Low-power Elliptic Curve Crypto Processor in 130nm technology

Vladimir Rožić, Yong Ki Lee, Junfeng Fan, Miroslav Knežević, Duško Karaklajić, Roel Maes, Lejla Batina and Ingrid Verbauwhede

Outline

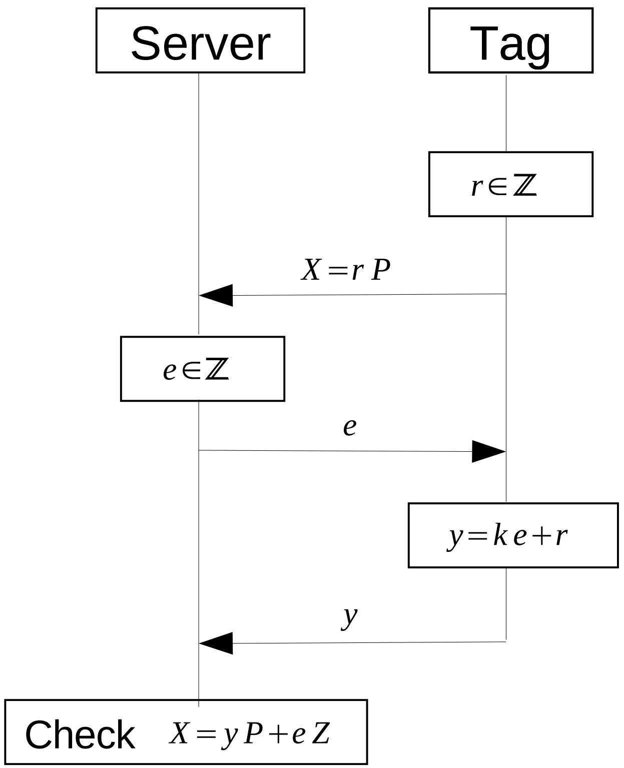
- Goal
- Background
- Architecture
- Testing Strategy
- FPGA prototype
- ASIC

Goal

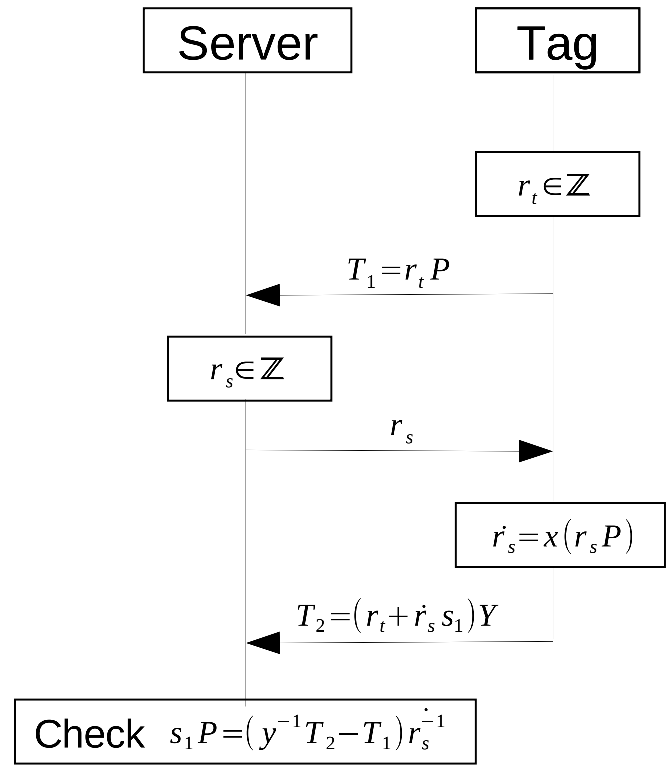
- Public-key cryptography on RFID tags.
  - Compact
  - Low power
  - Low latency

RFID authentication protocols

Schnorr's protocol

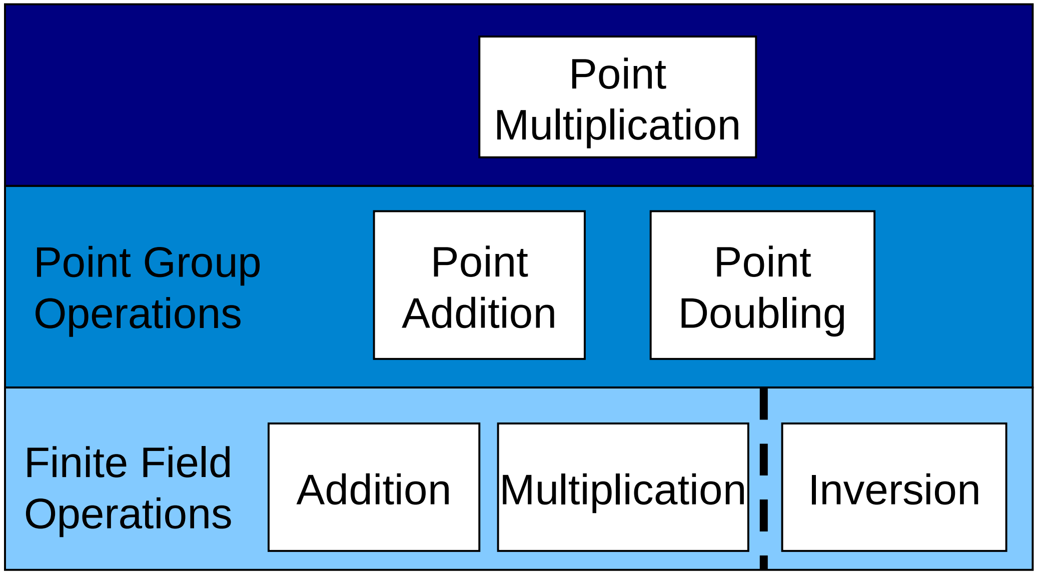


EC-RAC



EC operations

- Elliptic curve defined over GF(2^{163})
- Scalar Point Multiplication
- Montgomery Ladder
- Projective coordinates
- Common-Z coordinate system



Montgomery ladder

INPUT: Elliptic curve point *P*
t-bit integer *k>0*

OUTPUT: *k P*

$k \leftarrow 1, k_{t-2}, \ldots, k_1, k_0$

$P_1 \leftarrow P, P_2 \leftarrow 2\mathrm{P}$

$for\ i \leftarrow (t-2)\ downto\ 0\ do$

$if\ k=1\ then\ P_1 \leftarrow P_1 + P_2, P_2 \leftarrow 2P_1$

$else\ P_2 \leftarrow P_1 + P_2, P_1 \leftarrow 2P_1$

$end\ for$

$Return\ P_1$

- Algorithm for EC scalar multiplication
- Balanced computation
- Side-channel secure

Projective coordinates

- Point on an elliptic curve is represented with three coordinates such that

$$x = \frac{X}{Z} \qquad y = \frac{Y}{Z^2}$$

- Redundant coordinate is used to avoid the field inversion and to reduce the amount of computation.

Lopez- Dahab algorithm

$k \leftarrow k_{l-1} \ldots k_1 k_0$

$X_1 \leftarrow x, Z_1 \leftarrow 1, X_2 \leftarrow x^4 + b, Z_2 \leftarrow x^2$

$for\ i \leftarrow (t-2)\ downto\ 0\ do$

$\quad if\ k_i = 1\ then$

$\qquad (X_1, Z_1) \leftarrow Madd(X_1, Z_1, X_2, Z_2),$

$\qquad (X_2, Z_2) \leftarrow Mdouble(X_2, Z_2)$

$\quad else\ (X_2, Z_2) \leftarrow Madd(X_2, Z_2, X_1, Z_1),$

$\qquad (X_1, Z_1) \leftarrow Mdouble(X_1, Z_1)$

$end\ for$

$Return\ Q \leftarrow Mxy(X_1, Z_1, X_2, Z_2)$

Addition

$$Z_{Add} = (X_1 Z_2 + X_2 Z_1)^2$$
$$X_{Add} = x Z_{Add} + (X_1 Z_2)(X_2 Z_1)$$

Doubling

$$Z_{Double} = (X_1 Z_2)^2$$
$$X_{Double} = X_2^4 + b Z_2^4$$

- Addition of points whose difference is known
- No need to store the value of the Y coordinate

Common Z coordinate system

- Both points have the same Z-coordinate value in every loop run
- Reduced number of registers
- Additional operations required

Addition

$$Z_{Add} = (X_1 + X_2)^2 Z^2$$

$$X_{Add} = x Z_{Add} + (X_1 X_2 Z^2)$$

Doubling

$$Z_{Double} = (X_2 Z)^2$$

$$X_{Double} = (X_2^2 + c Z^2)^2$$

Additional steps

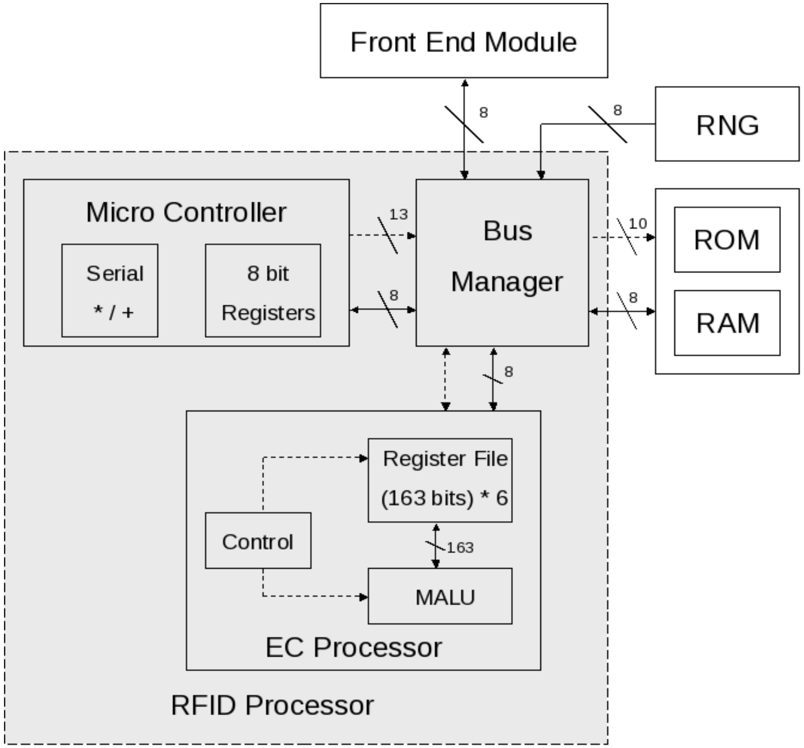
$$X_1 \leftarrow X_{Add} Z_{Double} = (x (X_1 + X_2)^2 + X_1 X_2)(X_2 Z)^2$$

$$X_2 \leftarrow X_{Double} Z_{Add} = (X_2^2 + c Z^2)^2 (X_1 + X_2)^2$$

$$Z \leftarrow Z_{Add} Z_{Double} = (X_1 + X_2)^2 (X_2 Z)^2$$

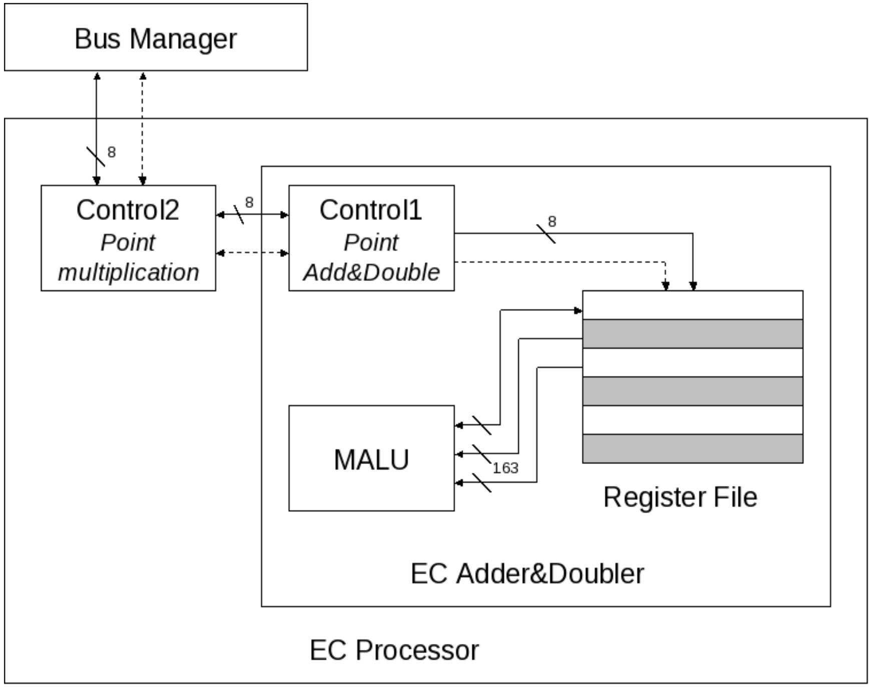
System Architecture

- EC Processor
  - Scalar point multiplication
- Micro controller
  - Executing the protocol
  - Modular addition and multiplication
- RAM and ROM
  - Storing the program and system parameters



EC Processor

- Reads an EC point and a scalar value, performs multiplication and writes the value in memory
- 2-level FSM
- 6 registers
- 163-bit ALU
- Digit-serial modular multiplication (d=4)



Micro controller

- 8-bit ALU
- Digit-serial addition and multiplication
- Block based addressing
  - Operates on 21-byte data blocks

Instruction set

Instruction	Description
Block_Mov (A, B)	Move one block of data from location B to location A
Block_Add (A, B)	Add the data stored at locations A and B and store the result in RAM[0]
Block_Mul (A, B)	Multiply the data stored at locations A and B and store the result in RAM[0]
Block_Comp (A, B)	Compare the data stored at locations A and B
Cond_Jump	Conditional Jump
Activate_ECP (A)	Start the ECP multiplication
Wait for ECP	Wait for the end of ECP multiplication
End_of_code	The end of the program

Schnorr's protocol

Block_Mov (RAM[4], RNG)	*Read r from RNG*
Activate_ECP (ROM[0])	$X = r P$
Wait_for_ECP	*Wait for the completion of ECP*
Block_Mov (Transmitter, RAM[2])	*Transmit X*
Block_Mov (RAM[0], Receiver)	*Receive e*
Block_Mul (RAM[0], ROM[1])	*Multiply a and e*
Block_Add (RAM[0], RAM[4])	$y = a e + r$
Block_Mov (Transmitter, RAM[0])	*Transmit y*

KATHOLIEKE UNIVERSITEIT LEUVEN

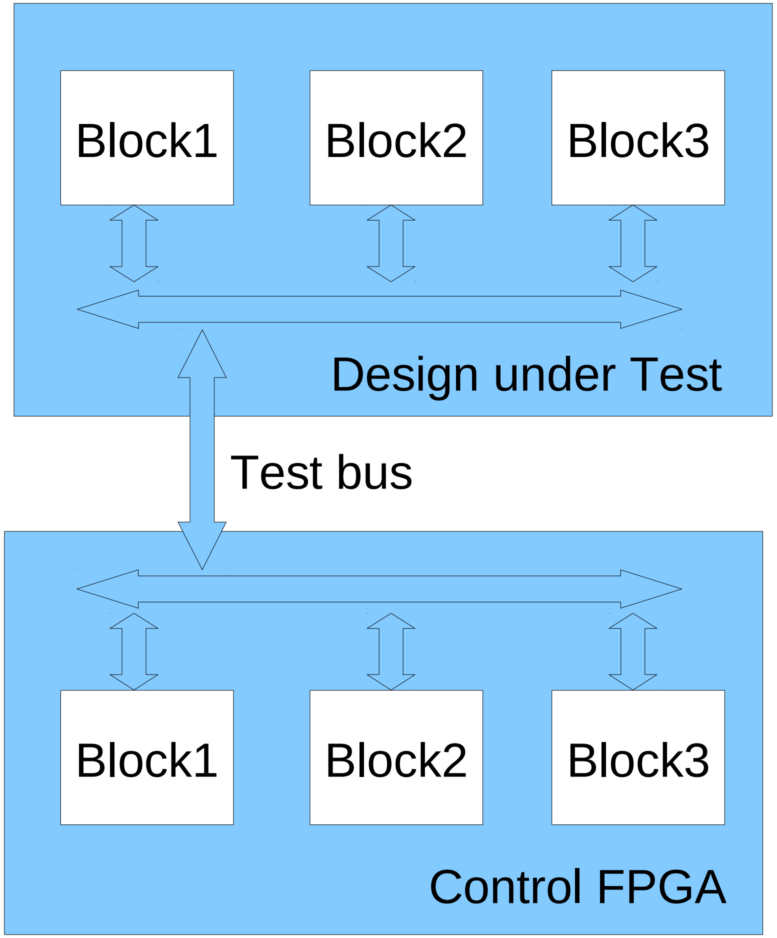
COSIC COMPUTER SECURITY AND INDUSTRIAL CRYPTOGRAPHY

EC-RAC

Block_Mov (RAM[4], RNG)	*Read r from RNG*
Activate_ECP (ROM[0])	$T_1 = r_t P$
Wait_for_ECP	*Wait for the completion of ECP*
Block_Mov (Transmitter, RAM[2])	*Transmit X*
Block_Mov (RAM[3], RAM[4])	*Move* r_t *to RAM[3]*
Block_Mov (RAM[4], Receiver)	*Receive* r_s
Activate_ECP (ROM[0])	$\dot{r}_s = x(r_s P)$
Wait_for_ECP	*Wait for the completion of ECP*
Block_Mov (RAM[1], RAM[2])	*Move* r_s *to RAM[1]*
Block_Mul (RAM[1], ROM[1])	$\dot{r}_s s_1$
Block_Add (RAM[3], RAM[0])	$v = r_t + \dot{r}_s s_1$
Block_Mov (RAM[4], RAM[0])	*Move v to RAM[4]*
Activate_ECP (ROM[2])	$T_2 = vY$
Wait_for_ECP	*Wait for the completion of ECP*
Block_Mov (Transmitter, RAM[2])	*Transmit* T_2

Test Strategy

- Block-based Design
- Shadow implementation
- Control FPGA



SASEBO

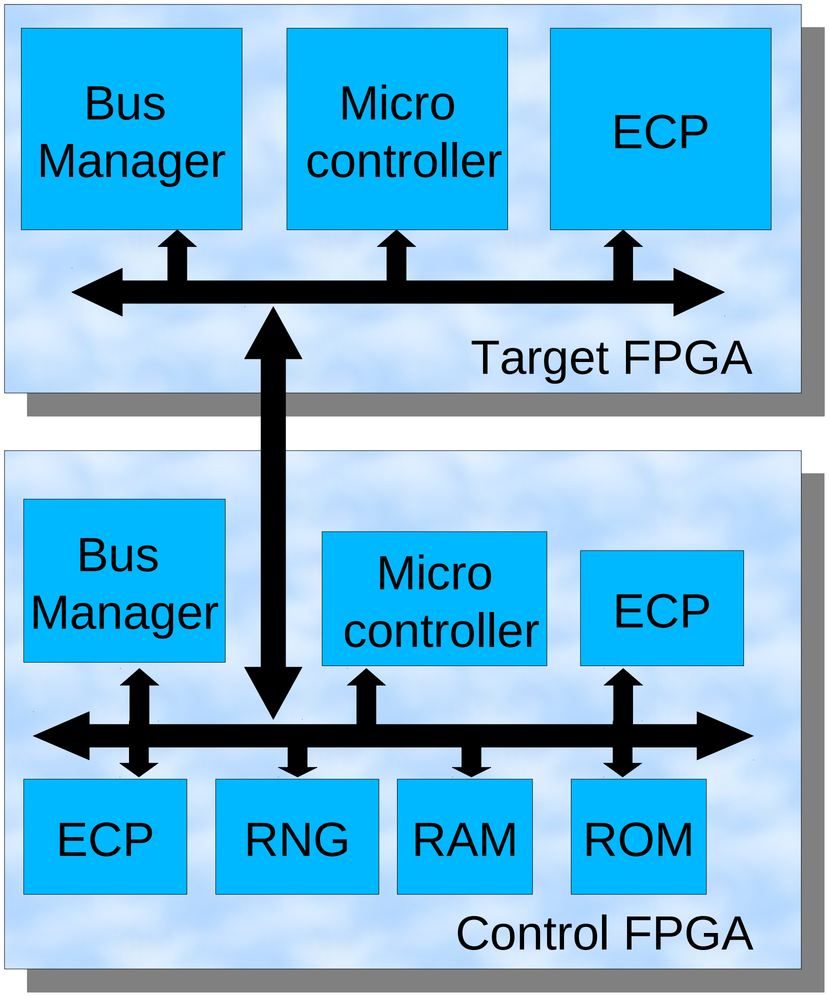
- Side-channel attack standard evaluation board
  - Standard platform for side-channel attack experiments
- Developed by Research Center for Information Security (RCIS), Akihabara, Tokyo
- For this project we used two versions of SASEBO
  - SASEBO G : Contains two Xilinx Virtex II Pro FPGA devices (xc2vp7 and xc2vp30)
  - SASEBO R: Contains xc2vp30 and ASIC

KATHOLIEKE UNIVERSITEIT LEUVEN

FPGA prototype

COSIC
COMPUTER SECURITY AND INDUSTRIAL CRYPTOGRAPHY

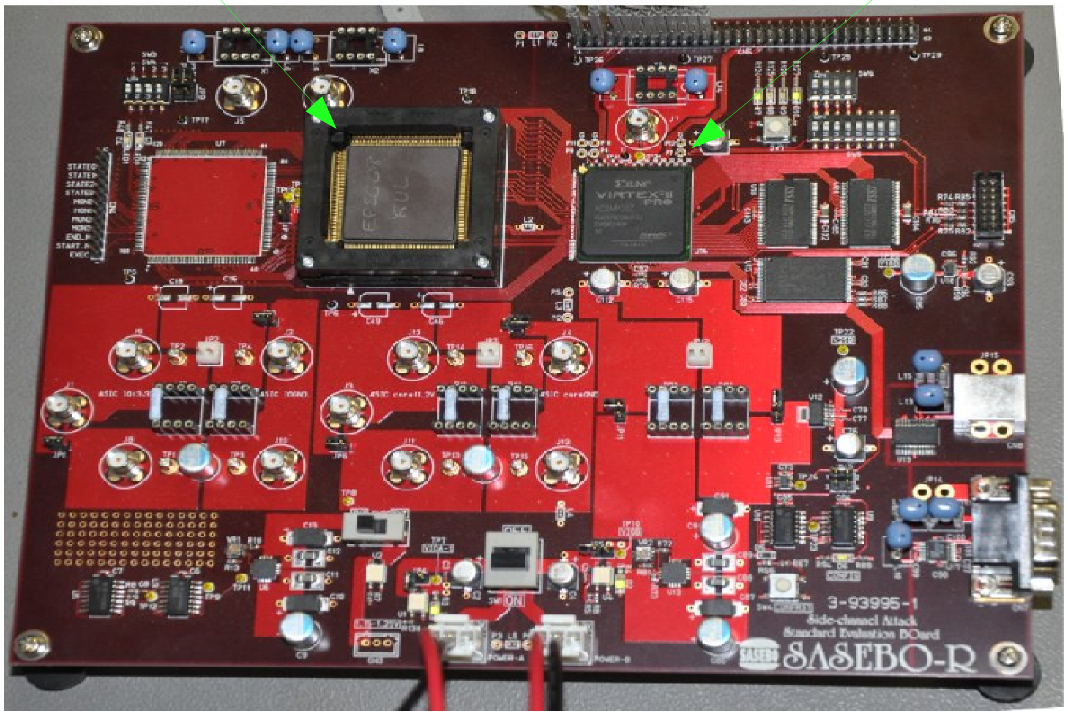
- Normal mode
- Test mode
  - Each component of the chip can be tested separately



Test Board

Cryptographic Chip

Control FPGA



Chip features

Technology	UMC 130 nm 1P8M CMOS
Supply Voltage	Core 1.2 V, I/O 3.3 V
Core Area	735 µm by 735 µm
Operating Frequency	847 kHz
Power consumption	50.4 µW
Throughput	9.8 point multiplications / s
Energy consumption	5.1 µJ / point multiplication

Conclusion

- Public key cryptography is suitable for use in RFID systems
  - Low power
  - Compact
  - Small number of cycles
- Future work
  - Evaluation of side-channel security
  - Resistance against fault attacks