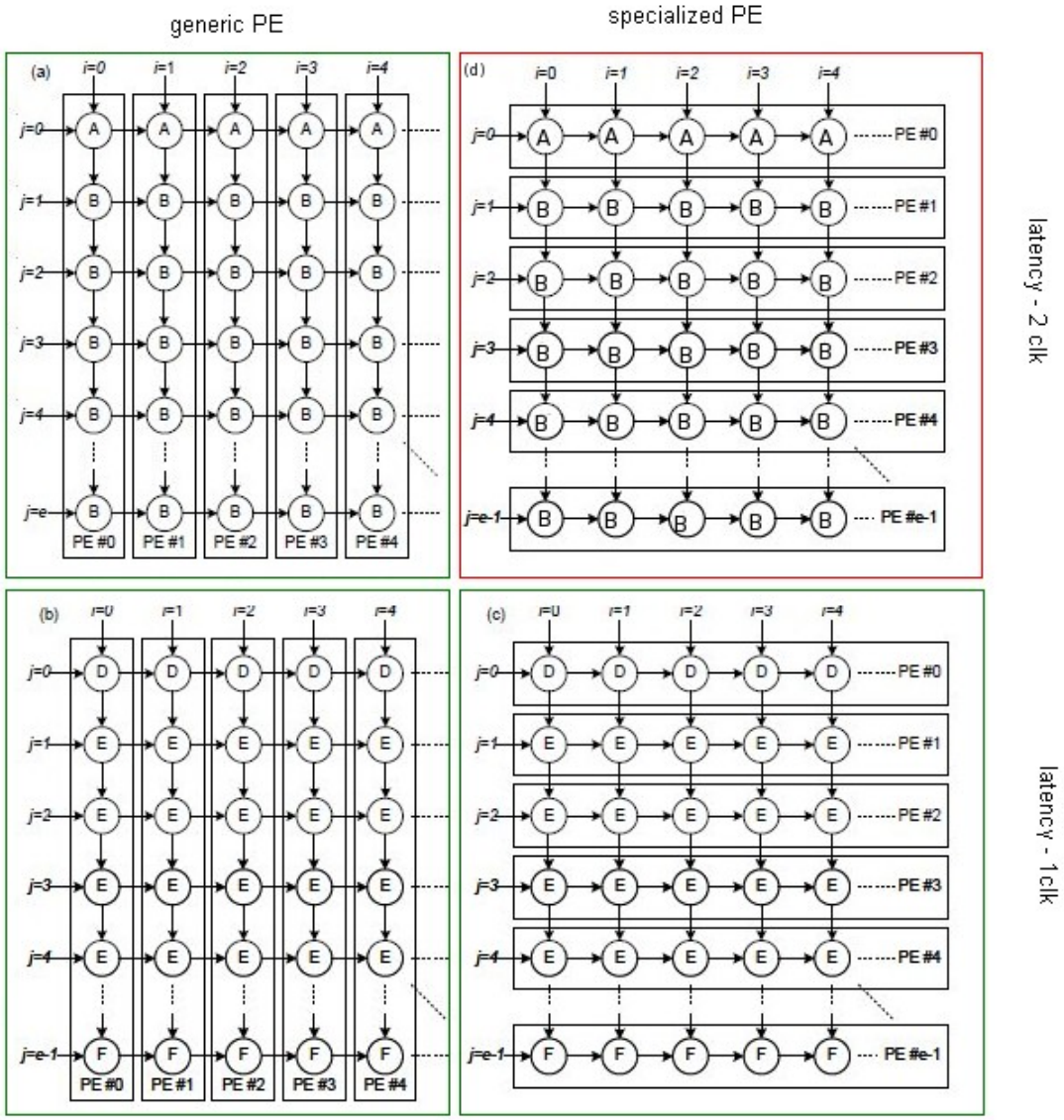
Normalized Latency
New Architecture vs. Previous Architectures



Normalized Product Latency Times Area
New Architecture vs. Previous Architectures



Further Research



Conclusions

- New optimized architecture for the word-based Montgomery Multiplier

Compared to the classical design by Tenca & Koc:

- Minimum latency smaller by a factor of about **1.7**, in terms of both clock cycles and absolute time units
- Comparable circuit area for minimum latency
- Improvement in terms of the product of latency times area by a factor of about **1.3**
- Reduced scalability in Architecture 2, comparable scalability in Architecture 1

Compared to the newer design by McIvor et al.:

- Area smaller by about 25%
- Improvement in terms of the product of latency times area by a factor between **1.06** and **1.61**
- Similar scalability

Thank you!

Questions???