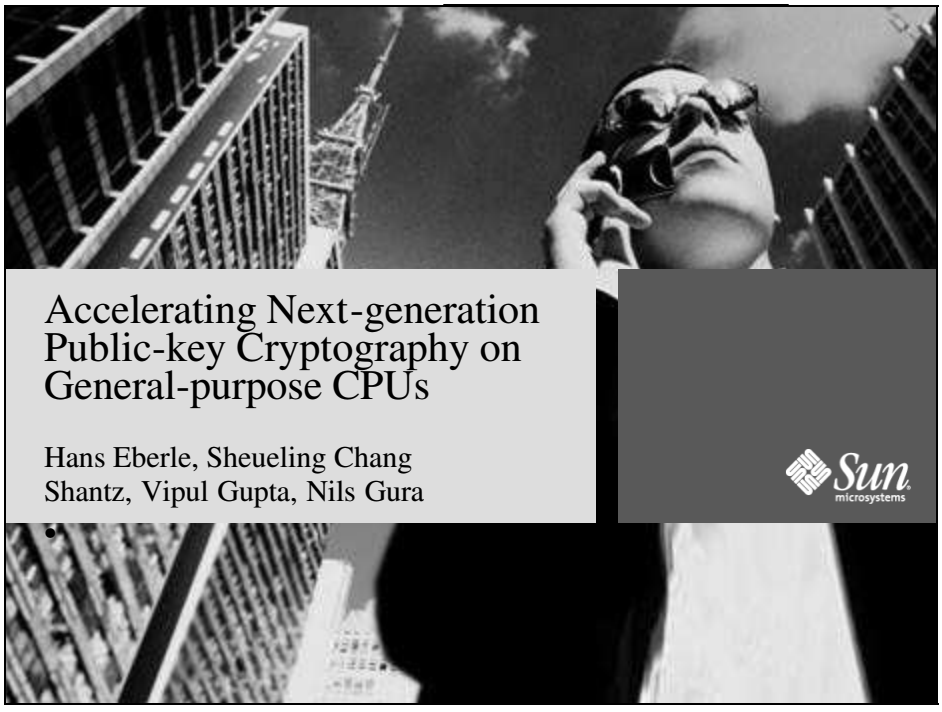


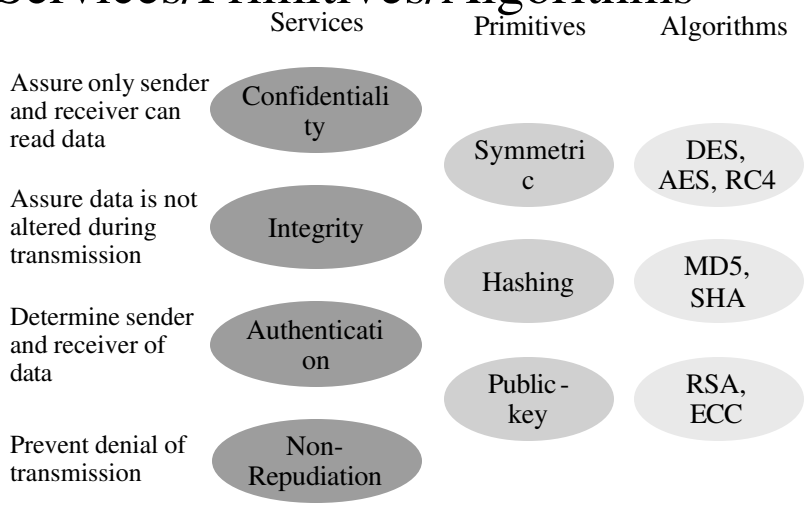
# Accelerating Next-generation Public-key Cryptography on General-purpose CPUs

Hans Eberle, Sheueling Chang Shantz, Vipul Gupta, Nils Gura

## Goals

- Accelerate public-key cryptosystems on SPARC CPUs
  - Next-generation cryptosystem ECC
  - Legacy cryptosystem RSA
- Provide server-class performance for the next level of Internet security
- Aim at light-weight implementation that reuses data path of a general-purpose SPARC CPU
- Make cryptographic functionality an integral part of future SPARC CPUs

# Security Services/Primitives/Algorithms




3

# Need for Overhauling Internet Security

Web Clients (IE, Mozilla)

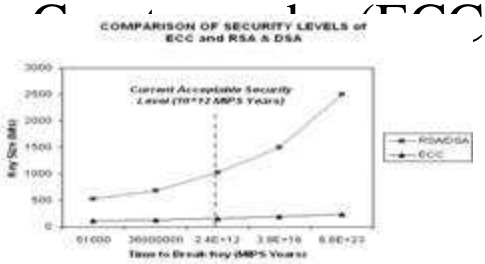

Sun Java™ Enterprise System, Apache Web Server

- **Growing e-commerce and wireless markets**
- **More light-weight devices connected to Internet (PDAs, mobile phones, sensors, etc.)**
- **Insufficient security offered by RSA-1024, 3DES, RC4, MD5:**

| | Today | Tomorrow |
|---|---|---|
| Public-key | RSA | RSA, ECC |
| Symmetric | DES, 3DES, RC4 | AES |
| Hashing | SHA1, MD5 | SHA256 |

4

# Next-Generation Public-Key Cryptosystem: Elliptic Curve Cryptography (ECC)




RSA/ECC Keysize Growth Ratio

| Sym. | RSA | ECC | Ratio | MIPS yrs |
|---|---|---|---|---|
| 80 | 1,024 | 160 | 6:1 | $10^{12}$ |
| 112 | 2,048 | 224 | 9:1 | $10^{24}$ |
| 128 | 3,072 | 256 | 12:1 | $10^{28}$ |
| 192 | 7,680 | 384 | 20:1 | $10^{47}$ |
| 256 | 15,360 | 521 | 30:1 | $10^{66}$ |

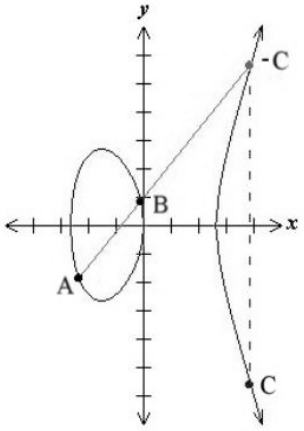
- Computationally most efficient public-key cryptosystem, highest security strength per bit
  - Savings in compute power, memory capacity, bandwidth, power consumption
  - Advantage improves as security needs increase
- Endorsed/standardized by NIST, ANSI, IEEE, IETF
- Good match for AES

5

# ECC Point Multiplication

- ECC operation
  point multiplication
  $Q(x,y) = k * P(x,y)$

  Q = public key
  k = private key
  P = base point (curve parameter)
- Hard problem
  given public key kP find private key k
  (Discrete Logarithm Problem,
  no known subexponential solutions)
- Repeated **point additions** and doubling
  $9P = P + 2 * (2 * (2 * P))$
- Curves defined over binary polynomial fields $GF(2^m)$ and prime integer fields GF(p)




6

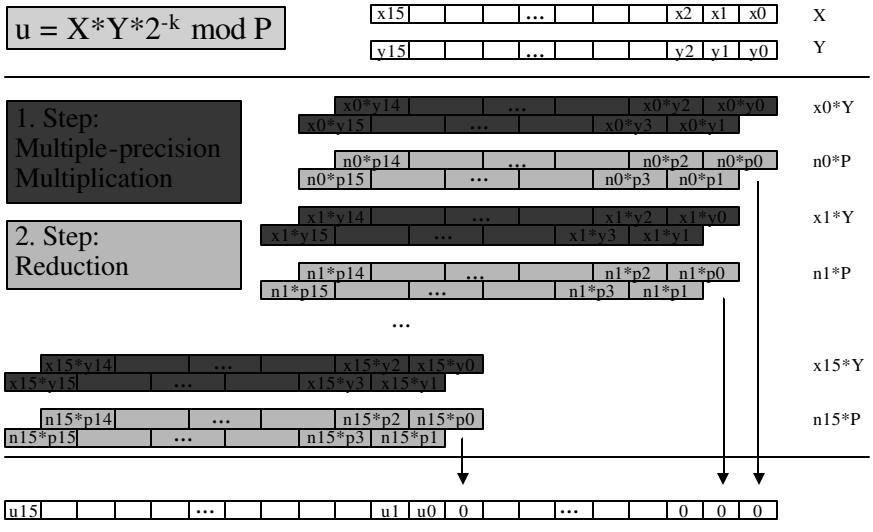
# Accelerating Public-key Cryptography

- RSA is based on modular exponentiation
  $C = M^e \bmod n$
- ECC is based on point multiplication
  $Q = k * P$
- Modular multiple-precision multiplication is the key underlying function
- ECC math is more complicated; also requires multiple-precision addition/subtraction/division
- Optimizations
  - Algorithms, ISA, firmware, circuits

7

# Montgomery Multiplication

$u = X*Y*2^{-k} \bmod P$




8

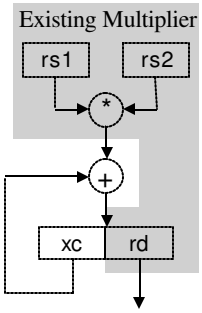
# Acceleration Techniques

- Optimizing modular multiple-precision multiplication achieves highest performance gain for RSA and ECC
  - Reuse CPU data path for public-key crypto operations
- Dual-field multiplier
  - Integer multiplication result for RSA and ECC GF(p)
  - XOR multiplication result for ECC GF($2^m$)
- Efficient scheduling
  - Goal: keep multiplier busy at all times
  - Combined multiplication/accumulation step



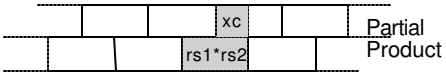
# Multiply-accumulate Primitive

**Enhanced general-purpose multiplier**




mulacc rs1,rs2,rd

rd = (rs1*rs2+xc)[63:0]
xc = (rs1*rs2+xc)[127:64]




Multiple-precision multiplication support:

- 1 mulacc replaces 1 mul & 1 add
- Reduces register pressure
- >2x performance improvement
- Same performance as 2 parallel multipliers

# XOR Multiplication

GF($2^m$) Addition

- Corresponds to a bitwise XOR

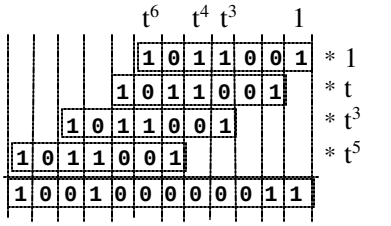
$(t^6+t^4+t^3+1) + (t^5+t^3+t+1) = t^6+t^5+t^4+t$

1011001 XOR 0101011 = 1110010

GF($2^m$) Multiplication

- Corresponds to XOR of partial products
- No carries

$(t^6+t^4+t^3+1) * (t^5+t^3+t+1) = t^{11}+t^8+t+1$

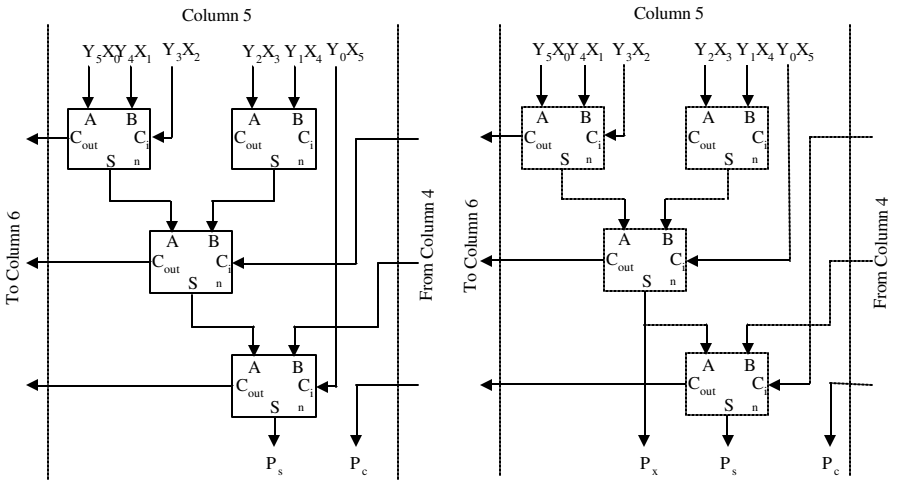





# Dual-field Multiplier

- Generates multiplication results over GF(p) and GF($2^m$)
- Modified integer multiplier
- Low cost: 5% size increase for 64-bit multiplier
- High performance
  - Over 100 instructions w/o XOR multiplication
  - 13.8x speedup for multiple-precision multiplication
  - 8.5x speedup for ECC point multiplication
- Rearrange carry-save-adder tree to obtain additional XOR multiplication result:
  - FA: $S = A \oplus B \oplus C_{in}$ $C_{out} = A \cdot B + A \cdot C_{in} + B \cdot C_{in}$
  - HA: $S = A \oplus B$ $C_{out} = A \cdot B$

# Modified Carry Save Adder Tree



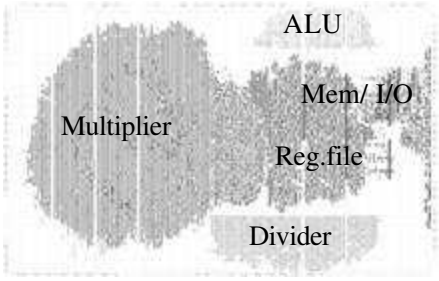

Integer Multiplier

Dual-Field Multiplier

13

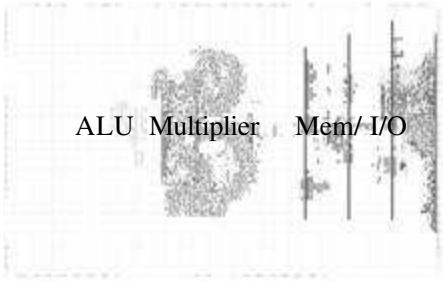
# Crypto Accelerator FPGA Prototypes

1st Gen. ECC Accelerator




- Supports ECC GF($2^m$)
- Dedicated 256-bit coprocessor
- Fastest reported ECC implementation (66 MHz, 3..4-cycle non-pipelined 256x256 mul)
  - 6,987 ECC-163 op/s

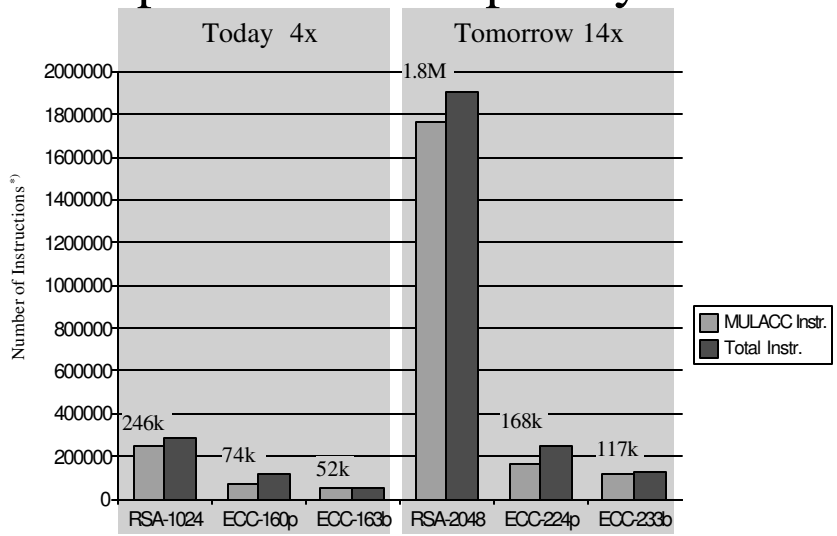
2nd Gen. ECC & RSA Accelerator




- Supports ECC GF($2^m$), ECC GF(p
- General-purpose 64-bit processor
- Projected performance (1.5 GHz, 2-cycle 64x64 pipelined mul)
  - 10,000 ECC-163 op/s
  - 2,500 RSA-1024 op/s

14

# Computational Complexity




*)2nd Gen. ECC & RSA Accelerator

15

# Crypto Core for Next-Generation SPARC®

- 1.5 GHz shared datapath
- Dual-field pipelined 64-bit multiplier
- FSM for RSA modular exponentiation & ECC point multiplication

| Today | op/s |
|---|---|
| ECC-163 | 12,335 |
| RSA-1024 | 4,998 |

| Tomorrow | op/s |
|---|---|
| ECC-233 | 6397 |
| RSA-2048 | 834 |

16

# Public-Key Crypto on Small Devices

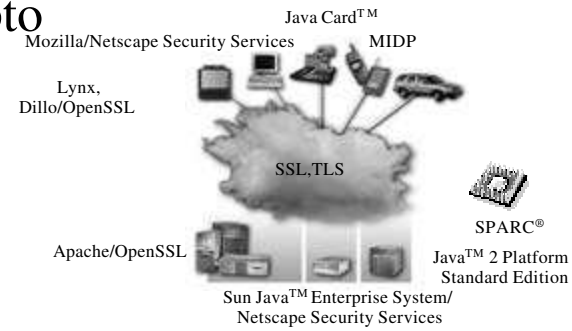
| Today | | | Tomorrow | | |
|---|---|---|---|---|---|
| | Time [s] | Speedup ECC:RSA | | Time [s] | Speedup ECC:RSA |
| RSA-1024[1)2)] | 10.99 | 1 | RSA-2048[1)2)] | 83.26 | 1 |
| ECC-160[3)] | 0.81 | 13.6 | ECC-224[2)] | 2.19 | 38 |
| ECC-163[3)] | 0.29 | 37.9 | ECC-233[3)] | 0.81 | 102.8 |

[1)] Private-key operations, CRT
[2)] Unmodified architecture
[3)] Extended architecture: dual-field multiplier, mulacc

- 8-bit microcontroller
  - (Extended) ATmega128 AVR, 8 MHz
  - Used for Motes sensor networks
- Sizzle
  - Small, server-side HTTPS stack for embedded systems
  - Interoperable with ECC-enabled Mozilla/OpenSSL



# Supporting Next-generation Public-key Crypto




- The computational efficiency of ECC enables public-key cryptography on light-weight and mobile devices
- Public-key cryptography can be efficiently supported on next-generation SPARC CPUs
  - Shared dual-field multiplier
  - Firmware or ISA extensions for optimal instruction scheduling

For more information:

http://research.sun.com/projects/crypto/