国立大学法人 電気通信大学

NATIONAL INSTITUTE OF ADVANCED INDUSTRIAL SCIENCE AND TECHNOLOGY (AIST)

Technology Research Association of Secure IoT Edge application based on RISC-V Open architecture

TEE Hardware for RISC-V Implementation

Authors: Ckristian Duran (UEC), Trong-Thuc Hoang (UEC/AIST), Akira Tsukamoto (AIST), Kuniyasu Suzaki (TRASIO/AIST), and Cong-Kha Pham (UEC)

Outline

1. Introduction
2. Trusted Execution Environment
3. TEE-Hardware System
4. Crypto-cores Accelerators
5. Other Hardware Modules
6. Chip Results & Conclusion

Outline

1. Introduction
2. Trusted Execution Environment
3. TEE-Hardware System
4. Crypto-cores Accelerators
5. Other Hardware Modules
6. Chip Results & Conclusion

1. Introduction (1/3)

CHIPYARD

Open-sources framework for agile development of Chisel-based System-on-Chip

Berkeley Architecture Research has developed and open-sourced:

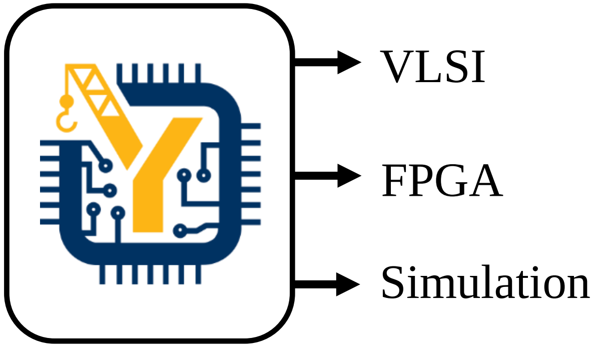
Chisel | BOOM Core | Diplomacy | FireSim | FIRRTL | Rocket Core | TileLink | Configuration System | RISC-V | Accelerators | Caches | Peripherals | HAMMER

Goal:

Make it easy for small teams to **design, integrate, simulate,** and **tape-out** a custom SoC

Berkeley Architecture Research

Fully customize hardware

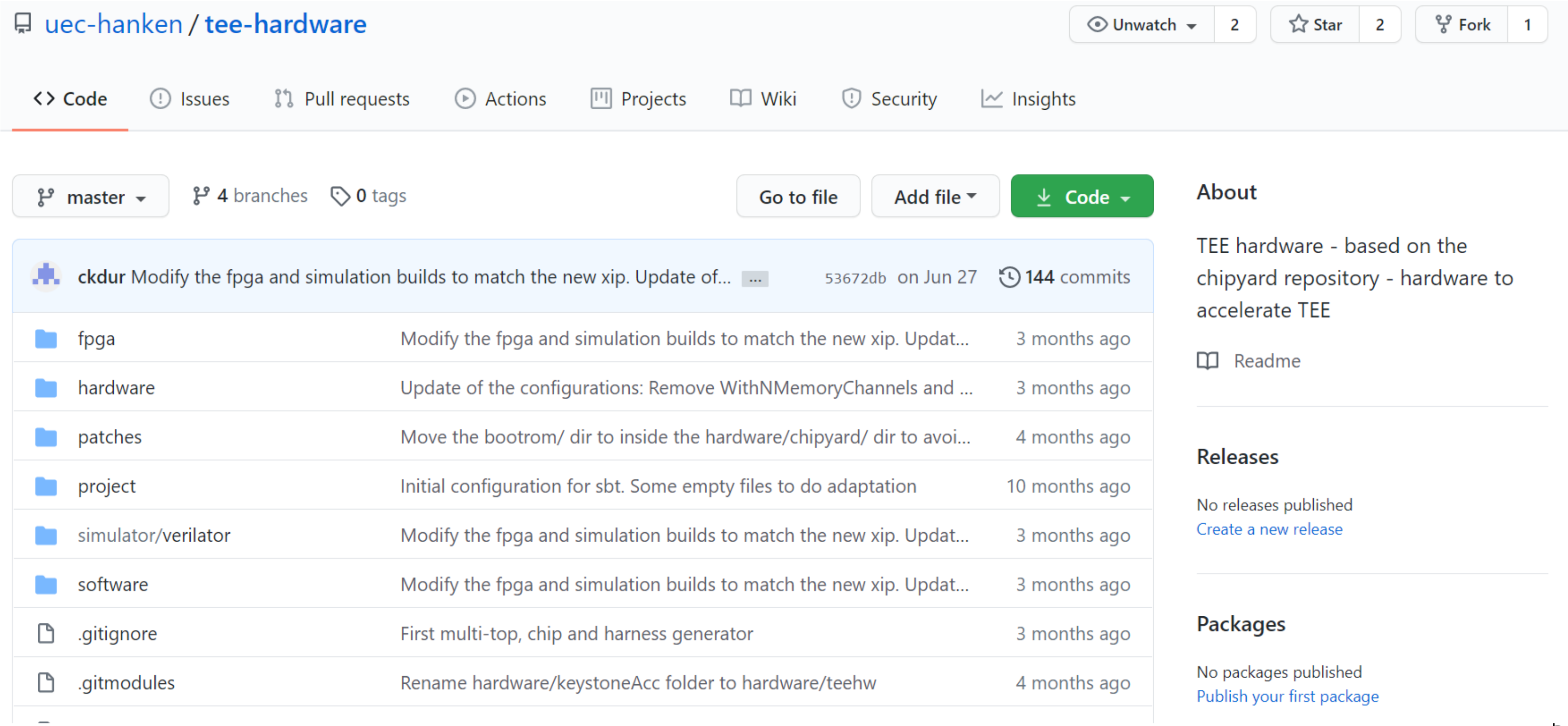


Perks:

- Most common RISC-V cores: Rocket-chip, BOOM, Arianne *(and updated frequently with the mainstream of those cores)*
- FPGA accelerators included *(Hwacha, Gemmini, NVDLA)*
- Simulation supported *(RTL: Verilator, FPGA: FireSim, VLSI: Hammer)*

1. Introduction (2/3)

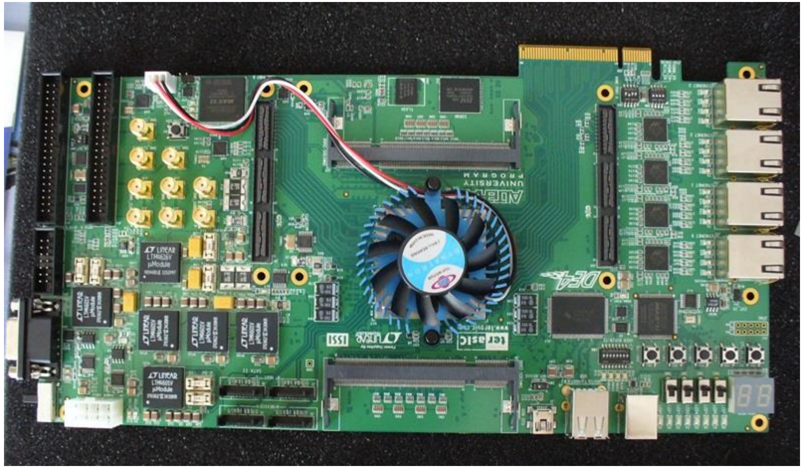
Based on Chipyard, a TEE-Hardware system is developed: https://github.com/uec-hanken/tee-hardware



1. Introduction (3/3)

TEE-HW has demos on:

Xilinx: VC707



Altera: DE4

Altera: TR4

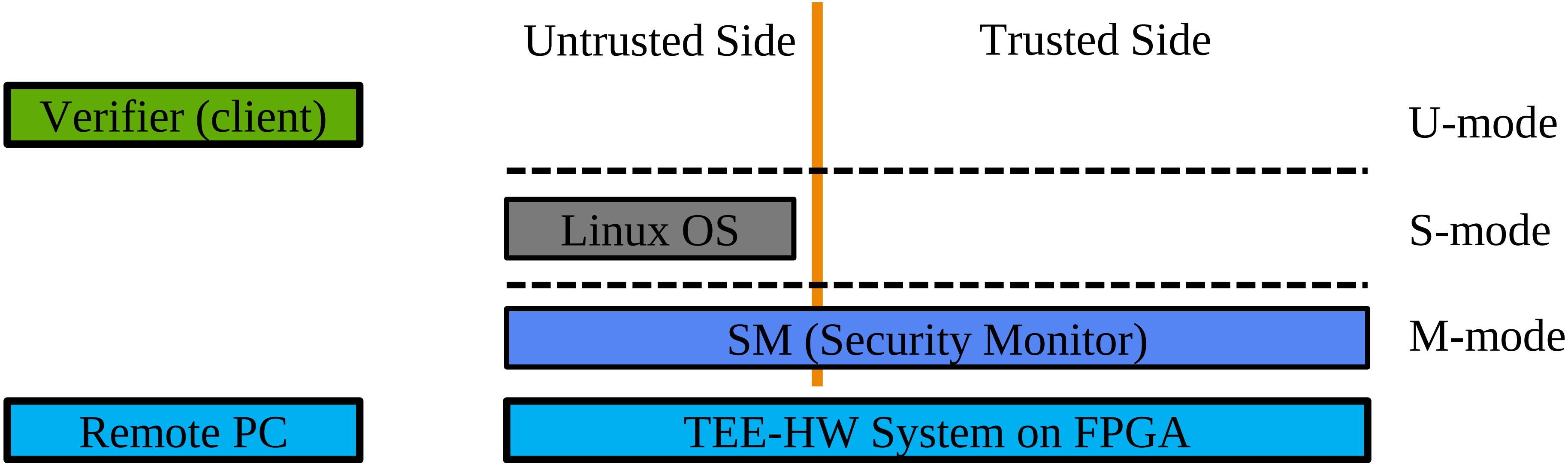
Outline

1. Introduction
2. Trusted Execution Environment
3. TEE-Hardware System
4. Crypto-cores Accelerators
5. Other Hardware Modules
6. Chip Results & Conclusion

2. Trusted Execution Environment (1/10)

TEE in-action *(using Keystone: A TEE Framework)*

Remote PC connects to FPGA via Serial *(UART)* terminal or a TCP connection

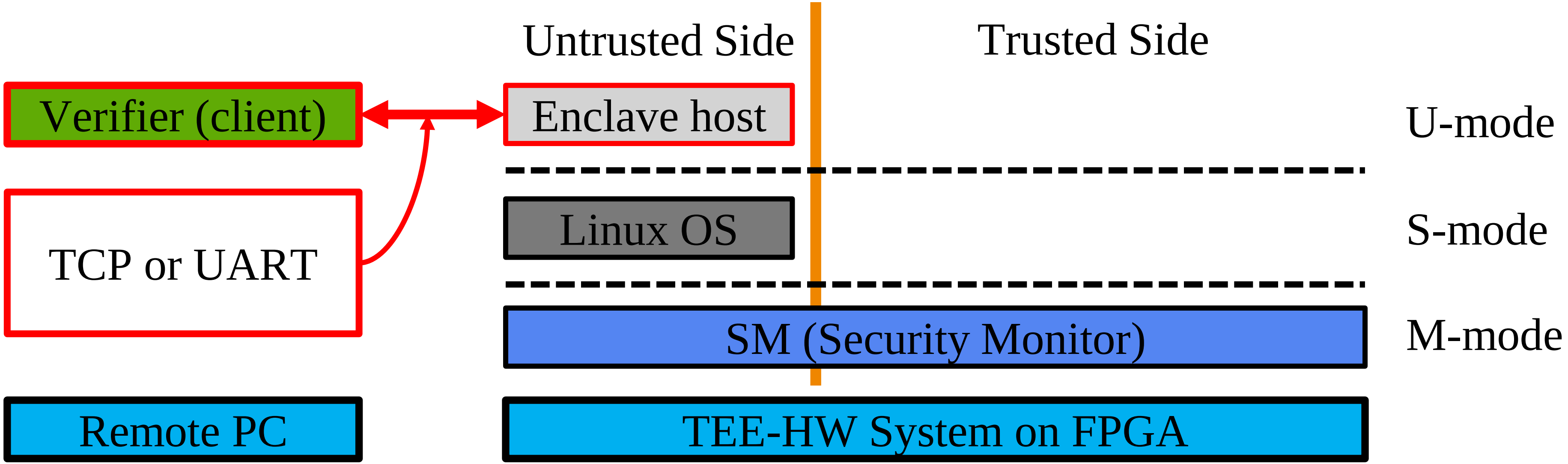


TEE *(Keystone in this case)* creates the Trusted-Side based on the chain-of-trust across multiple operating layers.
It allows client to create and operate an Enclave App in the Trusted Side.

2. Trusted Execution Environment (2/10)

TEE in-action *(using Keystone: A TEE Framework)*

Remote PC connects to FPGA via Serial *(UART)* terminal or a TCP connection

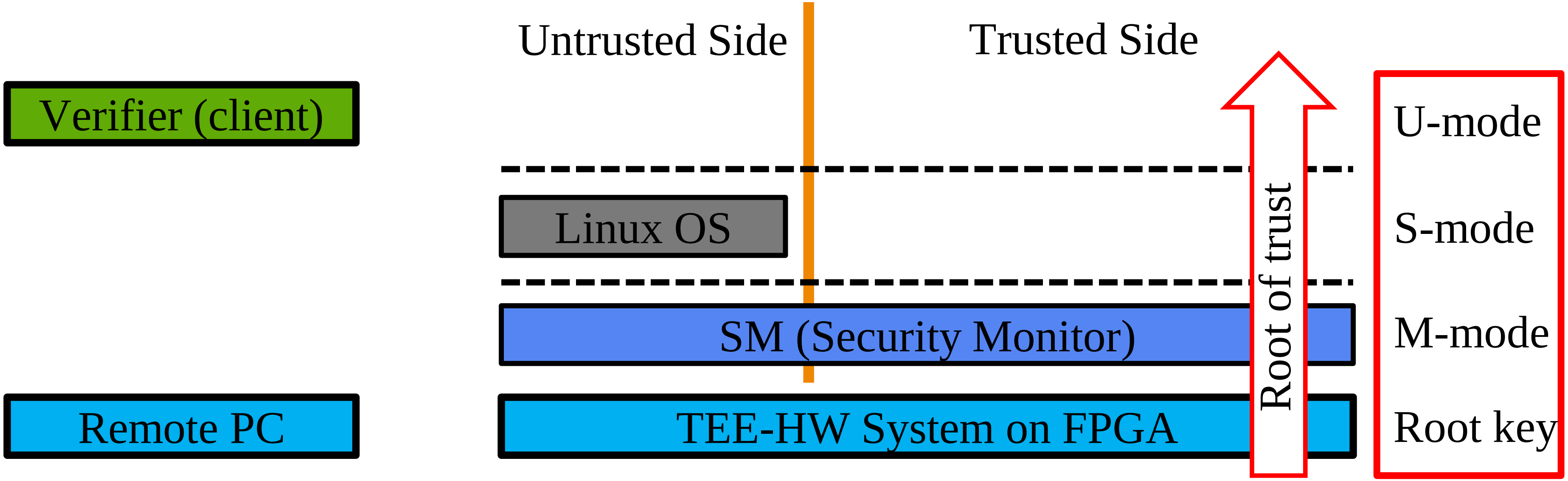


TEE *(Keystone in this case)* creates the Trusted-Side based on the chain-of-trust across multiple operating layers.

It allows client to create and operate an Enclave App in the Trusted Side.

TEE in-action *(using Keystone: A TEE Framework)*

Remote PC connects to FPGA via Serial *(UART)* terminal or a TCP connection

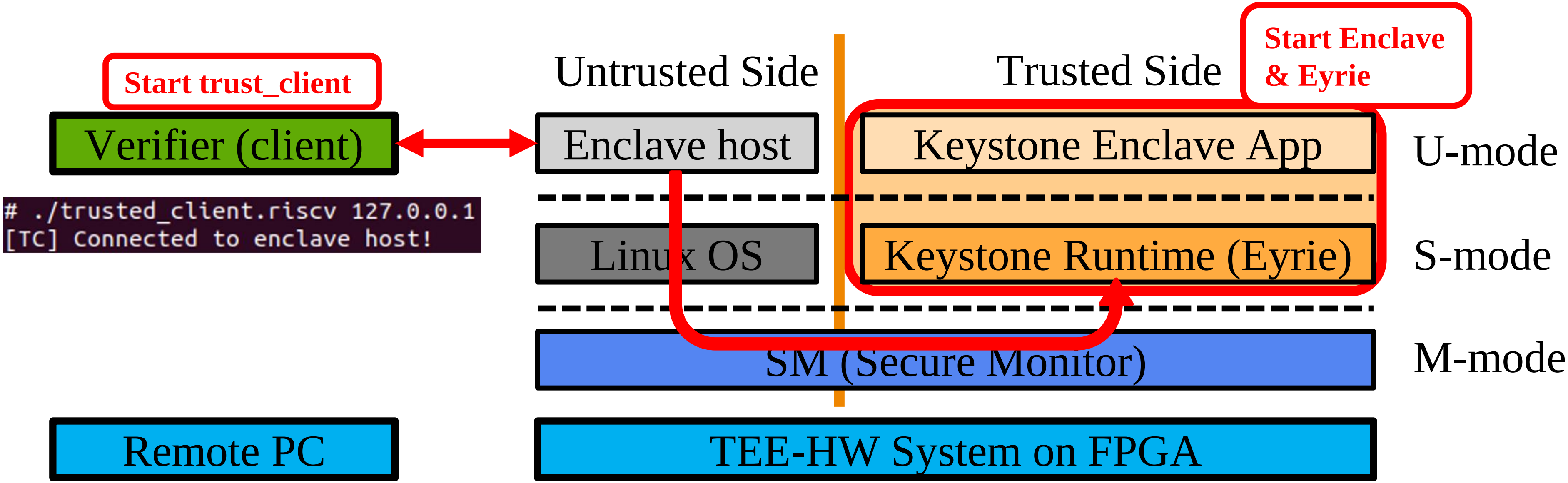


TEE *(Keystone in this case)* creates the Trusted-Side based on the chain-of-trust across multiple operating layers.

It allows client to create and operate an Enclave App in the Trusted Side.

2. Trusted Execution Environment (4/10)

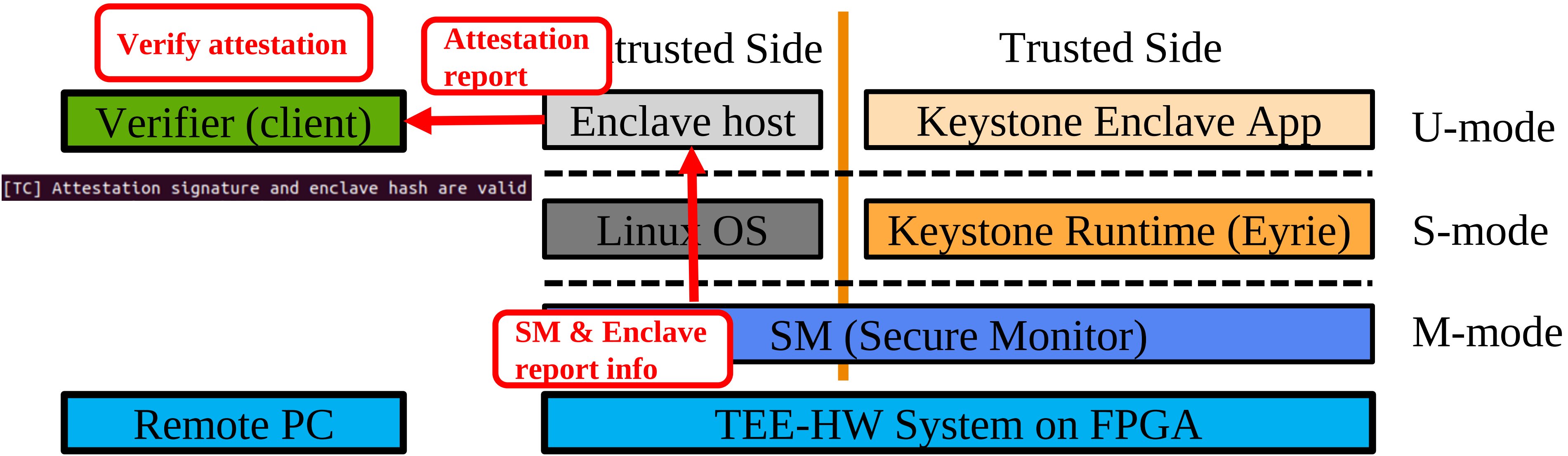
TEE in-action *(using Keystone: A TEE Framework)*



1. Connection with the Enclave host

2. Trusted Execution Environment (5/10)

TEE in-action *(using Keystone: A TEE Framework)*



1. Connection with the Enclave host
2. Verify attestation report

2. Trusted Execution Environment (6/10)

TEE in-action *(using Keystone: A TEE Framework)*

An example of attestation report:

```
[TC] Connected to enclave host!
                === Security Monitor ===
Hash: e56168887f2d0748cf7cd57c73b46c6a60fd8bcf80a852e4c134326efa6259f5c8c4a38543
514612d685baaf6de15edf330d4b74e7bf0f5405257029e79fcd20
Pubkey: cd98f4a28a8523ba8ecd31175aa0e2330b2f46e703454525466012 6a9f3b8cb9
Signature: d5c4b1647d444d5b76bd5ecf24ad5e774cb761b4a7a864c754683b1a8db8340adcfd6
ebf7da099d35ef7aef26834e5ffbdd86ca7815a6a6602cc4ee721a14f01

                === Enclave Application ===
Hash: 84b2193e0f5ec391672d9f68415fbf8a928e1a25b89dfb88257d7a3becf310229d8bed1534
5884f90f69792f6da237d8b6a9d55abe254f70b4181961989cbe7b
Signature: c443ec369888c4065dbaaaed9210f1d967bace5395cb3e679ca29a1aa8a8afa34ff2
80e597ac229b1a87d29acaec5d5db2b93c2ccd415b5909042a19a181b0e
Enclave Data: 8d571e0103e72ee6986407e67d5789dd8bc3318d663e519890f078f66b05ef57
                -- Device pubkey --
0faad4ff01178583baa588966f7c1ff32564dd17d7dc2b46cb50a84a69270b4c
```

SM hash

SM signature

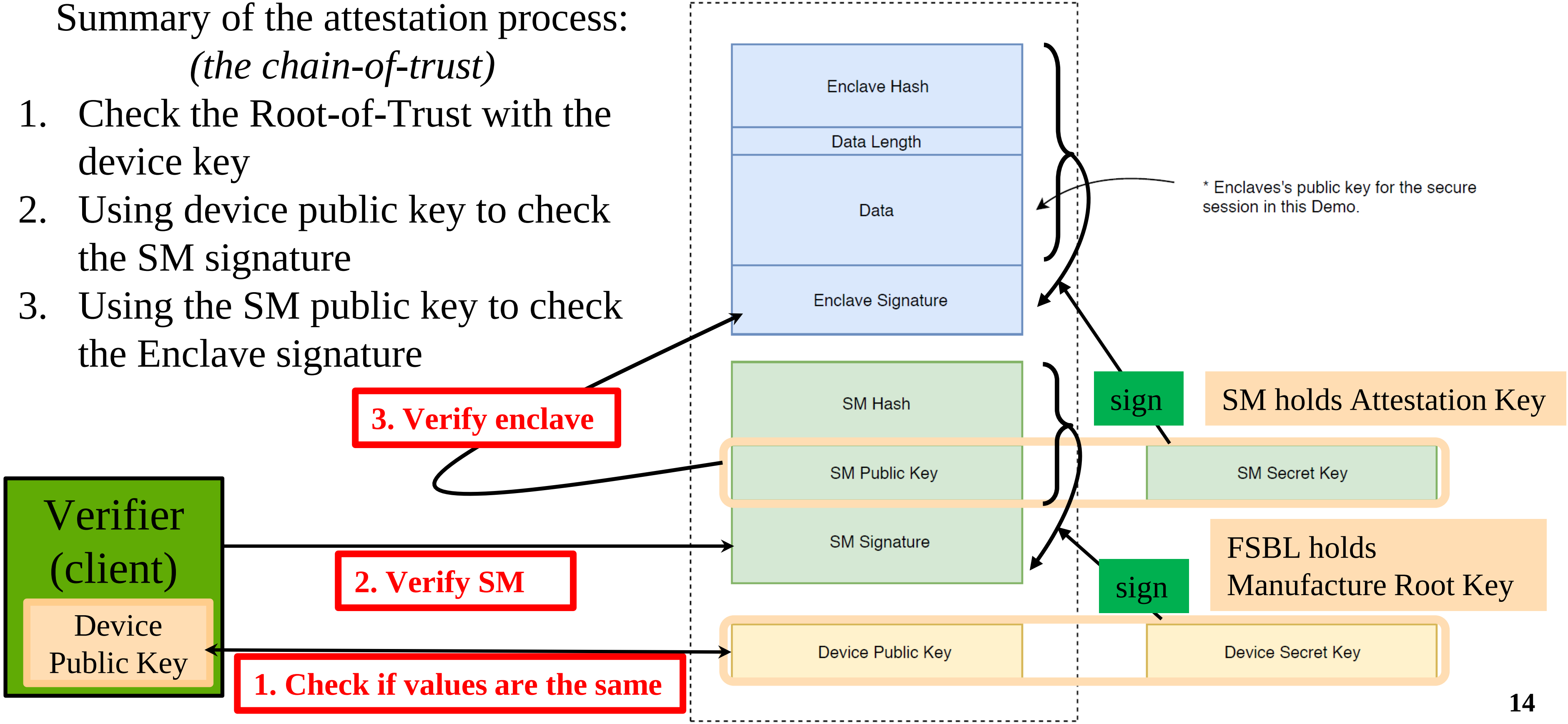
Enclave App hash

Device public key

TEE in-action *(using Keystone: A TEE Framework)*

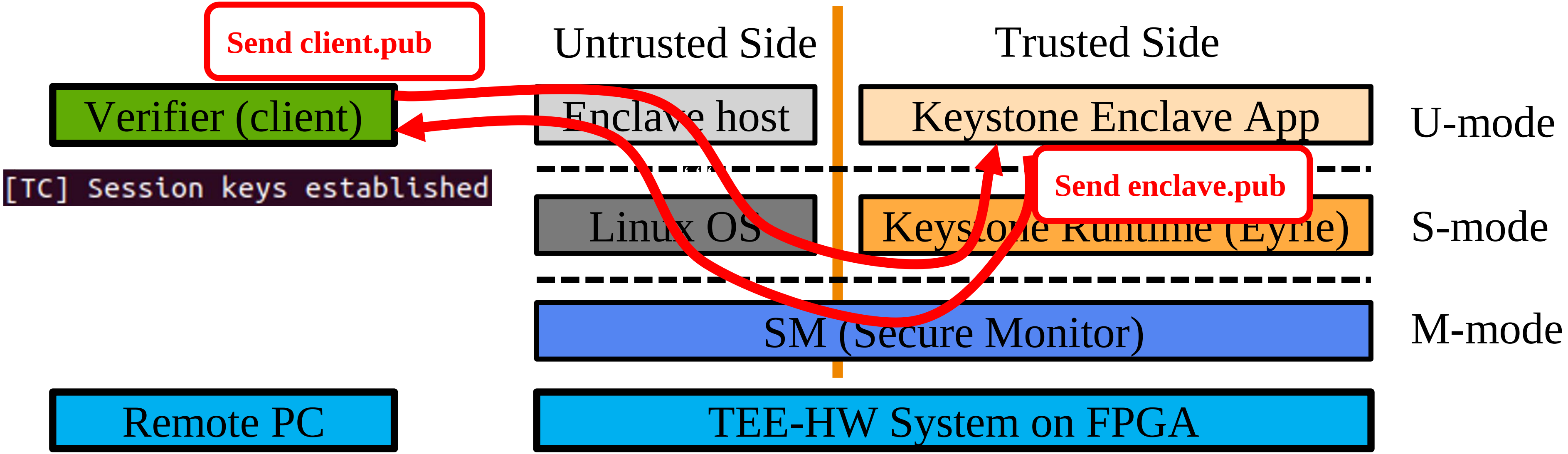
Summary of the attestation process:
(the chain-of-trust)

1. Check the Root-of-Trust with the device key
2. Using device public key to check the SM signature
3. Using the SM public key to check the Enclave signature



2. Trusted Execution Environment (8/10)

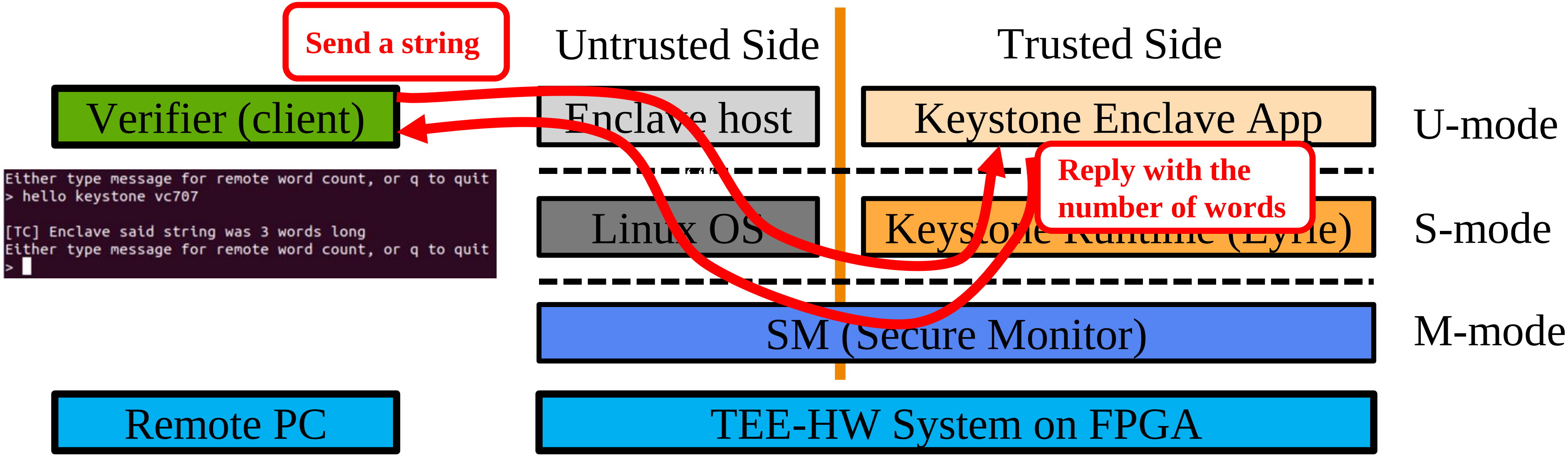
TEE in-action *(using Keystone: A TEE Framework)*



1. Connection with the Enclave host
2. Verify attestation report
3. Exchange communication keys

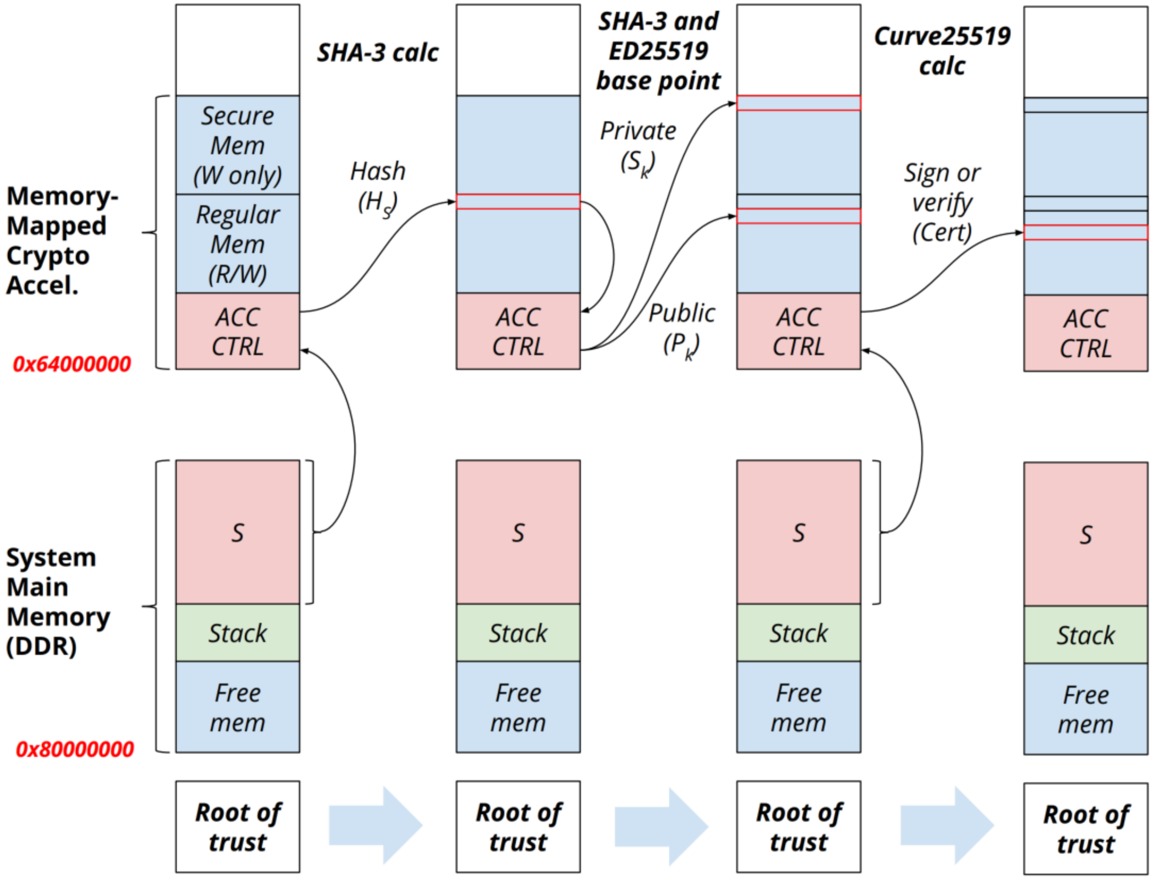
TEE in-action *(using Keystone: A TEE Framework)*

Keystone demo: (1) client sends strings, then (2) request calculation from the Enclave, finally (3) the Enclave replies with the number of words



1. Connection with the Enclave host
2. Verify attestation report
3. Exchange communication keys
4. Client’s app runs on the established TEE

TEE Secure Boot Sequence *(with HW Accelerators)*



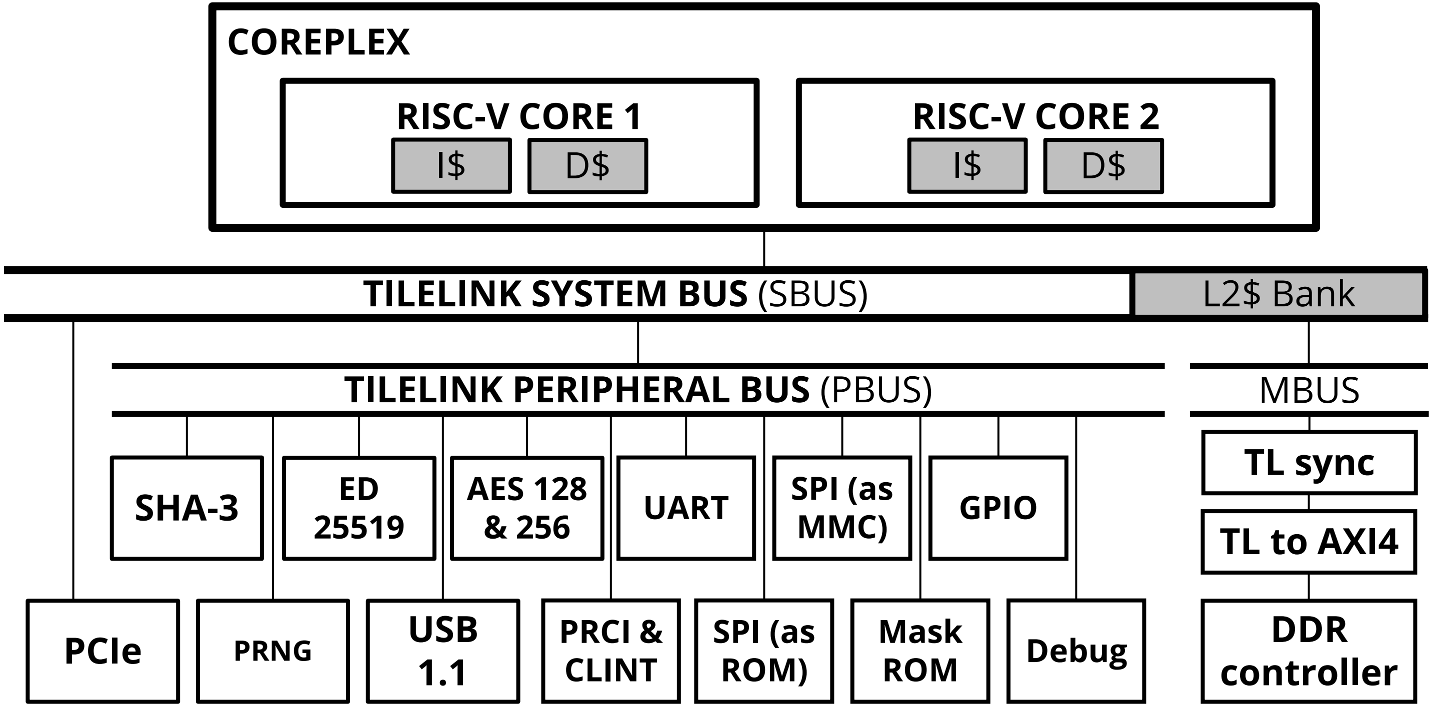
- The H_S value is automatically transferred between acts, thus it is not exposed to the software.
- The data in W-only memory are also not exposed to the software.

Outline

1. Introduction
2. Trusted Execution Environment
3. TEE-Hardware System
4. Crypto-cores Accelerators
5. Other Hardware Modules
6. Chip Results & Conclusion

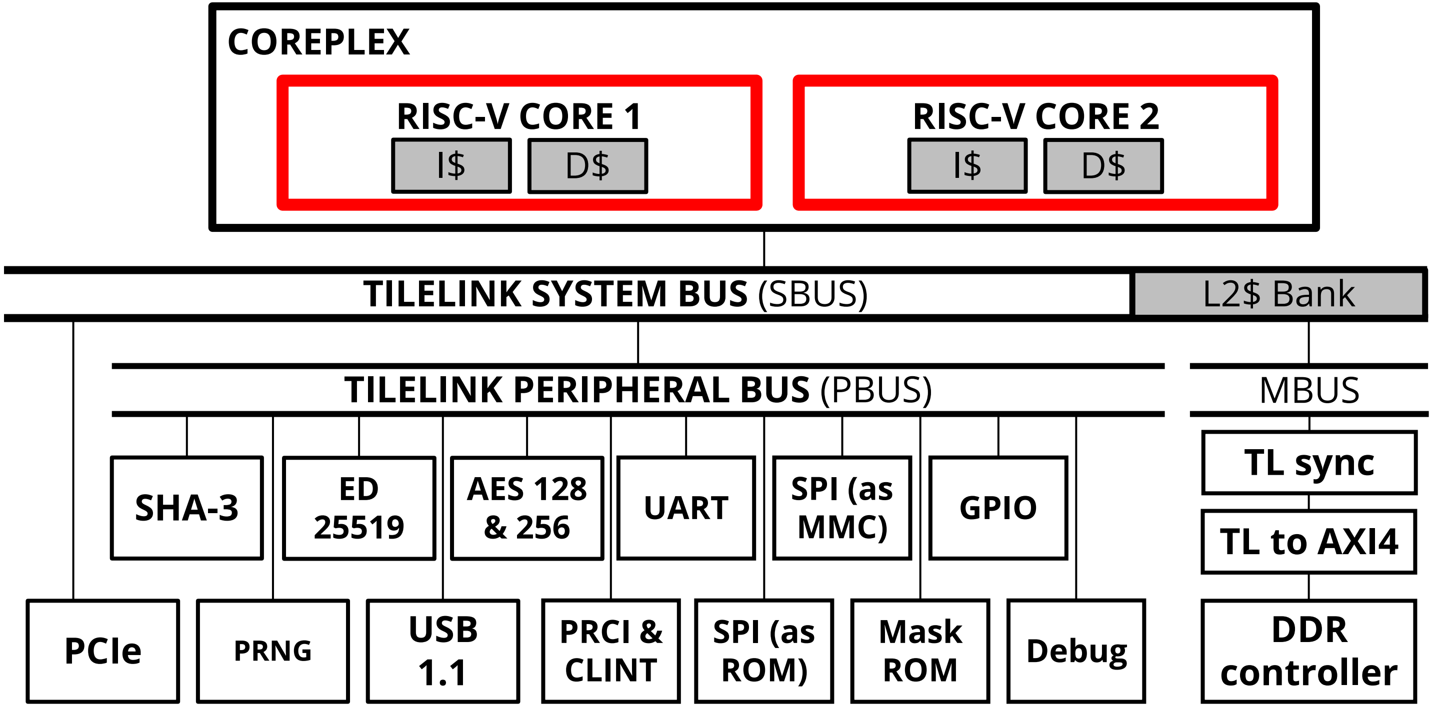
3. TEE-Hardware System (1/6)

System Architecture:



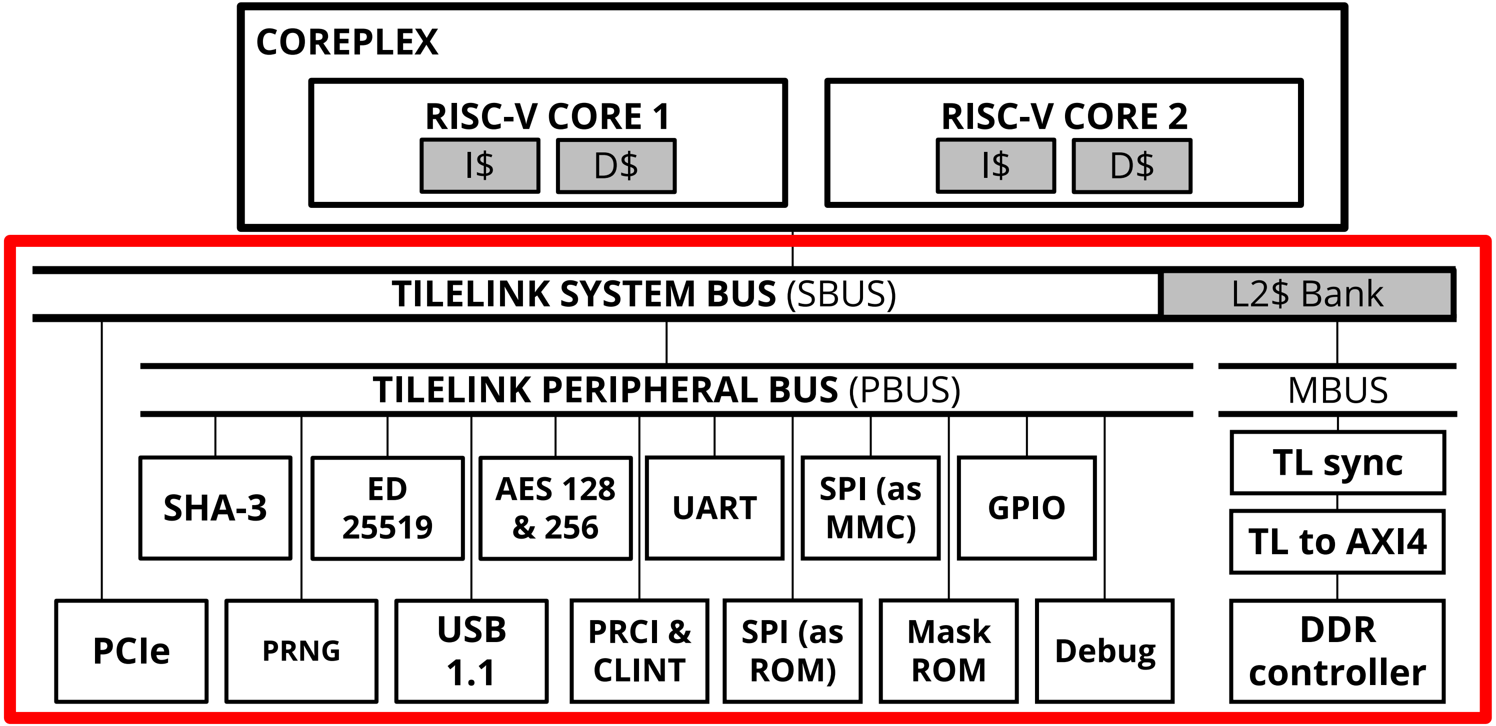
- Not fixed at dual-core, can increase/decrease the number of cores as you wanted *(as long as that fits the FPGA board)*
- Some hardware modules can be easily included/excluded to/from the system

3. TEE-Hardware System (2/6)



- Available cores in the system are **Rocket-chip** and **BOOM**
- Because BOOMv3 isn't stable yet, so both BOOMv2 and BOOMv3 are available on the GitHub with different branches.

3. TEE-Hardware System (3/6)



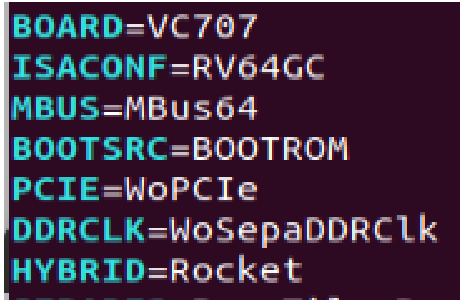
- System Bus (SBUS), Memory Bus (MBUS), and Peripheral Bus (PBUS) hierarchy.
- Several Peripheral devices for IO (GPIO, MMC, UART, PCIe, USB), memory (DDR, SPI ROM, Mask ROM), and Crypto-cores (SHA-3, ED25519, AES, PRNG)

3. TEE-Hardware System (4/6)

Variable	Available option	Description
BOARD	- VC707 - DE4 - TR4	Select the FPGA board
ISACONF	- RV64GC - RV64IMAC - RV32GC - RV32IMAC	Select the ISA
MBUS	- MBus64 - MBus32	Select the bit-width for the memory bus
BOOTSRC	- BOOTROM - QSPI	Select the boot source
PCIE	- WPCIe - WoPCIe	- Include PCIe module in the system - Remove PCIe module from the system
DDRCLK	- WSepaDDRClk - WoSepaDDRClk	- Separate DDR-clock with System-clock - Not separate DDR-clock with System-clock
HYBRID	- Rocket - Boom - RocketBoom - BoomRocket	- Two Rocket cores - Two Boom cores - Rocket core 1st, Boom core 2nd - Boom core 1st, Rocket core 2nd

In the Makefile system, these variables are available.

Example usage:



3. TEE-Hardware System (4/5)

TEE-HW with various core configurations

Boom

```
# cat /proc/cpuinfo
hart     : 0
isa      : rv64imafdc
mmu      : sv39
uarch    : ucb-bar,boom0

hart     : 1
isa      : rv64imafdc
mmu      : sv39
uarch    : ucb-bar,boom0
```

Rocket

```
# cat /proc/cpuinfo
hart     : 0
isa      : rv64imafdc
mmu      : sv39
uarch    : sifive,rocket0

hart     : 1
isa      : rv64imafdc
mmu      : sv39
uarch    : sifive,rocket0
```

BoomRocket

```
# cat /proc/cpuinfo
hart     : 0
isa      : rv64imafdc
mmu      : sv39
uarch    : ucb-bar,boom0

hart     : 1
isa      : rv64imafdc
mmu      : sv39
uarch    : sifive,rocket0
```

RocketBoom

```
# cat /proc/cpuinfo
hart     : 0
isa      : rv64imafdc
mmu      : sv39
uarch    : sifive,rocket0

hart     : 1
isa      : rv64imafdc
mmu      : sv39
uarch    : ucb-bar,boom0
```

RV64GC

```
# cat /proc/cpuinfo
hart     : 0
isa      : rv64imafdc
mmu      : sv39
uarch    : sifive,rocket0

hart     : 1
isa      : rv64imafdc
mmu      : sv39
uarch    : sifive,rocket0
```

RV64IMAC

```
# cat /proc/cpuinfo
hart     : 0
isa      : rv64imac
mmu      : sv39
uarch    : sifive,rocket0

hart     : 1
isa      : rv64imac
mmu      : sv39
uarch    : sifive,rocket0
```

RV32GC

```
# cat /proc/cpuinfo
hart     : 0
isa      : rv32imafdc
mmu      : sv32
uarch    : sifive,rocket0

hart     : 1
isa      : rv32imafdc
mmu      : sv32
uarch    : sifive,rocket0
```

RV32IMAC

```
# cat /proc/cpuinfo
hart     : 0
isa      : rv32imac
mmu      : sv32
uarch    : sifive,rocket0

hart     : 1
isa      : rv32imac
mmu      : sv32
uarch    : sifive,rocket0
```

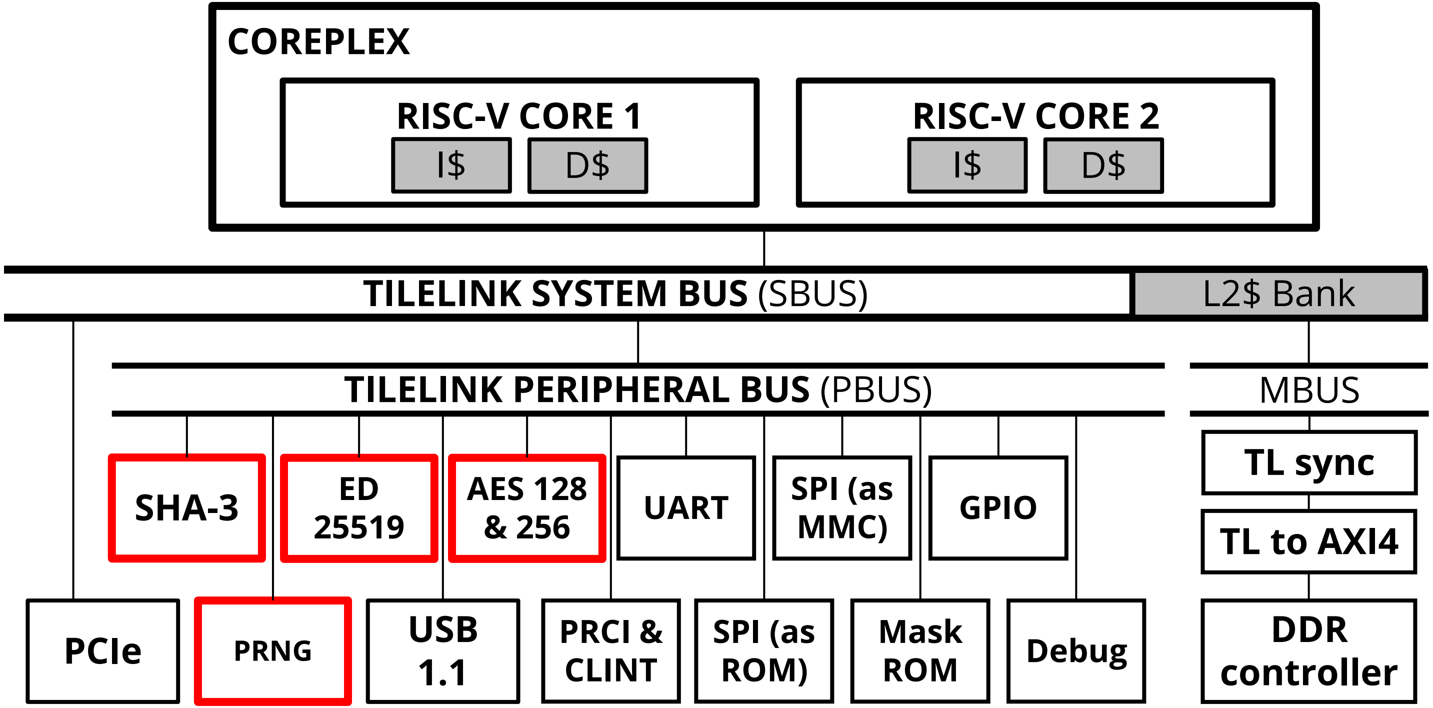
3. TEE-Hardware System (5/5)

Summary table of FPGA logic utilization *(on VC707)* with various core configurations:

<table>
<tr><th rowspan="2">ISACONF</th><th colspan="2">HYBRID</th><th colspan="2" rowspan="2">FPGA logic utilization
(LUT) (on VC707)</th></tr>
<tr><th>Core0</th><th>Core1</th></tr>
<tr><td rowspan="4">RV64GC</td><td>Boom</td><td>Boom</td><td>160,873</td><td>52.99%</td></tr>
<tr><td>Rocket</td><td>Rocket</td><td>96,571</td><td>31.81%</td></tr>
<tr><td>Boom</td><td>Rocket</td><td>128,708</td><td>42.39%</td></tr>
<tr><td>Rocket</td><td>Boom</td><td>128,719</td><td>42.40%</td></tr>
<tr><td>RV64GC</td><td rowspan="4">Rocket</td><td rowspan="4">Rocket</td><td>96,571</td><td>31.81%</td></tr>
<tr><td>RV64IMAC</td><td>72,007</td><td>23.72%</td></tr>
<tr><td>RV32GC</td><td>89,356</td><td>29.43%</td></tr>
<tr><td>RV32IMAC</td><td>65,899</td><td>21.71%</td></tr>
</table>

Outline

4. Crypto-core Accelerators (1/6)



Crypto-cores:

- SHA-3 512
- AES-128/256
- Ed25519 *(genkey and signature)*
- PRNG *(Pseudo-random generator)*

Some feature notes

- Crypto-Core can be implemented as a custom instruction (ROCC)
- AES supports on-the-fly 128 and 256 bits, and can be changed
- Ed25519 contains:
  - Ed25519-Mult for pair-key generation
  - Ed25519-Sign for signature verification
- PRNG uses LFSR *(Linear-Feedback Shift Register)*; and is based on ARM TrustZone RNG register model

Crypto-cores on Stratix-IV FPGA

	SHA-3	AES-128/256	Ed25519	
			Mult	Sign
ALUT	8,108	3,195	2,737	3,969
Registers	2,790	2,854	4,778	4,617
Fmax (MHz)	100	100	100	100
Memory	0	0	8KB	0
DSP block	0	0	48	130
Total (%)	**1.1**	**0.6**	**3.3**	**5.9**

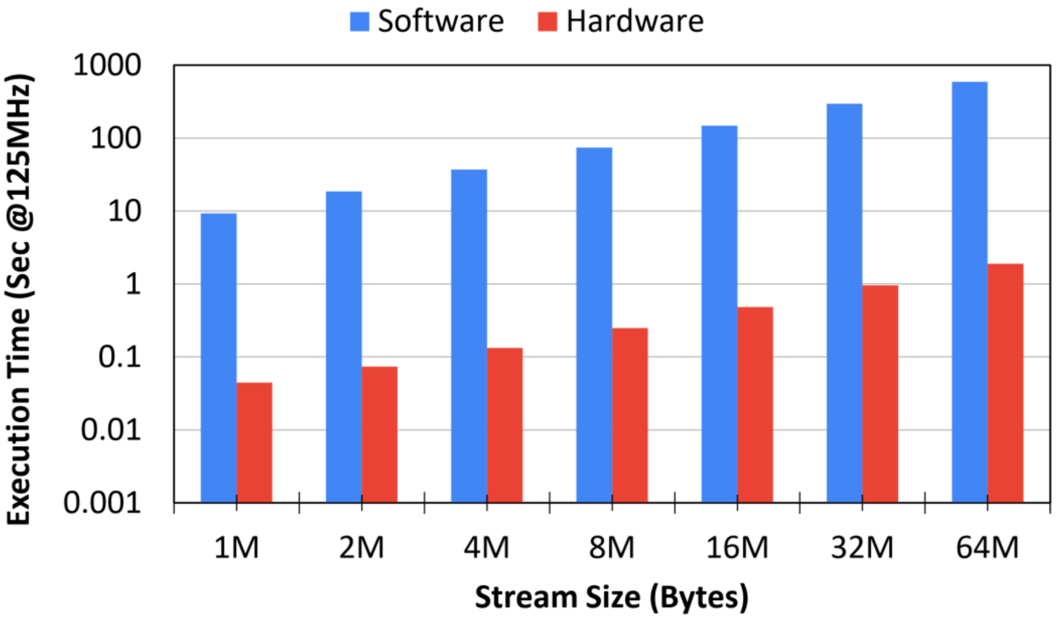
Crypto-cores in ASIC *(ROHM-180nm)*

	SHA-3	AES-128/256	Ed25519	
			Mult	Sign
Size	1,150µm × 1,150µm	808.96µm × 806.4µm	1,694.72µm × 1,693.44µm	1,346.56µm × 1,345.68µm
Gate-count (NAND)	102,500	50,560	222,432	140,442
Fmax (MHz)	104	90	106	91
Power (mW)	42.745	37.566	53.061	80.894

4. Crypto-core Accelerators (5/6)

The result of using crypto-core hardware accelerators *(applying at boot stage)*

The test was done on Stratix-IV FPGA with Rocket-chip RV64GC core



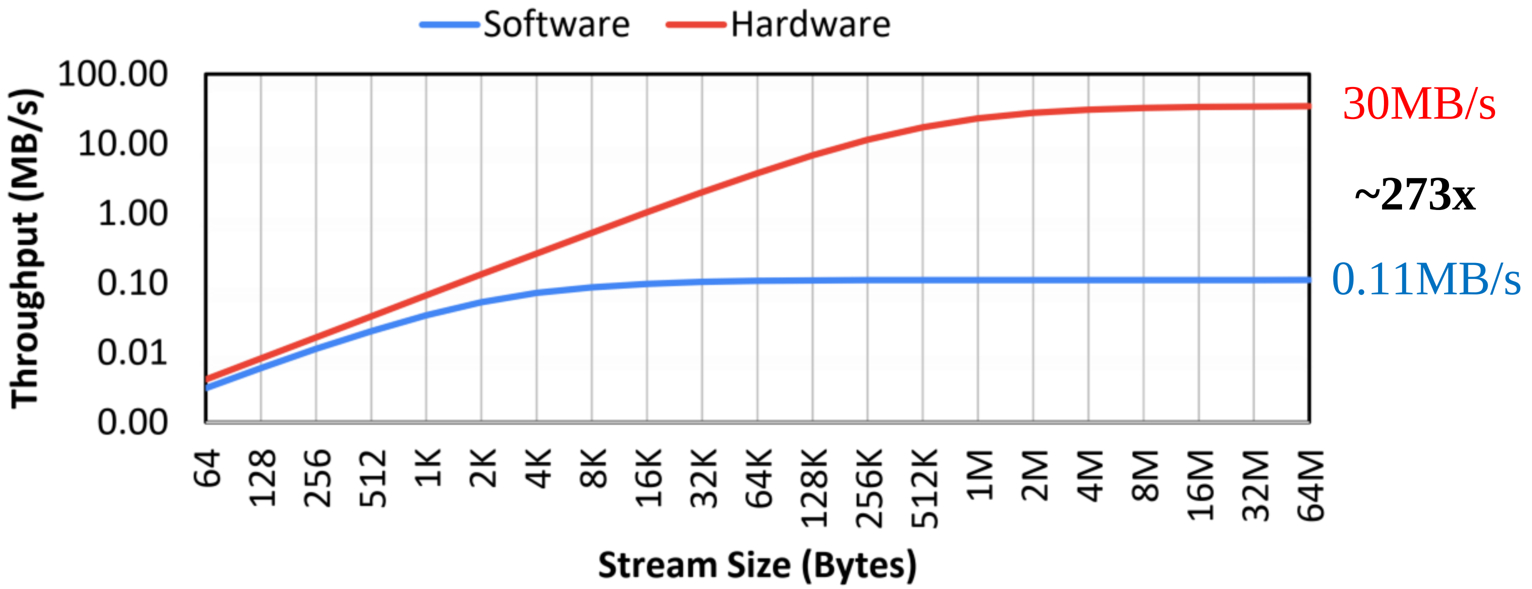
Software vs. hardware of SHA-3 execution times in the TEE framework.

Hardware is faster about 2.5 decades

4. Crypto-core Accelerators (6/6)

The result of using crypto-core hardware accelerators *(applying at boot stage)*

The test was done on Stratix-IV FPGA with Rocket-chip RV64GC core

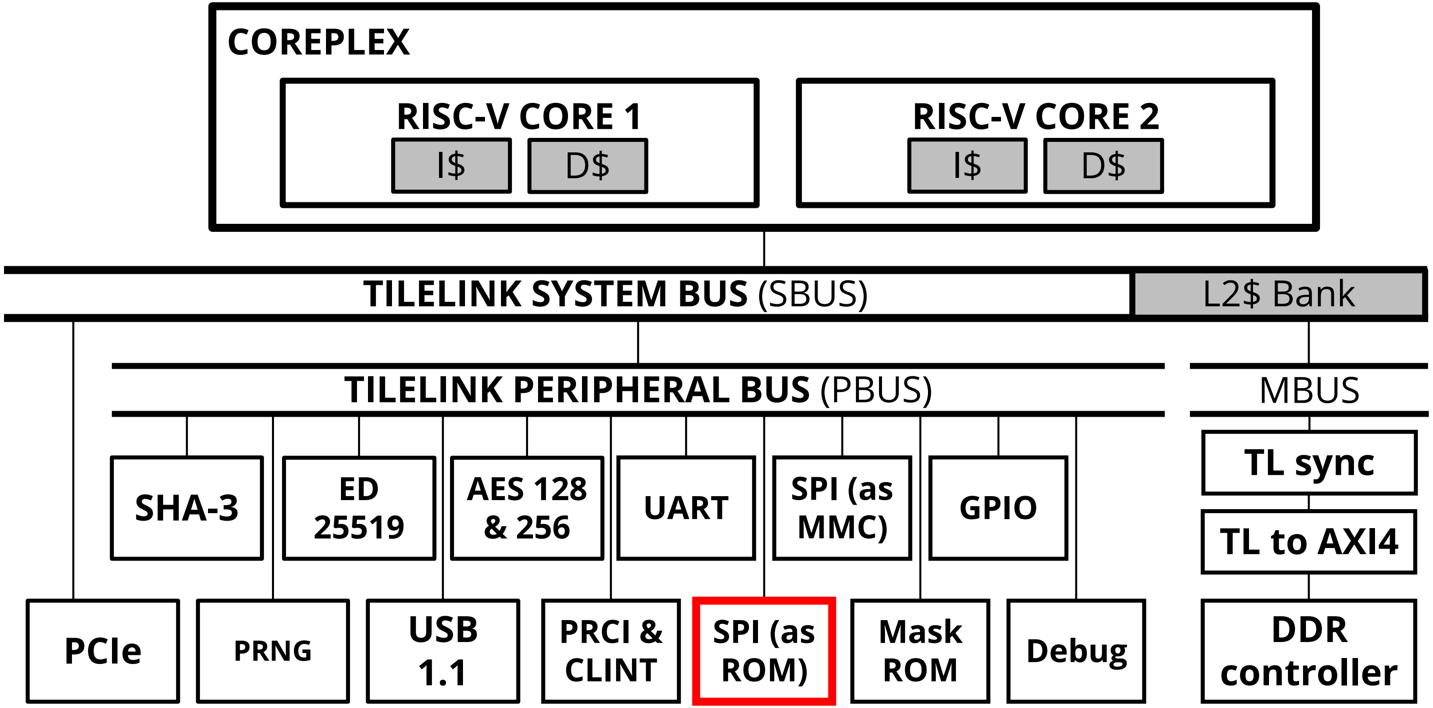


Software vs. hardware of SHA-3 operation throughput.

Outline

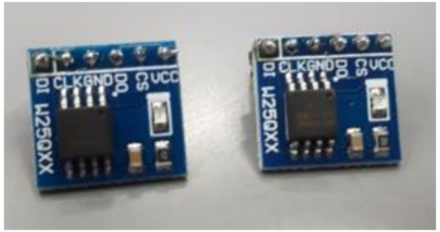
1. Introduction
2. Trusted Execution Environment
3. TEE-Hardware System
4. Crypto-cores Accelerators
5. Other Hardware Modules
6. Chip Results & Conclusion

5. Other Hardware Modules (1/4)



QSPI: to use Flash outside

5. Other Hardware Modules (2/4)



Flash modules
(cheap, bundle, and easy to plug-in with FPGA boards)

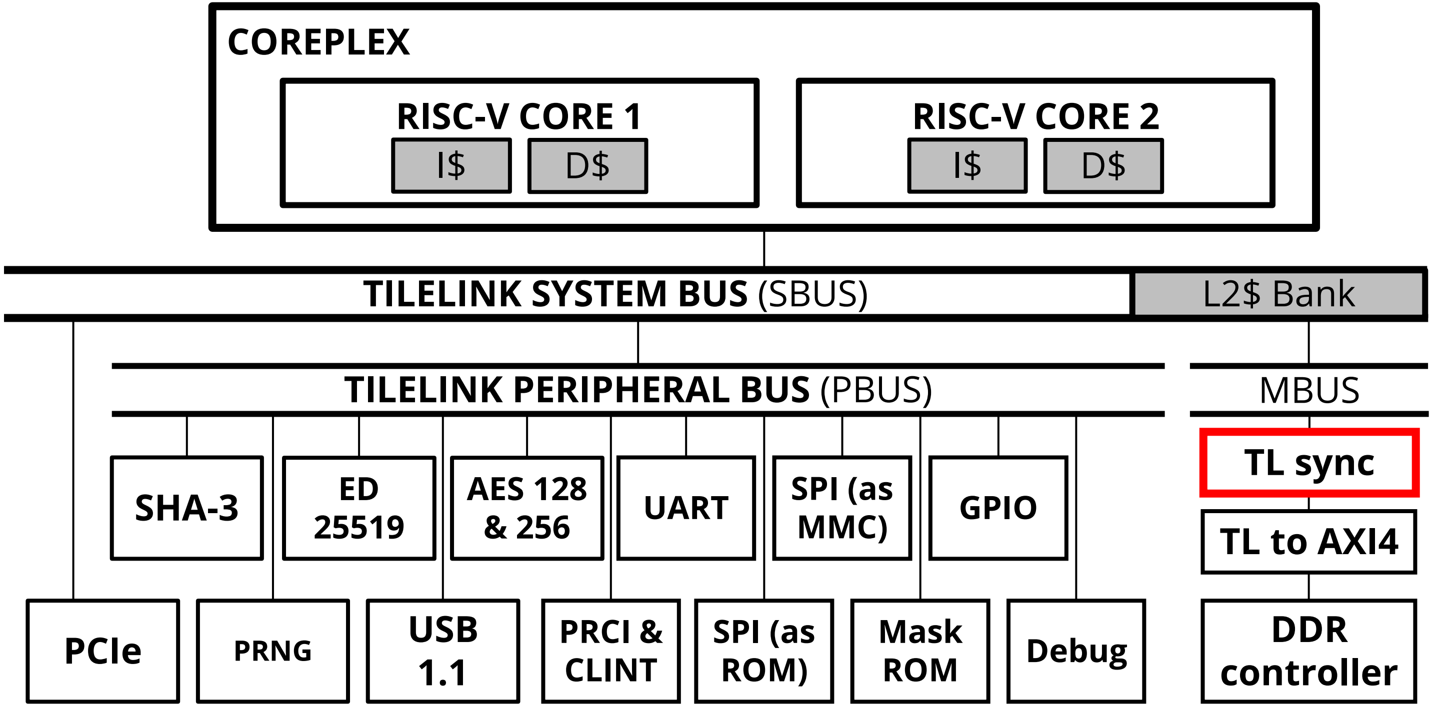
```
class BOOTROM extends Config((site, here, up) => {
  case PeripheryMaskROMKey => List(
    MaskROMParams(address = BigInt(0x20000000), depth = 2048, name = "BootROM"))
  case PeripherySPIFlashKey => List() // disable SPIFlash
})

class QSPI extends Config((site, here, up) => {
  case PeripheryMaskROMKey => List( //move BootROM back to 0x10000
    MaskROMParams(address = 0x10000, depth = 16, name = "BootROM")) //smallest allowed depth is 16
  case PeripherySPIFlashKey => List(
    SPIFlashParams(fAddress = 0x20000000, rAddress = 0x64005000, defaultSampleDel = 3))
})
```

- BOOTROM scenario:
  - Disable QSPI
  - BootROM at `0x20000000`, ZSBL in BootROM
- QSPI scenario:
  - Enable QSPI at `0x20000000`, ZSBL now in Flash
  - BootROM moved back to `0x10000`, in BootROM now just a simple instruction to jump directly to `0x20000000`

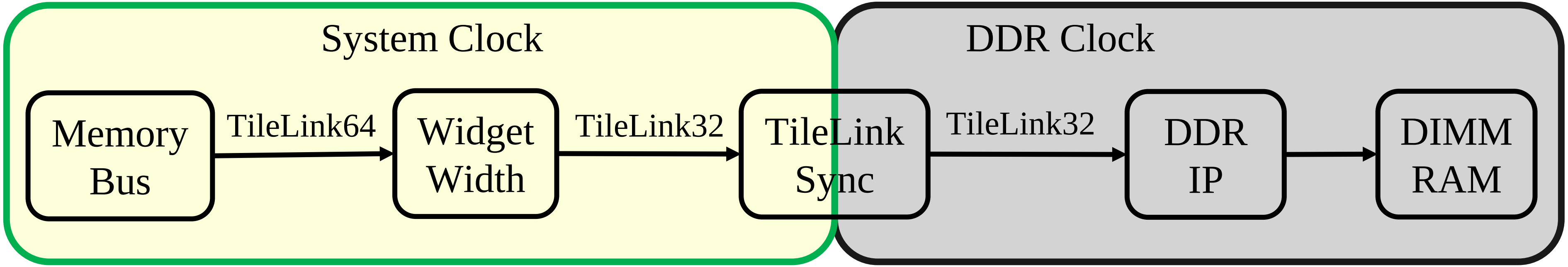
Easy to on/off the using of QSPI

5. Other Hardware Modules (3/4)



TileLink Sync: synchronize between different clock domains

5. Other Hardware Modules (4/4)



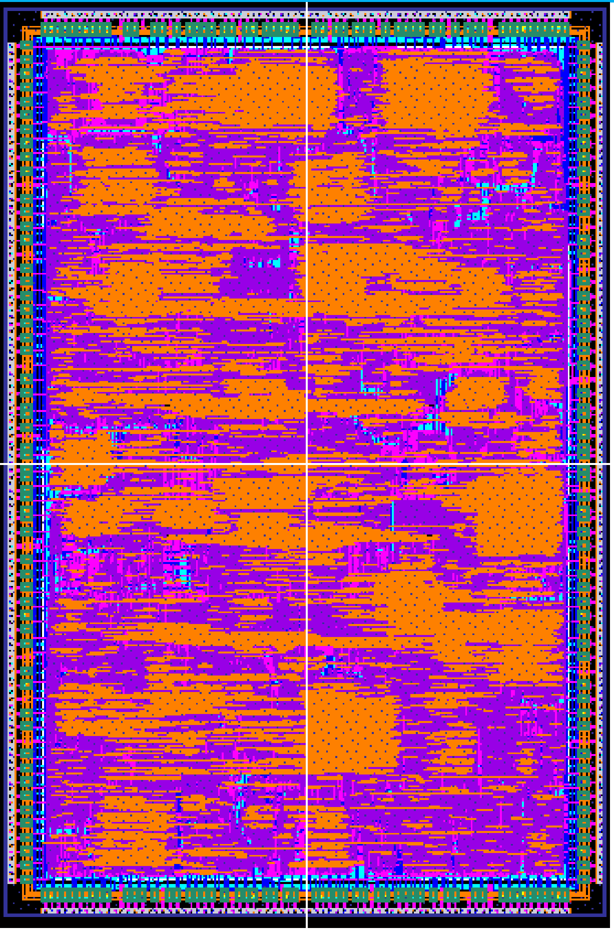
Separate the inner system clock with outer DDR clock:

- Sometime inner system cannot run at high-speed
  → System-clock < DDR-clock
  → Keep the DDR bandwidth still at high-speed
- Sometime *(depends on board)* DDR IP is fixed at lower clock rate *(for example, 100MHz)* than the CPU *(for example, 125MHz)*
  → System-clock > DDR-clock
  → Keep the CPU runs at higher clock rate

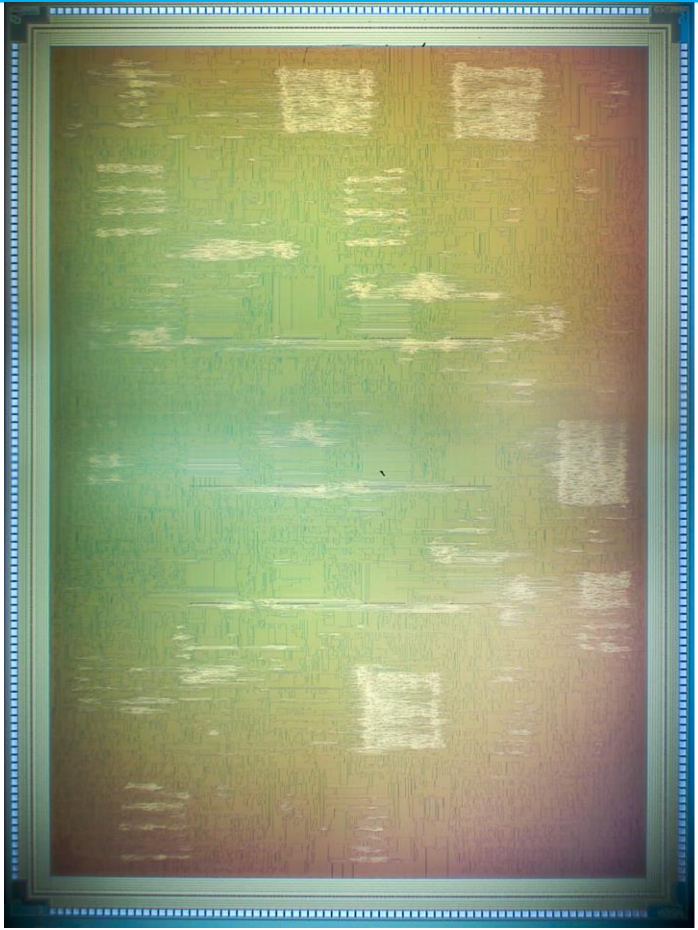
Outline

1. Introduction
2. Trusted Execution Environment
3. TEE-Hardware System
4. Crypto-cores Accelerators
5. Other Hardware Modules
6. Chip Results & Conclusion

6. Conclusion (1/4)



Layout



Barechip

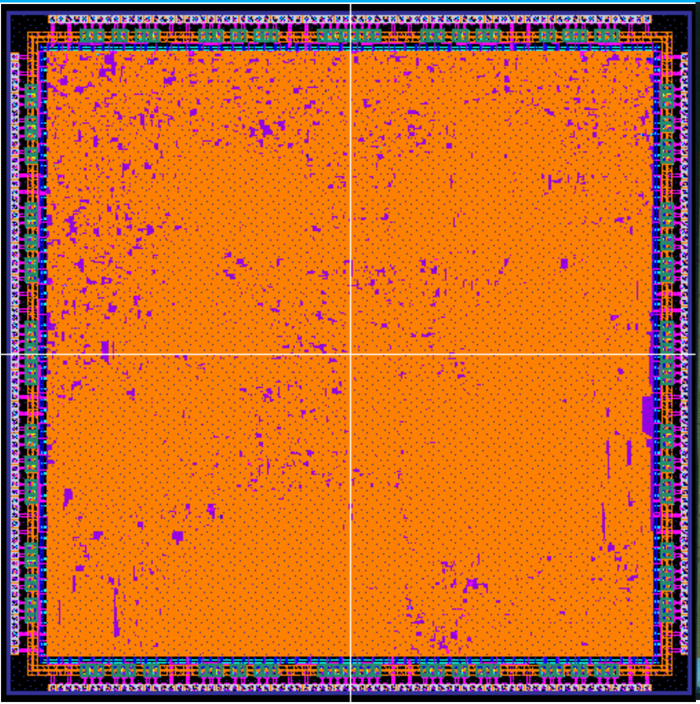
Features

- Cores: Rocket-chip (**x4**)
- ISA: RV64GC
  (crypto-cores aren't included)

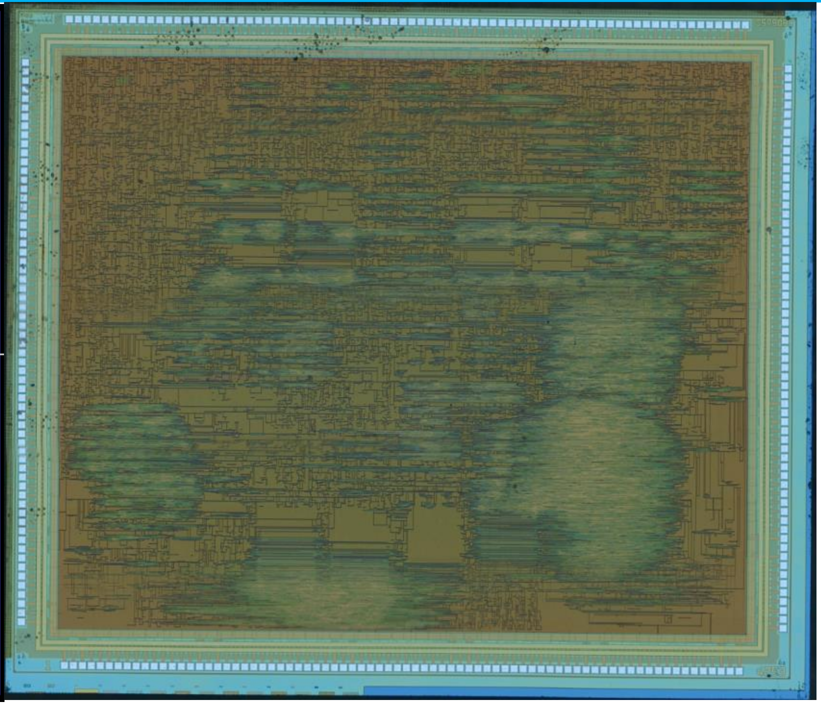
- Size: 4,512μm × 7,172μm
- Fmax: 92 MHz
- Power: 391.125 mW

- Process: ROHM 180nm
- Fabricate: 10/2019

6. Conclusion (2/4)



Layout



Barechip

Features

- Core: Rocket-chip (**x2**)
- ISA: RV64GC

- Crypto-cores: SHA3-512, AES-128/256, Ed25519 *(both Mult and Sign)*
- Other: QSPI *(for Flash)*, USB1.1

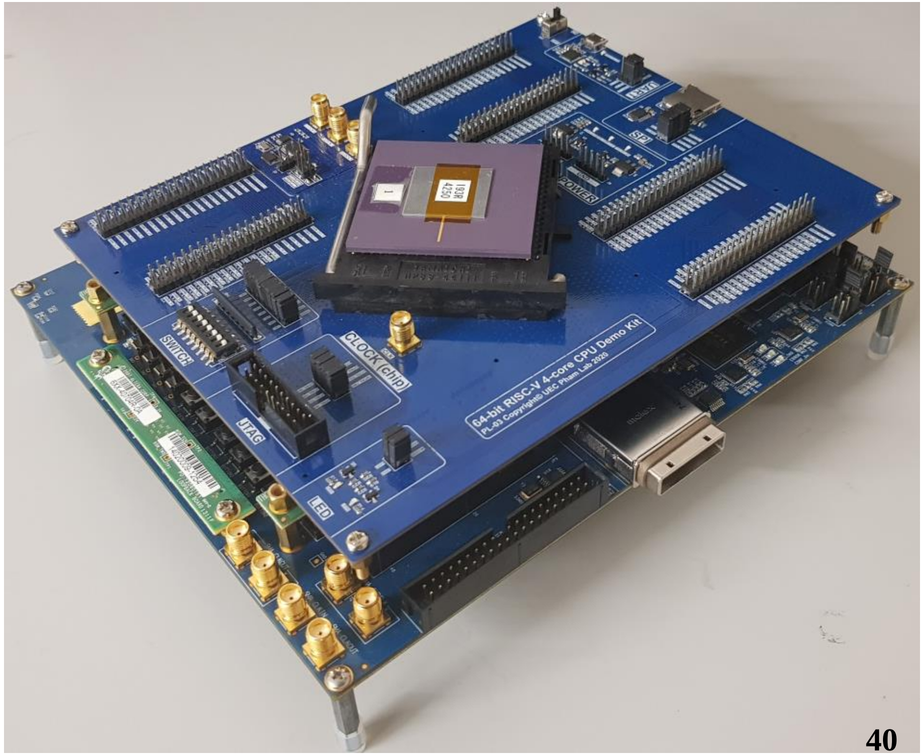
- Size: $4{,}573\mu m \times 4{,}578\mu m$
- Fmax: 98 MHz
- Power: 706.635 mW

- Process: ROHM 180nm
- Fabricate: 01/2020

6. Conclusion (3/4)

Solving the DDR problem for the chip by:

1. Using the DIMM RAM in the TR4
2. Having the PCB *(with socket-chip)* mounted on the TR4



6. Conclusion (4/4)

- We presented a system platform for Trusted Execution Environment (TEE) featuring crypto-cores accelerators.
- Completed TEE-Hardware system was developed with various configurations to fit specific needs; such as core options, hybrid options, ISA options, etc.
- The system was implemented and tested on various FPGAs *(VC707*, *DE4*, *TR4)* and ASIC *(ROHM-180nm)*.
- The execution time of the TEE with hardware accelerators dropped significantly compared to software.

Acknowledge

- The presented work is based on results obtained from a project (JPNP16007) commissioned by the New Energy and Industrial Technology Development Organization (NEDO).

国立大学法人 電気通信大学

UEC TOKYO

NATIONAL INSTITUTE OF ADVANCED INDUSTRIAL SCIENCE AND TECHNOLOGY (AIST)

TRASIO

Technology Research Association of Secure IoT Edge application based on RISC-V Open architecture

THANK YOU FOR YOUR LISTENING