High-efficiency protection solution for off-chip memory in embedded systems

Vaslin Romain[1], Guy Gogniat[1], Jean-Philippe Diguet[1], Eduardo Wanderley[1], Russell Tessier[2], Wayne Burleson[2]

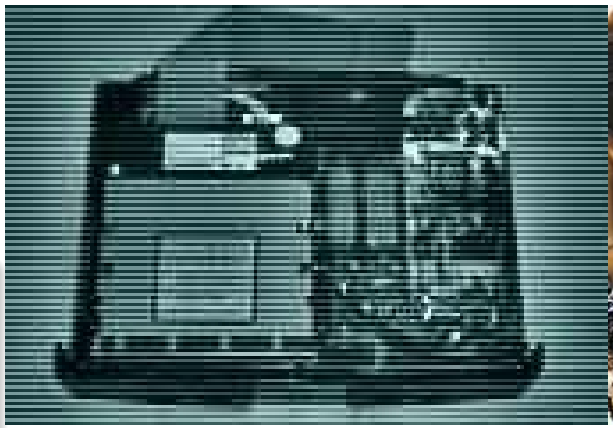
[1]LESTER UBS/CNRS FRE 2734 – University of South Brittany
56123 Lorient FRANCE

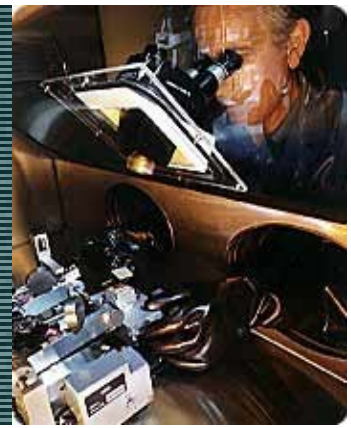
[2]ECE department – University of Massachusetts
01003 Amherst USA

Cryptarchi 2007

Introduction

Security in embedded systems : essential issue for external communication and architecture core





New threats on embedded systems :

- Hardware attacks

Introduction

Security in embedded systems : essential issue for external communication and architecture core

New threats on embedded systems :

- Hardware attacks
- Software attacks

Introduction

Security in embedded systems : essential issue for external communication and architecture core

New threats on embedded systems :

- Hardware attacks
- Software attacks

New adapted solutions :

- Architecture solutions
- Constraint requirements

Outline

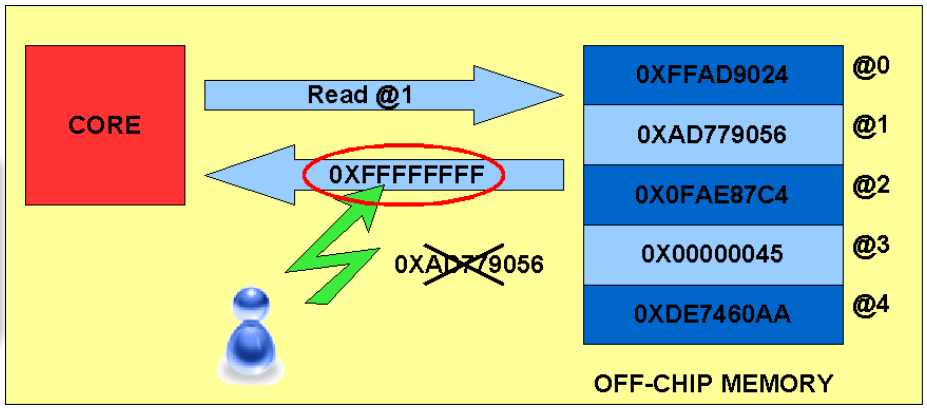
1. Threat model & common solutions
   - Targeted threats
   - Some solutions
2. Extended OTP solution
   - One-Time-Pad architecture
   - Extended OTP latency standpoint
3. Experiments & results
   - Cost of security
   - Comparison with previous solutions

Outline

Targeted threats

Targeted attacks

- Spoofing attacks

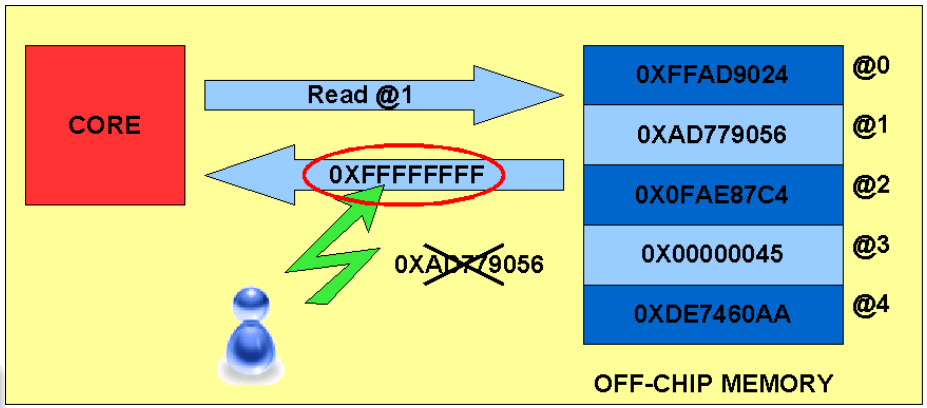


Spoofind attack

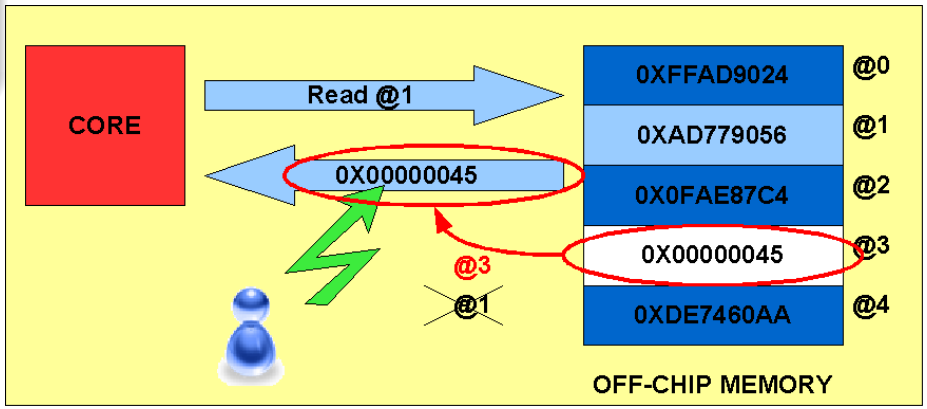
Targeted threats

Targeted attacks

- Spoofing attacks
- Relocation attacks



Spoofind attack

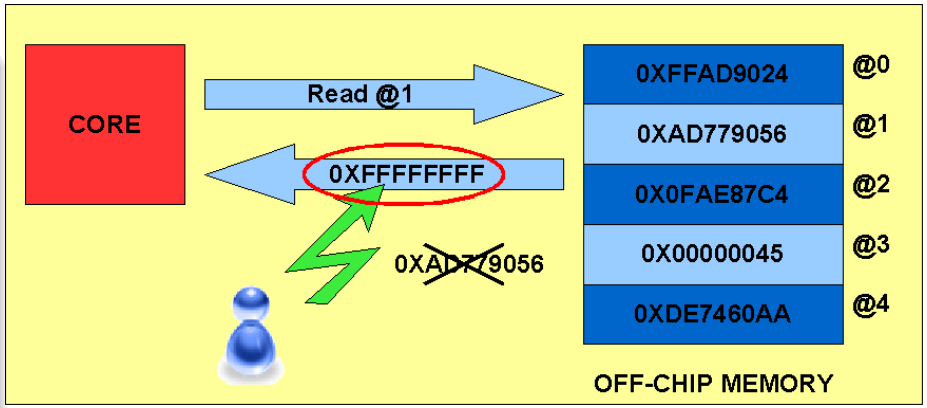


Relocation attack

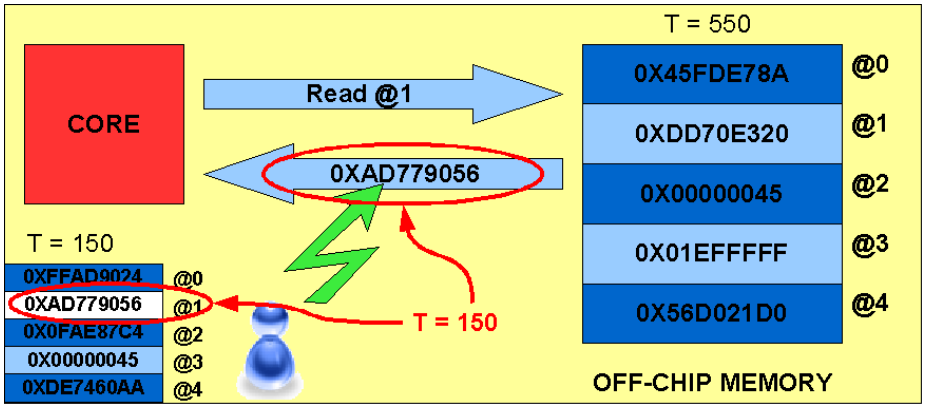
Targeted threats

Targeted attacks

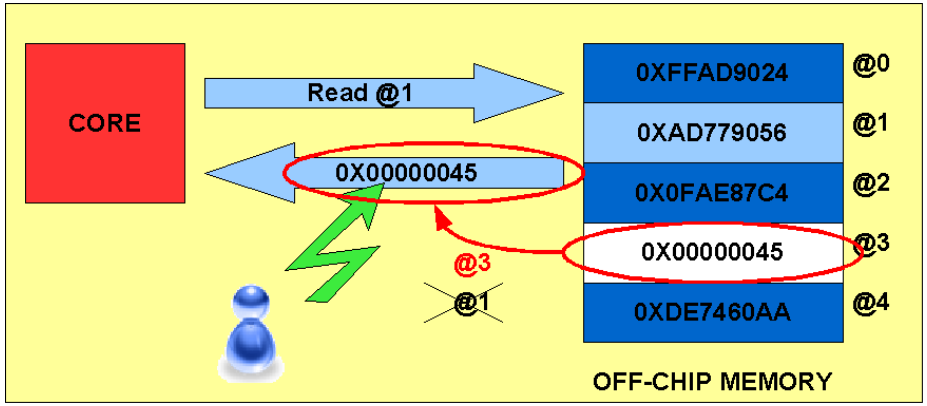
- Spoofing attacks
- Relocation attacks
- Replay attacks



Spoofind attack



Replay attack



Relocation attack

Outline

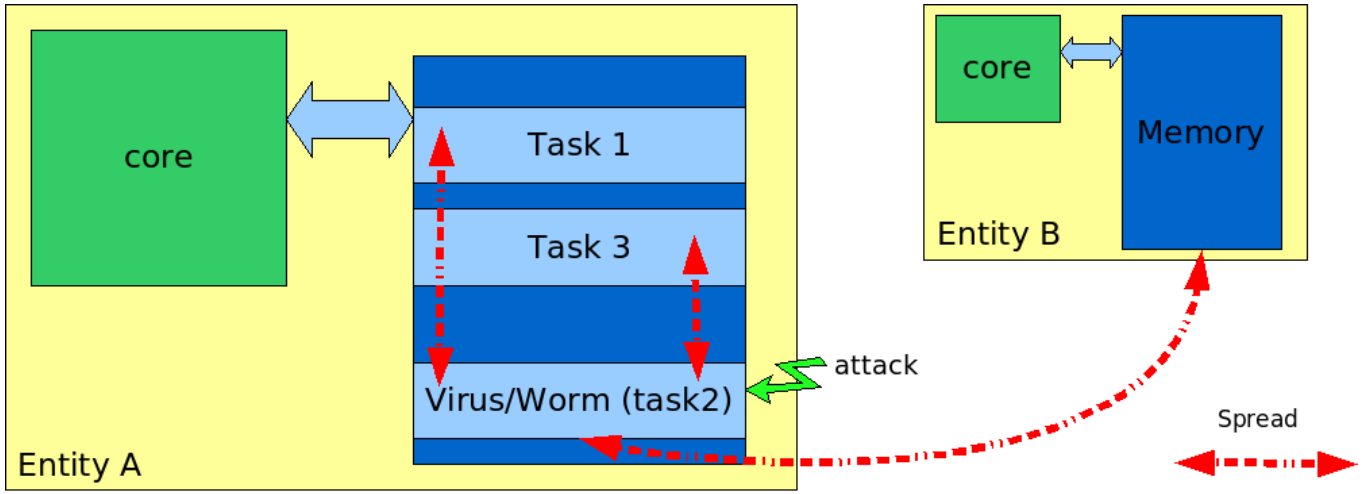
Some solutions

Attacks

- Memory modification (Integrity)
- Data extraction (Confidentiality)

Solutions

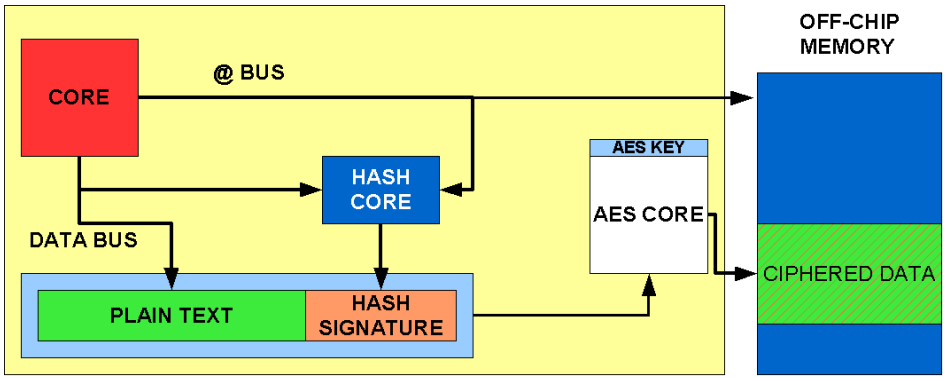
- Data hashing (MD5, SHA familly,...)
- Data ciphering (AES, RSA, ECC, ...)



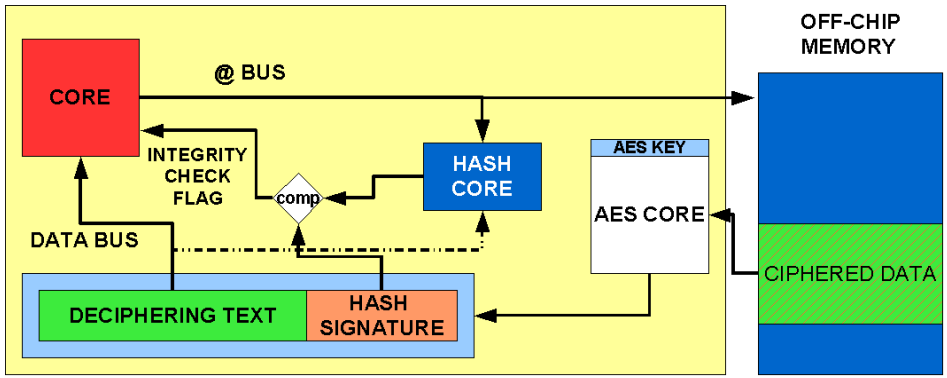
Existing solutions

Existing solutions

- XOM : memory ciphering (AES) and hashing (HMAC)
- PE-ICE : memory ciphering and hashing (only AES)
- AEGIS : memory ciphering (OTP) and hashing (SHA-1)



XOM write request

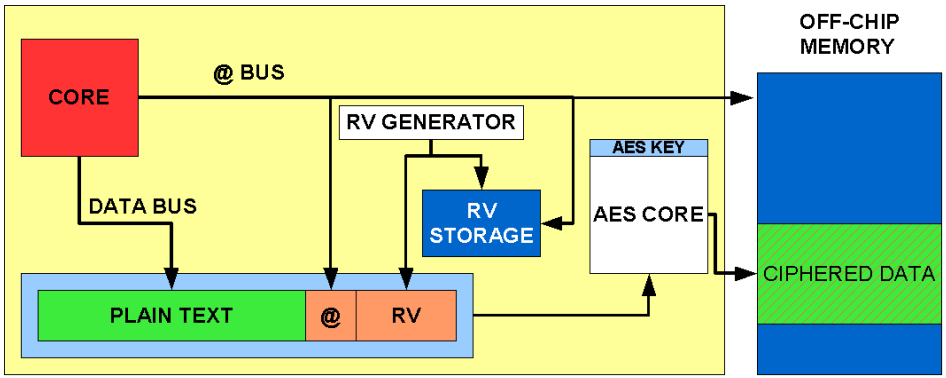


XOM read request

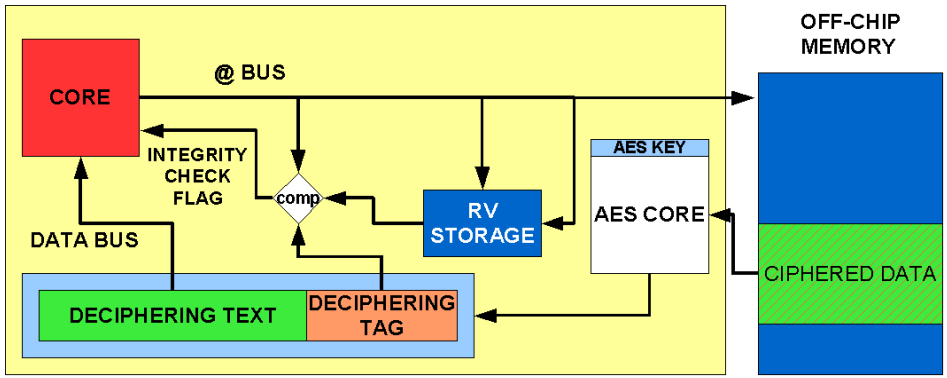
Existing solutions

Existing solutions

- XOM : memory ciphering (AES) and hashing (HMAC)
- PE-ICE : memory ciphering and hashing (only AES)
- AEGIS : memory ciphering (OTP) and hashing (SHA-1)



PE-ICE write request



PE-ICE read request

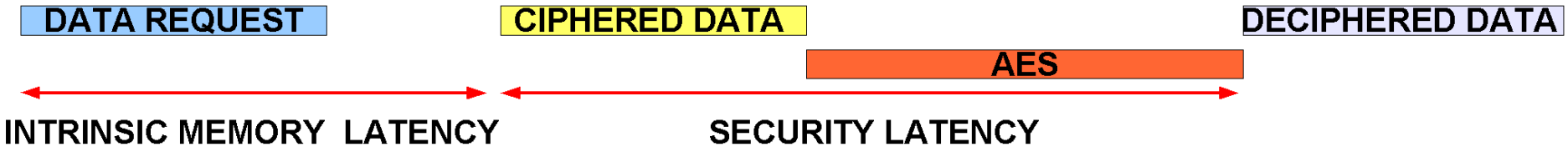
Existing solutions

Existing solutions

- XOM : memory ciphering (AES) and hashing (HMAC)
- PE-ICE : memory ciphering and hashing (only AES)
- AEGIS : memory ciphering (OTP) and hashing (SHA-1)

PROBLEM

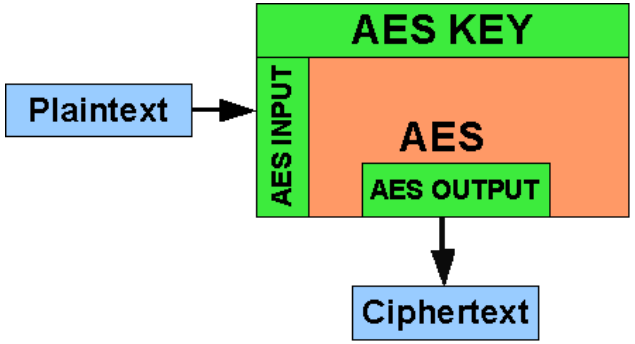
- Latency memory overhead adds by security solution



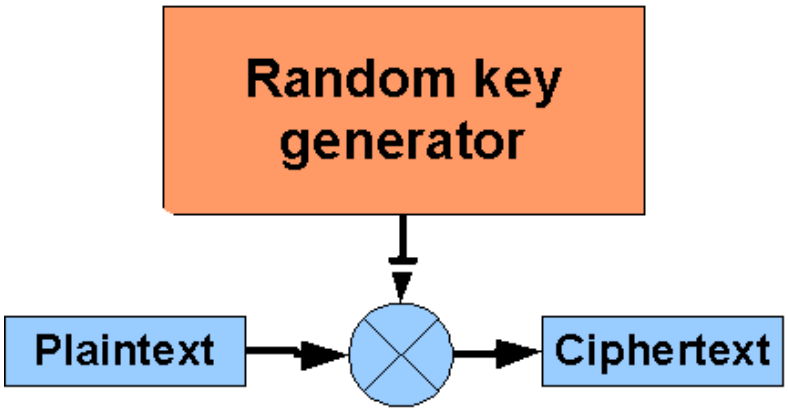
Outline

Extended One-Time-Pad encryption principals

Standard encryption

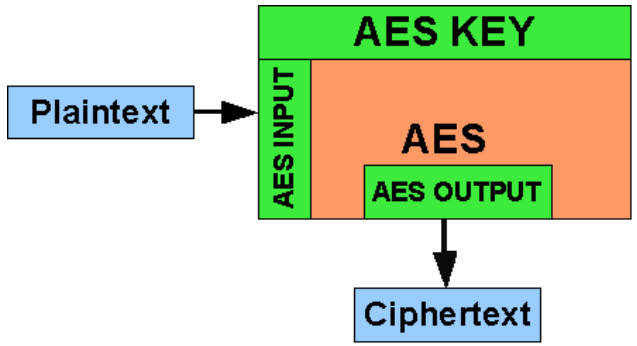


OTP encryption

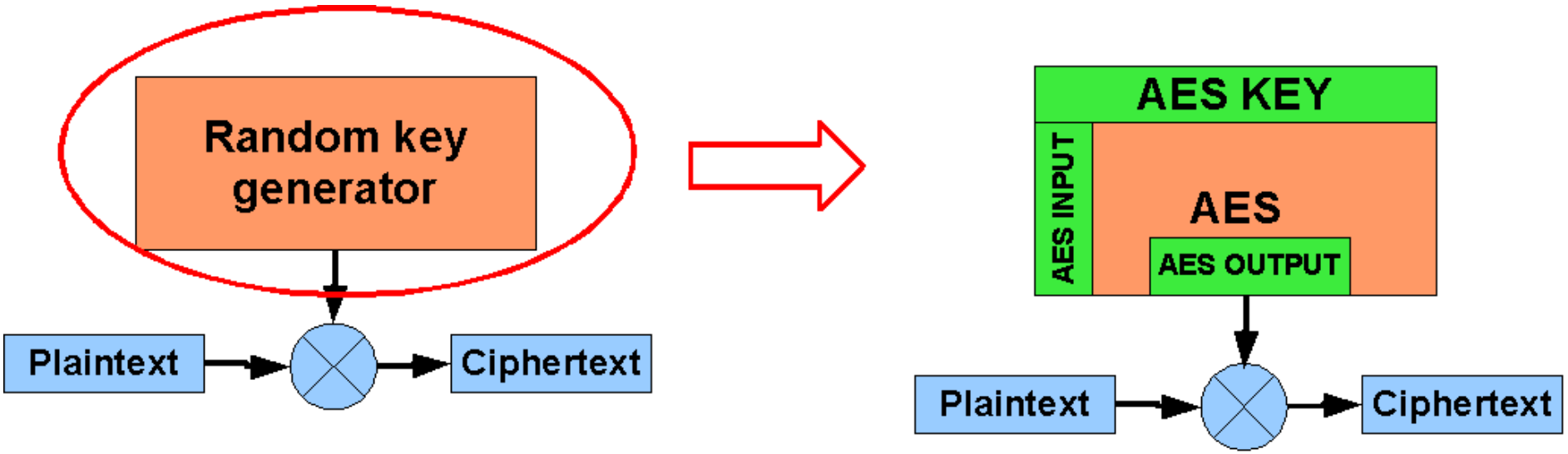


Extended One-Time-Pad encryption principals

Standard encryption

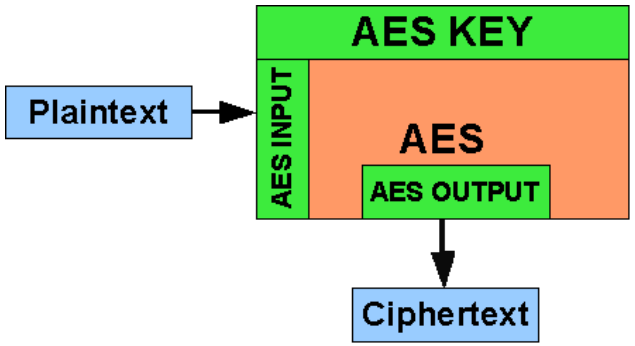


OTP encryption

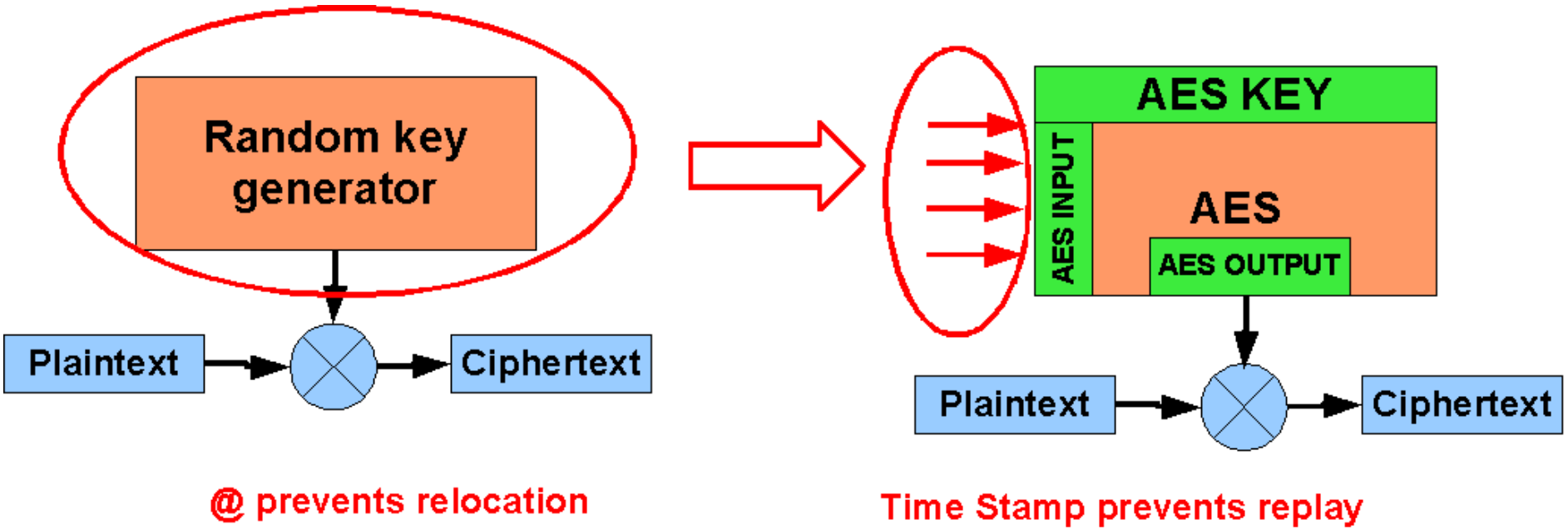


Extended One-Time-Pad encryption principals

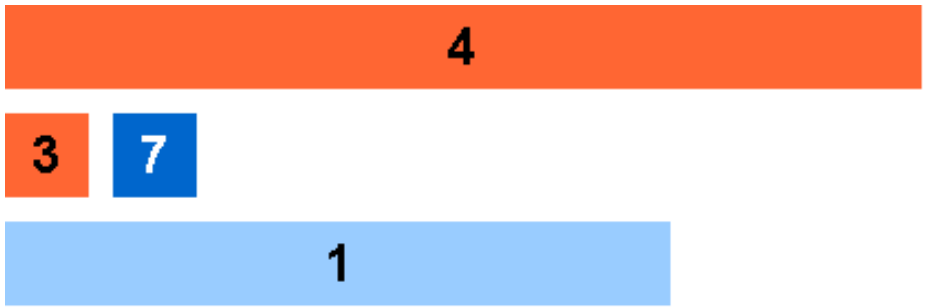
Standard encryption

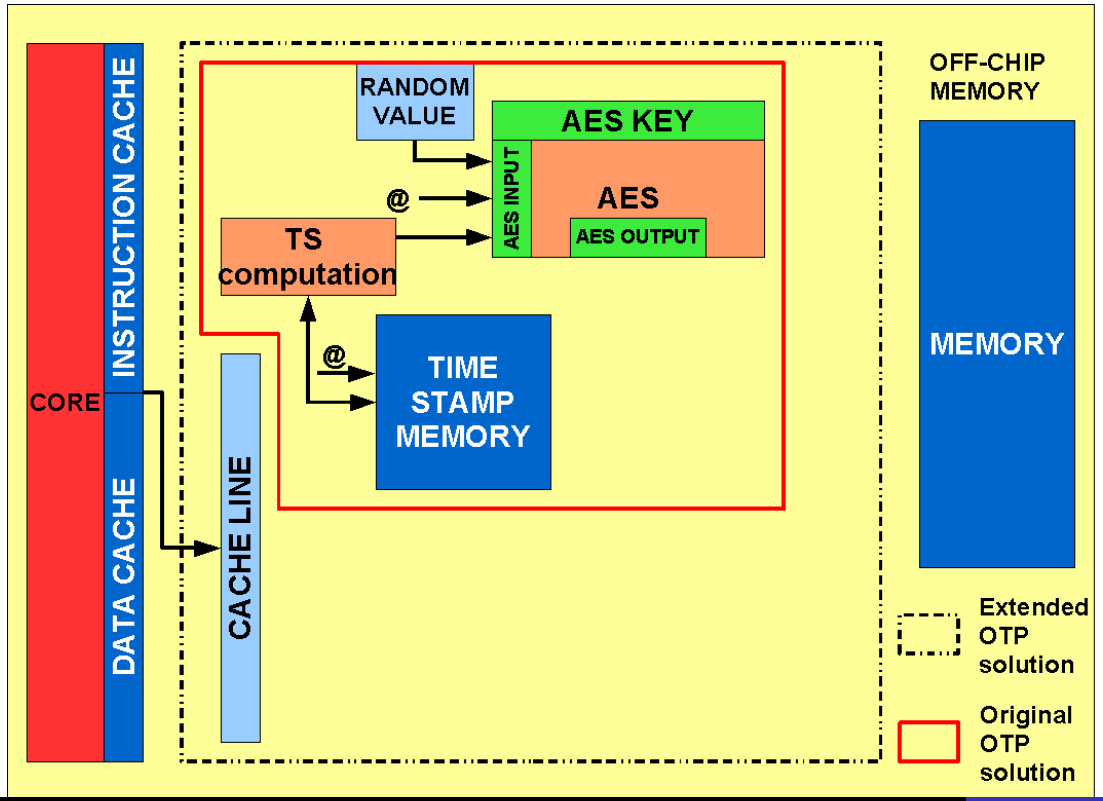


OTP encryption



OTP sequence





Write request :

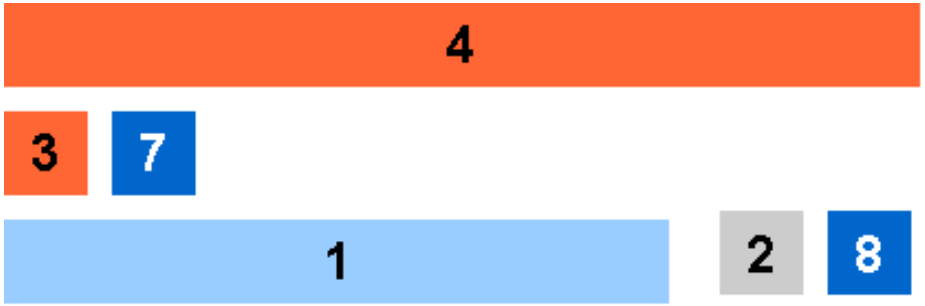
1 – *Get data* ⇐ *cache memory*

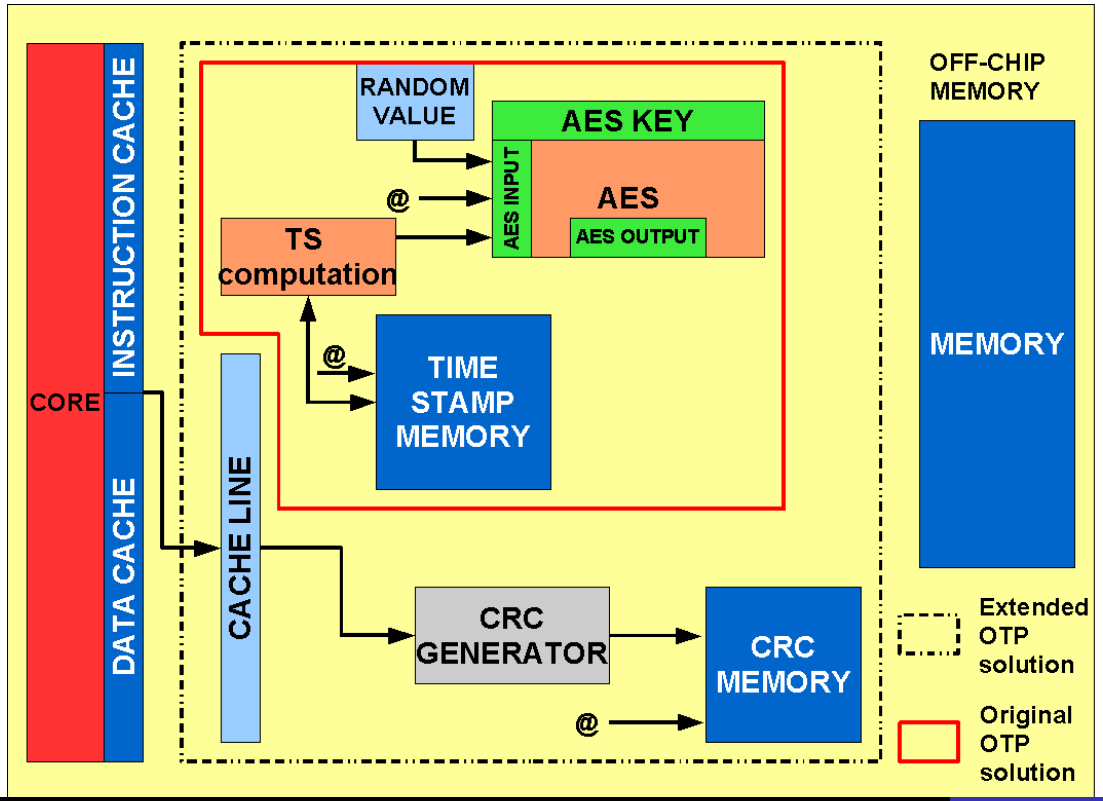
3 – $TS(@) = TS(@) + 1$

4 – $OTP = AES\{TS(@), @, RV\}$

7 – $TS(@) \Rightarrow$ *TS memory*

OTP sequence





Write request :

1 – *Get data* $\Leftarrow$ *cache memory*

2 – $CRC(@) = CRC\{plaintext\}$

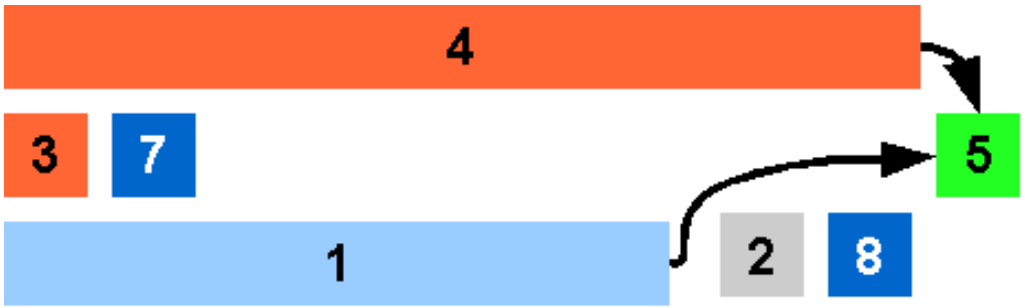
3 – $TS(@) = TS(@) + 1$

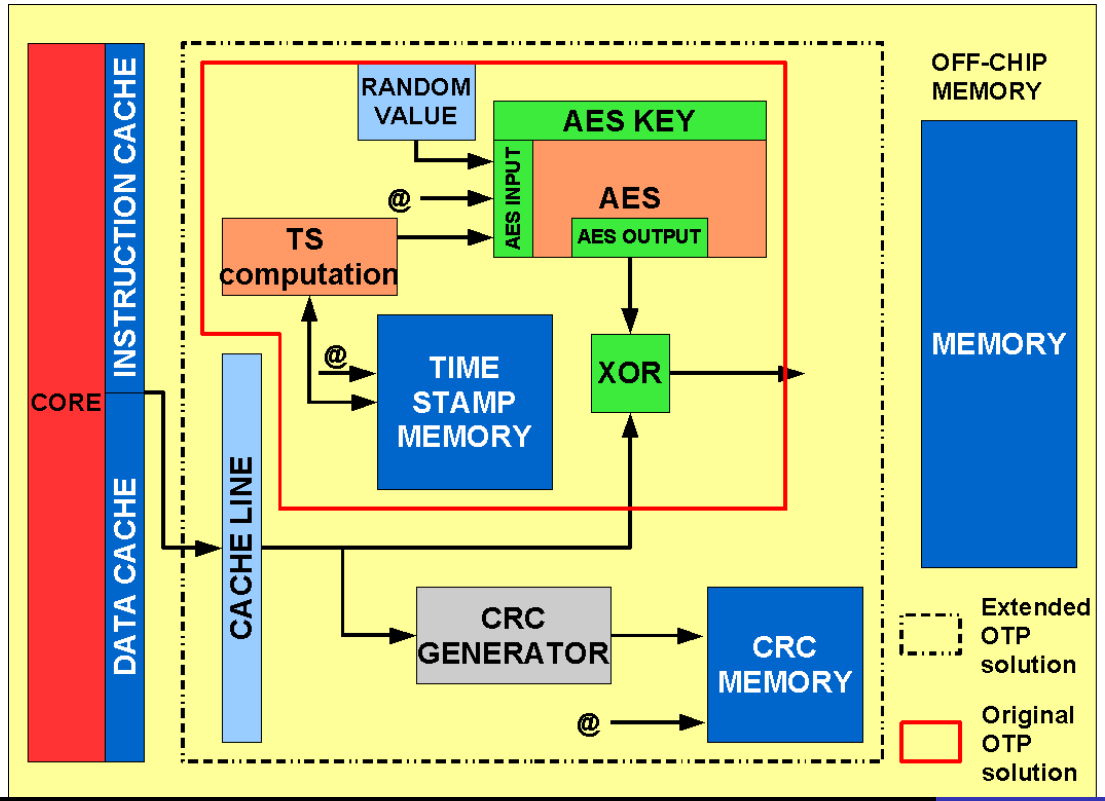
4 – $OTP = AES\{TS(@), @, RV\}$

7 – $TS(@) \Rightarrow$ *TS memory*

8 – $CRC(@) \Rightarrow$ *CRC memory*

OTP sequence





Write request :

1 – *Get data* $\Leftarrow$ *cache memory*

2 – $CRC(@) = CRC\{plaintext\}$

3 – $TS(@) = TS(@) + 1$

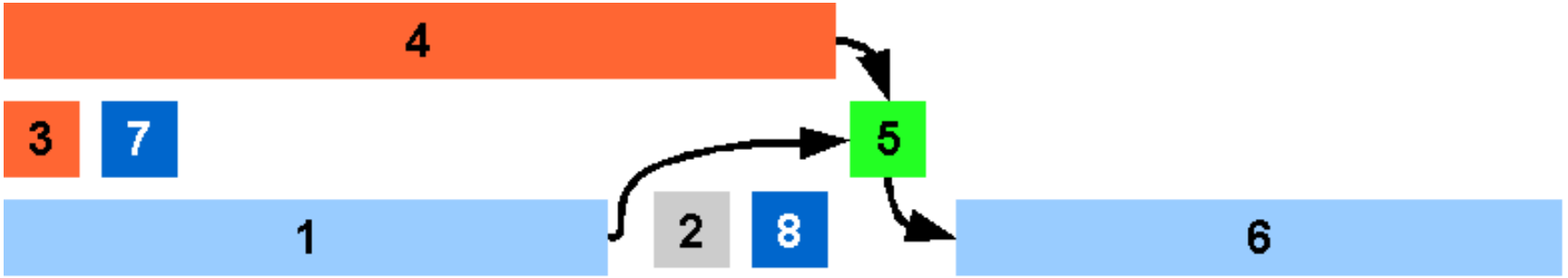
4 – $OTP = AES\{TS(@), @, RV\}$

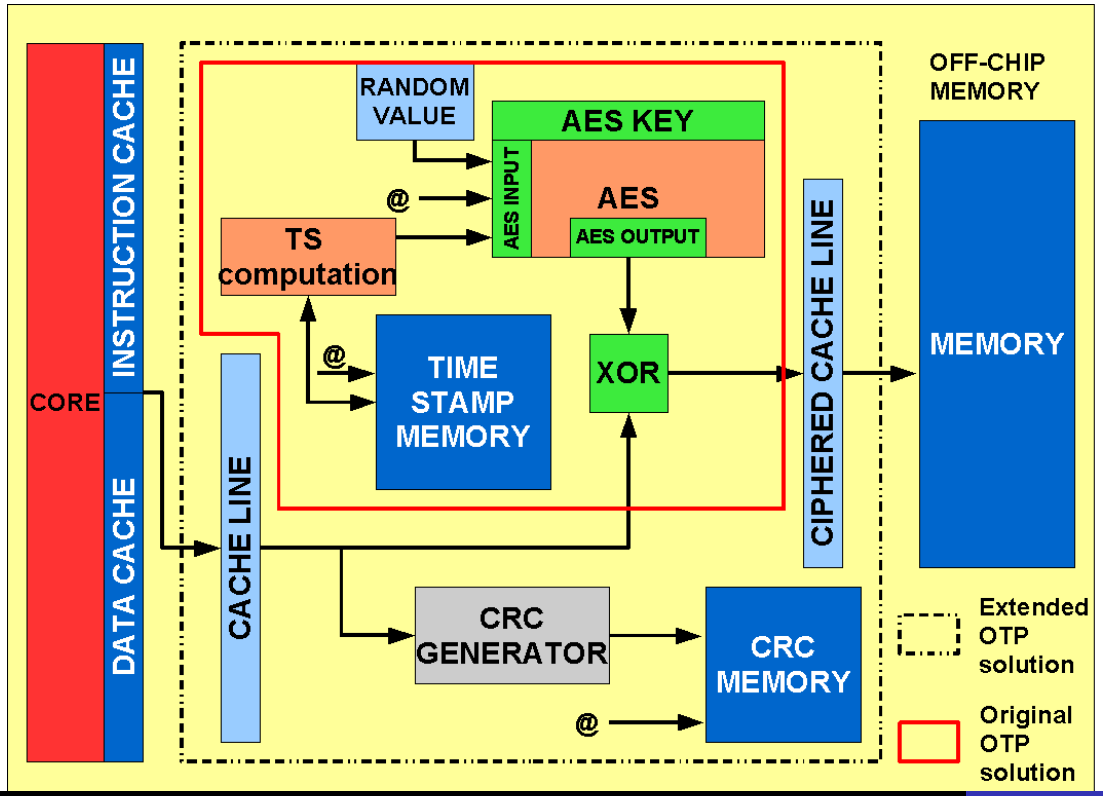
5 – *Ciphered data* $=$ *plaintext* $\oplus$ *OTP*

7 – $TS(@) \Rightarrow$ *TS memory*

8 – $CRC(@) \Rightarrow$ *CRC memory*

OTP sequence





Write request :

1 – *Get data* $\Leftarrow$ *cache memory*

2 – $CRC(@) = CRC\{plaintext\}$

3 – $TS(@) = TS(@) + 1$

4 – $OTP = AES\{TS(@), @, RV\}$

5 – *Ciphered data* $= plaintext \oplus OTP$

6 – *Ciphered data* $\Rightarrow$ *memory*

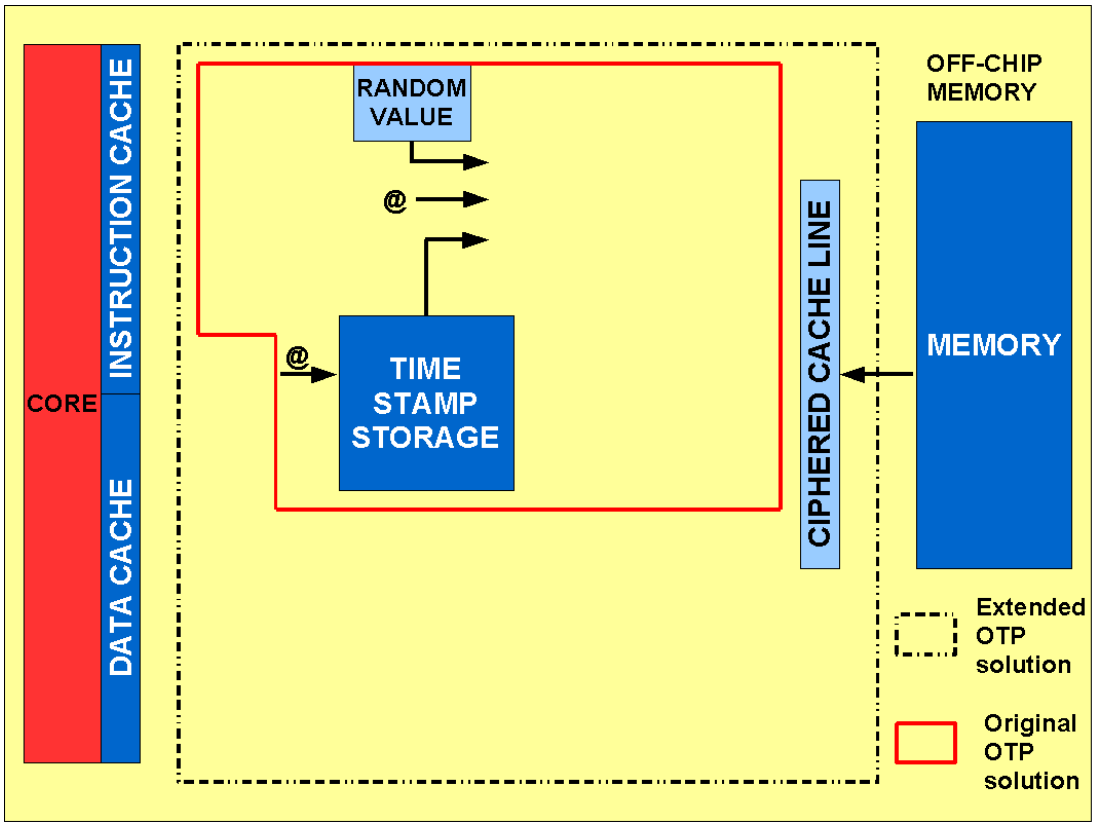
7 – $TS(@) \Rightarrow$ *TS memory*

8 – $CRC(@) \Rightarrow$ *CRC memory*

OTP sequence

1

4

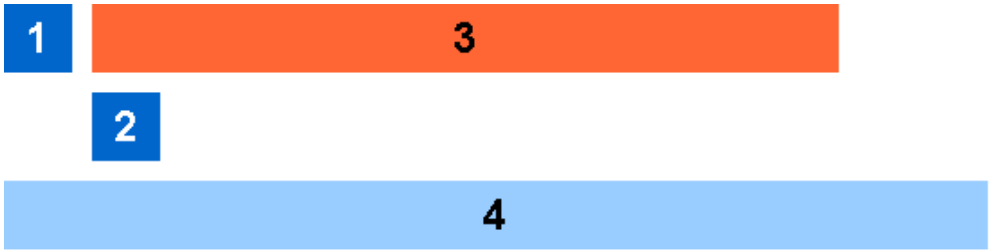


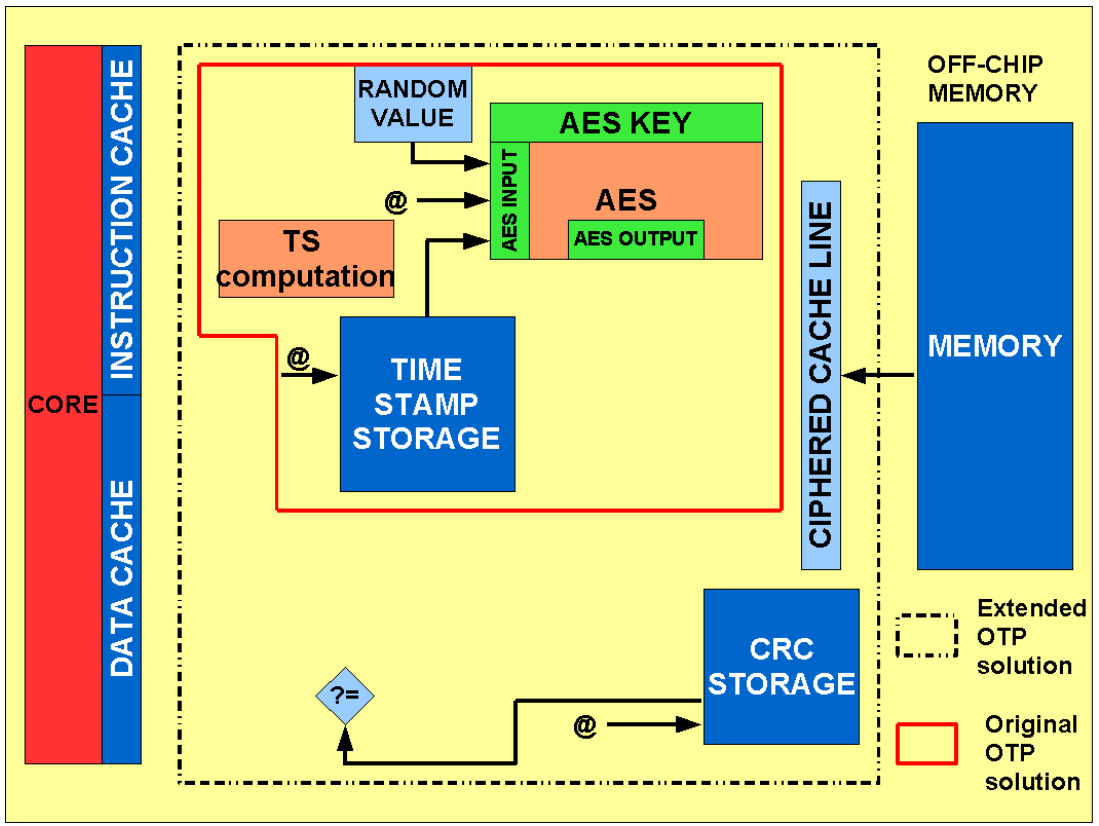
Read request :

1 – *Get TS (@) ⇐ TS memory*

4 – *Get ciphered data ⇐ memory*

OTP sequence





Read request :

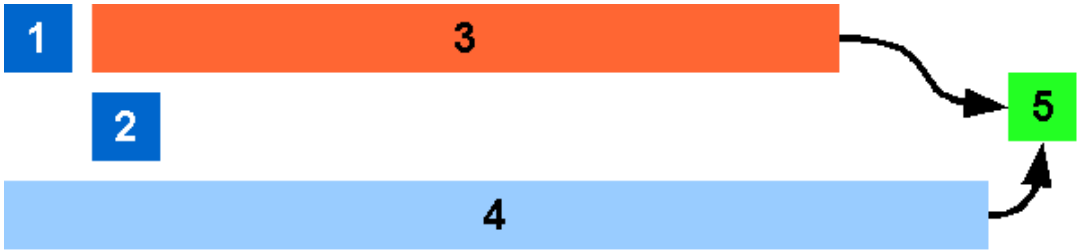
1 – *Get TS (@) ⇐ TS memory*

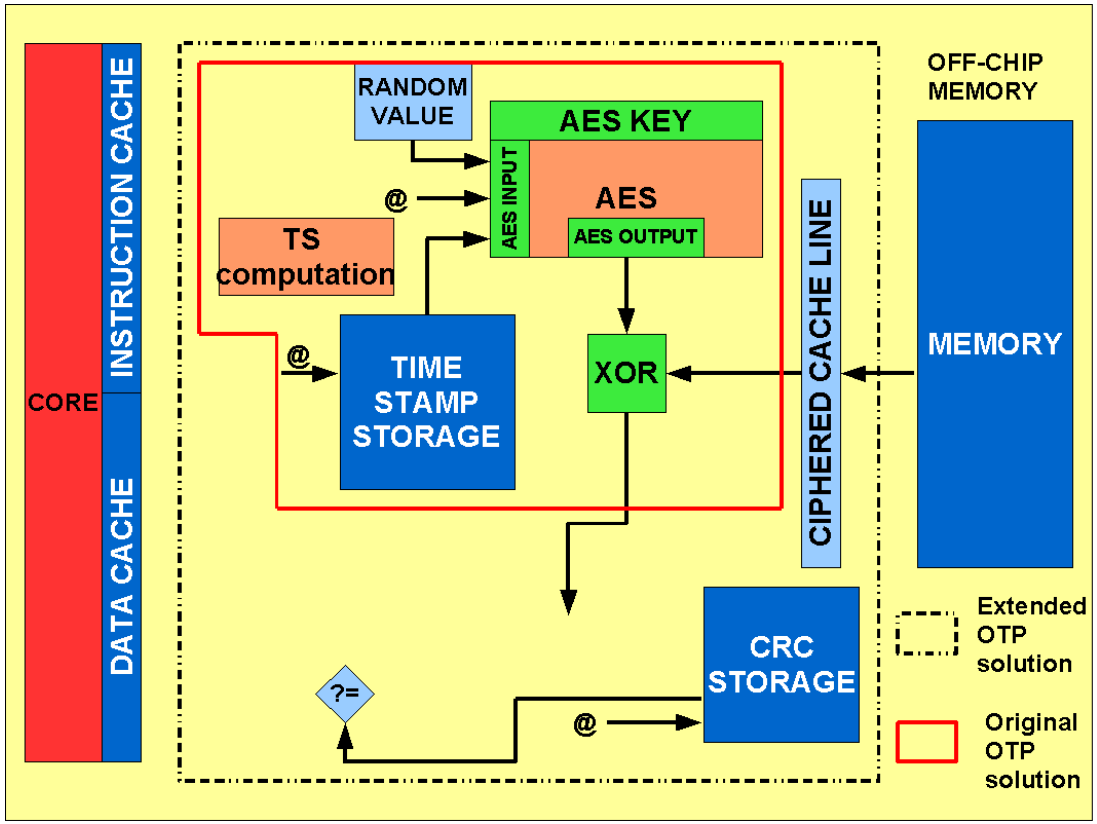
2 – *Get CRC (@) ⇐ CRC memory*

3 – $OTP = AES\{TS(@), @, RV\}$

4 – *Get ciphered data ⇐ memory*

OTP sequence





Read request :

1 – *Get TS* (@) ⇐ *TS memory*

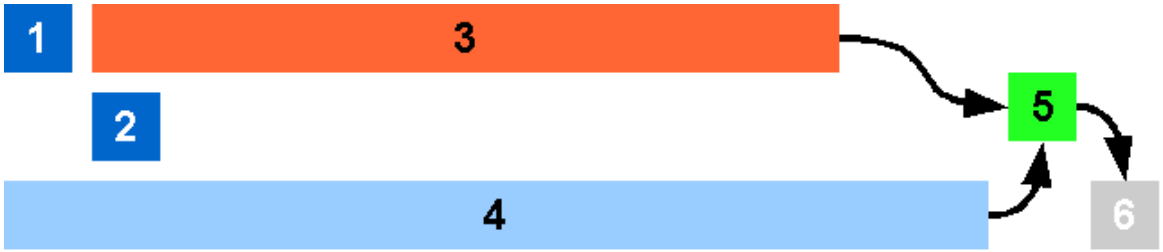
2 – *Get CRC* (@) ⇐ *CRC memory*

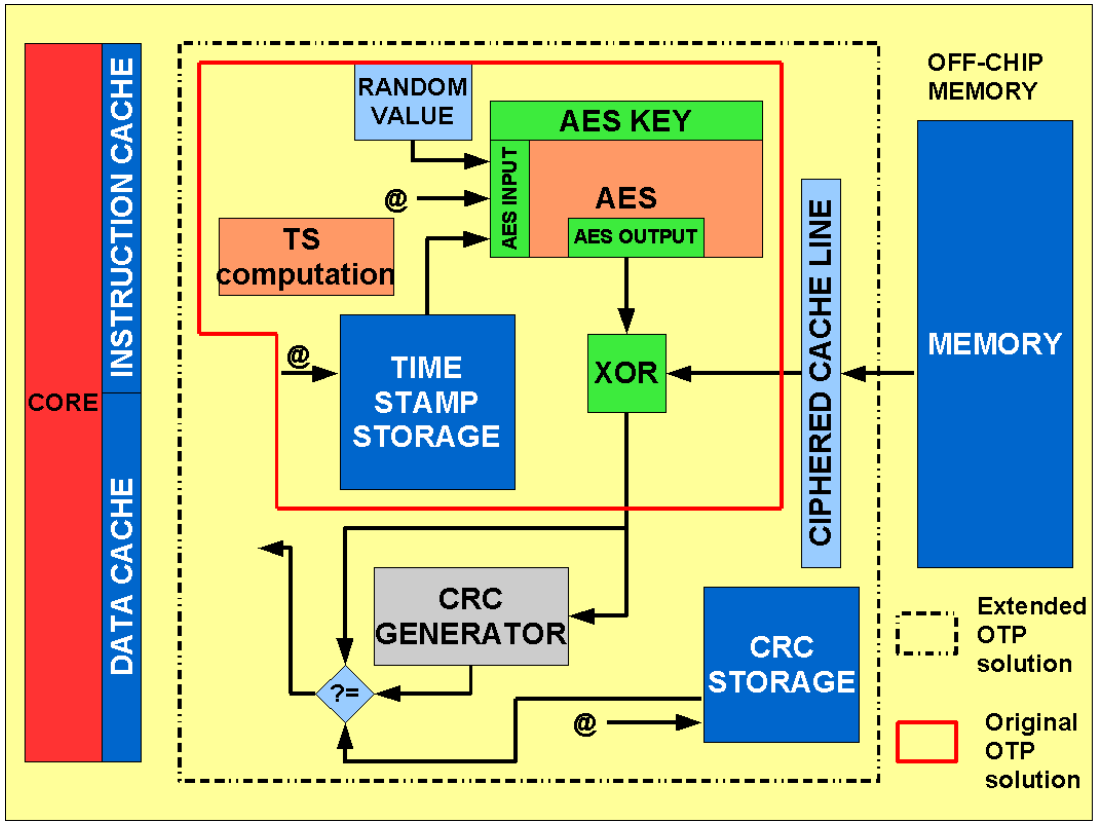
3 – $OTP = AES\{TS(@), @, RV\}$

4 – *Get ciphered data* ⇐ *memory*

5 – *Plaintext* = *Ciphered data* ⊕ *OTP*

OTP sequence





Read request :

1 – *Get TS* (@) ⇐ *TS memory*

2 – *Get CRC* (@) ⇐ *CRC memory*

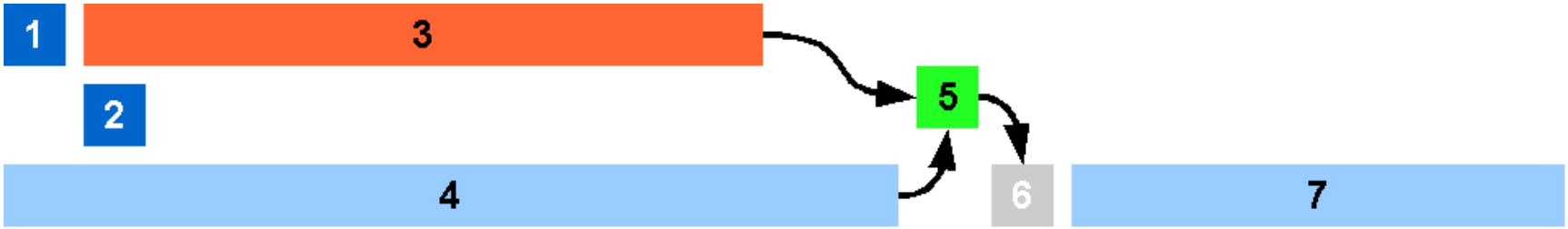
3 – $OTP = AES\{TS(@), @, RV\}$

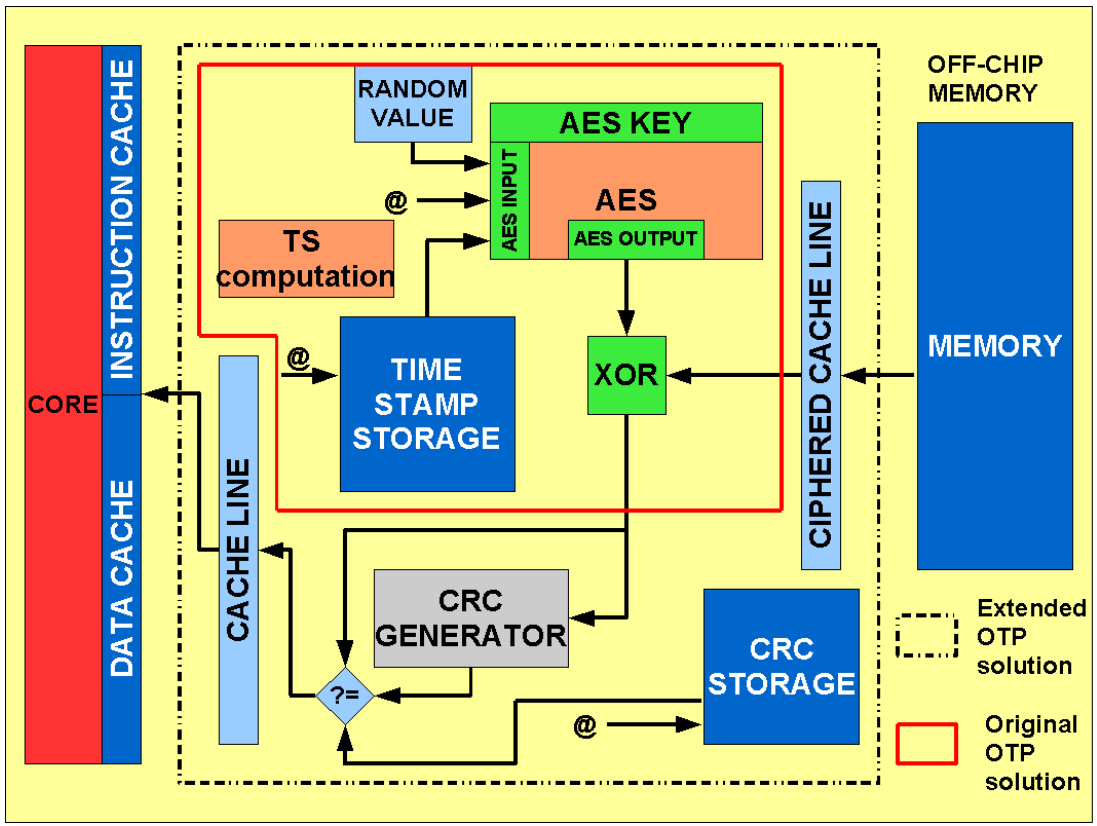
4 – *Get ciphered data* ⇐ *memory*

5 – *Plaintext* = *Ciphered data* ⊕ *OTP*

6 – $CRC(@) \equiv CRC\{plaintext\}$

OTP sequence





Read request :

1 – *Get TS* (@) ⇐ *TS memory*

2 – *Get CRC* (@) ⇐ *CRC memory*

3 – $OTP = AES\{TS(@), @, RV\}$

4 – *Get ciphered data* ⇐ *memory*

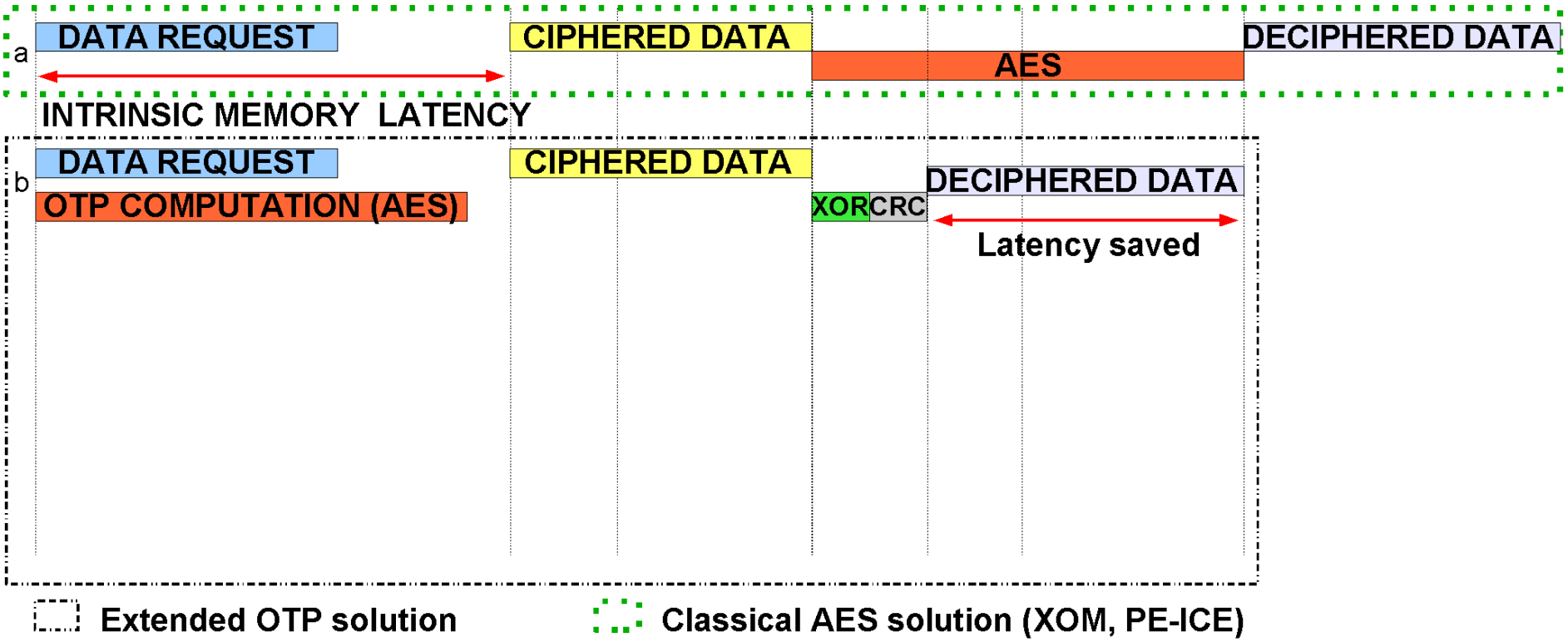
5 – *Plaintext* = *Ciphered data* ⊕ *OTP*

6 – *CRC* (@) ≡ *CRC* {*plaintext*}

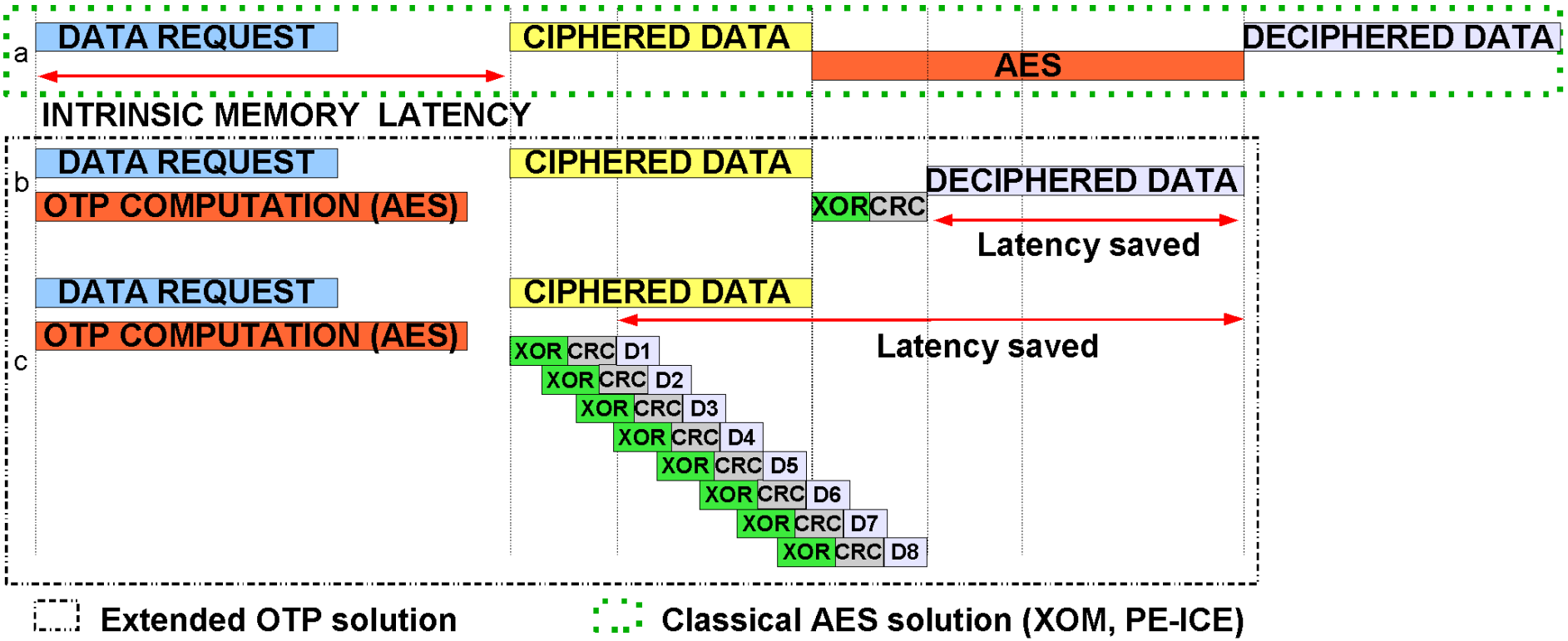
7 – *Plaintext* ⇒ *cache memory*

Outline

Latency with the extended OTP



Latency with the extended OTP



Outline

Global architecture features

Architecture features

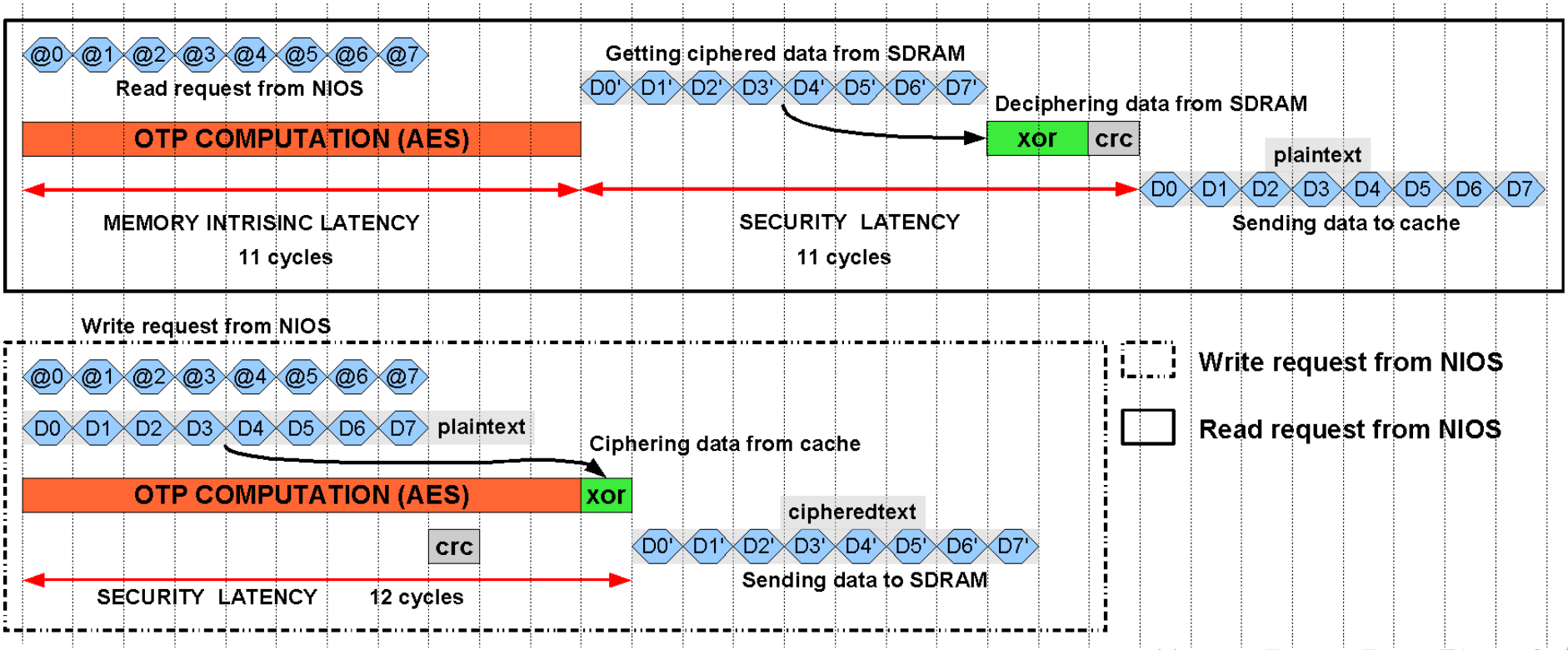
- ALTERA NIOS 2 processor
  - NIOS 2 core fast version
  - Instruction cache : 512 bytes with 256 bits per line
  - Data cache : 512 bytes with 256 bits per line
- SDRAM memory : 512 Kbytes (for code and rw data)
- On-chip-memory : 96 Kbytes (for TS and CRC)

OTP memory consumption

$$OTP_{STORAGE} = TS_{STORAGE} + CRC_{STORAGE}$$
$$TS_{STORAGE} = \left(\frac{RW\_DATA\_MEMORY\_SIZE}{CACHE\_LINE\_WIDTH}\right) * TS\ SIZE$$
$$CRC32_{STORAGE} = \left(\frac{TOTAL\_MEMORY\_SIZE}{CACHE\_LINE\_WIDTH}\right) * CRC\ SIZE$$

Cost of security with NIOS

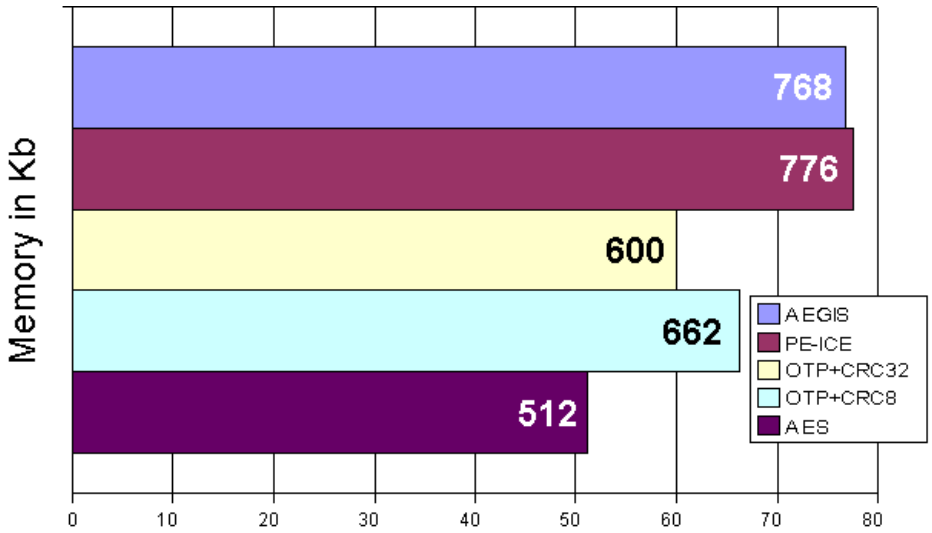
	Base NIOS	NIOS + OTP128 + CRC32		NIOS + OTP128 + CRC8	
			overhead		overhead
Logic (ALUTs)	2198	6193	x2.81	6095	x2.77
Memory (KB)	512	600	+18.75%	662	+31.25%
Read latency (cycles)	0	11(8+3)	+11	3(0+3)	+3
Write latency (cycles)	0	12(8+4)	+12	12(8+4)	+12

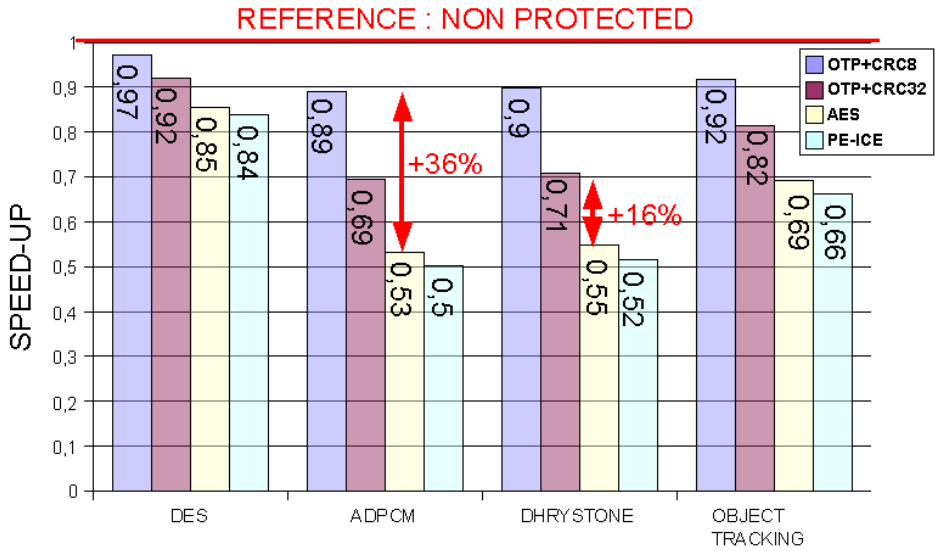


Outline

Comparison with previous solutions

	base AES (no integrity)	our solution OTP + CRC32		our solution OTP + CRC8		PE-ICE AES		AEGIS OTP + hash trees	
			overhead		overhead		overhead		overhead
Memory (KB)	512	600	+18.75%	662	+31.25%	776	+50.7%	768	+50%
Rd latency (cycles)	22(14+8)	11(8+3)	-11	3(0+3)	-19	25(17+8)	+3	≈SHA-1	+4502/69
Wr latency (cycles)	22(14+8)	12(8+4)	-10	12(8+4)	-10	26(18+8)	+4	-	-





Conclusions on the extended OTP

OTP features

- Efficient software execution
- Minimize the memory overhead
- Confidentiality protection
- Integrity protection
- But need for extra logic

Trade-off memory overhead/software execution

- software execution $++ \Rightarrow$ memory $++$
- memory $-- \Rightarrow$ software execution $--$

Perspectives

Increasing security level

- Providing security against hardware attacks (side-channel for example)
- Extending the threat model (reducing the trusted zone)

Security issues

- Provide a deep evaluation of the security level of the architecture (depending on the CRC size, the cache line size)

Perspectives

Architecture exploration

- Exploration for different architecture features (cache size, cache line size, CRC size)
- Reduce the on-chip memory footprint
- Store securely TS and CRC in off-chip memory

Future orientation

- Evaluation of the power consumption cost due to security
- Memory protection management with a RTOS
- Use the reconfigurable features of the FPGA for security and power management purposes

Conclusion

Alternative

- Alternative to standard solutions
- Very high performances
- Adapted to embedded systems constraints

Future orientation

- Many opportunities for OTP solution
- Security issues
- Architecture issues