UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Low-Latency Encryption on Hardware: Trade-offs and Recommendations

Elif Bilge Kavun

September 10, 2023

Tutorial Plan

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Part 1
  - Fundamental concepts
    - When is low latency a need for hardware?
    - How to achieve?
  - Challenges of low latency on hardware implementations
  - Existing low-latency symmetric encryption ciphers
    - PRINCE and PRINCEv2 (2012 and 2020)
    - Midori (2015)
    - MANTIS (2016)
    - QARMA (2017*)
    - Kcipher (2020)
    - SPEEDY (2021)
    - SCARF (2022*)
    - LLWBC (2022)
    - Sonic and SuperSonic (2023)
    - ASCON low-latency performance?
  - Physical attack resistant variants

Tutorial Plan

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Part 2
  - Implementation (tool information next slides)
    - Sbox performance for low latency
      - ASIC vs. FPGA
    - PRINCE and PRINCEv2
      - Simulation
      - Synthesis
      - Unrolled vs. round-based performance (FPGA, ASIC)
    - Codes and info: https://github.com/sec-int/lowlatency_tutorial

UNIVERSITÄT PASSAU

Fakultät für Informatik und Mathematik

ModelSim Environment

ModelSim Environment

- You can download ModelSim Intel FPGA edition from
  https://cdrdv2.intel.com/v1/dl/downloadStart/660921/660958?filename=ModelSimSetup-20.1.1.720-windows.exe
  https://cdrdv2.intel.com/v1/dl/getContent/661030/661059?filename=ModelSimSetup-20.1.0.711-linux.run

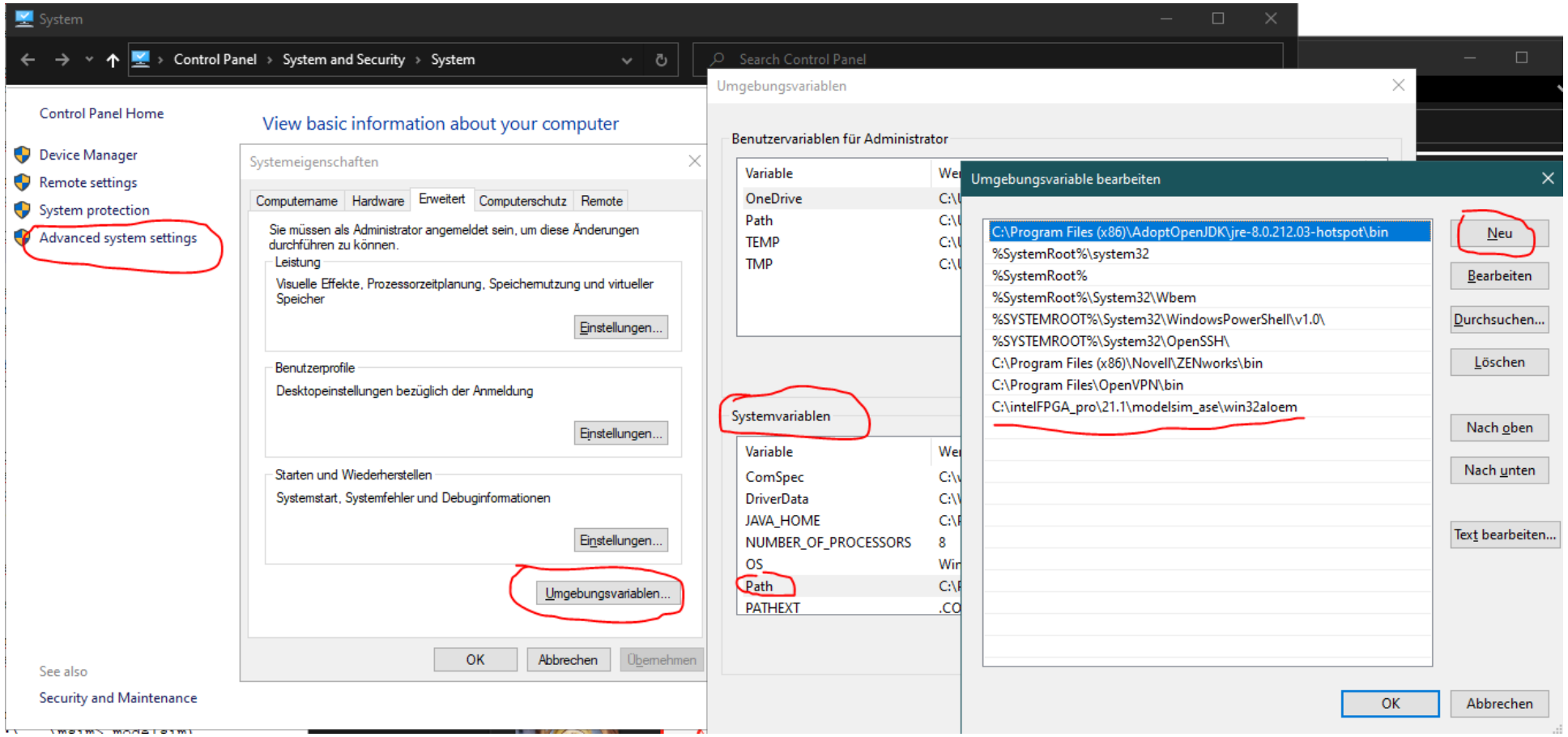
- After the download, install “ModelSimSetup-20.1.1.720-windows.exe” (for Windows) or “ModelSimSetup-20.1.1.720-linux.run” (for Linux)

- Installation on Linux requires running these commands in your directory:

  > chmod +x ModelSimSetup-20.1.1.720-linux.run
  > ./ModelSimSetup-20.1.1.720-linux.run

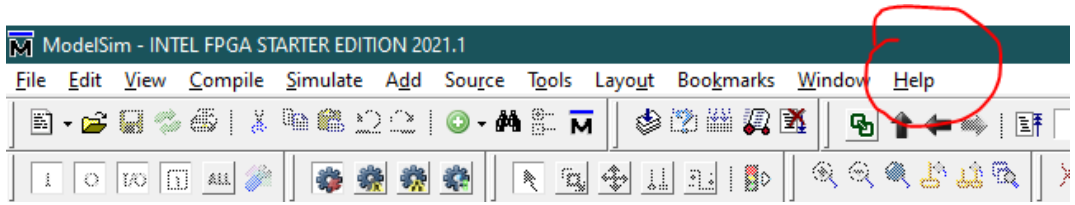
ModelSim Environment

- Make sure that ModelSim executable directory is added to your DOS path (an option may be given to you during the installation)



ModelSim Environment

- Once you have installed the program, you can start ModelSim graphical interface and access all documents under Help
  - You are strongly adviced to "at least" read the Tutorial and Manuals under Help/PDF Documentation



- You can create a project or directly use ModelSim via "do" files (scripts) as given in the example implementations provided after the lab

- For each of your homeworks/projects create the following directory structure:

proj_dir → **vlog** (directory for your Verilog-HDL sources)

→ **msim** (directory for your ModelSim files)

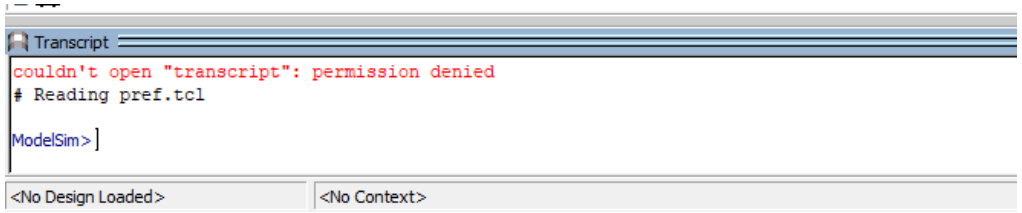
ModelSim Environment

Under **vlog**, place your Verilog-HDL sources. Under **msim**, copy the *sim.do* and *wave.do* files given with the examples
Normally, if you name your testbench top module (in the ***<design>_tb.v*** file) as test, you can use the *sim.do* file as is. Otherwise, edit the *sim.do* file and correct the name of the testbench top module to the name you picked. You will probably need to edit *wave.do* file anyway, since this is where you specify the input, output and internal nets you want to see on the waveform viewer

```
1  vlib work
2  vmap work work
3  vlog -work work ../vlog/*.v
4  vsim -voptargs="+acc" -t ns work.test
5  do wave.do
6  run -all
```

→ example “sim.do” file

Either start ModelSim by clicking on the Windows icon and then from ModelSim command editor (located at the lower part of the GUI) change to your **msim** directory (e.g., cd C:/Users/username/Desktop/tutorial/low_latency/msim)
Note that you can enter most Unix-type commands from the ModelSim command editor

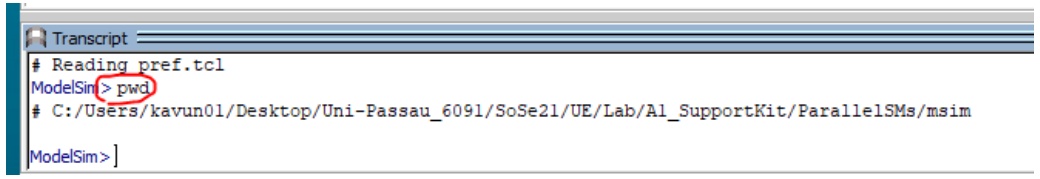


Or open a DOS command window. Change to your msim directory (cd c:\...\msim) and start ModelSim by entering the “modelsim" command from the command window (e.g. c:\...\msim> modelsim)

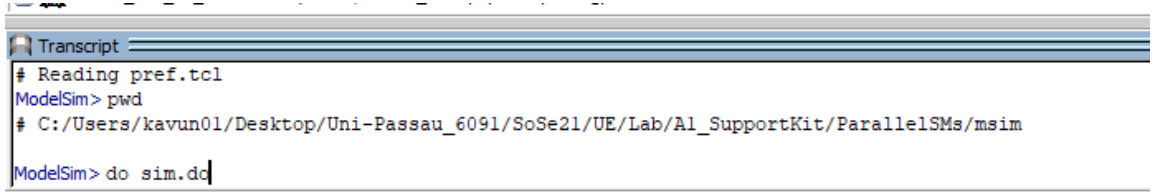
```
C:\Users\kavun01\Desktop\Uni-Passau_6091\SoSe21\UE\Lab\A1_SupportKit\ParallelSMs>modelsim
```

ModelSim Environment

Make sure you are in the correct directory, by typing "pwd" in the ModelSim command editor



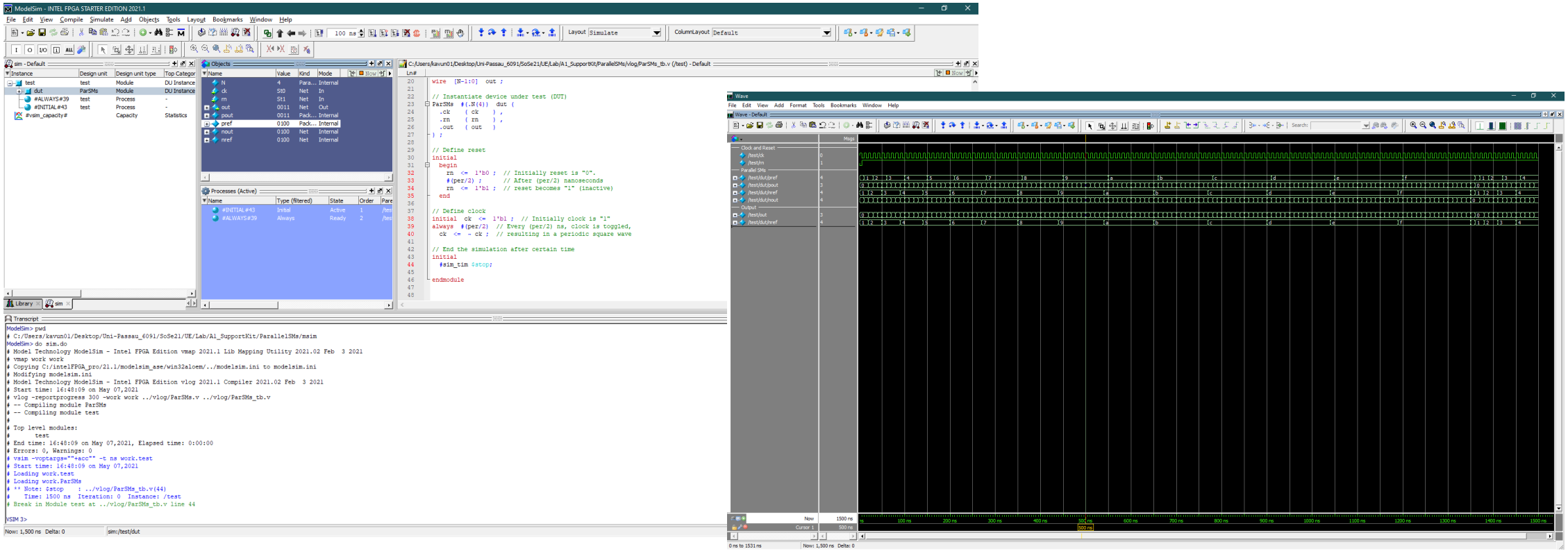
From the ModelSim command editor, type "do sim.do". This will compile your source files, run simulation, open waveform viewer (you can pop the window out later if it does not happen directly, it will stay popped out after first run: option named “Dock/Undock”) and plot all requested waveforms



In case of an execution error, carefully inspect the error/warning messages. Correct your HDL files. You can work on a separate code editor program (Notepad++, etc.) for convenience. However, ModelSim may ask you to confirm that the source file(s) is/are updated, in case any source file was open in the simulation tool window (i.e., this would probably happen if you modify the testbench file as it opens in ModelSim tool by default when you start the program). You should confirm these changes. After that, execute "do sim.do" again

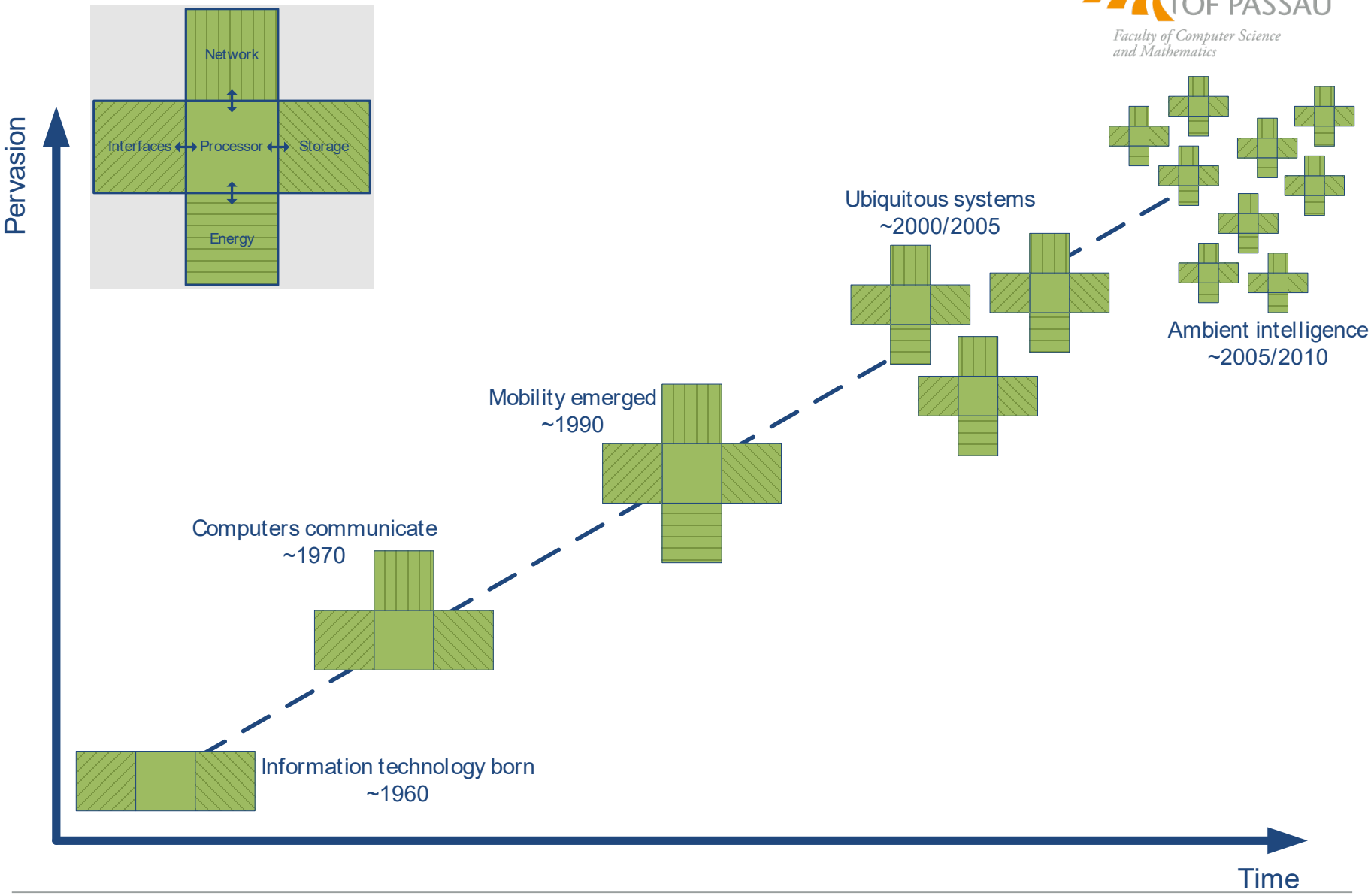
ModelSim Environment

Other options are also available in ModelSim simulation tool, such as adding waves to your current simulation using the tool (i.e., modifying the wave.do file is not the only option for this) and running your simulation from scratch or step-by-step using the corresponding buttons in the tool. You can read the tutorial to learn these



If you want to switch to another directory to run a different design simulation, you need to quit current one by typing "quit -sim". You can then change to the corresponding **msim** directory as explained above

Ubiquitous Computing Era



Ubiquitous Computing Era

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Electronic passports

Smartphones
Mobile applications

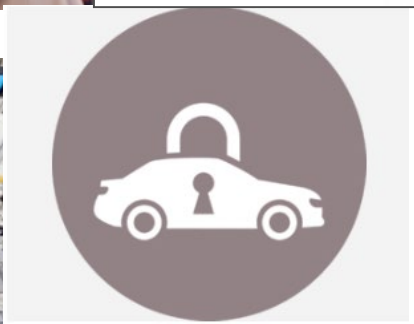
Smart home

Medical & sensor systems

Payment & toll-collection cards

Products, IPs

Automation components
Asset tracking systems



Car key systems

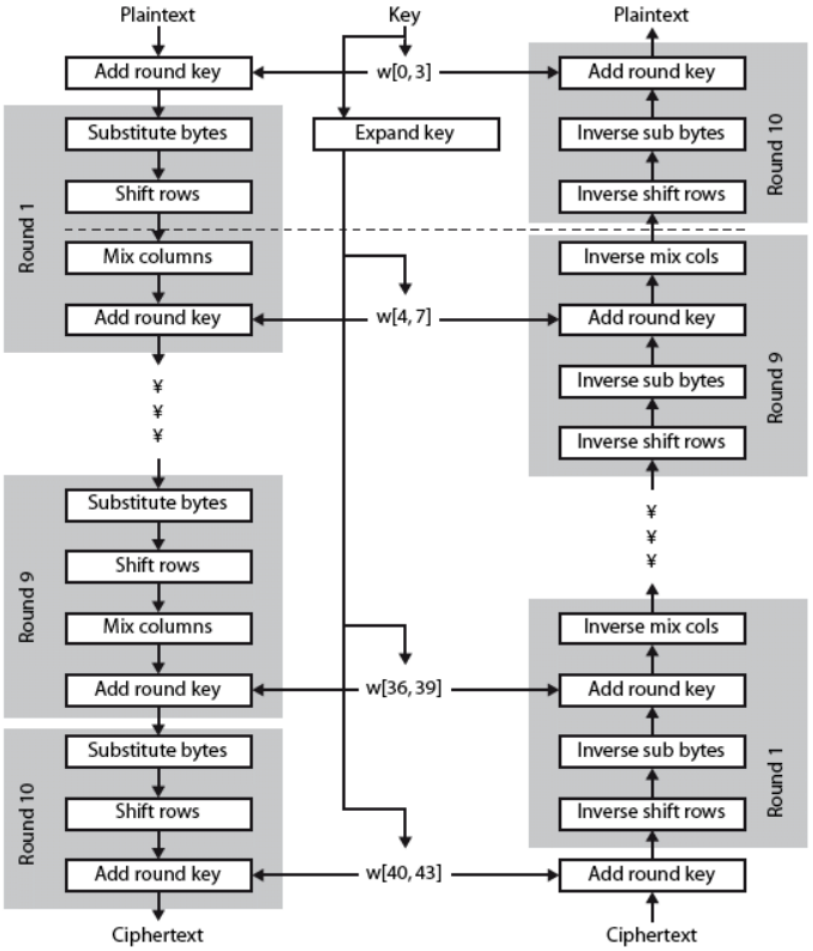
Security Concerns!!!

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Access control
- Data confidentiality
- Security
- Counterfeiting mitigations

Good security designs and architecture needed to resist attacks!

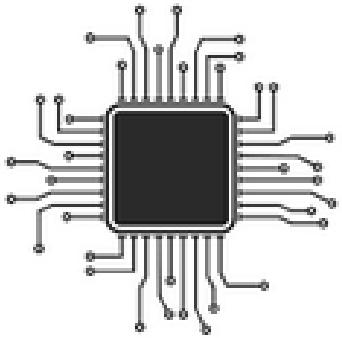
Need for Security: Same Level?



- **Conventional cryptography**
  - RSA
  - Standard block cipher solutions (AES, etc.)
- **Applications in**
  - Servers
  - PCs
  - “Strong” tablets, smartphones

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Embedded/IoT devices ➔ **Resource-constrained!**



Chip Area: Limited!

Power and Energy Consumption: Lowest Possible!

➔ to match these constraints:

Need for *Tailored* Cryptography: *Lightweight* Cryptography

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Reduces computational efforts to provide security
  - Cheaper than traditional crypto
  - Not weak, but “sufficient„ security
- Many different proposals/implementations especially in the last decade
  - Public-key cryptography: ECC
  - Stream ciphers: Grain, Trivium, etc.
  - Hash functions: Photon, Quark, etc.
  - **Block ciphers**
    - Core for symmetric cryptography, stream ciphers, MACs, etc.

Initial Proposals

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

ISO/IEC 29192-2:2019
Information security — Lightweight cryptography — Part 2: Block ciphers

PRESENT: An Ultra-Lightweight Block Cipher

A. Bogdanov[1], L.R. Knudsen[2], G. Leander[1], C. Paar[1], A. Poschmann[1], M.J.B. Robshaw[3], Y. Seurin[3], and C. Vikkelsoe[2]

[1] Horst-Görtz-Institute for IT-Security, Ruhr-University Bochum, Germany
[2] Technical University Denmark, DK-2800 Kgs. Lyngby, Denmark
[3] France Telecom R&D, Issy les Moulineaux, France
leander@rub.de, {abogdanov,cpaar,poschmann}@crypto.rub.de
lars@ramkilde.com, chv@mat.dtu.dk
{matt.robshaw,yannick.seurin}@orange-ftgroup.com

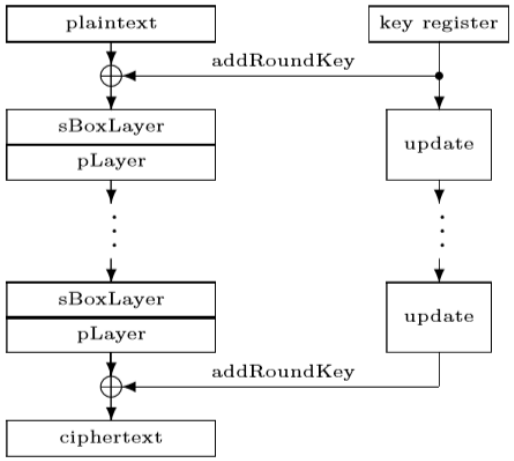
Abstract. With the establishment of the AES the need for new block ciphers has been greatly diminished; for almost all block cipher applications the AES is an excellent and preferred choice. However, despite recent implementation advances, the AES is not suitable for extremely constrained environments such as RFID tags and sensor networks. In this paper we describe an ultra-lightweight block cipher, PRESENT. Both security and hardware efficiency have been equally important during the design of the cipher and at 1570 GE, the hardware requirements for PRESENT are competitive with today's leading compact stream ciphers.

- Simple but strong design
  - Well-studied substitution-permutation network (SPN)
- Targeting hardware
- Low-area
  - Permutation is just wiring in hardware!

Initial Proposals

ISO/IEC 29192-2:2019
Information security — Lightweight cryptography — Part 2: Block ciphers

```
generateRoundKeys()
for i = 1 to 31 do
    addRoundKey(STATE, K_i)
    sBoxLayer(STATE)
    pLayer(STATE)
end for
addRoundKey(STATE, K_32)
```



- Simple but strong design
  - Well-studied substitution-permutation network (SPN)
- Targeting hardware
- Low-area
  - Permutation is just wiring in hardware!

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

KLEIN: A New Family of Lightweight Block Ciphers

Zheng Gong[1], Svetla Nikova[2,3] and Yee Wei Law[4]

[1] School of Computer Science, South China Normal University, China
cis.gong@gmail.com
[2] Faculty of EWI, University of Twente, The Netherlands
[3] Dept. ESAT/SCD-COSIC, Katholieke Universiteit Leuven, Belgium
s.i.nikova@utwente.nl
[4] Department of EEE, The University of Melbourne, Australia
yee.wei.law@gmail.com

Abstract. Resource-efficient cryptographic primitives are essential for realizing both security and efficiency in embedded systems like RFID tags and sensor nodes. Among those primitives, lightweight block cipher plays a major role as a building block for security protocols. In this paper, we describe a new family of lightweight block ciphers named KLEIN, which is designed for resource-constrained devices such as wireless sensors and RFID tags. Compared to related proposals, KLEIN has advantage in the software performance on legacy sensor platforms, while its hardware implementation can be compact as well.

- AES-like
- Works on nibbles
- Involution Sbox

Initial Proposals

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

The LED Block Cipher*

Jian Guo[1], Thomas Peyrin[2,†], Axel Poschmann[2,†], and Matt Robshaw[3,‡]

[1] Institute for Infocomm Research, Singapore
[2] Nanyang Technological University, Singapore
[3] Applied Cryptography Group, Orange Labs, France
{ntu.guo,thomas.peyrin}@gmail.com
aposchmann@ntu.edu.sg
matt.robshaw@orange.com

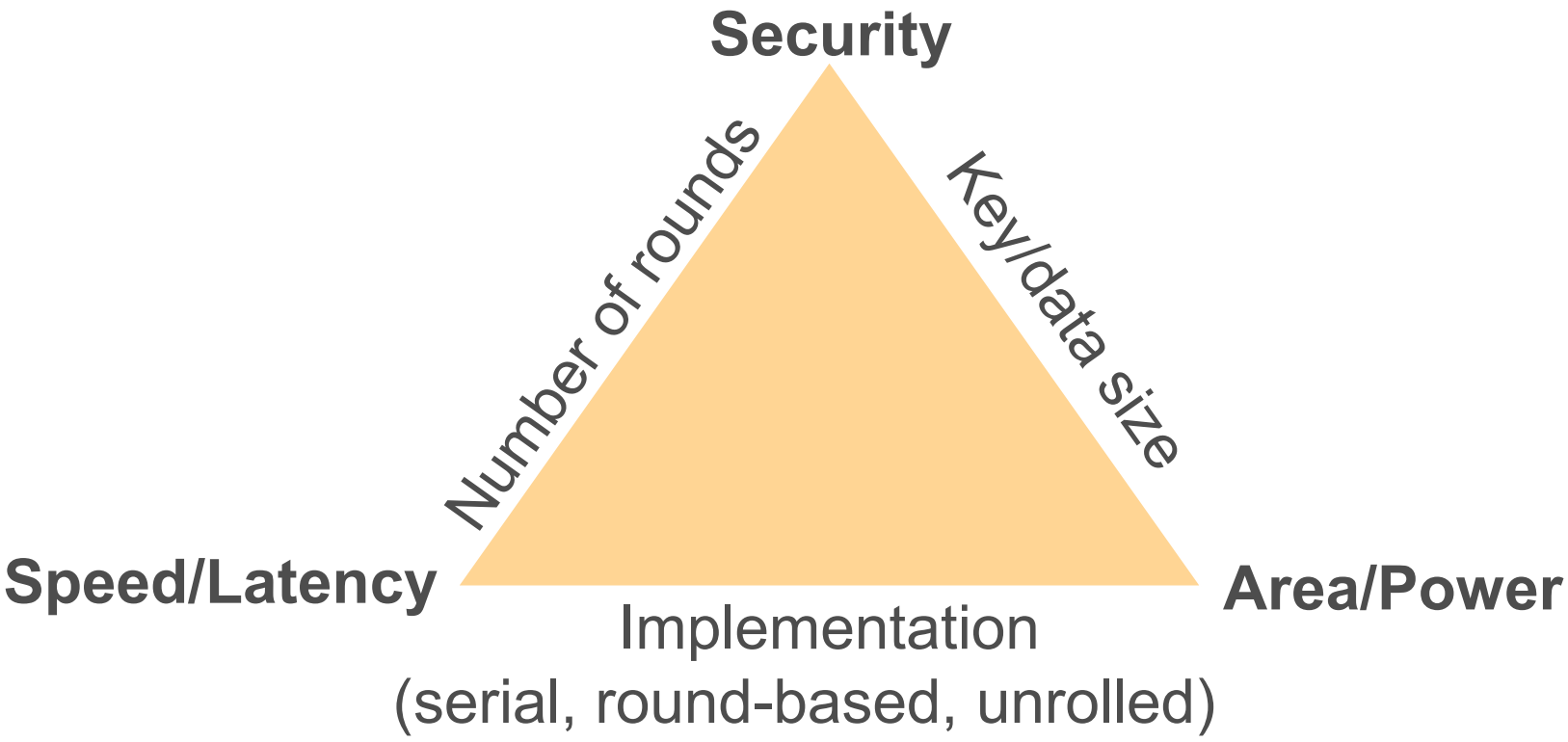
Abstract. We present a new block cipher LED. While dedicated to compact hardware implementation, and offering the smallest silicon footprint among comparable block ciphers, the cipher has been designed to simultaneously tackle three additional goals. First, we explore the role of an ultra-light (in fact non-existent) key schedule. Second, we consider the resistance of ciphers, and LED in particular, to related-key attacks: we are able to derive simple yet interesting AES-like security proofs for LED regarding related- or single-key attacks. And third, while we provide a block cipher that is very compact in hardware, we aim to maintain a reasonable performance profile for software implementation.

- AES-like
- Uses PRESENT Sbox
- Consists of steps
  - Number based on key size
  - Each step 4 rounds

Proposals vs. Metrics

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Initial proposals mostly address area in hardware / speed in software
- Other important metrics?



Hardware Metrics (*)

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Area
  - Usually measured in µm2, but depends on technology and the standard cell library
  - Hence stated in gate equivalents (GE) independent of the technology and library
  - One GE is equivalent to the area required to implement the two-input NAND gate (area derived by dividing the area in µm2 by the area of a two-input NAND gate)
- Cycles
  - # of clock cycles required to compute and output the results
- Time
  - Required time for a certain operation, i. e., # of cycles divided by operating frequency

* Shahram Rasoolzadeh, Hardware-Oriented SPN Block Ciphers, PhD Thesis, RUB, 2020

Hardware Metrics (*)

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Throughput
  - Bit rate production of a new output w.r.t., i. e., # of output bits divided by time (expressed in bits per second – bps –)
- Power
  - Usually the power consumption estimated on the gate level by the synthesizer tool (typically in µW)
  - Power estimations on transistor level are more accurate (more steps in design flow)
- Energy
  - Power consumption over a certain time period, i. e., multiplicating the power consumption with the required time (typically in µJ)

* Shahram Rasoolzadeh, Hardware-Oriented SPN Block Ciphers, PhD Thesis, RUB, 2020

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Unrolled
  - Whole encryption or decryption process is computed within only one clock cycle without using any registers in combinatorial circuit
  - Low-latency

- Pipelined
  - Circuit for whole encryption or decryption process is implemented (similar to unrolled), some registers are inserted in the critical path (path with maximum delay) to increase
  - Higher throughput rate but with the cost of higher area and power consumption

* Shahram Rasoolzadeh, Hardware-Oriented SPN Block Ciphers, PhD Thesis, RUB, 2020

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Round-based
  - Each round function of the cipher is computed within one clock cycle
  - Reduces area and power at cost of decreasing throughput

- Serialized
  - Each round function computed in several clock cycles, and in each clock cycle, a small part of the round function is computed (e. g., only one S-box, or only one word of the linear layer)
  - Lower area & power consumption, but also lowest throughput
  - After a point, implementing control logic may require more overhead than before

* Shahram Rasoolzadeh, Hardware-Oriented SPN Block Ciphers, PhD Thesis, RUB, 2020

Proposals vs. Metrics

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

	Hardware	Software
Area/ Code size	mCrypton, LBlock, HIGHT, KLEIN, KATAN, CLEFIA, TWINE, Piccolo, KTANTAN, LED, PRESENT, SIMON	ITUBee, KLEIN, SEA, SPECK
Latency/ Execution time	?	LBlock, TWINE, KLEIN, SPECK

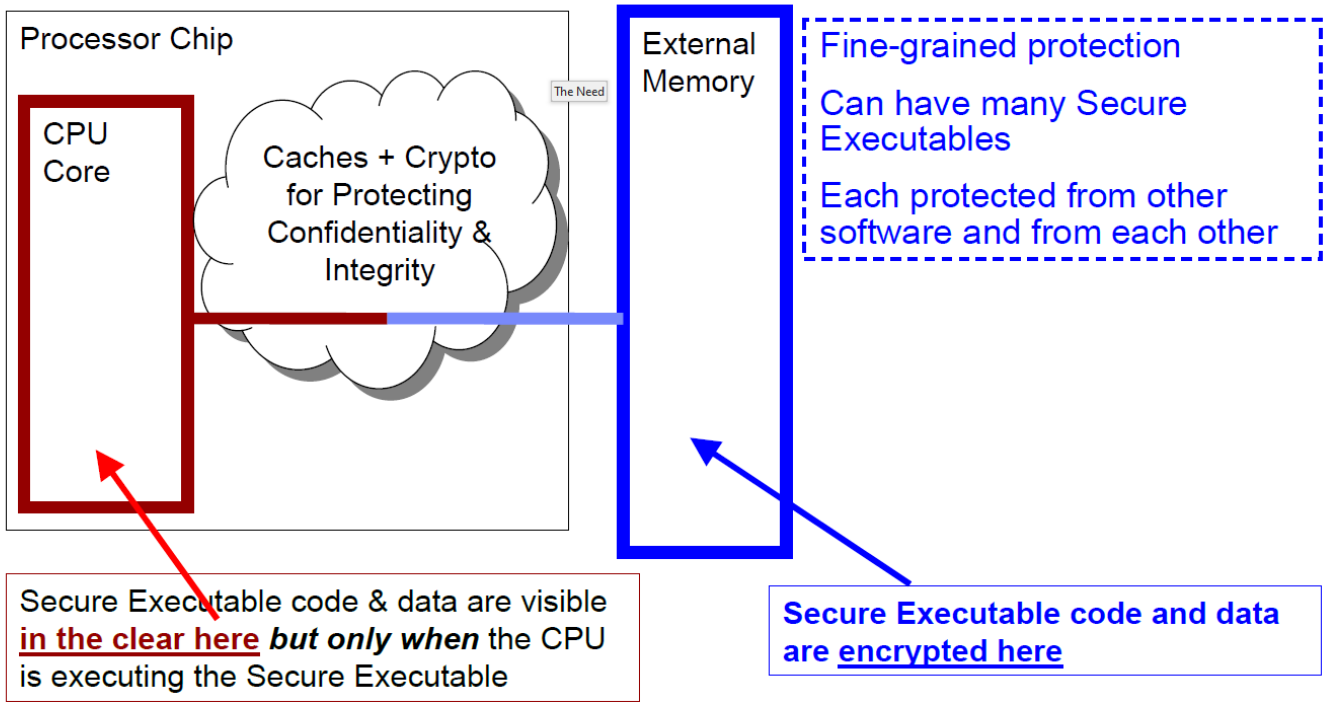
UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

IBM

NSA TRUSTED COMPUTING Conference & Exposition

New Twist on SecureBlue: SecureBlue++

- Like SecureBlue, but provides more fine-grained SecureBlue-like crypto protection to protect information in one program from other S/W (including OS)
  - Protect confidentiality & integrity of information so other S/W cannot read it or undetectably tamper with it

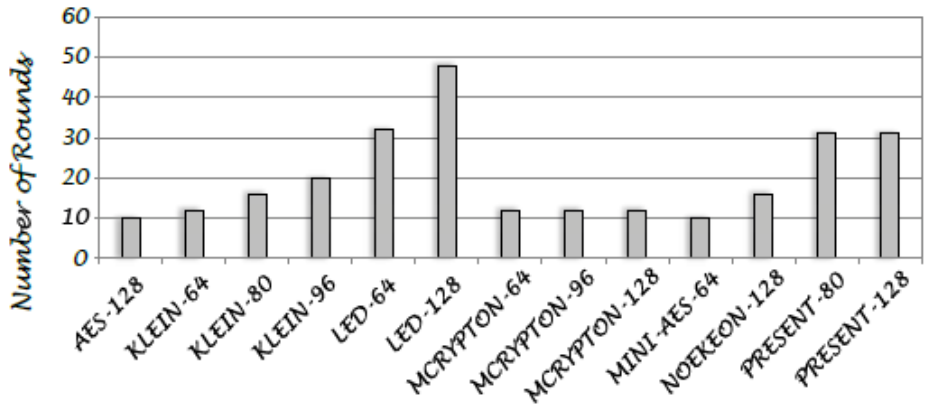


10

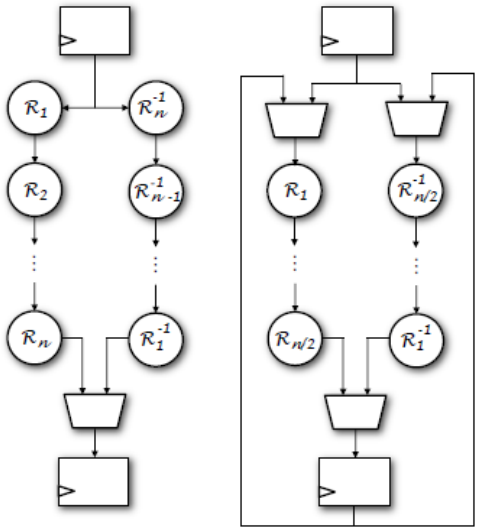
Also,

- Intel SGX
- AMD SEV

Low-latency in Focus

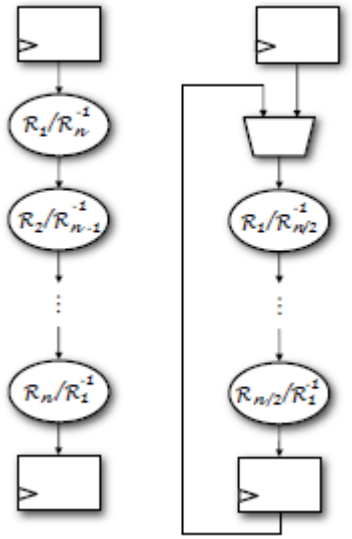


Unrolled



Enc/Dec

1-cycle 2 cycles



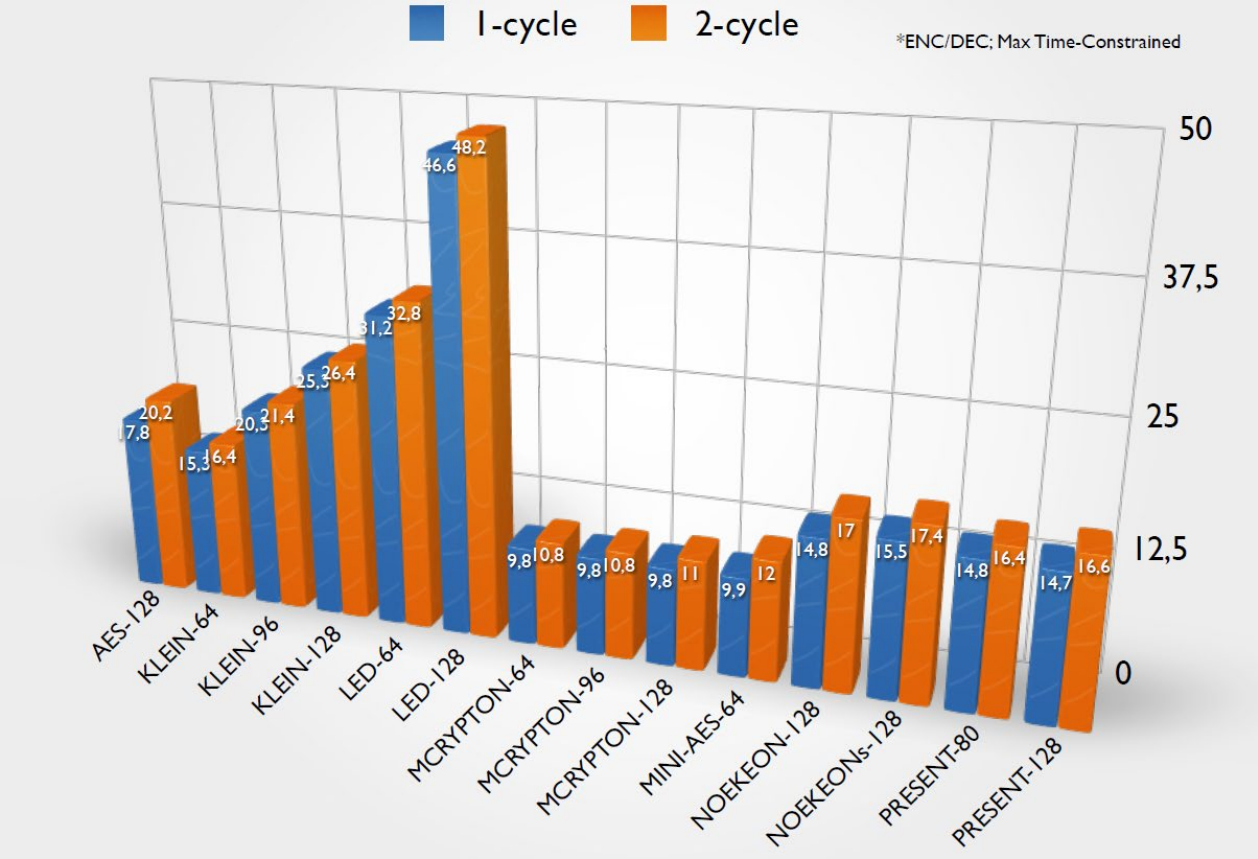
Enc/Dec
Shared Datapath

1-cycle 2 cycles

* Knezevic et al., Low-Latency Encryption – Is “Lightweight = Light + Wait”?, CHES, 2012

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

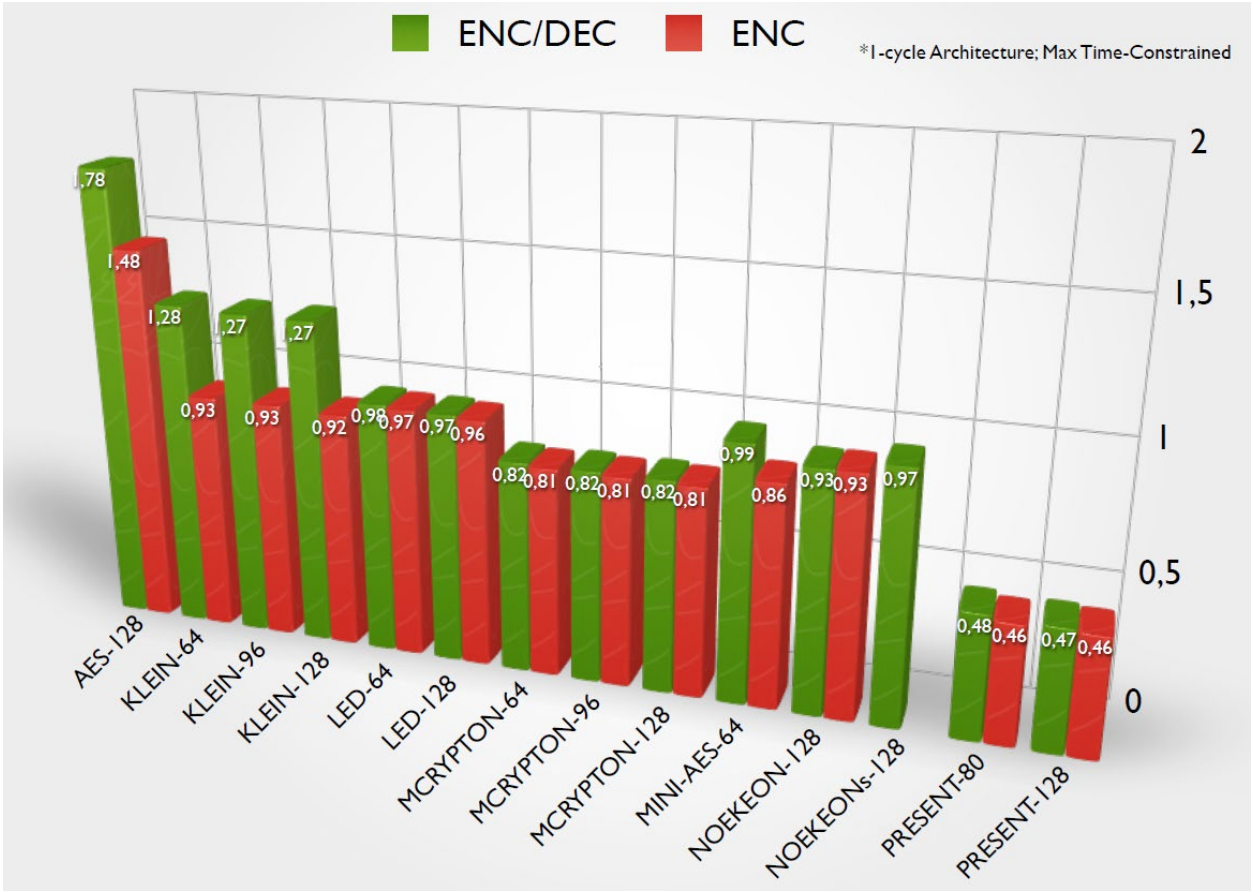
Latency: Time to encrypt one block of data (ns)



* Knezevic et al., Low-Latency Encryption – Is “Lightweight = Light + Wait”?, CHES, 2012

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

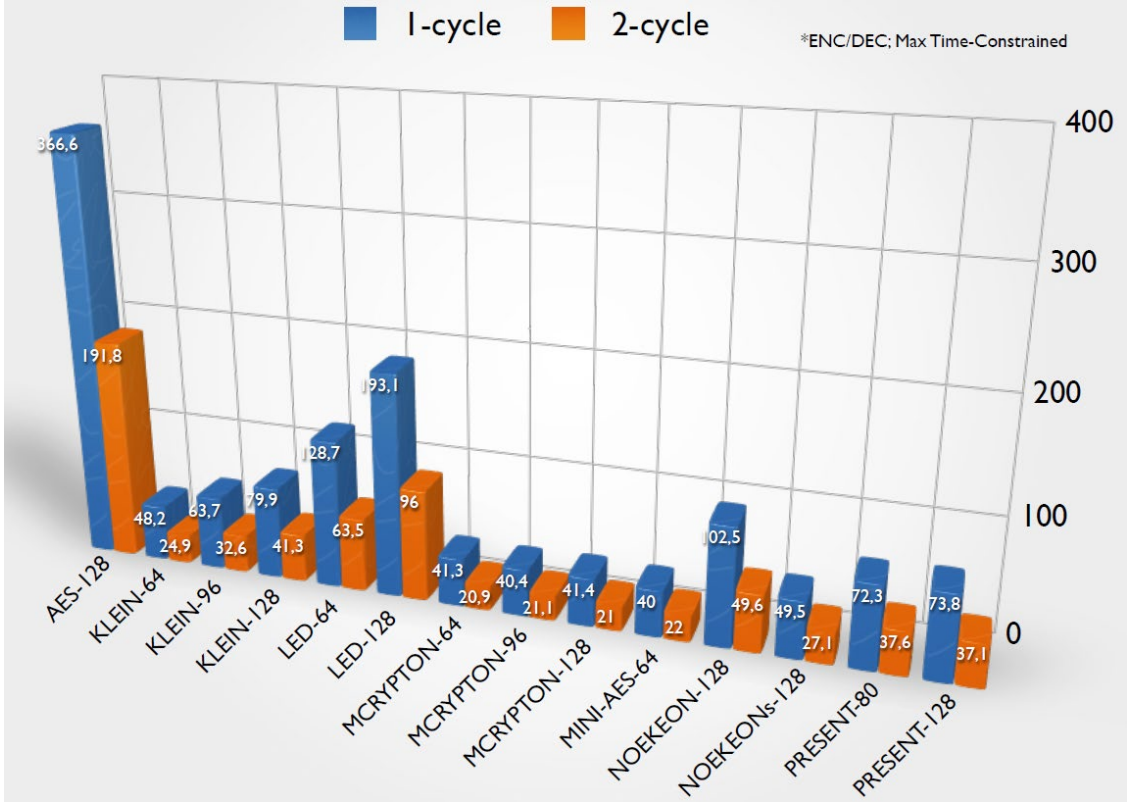
Latency: Time to encrypt one block of data (ns) → per round



* Knezevic et al., Low-Latency Encryption – Is “Lightweight = Light + Wait”?, CHES, 2012

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Latency: Time to encrypt one block of data (Corresponding area cost in GE)

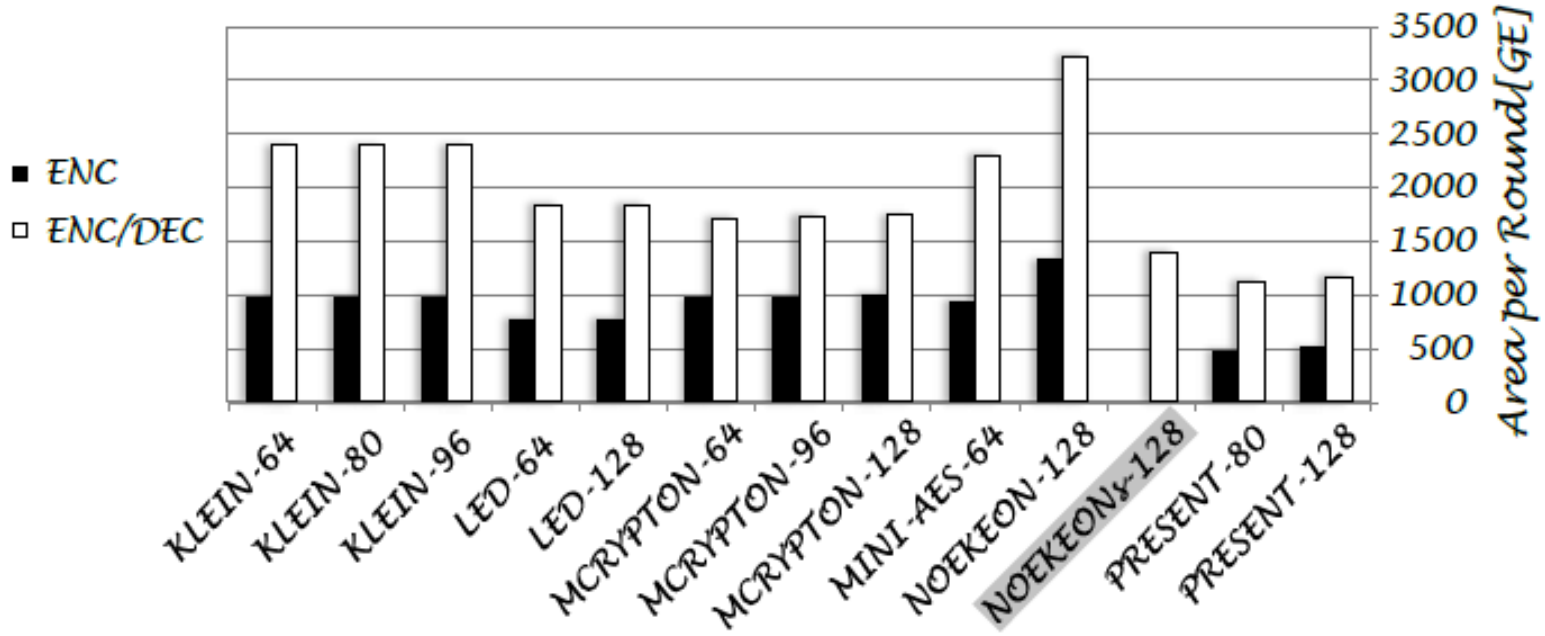


Less area possible for encryption of one block of data?

* Knezevic et al., Low-Latency Encryption – Is “Lightweight = Light + Wait”?, CHES, 2012

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Latency: Time to encrypt one block of data
(Corresponding area cost in GE → per round)



Less area possible for encryption of one block of data?

* Knezevic et al., Low-Latency Encryption – Is “Lightweight = Light + Wait”?, CHES, 2012

Low-latency in Focus: What is Important? (1/2)

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Keep the hardware cost of one round as low as possible
  - Main savings in Sbox, smaller (3/4 bit Sboxes better)
  - Even among these there are significant differences
- All rounds are unrolled
  - Cipher can be thought as one big round
  - Number of rounds hence is important, should be minimized
- All rounds same, decreases cost
  - Less round complexity as well based on components, not too low

Low-latency in Focus: What is Important? (2/2)

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Slightly heavy round with less/balanced number of rounds
- Simpler key schedule
  - Should be independent of number of rounds
  - Constant addition instead of key schedule should be preferred, if possible
- Minimum overhead for encryption and decryption
  - Use involution

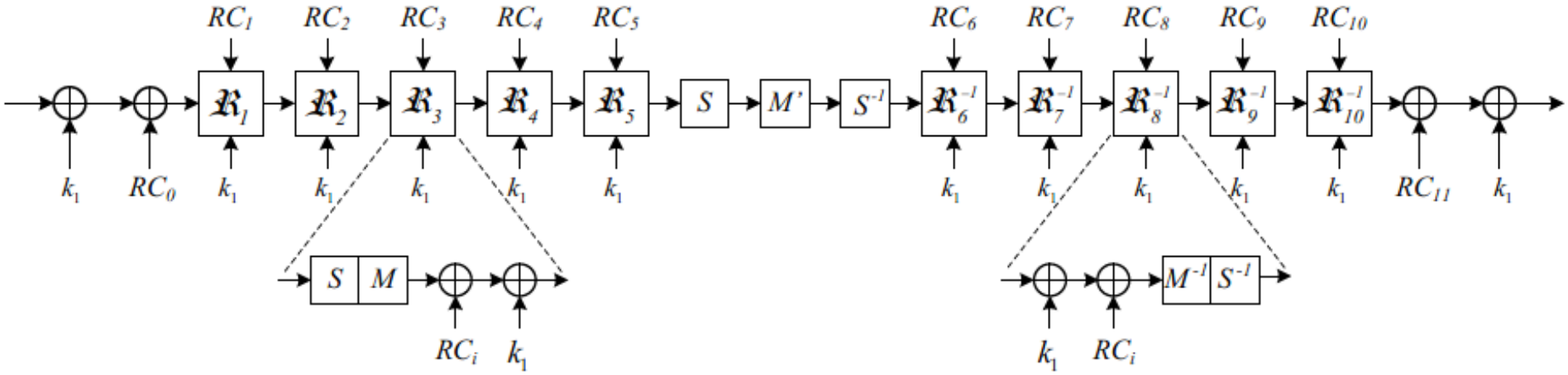
Proposals vs. Metrics

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

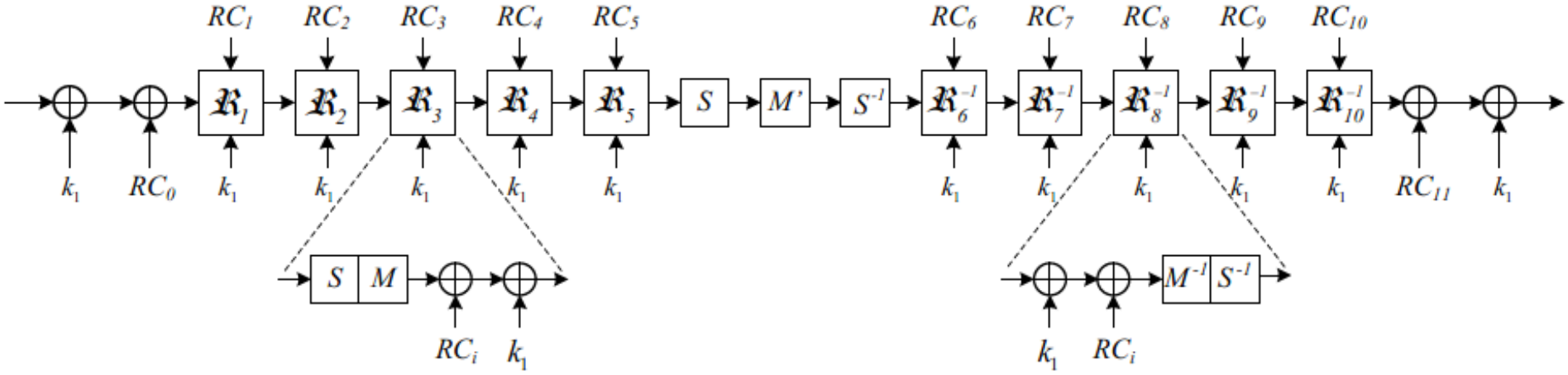
	Hardware	Software
Area/ Code size	mCrypton, LBlock, HIGHT, KLEIN, CLEFIA, KATAN, TWINE, Piccolo, KTANTAN, PRESENT, LED, SIMON	ITUBee, KLEIN, SEA, SPECK
Latency/ Execution time	PRINCE, ?	LBlock, TWINE, KLEIN, SPECK

Low-latency: PRINCE Cipher

- 64-bit block, 128-bit key
- Core cipher with 64-bit key
- 64-bit whitening keys (FX construction)
- 12 rounds



Low-latency: PRINCE Core



$$R = \mathtt{SR} \circ \mathtt{MC} \circ \mathtt{SB}\,, \quad R'_{\mathtt{PRINCE}} = \mathtt{SB}^{-1} \circ \mathtt{MC} \circ \mathtt{SB} \quad \text{and} \quad R^{-1} = \mathtt{SB}^{-1} \circ \mathtt{MC} \circ \mathtt{SR}^{-1}$$

$$\oplus_{k_i}(x) := x + k_i$$

$$\oplus_{RC_i}(x) = x + RC_i$$

Low-latency: PRINCE Core Steps

$$R = \mathtt{SR} \circ \mathtt{MC} \circ \mathtt{SB}\,, \quad R'_{\mathtt{PRINCE}} = \mathtt{SB}^{-1} \circ \mathtt{MC} \circ \mathtt{SB} \quad \text{and} \quad R^{-1} = \mathtt{SB}^{-1} \circ \mathtt{MC} \circ \mathtt{SR}^{-1}$$

$$\oplus_{k_i}(x) := x + k_i$$

$$\oplus_{RC_i}(x) = x + RC_i$$

SB Sbox

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S[x]$	B	F	3	2	A	C	9	1	6	7	8	0	E	5	D	4

$$\widehat{M}^{(0)} = \begin{pmatrix} M_1 & M_2 & M_3 & M_4 \\ M_2 & M_3 & M_4 & M_1 \\ M_3 & M_4 & M_1 & M_2 \\ M_4 & M_1 & M_2 & M_3 \end{pmatrix}, \quad \widehat{M}^{(1)} = \begin{pmatrix} M_2 & M_3 & M_4 & M_1 \\ M_3 & M_4 & M_1 & M_2 \\ M_4 & M_1 & M_2 & M_3 \\ M_1 & M_2 & M_3 & M_4 \end{pmatrix}, \quad M' = \begin{pmatrix} \widehat{M}^{(0)} & 0 & 0 & 0 \\ 0 & \widehat{M}^{(1)} & 0 & 0 \\ 0 & 0 & \widehat{M}^{(1)} & 0 \\ 0 & 0 & 0 & \widehat{M}^{(0)} \end{pmatrix}$$

Permutation of SR

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

M_i is the 4×4 identity matrix

MC-layer multiplies the state with M'

involution

Low-latency: PRINCE Core Steps

$$R = \mathtt{SR} \circ \mathtt{MC} \circ \mathtt{SB}\,, \quad R'_{\mathtt{PRINCE}} = \mathtt{SB}^{-1} \circ \mathtt{MC} \circ \mathtt{SB} \quad \text{and} \quad R^{-1} = \mathtt{SB}^{-1} \circ \mathtt{MC} \circ \mathtt{SR}^{-1}$$

$$\oplus_{k_i}(x) := x + k_i$$

$$\oplus_{RC_i}(x) = x + RC_i$$

RC_0	0000000000000000
RC_1	13198a2e03707344
RC_2	a4093822299f31d0
RC_3	082efa98ec4e6c89
RC_4	452821e638d01377
RC_5	be5466cf34e90c6c
RC_6	7ef84f78fd955cb1
RC_7	85840851f1ac43aa
RC_8	c882d32f25323c54
RC_9	64a51195e0e3610d
RC_{10}	d3b5a399ca0c2399
RC_{11}	c0ac29b7c97c50dd → $\alpha = \mathtt{c0ac29b7c97c50dd}$

$$(k_0||k_1) \mapsto (k_0||P(k_0)||k_1)$$

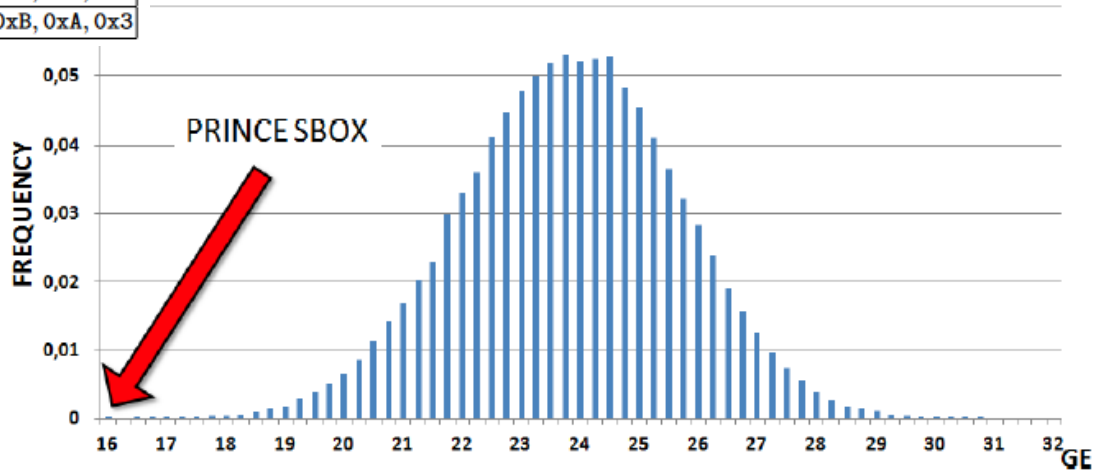
$$P(x) = (x \ggg 1) \oplus (x \gg 63)$$

Low-latency: PRINCE Sbox

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- 64000 Sboxes (and their inverses) with good cryptographic criteria are implemented and synthesized to obtain average gate counts
- Smallest Sbox selected

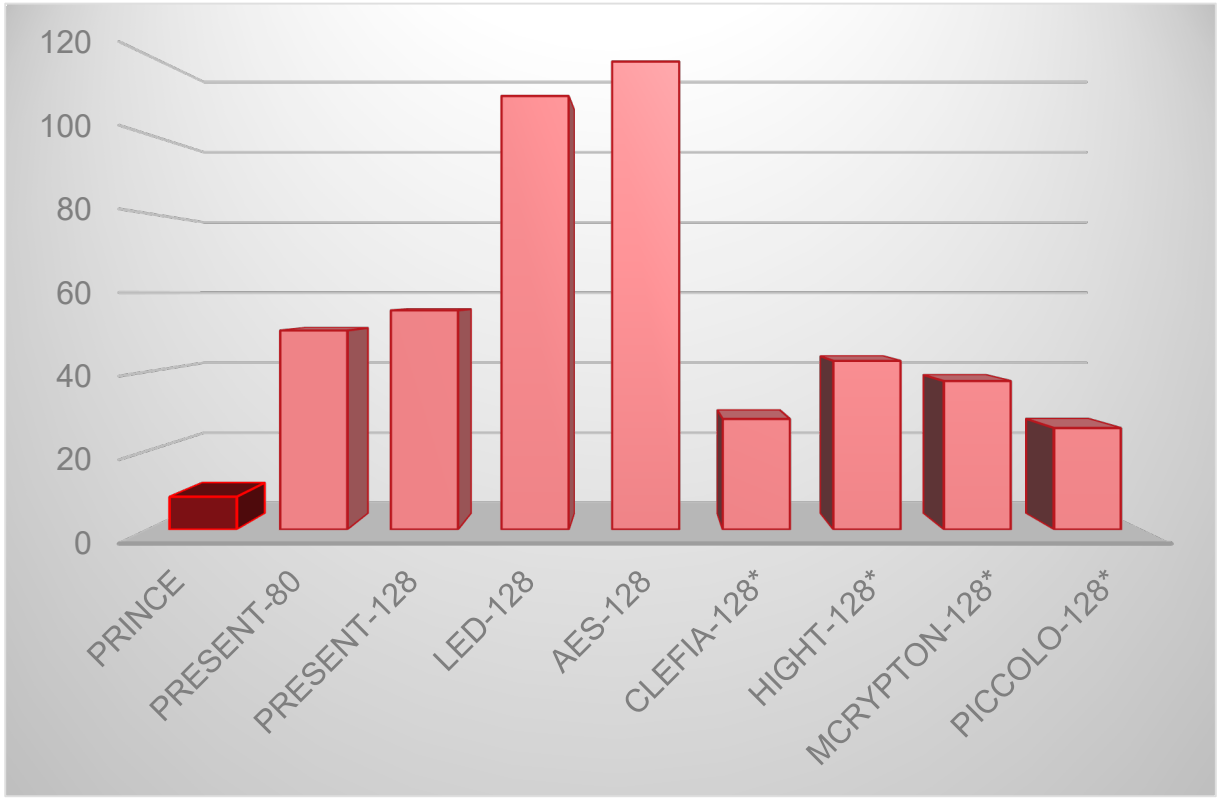
S_0	0x0, 0x1, 0x2, 0xD, 0x4, 0x7, 0xF, 0x6, 0x8, 0xC, 0x5, 0x3, 0xA, 0xE, 0xB, 0x9
S_1	0x0, 0x1, 0x2, 0xD, 0x4, 0x7, 0xF, 0x6, 0x8, 0xC, 0x9, 0xB, 0xA, 0xE, 0x5, 0x3
S_2	0x0, 0x1, 0x2, 0xD, 0x4, 0x7, 0xF, 0x6, 0x8, 0xC, 0xB, 0x9, 0xA, 0xE, 0x3, 0x5
S_3	0x0, 0x1, 0x2, 0xD, 0x4, 0x7, 0xF, 0x6, 0x8, 0xC, 0xB, 0x9, 0xA, 0xE, 0x5, 0x3
S_4	0x0, 0x1, 0x2, 0xD, 0x4, 0x7, 0xF, 0x6, 0x8, 0xC, 0xE, 0xB, 0xA, 0x9, 0x3, 0x5
S_5	0x0, 0x1, 0x2, 0xD, 0x4, 0x7, 0xF, 0x6, 0x8, 0xE, 0xB, 0xA, 0x5, 0x9, 0xC, 0x3
S_6	0x0, 0x1, 0x2, 0xD, 0x4, 0x7, 0xF, 0x6, 0x8, 0xE, 0xB, 0xA, 0x9, 0x3, 0xC, 0x5
S_7	0x0, 0x1, 0x2, 0xD, 0x4, 0x7, 0xF, 0x6, 0x8, 0xE, 0xC, 0x9, 0x5, 0xB, 0xA, 0x3



Area distribution of *good* Sboxes (90 nm)

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Results

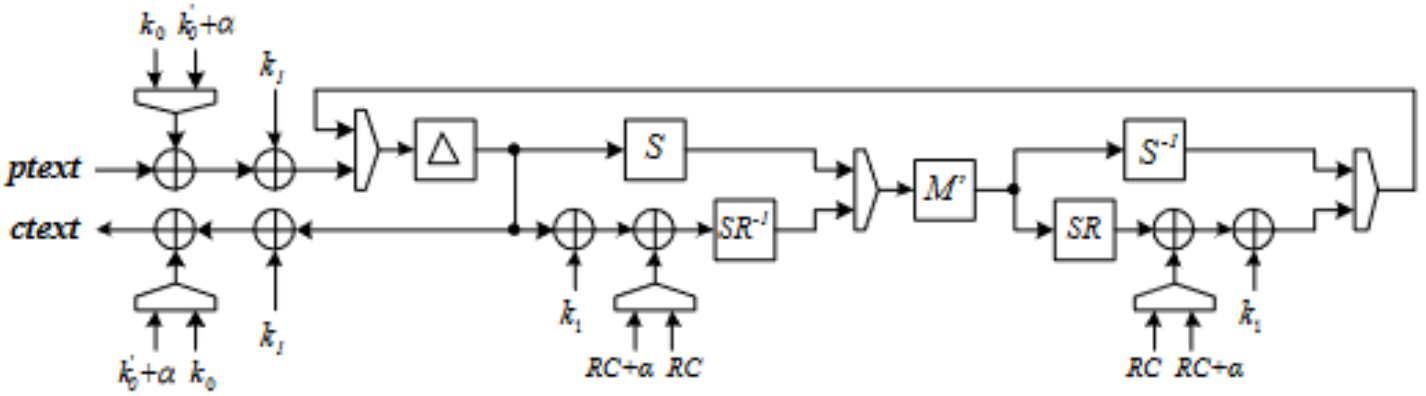


UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Results

	Tech.	Nangate 45nm Generic			UMC 90nm Faraday			UMC 130nm Faraday		
	Constr.(UD)	1000	3162	10000	1000	3162	10000	1000	3162	10000
PRINCE$^-$	Area(GE)	8260	8263	8263	7996	7996	7996	8679	8679	8679
	Power(mW)	38.5	17.9	8.3	26.3	10.9	3.9	29.8	11.8	4.1
PRESENT-80	Area(GE)	63942	51631	50429	113062	49723	49698	119196	51790	51790
	Power(mW)	1304.6	320.9	98.0	1436.9	144.9	45.5	1578.4	134.9	42.7
PRESENT-128	Area(GE)	68908	56668	55467	120271	54576	54525	126351	56732	56722
	Power(mW)	1327.1	330.4	99.1	1491.1	149.9	47.8	1638.7	137.4	43.6
LED-128	Area(GE)	109811	109958	109697	281240	286779	98100	236770	235106	111496
	Power(mW)	2470.7	835.7	252.3	5405.0	1076.3	133.7	5274.8	1133.9	163.6
AES-128	Area (GE)	135051	135093	118440	421997	130835	118522	347860	141060	130764
	Power (mW)	3265.8	1165.7	301.6	8903.2	587.4	186.8	8911.2	876.8	229.1

Low-latency: PRINCE Cipher (Round-based)

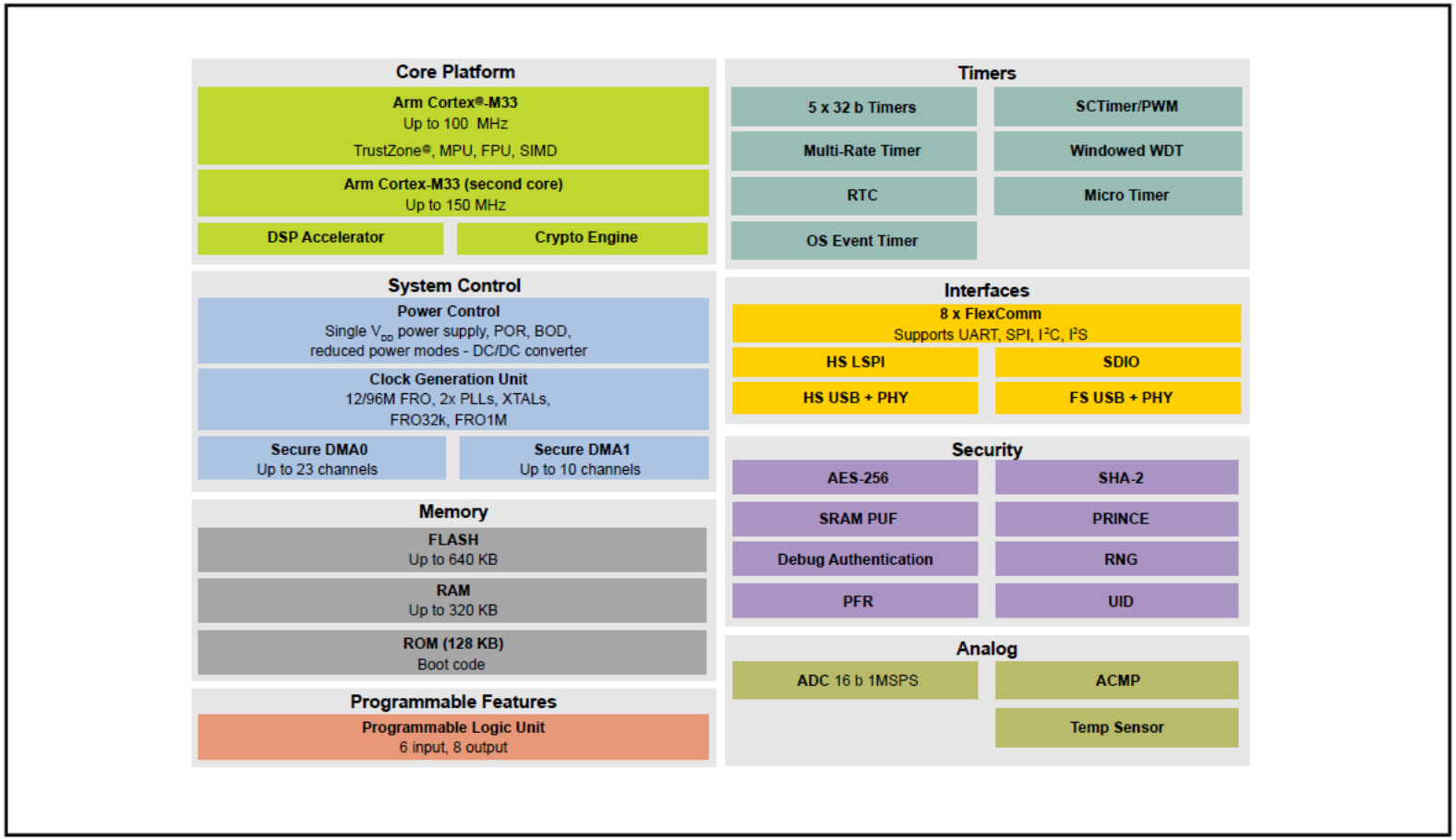


Results

	Nangate 45nm Generic				UMC 90nm Faraday				UMC 130nm Faraday			
	Area (GE)	Freq. (MHz)	Power (mW)	Tput (Gbps)	Area (GE)	Freq. (MHz)	Power (mW)	Tput (Gbps)	Area (GE)	Freq. (MHz)	Power (mW)	Tput (Gbps)
PRINCE	3779	666.7	5.7	3.56	3286	188.7	4.5	1.00	3491	153.8	5.8	0.82
PRESENT-80	3105	833.3	1.2	1.67	2795	222.2	2.1	0.44	2909	196.1	2.5	0.39
PRESENT-128	3707	833.3	1.6	1.67	3301	294.1	3.4	0.59	3458	196.1	2.9	0.39
LED-128	3309	312.5	0.5	0.41	3076	103.1	1.9	0.13	3407	78.13	2.4	0.10
AES-128	15880	250.0	5.8	2.91	14691	78.1	14.3	0.91	16212	61.3	18.8	0.71

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

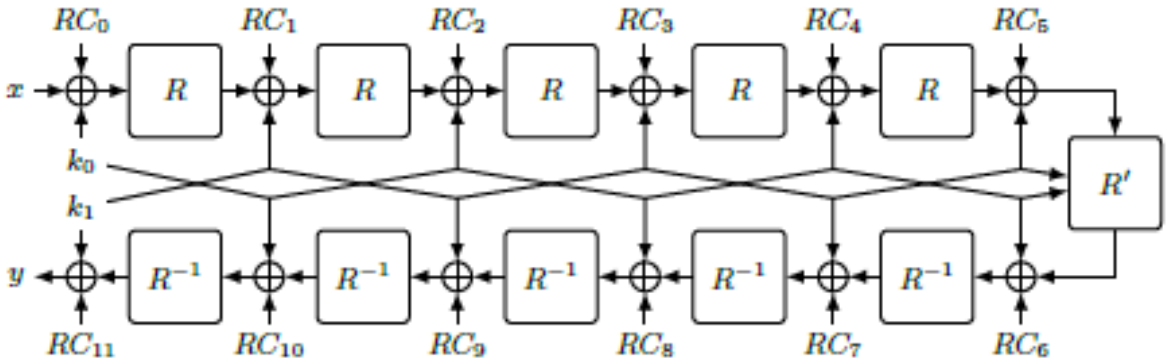
LPC55S6x MCU Block Diagram



PRINCEv2 Cipher

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

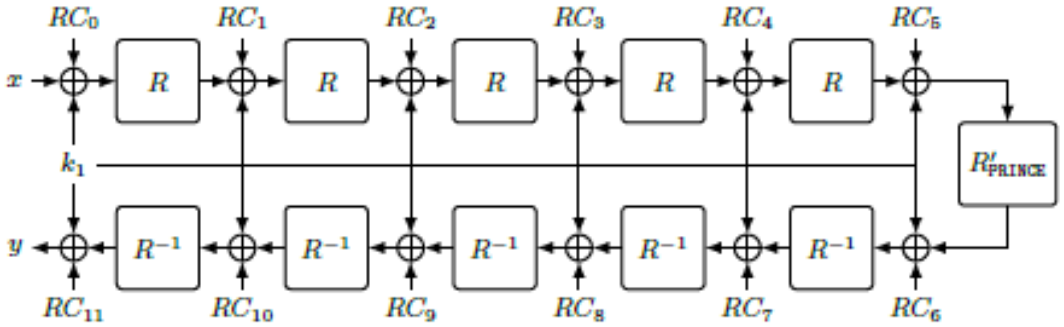
- NIST lightweight security requirement:
  - 112-bit security with at most 2^{50} bytes of chosen data
  - *PRINCE cannot reach*: Data complexity 2^n, time complexity 2^{126-n}
- PRINCEv2
  - 64-bit block, 128-bit key
  - Core cipher with 64-bit key
  - 12 rounds



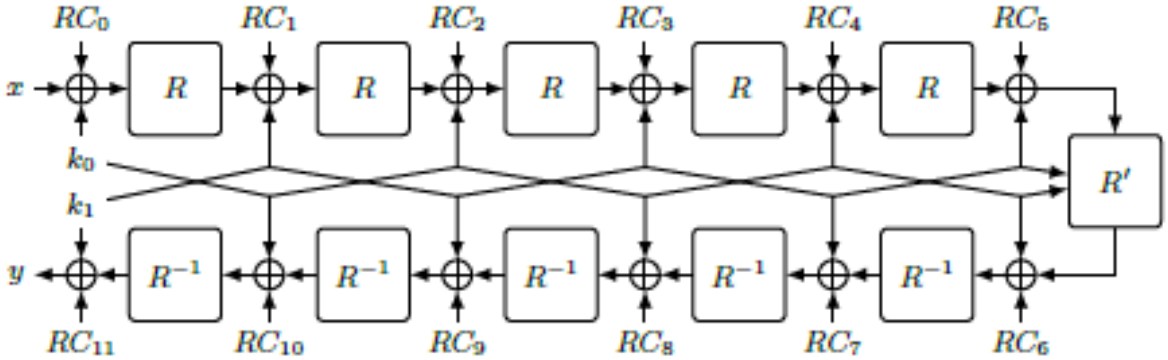
* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

PRINCEv2 Cipher

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics



- Differences to PRINCE
  - FX construction and alpha reflection removed
  - Key schedule changed, new round constant introduced
  - Keyed middle layer



* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

PRINCEv2 New Components

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Differences to PRINCE
  - FX construction and alpha reflection removed
  - Key schedule changed, new round constant introduced
  - Keyed middle layer

$$R' = \mathtt{SB}^{-1} \circ \oplus_{RC_{11}+k_1} \circ \mathtt{MC} \circ \oplus_{k_0} \circ \mathtt{SB}$$

$$\mathtt{Swap}(k_0, k_1, \mathrm{dec}) = \begin{cases} k_0, k_1 & \text{if } \mathrm{dec} = 0 \\ k_1 \oplus \beta, k_0 \oplus \alpha & \text{if } \mathrm{dec} = 1 \end{cases}$$

Constants	
$RC_0 = 0000000000000000$	$RC_6 = 7ef84f78fd955cb1 = RC_5 \oplus \alpha$
$RC_1 = 13198a2e03707344$	$RC_7 = 7aacf4538d971a60 = RC_4 \oplus \beta$
$RC_2 = a4093822299f31d0$	$RC_8 = c882d32f25323c54 = RC_3 \oplus \alpha$
$RC_3 = 082efa98ec4e6c89$	$RC_9 = 9b8ded979cd838c7 = RC_2 \oplus \beta$
$RC_4 = 452821e638d01377$	$RC_{10} = d3b5a399ca0c2399 = RC_1 \oplus \alpha$
$RC_5 = be5466cf34e90c6c$	$RC_{11} = 3f84d5b5b5470917 = RC_0 \oplus \beta$
$\alpha = c0ac29b7c97c50dd$	$\beta = 3f84d5b5b5470917$

* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

- Differences to PRINCE
  - PRINCE round keys

$k_0 \oplus k_1$, k_1 , k_1 , k_1 , k_1 , k_1 , $k_1 \oplus \alpha$, $k_1 \oplus \alpha$, $k_1 \oplus \alpha$, $k_1 \oplus \alpha$, $k_1 \oplus \alpha$, $k_0' \oplus k_1 \oplus \alpha$

  - PRINCEv2 round keys

k_0 , k_1 , k_0 , k_1 , k_0 , k_1 , k_0 , $k_1 \oplus \beta$, $k_0 \oplus \alpha$, $k_1 \oplus \beta$, $k_0 \oplus \alpha$, $k_1 \oplus \beta$, $k_0 \oplus \alpha$, $k_1 \oplus \beta$

$$k_0 \rightarrow k_1 \rightarrow k_0 \rightarrow k_1 \rightarrow k_0 \rightarrow k_1 \rightarrow k_0$$
$$\downarrow$$
$$k_1 \oplus \beta \leftarrow k_0 \oplus \alpha \leftarrow k_1 \oplus \beta \leftarrow k_0 \oplus \beta \leftarrow k_1 \oplus \beta \leftarrow k_0 \oplus \alpha \leftarrow k_1 \oplus \beta$$

$k_0 \leftarrow k_1 \oplus \beta$ and $k_1 \leftarrow k_0 \oplus \alpha$:

$$k_1 \oplus \beta \rightarrow k_0 \oplus \alpha \rightarrow k_1 \oplus \beta \rightarrow k_0 \oplus \beta \rightarrow k_1 \oplus \beta \rightarrow k_0 \oplus \alpha \rightarrow k_1 \oplus \beta$$
$$\downarrow$$
$$k_0 \oplus \alpha \oplus \beta \leftarrow k_1 \oplus \alpha \oplus \beta \leftarrow k_0 \oplus \alpha \oplus \beta \leftarrow k_1 \oplus \alpha \oplus \beta \leftarrow k_0 \oplus \alpha \oplus \beta \leftarrow k_1 \oplus \alpha \oplus \beta \leftarrow k_0 \oplus \alpha \oplus \beta$$

* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Minimum latency constrained

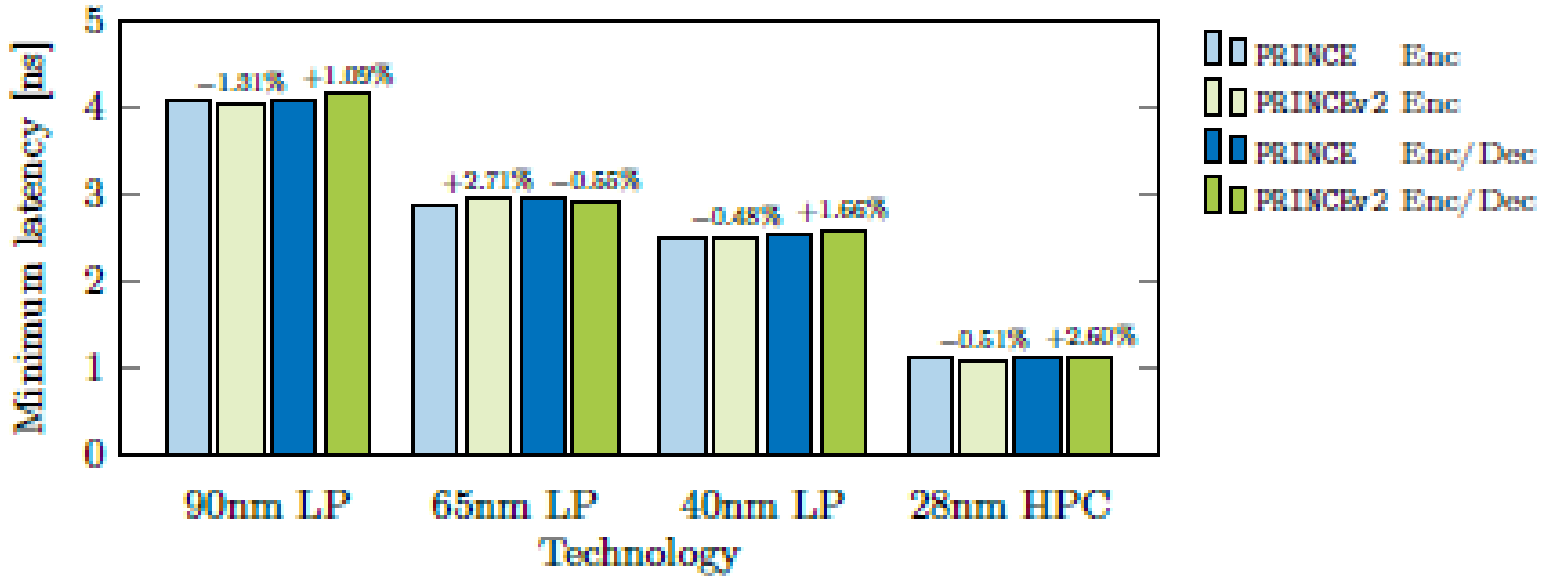
Techn.	Mode	Cipher	Area [GE]	Latency [ns]	Energy [pJ]
90 nm LP*	ENC	PRINCE	16244.25	**4.101177**	1.993172
		PRINCEv2	17661.25	**4.047311**	2.230068
	ENC/DEC	PRINCE	17808.00	**4.106262**	2.213275
		PRINCEv2	18888.75	**4.151113**	2.424250
65 nm LP*	ENC	PRINCE	19877.75	**2.866749**	1.602513
		PRINCEv2	18798.25	**2.944367**	1.492794
	ENC/DEC	PRINCE	19966.00	**2.946442**	1.594025
		PRINCEv2	21171.25	**2.930153**	1.696559
40 nm LP*	ENC	PRINCE	17177.00	**2.521302**	0.617719
		PRINCEv2	16556.50	**2.509131**	0.592155
	ENC/DEC	PRINCE	17377.50	**2.541220**	0.630223
		PRINCEv2	17799.50	**2.583466**	0.648450
28 nm HPC**	ENC	PRINCE	38145.33	**1.108886**	1.258586
		PRINCEv2	33470.33	**1.103273**	1.108789
	ENC/DEC	PRINCE	35297.67	**1.119593**	1.181171
		PRINCEv2	38962.33	**1.148693**	1.299172

* LP = Low Power
** HPC = High Performance Computing

* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Minimum latency constrained



* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Minimum area constrained

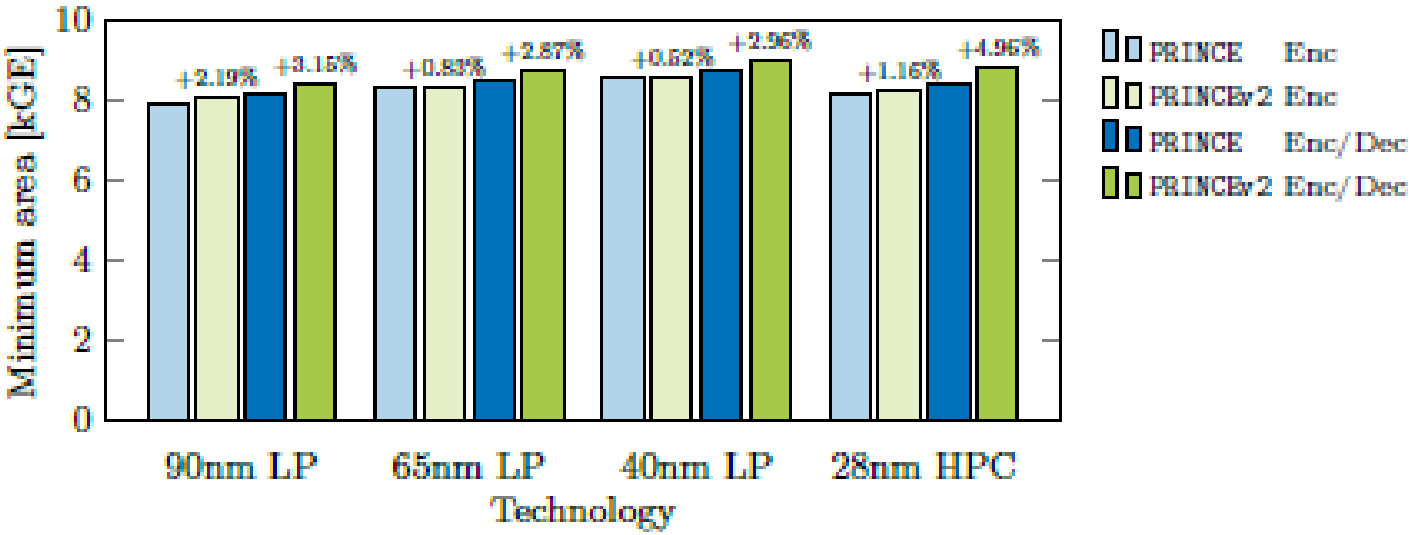
Techn.	Mode	Cipher	Area [GE]	Latency [ns]	Energy [pJ]
90 nm LP*	ENC	PRINCE	**7937.50**	12.859 908	0.569694
		PRINCEv2	**8111.25**	12.856 450	0.574683
	ENC/DEC	PRINCE	**8183.00**	14.015 245	0.616671
		PRINCEv2	**8440.75**	15.513 536	0.628298
65 nm LP*	ENC	PRINCE	**8316.00**	11.434 771	0.433378
		PRINCEv2	**8385.25**	11.504 968	0.430286
	ENC/DEC	PRINCE	**8547.75**	12.349 355	0.440872
		PRINCEv2	**8792.75**	13.376 949	0.456154
40 nm LP*	ENC	PRINCE	**8563.75**	10.144 847	0.212027
		PRINCEv2	**8608.50**	10.063 908	0.207317
	ENC/DEC	PRINCE	**8780.00**	10.886 960	0.217739
		PRINCEv2	**9039.75**	11.798 657	0.226534
28 nm HPC**	ENC	PRINCE	**8197.00**	3.599 936	0.127798
		PRINCEv2	**8292.00**	3.682 593	0.127786
	ENC/DEC	PRINCE	**8426.33**	4.260 999	0.131239
		PRINCEv2	**8844.67**	4.323 993	0.134909

* LP = Low Power
** HPC = High Performance Computing

* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

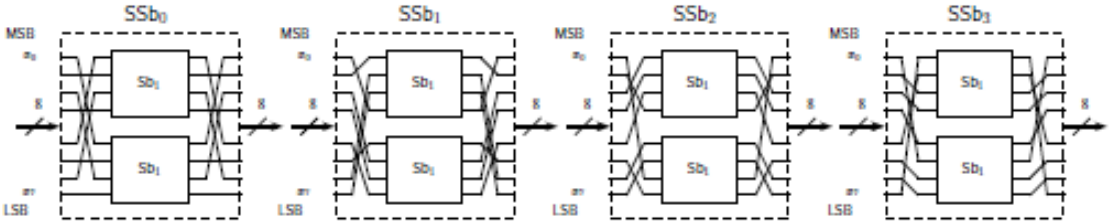
Minimum area constrained

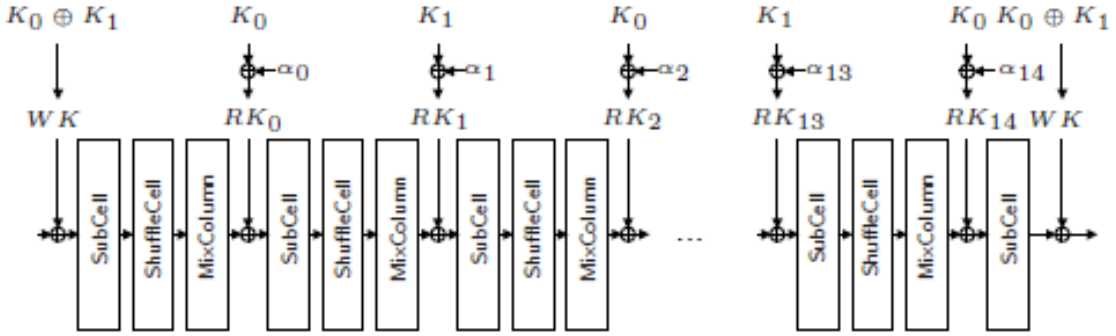


* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

Midori

- Low energy oriented design
  - Not necessarily for low latency but compared with PRINCE
- Cipher specifics
  - 64/128-bit block, 128-bit key
  - SPN: 2 bijective Sboxes (nonlinear) and involutive binary matrix (linear)
  - 16/20 rounds



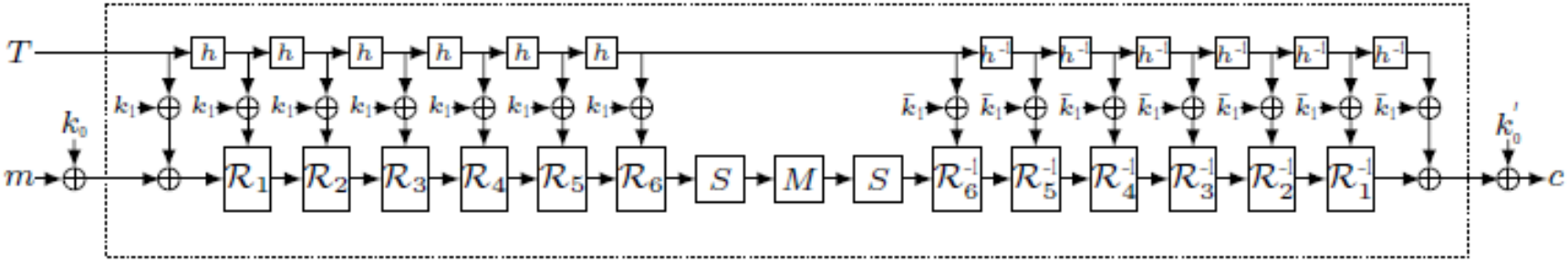


* Banik et al., Midori: A Block Cipher for Low Energy, ASIACRYPT, 2015

MANTIS

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Turning PRINCE into a tweakable block cipher
  - Well understood when TWEAKEY framework is used
- Cipher specifics
  - 64-bit block, 128-bit key, 64-bit tweak
  - FX-design, SPN
    - Midori Sbox used as it has better latency than PRINCE Sbox
  - 14 rounds (PRINCE-like middle)

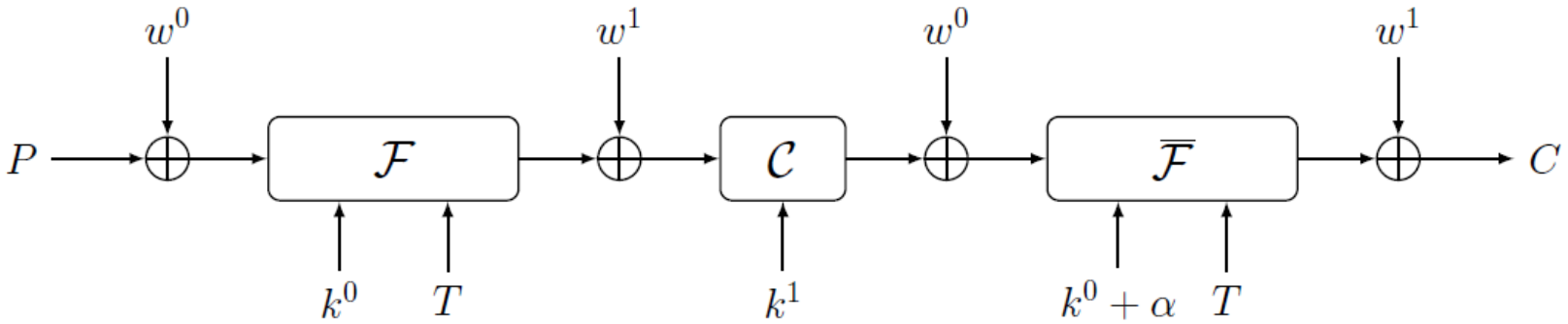


* Beierle et al., The SKINNY Family of Block Ciphers and its Low-Latency Variant MANTIS, CRYPTO, 2016

QARMA

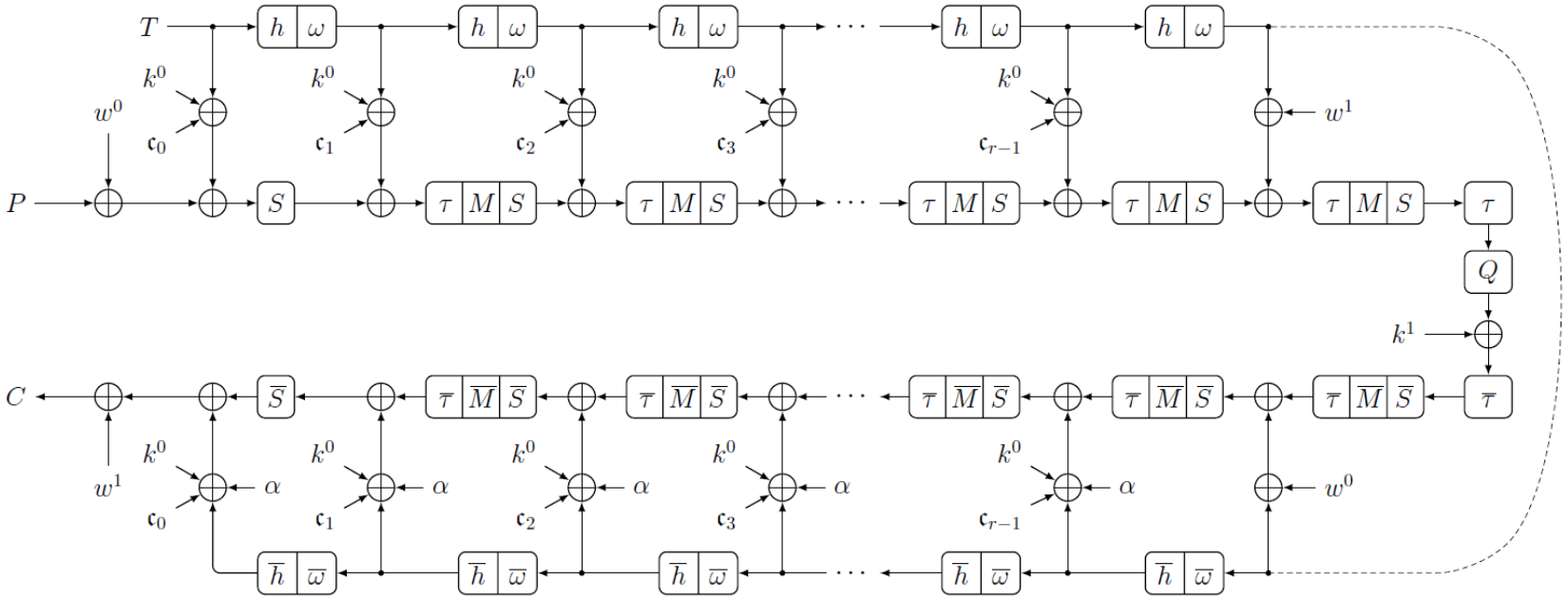
UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- QARMA (from Qualcomm ARM Authenticator)
  - Lightweight tweakable block cipher
  - Primarily known for its use in the ARMv8 architecture
    - For protection of software as a cryptographic hash for the Pointer Authentication Code
- Cipher specifics
  - 64/128-bit block, 128/256-bit key (round numbers 7/10 in permutation)
  - An Even-Mansour cipher using three stages, with whitening keys w^0 and w^1 XORed in between permutation F (and its inverse) which uses using core key k^0 and parameterized by a tweak T and "central" permutation C which uses key k^1 and is designed to be reversible via a simple key transformation



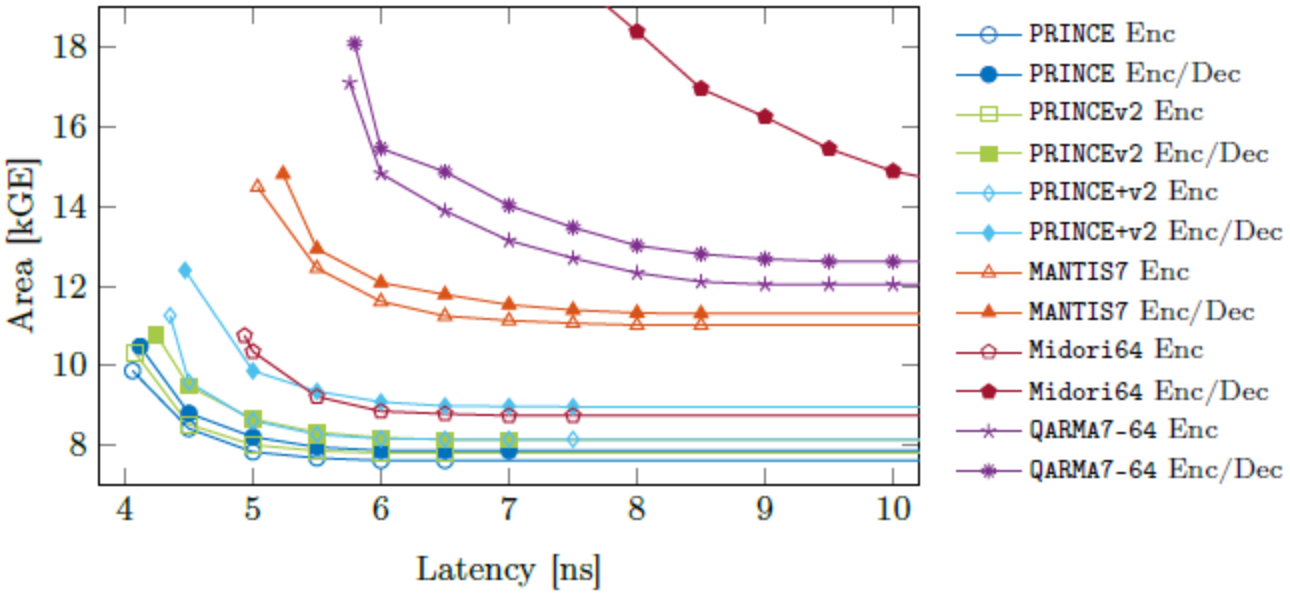
* Roberto Avanzi, The QARMA Block Cipher Family, ToSC, vol. 17, 2017

QARMA



* Roberto Avanzi, The QARMA Block Cipher Family, ToSC, vol. 17, 2017

Comparison so far…



Comparison of unrolled block ciphers in NanGate 45 nm Open Cell Library.

* Bozilov et al., PRINCEv2: More Security for (Almost) No Overhead, SAC, 2020

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

K-Cipher: A Low Latency, Bit Length Parameterizable Cipher

Michael Kounavis, Sergej Deutsch, Santosh Ghosh, and David Durham

Intel Labs, Intel Corporation, 2111, NE 25th Avenue, Hillsboro, OR 97124
Email: {michael.e.kounavis, sergej.deutsch, santosh.ghosh, david.durham}@intel.com

Abstract— We present the design of a novel low latency, bit length parameterizable cipher, called the "K-Cipher". K-Cipher is particularly useful to applications that need to support ultra low latency encryption at arbitrary ciphertext lengths. We can think of a range of networking, gaming and computing applications that may require encrypting data at unusual block lengths for many different reasons, such as to make space for other unencrypted state values. Furthermore, in modern applications, encryption is typically required to complete inside stringent time frames in order not to affect performance. K-Cipher has been designed to meet these requirements. In the paper we present the K-Cipher design and discuss its rationale. We also present results from our ongoing security analysis which suggest that only 2 to 4 rounds are sufficient to make the cipher operate securely. Finally, we present synthesis results from 2-round 32-bit and 64-bit K-Cipher encrypt datapaths, produced using Intel's ® 10 nm process technology. Our results show that the encrypt datapaths can complete in no more than 767 psec, or 3 clocks in 3.9-4.9 GHz frequencies, and are associated with a maximum area requirement of 1875 μm^2.

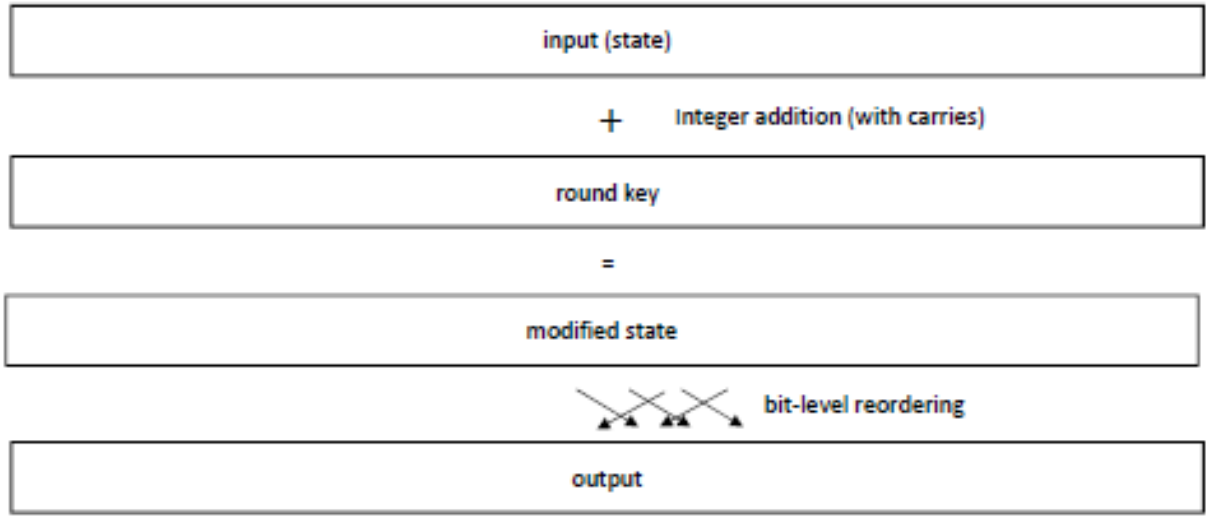


Fig. 1. The Aggressive Adder Component of the K-Cipher Round

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

K-Cipher: A Low Latency, Bit Length Parameterizable Cipher

Michael Kounavis, Sergej Deutsch, Santosh Ghosh, and David Durham

Intel Labs, Intel Corporation, 2111, NE 25th Avenue, Hillsboro, OR 97124
Email: {michael.e.kounavis, sergej.deutsch, santosh.ghosh, david.durham}@intel.com

Abstract— We present the design of a novel low latency, bit length parameterizable cipher, called the "K-Cipher". K-Cipher is particularly useful to applications that need to support ultra low latency encryption at arbitrary ciphertext lengths. We can think of a range of networking, gaming and computing applications that may require encrypting data at unusual block lengths for many different reasons, such as to make space for other unencrypted state values. Furthermore, in modern applications, encryption is typically required to complete inside stringent time frames in order not to affect performance. K-Cipher has been designed to meet these requirements. In the paper we present the K-Cipher design and discuss its rationale. We also present results from our ongoing security analysis which suggest that only 2 to 4 rounds are sufficient to make the cipher operate securely. Finally, we present synthesis results from 2-round 32-bit and 64-bit K-Cipher encrypt datapaths, produced using Intel's ® 10 nm process technology. Our results show that the encrypt datapaths can complete in no more than 767 psec, or 3 clocks in 3.9-4.9 GHz frequencies, and are associated with a maximum area requirement of 1875 μm^2.

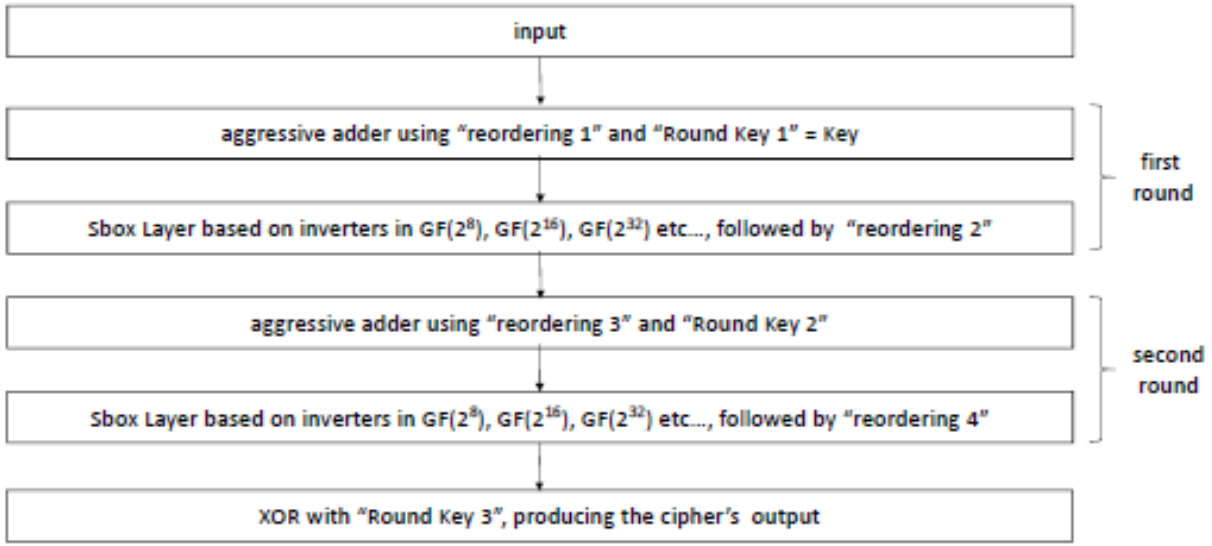


Fig. 2. Two Round K-Cipher Specification

cipher	area (μm^2)	latency (psec)	number of clocks	freq.
K-Cipher Enc-32, $r = 2$	614	**613**	3	4.9 GHz
K-Cipher Enc-64, $r = 2$	1875	**767**	3	3.9 GHz

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

The SPEEDY Family of Block Ciphers

Engineering an Ultra Low-Latency Cipher from Gate Level for Secure Processor Architectures

Gregor Leander[1], Thorben Moos[1], Amir Moradi[1] and Shahram Rasoolzadeh*[2]

[1] Ruhr University Bochum, Horst Görtz Institute for IT Security, Bochum, Germany
firstname.lastname@rub.de
[2] Radboud University, Nijmegen, The Netherlands
firstname.lastname@ru.nl

Abstract. We introduce SPEEDY, a family of ultra low-latency block ciphers. We mix engineering expertise into each step of the cipher's design process in order to create a secure encryption primitive with an extremely low latency in CMOS hardware. The centerpiece of our constructions is a high-speed 6-bit substitution box whose coordinate functions are realized as two-level NAND trees. In contrast to other low-latency block ciphers such as PRINCE, PRINCEv2, MANTIS and QARMA, we neither constrain ourselves by demanding decryption at low overhead, nor by requiring a super low area or energy. This freedom together with our gate- and transistor-level considerations allows us to create an ultra low-latency cipher which outperforms all known solutions in single-cycle encryption speed. Our main result, SPEEDY-6-192, is a 6-round 192-bit block and 192-bit key cipher which can be executed faster in hardware than any other known encryption primitive (including Gimli in Even-Mansour scheme and the Orthros pseudorandom function) and offers 128-bit security. One round more, i.e., SPEEDY-7-192, provides full 192-bit security. SPEEDY primarily targets hardware security solutions embedded in high-end CPUs, where area and energy restrictions are secondary while high performance is the number one priority.

Keywords: Low-Latency Cryptography, High-Speed Encryption, Block Cipher

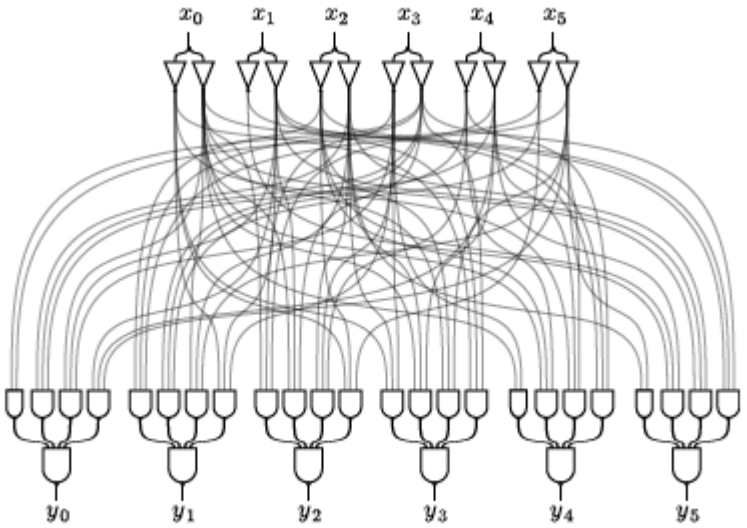
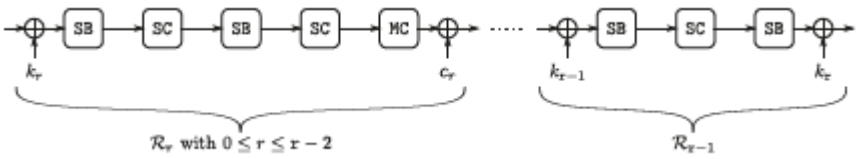


Figure 3: Implementation of the 6-bit S-box of SPEEDY based on two-level NAND trees.



UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Cipher	Minimum Latency [ns]					
	Commercial Foundry				NanGate OCL	
	90 nm LP	65 nm LP	40 nm LP	28 nm HPC	45 nm	15 nm
Ginli E-N	4.532467	3.330192	2.794736	1.178424	4.537304	0.435069
NANTIS$_6$	4.625529	3.405490	2.891383	1.278725	4.479773	0.437595
NANTIS$_7$	5.201681	3.722473	3.234409	1.421365	5.074452	0.492703
NANTIS$_8$	5.823127	4.233543	3.631438	1.594997	5.739020	0.552384
Nidori	5.061255	3.582221	3.142355	1.362237	4.934847	0.481522
Orthros	3.862139	2.678637	2.401275	1.087139	3.774836	0.369497
PRINCE	4.101177	2.866749	2.521302	1.108886	4.059997	0.389144
PRINCEv2	4.047311	2.944367	2.509131	1.103273	4.077636	0.387146
QARMA$_5$-64-σ_0	4.075846	2.920377	2.498908	1.134901	4.014516	0.385281
QARMA$_6$-64-σ_0	4.770325	3.418600	2.951308	1.308331	4.554445	0.448931
QARMA$_7$-64-σ_0	5.449707	3.909138	3.389576	1.538606	5.336362	0.517093
QARMA$_8$-64-σ_0	6.103768	4.396543	3.814078	1.697027	5.966323	0.575525
QARMA$_5$-64-σ_1	4.515514	3.284252	2.815788	1.219624	4.367899	0.408580
QARMA$_6$-64-σ_1	5.297867	3.808675	3.271455	1.388353	4.944635	0.472798
QARMA$_7$-64-σ_1	6.014477	4.371963	3.745959	1.601572	5.800633	0.542712
QARMA$_8$-64-σ_1	6.720944	4.904521	4.202632	1.797539	6.498429	0.608985
SPEEDY-5-192	2.994643	2.178075	1.867064	0.847761	3.187368	0.300466
SPEEDY-6-192	3.637978	2.639186	2.277422	1.032206	3.848132	0.366762
SPEEDY-7-192	4.261928	3.087257	2.663004	1.217946	4.515505	0.431032
SPEEDY-5-192 *	2.941130	2.121748	1.820950	0.826217	2.817971	0.290961
SPEEDY-6-192 *	3.559981	2.573561	2.223863	1.011173	3.382270	0.353391
SPEEDY-7-192 *	4.174183	3.029217	2.620612	1.186598	3.995325	0.413950

* = Optimized HDL code with direct instantiation of library cells based on Figures 3 and 4.

Cipher	Area [GE]					
	Commercial Foundry				NanGate OCL	
	90 nm LP	65 nm LP	40 nm LP	28 nm HPC	45 nm	15 nm
Ginli E-N	72644.00	82781.00	63100.50	144036.33	52038.67	57551.25
NANTIS$_6$	21045.75	23264.50	20448.25	36073.33	12660.67	15954.00
NANTIS$_7$	23229.25	26385.75	23192.50	43220.33	14225.67	17522.50
NANTIS$_8$	26365.75	30316.75	25429.75	50793.00	15663.33	19707.50
Nidori	18678.50	21964.00	17562.25	41450.67	10675.33	13927.25
Orthros	49639.75	61657.00	44715.75	74384.67	31317.33	39165.00
PRINCE	16244.25	19877.75	17177.00	38145.33	9873.33	13291.00
PRINCEv2	17661.25	18798.25	16556.50	33470.33	10332.00	13069.50
QARMA$_5$-64-σ_0	19590.75	21706.75	20255.00	31703.00	11824.67	14880.75
QARMA$_6$-64-σ_0	22624.25	25349.50	22689.00	38813.67	14165.67	17621.75
QARMA$_7$-64-σ_0	25614.00	29323.00	24656.25	40494.33	15769.33	19770.25
QARMA$_8$-64-σ_0	28813.75	32780.75	28262.75	47952.33	17908.00	22074.00
QARMA$_5$-64-σ_1	20264.75	23753.00	20202.25	34302.00	12350.33	15588.75
QARMA$_6$-64-σ_1	23162.25	26941.25	23333.75	45419.00	15066.00	18164.00
QARMA$_7$-64-σ_1	26563.75	31495.00	27059.50	52108.00	16641.00	20670.25
QARMA$_8$-64-σ_1	30534.50	35787.75	29116.50	54967.00	18963.67	22761.75
SPEEDY-5-192	47364.00	53856.00	47528.50	74467.00	27903.33	34649.00
SPEEDY-6-192	57322.00	64438.25	56816.00	88932.00	34085.00	41443.25
SPEEDY-7-192	68370.00	75273.00	65422.00	95235.67	39853.33	48727.75
SPEEDY-5-192 *	49902.00	58796.25	55846.75	80313.33	29839.00	38075.25
SPEEDY-6-192 *	59688.00	70653.00	66553.00	98950.00	36523.33	46266.50
SPEEDY-7-192 *	73397.75	84745.00	77519.75	111754.33	42813.33	54193.25

* = Optimized HDL code with direct instantiation of library cells based on Figures 3 and 4.

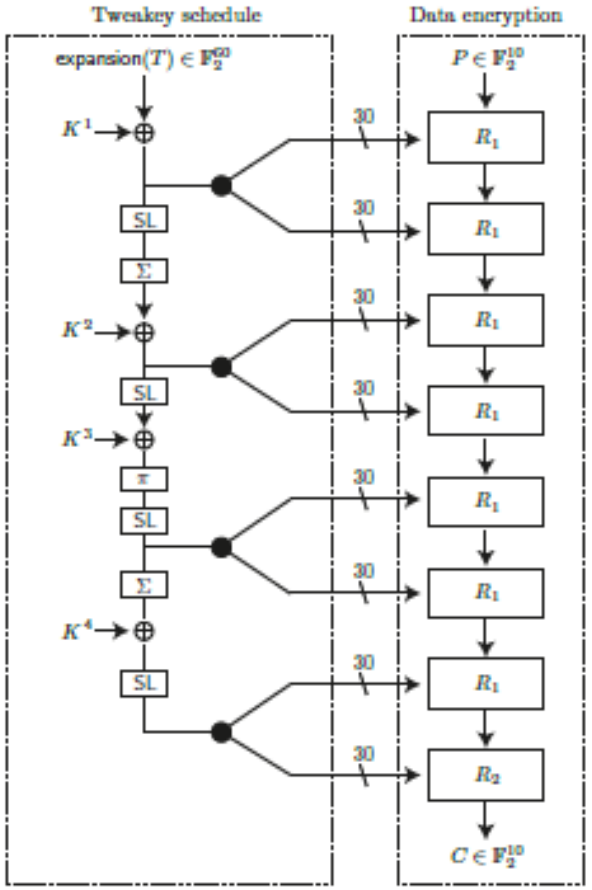
UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

SCARF – A Low-Latency Block Cipher for Secure Cache-Randomization

Federico Canale[1], Tim Güneysu[1,4], Gregor Leander[1], Jan Philipp Thoma[1], Yosuke Todo[2], and Rei Ueno[3]

[1] Ruhr University Bochum, Bochum, Germany firstname.lastname@rub.de
[2] NTT Social Informatics Laboratories, Tokyo, Japan yosuke.todo@ntt.com
[3] Tohoku University, Sendai-shi, Japan. rei.ueno.a8@tohoku.ac.jp
[4] DFKI, Bremen, Germany.

Abstract. Randomized cache architectures have proven to significantly increase the complexity of contention-based cache side-channel attacks and therefore present an important building block for side-channel secure microarchitectures. By randomizing the address-to-cache-index mapping, attackers can no longer trivially construct minimal eviction sets which are fundamental for contention-based cache attacks. At the same time, randomized caches maintain the flexibility of traditional caches, making them broadly applicable across various CPU types. This is a major advantage over cache partitioning approaches.
A large variety of randomized cache architectures has been proposed. However, the actual randomization function received little attention and is often neglected in these proposals. Since the randomization operates directly on the critical path of the cache lookup, the function needs to have extremely low latency. At the same time, attackers must not be able to bypass the randomization which would nullify the security benefit of the randomized mapping. In this paper, we propose SCARF (Secure CAche Randomization Function), the first dedicated cache randomization cipher which achieves low latency and is cryptographically secure in the cache attacker model. The design methodology for this dedicated cache cipher enters new territory in the field of block ciphers with a small 10-bit block length and heavy key-dependency in few rounds.



UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

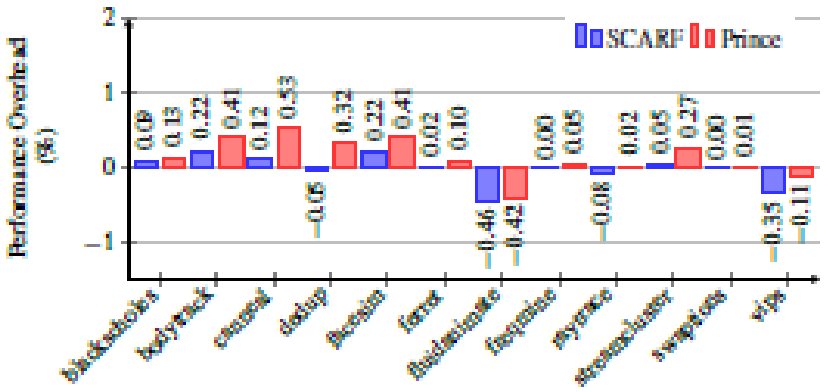
SCARF – A Low-Latency Block Cipher for Secure Cache-Randomization

Federico Canale[1], Tim Güneysu[1,4], Gregor Leander[1], Jan Philipp Thoma[1], Yosuke Todo[2], and Rei Ueno[3]

[1] Ruhr University Bochum, Bochum, Germany firstname.lastname@rub.de
[2] NTT Social Informatics Laboratories, Tokyo, Japan yosuke.todo@ntt.com
[3] Tohoku University, Sendai-shi, Japan. rei.ueno.a8@tohoku.ac.jp
[4] DFKI, Bremen, Germany.

Abstract. Randomized cache architectures have proven to significantly increase the complexity of contention-based cache side-channel attacks and therefore present an important building block for side-channel secure microarchitectures. By randomizing the address-to-cache-index mapping, attackers can no longer trivially construct minimal eviction sets which are fundamental for contention-based cache attacks. At the same time, randomized caches maintain the flexibility of traditional caches, making them broadly applicable across various CPU types. This is a major advantage over cache partitioning approaches.
A large variety of randomized cache architectures has been proposed. However, the actual randomization function received little attention and is often neglected in these proposals. Since the randomization operates directly on the critical path of the cache lookup, the function needs to have extremely low latency. At the same time, attackers must not be able to bypass the randomization which would nullify the security benefit of the randomized mapping. In this paper, we propose SCARF (Secure CAche Randomization Function), the first dedicated cache randomization cipher which achieves low latency and is cryptographically secure in the cache attacker model. The design methodology for this dedicated cache cipher enters new territory in the field of block ciphers with a small 10-bit block length and heavy key-dependency in few rounds.

Technology	45 nm		15 nm	
	Latency [ns]	Area [GE]	Latency [ps]	Area [GE]
PRINCE	4.74	12,554	628.49	17,484
MANTIS6	4.73	13,129	630.07	17,641
QARMA5	4.40	13,915	563.62	18,455
SCARF	**2.26**	**7,335**	**305.76**	**8,118**



UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

LLLWBC: A New Low-Latency Light-Weight Block Cipher

Lei Zhang[1,2(✉)], Ruichen Wu[1], Yuhan Zhang[1], Yafei Zheng[1,2], and Wenling Wu[1]

[1] Institute of Software, Chinese Academy of Sciences, Beijing 100190, China
zhanglei@iscas.ac.cn
[2] State Key Laboratory of Cryptology, P. O. Box 5159, Beijing, China

Abstract. Lightweight cipher suitable for resource constrained environment is crucial to the security of applications such as RFID, Internet of Things, etc. Moreover, in recent years low-latency is becoming more important and highly desirable by some specific applications which need instant response and real-time security. In this paper, we propose a new low-latency block cipher named LLLWBC. Similar to other known low-latency block ciphers, LLLWBC preserves the important α-reflection property, namely the decryption for a key K is equal to encryption with a key $K \oplus \alpha$ where α is a fixed constant. However, instead of the normally used SP-type construction, the core cipher employs a variant of generalized Feistel structure called extended GFS. It has 8 branches and employs byte-wise round function and nibble-wise round permutation iterated for 21 rounds. We choose the round permutations carefully together with a novel key schedule to guarantee the α-reflection property. This allows an efficient fully unrolled implementation of LLLWBC in hardware and the overhead of decryption on top of encryption is negligible. Moreover, because of the involutory property of extended GFS, the inverse round function is not needed, which makes it possible to be implemented in round-based architecture with a competitive area cost. Furthermore, our security evaluation shows that LLLWBC can achieve enough security margin within the constraints of security claims. Finally, we evaluate the hardware and software performances of LLLWBC on various platforms and a brief comparison with other low-latency ciphers is also presented.

Keywords: Block cipher · Low-latency · Lightweight · Extended GFS

Table 7. Performance results of fully unrolled version of LLLWBC and other ciphers.

Cipher	Technology	Latency(ns)	Area(GE)	Source
LLLWBC	NanGate 45 nm Generic	11.76	8226.85	This paper
PRINCE	NanGate 45 nm Generic	–	8263	[6]
MANTIS$_7$	UMC L180 0.18 μm 1P6M	20.50	11209	[4]
QARMA$_7$	FinFet 7 nm	6.04	17109	[1]

Table 8. Performance results of round-based version of LLLWBC and PRINCE.

Cipher	Technology	Latency(ns)	Area(GE)	Source
LLLWBC	NanGate 45 nm Generic	0.64	1024.10	This paper
PRINCE	NanGate 45 nm Generic	–	3779	[6]

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Introducing two Low-Latency Cipher Families: Sonic and SuperSonic

Yanis Belkheyar[1], Joan Daemen[1], Christoph Dobraunig[2], Santosh Ghosh[2] and Shahram Rasoolzadeh[1]

[1] Digital Security Group, Radboud University, Nijmegen, The Netherlands
firstname.lastname@ru.nl
[2] Intel Labs, Hillsboro, USA firstname.lastname@intel.com

Abstract. For many latency-critical operations in computer systems, like memory reads/writes, adding encryption can have a big impact on the performance. Hence, the existence of cryptographic primitives with good security properties and minimal latency is a key element in the wide-spread implementation of such security measures. In this paper, we introduce two new families of low-latency permutations/block ciphers called SONIC and SUPERSONIC, inspired by the SIMON block ciphers.

Keywords: low-latency, Simon, Sonic, SuperSonic, Feistel structure, gate-delay-balenced Feistel, block cipher

cipher	word size (w) [bits]	block size (b) [bits]	number of rounds	target security [bits]
SONIC256	128	256	24	128
SONIC512	256	512	24	128
SUPERSONIC256	128	256	21	128
SUPERSONIC512	256	512	21	128

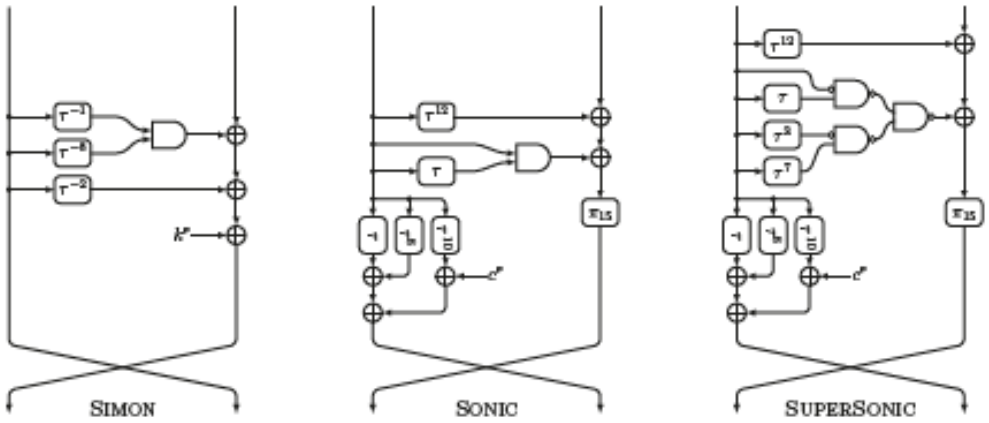
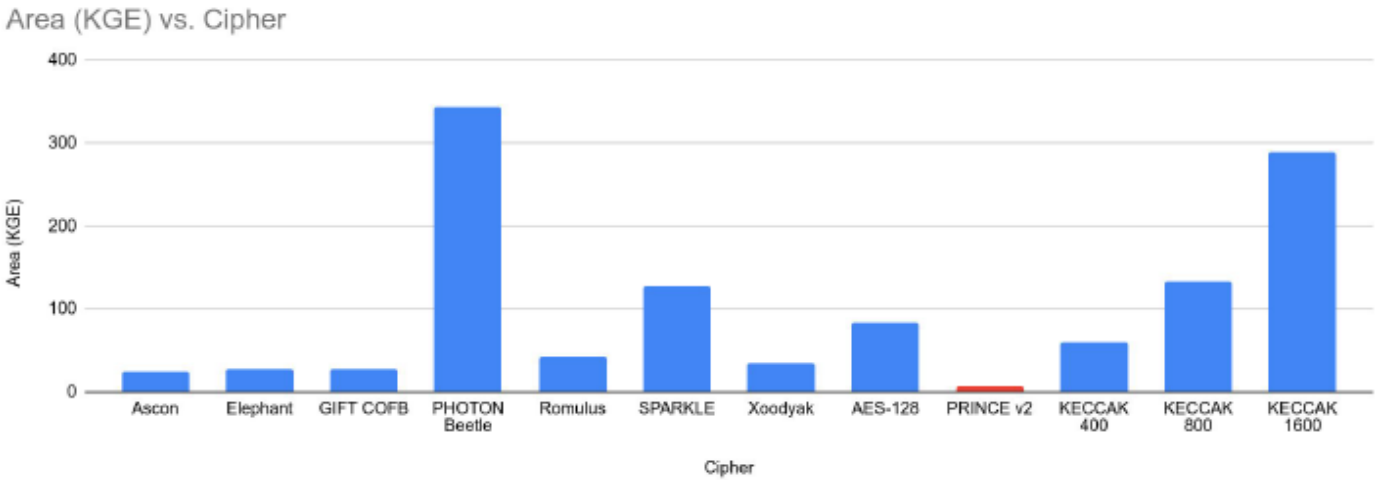


Figure 1: Comparison of round functions of SIMON, SONIC, and SUPERSONIC.

ASCON?

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Compactness												
	Ascon	Elephant	GIFT COFB	PHOTON Beetle	Romulus	SPARKLE	Xoodyak	AES-128	**PRINCE v2**	KECCAK 400	KECCAK 800	KECCAK 1600
Area (KGE)	24.437	28.350	28.709	343.55	43.392	128.51	34.148	83.581	**7.9813**	60.128	132.74	289.92

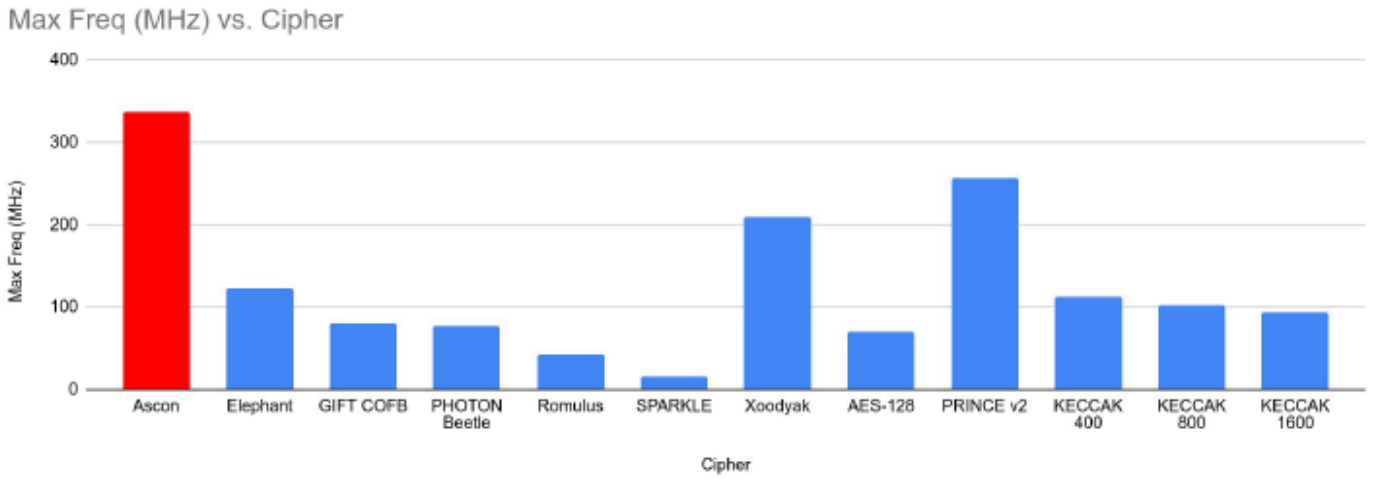


* Yalcin et al., Need for Low-latency Ciphers: A Comparative Study of NIST LWC Finalists, NIST Lightweight Crypto WS, 2022

ASCON?

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

High Frequency												
	Ascon	Elephant	GIFT COFB	PHOTON Beetle	Romulus	SPARKLE	Xoodyak	AES-128	PRINCE v2	KECCAK 400	KECCAK 800	KECCAK 1600
Max Freq (MHz)	**336.7**	122.55	79.81	77.700	43.535	16.736	209.21	69.735	257.07	112.36	102.88	93.897

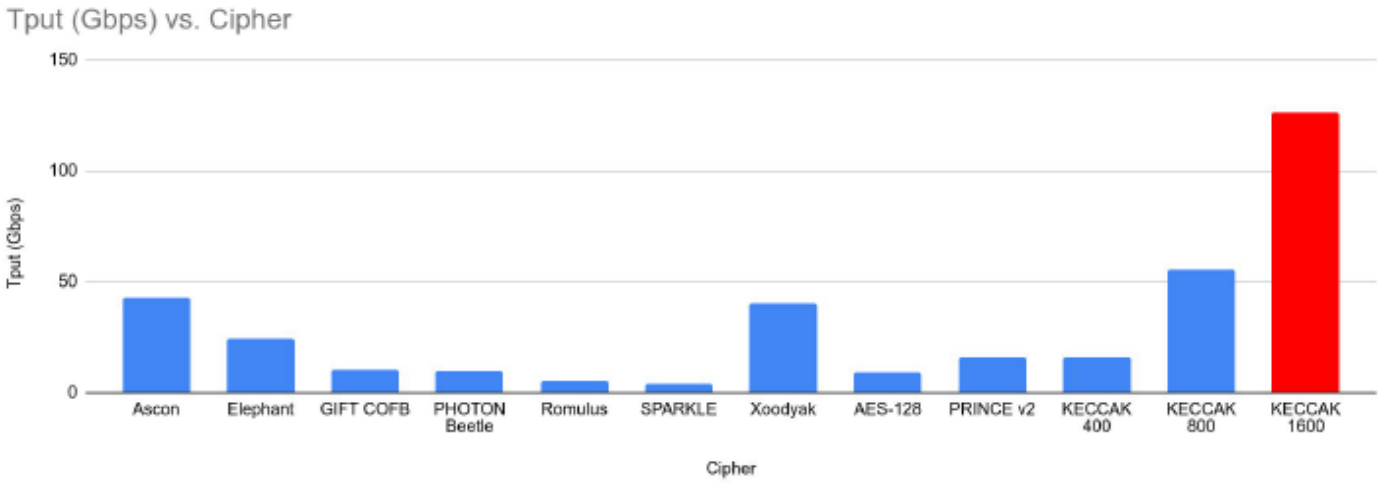


* Yalcin et al., Need for Low-latency Ciphers: A Comparative Study of NIST LWC Finalists, NIST Lightweight Crypto WS, 2022

ASCON?

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

High Throughput												
	Ascon	Elephant	GIFT COFB	PHOTON Beetle	Romulus	SPARKLE	Xoodyak	AES-128	PRINCE v2	KECCAK 400	KECCAK 800	**KECCAK 1600**
Tput (Gbps)	43.098	24.510	10.215	9.945	5.572	4.2845	40.167	8.9261	16.452	16.180	55.967	**126.20**

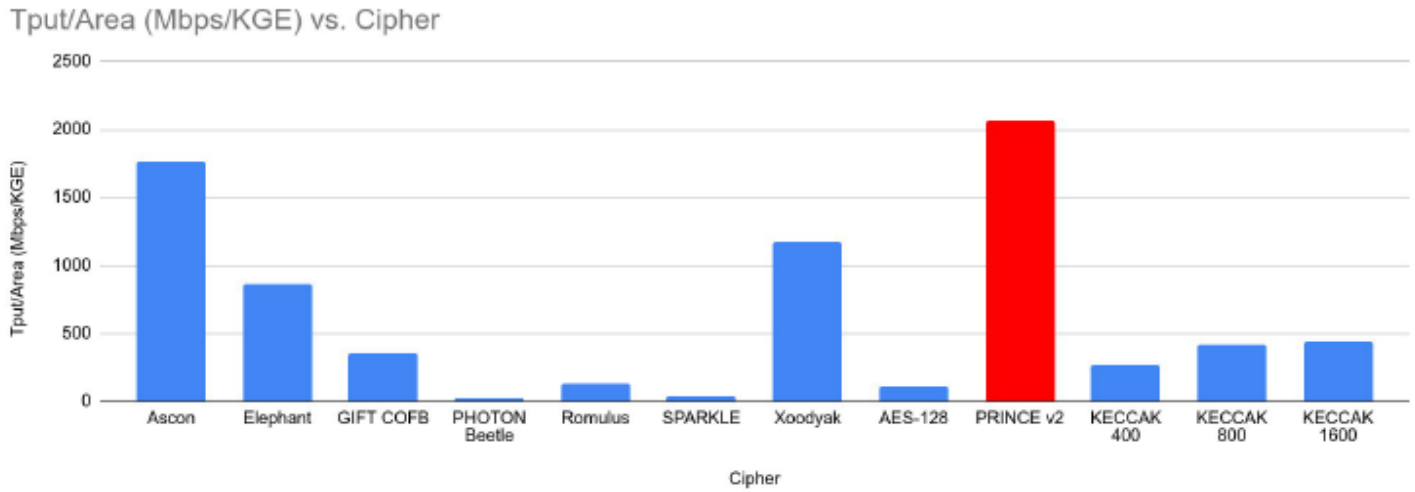


* Yalcin et al., Need for Low-latency Ciphers: A Comparative Study of NIST LWC Finalists, NIST Lightweight Crypto WS, 2022

ASCON?

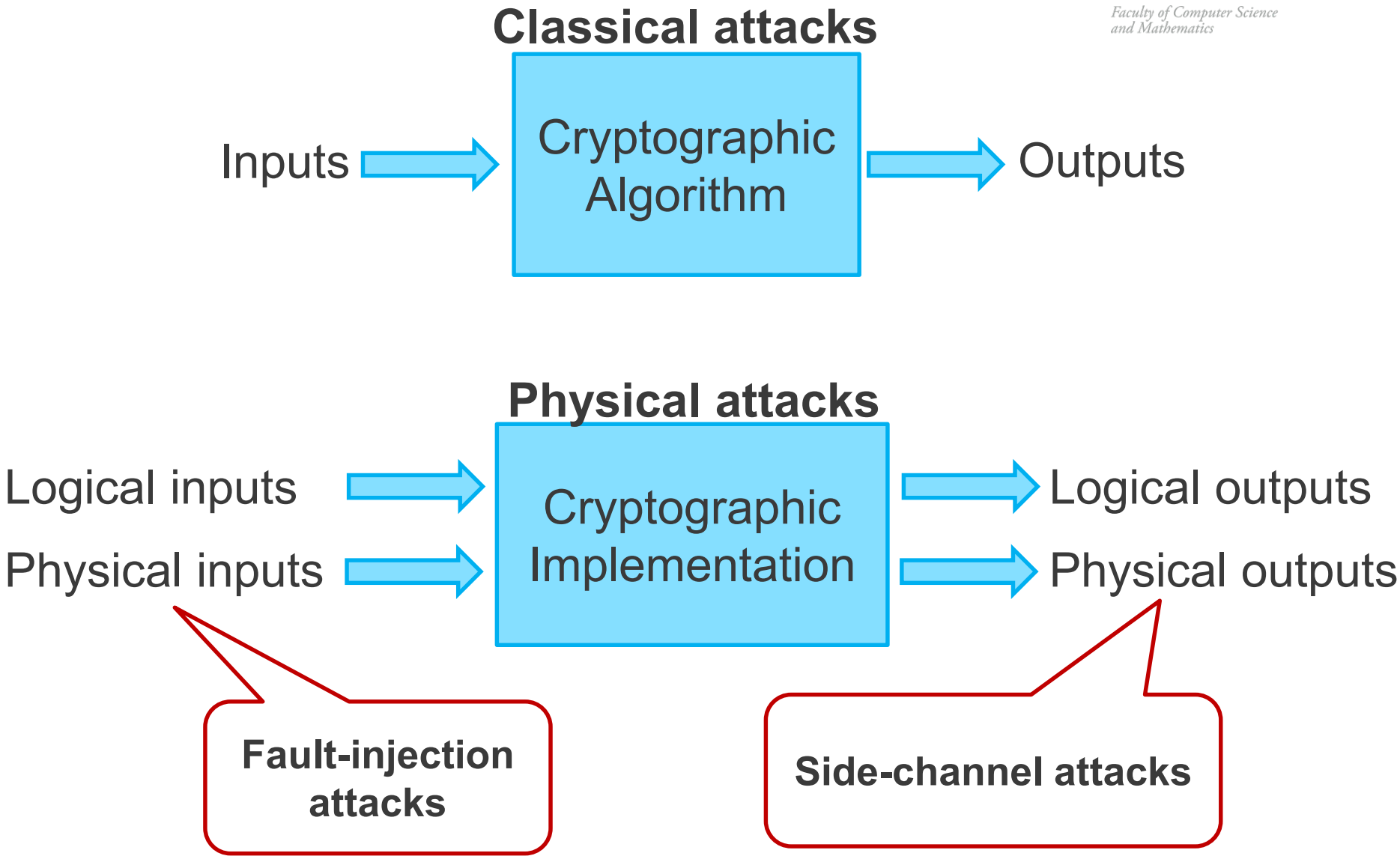
UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Throughput per Area												
	Ascon	Elephant	GIFT COFB	PHOTON Beetle	Romulus	SPARKLE	Xoodyak	AES-128	**PRINCE v2**	KECCAK 400	KECCAK 800	KECCAK 1600
Tput/Area (Mbps/KGE)	1763	864.54	355.82	28.949	128.42	33.339	1176	106.80	**2061**	269.09	421.64	435.28

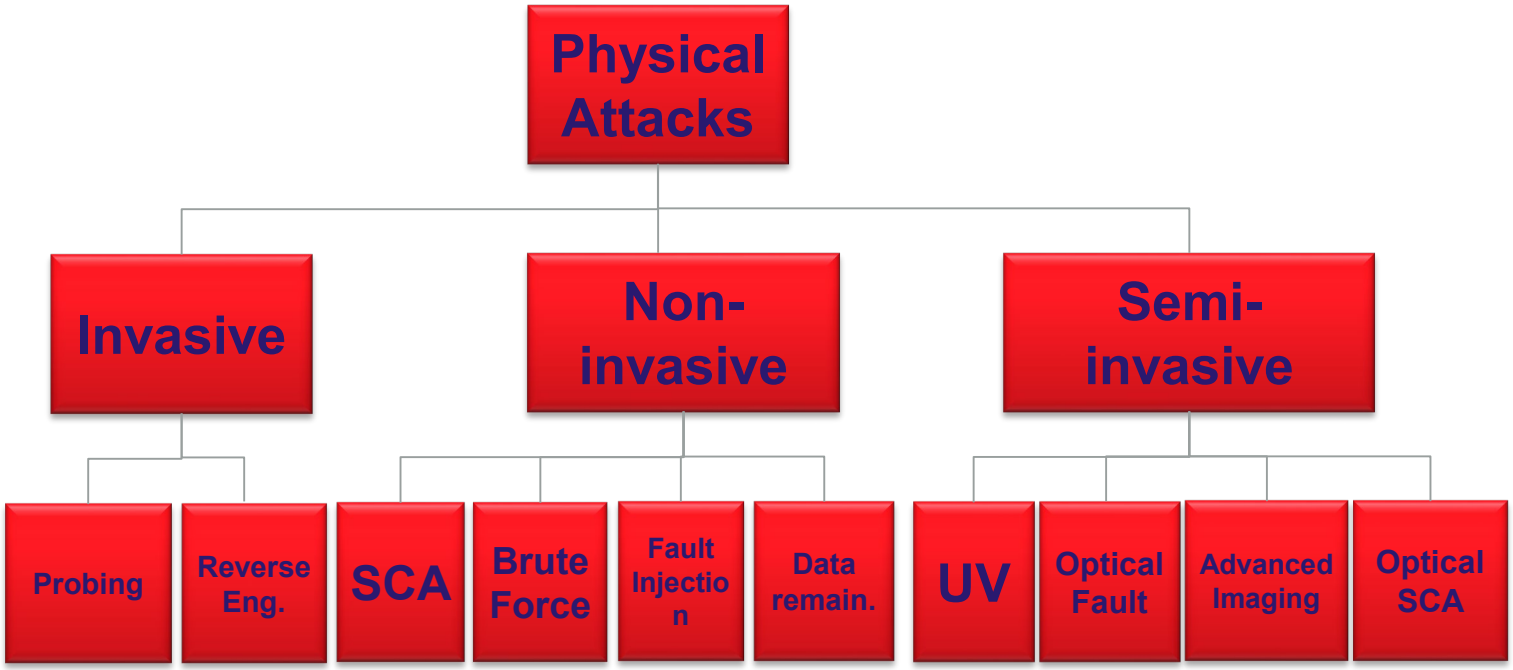


* Yalcin et al., Need for Low-latency Ciphers: A Comparative Study of NIST LWC Finalists, NIST Lightweight Crypto WS, 2022

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics



Reality: Physical Attacks



UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

SCA-resistant "Threshold Implementation" of PRINCE

Bozilov et al (KU Leuven) at LWC Workshop 2016

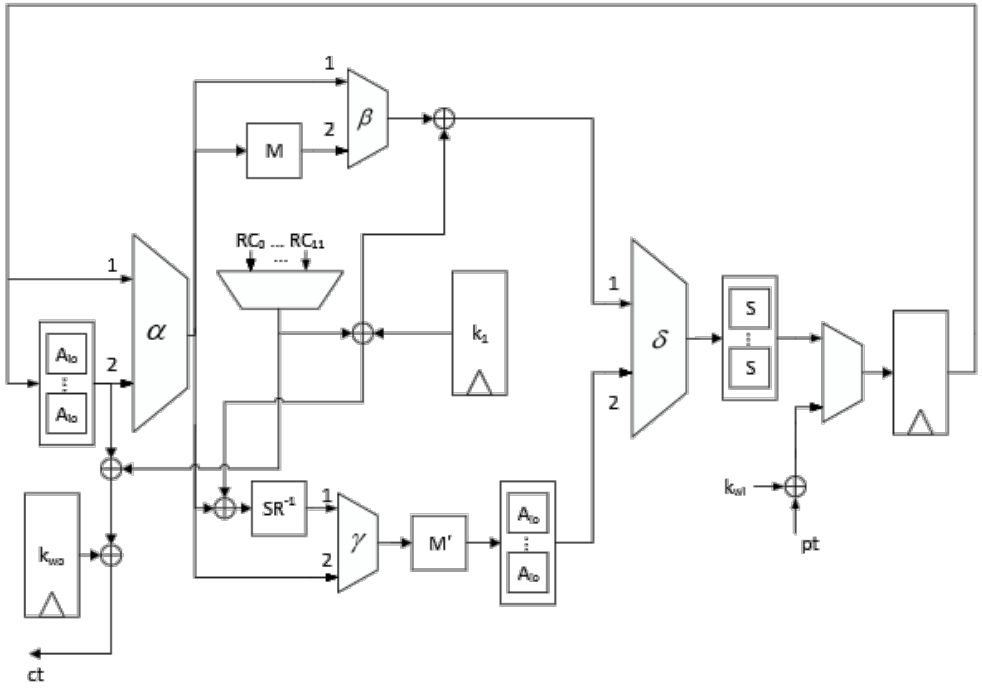
Threshold Implementations of PRINCE: The Cost of Physical Security

Abstract. Threshold implementations have recently emerged as one of the most popular masking countermeasures for hardware implementations of cryptographic primitives. In the original version of TI, the number of input shares was dependant on both security order d and algebraic degree of a function t, namely $td + 1$. At CRYPTO 2015 Reparaz et al. presented a way to perform d-th order secure implementation using $d + 1$ shares. Here we analyze $d + 1$ and $td + 1$ TI versions for first and second order secure implementations of the PRINCE block cipher. We compare a plain round-based implementation of PRINCE with its secured versions and we report hardware figures to indicate the overhead introduced by adding a side channel protection.

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

SCA-resistant "Threshold Implementation" of PRINCE

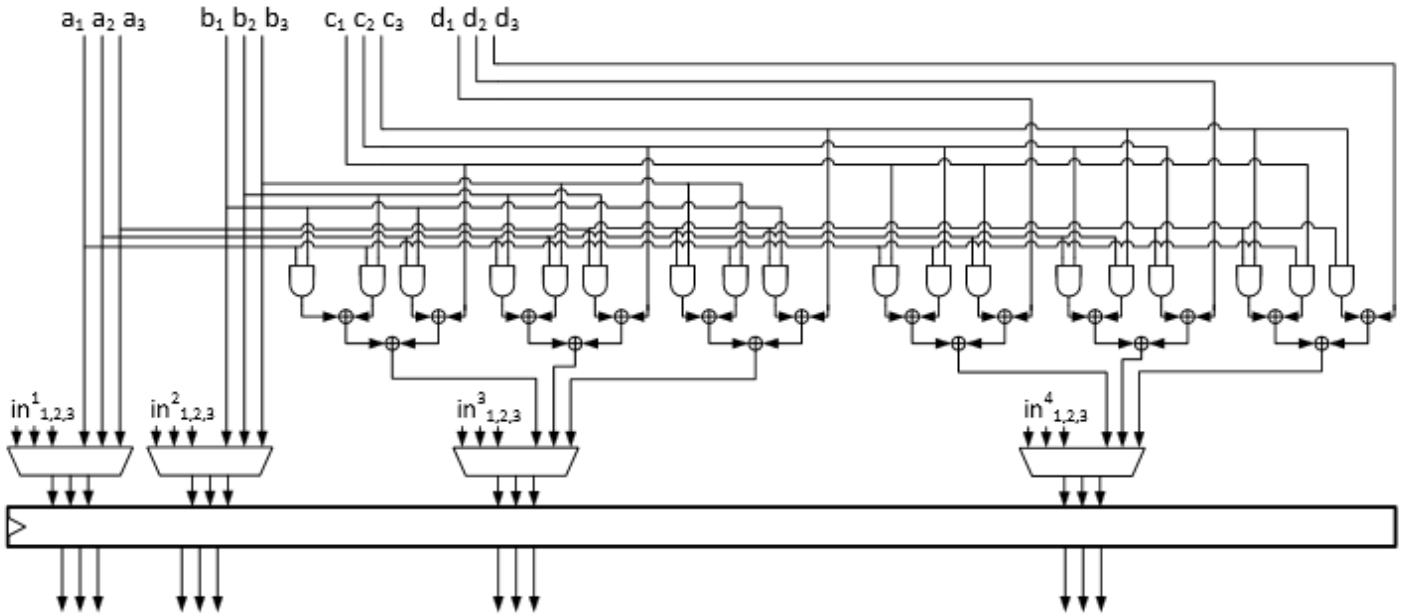
- Applied on PRINCE Sbox: Algebraic degree 3, Class Q_{294}
- Unprotected, round-based PRINCE



UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

SCA-resistant "Threshold Implementation" of PRINCE

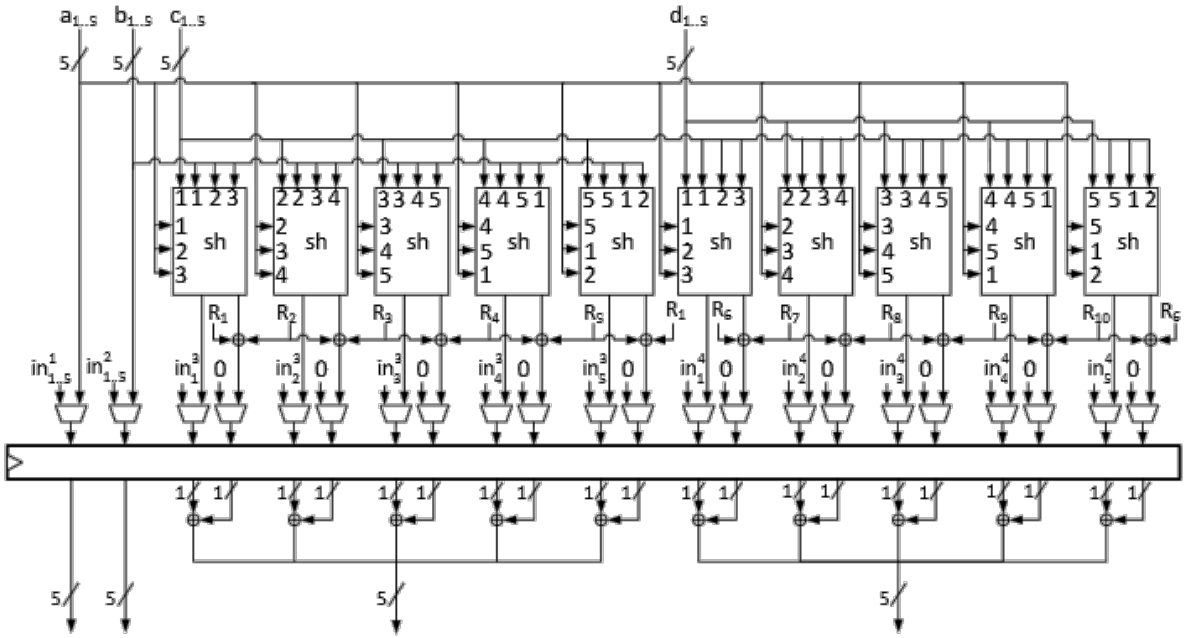
- Class Q_{294} sharing, first-order secure, 3 by 3 sharing
- No re-masking, sharing is uniform



UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

SCA-resistant "Threshold Implementation" of PRINCE

- Class Q_{294} sharing, second-order secure, 5 by 10 sharing
- Re-masking applied



UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

SCA-resistant "Threshold Implementation" of PRINCE

Results

PRINCE-128 (round-based implementation) unprotected

Technology	Area (GE)
ASIC, 90nm	3589

PRINCE-128 (round-based implementation) 1st-order secure

Technology	Area (GE)
ASIC, 90nm	11958

PRINCE-128 (round-based implementation) 2nd-order secure

Technology	Area (GE)
ASIC, 90nm	21879

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

- Development of novel resource-efficient ciphers
  - Both SCA and fault attacks in mind
  - With the cost of randomness in mind
    - Needed for countermeasures
  - Still more on low-latency!

UNIVERSITY OF PASSAU
Faculty of Computer Science and Mathematics

Thanks for listening!

Any questions?

(elif.kavun@uni-passau.de)