国立大学法人
電気通信大学

RISC-V-based System-on-Chip (SoC) Fully Equipped with Cryptographic Accelerators for Transport Layer Security (TLS) 1.3

Trong-Thuc HOANG and Cong-Kha PHAM

Tokyo, 2nd June, 2022

1. Introduction
2. RISC-V-based SoC
3. TLS 1.3 Cryptographic Accelerators
4. Experimental Result
5. Conclusion

1. Introduction
2. RISC-V-based SoC
3. TLS 1.3 Cryptographic Accelerators
4. Experimental Result
5. Conclusion

Introduction (1/2)

Transport Layer Security (TLS)

- The successor of Secure Sockets Layer (SSL)
- Among the most commonly used security protocols over the Internet
- The latest proposal is v1.3 [1] (2018)
- Is equipped with many current robust cryptography algorithms

↓

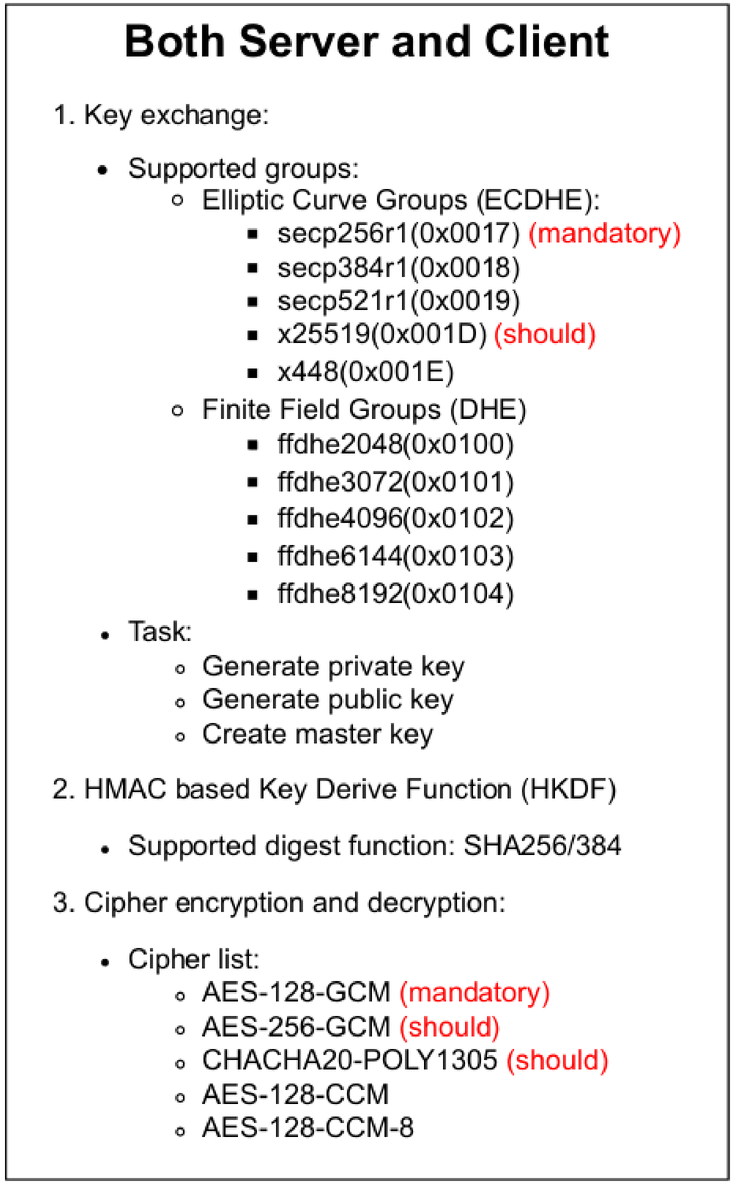
- TLS 1.3 workflow is not fixed yet; many cryptographic functions are left as optional

↓

This work aiming for developing TLS-1.3-oriented cryptographic accelerators
for RISC-V System-on-Chip (SoC)

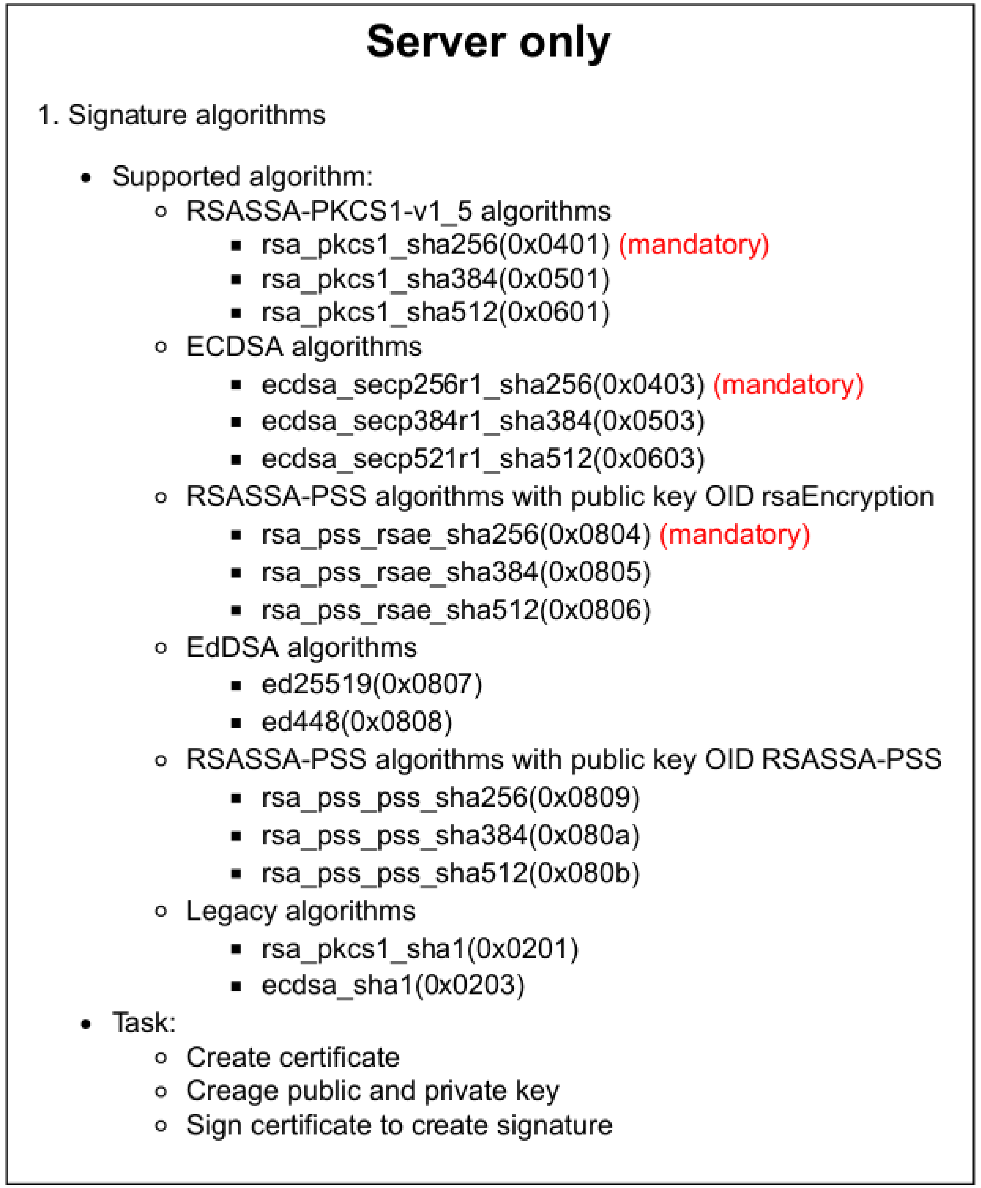
Introduction (2/2)

List of supported algorithms used in TLS-1.3



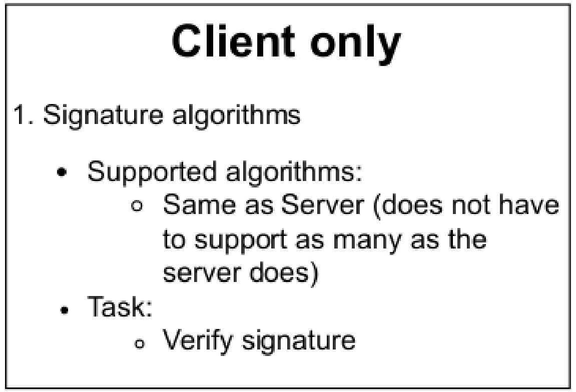
Both Server and Client

1. Key exchange:
   - Supported groups:
     - Elliptic Curve Groups (ECDHE):
       - secp256r1(0x0017) (mandatory)
       - secp384r1(0x0018)
       - secp521r1(0x0019)
       - x25519(0x001D) (should)
       - x448(0x001E)
     - Finite Field Groups (DHE)
       - ffdhe2048(0x0100)
       - ffdhe3072(0x0101)
       - ffdhe4096(0x0102)
       - ffdhe6144(0x0103)
       - ffdhe8192(0x0104)
   - Task:
     - Generate private key
     - Generate public key
     - Create master key
2. HMAC based Key Derive Function (HKDF)
   - Supported digest function: SHA256/384
3. Cipher encryption and decryption:
   - Cipher list:
     - AES-128-GCM (mandatory)
     - AES-256-GCM (should)
     - CHACHA20-POLY1305 (should)
     - AES-128-CCM
     - AES-128-CCM-8



Server only

1. Signature algorithms
   - Supported algorithm:
     - RSASSA-PKCS1-v1_5 algorithms
       - rsa_pkcs1_sha256(0x0401) (mandatory)
       - rsa_pkcs1_sha384(0x0501)
       - rsa_pkcs1_sha512(0x0601)
     - ECDSA algorithms
       - ecdsa_secp256r1_sha256(0x0403) (mandatory)
       - ecdsa_secp384r1_sha384(0x0503)
       - ecdsa_secp521r1_sha512(0x0603)
     - RSASSA-PSS algorithms with public key OID rsaEncryption
       - rsa_pss_rsae_sha256(0x0804) (mandatory)
       - rsa_pss_rsae_sha384(0x0805)
       - rsa_pss_rsae_sha512(0x0806)
     - EdDSA algorithms
       - ed25519(0x0807)
       - ed448(0x0808)
     - RSASSA-PSS algorithms with public key OID RSASSA-PSS
       - rsa_pss_pss_sha256(0x0809)
       - rsa_pss_pss_sha384(0x080a)
       - rsa_pss_pss_sha512(0x080b)
     - Legacy algorithms
       - rsa_pkcs1_sha1(0x0201)
       - ecdsa_sha1(0x0203)
   - Task:
     - Create certificate
     - Creage public and private key
     - Sign certificate to create signature



Client only

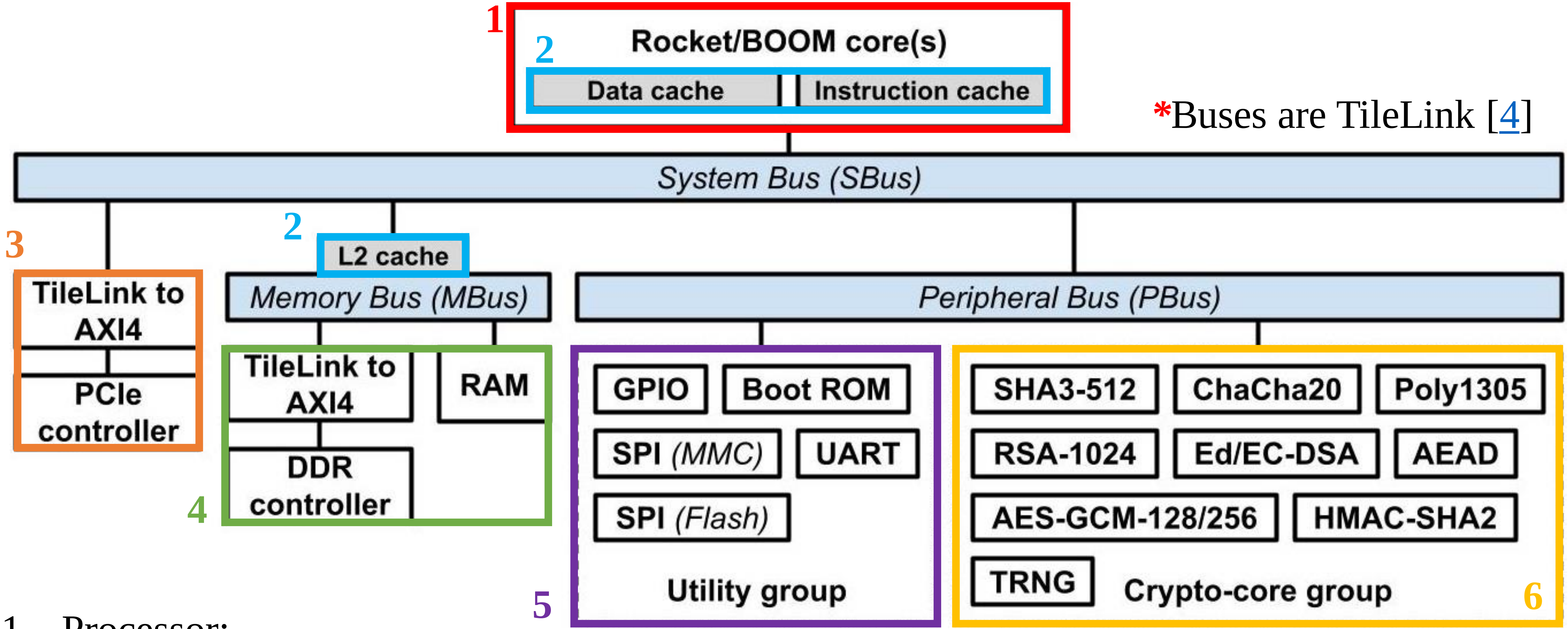
1. Signature algorithms
   - Supported algorithms:
     - Same as Server (does not have to support as many as the server does)
   - Task:
     - Verify signature

The key algorithms:

- ECDSA (*must*)
- Ed25519 (*should*)
- HMAC (*must*)
- AES-GCM (*must*)
- ChaCha-Poly (*should*)
- RSA (*must*)
- SHA2 (*must*)
- SHA3 (*should*)

1. Introduction
2. RISC-V-based SoC
3. TLS 1.3 Cryptographic Accelerators
4. Experimental Result
5. Conclusion

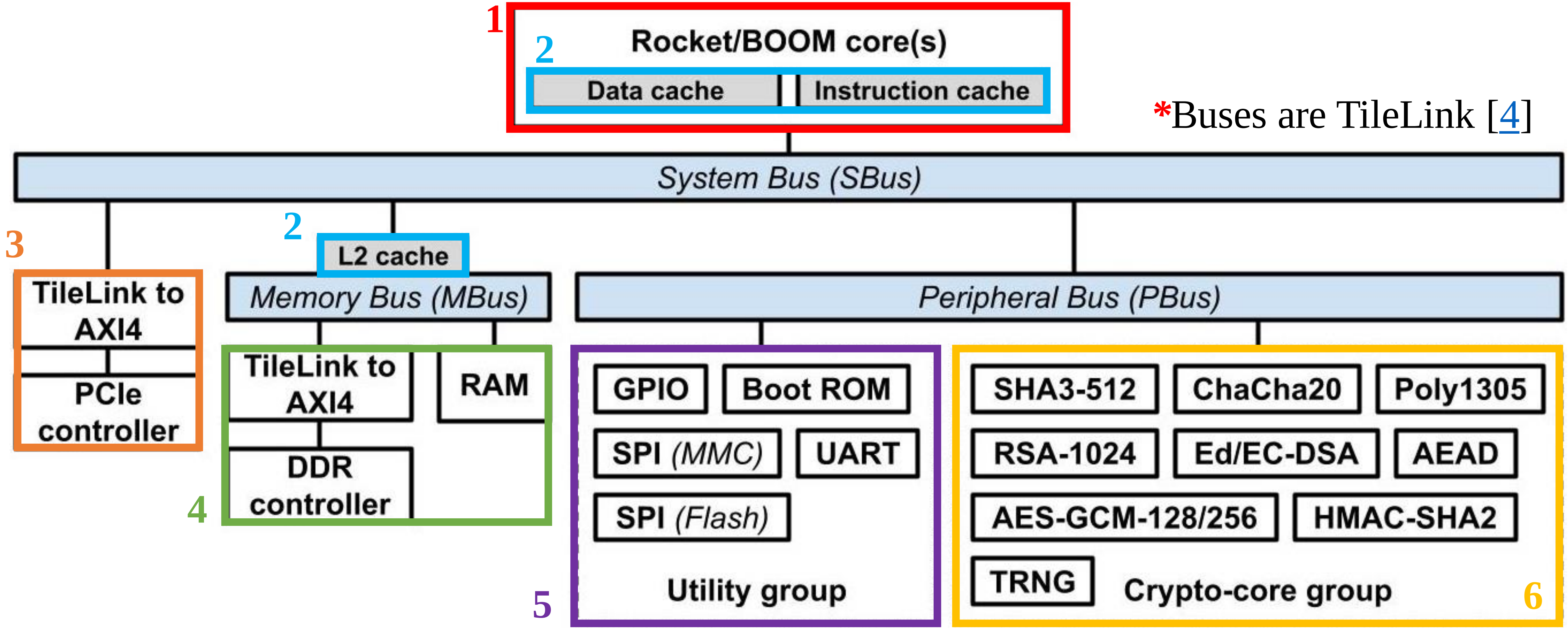
RISC-V-based SoC (1/2)



*Buses are TileLink [4]

1. Processor:
   - Rocket [2] or BOOM [3]
   - RV64GC or RV32IMAC
   - Configurable number of cores
2. Configurable caches sizes
3. PCIe can be included/excluded
4. Can use on-chip RAM or off-chip DDR (or both)

RISC-V-based SoC (2/2)



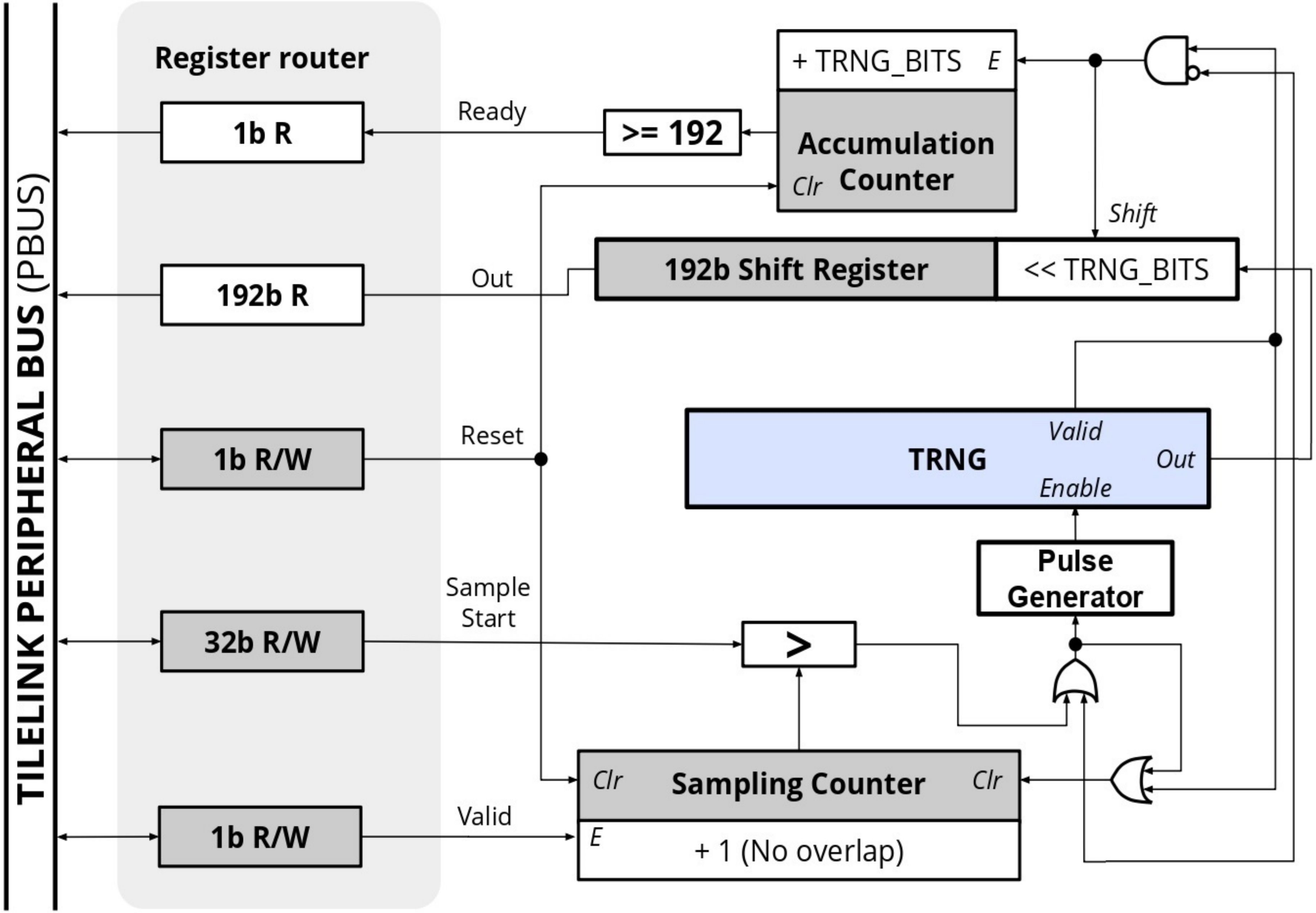
*Buses are TileLink [4]

5. Boot ROM for ZSBL,
   MMC for FSBL & linux image,
   and UART for communication.

6. Each crypto-core is optional and can be included/excluded.

1. Introduction
2. RISC-V-based SoC
3. TLS 1.3 Cryptographic Accelerators
4. Experimental Result
5. Conclusion

3. TLS 1.3 Crypto-core (1/9) TRNG

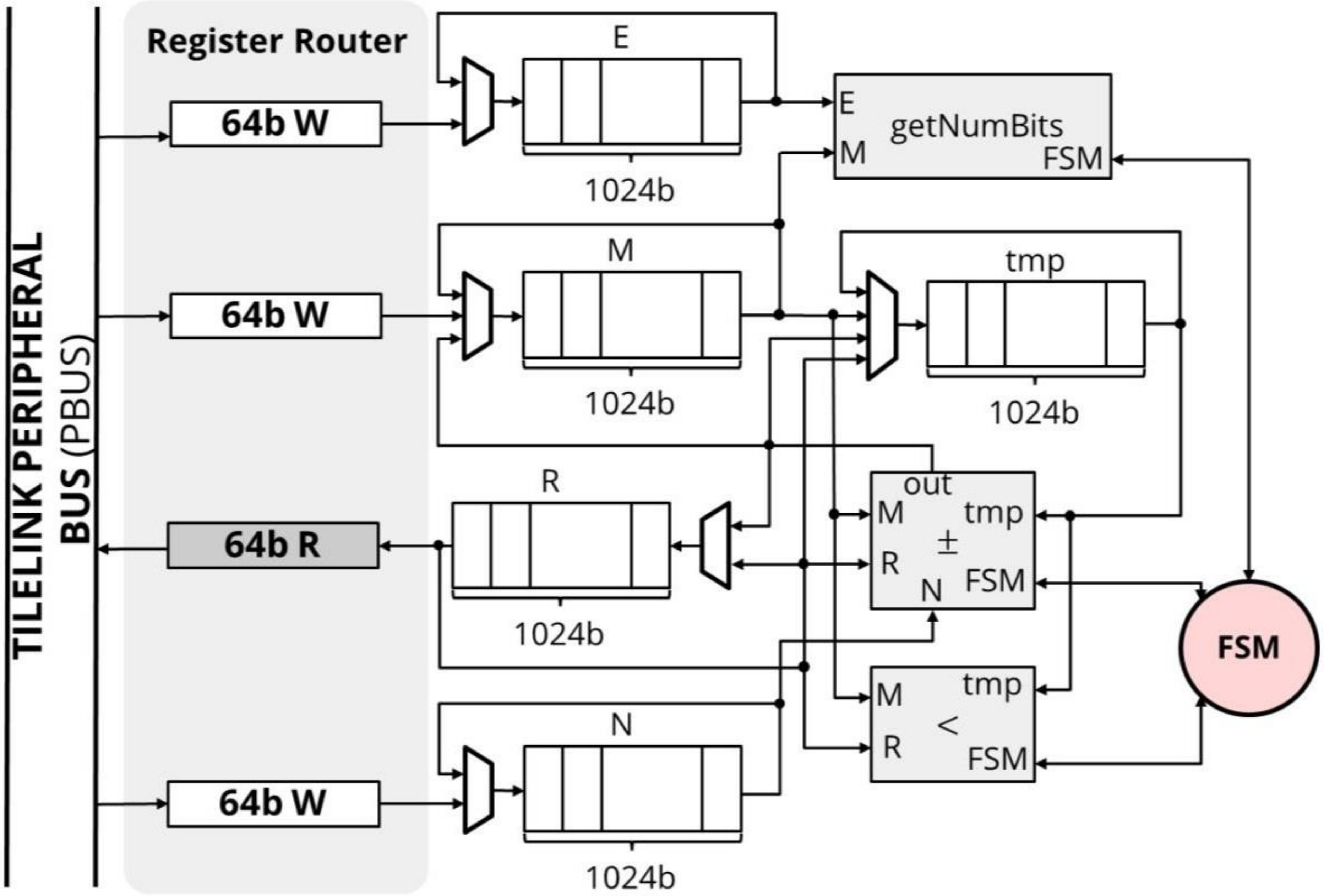


True Random Number Generator (TRNG)

- TRNG core is based on frequency collapse and ring oscillators.
- TRNG core passed NIST 800-22, 800-90B, and AIS31 tests.

[4] TRNG (2021)

3. TLS 1.3 Crypto-core (2/9) RSA-1024

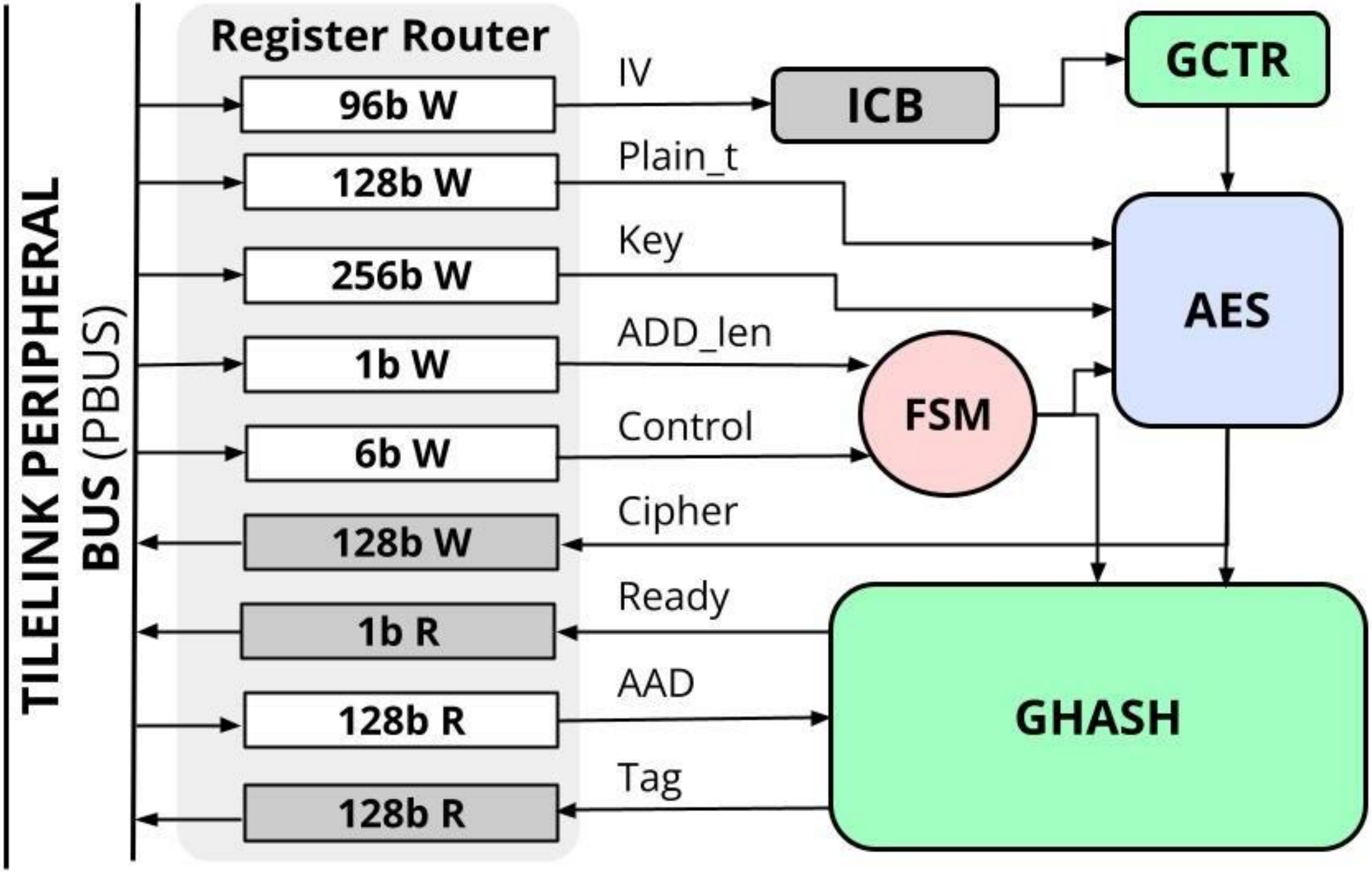


RSA-1024

- Minimize the area by using less “big”-registers as much possible
- Small tasks are done by primitives such as *getNumBits* (number of meaning LSBs), ±, and <
- Primitive functions execute 32-bit at a time

[5] RSA (1978)

3. TLS 1.3 Crypto-core (3/9) AES-GCM

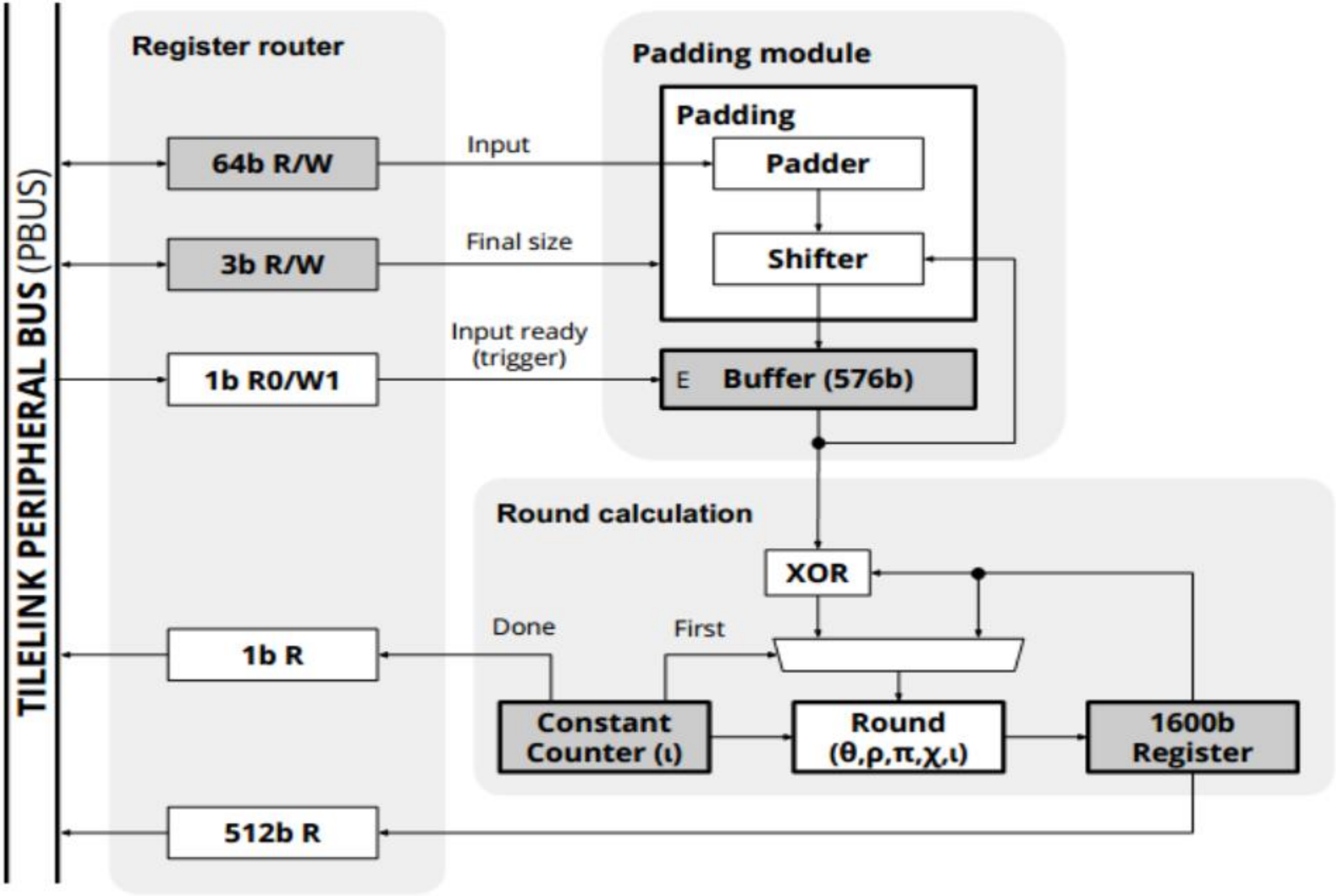


Advanced Encryption Standard (AES) – Galois/Counter Mode (GCM)

- Support encryption and decryption.
- Support 128-bit and 256-bit.

[6] GCM (2005)

3. TLS 1.3 Crypto-core (4/9) SHA3-512

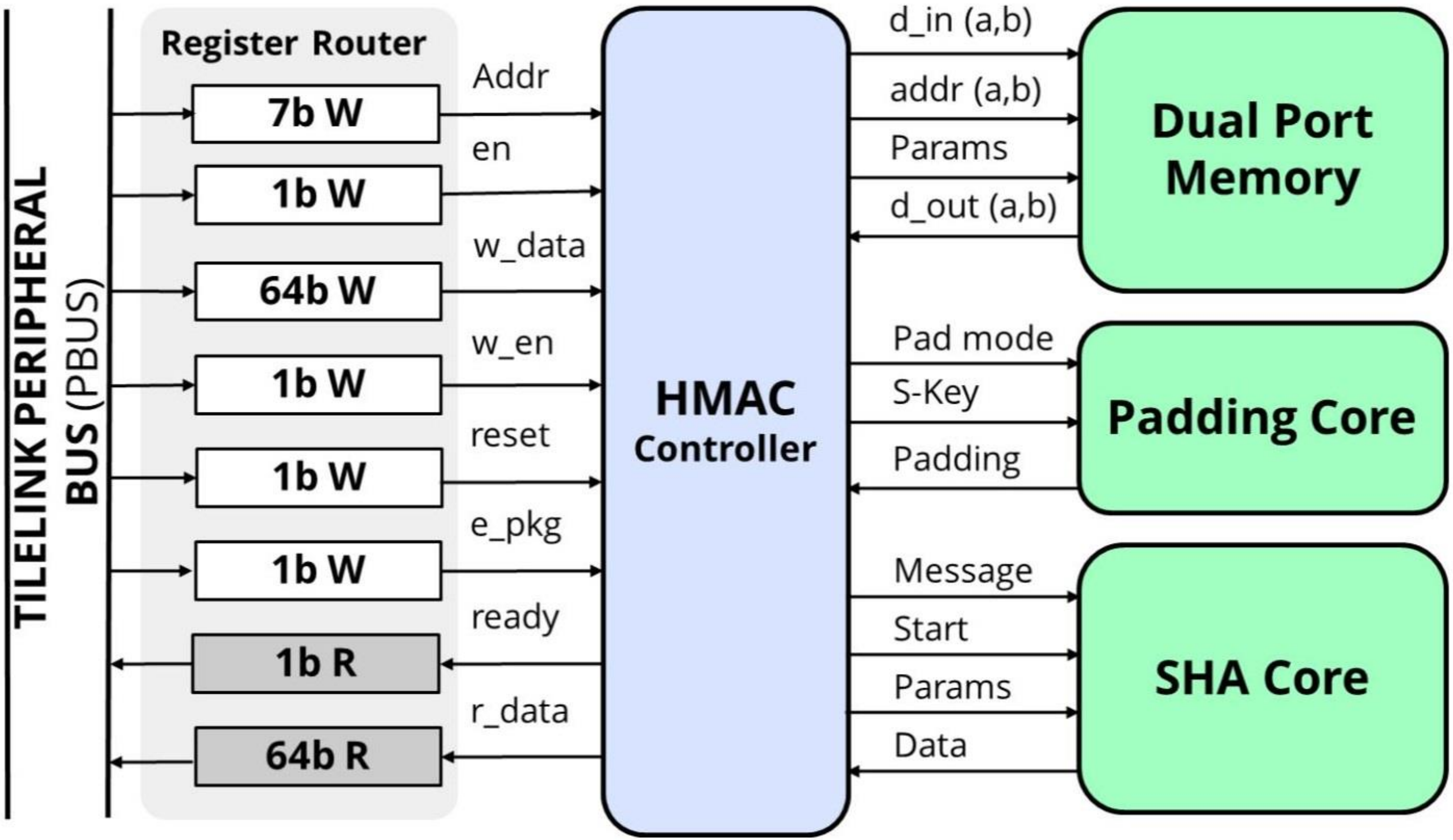


SHA3-512

- Support 512-bit.
- The core was developed based on an open-source project [7] (2018).

[7] SHA3 (2018)

3. TLS 1.3 Crypto-core (5/9) HMAC-SHA2

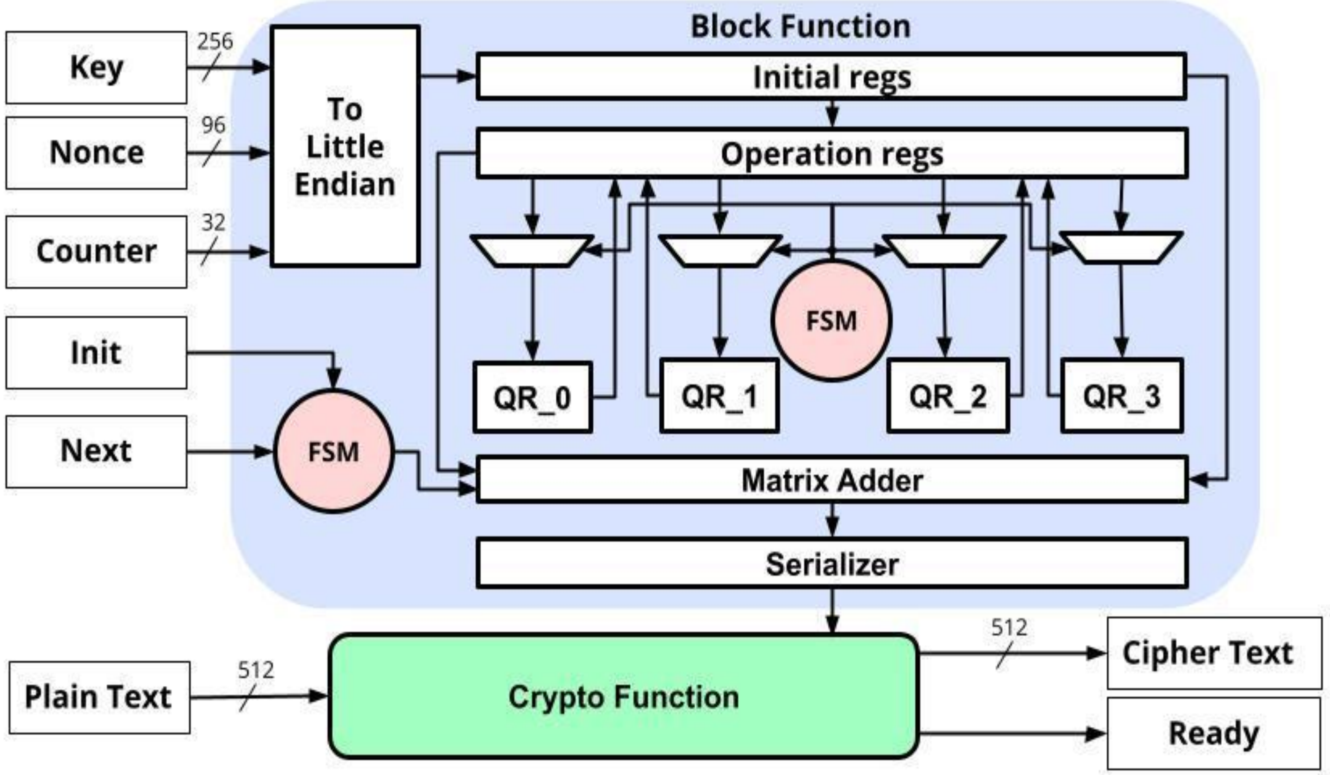


Hash-based Message Authentication Code (HMAC) - SHA2

- Has two modes: HMAC-SHA2 or SHA2 only.
- Support: 256-bit, 384-bit, and 512-bit.

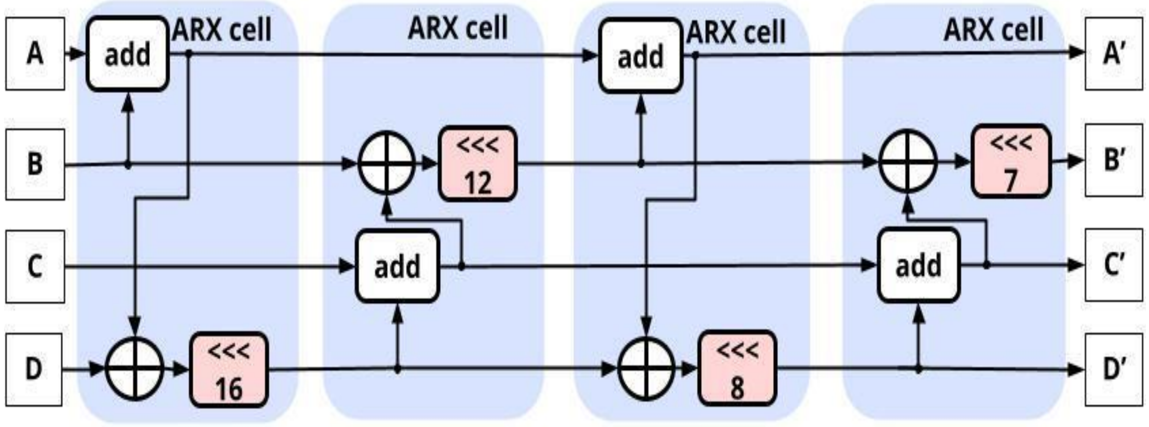
[8] HMAC-SHA2 (1997)

3. TLS 1.3 Crypto-core (6/9) ChaCha20



ChaCha20

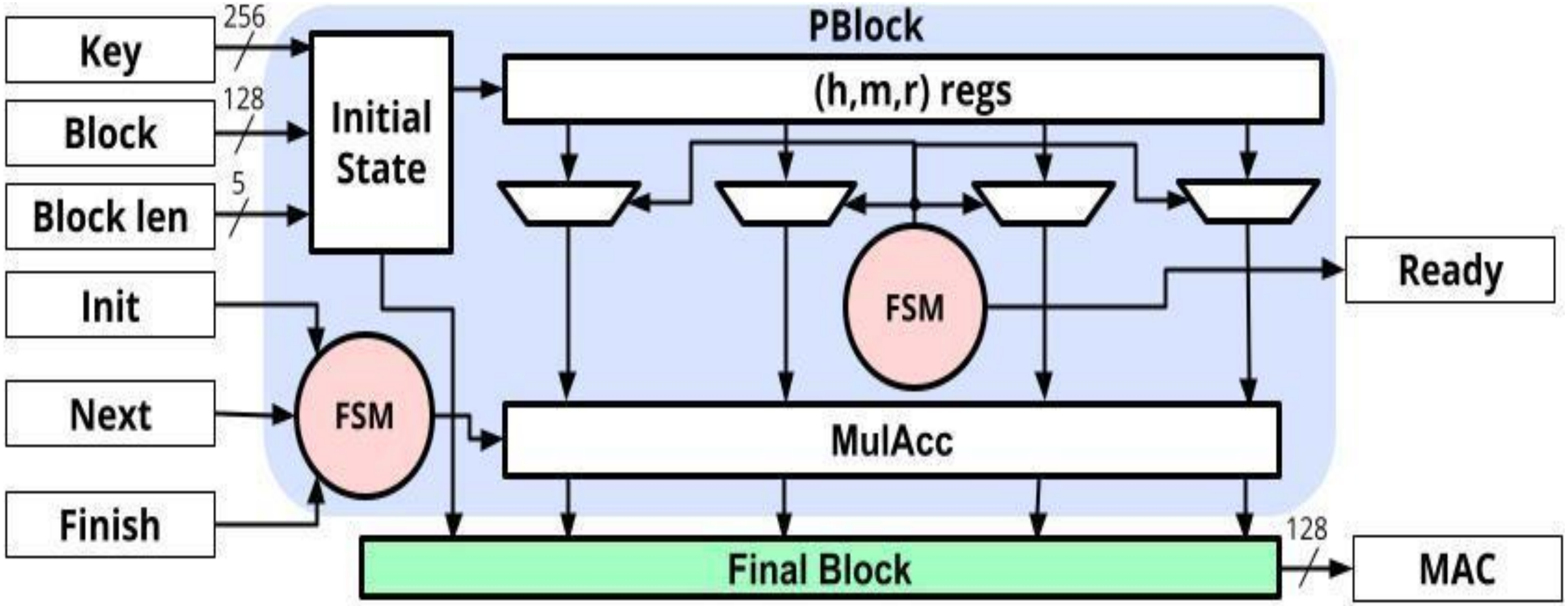
- A stream cipher that was standardized recently [9] (2018).
- Can work alone or team-up with Poly1305 to perform Authenticated Encryption with Additional Data (AEAD)

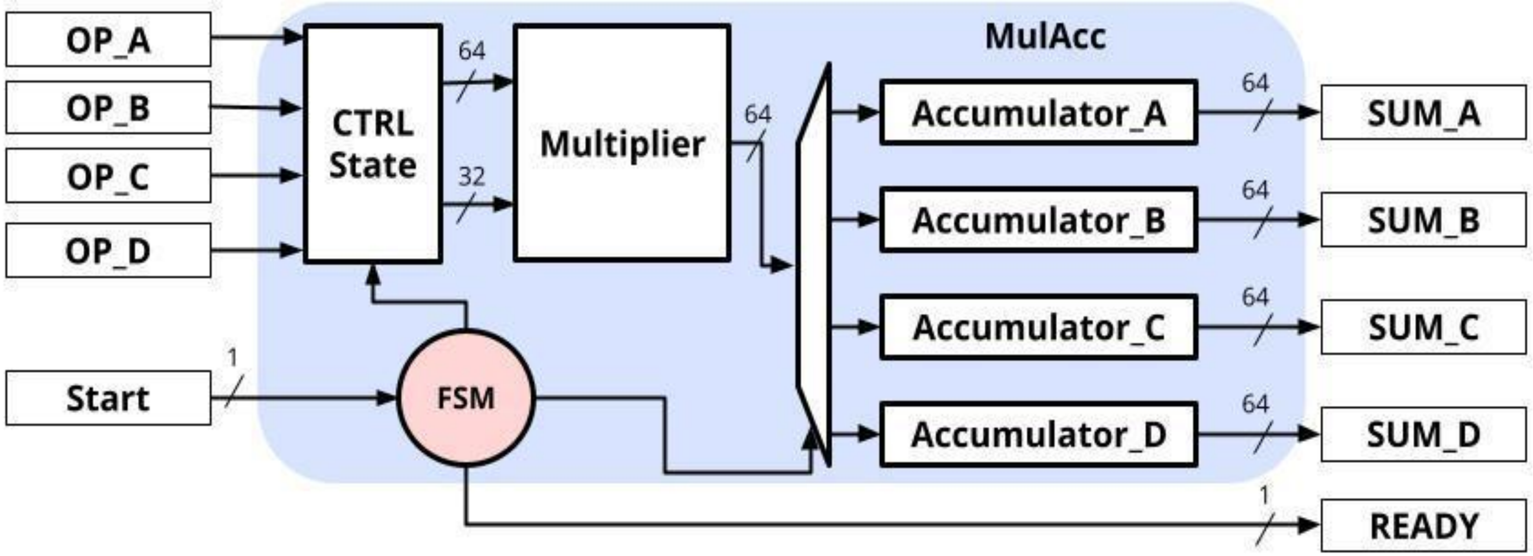


QR sub-module

[9] AEAD Protocol (2018)

3. TLS 1.3 Crypto-core (7/9) Poly1305





MulAcc sub-module

Poly1305

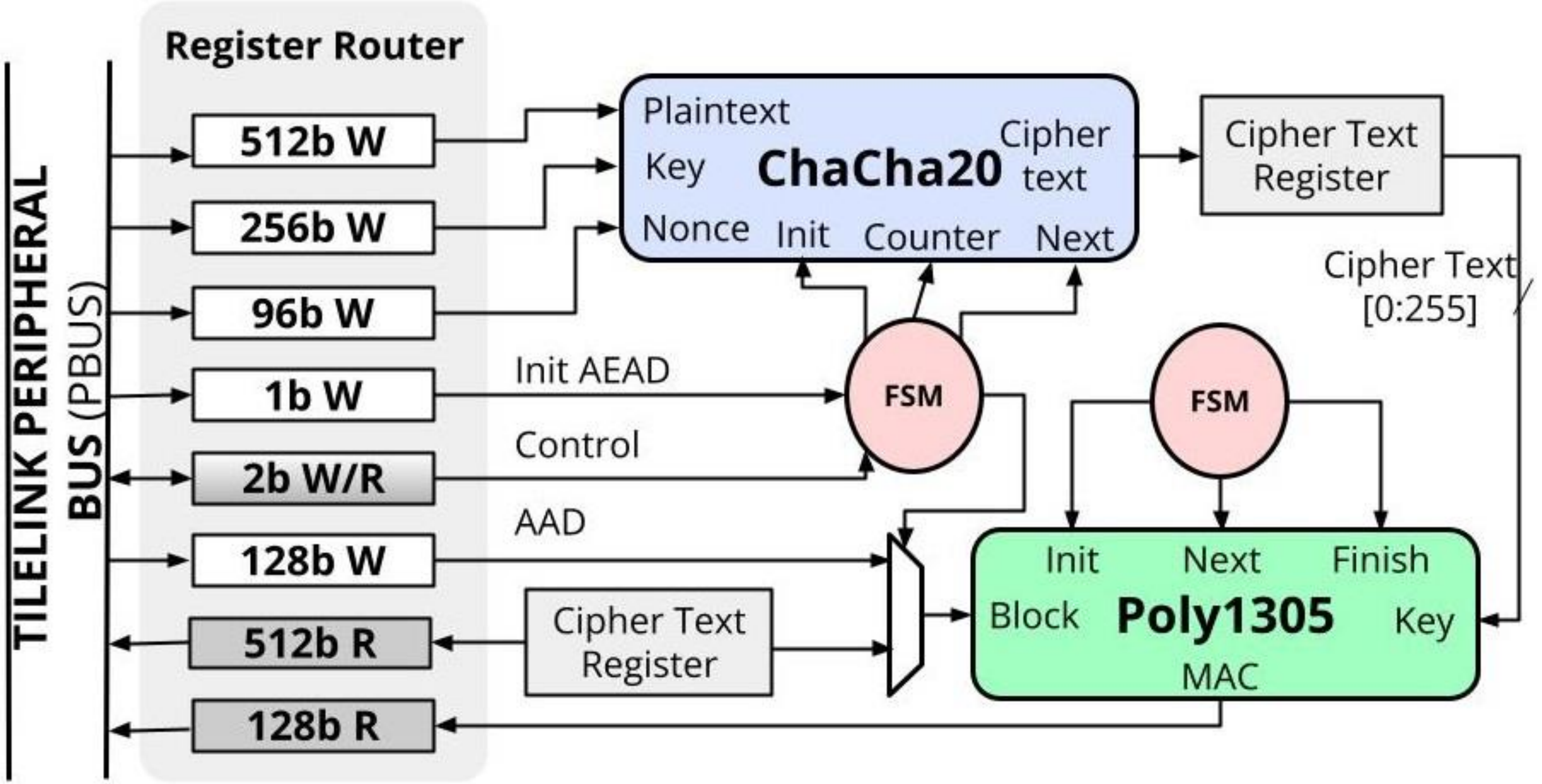
- A Message Authentication Code (MAC) that was standardized recently [9] (2018).
- Can work alone or team-up with ChaCha20 to perform Authenticated Encryption with Additional Data (AEAD)

[9] AEAD Protocol (2018)

3. TLS 1.3 Crypto-core (8/9) AEAD

Authenticated Encryption with Associated Data (AEAD)

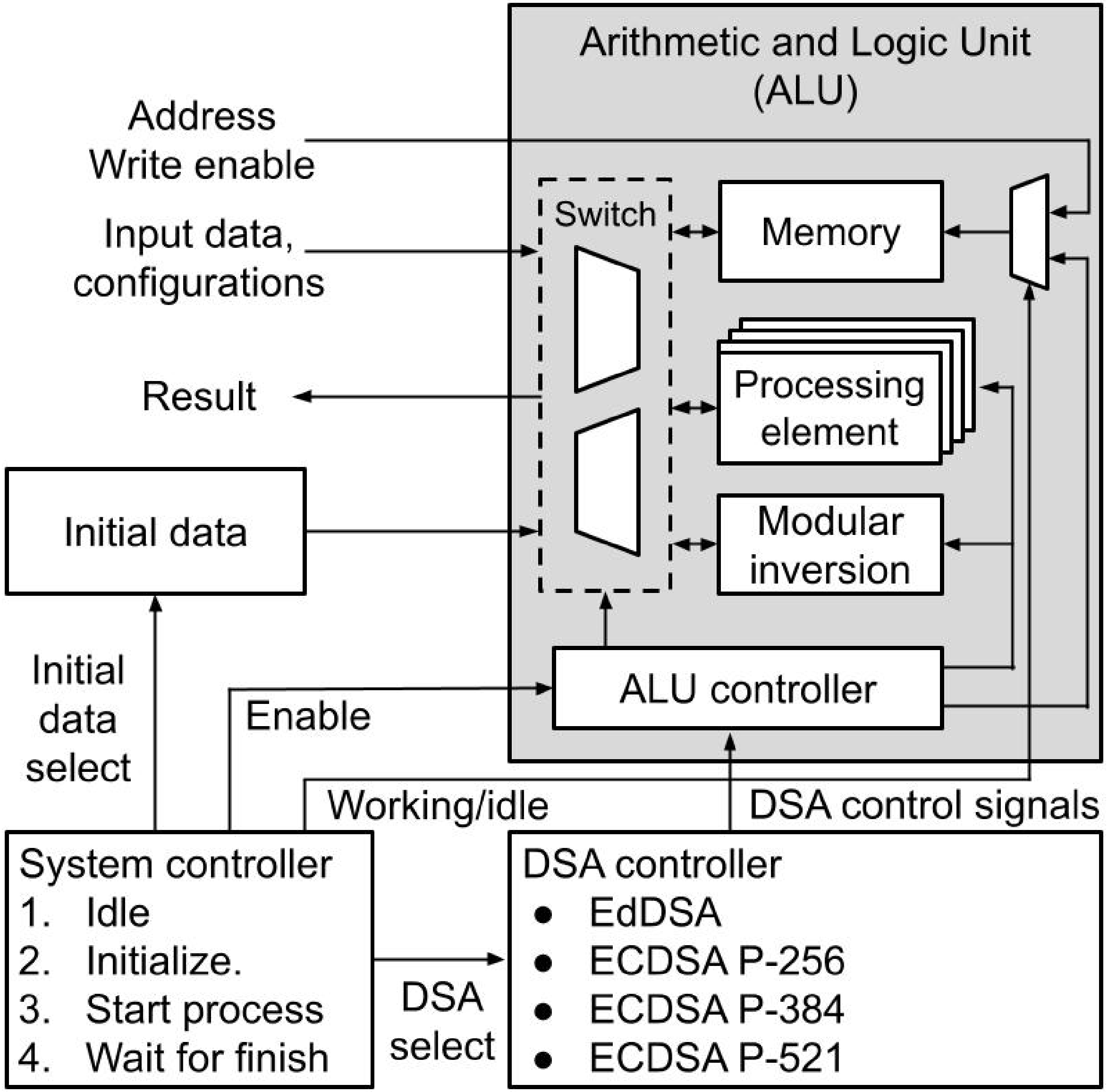
- Use ChaCha20 as a stream cipher and Poly1305 as a MAC.
- The ChaCha20 & Poly1305 cores are the same as mentioned earlier.



[9] AEAD Protocol (2018)

[10] AEAD Impl. (2021)

3. TLS 1.3 Crypto-core (9/9) EC/Ed-DSA



System controller
1. Idle
2. Initialize.
3. Start process
4. Wait for finish

DSA controller
- EdDSA
- ECDSA P-256
- ECDSA P-384
- ECDSA P-521

Ecliptic Curve (EC) + Edwards-curve (Ed) Digital Signature Algorithm (DSA)

- Support four curves: three of ECDSA and one of EdDSA
- Support 256-bit, 384-bit, and 512-bit
- Support functions: gen-key, sign, and verify

[11] ECDSA (2001) [12] EdDSA (2017) [13] EC/Ed-DSA Impl. (2022)

1. Introduction
2. RISC-V-based SoC
3. TLS 1.3 Cryptographic Accelerators
4. Experimental Result
5. Conclusion

4. Experimental Result (1/9) Virtex-7

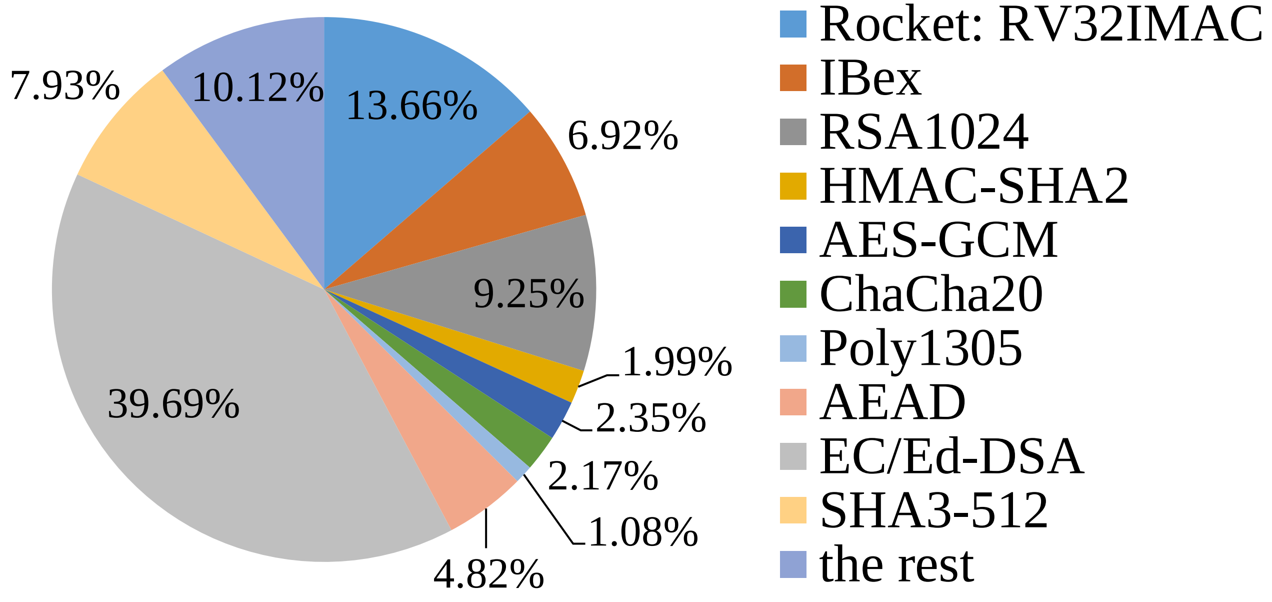
Build report with the FPGA chip of Virtex-7 (XC7VX485TFFG1761-2)

Core	**TRNG**	**RSA-1024**	**AES-GCM**	**SHA3**	**HMAC-SHA2**	**ChaCha20**	**Poly1305**	**AEAD**	**EC/Ed-DSA**
LUT	308	10,051	2,554	8,622	2,166	2,357	1,176	5,239	43,131
Register	563	6,284	3,650	2,231	1,351	1,088	1,506	3,377	9,021
Block RAM	0	0	0	0	2	0	0	0	7.5
DSP	0	0	0	0	0	0	7	7	0
Fmax (MHz)	N/A	102.44	150.35	118.91	139.59	123.06	112.37	111.98	97.4

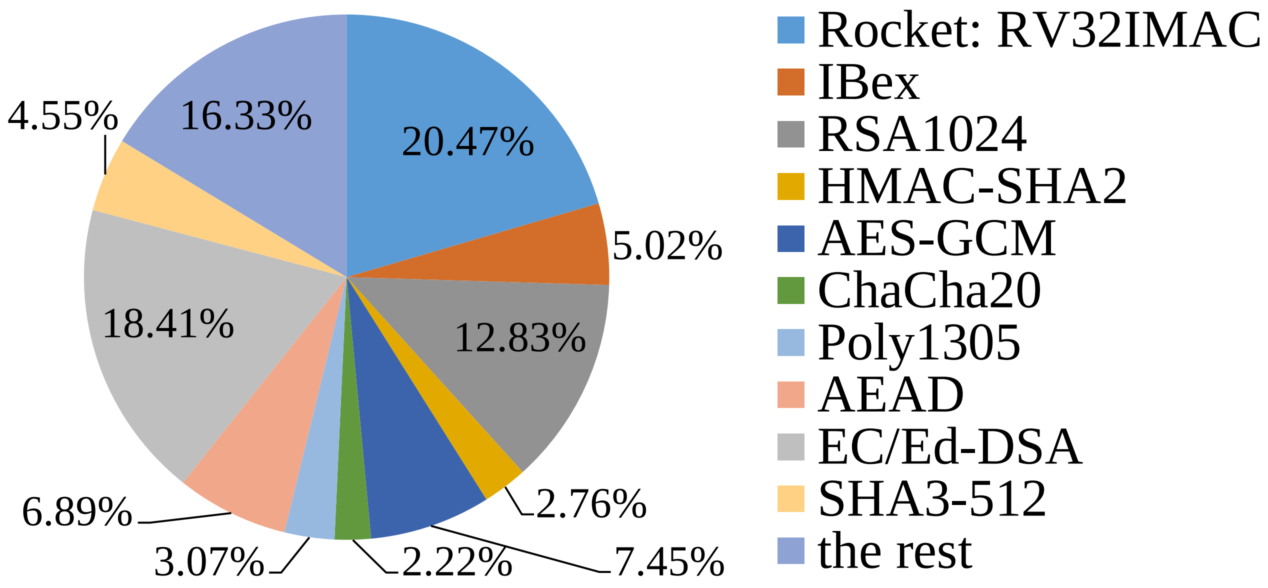
4. Experimental Result (2/9) Virtex-7

SoC resources utilization pie chart:
All crypto-cores + Single-core RV32IMAC Rocket

LUT Ultilization



Register Utilization



***Note**: “*the rest*” means all the buses, TRNG
and utility-group peripherals such as GPIO, SPI, boot ROM, etc.

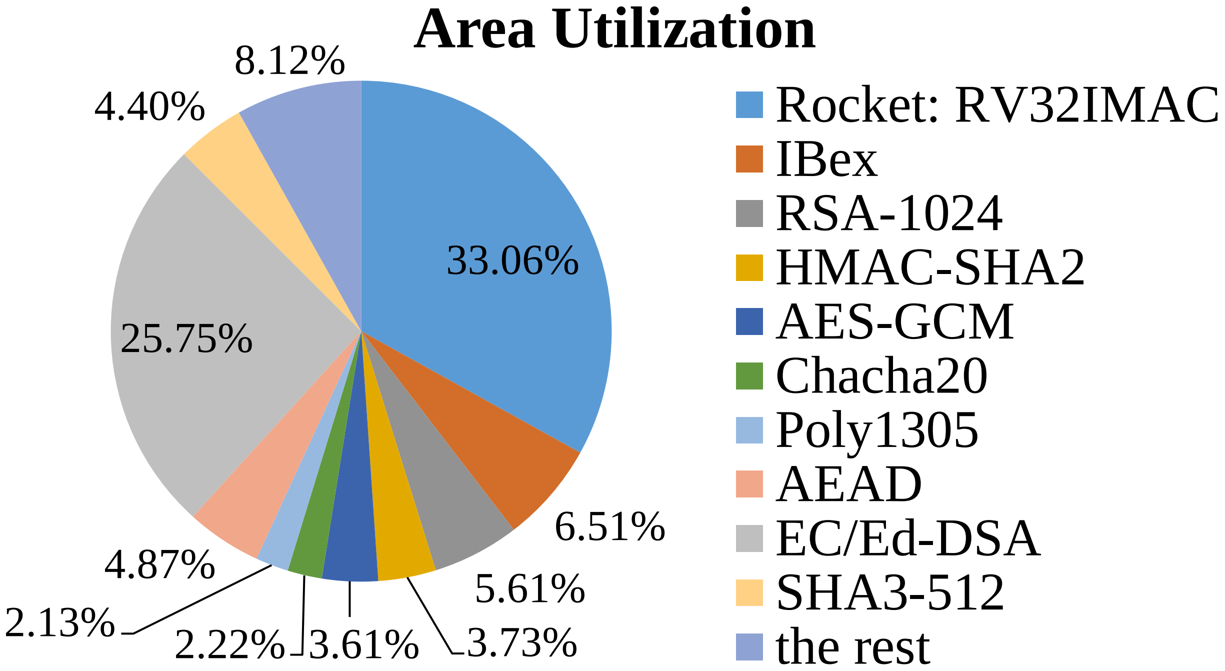
4. Experimental Result (3/9) ROHM180 synthesis

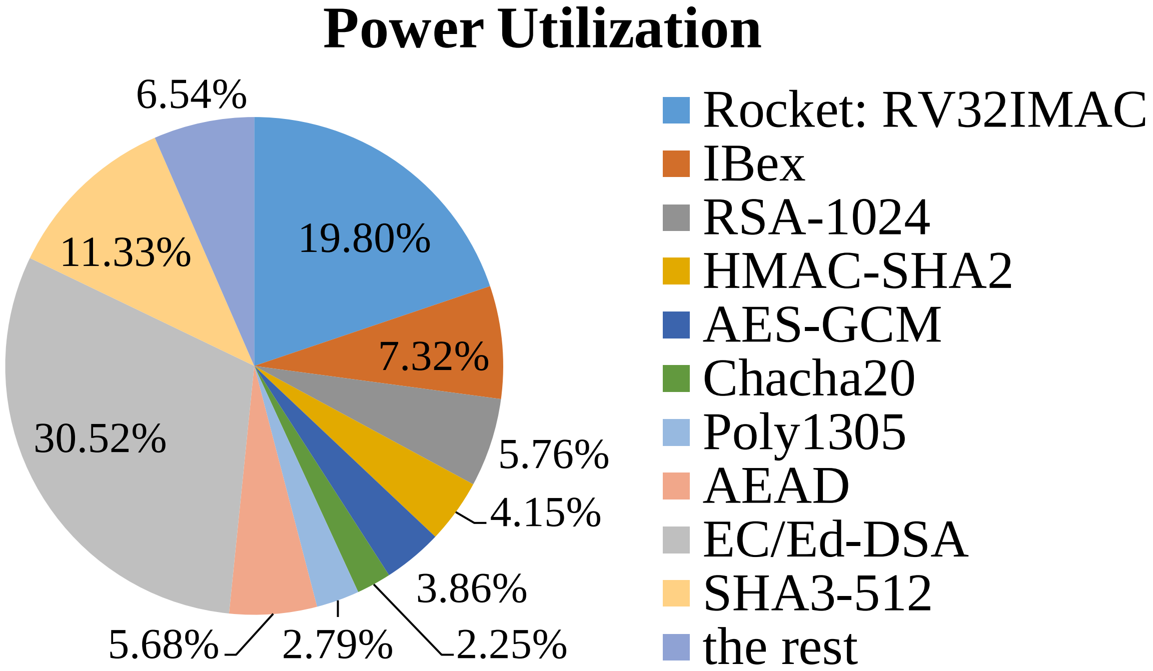
Build report with ROHM180 process library (synthesis result)

Core			**TRNG**	**RSA-1024**	**AES-GCM**	**SHA3**	**HMAC-SHA2**	**Chacha20**	**Poly1305**	**AEAD**	**EC/Ed-DSA**
@SYN	**#Cell**		2,223	34,496	17,764	24,087	12,675	11,782	9,328	24,558	161,943
	Area (μm^2)		55,851	777,479	474,954	581,908	515,131	288,656	275,650	645,688	3,567,223
	SRAM	**(μm^2)**	0	0	0	0	181,926.55	0	0	0	0
		(%)	0.00%	0.00%	0.00%	0.00%	35.32%	0.00%	0.00%	0.00%	0.00%
	Power (mW)		10.745	208.277	137.822	309.365	122.866	105.104	98.282	212.169	1,251.86
	Fmax (MHz)		119	93	114	129	94	94	94	94	93
@PNR	**#Gate**		5,866	82,827	51,586	63,221	53,656	31,625	29,417	69,478	371,014
	#Cell		2,261	36,085	19,416	24,461	12,951	12,778	9,991	26,029	163,999
	Area (μm^2)		90,768	1,037,057	666,160	811,858	688,293	410,419	392,675	898,855	4,756,399
	Density		71.83%	77.29%	78.78%	78.85%	75.44%	79.49%	77.39%	78.08%	75.48%
	Power (mW)		7.241	105.8	70.78	207.9	76.16	41.26	51.29	104.2	560.2
	Fmax (MHz)		119	101	113	112	100	100	100	100	100
	#MOSFET		27,542	362,608	234,180	273,250	242,918	135,960	136,308	313,842	1,649,480

4. Experimental Result (4/9) ROHM180 synthesis

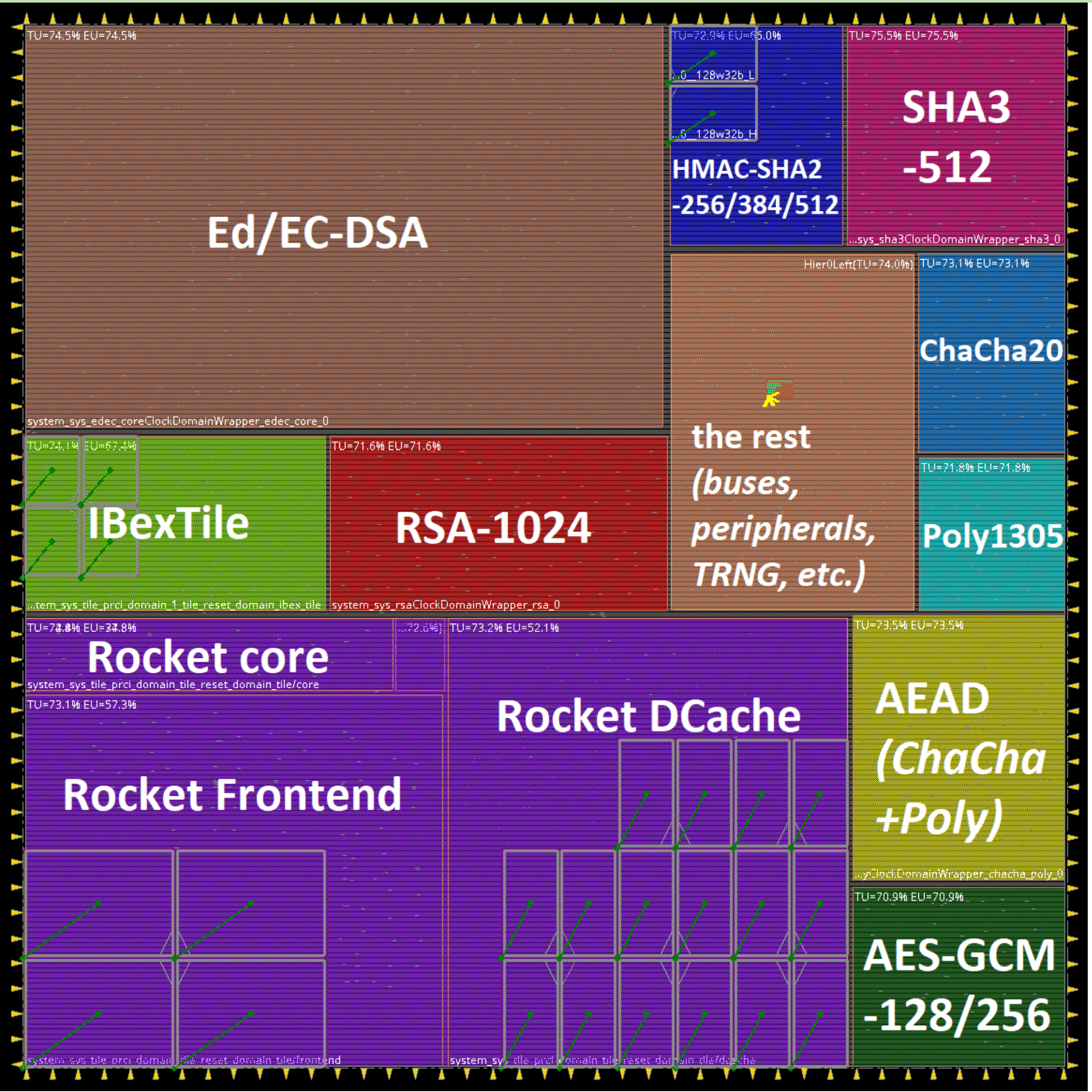
SoC resources utilization pie chart:
All crypto-cores + Single-core RV32IMAC Rocket





***Note**: “*the rest*” means all the buses, TRNG,
and utility-group peripherals such as GPIO, SPI, boot ROM, etc.

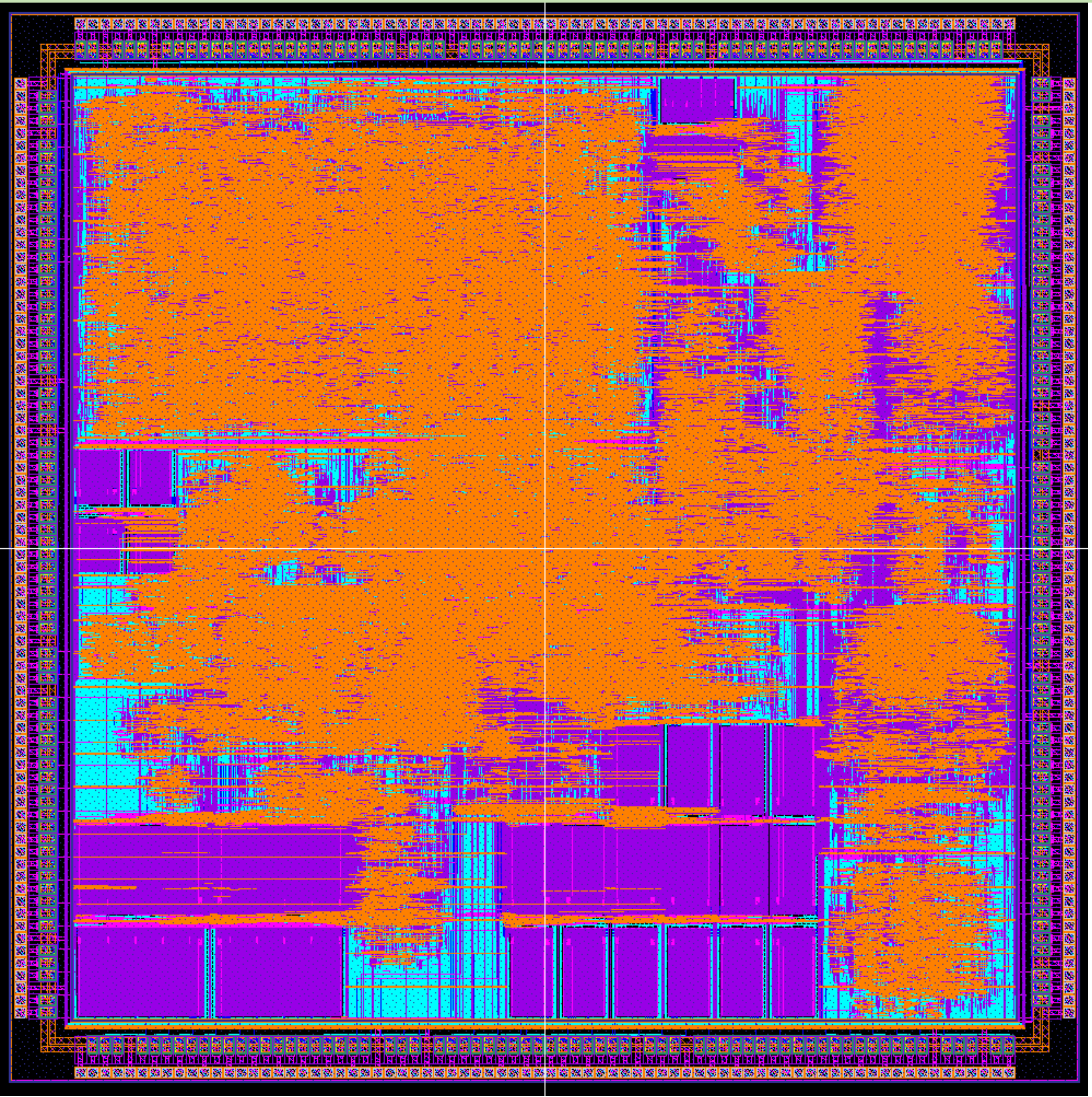
4. Experimental Result (5/9) ROHM180 chip



- One chip was fabricated in ROHM180 in Feb. 2022 *(expected to be delivered in June 2022)*
- Single-core Rocket with ISA of RV32IMAC
- All the crypto-cores for TLS-1.3 are included

Floorplan

4. Experimental Result (6/9) ROHM180 chip



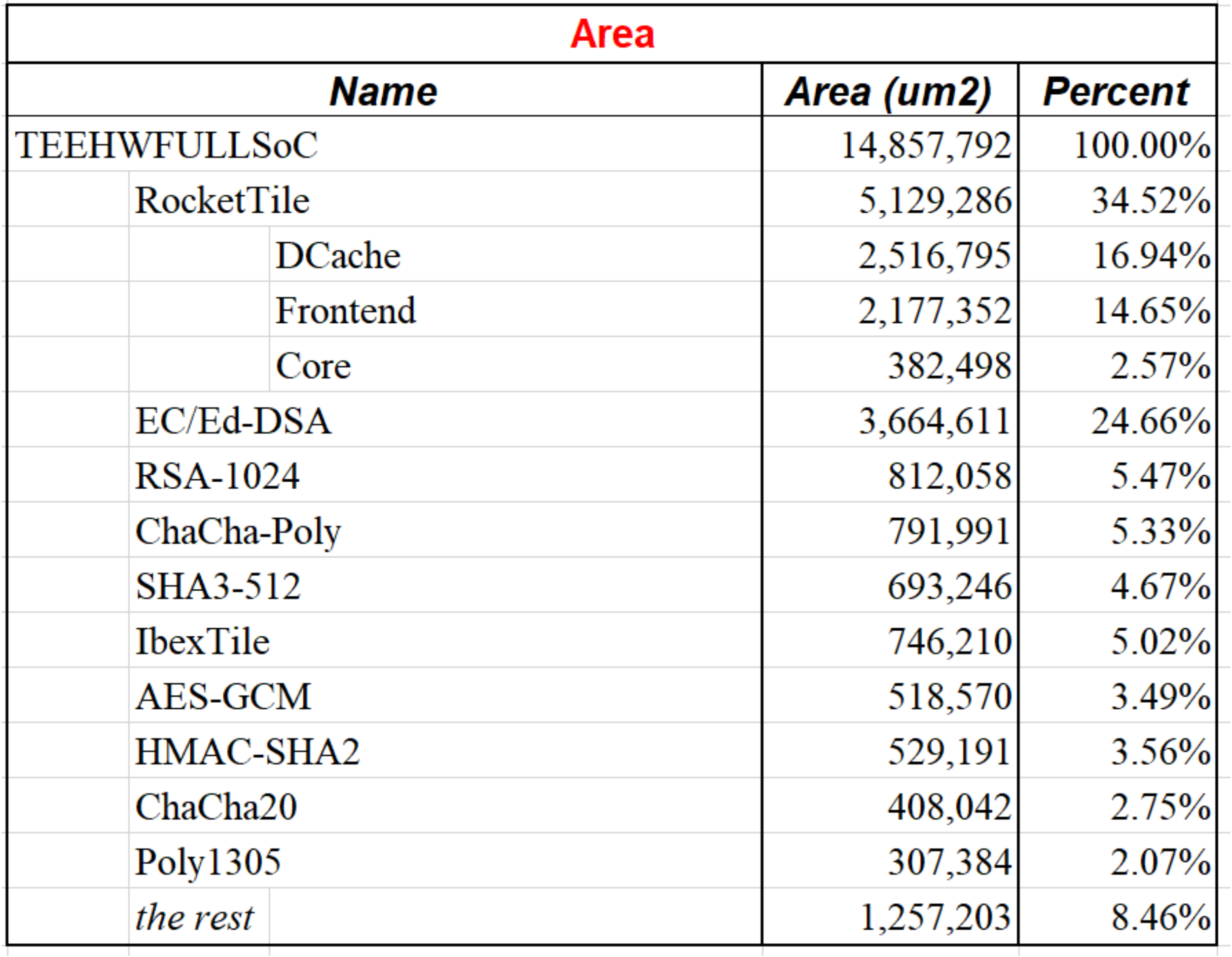
Key features summarize

Core			Rocket ×1
ISA			RV32IMAC
Cache			$I =16KB and $D = 16KB
Crypto-cores: TRNG, RSA, AES-GCM, SHA3, HMAC-SHA2, ChaCha20, Poly1305, AEAD, & EC/Ed-DSA			
@SYN	**#Cell**		460,195
	Area (μm^2)		14,744,115
	SRAM	**(μm^2)**	3,386,029
		(%)	22.97%
	Power (mW)		3,075
	Fmax (MHz)		18
@PNR	**#Gate**		1,535,403
	#Cell		466,882
	Area (μm^2)		20,799,437
	Density		71.43%
	Power (mW)		1,992
	Fmax (MHz)		71
	#MOSFET		7,982,582

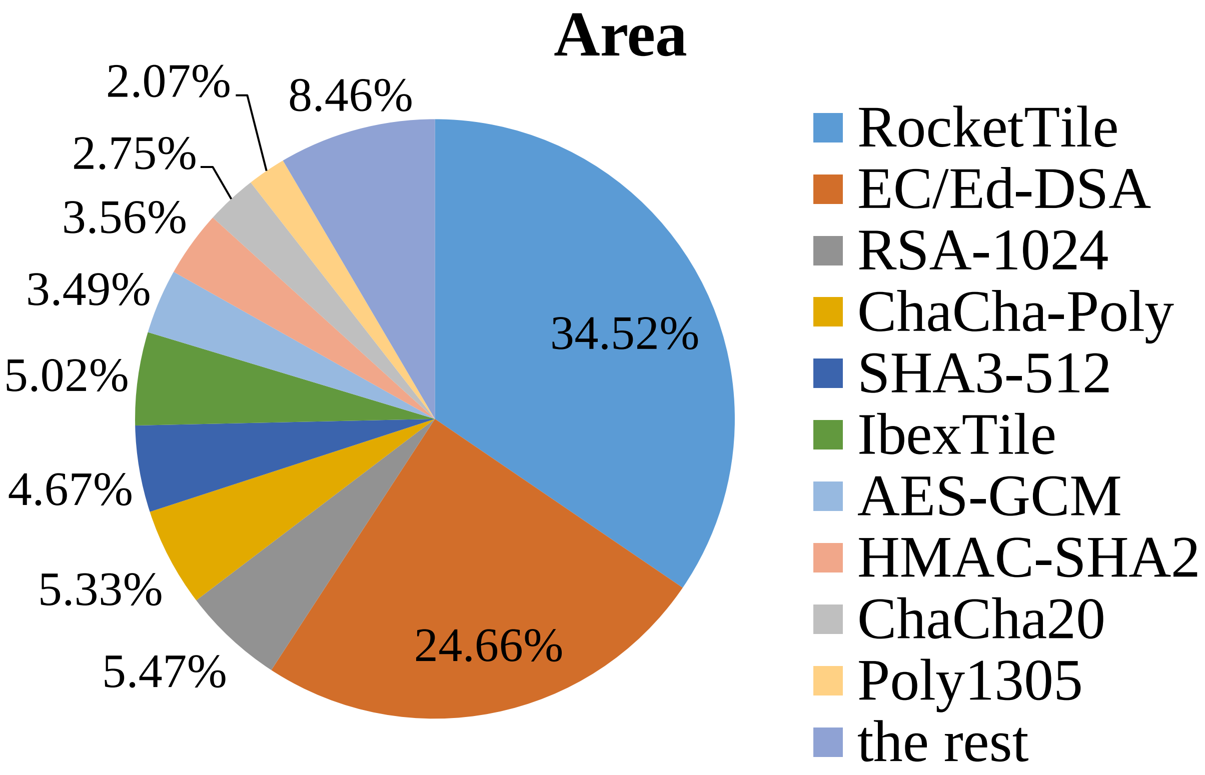
Layout

4. Experimental Result (7/9) ROHM180 chip

Chip resources utilization pie chart

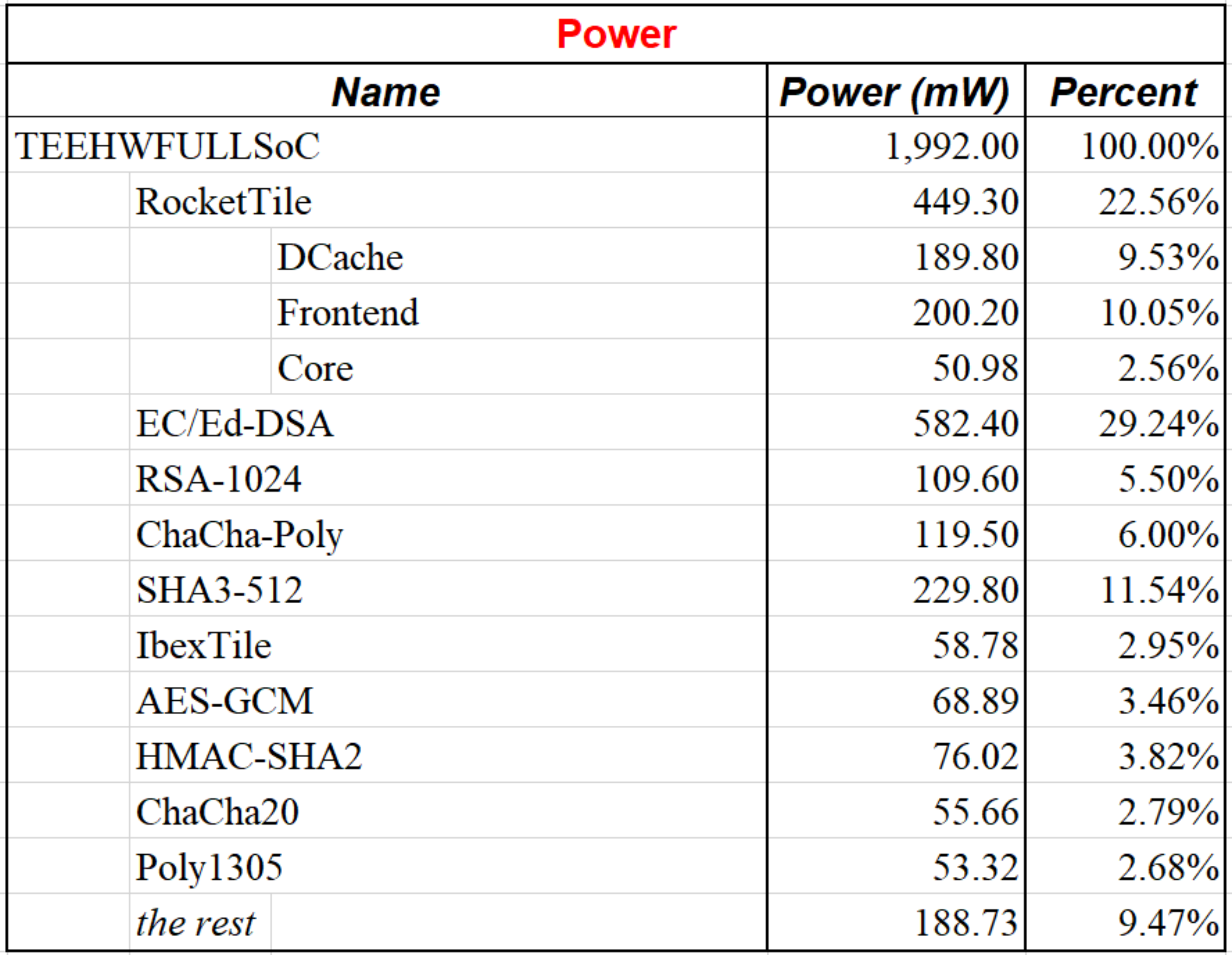


Area		
Name	***Area (um2)***	***Percent***
TEEHWFULLSoC	14,857,792	100.00%
RocketTile	5,129,286	34.52%
DCache	2,516,795	16.94%
Frontend	2,177,352	14.65%
Core	382,498	2.57%
EC/Ed-DSA	3,664,611	24.66%
RSA-1024	812,058	5.47%
ChaCha-Poly	791,991	5.33%
SHA3-512	693,246	4.67%
IbexTile	746,210	5.02%
AES-GCM	518,570	3.49%
HMAC-SHA2	529,191	3.56%
ChaCha20	408,042	2.75%
Poly1305	307,384	2.07%
the rest	1,257,203	8.46%



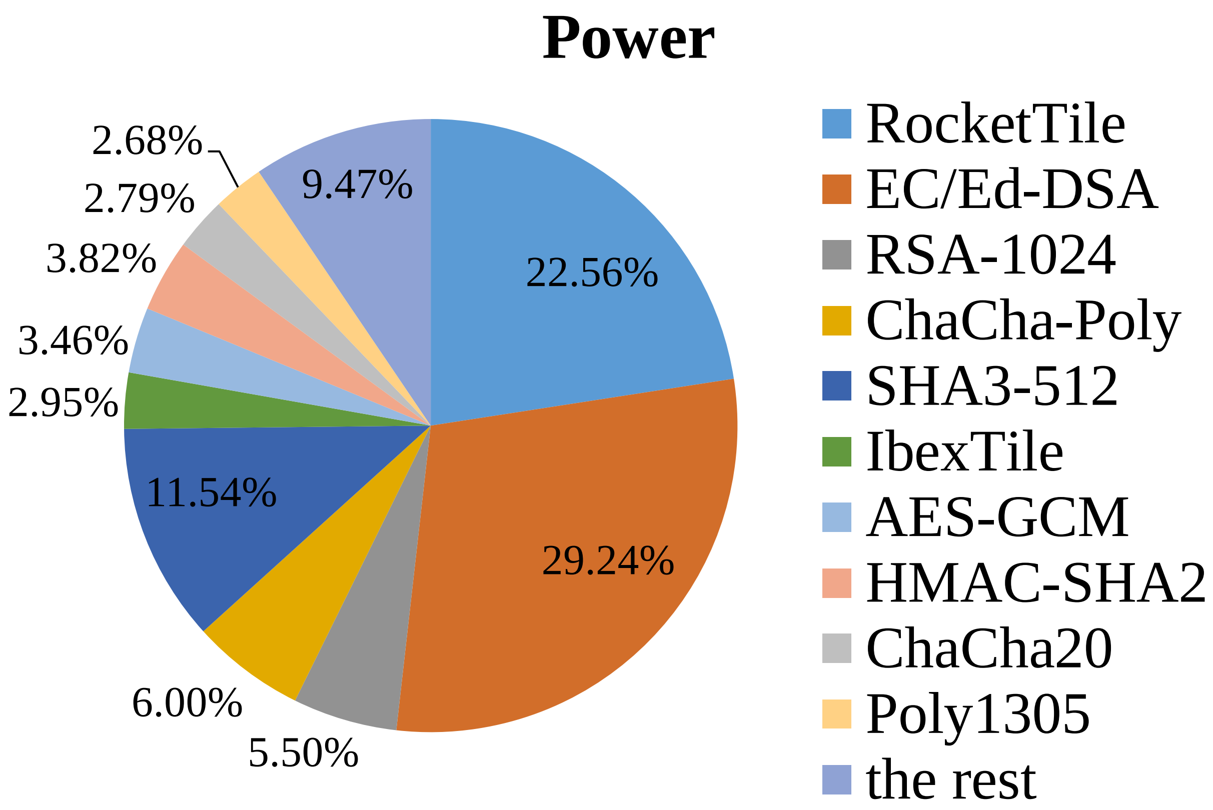
4. Experimental Result (8/9) ROHM180 chip

Chip resources utilization pie chart



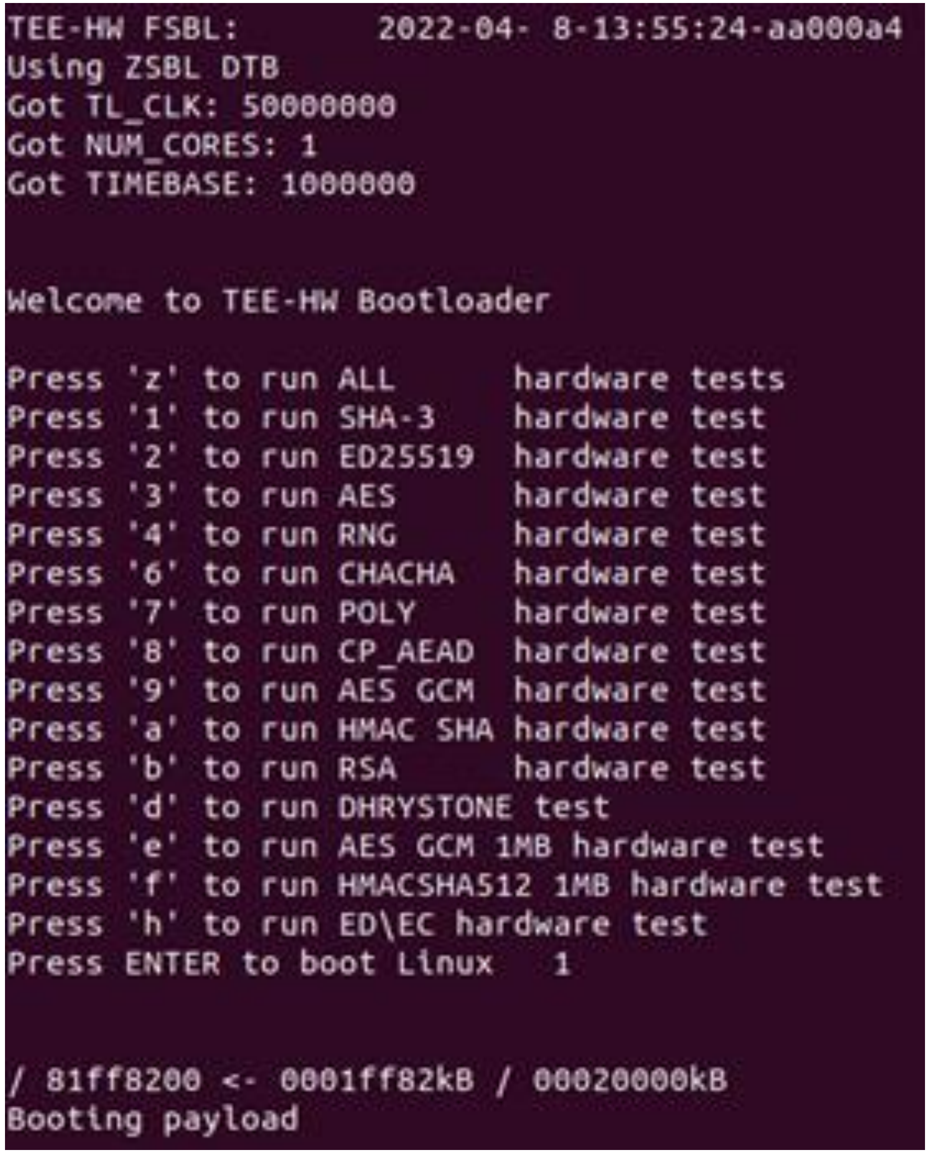
Power

Name	Power (mW)	Percent
TEEHWFULLSoC	1,992.00	100.00%
RocketTile	449.30	22.56%
DCache	189.80	9.53%
Frontend	200.20	10.05%
Core	50.98	2.56%
EC/Ed-DSA	582.40	29.24%
RSA-1024	109.60	5.50%
ChaCha-Poly	119.50	6.00%
SHA3-512	229.80	11.54%
IbexTile	58.78	2.95%
AES-GCM	68.89	3.46%
HMAC-SHA2	76.02	3.82%
ChaCha20	55.66	2.79%
Poly1305	53.32	2.68%
the rest	188.73	9.47%



4. Experimental Result (9/9) Crypto-core driver

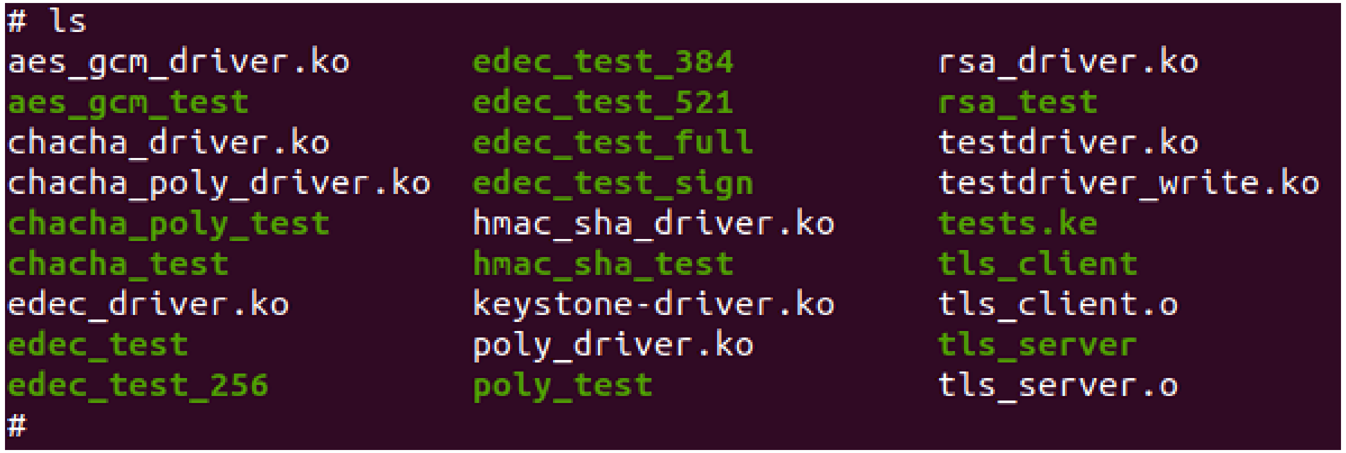
Initial test and driver for using crypto-cores were developed



```
TEE-HW FSBL:        2022-04- 8-13:55:24-aa000a4
Using ZSBL DTB
Got TL_CLK: 50000000
Got NUM_CORES: 1
Got TIMEBASE: 1000000


Welcome to TEE-HW Bootloader

Press 'z' to run ALL       hardware tests
Press '1' to run SHA-3     hardware test
Press '2' to run ED25519   hardware test
Press '3' to run AES       hardware test
Press '4' to run RNG       hardware test
Press '6' to run CHACHA    hardware test
Press '7' to run POLY      hardware test
Press '8' to run CP_AEAD   hardware test
Press '9' to run AES GCM   hardware test
Press 'a' to run HMAC SHA hardware test
Press 'b' to run RSA       hardware test
Press 'd' to run DHRYSTONE test
Press 'e' to run AES GCM 1MB hardware test
Press 'f' to run HMACSHA512 1MB hardware test
Press 'h' to run ED\EC hardware test
Press ENTER to boot Linux   1


/ 81ff8200 <- 0001ff82kB / 00020000kB
Booting payload
```

Tests at FSBL *(M-mode)*
before boot into Linux

Driver and tests *(U-mode)* after boot into Linux



```
# ls
aes_gcm_driver.ko       edec_test_384         rsa_driver.ko
aes_gcm_test            edec_test_521         rsa_test
chacha_driver.ko        edec_test_full        testdriver.ko
chacha_poly_driver.ko   edec_test_sign        testdriver_write.ko
chacha_poly_test        hmac_sha_driver.ko    tests.ke
chacha_test             hmac_sha_test         tls_client
edec_driver.ko          keystone-driver.ko    tls_client.o
edec_test               poly_driver.ko        tls_server
edec_test_256           poly_test             tls_server.o
#
```

1. Introduction
2. RISC-V-based SoC
3. TLS 1.3 Cryptographic Accelerators
4. Experimental Result
5. Conclusion

5. Conclusion (1/1)

- **RISC-V-based:** a complete SoC implementation aiming for TLS-1.3 was developed based on RISC-V architecture.
- **TLS-1.3 Crypto-cores:** a set of newly developed cryptographic accelerators were introduced, including TRNG, RSA-1024, AES-GCM, SHA3, HMAC-SHA2, ChaCha20, Poly1305, AEAD, and EC/Ed-DSA.
- **Flexibility & compact:** the whole SoC fitted nicely in a 5×5-mm^2 of ROHM180 chip. Many crypto-cores have multi-mode and multi-function such as HMAC-SHA2 and EC/Ed-DSA.
- The proposed SoC was **verified & tested** on both FPGA and VLSI implementations.

Thank You

Tokyo, 2nd June, 2022

References

1. E. Rescorla, “The Transport Layer Security (TLS) Protocol Version 1.3,” Tech. Rep. RFC 8446, Aug. 2018. [Online].
2. K. Asanović *et al.*, “The Rocket Chip Generator,” Tech. Rep. EECS-2016-17, Apr. 2016. [Online].
3. C. Celio *et al.*, “BROOM: An Open-Source Out-of-Order Processor With Resilient Low-Voltage Operation in 28-nm CMOS,” *IEEE Micro*, vol. 39, no. 2, pp. 52-60, Apr. 2019. [DOI].
4. R. Serrano *et al.*, “A Fully Digital True Random Number Generator With Entropy Source Based in Frequency Collapse,” *IEEE Access*, vol. 9, pp. 105748-105755, Jul. 2021. [DOI].
5. R. L. Rivest *et al.*, “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems,” *Comm. ACM*, vol. 21, no. 2, pp. 120-126, Feb. 1978. [DOI].
6. D. A. McGrew and J. Viega, “The Galois/Counter Mode of Operation (GCM),” 2005. [Online].
7. H. Hsing, “SHA3 (Keccak),” 2018. [Online].
8. H. Krawczyk *et al.*, “HMAC: Keyed-Hashing for Message Authentication,” Tech. Rep. RFC 2104, Feb. 1997. [Online].
9. Y. Nir *et al.*, “ChaCha20 and Poly1305 for IETF Protocols,” Tech. Rep. RFC 8439, Jun. 2018. [Online].
10. R. Serrano *et al.*, “ChaCha20-Poly1305 Crypto Core Compatible with Transport Layer Security 1.3,” *ISOCC*, Nov. 2021, Jeju, South Korea, pp. 17-18. [DOI].

References

11. D. Johnson *et al.*, “The Elliptic Curve Digital Signature Algorithm (ECDSA),” Int. Journal of Info. Secu., vol. 1, no. 1, pp. 36-63, Aug. 2001. [DOI].
12. S. Josefsson and I. Liusvaara, “Edwards-Curve Digital Signature Algorithm (EdDSA),” Tech. Rep. RFC 8032, Jan. 2017. [Online].
13. Binh Kieu-Do-Nguyen *et al.*, “Low-Cost Area-Efficient FPGA-Based Multi-Functional ECDSA/EdDSA,” *Cryptography*, vol. 6, no. 2, pp. 1-13, May 2022. [DOI].