High Performance ECC over NIST Primes on Commercial FPGAs

ECC 2008, Utrecht, September 22-24, 2008

Tim Güneysu
Horst Görtz Institute for IT-Security
Ruhr University of Bochum, Germany

Agenda

Horst-Görtz Institut für IT Sicherheit

- Introduction and Motivation
- Brief Survey on Reconfigurable Computing and FPGAs
- Modern FPGA devices and Arithmetic Applications
- Novel Architectures for ECC over NIST primes
- Results and Conclusions

Agenda

- **Introduction and Motivation**
- Brief Survey on Reconfigurable Computing and FPGAs
- Modern FPGA devices and Arithmetic Applications
- Novel Architectures for ECC over NIST primes
- Results and Conclusions

Introduction and Motivation

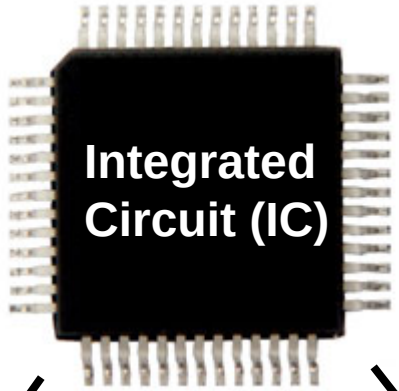
Horst-Görtz Institut für IT Sicherheit

- Some recent and future systems require high-speed cryptography facilities processing **hundreds of asymmetric message signatures per second**.
  - Car-to-car communication
  - Aggregators in wireless sensor node systems

- Typical challenges:
  - **Small and embedded systems** providing high-speed asymmetric crypto → best choice seems to be ECC!
  - Small µP (Atmel/ARM) are too slow for high-performance ECC → use **dedicated crypto hardware**
  - ECC using binary curves in hardware is most efficient but **patent situation on algorithms and implementations is unclear**
  - National bodies prefer **ECC over prime field** (FIPS 186-2, Suite B)

High Performance Hardware Implementations

- **Two main flavors** of application-specific hardware chips
  - ASICs
  - FPGAs
- This talk targets **ECC on FPGAs**
  - Reconfiguration feature enables adaption of security parameters and algorithms if necessary
  - Good choice for applications with low/medium market volume



Field Programmable Gate Arrays (FPGA)

- reconfigurable logic
- medium/high performance
- medium cost per chip
- quick/cheap development

Application Specific Integrated Circuit (ASIC)

- fixed logic
- very high performance
- low cost per chip
- expensive development

History of ECC Implementation on FPGAs

Horst-Görtz Institut für IT Sicherheit

- **First ECC implementation for prime fields with FPGAs in 2001:**

  G. Orlando, C. Paar, A scalable GF(p) elliptic curve processor architecture for programmable hardware, CHES 2001

- Since this milestone several **improvements** were made:

  - Use of dedicated multipliers in FPGAs, e.g. in
    C. McIvor, M. McLoone, J. McCanny, *An FPGA elliptic curve cryptographic accelerator over GF(p),* Irish Signals and Systems Conference, ISSC 2004.

  - Algorithmic optimizations, e.g. use of fabric-based CIOS multipliers:
    K. Sakiyama, N. Mentens, L. Batina, B. Preneel, and I. Verbauwhede, *Reconfigurable Modular Arithmetic Logic Unit Supporting High-performance RSA and ECC over GF(p),* International Journal of Electronics 2007.

ECC over Prime Fields on FPGAs

Horst-Görtz Institut für IT Sicherheit

- **Recent ECC solutions over primes fields on FPGAs are significantly slower than software-based approaches**
  - FPGA designs run at much **lower clock frequencies** than µP
    - Typical ECC designs on FPGAs run at 40-100 MHz
    - Point multiplication on FPGAs takes more than *3ms* for ECC-256
    - Software-based ECC (Core2Duo) is far below *1ms*!
  - Many hardware implementations use wide adders or multipliers
    → **slow carry propagation**
  - **Complex routing** within and between arithmetic units
    → long signal paths slow down clock frequency
- **Our high-performance ECC core based on standardized NIST primes for Xilinx Virtex-4 FPGAs closes this performance gap! [CHES 2008]**

Changing the Implementation Concept

Horst-Görtz Institut für IT Sicherheit

- **Our different concept how to accelerate ECC on FPGAs**:
  Shift all field operations into arithmetic hardcore extensions of FPGAs!

  – Modern FPGAs integrate arithmetic hardcores originally designed to accelerate Digital Signal Processing (DSP) applications

  – Compute all field operations with DSP hardcores instead of using the generic logic

  – Allows for higher clock rates AND saves logical resources of the FPGA

Agenda

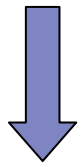
- Introduction and Motivation
- **Brief Survey on Reconfigurable Computing and FPGAs**
- Modern FPGA devices and Arithmetic Applications
- Novel Architectures for ECC over NIST primes
- Results and Conclusions

Brief History of FPGAs

- First FPGAs came up in mid 1980‘s with a gate complexity of 1200 gates (e.g., **Xilinx XC2064**)
  - Significantly too small for (asymmetric) crypto

1985

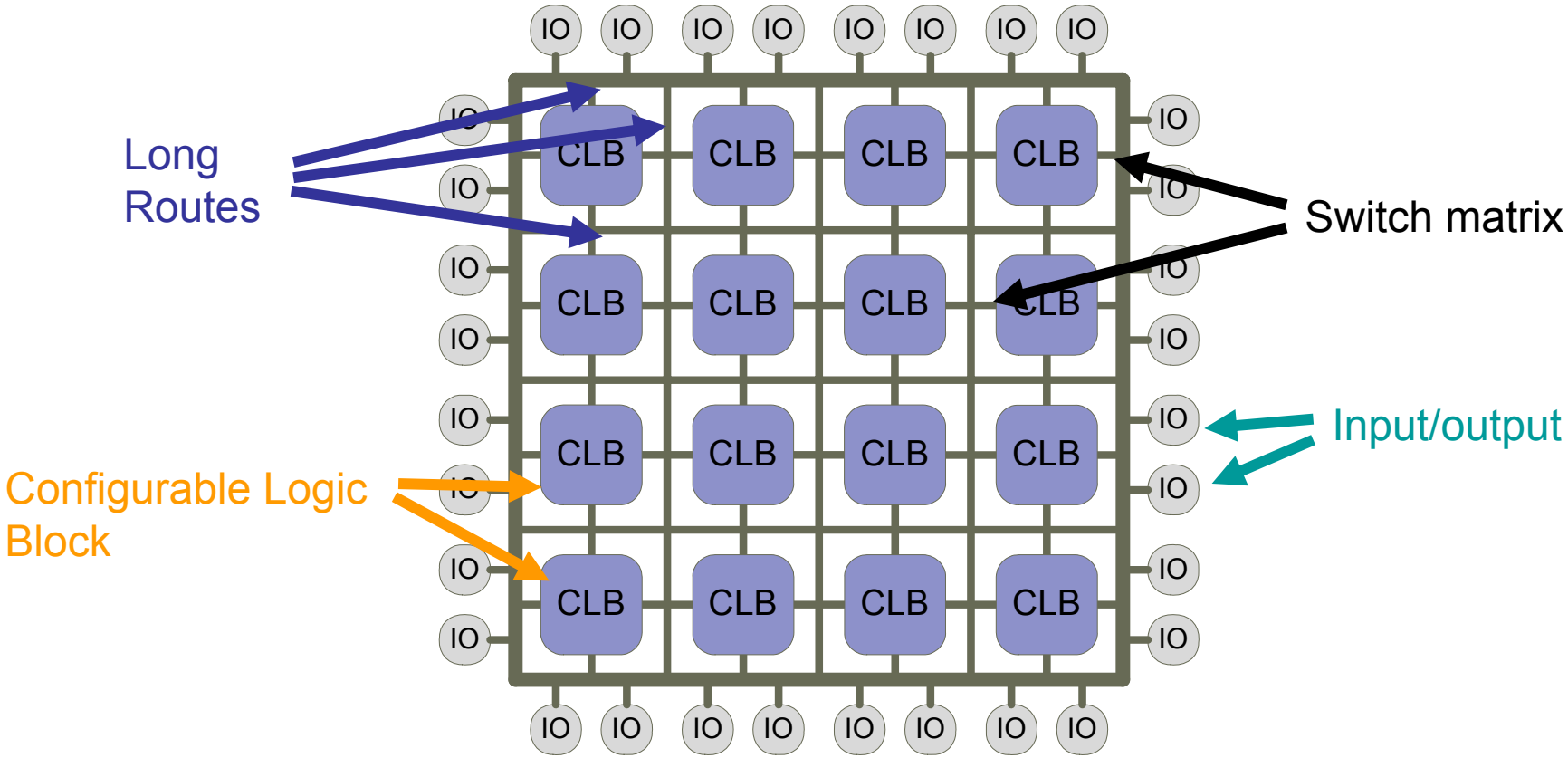
- **Luckily, Moore‘s Law still holds true!**
  - On average, the number of transistors per chip are (roughly) doubled each 18 months
  - With increasing chip complexity and features, FPGAs gained attractivity also for the cryptographic community
  - First ECC implementation over prime fields in 2001!



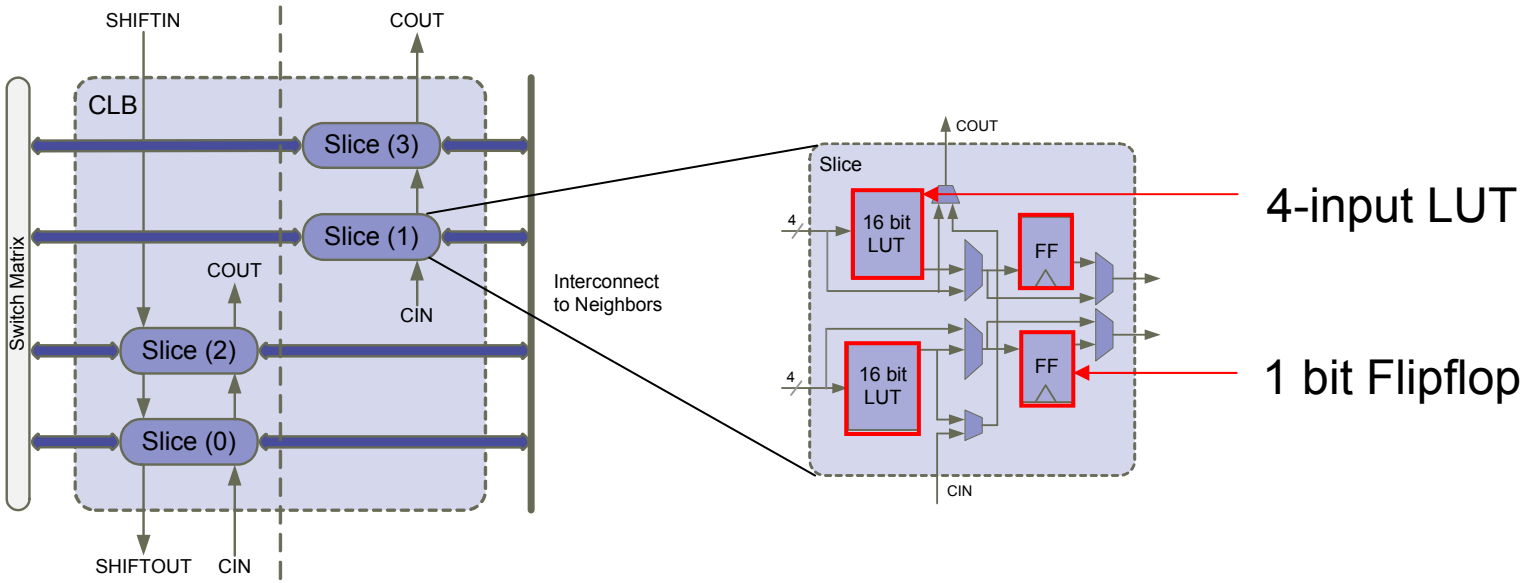
- **Todays (2008) FPGAs provide**
  - Several millions of logic gates (**Xilinx Virtex-5**)
  - Clock frequencies up to 550 MHz
  - Dedicated memories and function hardcores

2008

Generic FPGA Structure (simplified)



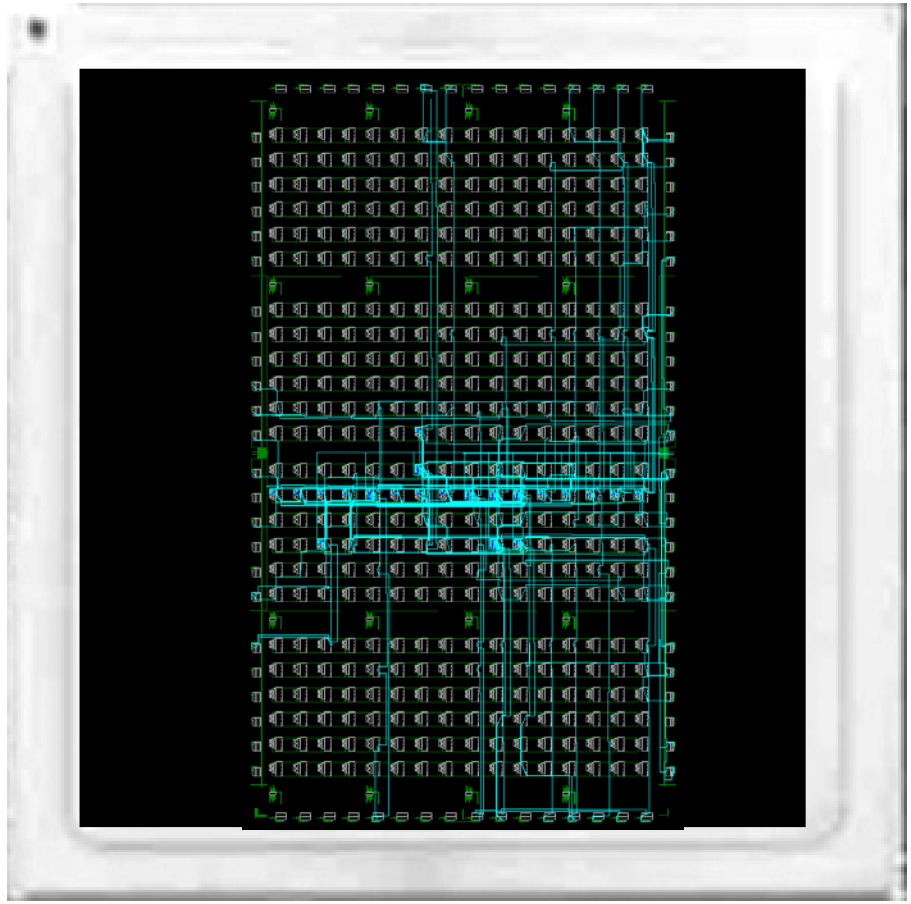
Configurable Logic Block (simplified)



- **A Configurable Logic Block (Virtex4) consists of 4 slices each with**
  - 4-to-1 bit Lookup Table (LUT) used as function generator (4 input, 1 output), 16-bit shift register, 16-bit RAM
  - Dedicated storage elements (1-bit flip flop)
  - Multiplexers, arithmetic gates for fast multipliers/carry logic
  - Connection to other FPGA elements either through switch matrix (long distance) and local routes (short distance)

Hardware Applications on FPGAs

- Most hardware applications are designed using **Hardware Description Languages** (no schematics anymore!!)
- **Description is translated and mapped** using powerful tools into CLBs
- **Golden rules** for high-performance hardware design (informal):
  - **R1**: Exploit **parallelism** as much as possible (only then FPGAs can do better than Pentiums)
  - **R2**: Use **pipelining** techniques (to reduce length of critical path)
  - **R3**: Aim for **uniform data flow** (avoid conditional branches)

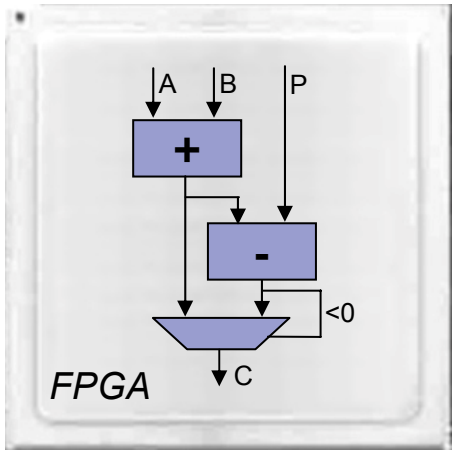


Floorplan of a 32-bit Counting Application on a (tiny) Virtex-E FPGA (XCV50E)

Example: Software vs. Hardware

Horst-Görtz Institut für IT Sicherheit

- **Modular addition** in software and hardware: C = A + B mod P



Approach in software:

```
{
  C = A + B;
  if (C > P) then
    C = C - P;
  end if;
}
```

conditional computation

Approach in hardware (C-like syntax):

```
{
  S = A + B;          [FA]
  T = S - P;          [FA]
  C = (T<0) ? S : T;  [MUX]
}
```

uniform data flow

Agenda

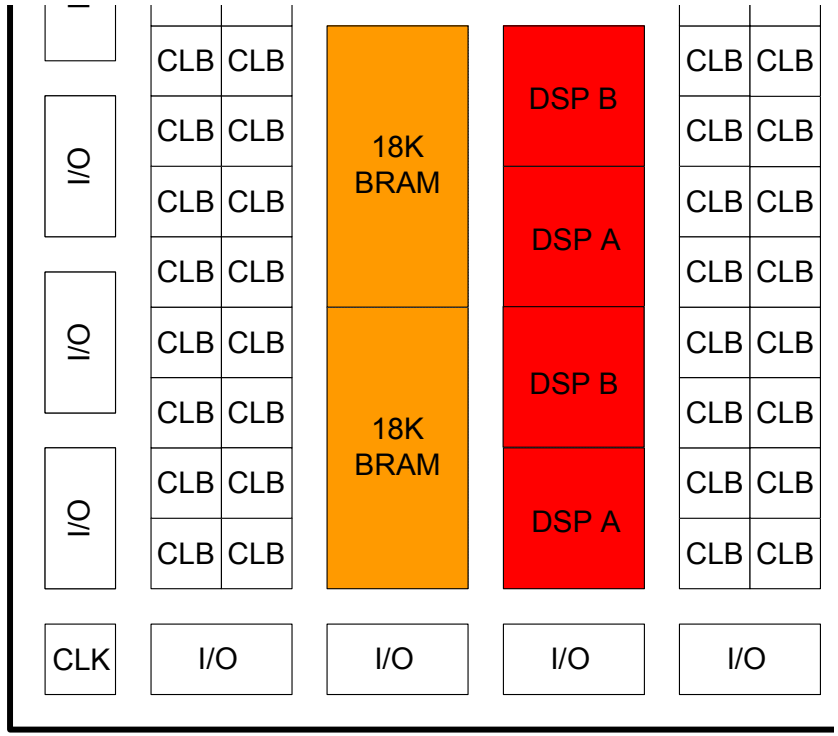
Horst-Görtz Institut für IT Sicherheit

- Introduction and Motivation
- Brief Survey on Reconfigurable Computing and FPGAs
- **Modern FPGA devices and Arithmetic Applications**
- Novel Architectures for ECC over NIST primes
- Results and Conclusions

Features of Modern FPGAs

Horst-Görtz Institut für IT Sicherheit

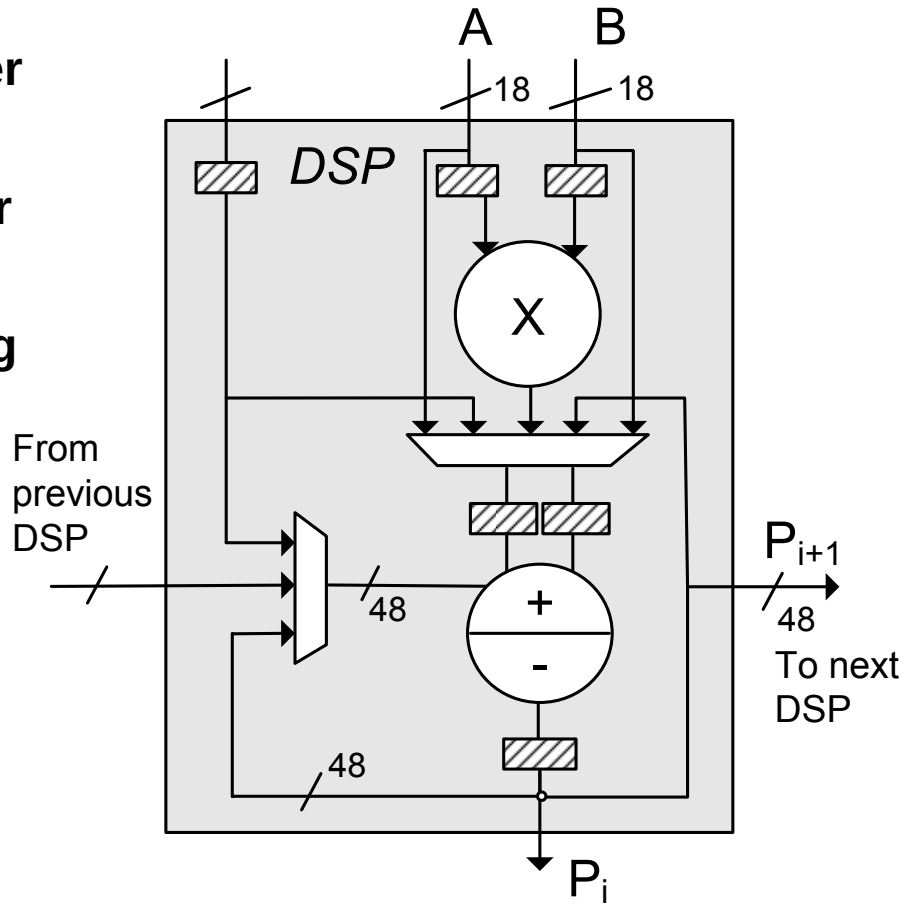
- Generic logic of FPGAs is great but it introduces a lot of **overhead**
- **Performance penalty** due to the dynamic logic w.r.t. to ASICs
- Hence, modern devices provide **additional dedicated functions** like block memories and arithmetic hardcores to accelerate DSP applications
- Since 2003, **DSP hardcores** are integrated, e.g., in Xilinx Virtex 4/5 and Altera Stratix II/II GX devices



Structure of a modern Xilinx Virtex-4 FPGA

DSP block of Virtex-4 Devices

- Contains an **18 bit signed multiplier**
- **48 bit three-input adder/subtracter**
- Can be **cascaded with neighboring** DSP using direct routes
- Can operate at the **maximum device speed** (500 MHz)
- Supports **several operation modes**
  - Adder/subtracter (ADD/SUB)
  - Multiplier (MUL)
  - Multiply & accumulate (MACC)



Multiply-Accumulate Mode (MACC)
$P = P_{i-1} \pm (A \cdot B + Carry)$

Additional Design Rules for DSP Blocks

Horst-Görtz Institut für IT Sicherheit

- For **maximum performance**, designs with DSP function blocks should obey additional rules:
  - **R4:** Use **pipeline register in the DSPs** to avoid performance penalty (they come for free since they are part of the actual hardcore)
  - **R5:** Use **interconnects with neighboring DSPs** wherever possible
  - **R6: Put registers before all input and outputs of the DSPs**
    → resolves placement dependencies between static components
  - **R7:** Use a **separate clock domain** for DSP-based computations
    - *High frequency clock f* (= 500 MHz) **only** for DSP units and their (directly) related inputs/outputs
    - *Half frequency clock f/2* (= 250 MHz) for the remainder of the design, e.g., control logic, communication interfaces, etc.

Agenda

- Introduction and Motivation
- Brief Survey on Reconfigurable Computing and FPGAs
- Modern FPGA devices and Arithmetic Applications
- **Novel Architectures for ECC over NIST primes**
- Results and Conclusions

ECC Hardware Design

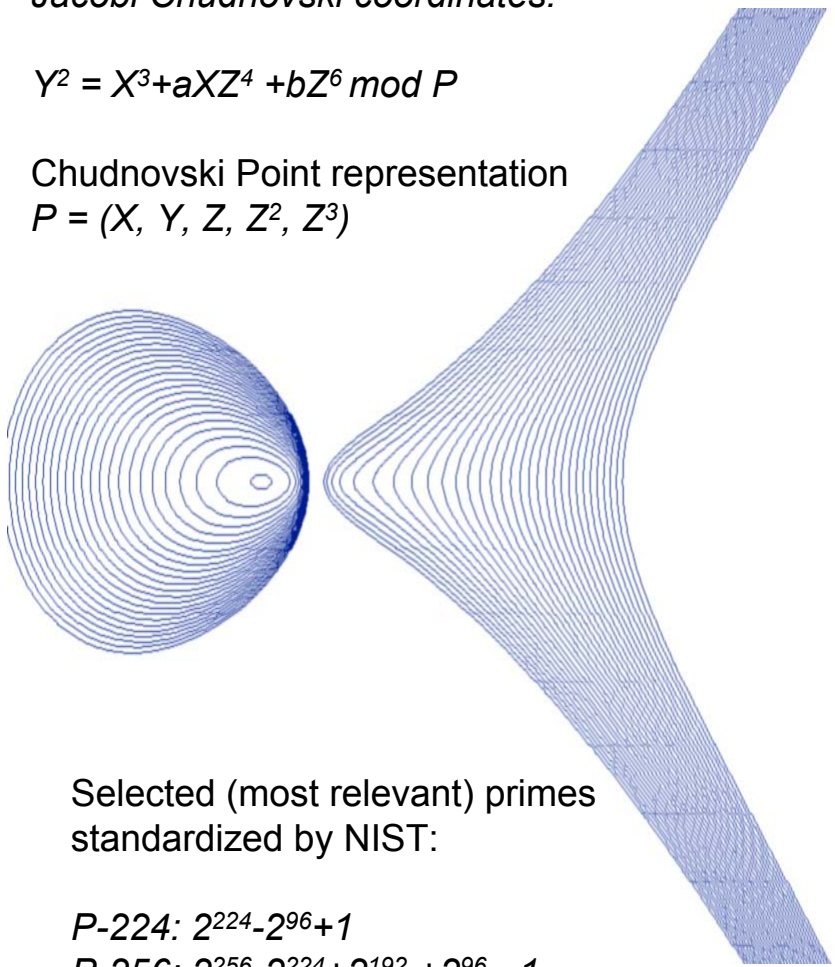
Horst-Görtz Institut für IT Sicherheit

- We use **ECC over prime fields with projective (Chudnovski) coordinates**.
- **Required field operations for ECC**
  - Modular Addition/Subtraction
  - Modular Multiplication
- Field Multiplication with arbitrary primes involves **costly reduction operations** (due to multi-precision divisions!)
- **But:** Generalized Mersenne primes $2^{m1}-2^{m2}\pm \ldots\pm 1$ replace such divisions by a series of additions/subtractions

Weierstraß equation for projective Jacobi Chudnovski coordinates:

$Y^2 = X^3+aXZ^4 +bZ^6 \bmod P$

Chudnovski Point representation
$P = (X, Y, Z, Z^2, Z^3)$



Selected (most relevant) primes standardized by NIST:

P-224: $2^{224}-2^{96}+1$
P-256: $2^{256}-2^{224}+2^{192}+2^{96}-1$

Modular Multiplication using DSPs

Horst-Görtz Institut für IT Sicherheit

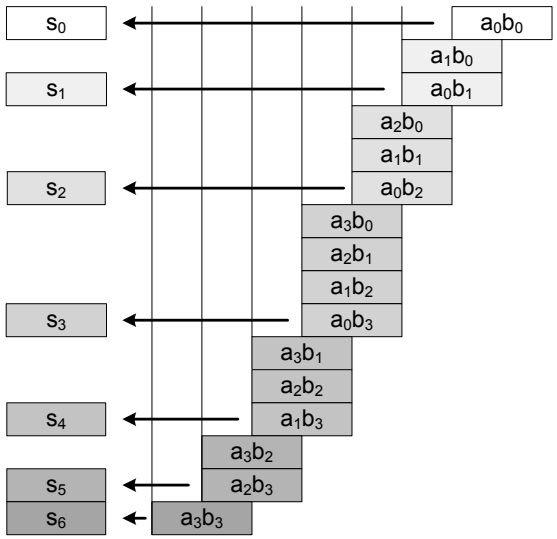
- Modular multiplication with NIST reduction for a full-length multiplication:

$$A \times B = (a_{(n-1)}, \ldots, a_0) \times (b_{(n-1)}, \ldots, b_0)$$

- Multiplication of inner products $a_i \times b_j$ using Comba's method (schoolbook)
- Parallelization of inner product among **several DSPs** by column interleaving

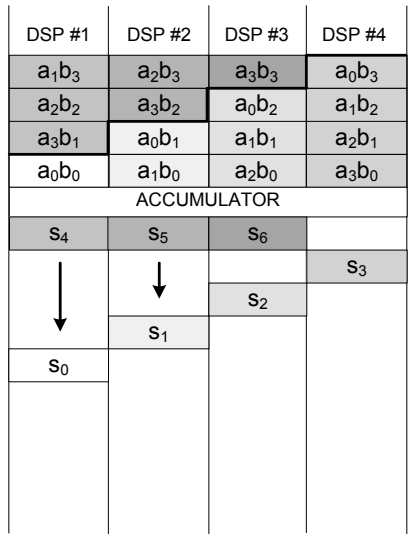
Example for *n=4* with word size ℓ

Standard multiplication *A x B* in product scanning form with single ℓ-bit multiplier



➔ $n^2 = 16$ cycles

Parallel comba multiplication of *A x B* using the MACC function of *n* DSPs



➔ 4 DSP units

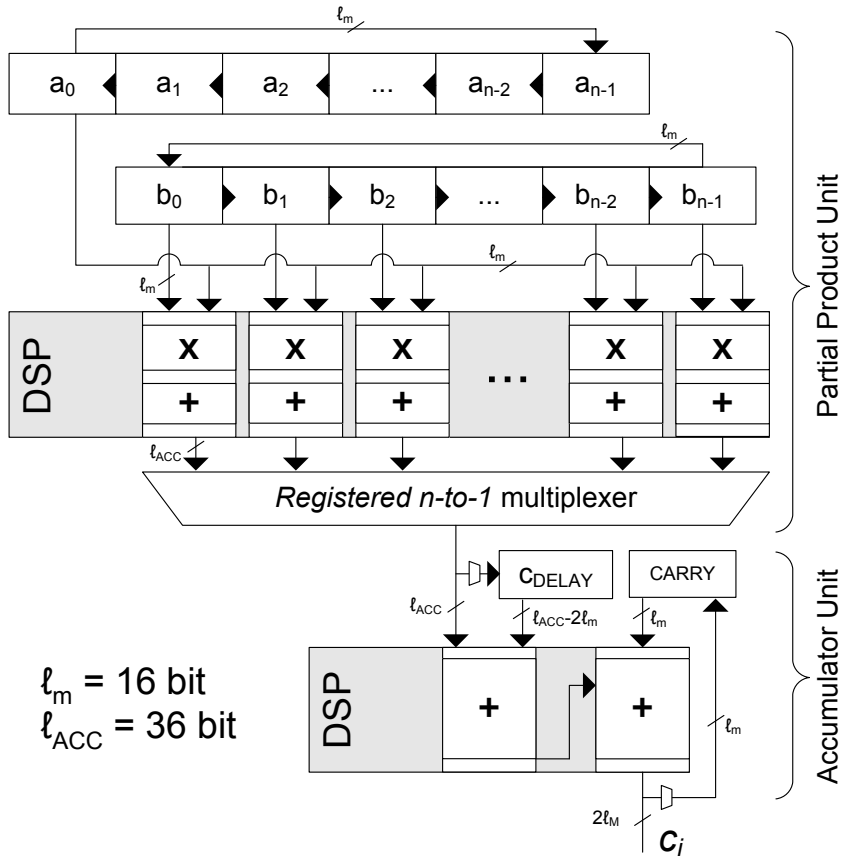
➔ *n = 4* cycles

Parallel Multiplier with DSP blocks

- **Full-width multiplier**
  P-256 → 16 DSP blocks (MACC)
  P-224 → 14 DSP blocks (MACC)

- **Several register stages** are required to compensate routing and logic delays, e.g., of the wide intermediate multiplexer

- **Subsequent accumulator** unit performs shift and alignment of accumulated products S_i
  → another 2 DSP blocks



NIST Reduction Scheme

Horst-Görtz Institut für IT Sicherheit

Algorithm 2 NIST Reduction with P-256 $= 2^{256} - 2^{224} + 2^{192} + 2^{96} - 1$

Input: Double-sized integer $c = (c_{15}, \ldots, c_2, c_1, c_0)$ in base 2^{32} and $0 \geq c \geq \text{P-256}^2$
Output: Single-sized integer c mod P-256.

1: Concatenate c_i to following 256-bit integers z_j:

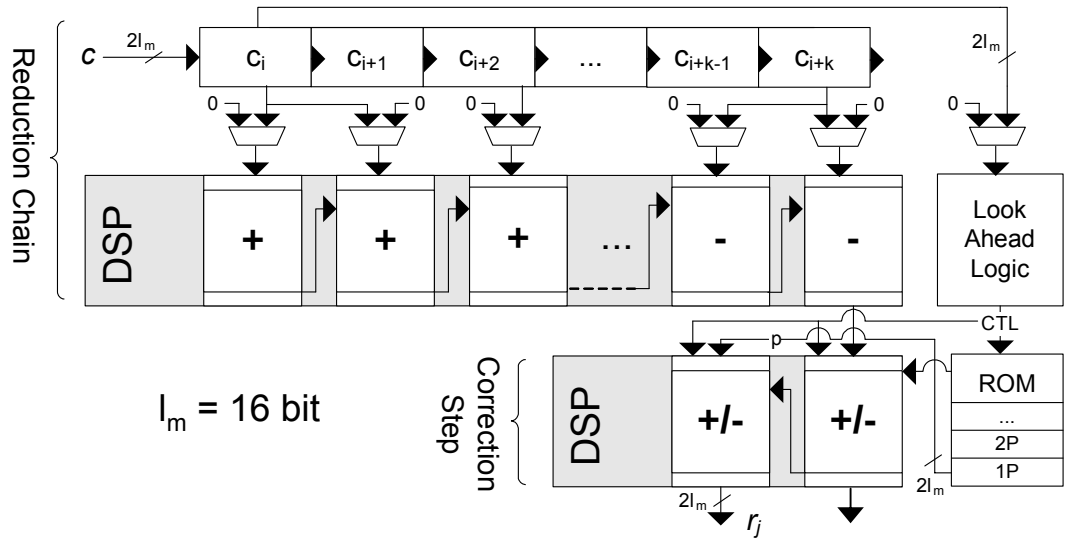
$$z_1 = (c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0), \; z_2 = (c_{15}, c_{14}, c_{13}, c_{12}, c_{11}, 0, 0, 0),$$
$$z_3 = (0, c_{15}, c_{14}, c_{13}, c_{12}, 0, 0, 0), \; z_4 = (c_{15}, c_{14}, 0, 0, 0, c_{10}, c_9, c_8),$$
$$z_5 = (c_8, c_{13}, c_{15}, c_{14}, c_{13}, c_{11}, c_{10}, c_9), \; z_6 = (c_{10}, c_8, 0, 0, 0, c_{13}, c_{12}, c_{11}),$$
$$z_7 = (c_{11}, c_9, 0, 0, c_{15}, c_{14}, c_{13}, c_{12}), \; z_8 = (c_{12}, 0, c_{10}, c_9, c_8, c_{15}, c_{14}, c_{13}),$$
$$z_9 = (c_{13}, 0, c_{11}, c_{10}, c_9, 0, c_{15}, c_{14})$$

2: Compute $c = (z_1 + 2z_2 + 2z_3 + z_4 + z_5 - z_6 - z_7 - z_8 - z_9 \mod \text{P-256})$

Result range
-4P < c < +5P

- Reduction scheme consists of **very basic operations**
  - Step 1: Rearrange words c_i of full product $C = A \times B$
  - Step 2: Add or subtract all rearranged integers z_i
  - Most complicated in hardware: Correct over- or underflow of the final result (requires conditional loop!!)

NIST Reduction with DSP Blocks



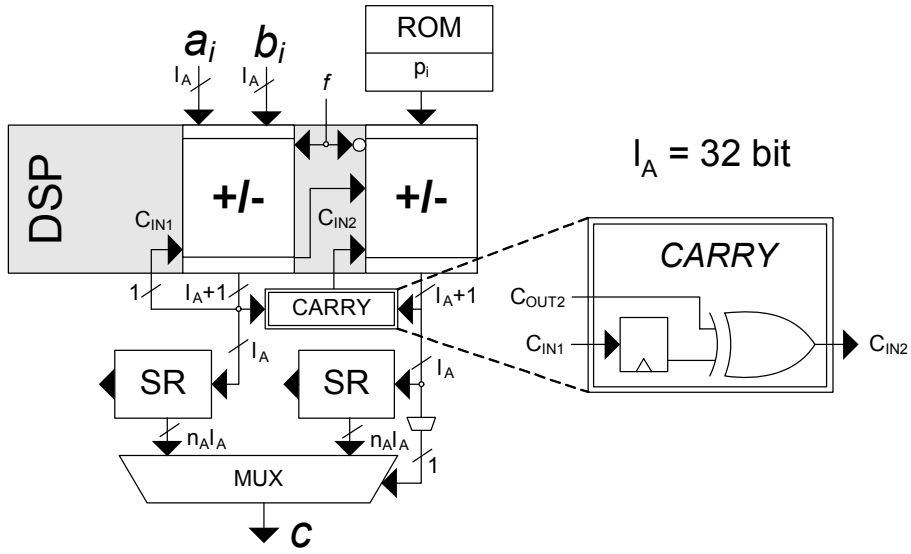
- **DSP block for each main step of addition or subtraction:**
  - 8 units for P-256
  - 4 units for P-224.

- **Look-Ahead Logic** (LAL) estimates the expected over-/underflow by computing the result of the *highest word* of each z_i in advance (using a dedicated DSP block).

- **Two DSP blocks perform the correction of the result range**
  - The first unit adds/subtracts a multiple of *p* dependant on the LAL output (underestimation)
  - The second unit compensates the error due to the previous LAL estimation

Modular Addition/Subtraction with DSP Blocks

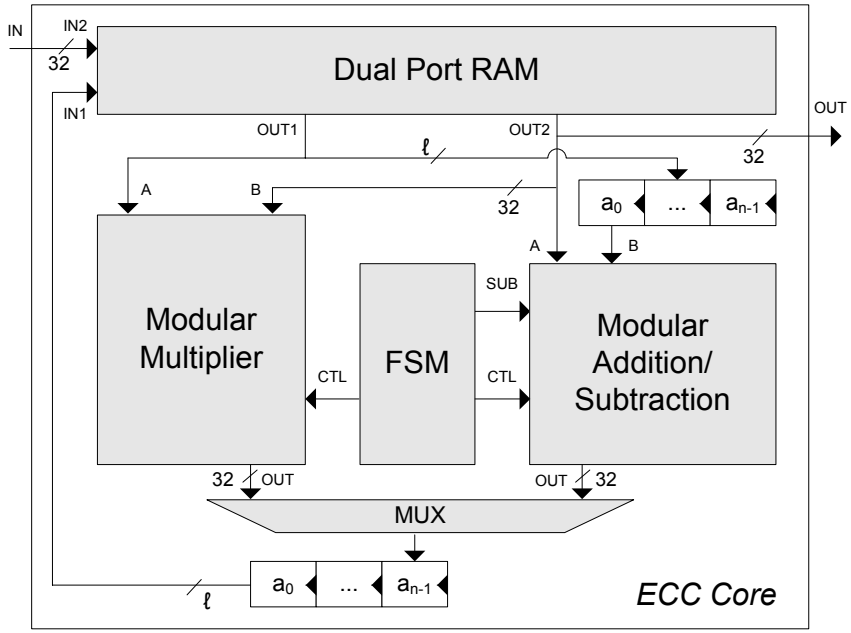
- For **modular addition** one DSP unit computes $S = A + B$ and a neighboring DSP unit $T = C - P$
- **Final shift register stores preliminary output** until it is determined if S or T is the valid result
- **For modular subtraction**, addition and subtraction within the DSPs operations are swapped
- **External carry logic is necessary** since no conditional carry propagation logic between neighboring DSP units is available



Full ECC Architecture

Horst-Görtz Institut für IT Sicherheit

- Common asymmetric **dual-ported memory** provides data both to multiplier and adder/subtracter
- **Use loadable shift registers for inputs/outputs** to decouple routing from wide memory block
- Design with two clock domains
  - **Full frequency domain** for performance-critical DSP operations
  - **Half frequency domain** for control logic and remaining design



Agenda

Horst-Görtz Institut für IT Sicherheit

- Introduction and Motivation
- Brief Survey on Reconfigurable Computing and FPGAs
- Modern FPGA devices and Arithmetic Applications
- Novel Architectures for ECC over NIST primes
- **Results and Conclusions**

A Word of Warning concerning the Comparability of Hardware Designs

Horst-Görtz Institut für IT Sicherheit

- **Problem for evaluation**: comparisons of FPGAs implementations for different devices are often bogus and unfair!

- **Reasons**:
  - Different elliptic curves, parameters and implementation constraints
  - **Different slice structures, features** and **metrics** of FPGA devices
    - 4x6-input LUT (Virtex-5) vs. 2x4-input LUT (Virtex-4) per slice
    - 36k BRAM (Virtex-5) vs. 4k BRAM (Spartan-II)
  - Common metrics like "**operations/slice**" or "**throughput/slice**" for FPGAs cannot be applied (to DSP-based implementations)
  - **Influence of synthesis tools** on the performance of the design
    - Different tool versions
    - Various tool vendors (Xilinx, Synplify, etc.)

Results of this Architecture

Horst-Görtz Institut für IT Sicherheit

- **Single ECC core** implementation on small XC4VFX12 FPGA
- **Multi-core implementation** (up to 16 cores) on large XC4VSX55 FPGA
- **Time column** shows the duration of a single point multiplication

Scheme	Device	Implementation	Logic	Clock	Time
This work	XC4VFX12-12 FPGA	224-bit GF(p), NIST	1580 LS/26 DSP	487 MHz	365 μs
	XC4VFX12-12 FPGA	256-bit GF(p), NIST	1715 LS/32 DSP	490 MHz	495 μs
	XC4VSX55-12 FPGA	224-bit GF(p), NIST	24452 LS/468 DSP	372 MHz	26.5 μs
	XC4VSX55-12 FPGA	256-bit GF(p), NIST	24574 LS/512 DSP	375 MHz	40.5 μs
ECC [23]	XCV1000E FPGA	192-bit GF(p), NIST	5708 LS	40 MHz	3 ms
ECC [19]	XC2VP125-7	256-bit GF(p), any	15755 LS/256 MUL	39.5 MHz	3.84 ms
ECC [24]	0.13 μm CMOS	160-bit GF(p), any	117500 GE	137.7 MHz	1.21 ms
ECC [3]	Intel Pentium4	224-bit GF(p), NIST	32 bit μP	1.4 GHz	599 μs
ECC [11]	Intel Core2 Duo	256-bit GF(p), NIST	64 bit μP	2.13 GHz	669[a] μs
ECC [13]	Intel Core2 Duo	255-bit GF($2^{255} - 19$)	64 bit μP	2.66 GHz	145 μs
RSA[4]	XC40250XV FPGA	1024bit, high radix-16	6826 CLB	45.2 MHz	3.1 ms
RSA[27]	XC4VFX12-10 FPGA	1024-bit with DSP	3937 LS/17 DSP	400 MHz	1.71 ms
RSA[25]	0.5 μm CMOS	1024-bit, unified	28,000 GE	64 MHz	46 ms

[a] Note that this figure reflects a full ECDSA signature generation rather than a point multiplication.

Conclusions

Horst-Görtz Institut für IT Sicherheit

- To our knowledge, **fastest ECC engine for FPGAs** (for NIST primes)
- This design **closes the gap** between ECC engines on high-end CPUs and hardware approaches
- Little resource consumption allows further functions on same FPGA
- Estimation: Up to **37.000 point multiplications/sec** for P-224 using sliding window (*w=4*) are feasible
- **Heat dissipation** is a big issue, especially in embedded applications (requires extensive cooling or lower clock frequency)
- Note that the cost-performance ratio of Intel Core 2 Duo is still better than that of a Virtex-4 FPGA for 256 bit ECC:
  - **Core 2 Duo:** 6900 ops/sec @ $180 → 38 ops/sec for $1
  - **Xilinx XC4VSX55:** 24700 ops/sec @ $1170 → 21 ops/sec for $1

Thanks for your attention!

Horst-Görtz Institut für IT Sicherheit

Questions?

Tim Güneysu
gueneysu@crypto.rub.de