New Directions in Lightweight Cryptographic Primitives for RFID Applications

RFID CUSP Workshop

January 23-24, 2008

John Hopkins University

Christof Paar

University of Bochum and escrypt Inc. – Embedded Security

www.crypto.rub.de

Acknowledgements

Joint work with

- Sandeep Kumar
- Lars Knudsen
- Gregor Leander
- Axel Poschmann
- Matt Robshaw
- Kai Schramm

Contents

1. **Some general thoughts about cheap crypto**
2. Lightweight Block Ciphers
3. Lightweight Asymmetric Cryptography
4. Lightweight Hash Functions

Why Do We Need Cheap Crypto?

1. **There is no other choice (aka RFID)**
   "We need security with less than 2000 gates"
   Sanjay Sarma, AUTO-ID Labs, CHES 2002

2. **There is another choice, but we like a long battery life**
   Small ciphers improve usability of mobile devices

3. **There is another choice, but we like to save money**
   A cipher X that saves $0.01 over cipher Y can be very attractive in many products (esp. in high volume applications!)
   $\Rightarrow$ Important for the myriad pervasive computing devices

Approaches to Lighweight Crypto

1. **Design highly efficient implementation of established cipher**, e.g., AES, ECC
   Ex: [Feldhofer et al., CHES 04]

2. **Choose established cipher with short parameters**
   (works mainly for asymmetric schemes)
   Ex: SECG standards, ECC with 112bit etc.

3. **Design new lightweight ciphers**
   Ex: PRESENT, eSTREAM

Note: Option 3 is promising but daring.

New Lightweight Ciphers vs. Standardized Ciphers

- Most standardized ciphers (AES, 3DES, ECC, DSA,…) are by definition **universal** ciphers.
- Universal ciphers must provide very high security for **all possible** applications, costs are secondary
- Domain-specific ciphers (here: lightweight) can be better match for certain applications
- BIG question: security!

Lightweight ciphers exploit the trust-performance trade-off

Read: If possible, use AES – if you want to trade trust-in-cipher for costs, use PRESENT or such.

The cryptographic toolkit

Cryptographic Algorithms

Symmetric

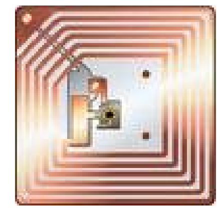
Public-key

Hash functions

Contents

Lightweight Cryptography

- "We need security with less than 2000 gates"
  Sanjay Sarma, AUTO-ID Labs, CHES 2002



- $3 trillions annually due to product piracy* (> US budget '07)

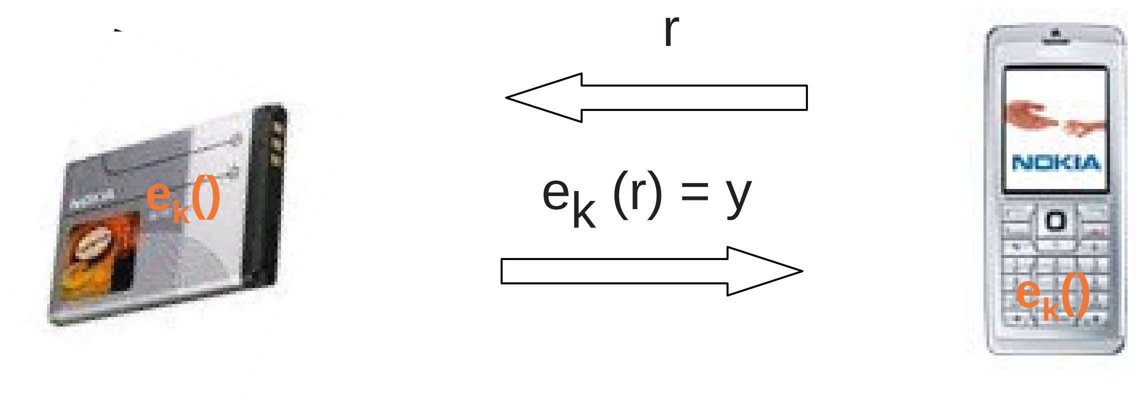
*Source: www.bascap.com

⇒ Authentication & identification problem: can both be fixed with cryptography

⇒ How cheap can we make symmetric ciphers?

eurobits
European Competence Center for IT Security

Strong Identification (w/ symmetric crypto)

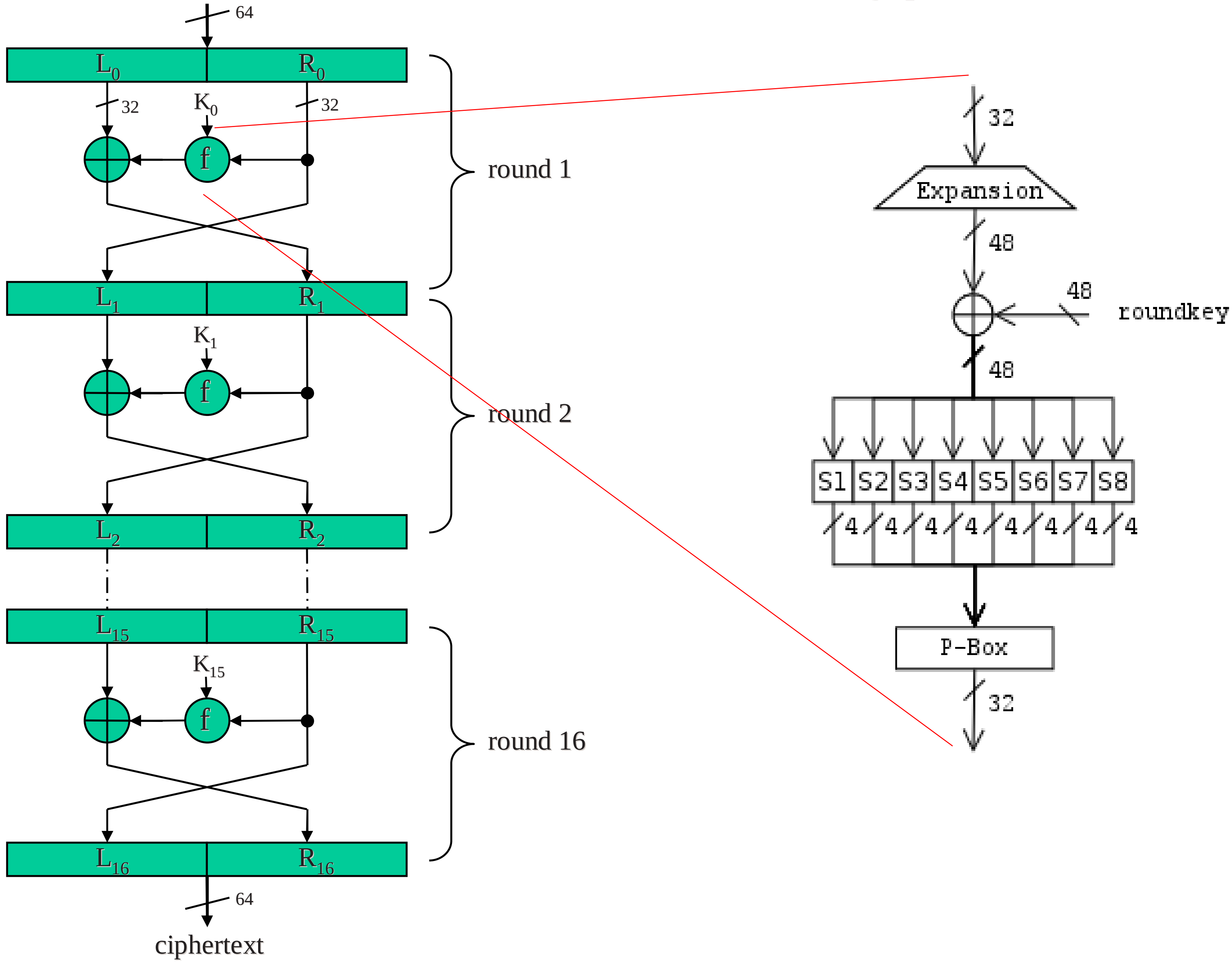


1. random challenge r
2. encrypted response y
3. verification
   $e_k(r) = y'$
   $y == y'$

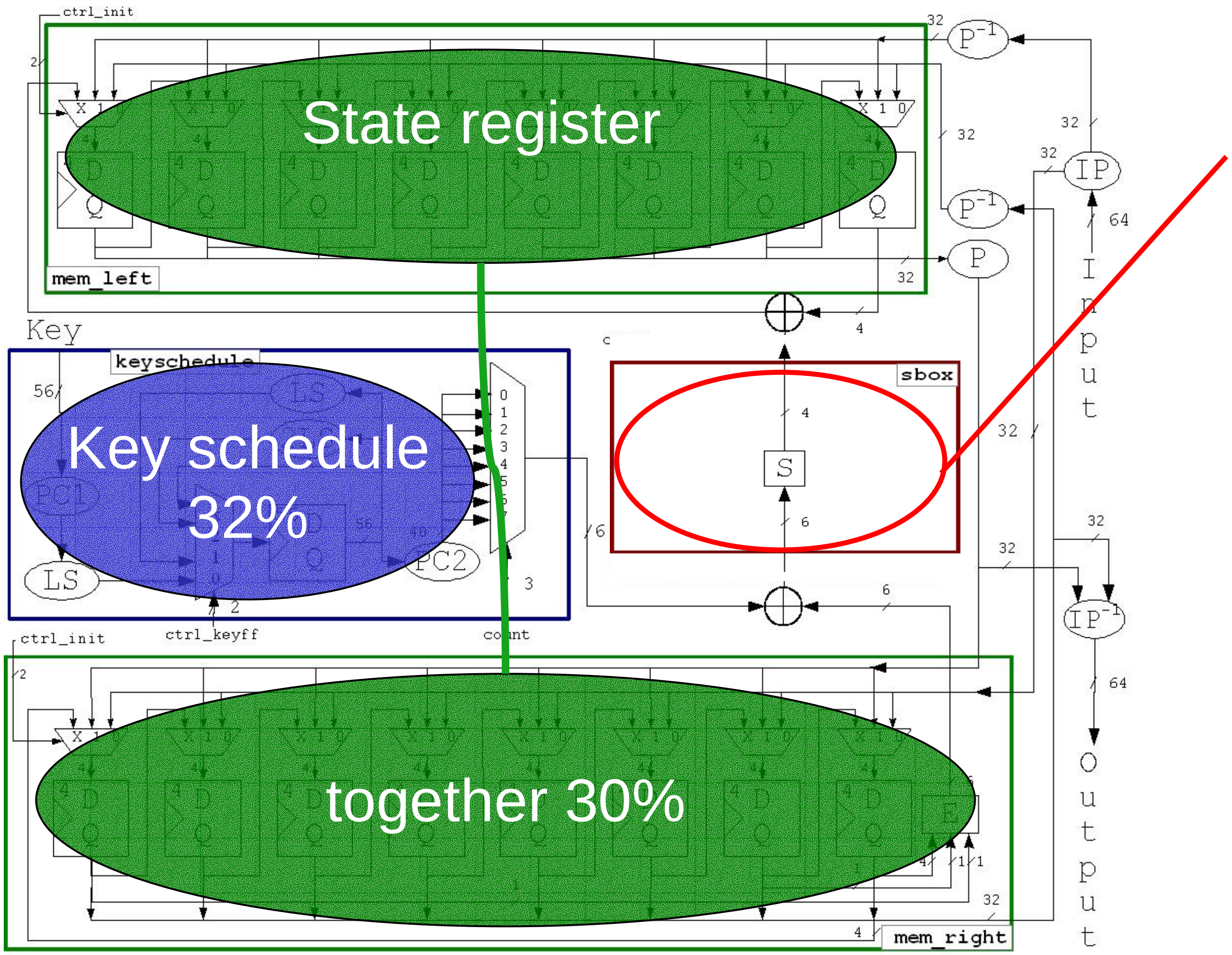
Challenge: Encryption function e() at extremely low cost

- almost all symmetric ciphers optimized with SW in mind
- exception: DES

DES – Data Encryption Standard



Lightweight DES Architecture



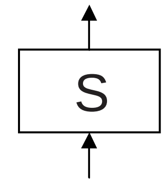
S-Boxes

- 6-to-4 substitution tables
- highly non-linear
  → high Boolean compl.
- **34% of area!**

Idea:

- Replace S1...S8 by S

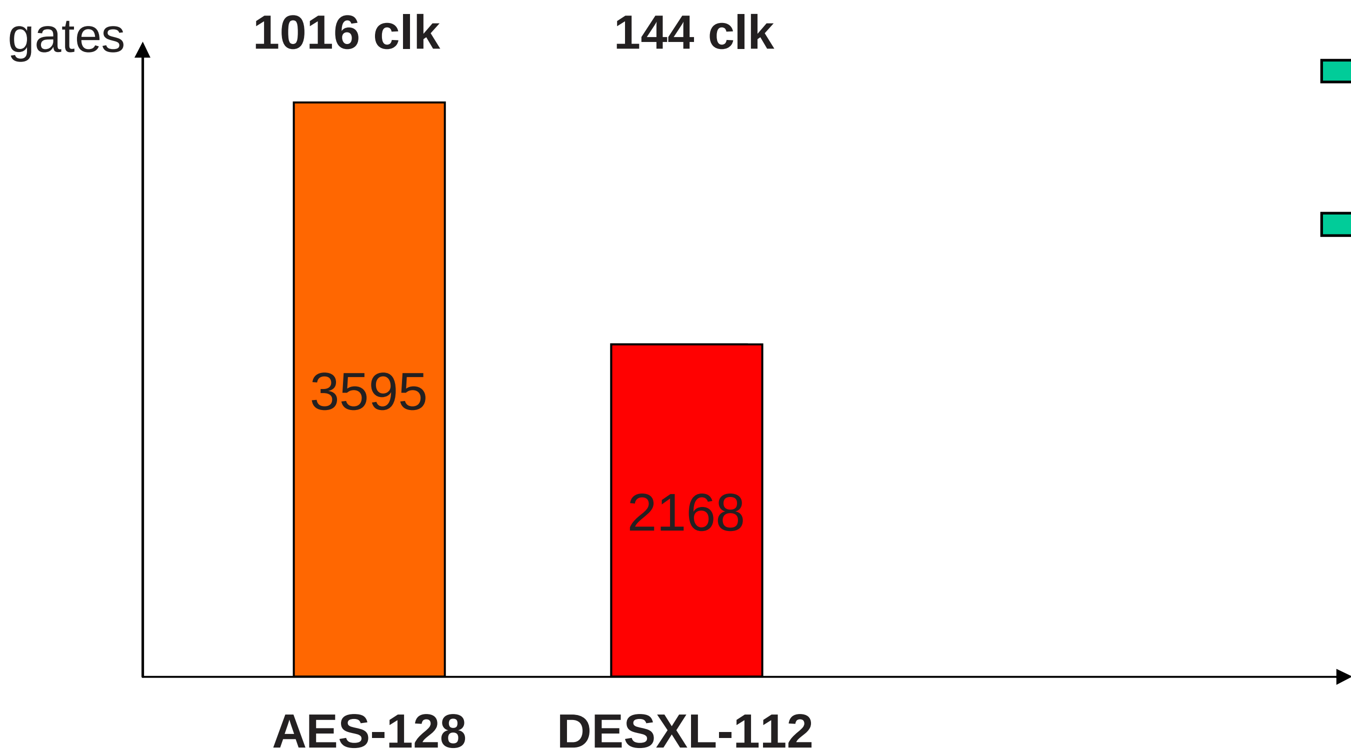
… 12 months later: new Sbox

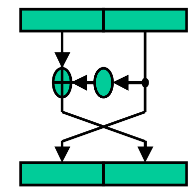


S															
14	5	7	2	11	8	1	15	0	10	9	4	6	13	12	3
5	0	8	15	14	3	2	12	11	7	6	9	13	4	1	10
4	9	2	14	8	7	13	0	10	12	15	1	5	11	3	6
9	6	15	5	3	8	4	11	7	1	12	2	0	14	10	13

- S replaces S1…S8
- S more robust against differential, linear, and David-Murphy attack than S1…S8
- no previous work (!)

Results – Lightweight DES



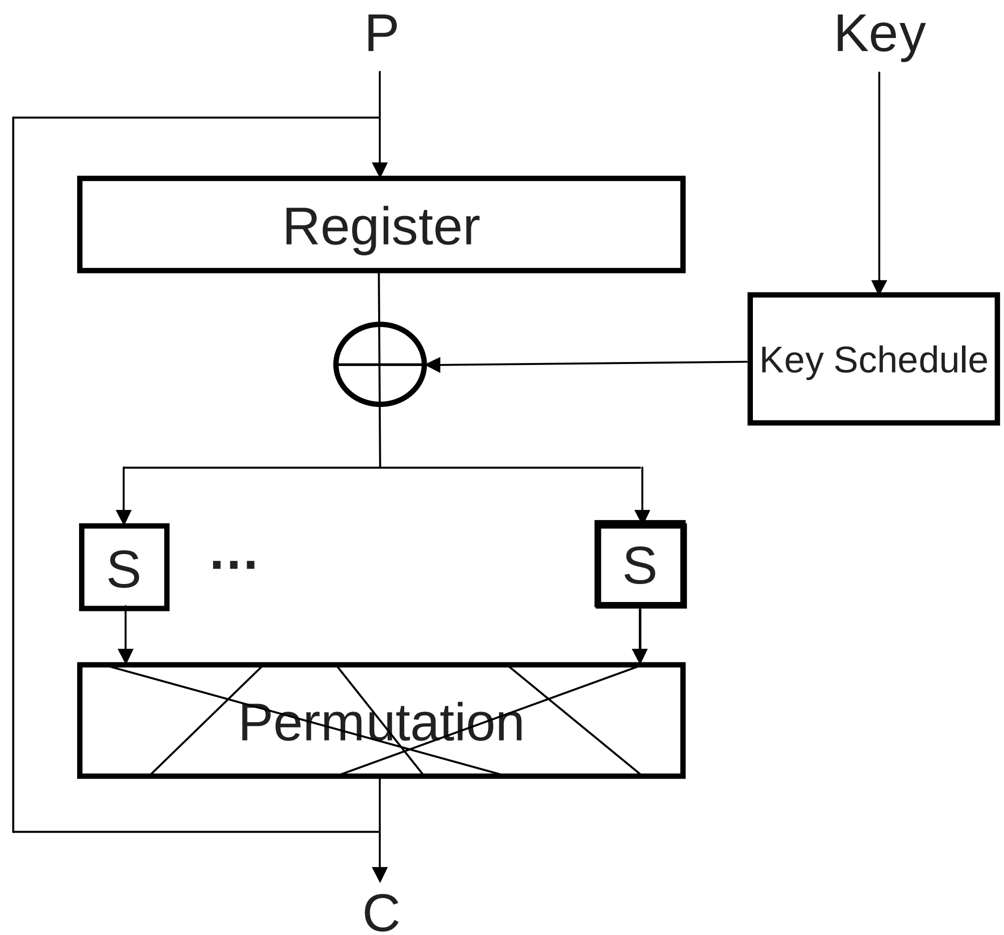


- based on (extremely) well-studied cipher
- TA product 12 times better than smallest AES architecture
- details: FSE '07 paper

Q: Can we do better??

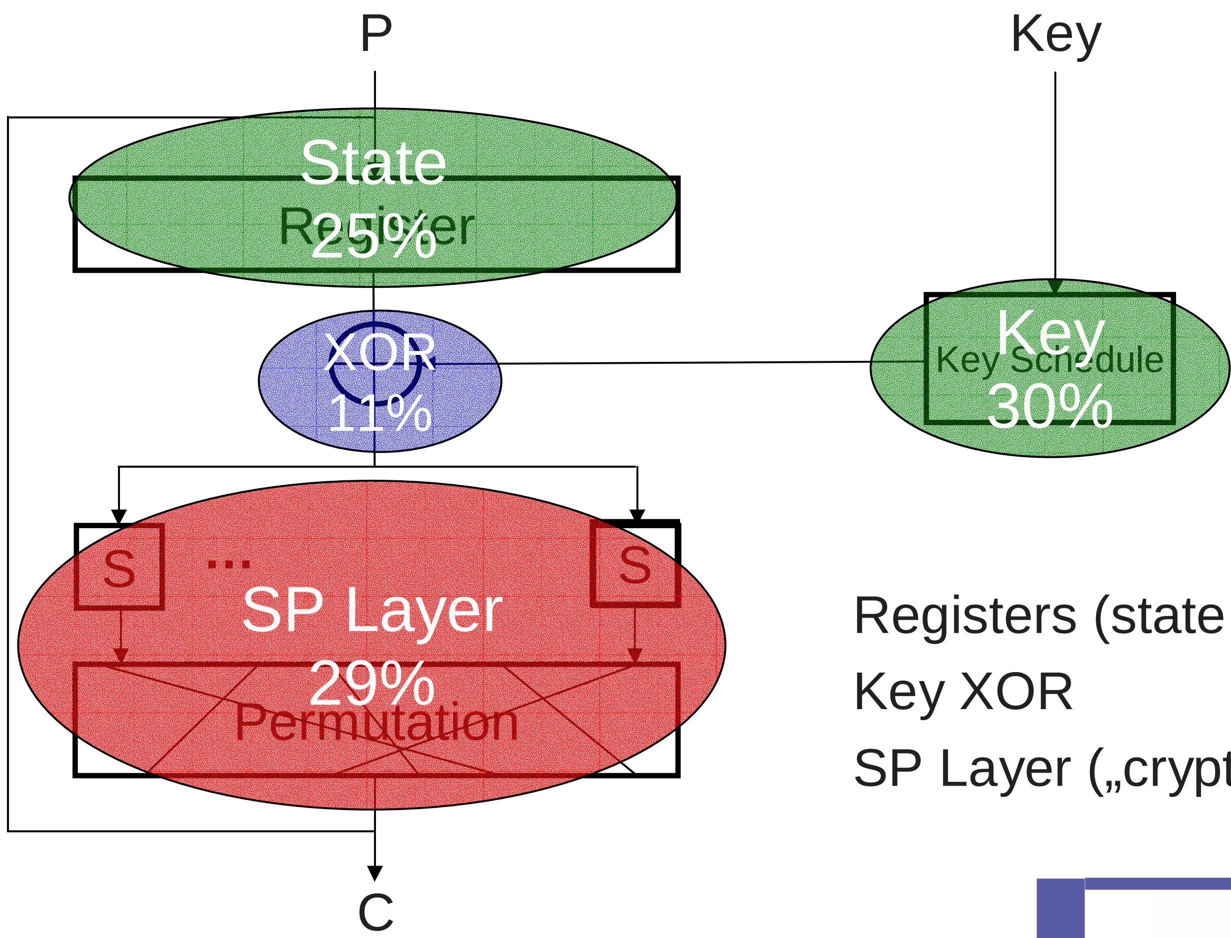
PRESENT – An agressively hardware optimized block cipher for RFID

- pure substitution-permutation network
- 64 bit block, 80/128 bit key
- 4-4 bit Sbox
- 31 round (32 clks)
- „provable secure“ against DC, LC
- joint work with Lars Knudsen, Matt Robshaw et al.



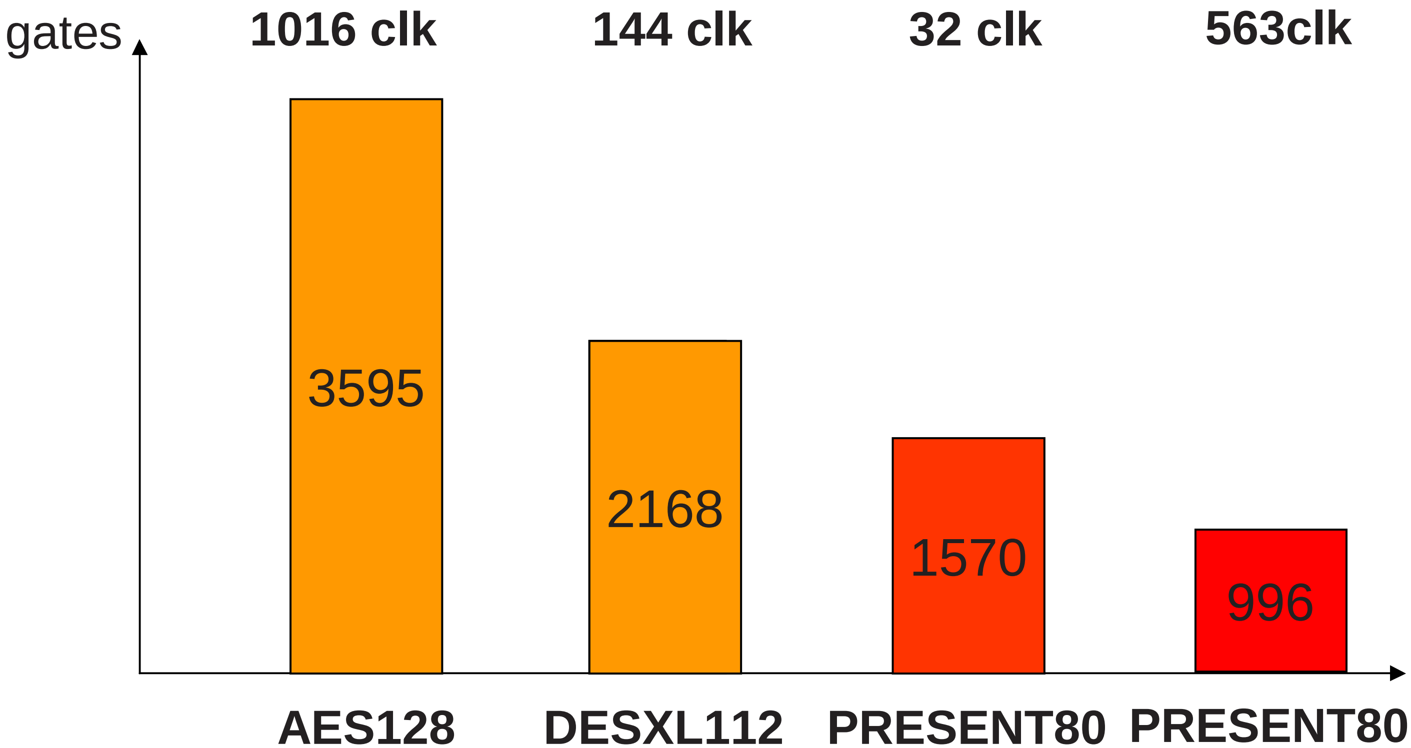
Resource use within lightweight ciphers

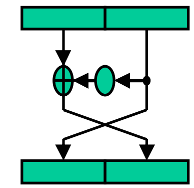
Round-parallel implementation of PRESENT (1570ge)



Registers (state + key)	55%
Key XOR	11%
SP Layer („crypto“)	29%

Results – PRESENT

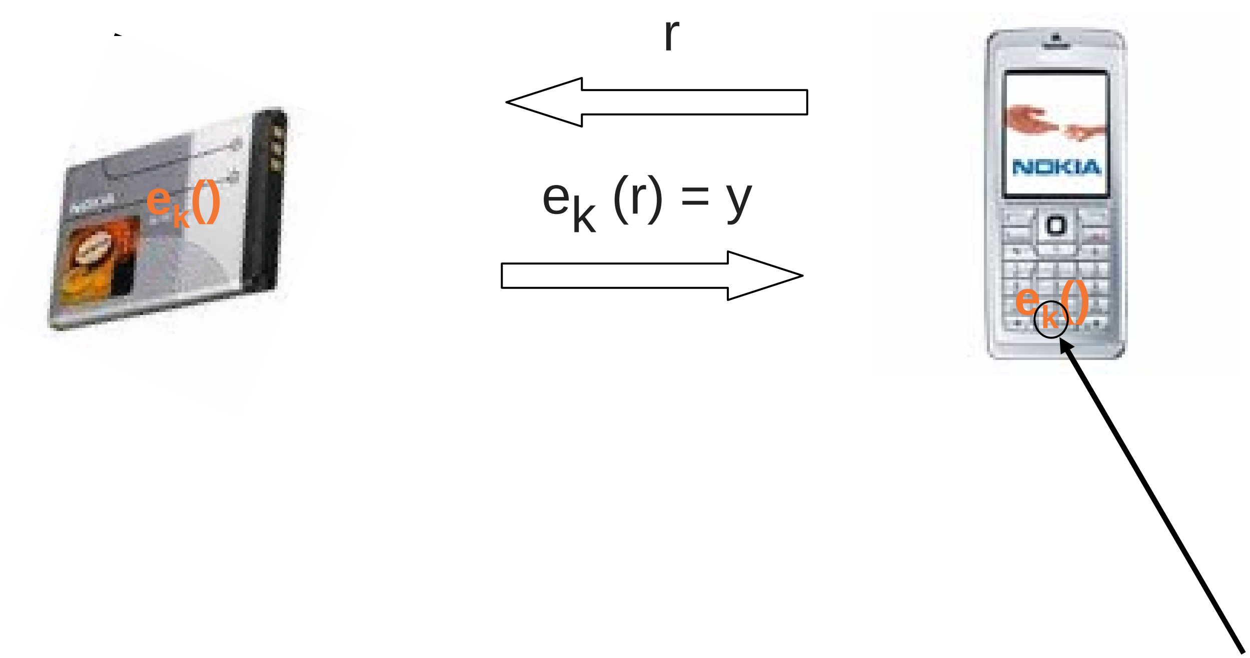




- TA product 1-2 orders of magnitude better than smallest AES architecture
- Serial implementation approaches theoretical complexity limit:
  almost all area is used for the 144 bit state (key + data path)
- smaller than all stream ciphers
- details: CHES ’07 paper

Contents

Strong Identification (w/ symmetric crypto)



Potential weakness: attacker gets access to key on host device (e.g. firmware exploits) and starts cloning batteries

Strong Identification (w/ asymmetric crypto)

r

$sig_{kpr}(r) = y$

1. random challenge r
2. signed response y
3. verification
   $ver_{kpub}(r,y) = t/f$

private key is hard to reverse engineer

Attacker can only access public key from host device

⇒ But how cheap can we build public-key algorithms?

eurobits
European Competence Center for IT Security

Elliptic Curve Primitive

- Given a Point *P* on an elliptic curve E over GF(p):

$$E: \; y^2 = x^3 + ax + b \bmod p$$

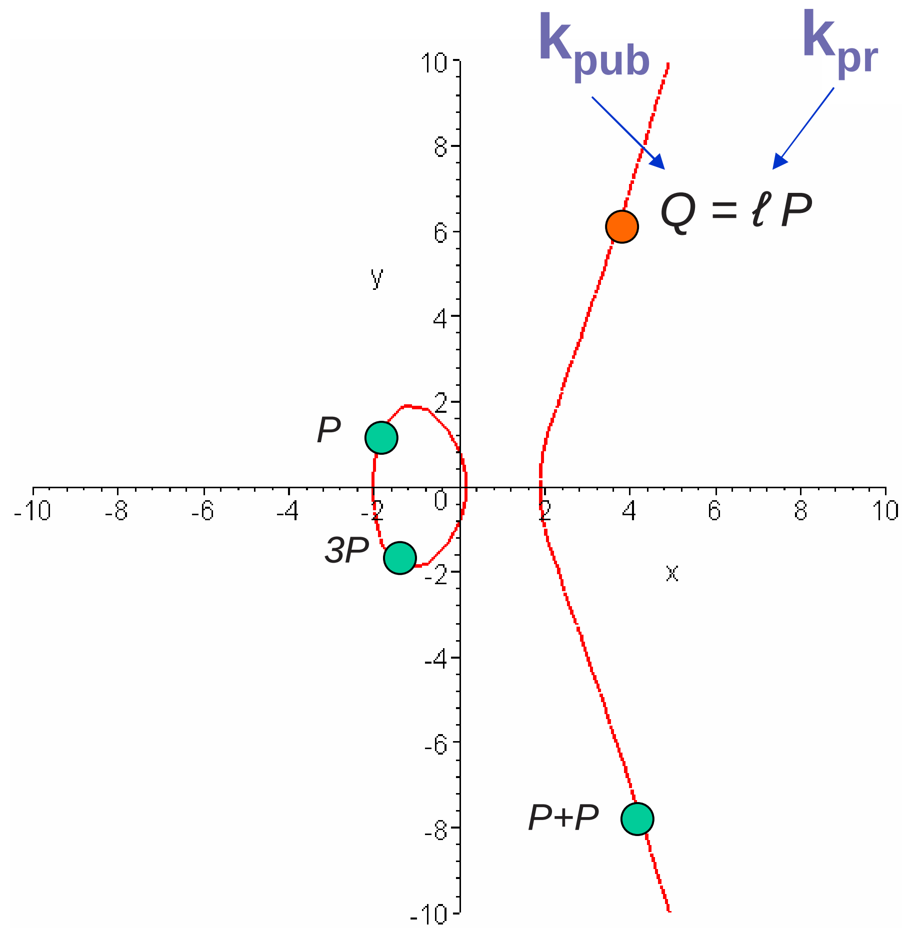
- Public key *Q* is multiple of base point P

group operation

$$Q = P + P + \ldots + P = \ell P$$

- EC discrete logarithm problem:

$$\ell = dlog_P(Q)$$



Design Principles for *Tiny ECC Processor*

- Reduce memory requirements

: memory amounts to more than 50% of design

- Reduce arithemtic unit area

: avoid units like inverter

+ designed for specific size

- Keep it simple but efficient

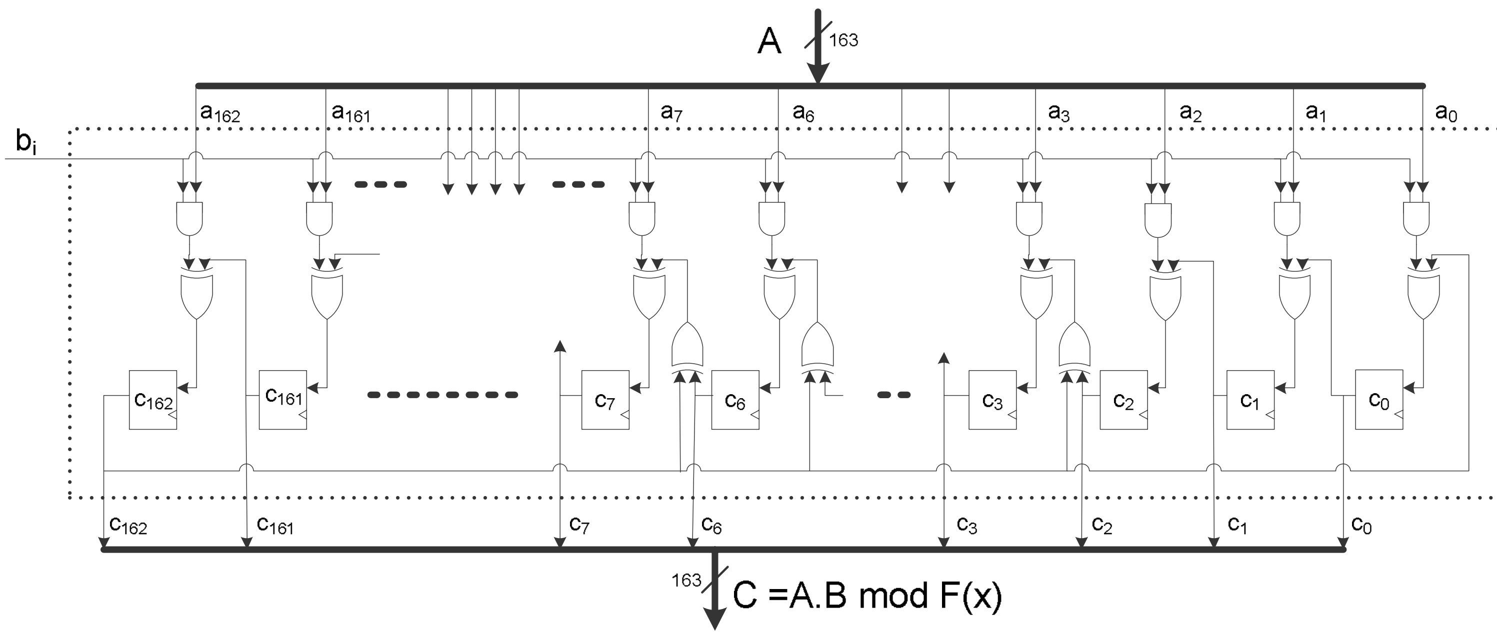
: reduce control logic area - multiplexers

Tiny ECC Processor Units

- Arithmetic Units
  - Multiplier
    - Most-Significant Bit Mult.
  - Squarer
  - inverter
- Point Multiplier
  - Control Unit
- Memory Unit

The Implementation: MSB Multiplier

$C(x)=A(x)\times B(x)=(\quad(A\times b_{m-1}x+A\times b_{m-2})x\quad)x+A\times b_0 \bmod F($

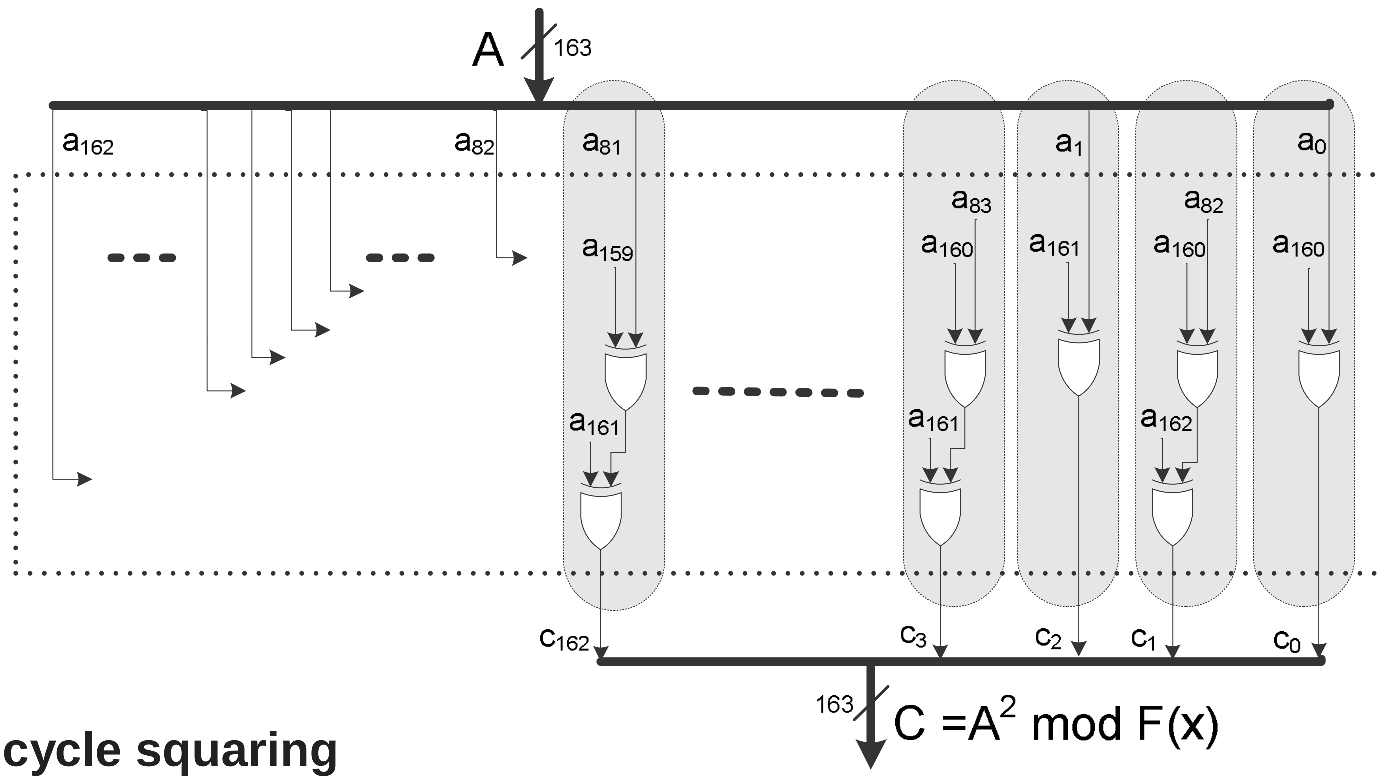


Most-Significant Bit (MSB) Multiplier: **n cycles for n-bit multiplier**

Tiny ECC Processor: Design decisions

- Arithmetic Units
  - Multiplier – Most-Significant Bit Mult.
  - Squarer – Parallel Squaring
  - inverter
- Point Multiplier
  - Control Unit
- Memory Unit

The Implementation: Squarer



- **single cycle squaring**
- low gate count
- low critical path

eurobits European Competence Center for IT Security

Tiny ECC Processor Units

- Arithmetic Units
  - Multiplier
  - Squarer
  - inverter
  - Most-Significant Bit Mult.
  - Parallel Squaring
  - Fermat's Little Theorem
- Point Multiplier
  - Control Unit
- Memory Unit

Inverter – Some basic number theory

Fermat's Little Theorem

$A^{-1} \equiv A^{2^m-2}$ if $A \in GF(2^m)^*$

Straightforward exponentiation: 161 **MUL** + 162 **SQ**

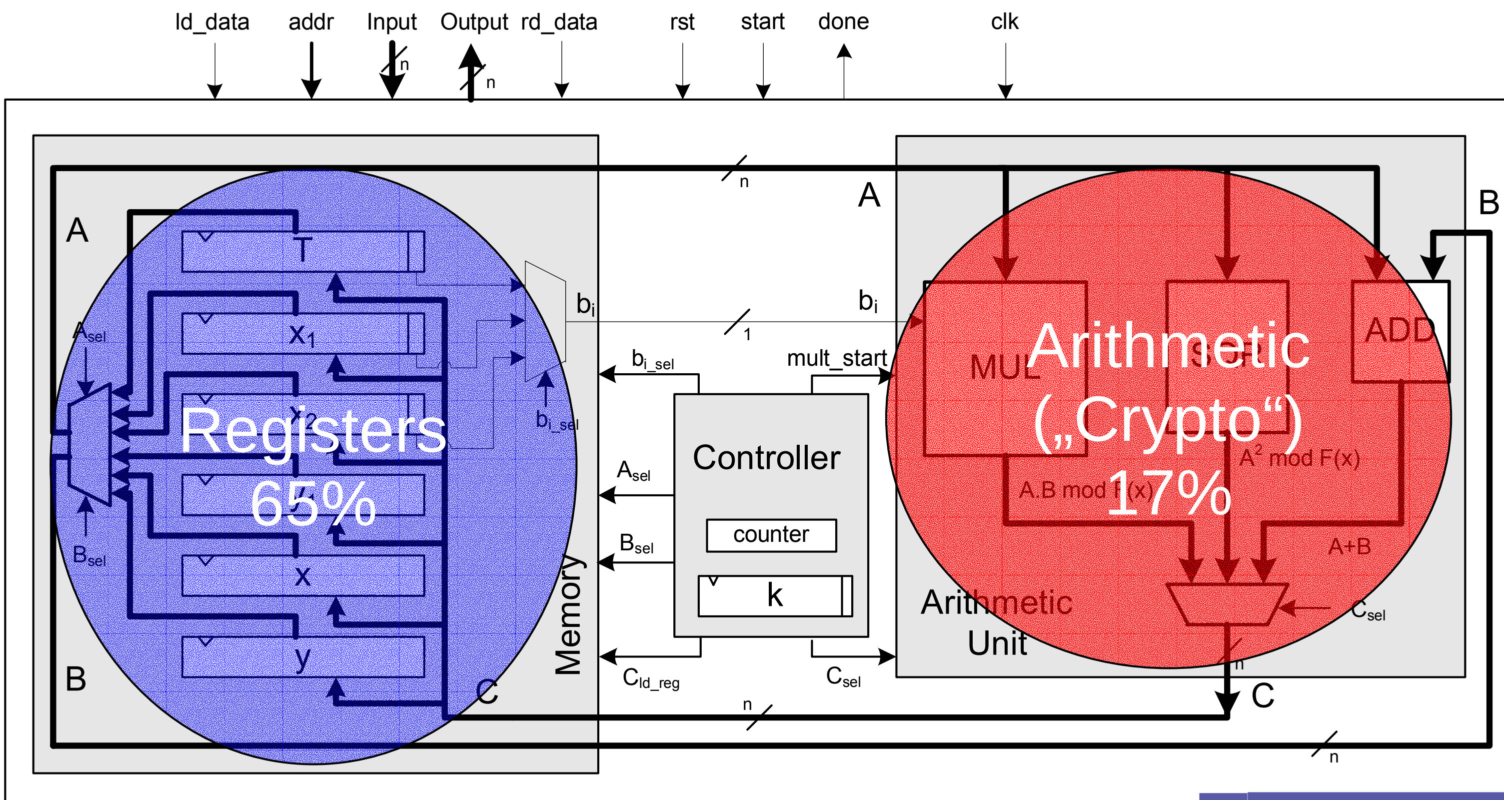
Exploit exponent structure: $A^{2^m-2} = A^{111\ldots110}$ (Itoh-Tsujii)

$\#MUL = \log_2(m-1) + HW(m-1) - 1$
$\#SQ = m-1$

For m=163: 9 **MUL** + 162 **SQ**

The Tiny ECC Processor Design

- ECC processor implementation for $2^{113}, 2^{131}, 2^{163}, 2^{193}$



Performance and Results

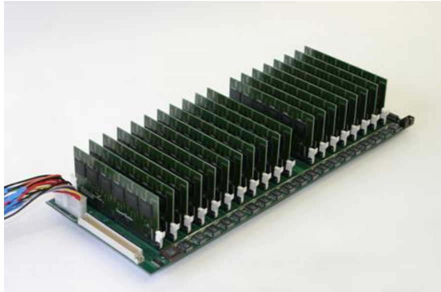
Performance @ 4 MHz for standardized curves

Field Size	**Arithmetic Unit**(gates)	**Memory** (gates)	**Total** (gates)	**Time** (ms)
113	1,625	6,686	**10,112**	**47**
131	2,071	7,747	**11,969**	**61**
163	2,572	9,632	**15,094**	**108**
193	2,776	11,400	**17,723**	**139**

131, 163 bit: very practical bit sizes

Security levels?

Security of mid-size ECC



Costs for breaking ECC in *one year* w/ optimized attack *ASICs*:

ECC131p $\approx$ \$2 million

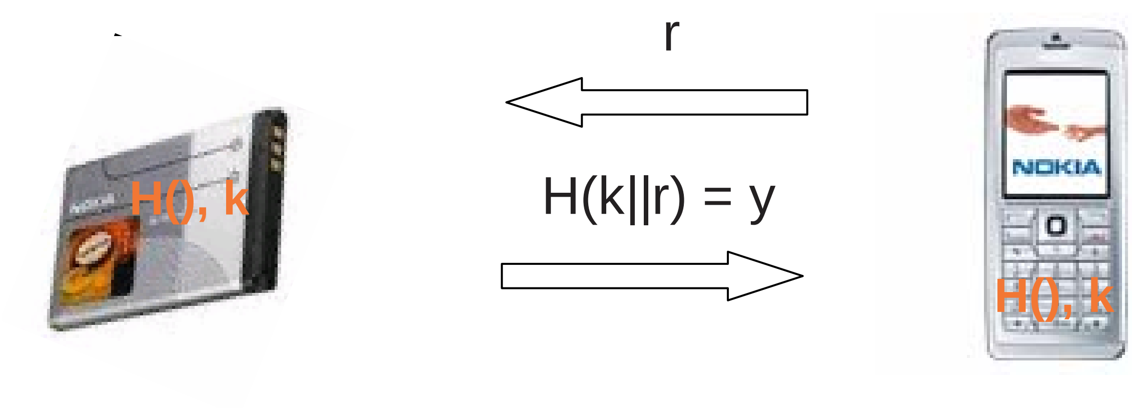
ECC163p: $\approx$ \$1 trillion (> 20 years security)

cf. COPACOBANA @ [CHES06]

Contents

1. Some general thoughts about cheap crypto
2. Lightweight Block Ciphers
3. Lightweight Asymmetric Cryptography
4. **Lightweight Hash Functions**
   (Special thanks to Matt Robshaw)

Hash-based authentication



1. random challenge r
2. encrypted response y
3. verification
   H(k||r) = y‘
   y == y‘

Conventional wisdom:
Hashing is very cheap compared to “real” crypto algorithms
(e.g., popular assumption in ad-hoc network security community)

Lightweight Hash Function

„Best“ results from literature

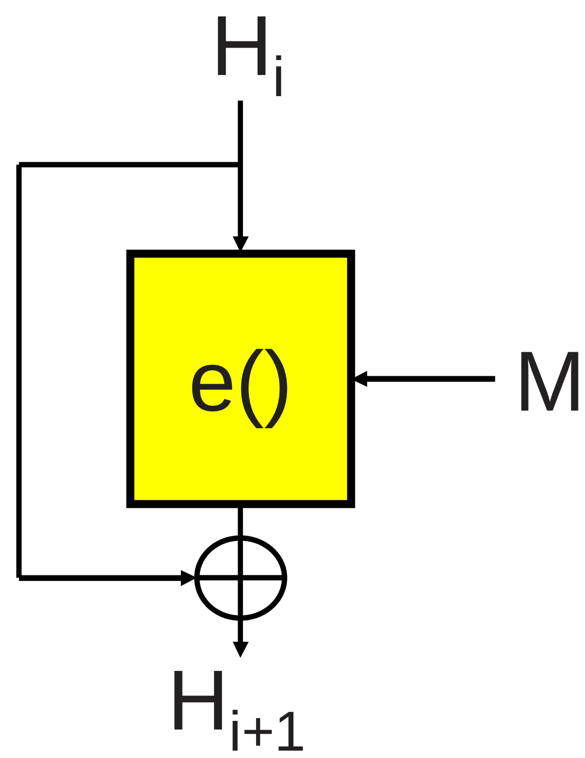
Hash Fct.	Output length	#Clk	Gate equiv.
MD5	128	612	**8,400**
SHA-1	160	1274	**8,120**
SHA-256	256	1128	**10,868**

- hash functions are far worse than block ciphers in hardware
- but we can build hash fct. from block ciphers

Hashfunctions from Block Ciphers (1)

Run cipher in Davies-Meyer mode

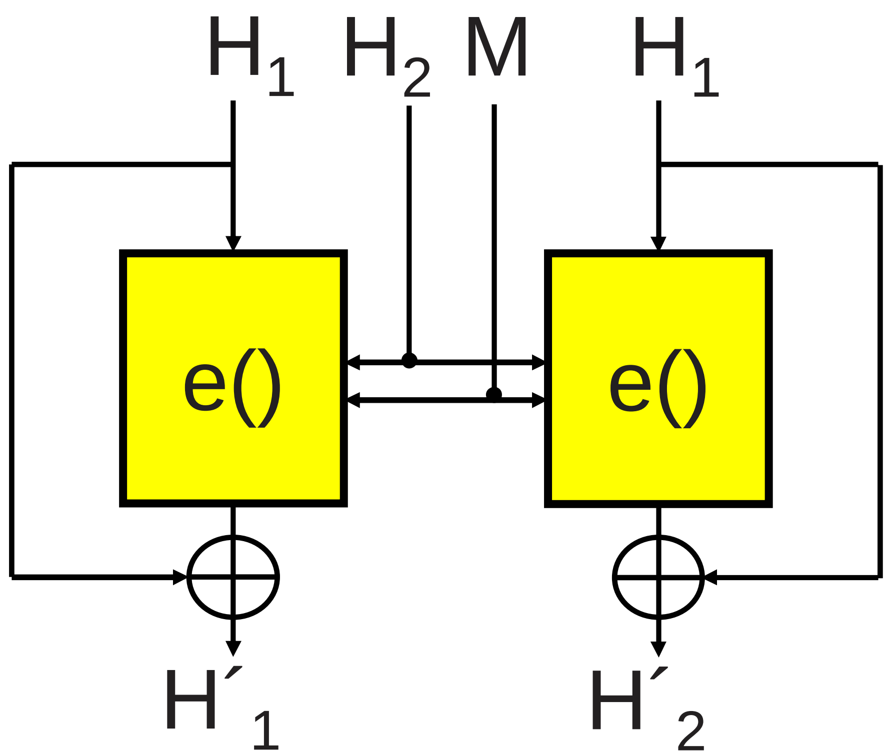
- with AES: ≈ 4000 ge, 1024 clk/block
- drawback: hash size = block size
- Rijndael with 192 or 256 bit block is appealing
- but area increases even more
- DES, PRESENT etc. not suited since 64 bit block



Hashfunctions from Block Ciphers (2)

Double-block length hash
(Hirose construction)

- with PRESENT ≈ 4000 ge, 32 clk/block
- 128 bit hash output
- extension to triple block length possible but many cipher instances needed

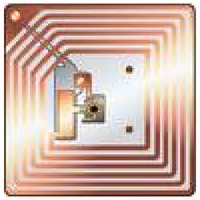


We need dedicated lightweight hash functions!

Some open problems

1. Lighweight hash functions?
2. Lightweight public-key schemes?
3. Lightweight side-channel analysis (SCA) resistance?
4. Interaction lightweight crypto $\leftrightarrow$ SCA resistance?

Related Workshops



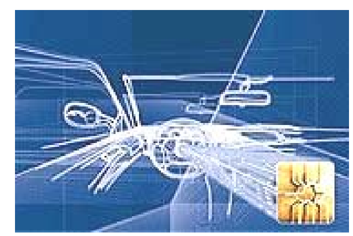
SECSI – Secure Component and Systems Identification
March 2008, Berlin

RFIDSec 2008
July 2008, Budapest

RFIDSec 06

CHES – Cryptographic Hardware and Embedded Systems
August 2008, Washington D.C.

escar – Embedded Security in Cars
November 2008, Hamburg



Further Reading

Individual Ciphers

1. M. Feldhofer, J. Wolkerstorfer, V. Rijmen. *AES Implementation on a Grain of Sand,* Information Security, IEE Proceedings, 152(1):13–20, 2005.
2. G. Leander et al., *New Lightweight DES Variants Suited for RFID Applications*, FSE 2007.
3. A. Bogdanov et al., *PRESENT – A Lightweight Block Cipher for RFID*, CHES 2007.
4. S. Kumar, *Elliptic Curve Cryptography for Constrained Devices*, PhD thesis, ECE Dept., Ruhr University Bochum, 2006.
5. S. Hirose, *Some Plausible Constructions of Double-Block-Length Hash Functions*, FSE 2006.
6. S. Kumar et al., *Breaking Ciphers with COPACOBANA – A Cost-Optimized Parallel Code Breaker,* CHES 2006.

Surveys

7. T. Eisenbarth et al., *A Survey of Lightweight Cryptography Implementations*, IEEE Design and Test, 2007.
8. J.-P. Kaps, G. Gaubatz, B. Sunar, *Cryptography on a Speck of Dust,* IEEE Computer Magazine, 2007.