# Implementing the Elliptic Curve Method of Factoring in Reconfigurable Hardware


Kris Gaj

Soonhak Kwon

Patrick Baier

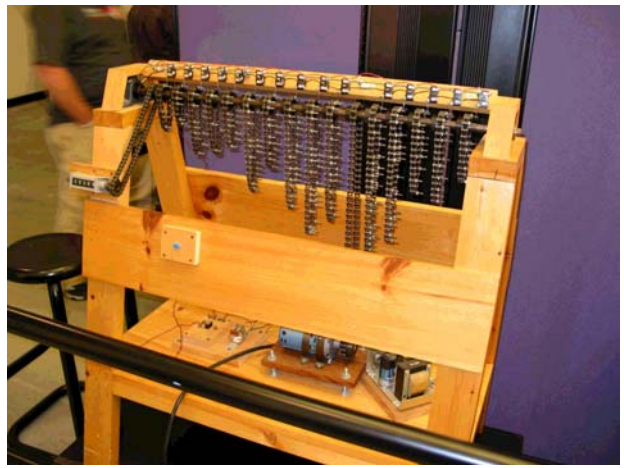
Paul Kohlbrenner

Hoang Le

Khaleeluddin Mohammed

Ramakrishna Bachimanchi

George Mason University

# GMU Team

## Computer Engineer / Cryptographer

### Kris Gaj

Ph.D in Electrical Engineering,
Warsaw University of Technology, Poland
Associate Professor at George Mason University

## Mathematicians/ Cryptographers

### Soonhak Kwon

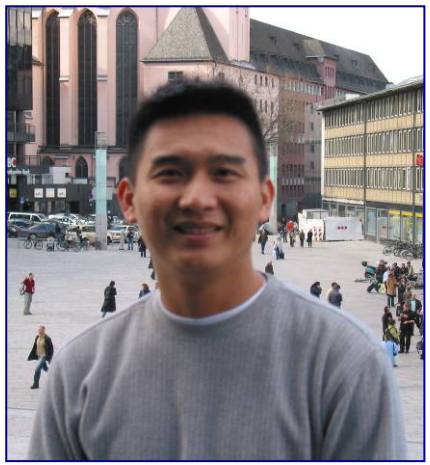
Ph.D in Mathematics,
Johns Hopkins University
Maryland, U.S
Visiting professor at GMU on leave from Sungkyunkwan University, Suwon, Korea

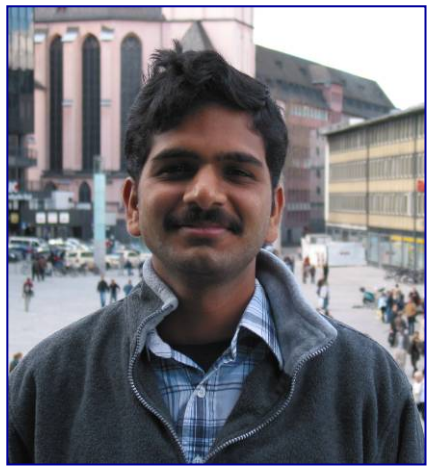
### Patrick Baier

D. Phil. in Mathematics,
Oxford University
Oxford, U.K
Affiliated with George Washington Univeristy

# GMU Team

## Hardware design






Hoang Le

Ramakrishna Bachimanchi

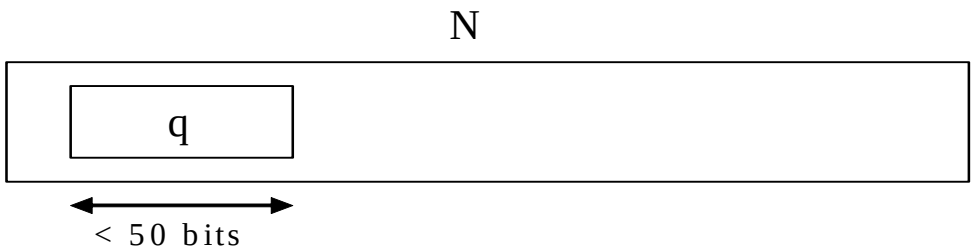
Khaleeluddin Mohammed

MS in Computer Engineering students
ECE Department
George Mason University
Virginia, U.S.A.

# What is ECM?

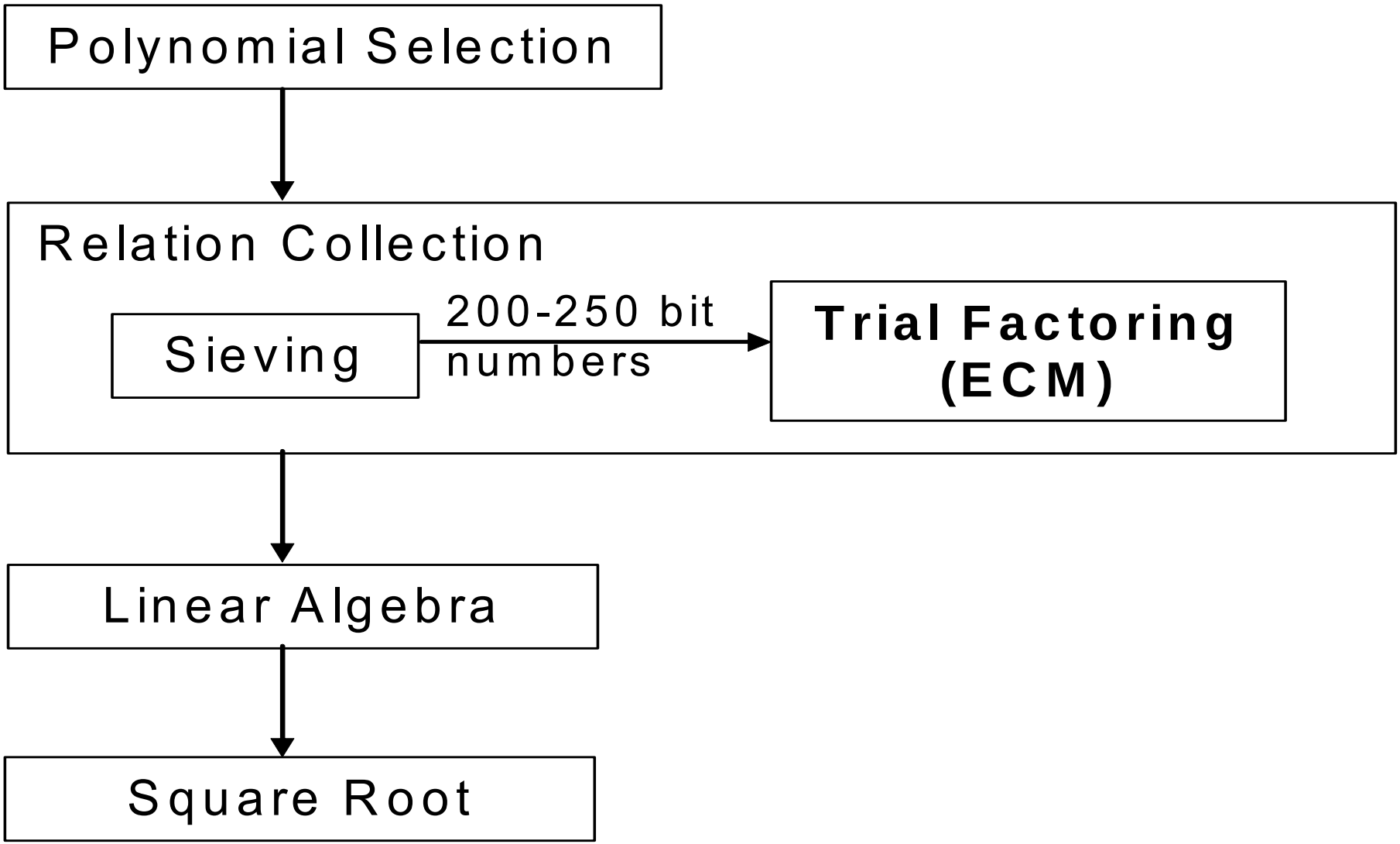
Elliptic Curve Method of Factoring

| | | |
|---|---|---|
| Lenstra | 1985 | Phase 1 |
| Brent, Montgomery | 1986-87 | Phase 2 |



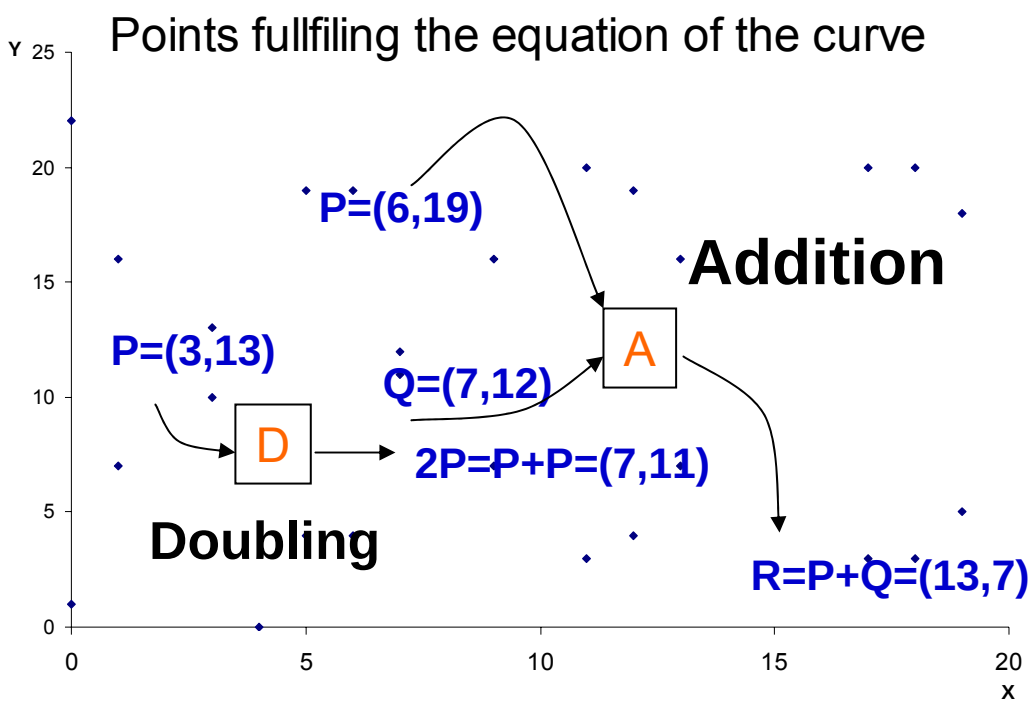

Factoring time depends mainly on the size of factor q

# ECM in the Number Field Sieve (NFS)



# Elliptic Curve

$$Y^2 = X^3 + X + 1 \mod p \qquad (p = 23)$$




+ special point $\vartheta$ (point at infinity) such that:

$$P + \vartheta = \vartheta + P = P$$

$$\exists\ P: \qquad \underbrace{P, 2P, 3P, \ldots\ldots\ldots\ldots, nP = \vartheta}_{\text{all points of the curve}}, (n+1)P = P, 2P$$

# Projective vs. Affine coordinates

- affine coordinates

$$P_a=(X_P, Y_P)$$

  - addition and doubling **require inversion**
- projective coordinates

$$P_p=(x_P, y_P, z_P)$$

  - addition and doubling can be done **without inversion**

- projective coordinates for Montgomery form of the curve
  - addition and doubling **do not require y coordinate**

    (y coordinate can be recovered from x and z at the end of a long chain of computations)

$$P_{pM} = (x_P : : z_P)$$

$$\vartheta = (0 : : 0)$$

# ECM Algorithm

**Inputs :**

$N$ – number to be factored

$E$ – elliptic curve

$P_0$ – point of the curve $E$ : initial point

$B_1$ – smoothness bound for Phase1

$B_2$ – smoothness bound for Phase2

**Outputs:**

$q$ - factor of N, $1 < q \leq N$

or FAIL

# ECM algorithm – Phase 1

*precomputations*

1: $k \leftarrow \prod_{p_i} p_i^{e_i}$ such that $p_i$ - consecutive primes $\leq B_1$

$e_i$ - largest exponent such that $p_i^{e_i} \leq B_1$

---

2: $Q_0 \leftarrow kP_0 = (x_{Q_0} : \ : z_{Q_0})$ *main computations*

---

3: $q \leftarrow \gcd(z_{Q_0}, N)$ *postcomputations*

4: if $q > 1$

5: return $q$ (factor of $N$)

6: else

7: go to Phase 2

8: end if

# ECM algorithm – Phase 2

09: $d \leftarrow 1$

10: for each prime $p = B_1$ to $B_2$ do

11: $(x_{pQ_0}, y_{pQ_0}, z_{pQ_0}) \leftarrow pQ_0$

12: $d \leftarrow d \cdot z_{pQ_0} \pmod{N}$

13: end for

*main computations*

---

14: $q \leftarrow \gcd(d, N)$

15: if $q > 1$ then

16: return $q$

17: else

18: return FAIL

19: end if

*postcomputations*

# Phase 1 – Numerical example

$N = 1\,740\,719 = 1279 \cdot 1361$

$E : y^2 = x^3 + 14x + 1 \pmod{1\,740\,719}$

$P_0 = (5 : : 1)$

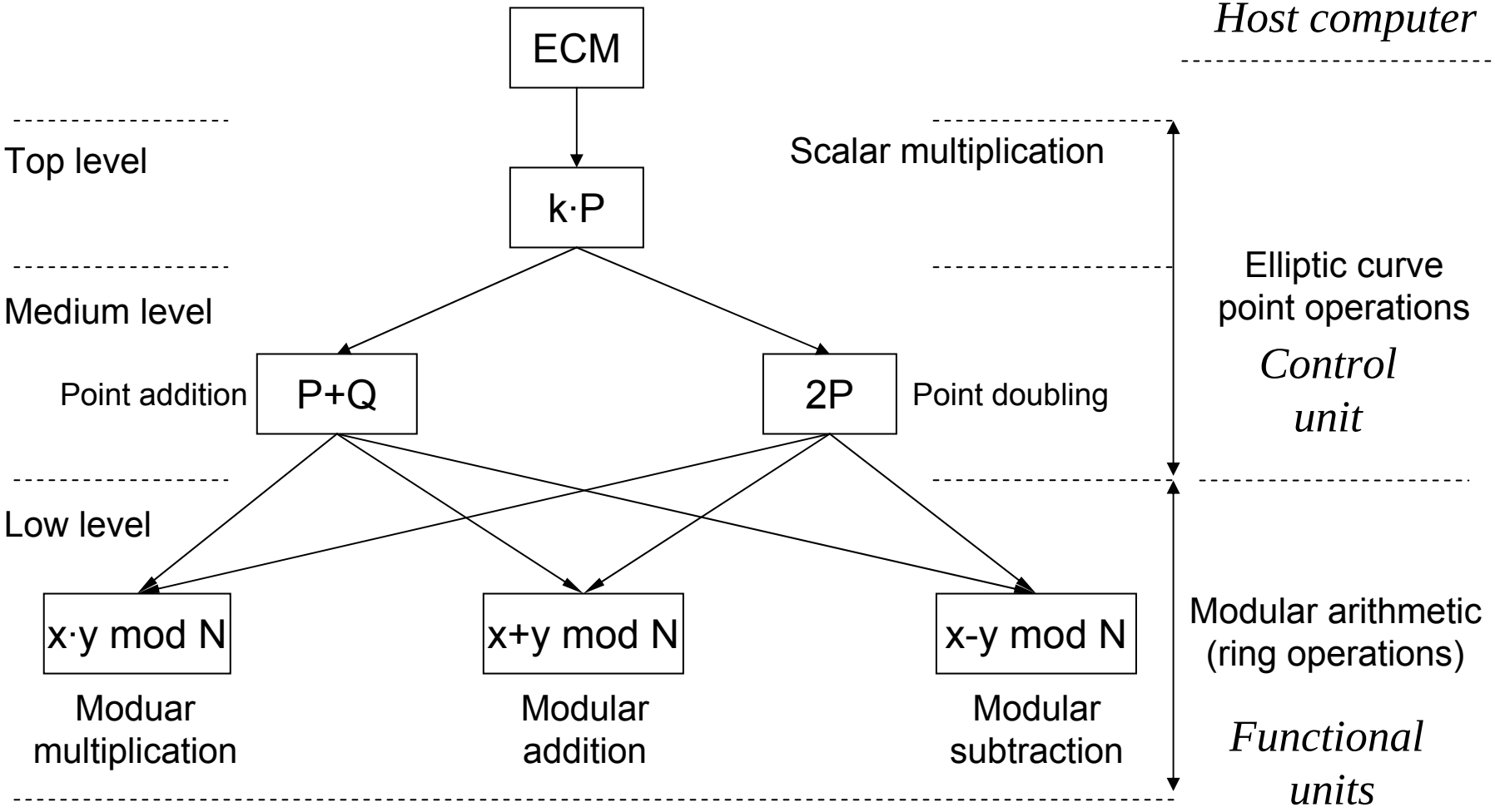
$B_1 = 20$

$k = 2^4 \cdot 3^2 \cdot 5 \cdot 7 \cdot 11 \cdot 13 \cdot 17 \cdot 19 = 232\,792\,560$

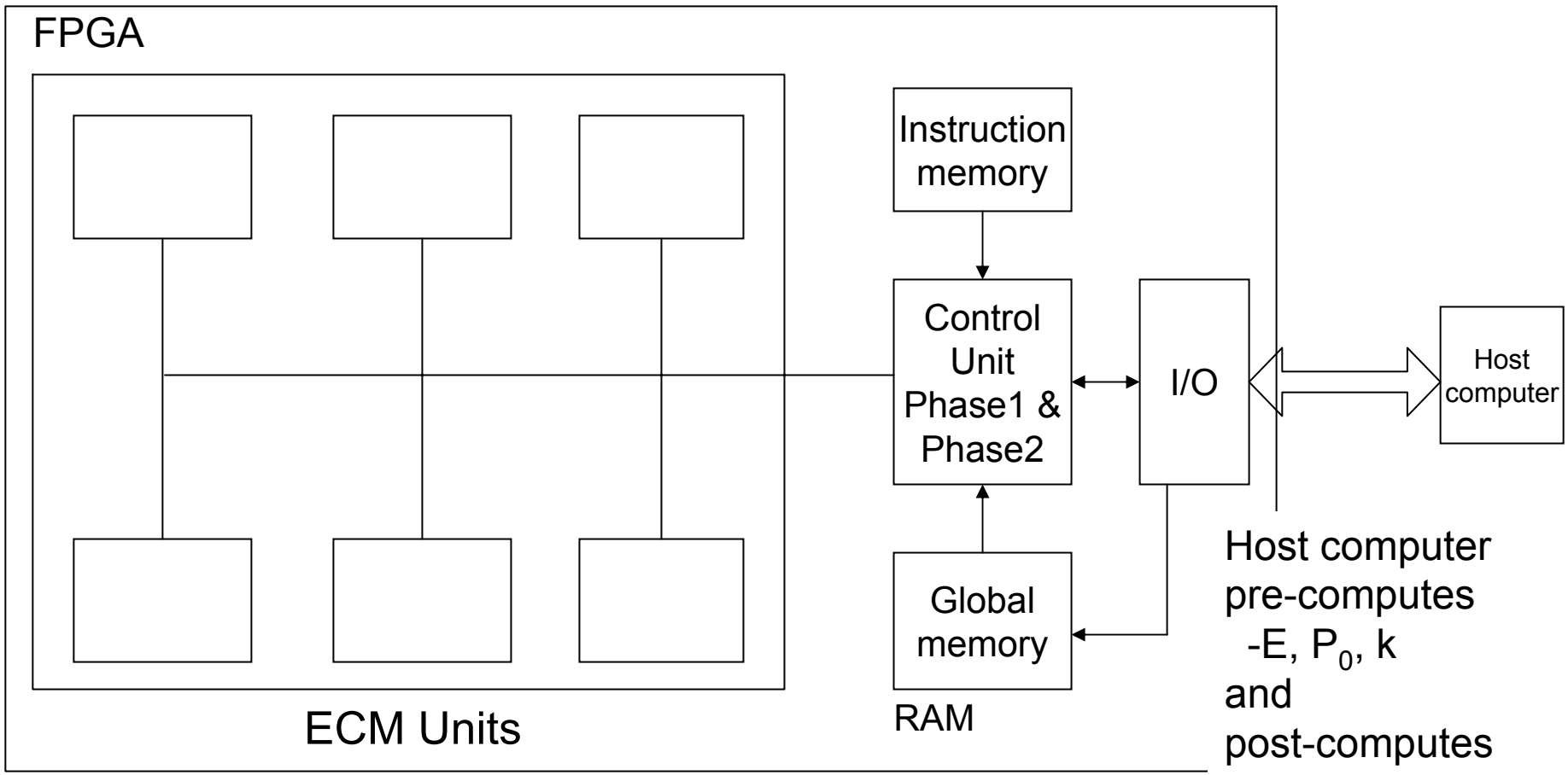
$kP_0 = (707\,838 : \ : 1\,686\,279)$

gcd (1 686 279 ; 1 740 719) = **1361**

# Hierarchy of Elliptic Curve Operations



# Our architecture : Top-level view



# ECM in Hardware

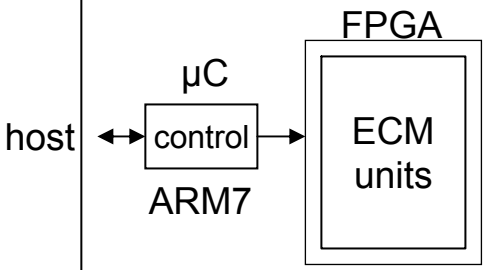
## Previous Proof-of-Concept Design

| | | |
|---|---|---|
| Pelzl, Šimka, | SHARCS | Feb 2005 |
| Kleinjung, Franke, | FCCM | Apr 2005 |
| Priplata, Stahlke, | IEE Proc. | Oct 2005 |
| Drutarovský, Fischer, | | |
| Paar | | |

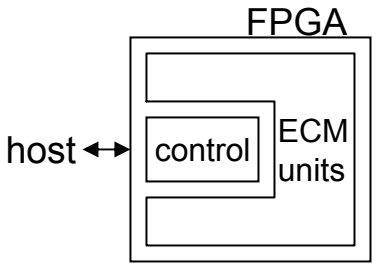
# Modifications compared to Pelzl, Šimka

## Internal vs. External control

Pelzl, Šimka



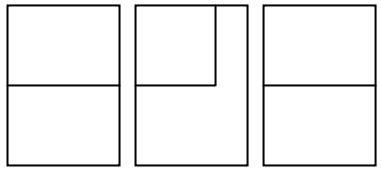

Ours




## Memory management

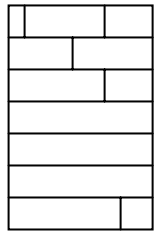
Pelzl, Šimka

suboptimal use of memory space



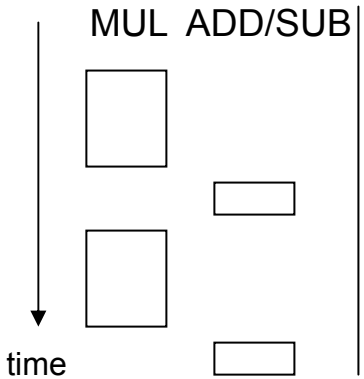
Ours

bit tables & consolidation of memory resources

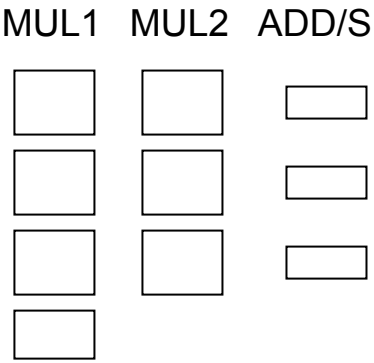


## Functional units

Pelzl, Šimka




Ours




## Montgomery multiplier

Pelzl, Šimka

Based on

Tenca, Koc

CHES 1999

IEEE Trans. Comp. 2003

word-based CPA and/or CSA

Ours

Based on

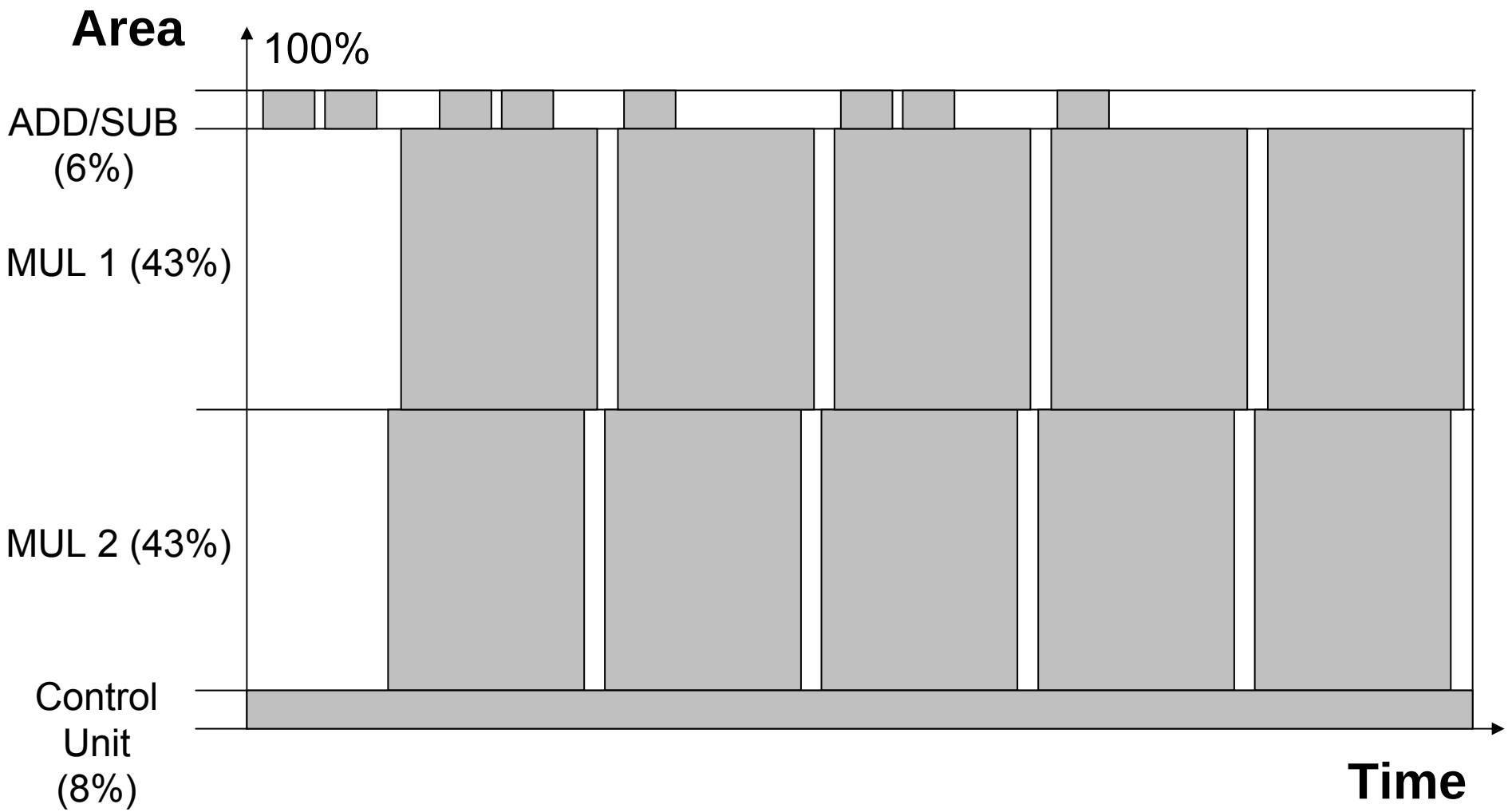
McIvor, McLoone et al.

Asilomar 2003

full-length CSAs

word-length CPAs

# Resources utilization in time – Phase 1

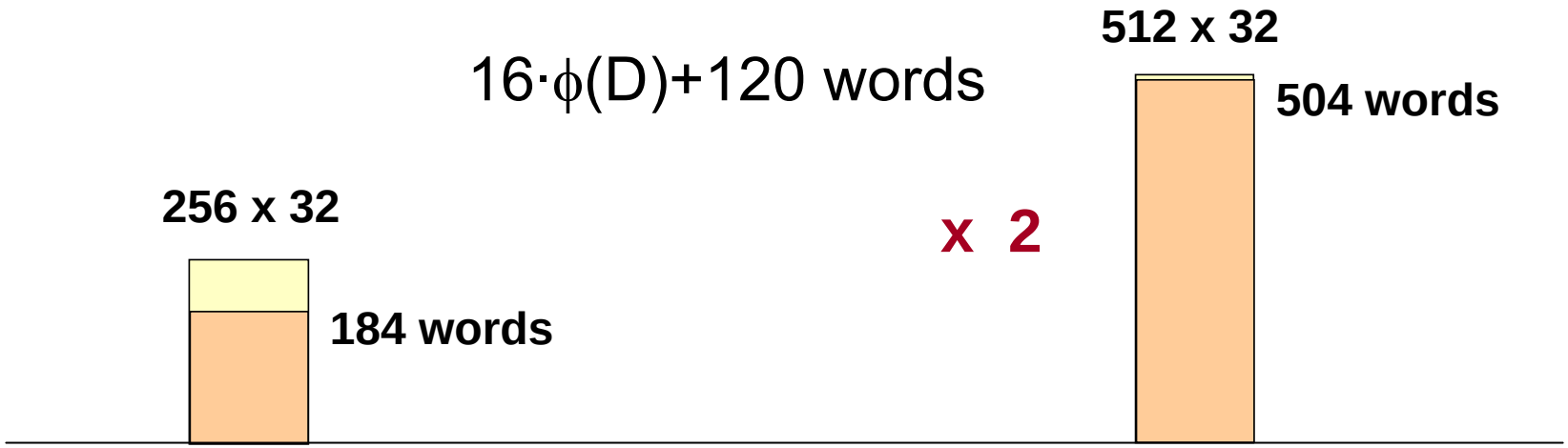


# Phase 2 Parameter D

**D=30=2·3·5** **D=210=2·3·5·7**

## Memory of ECM Unit

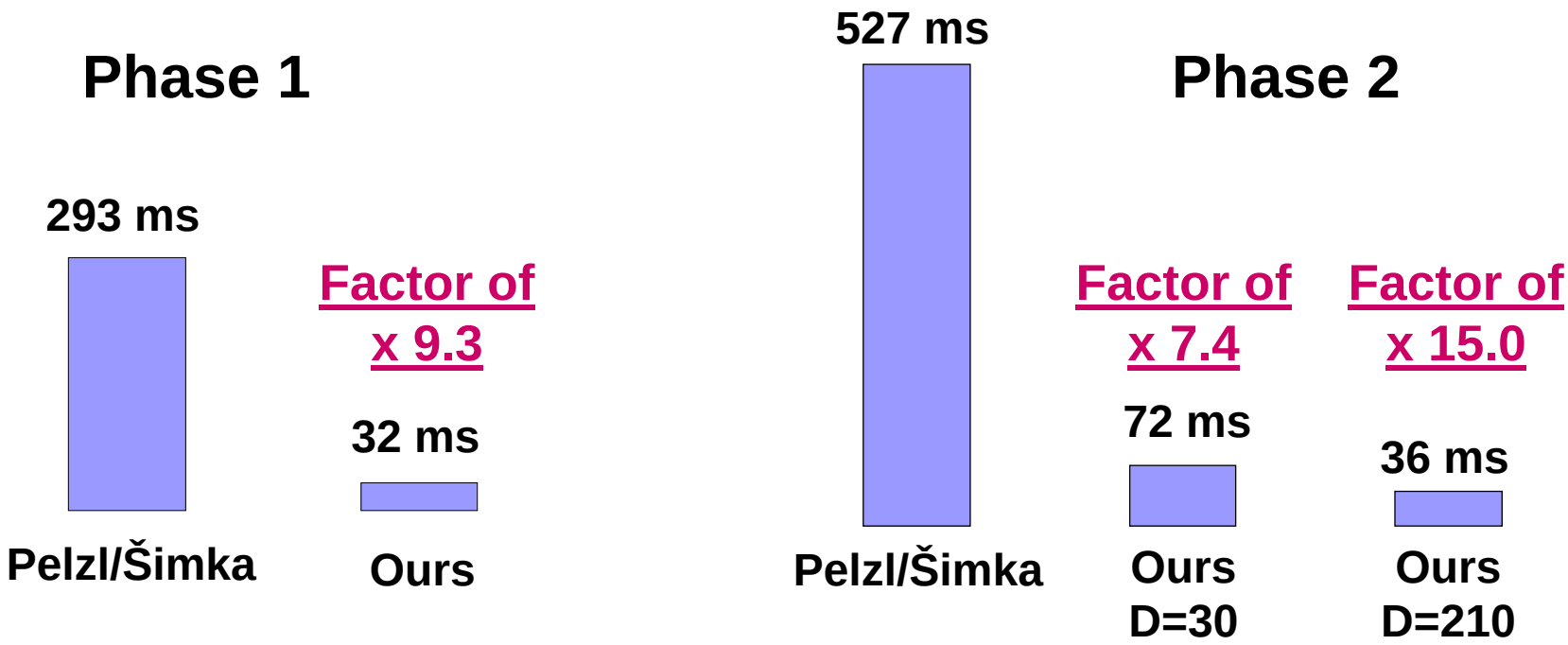
$16 \cdot \phi(D)+120$ words




## Phase 2 Execution Time

# Comparison with the Proof-of-Concept Design by Pelzl and Šimka: Timing

## (198-bit numbers N; B1 = 960, B2 = 57000)



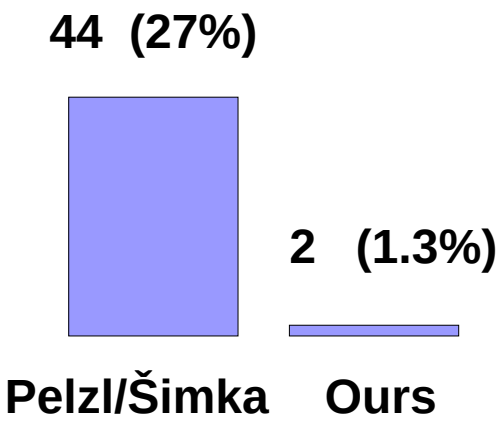

**Major Contributors to the speed up:**

- Different design for the multiplier ( x 5 )
- Two multipliers working in parallel ( x 1.9 )
- Different parameter of Phase 2, D ( x 2 )

# Comparison with the Proof-of-Concept Design by Pelzl and Šimka

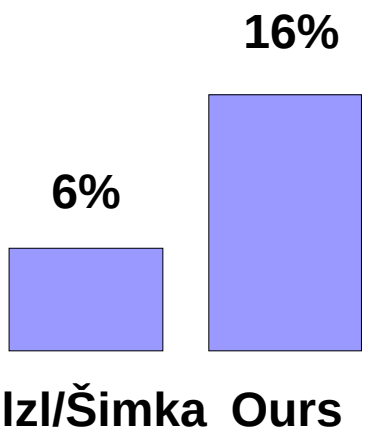
## Resources (D=30)

**Memory (BRAMs)**



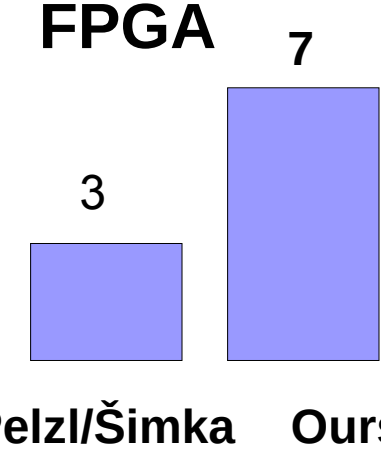

Factor of x 22

**Area (CLB Slices)**




Factor of x 2.7

**ECM Units / Virtex 2000E FPGA**




Limited by BRAMs

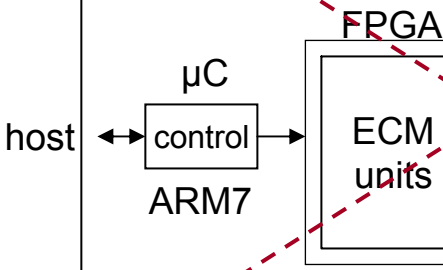
Limited by CLB Slices

Factor of x 2.33

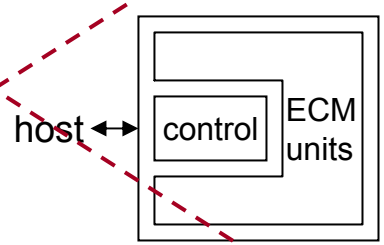
# Modifications compared to Pelzl, Šimka

## Internal vs. External control

Pelzl, Šimka



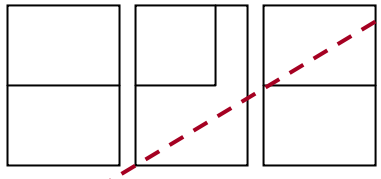

Ours




## Memory management

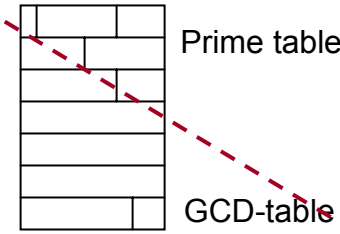
Pelzl, Šimka

suboptimal use of memory space



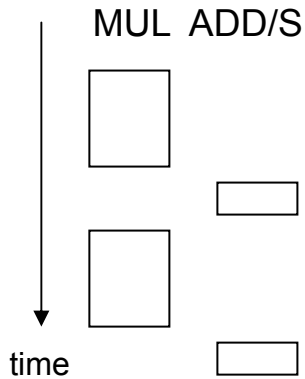
Ours

bit tables & consolidation of memory resources

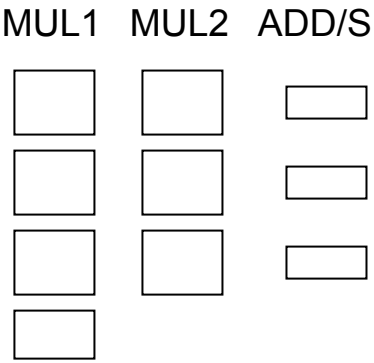



## Functional units

Pelzl, Šimka




Ours




## Montgomery multiplier

Pelzl, Šimka

Based on
Tenca, Koc
CHES 1999
IEEE Trans. Comp. 2003
word-based CPA and CSA

Ours

Based on
McIvor, McLoone et al.
Asilomar 2003
full-length CSAs
word-length CPAs

# Comparison with the Proof-of-Concept Design by Pelzl and Šimka

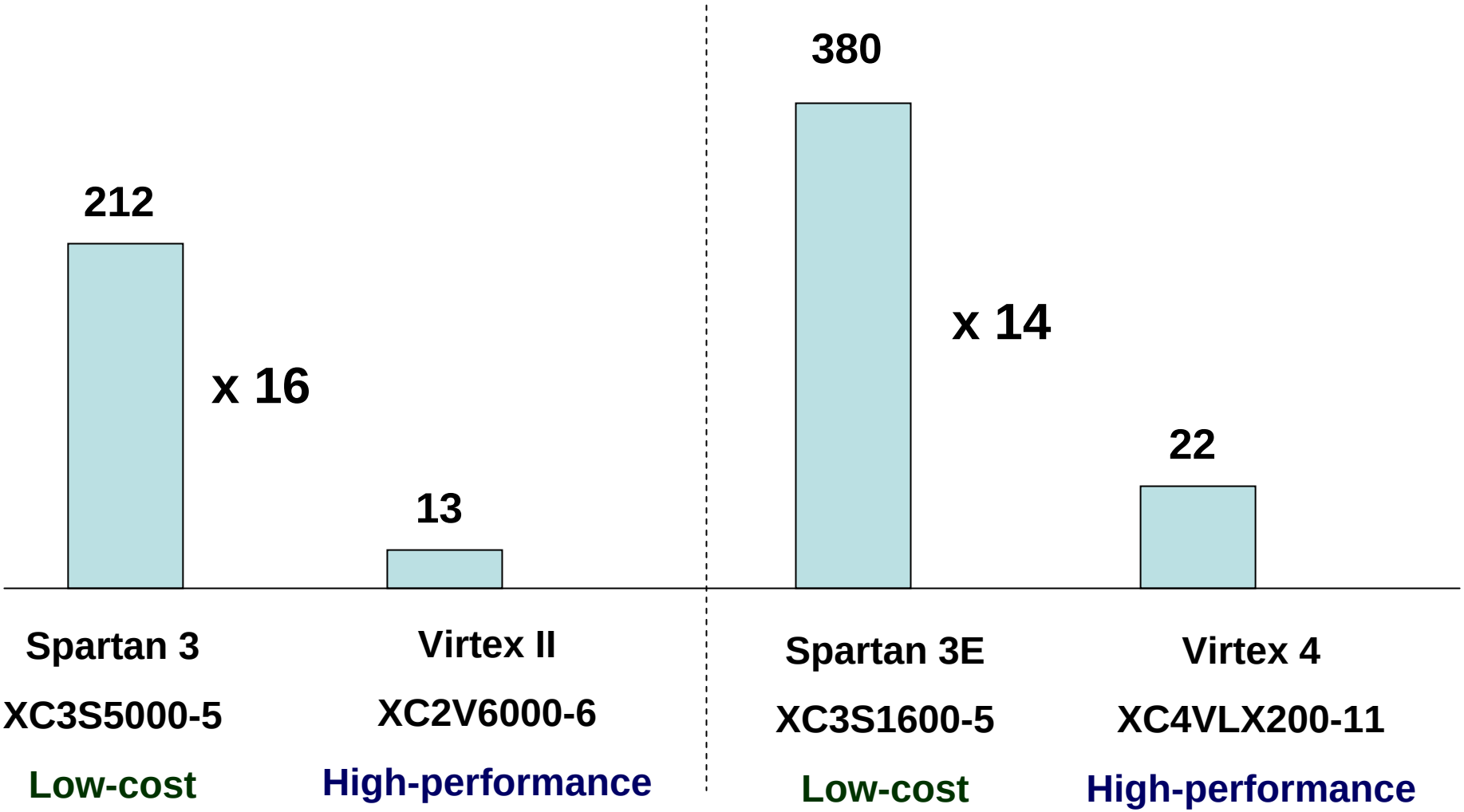
## Time x Area Product

**Assuming the same control unit and the same memory management**

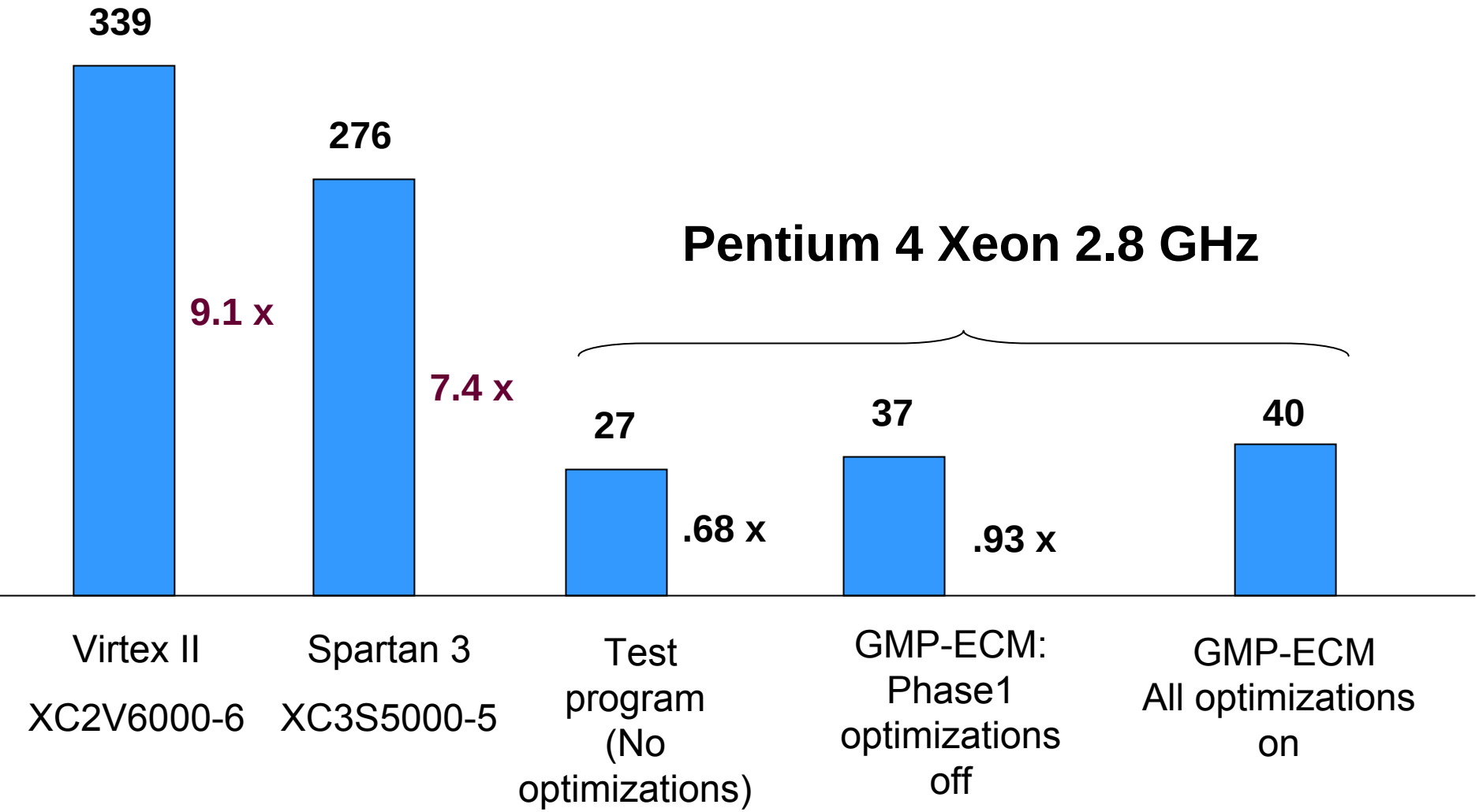
**(i.e., significantly improved design in Pelzl/Šimka):**

| | **Improvement** |
|---|---|
| **Phase 1** | **x 3.4** |
| **Phase 2** | **x 5.6** |

# Performance to cost ratio
# Number of Phase 1 & Phase 2 operations per second per $100



# FPGAs vs Microprocessors
# # Phase 1 & Phase 2 computations per second



# Experimental testing using SRC 6 reconfigurable computer


**SRC 6** from
SRC Computers

Basic unit:

2 x Pentium Xeon 3 GHz

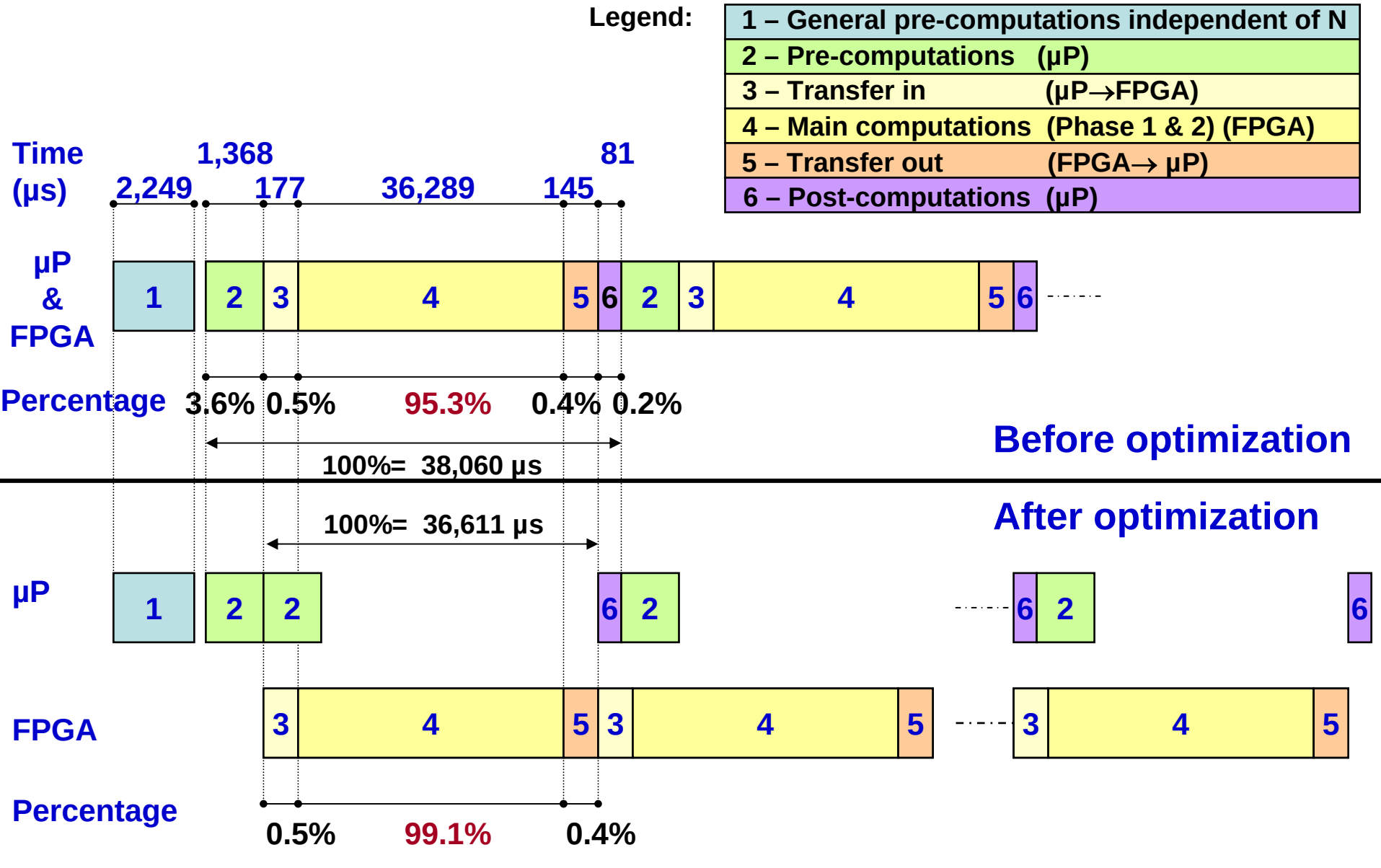
2 x Xilinx Virtex II FPGA
XC2V6000 running at 100 MHz

24 MB of the FPGA-board RAM

Fast communication interface between the microprocessor board and the FPGA board, 1600 MB/s

Multiple basic units can be connected using Hi-Bar Switch and Global Common Memory

# Results of experimental testing using SRC 6 reconfigurable computer



# Conclusions

**Hardware implementations of ECM provide**
**a substantial improvement**
**vs. optimized software implementations**
**in terms of the performance to cost ratio**

- **low-cost FPGAs vs. microprocessors > 10 x**

**Best environment for prototyping**
**of hardware implementations of codebreakers**

- **general-purpose reconfigurable computers (e.g., SRC)**

**Best environment for the final design**
**of the cost-optimized cipher breaker**

- **special-purpose machines based on**
  - **low-cost FPGAs (or ASICs for very high volumes)**

# Thank you!

Questions??
?