What the Fork: Implementation Aspects of the Forkcipher Primitive

NIST LWC Workshop 2019

Antoon Purnal[1] and **Elena Andreeva**[2]
(joint work with Arnab Roy[3], Damian Vizár[4])

imec-COSIC KU Leuven, Belgium[1]

Technical University of Denmark, Denmark[2]

University of Bristol, UK[3]

CSEM SA, Switzerland[4]

ForkAE

ForkAE

ForkAE is a NIST second round candidate

Designers: Elena Andreeva,Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, Damian Vizár

NEWS: ForkAE is accepted at ASIACRYPT 2019
full version: ePrint report 2019/1004
Optimized SW, HW, SC implementations, new modes coming!

https://www.esat.kuleuven.be/cosic/forkae/

We will announce cryptanalysis challenges and PRIZES for new ForkAE cryptanalysis/security results: to be handed at **SILC** - Security and Implementation of Lightweight Cryptography - EC2020 collocated Workshop, May 10, 2020

https://www.esat.kuleuven.be/cosic/events/silc2020/

ForkAE Project

Mar, 2017	Sept, 2017	2017	Aug 27, 2018	Feb 25, 2019
NIST IR 8114, Report on LW Cryptography	ForkAE team started design work	*ECRYPT-CSA Report* *"The performance target is wrong... increasingly common scenario is that an authenticated cipher is applied to many small messages ... The challenge here is to* ***minimize overhead****."*	NIST Requirements and Evaluation Criteria for LW Cryptography *"...optimized to be efficient for* ***short messages (e.g., as short as 8 bytes)****"*	NIST LW Submission deadline

Short messages

Numerous examples

- ✓ **CAN-FD automotive protocol** by 2020
- ✓ Bluetooth, SigFox, LoraWan and ZigBee protocols
- ✓ **Narrowband IoT** (NB-IoT) applications
  - ☐ smart sensors
  - ☐ traffic lights
  - ☐ smart parking
  - ☐ smart anything
- ✓ **Health applications**
- ✓ **Industrial control systems**

Most cryptosystems optimized for long messages

Design Goals

Secure

- ✓ Well-analysed: based on SKINNY
- ✓ Provably secure: PAEF and SAEF

Efficient

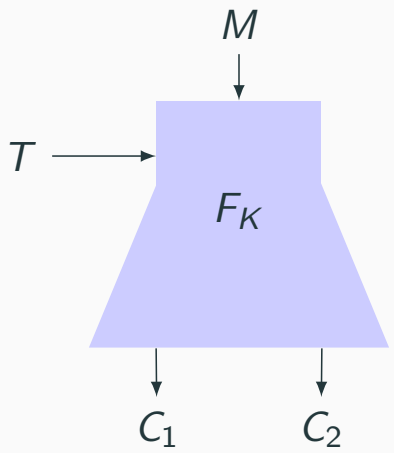
- ✓ **Excellent performance for small messages**
- ✓ Excellent throughput per area in HW
- ✓ Inherits LW implementation features of SKINNY
- ✓ Multiple trade-offs in speed-resource design space

Flexible

key size: 128 bits and variable block, nonce, tag sizes

ForkAE Design Idea

$$F : \mathcal{K} \times \mathcal{M} \times \mathcal{T} \mapsto \mathcal{C}_1 \times \mathcal{C}_2 \text{ with } |\mathcal{M}| = |\mathcal{C}_1| = |\mathcal{C}_2|$$

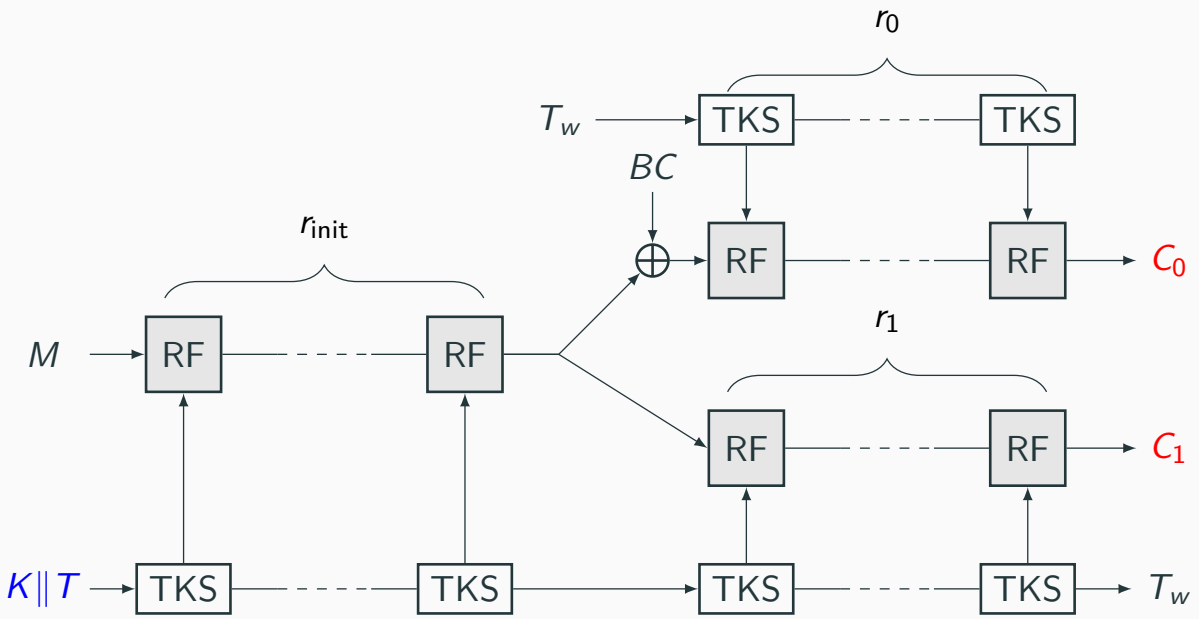


Primitive F	$\lvert M\rvert$	$\lvert T\rvert$	$\lvert T\rvert + \lvert K\rvert$
ForkSkinny-64-192	64	64	192
ForkSkinny-128-192	128	64	192
ForkSkinny-128-256	128	128	256
ForkSkinny-128-288	128	128	288

Minimizes overhead

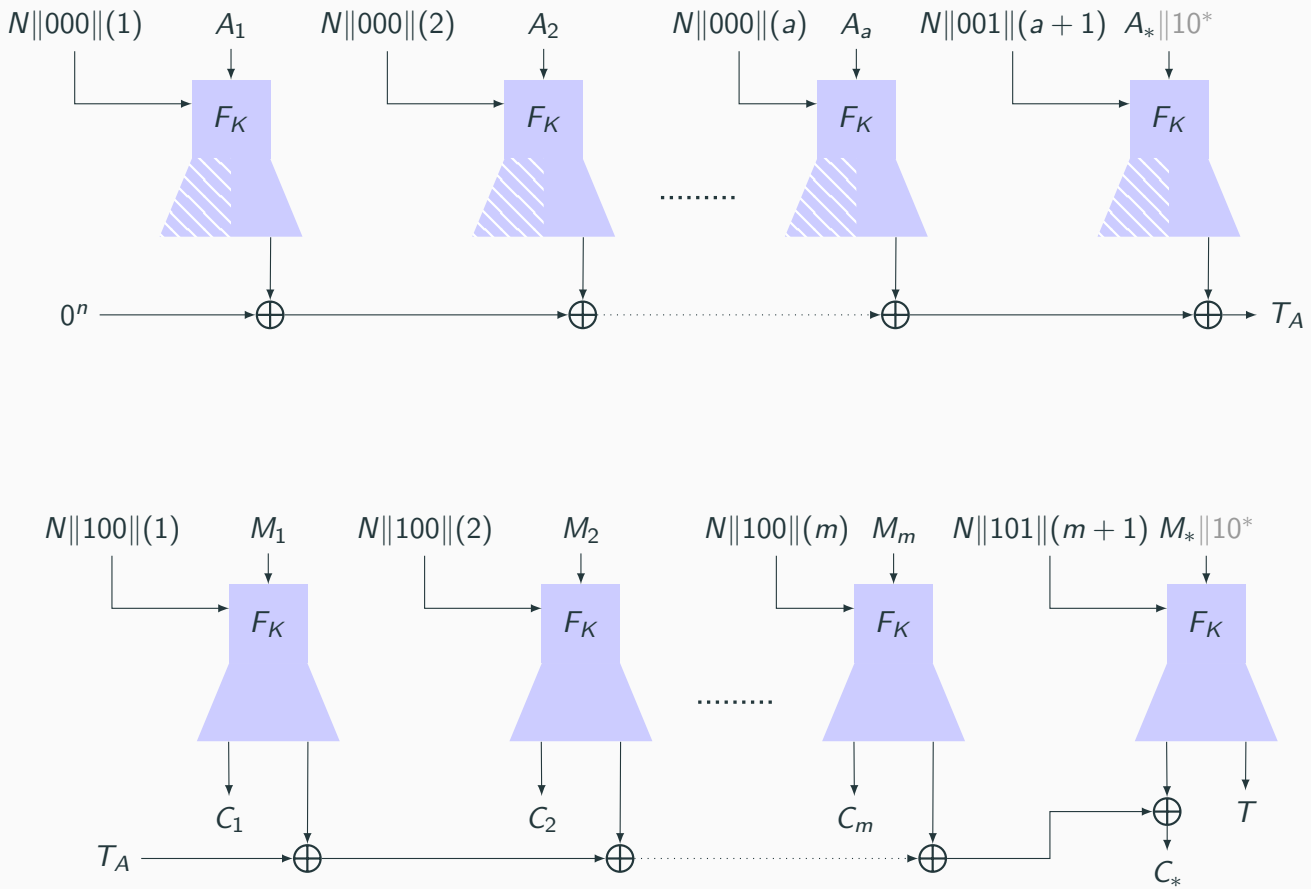
Single ForkSkinny call both authenticates and encrypts M

Forkcipher Primitive

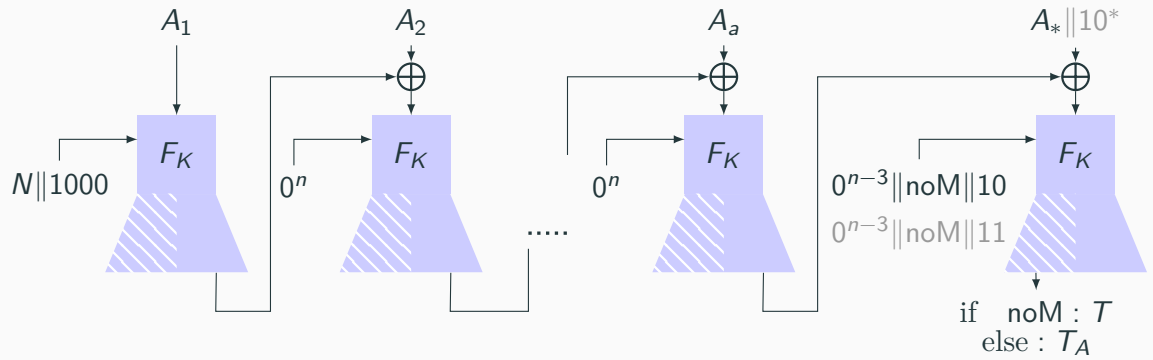


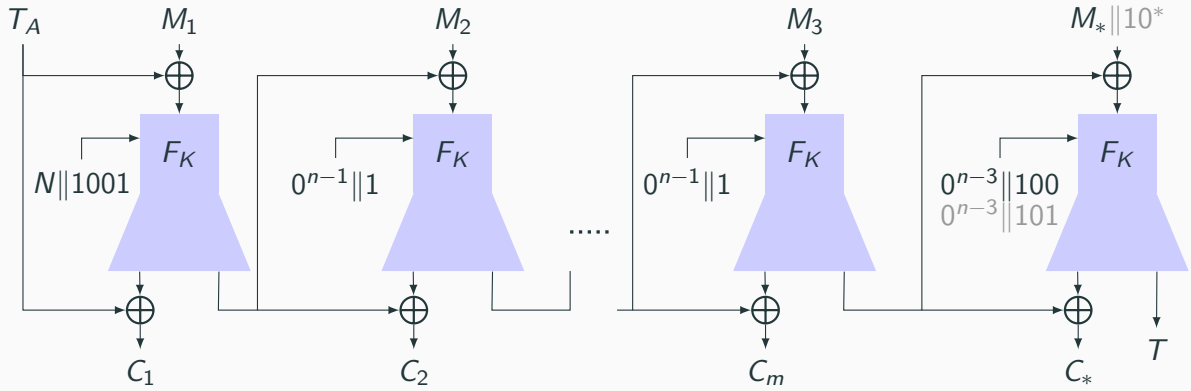
Primitive	block	tweak	tweakey	r_{init}	r_0	r_1
ForkSkinny-64-192	64	64	192	17	23	23
ForkSkinny-128-192	128	64	192	21	27	27
ForkSkinny-128-256	128	128	256	21	27	27
ForkSkinny-128-288	128	128	288	25	31	31

Mode: PAEF


$N\|000\|(1)$
A_1
$N\|000\|(2)$
A_2
$N\|000\|(a)$
A_a
$N\|001\|(a+1)$
$A_*\|10^*$
F_K
0^n
T_A
$N\|100\|(1)$
M_1
$N\|100\|(2)$
M_2
$N\|100\|(m)$
M_m
$N\|101\|(m+1)$
$M_*\|10^*$
C_1
C_2
C_m
C_*
T

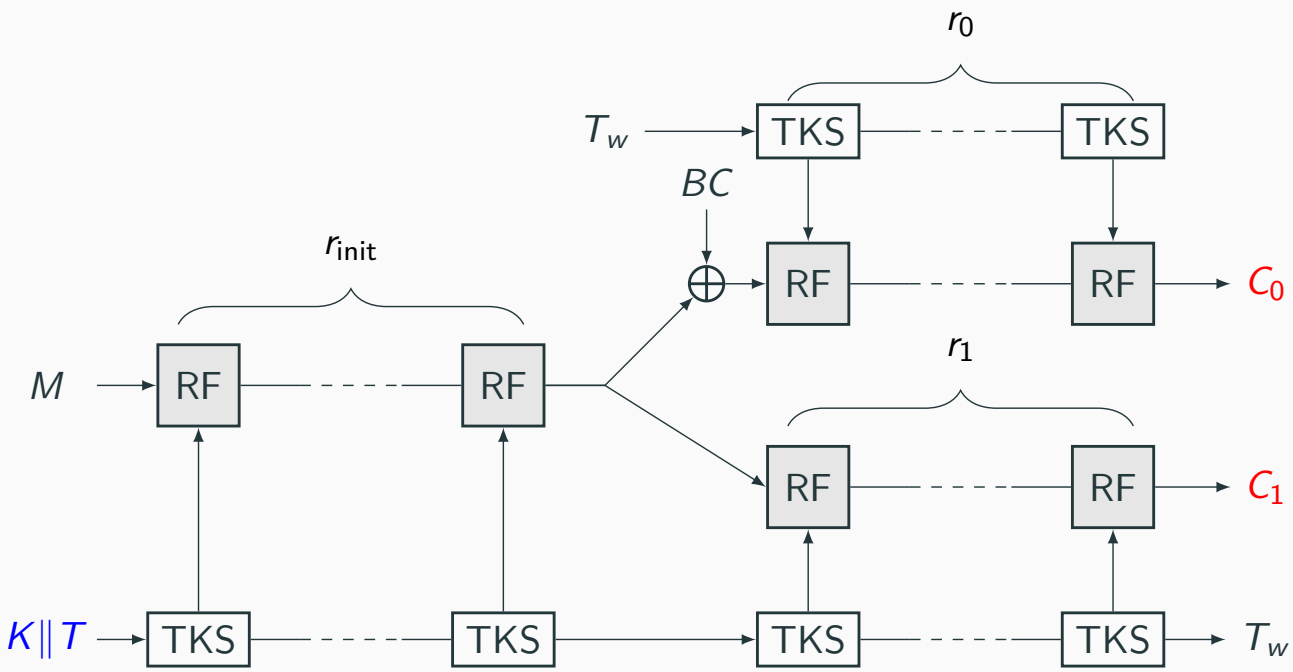
Mode: SAEF


A_1
A_2
A_a
$A_*\|10^*$
F_K
$N\|1000$
0^n
.....
$0^{n-3}\|\text{noM}\|10$
$0^{n-3}\|\text{noM}\|11$
if noM : T
else : T_A


T_A
M_1
M_2
M_3
$M_*\|10^*$
F_K
$N\|1001$
$0^{n-1}\|1$
.....
$0^{n-3}\|100$
$0^{n-3}\|101$
C_1
C_2
C_m
C_*
T

Hardware implementation aspects

Hardware implementation aspects



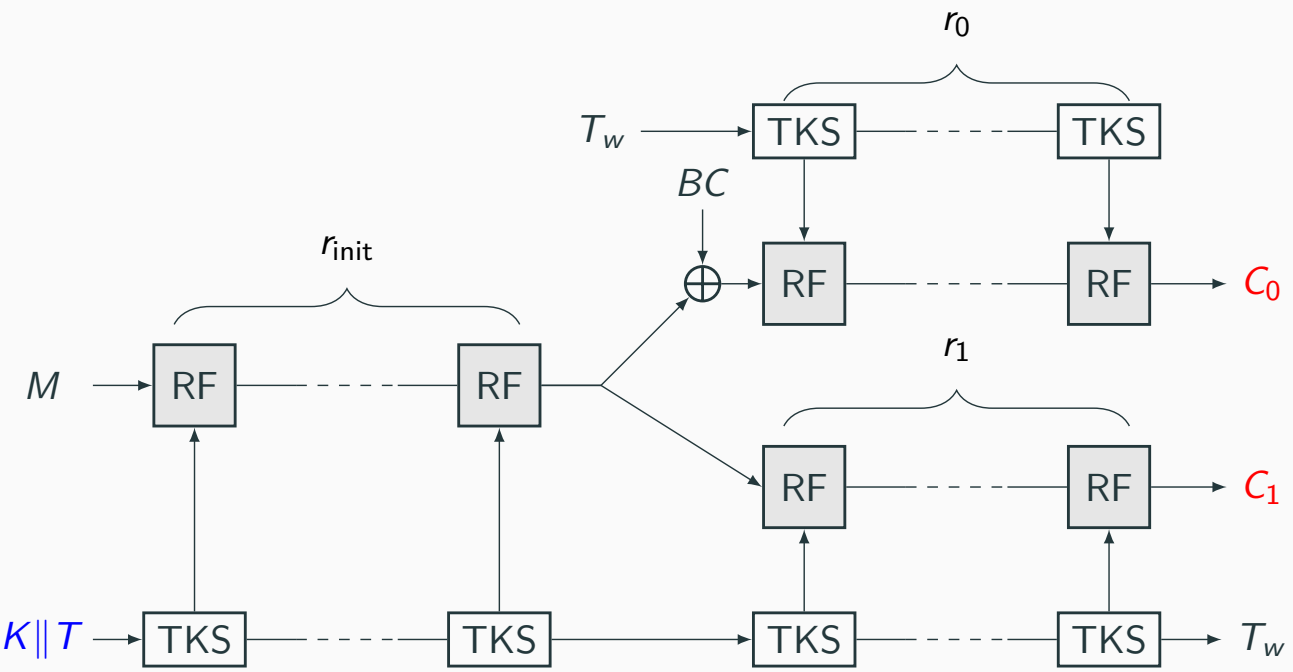
Hardware implementation aspects

Round-based implementations

Primitive-level parallelism

Stateless forkcipher?

Unrolling strategies



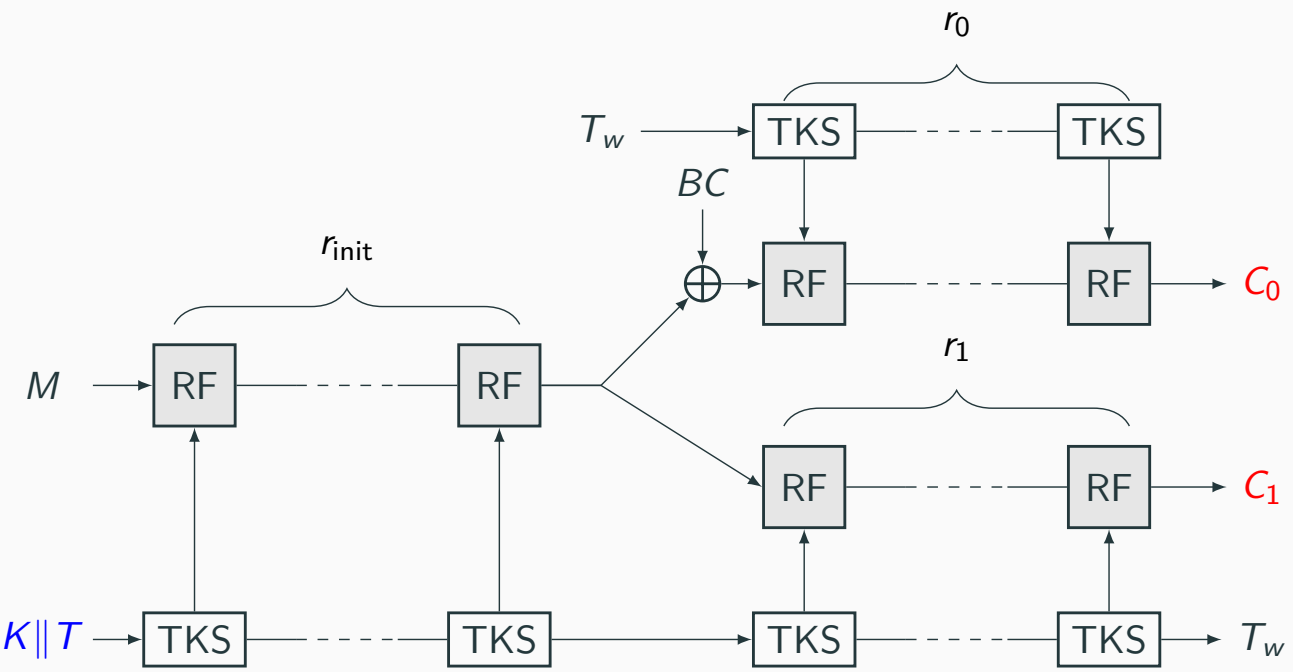
Hardware implementation aspects

Round-based implementations

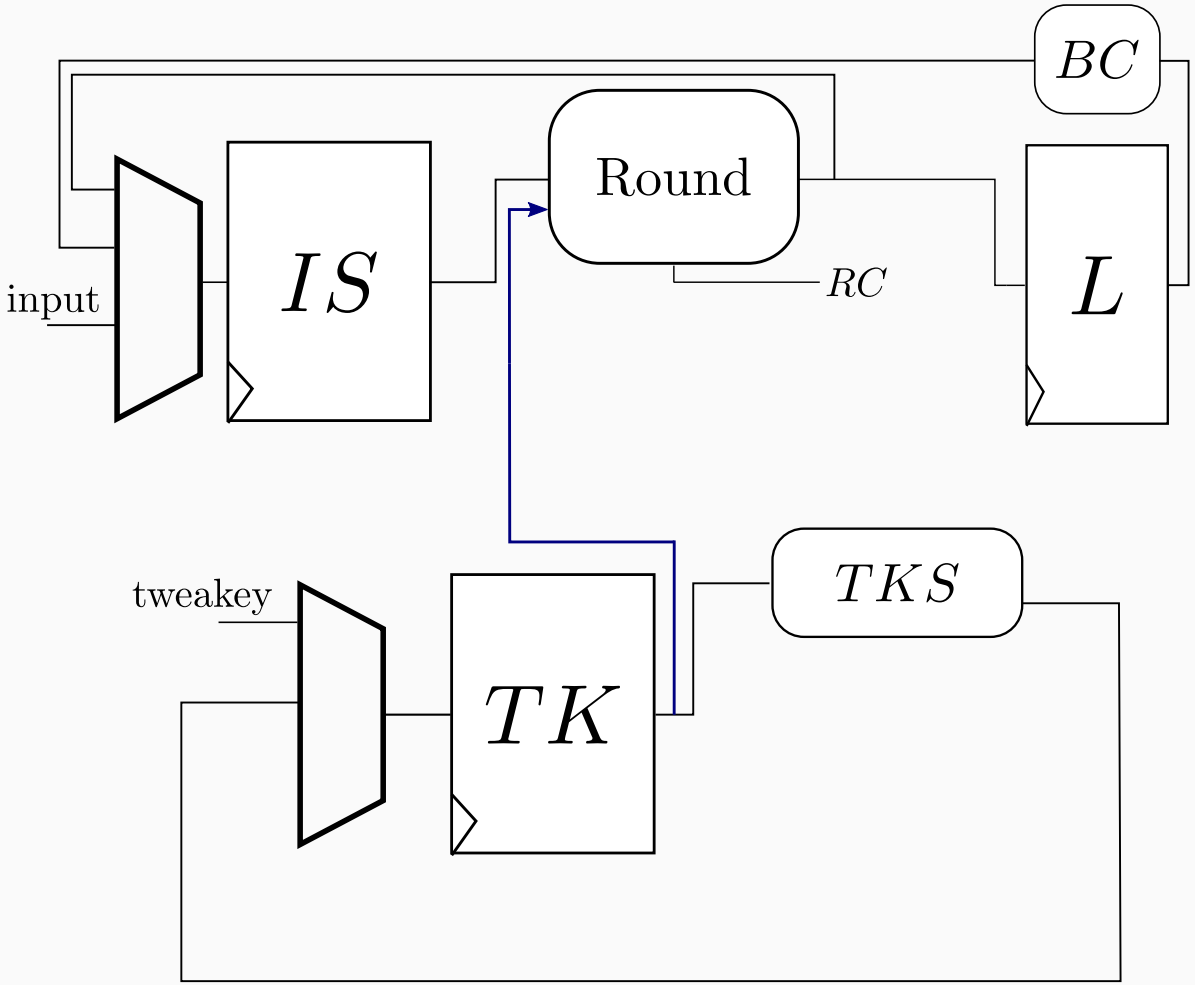
Primitive-level parallelism

Stateless forkcipher?

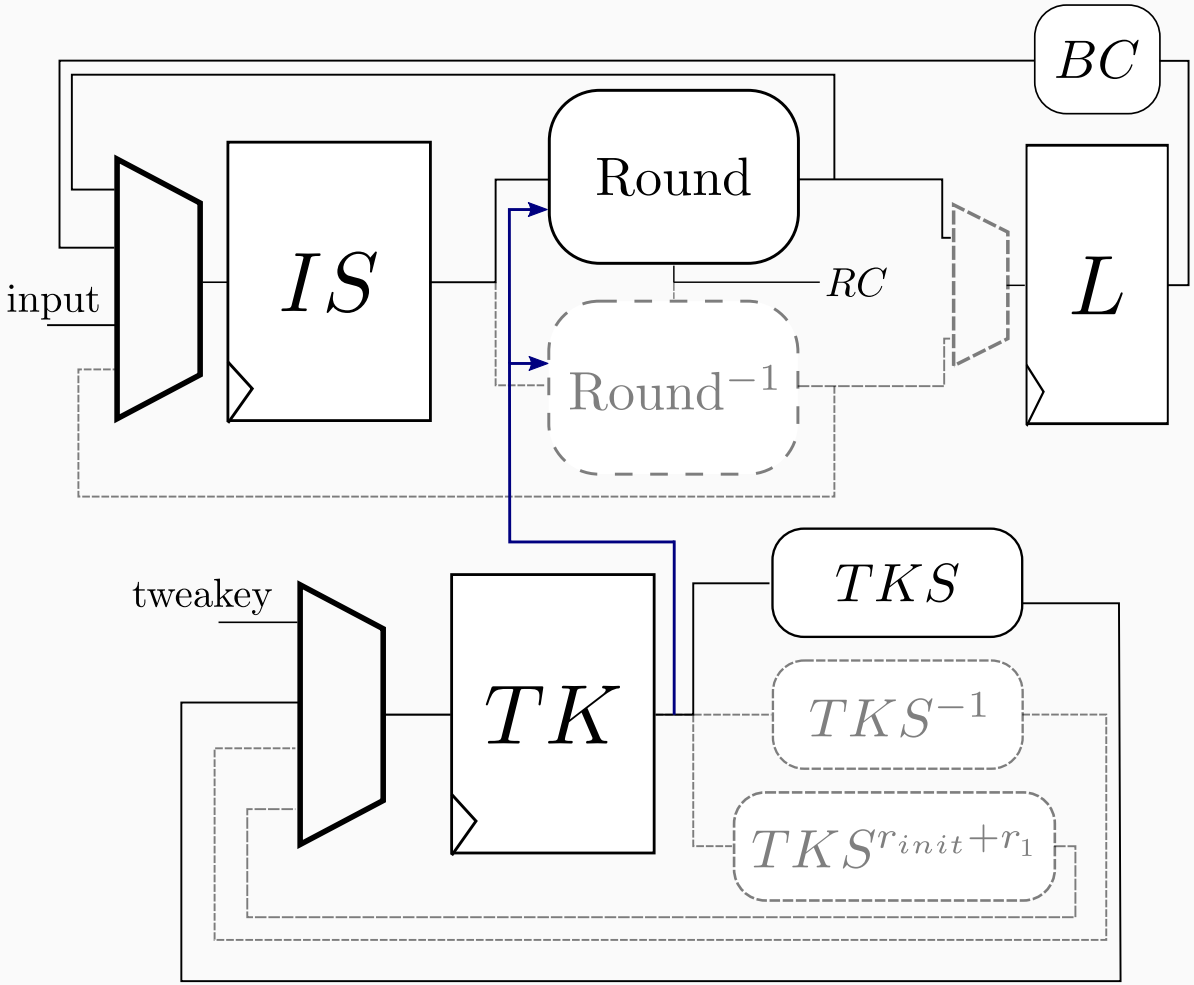
Unrolling strategies



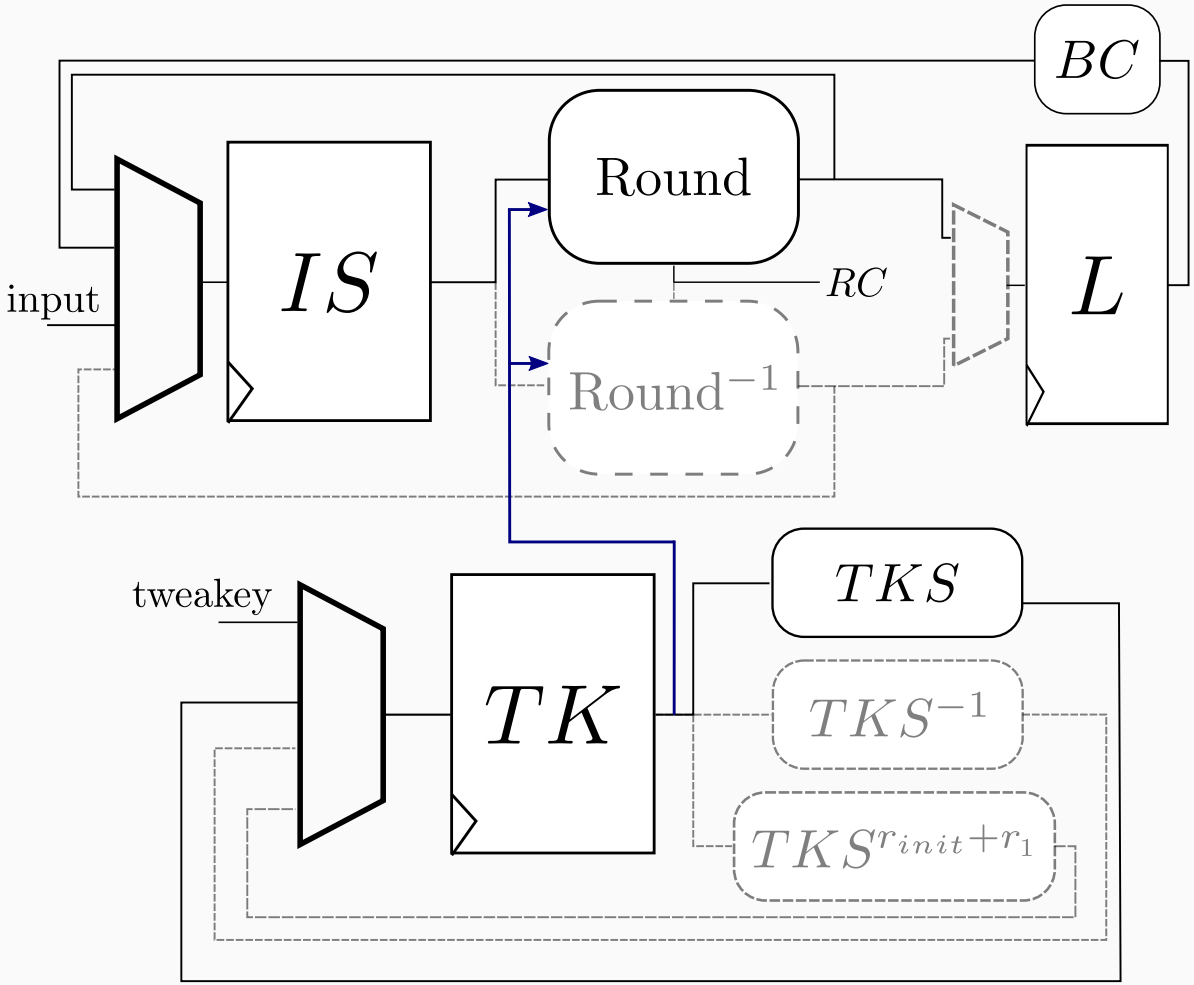
Round-based architecture



Round-based architecture



Round-based architecture



Instance-specific optimizations: key schedule, empty *TK* bits

Skinny and ForkSkinny

Area [GE]		
Primitive	Enc-only	Enc+Dec
Skinny-64-192	3003	4522
Skinny-128-256	4992	6355
Skinny-128-384	5914	8311
ForkSkinny-64-192	3692	5362
ForkSkinny-128-192	5299	7305
ForkSkinny-128-256	5842	8101
ForkSkinny-128-288	6751	9182

Maximal frequency [MHz]		
Primitive	Enc-only	Enc+Dec
Skinny-64-192	1351	1087
Skinny-128-256	1087	1020
Skinny-128-384	1020	962
ForkSkinny-64-192	1282	980
ForkSkinny-128-192	1064	877
ForkSkinny-128-256	1064	917
ForkSkinny-128-288	990	862

Synthesis: Nangate 45nm in typical operating conditions

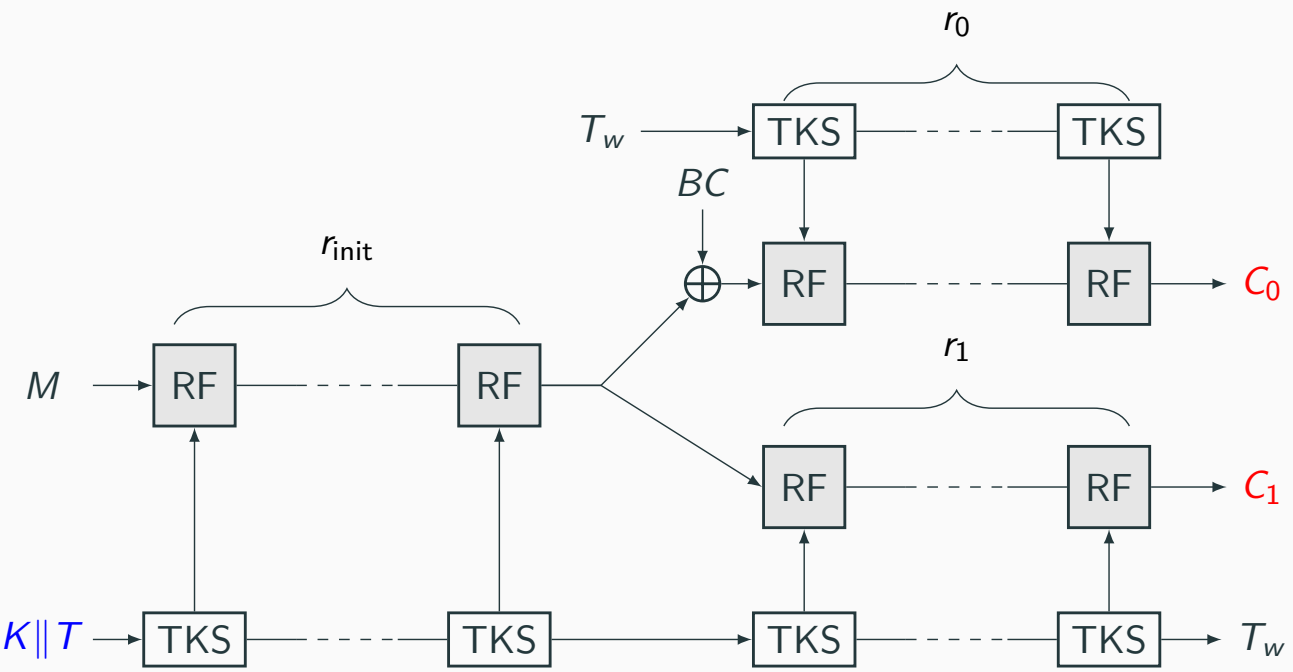
Hardware implementation aspects

Round-based implementations

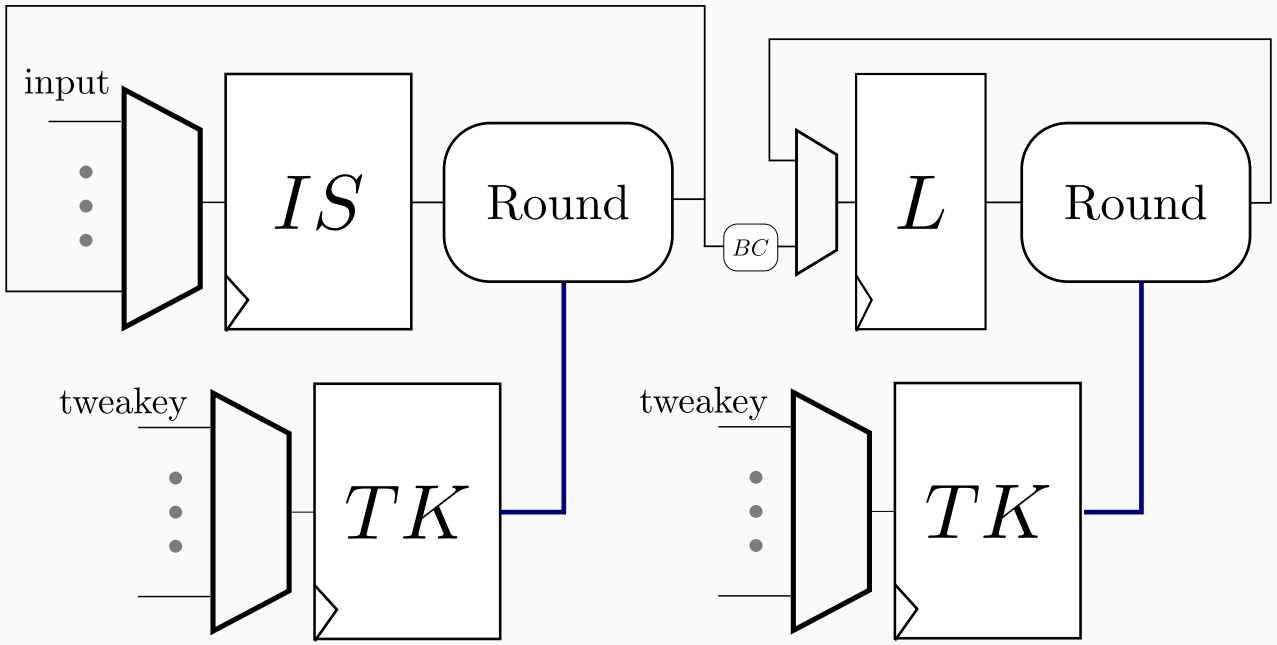
Primitive-level parallelism

Stateless forkcipher?

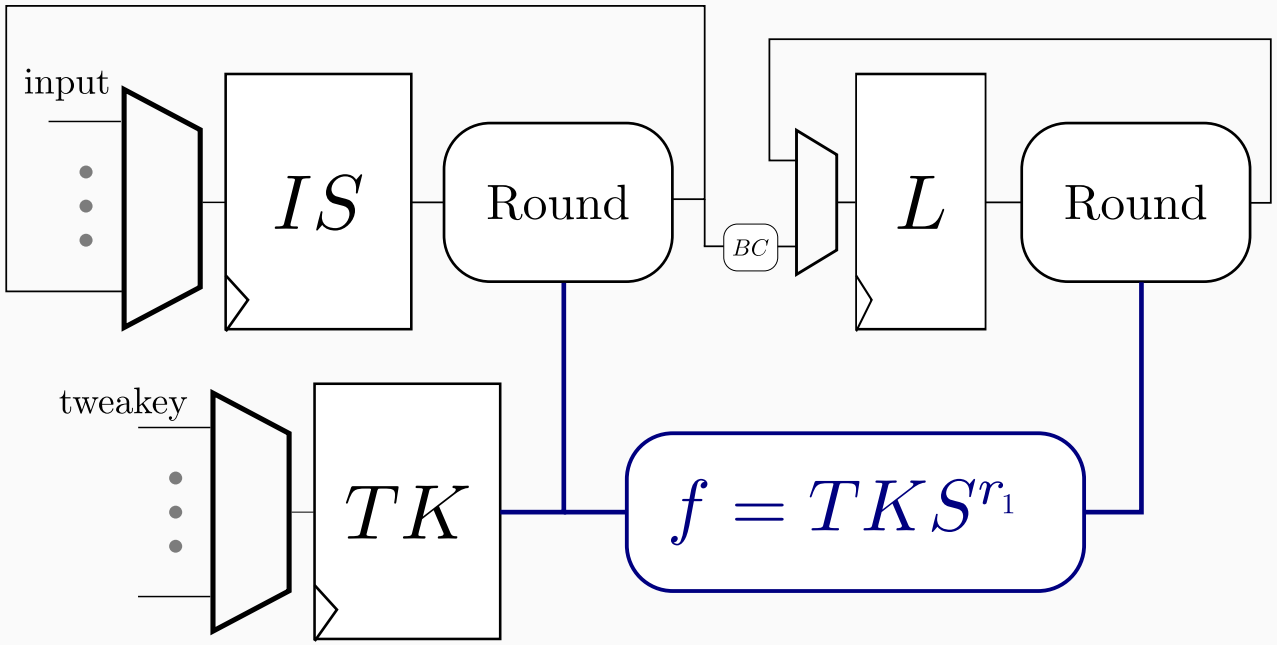
Unrolling strategies



Fast-forwarding – concept



Fast-forwarding – concept

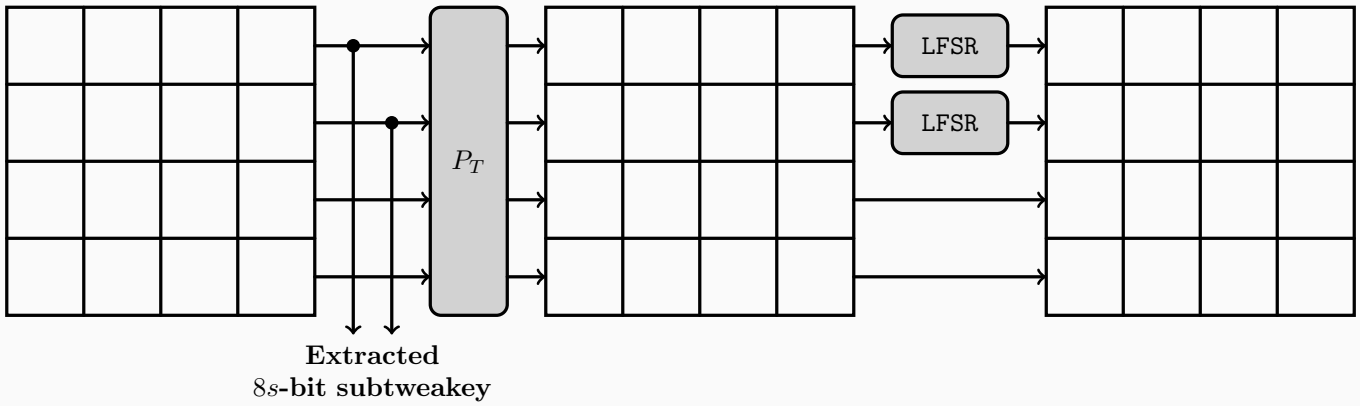


Lightweight parallelism (//)

Two forward round functions (**one** inverse for decryption)

TK only **once**

Fast-forwarding – critical path



Fast-forwarding **only slightly influences critical path**

Skinny and ForkSkinny – parallel

Area [GE]				
Primitive	Enc-only		Enc+Dec	
	Regular	Parallel	Regular	Parallel
Skinny-64-192	3003	/	4522	/
Skinny-128-256	4992	/	6355	/
Skinny-128-384	5914	/	8311	/
ForkSkinny-64-192	3692	4307	5362	6229
ForkSkinny-128-192	5299	6113	7305	8608
ForkSkinny-128-256	5842	6688	8101	9450
ForkSkinny-128-288	6751	7917	9182	10876

Maximal frequency [MHz]				
Primitive	Enc-only		Enc+Dec	
	Regular	Parallel	Regular	Parallel
Skinny-64-192	1351	/	1087	/
Skinny-128-256	1087	/	1020	/
Skinny-128-384	1020	/	962	/
ForkSkinny-64-192	1282	1253	980	952
ForkSkinny-128-192	1064	1020	877	877
ForkSkinny-128-256	1064	1020	917	884
ForkSkinny-128-288	990	962	862	820

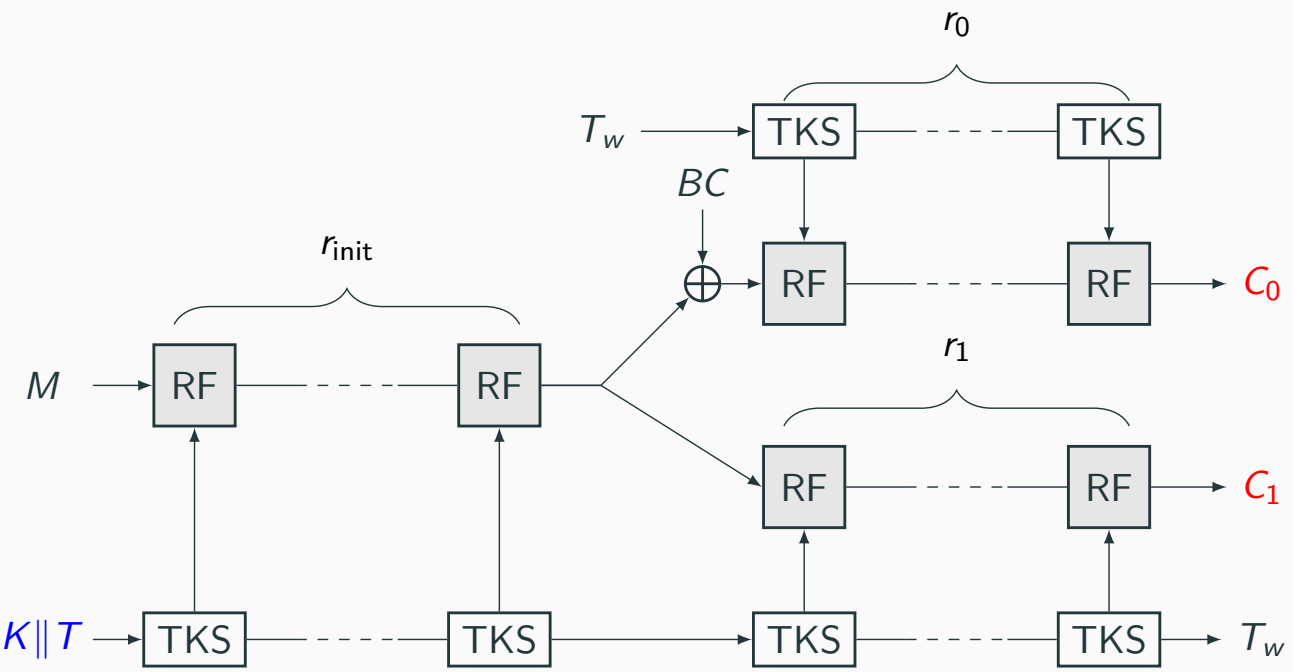
Hardware implementation aspects

Round-based implementations

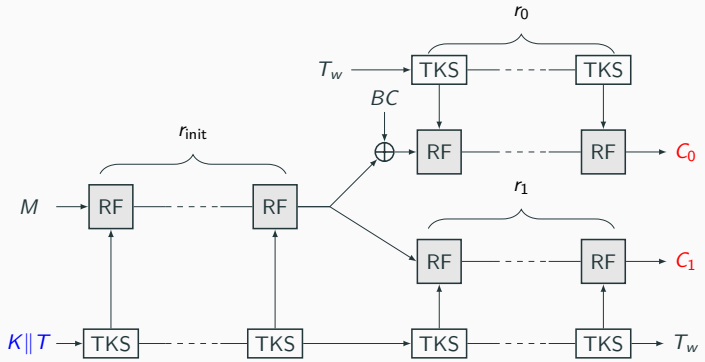
Primitive-level parallelism

Stateless forkcipher?

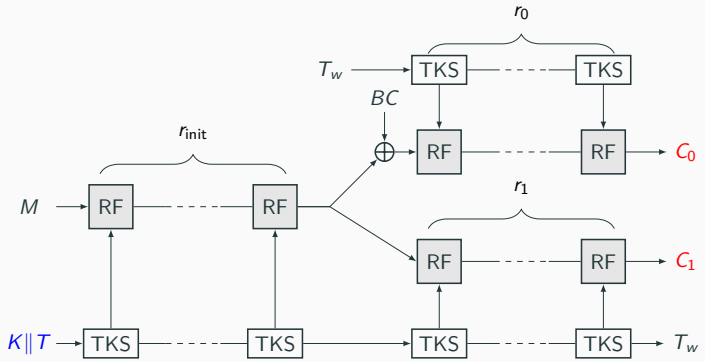
Unrolling strategies



Can be small, too



Can be small, too

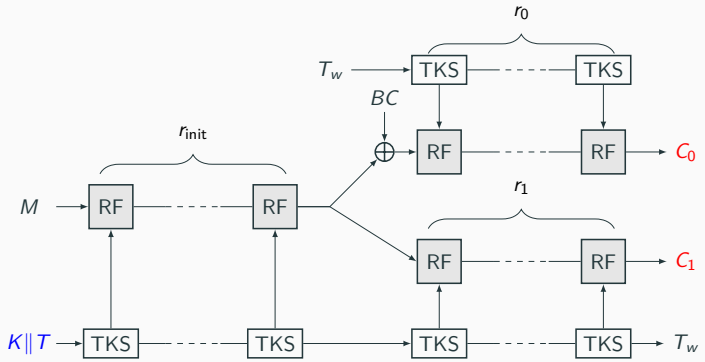


Forkcipher implementation **without storing state**

Restart

Recompute

Can be small, too



Forkcipher implementation **without storing state**

Restart

Recompute (confirmed by Balli-Banik for FORKAES [1])

[1] https://eprint.iacr.org/2019/1213

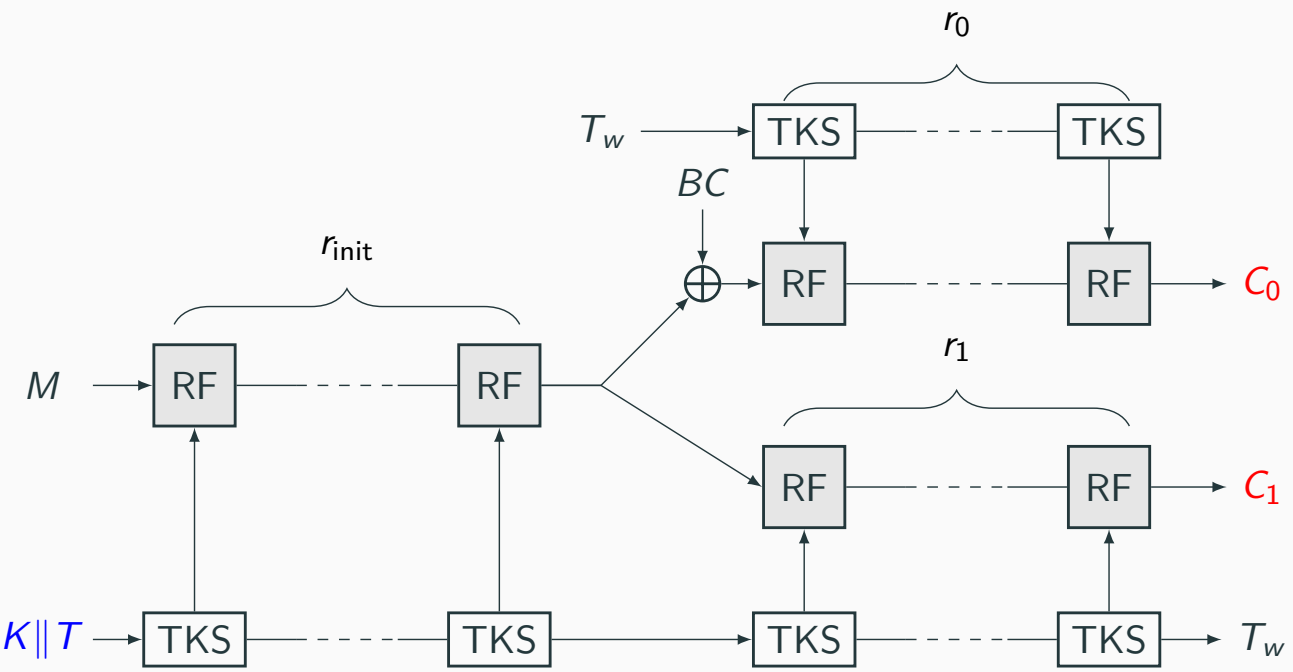
Hardware implementation aspects

Round-based implementations

Primitive-level parallelism

Stateless forkcipher?

Unrolling strategies



Unrolling strategies

Instance **(1 cycle)**	Area [GE]	Critical path [ns]
ForkSkinny-64-192	34167	26
ForkSkinny-128-192	62387	37

Unrolling strategies

Instance **(1 cycle)**	Area [GE]	Critical path [ns]
ForkSkinny-64-192	34167	26
ForkSkinny-128-192	62387	37

Instance **(3 cycles)**	Area [GE]	Critical path [ns]
ForkSkinny-64-192	16221	14
ForkSkinny-128-192	29666	20

Comparison

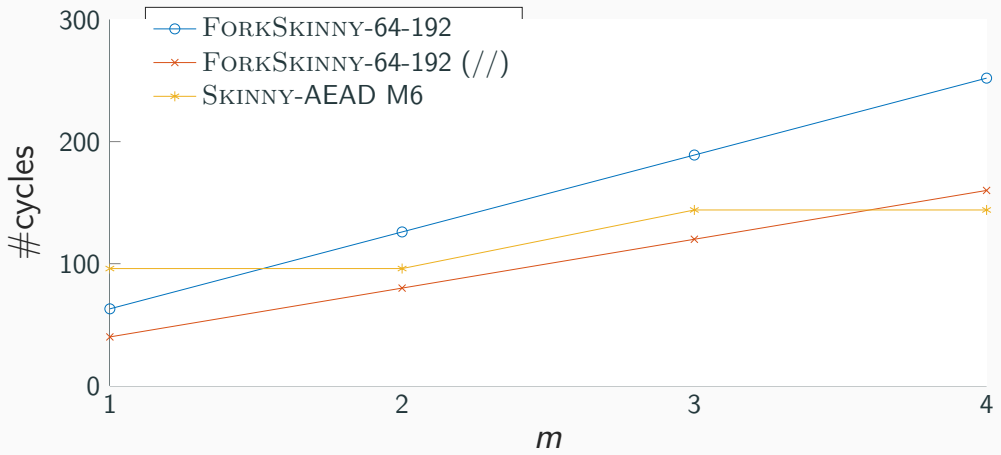
Comparison with Skinny-based designs

Implementation (round-based)	Area [GE] E-ONLY	Area [GE] ENCDEC	Number of cycles for encrypting $(a+m)$ **64-bit** blocks						
			$a=0$			$a=1$			General
			$m=1$	$m=2$	$m=3$	$m=0$	$m=1$	$m=2$	
SK-AEAD M6	8095	9458	96	96	144	48	96	**96**	$48(\lceil \frac{a}{2} \rceil + \lceil \frac{m}{2} \rceil + 1)$
PAEF-64-192	5034	6704	**63**	126	189	**40**	103	166	$40(a+1.575m)$
PAEF-64-192 (//)	5500	7422	**40**	**80**	**120**	**40**	**80**	120	$40(a+m)$

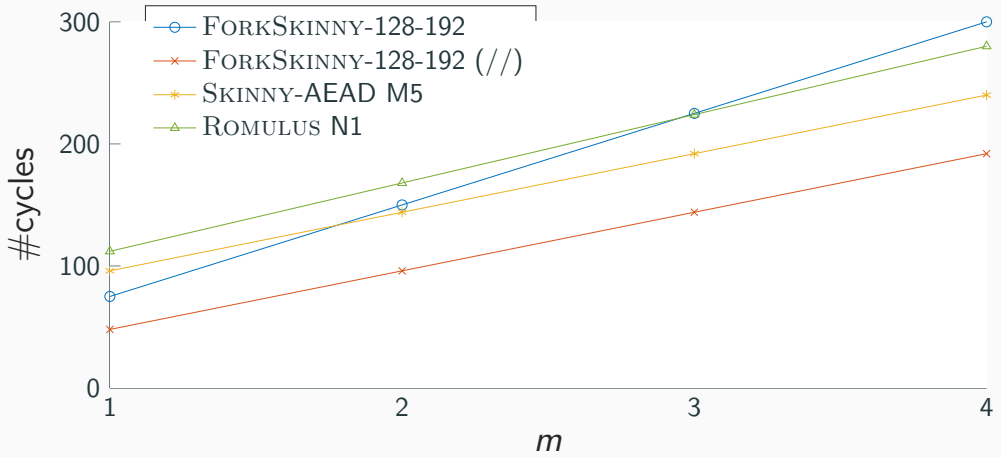
Implementation (round-based)	Area [GE] E-ONLY	Area [GE] ENCDEC	Number of cycles for encrypting $(a+m)$ **128-bit** blocks						
			$a=0$			$a=1$			General
			$m=1$	$m=2$	$m=3$	$m=0$	$m=1$	$m=2$	
ROMULUS-N3	6288	6406	96	144	192	**48**	**96**	**144**	$48(\lceil \frac{a-1}{1.75} \rceil + m + 1)$
SAEF-128-192	7197	9203	**75**	150	225	**48**	123	198	$48(a+1.562m)$
SAEF-128-256	7740	9999	**75**	150	225	**48**	123	198	$48(a+1.562m)$
SAEF-128-192 (//)	7713	10804	**48**	**96**	**144**	**48**	**96**	**144**	$48(a+m)$
SAEF-128-256 (//)	8288	11646	**48**	**96**	**144**	**48**	**96**	**144**	$48(a+m)$
SK-AEAD M5	8746	10109	96	144	192	96	144	192	$48(a+m+1)$
PAEF-128-192 (//)	8020	11112	**48**	**96**	**144**	**48**	**96**	**144**	$\mathbf{48(a+m)}$
PAEF-128-256 (//)	8745	12103	**48**	**96**	**144**	**48**	**96**	**144**	$\mathbf{48(a+m)}$
ROMULUS-N1	7018	7136	112	168	224	**56**	**112**	**168**	$56(\lceil \frac{a-1}{2} \rceil + m + 1)$
SK-AEAD M1-2	9966	12363	112	168	224	112	168	224	$56(a+m+1)$
PAEF-128-288	9274	11705	**87**	174	261	**56**	**143**	230	$56(a+1.553m)$
PAEF-128-288 (//)	10141	13697	**56**	**112**	**168**	**56**	**112**	**168**	$56(a+m)$

Hybrid area estimation: primitive synthesized, mode estimated (SFF/MUX/XOR/NAND with `7.67/2.33/2/1 GE`)

Message sizes

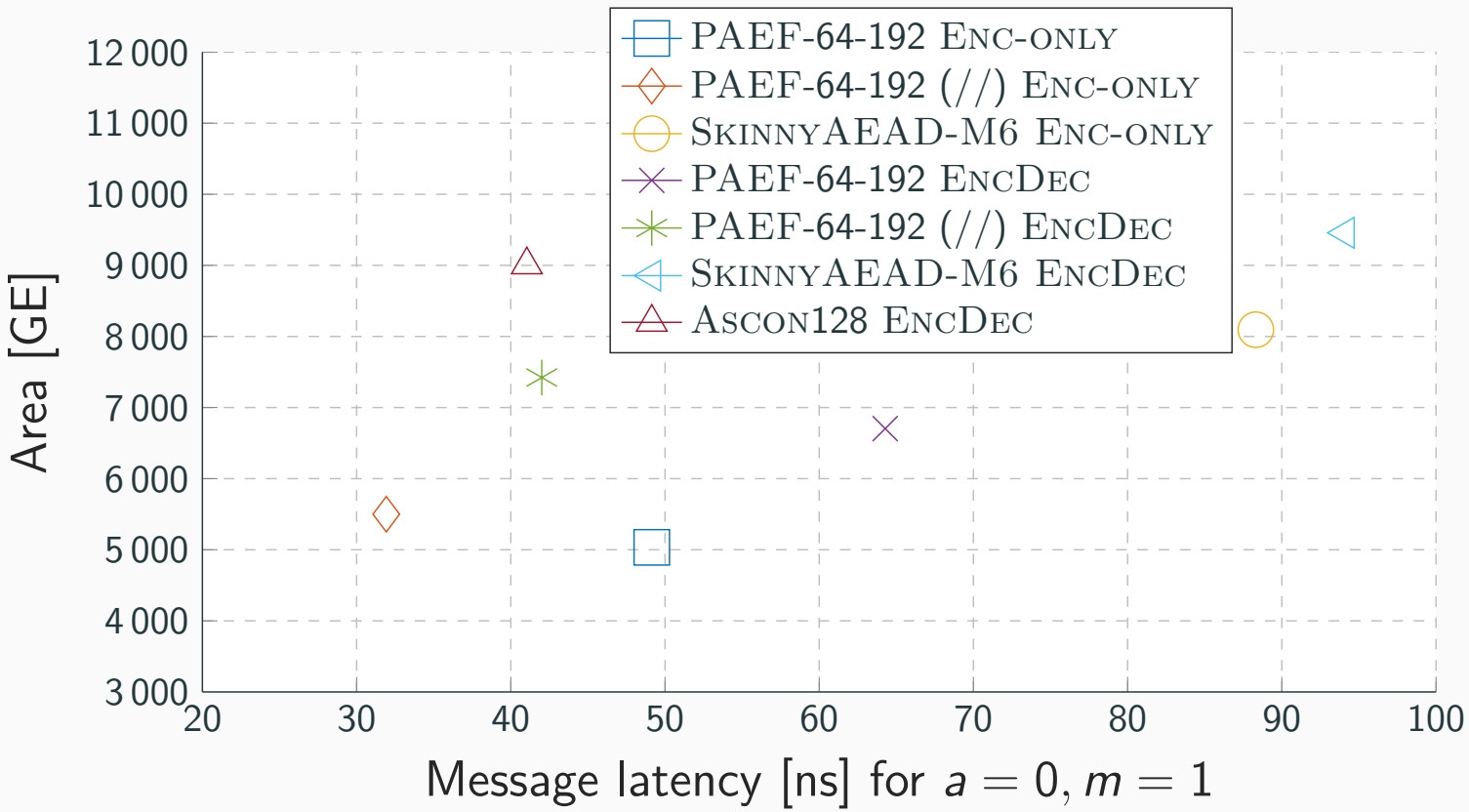


64-bit blocks



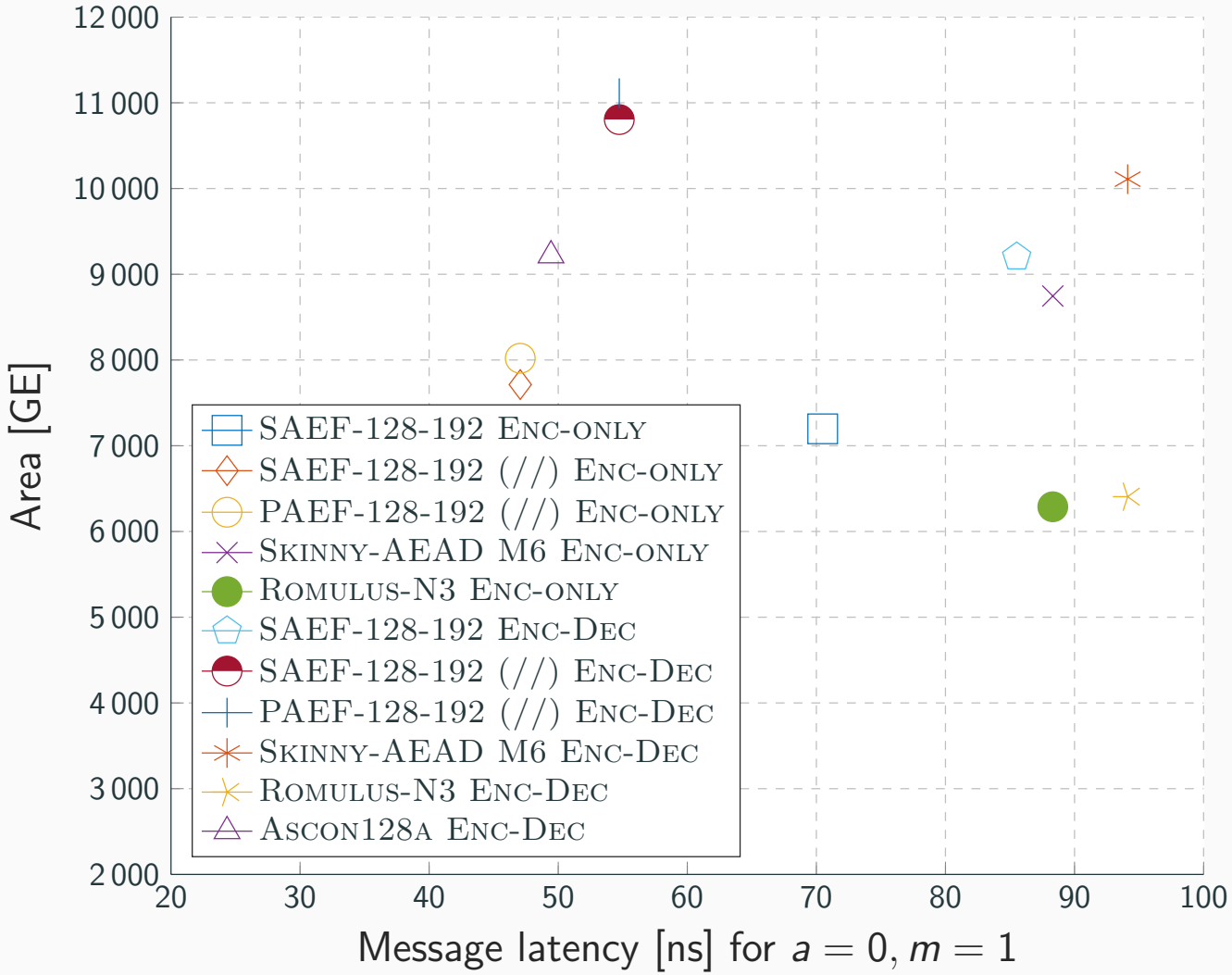
128-bit blocks

Speed-area exploration (64 bits)



64-bit blocks ($a = 0,\ m = 1$)

Speed-area exploration (128 bits)



128-bit blocks ($a = 0,\ m = 1$)

Conclusion

Conclusion

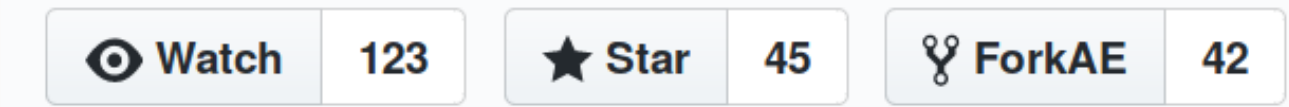
- Excellent performance on **shortest of messages**
- Novel **implementation aspects** of forkcipher
- Cheap **primitive-level parallelism**

`https://github.com/byt3bit/forkae/`

Conclusion

- Excellent performance on **shortest of messages**
- Novel **implementation aspects** of forkcipher
- Cheap **primitive-level parallelism**

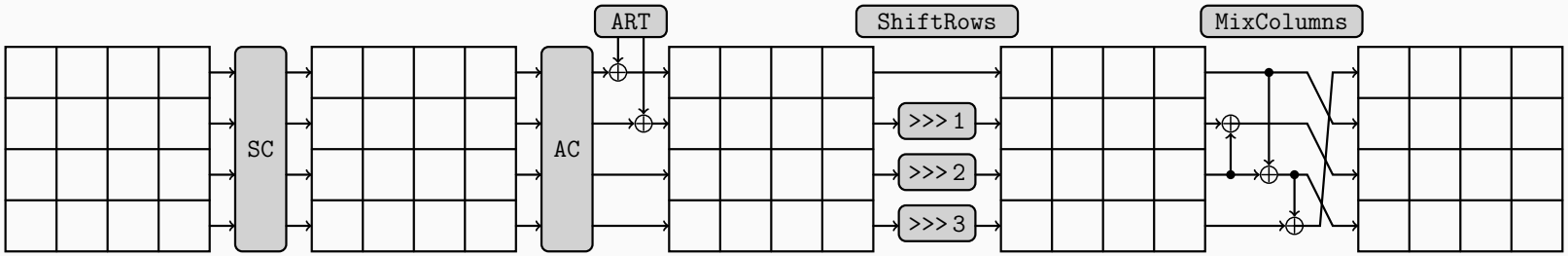
`https://github.com/byt3bit/forkae/`



Thank you!

Extra slides

Skinny round function



SKINNY round function