An Easily Testable and Reconfigurable Pipeline for Symmetric Block Ciphers

Myeong-Hyeon Lee and Yoon-Hwa Choi

Computer Engineering Department

Hongik University, Seoul, Korea

Outline

- Motivation
- Testable and reconfigurable pipeline for block ciphers
- Scheduling for error detection and reconfiguration
- Performance
- Conclusions

Motivation

- Several CED techniques for symmetric block ciphers have been proposed to detect errors during normal operation.
  - Time/space redundancies
  - Coding techniques
  - Exploitation of the inverse relationship between encryption and decryption
  - Parity-based error detection
- It would be desirable for the system to function correctly even with some faults (Fault tolerance).

Main Idea

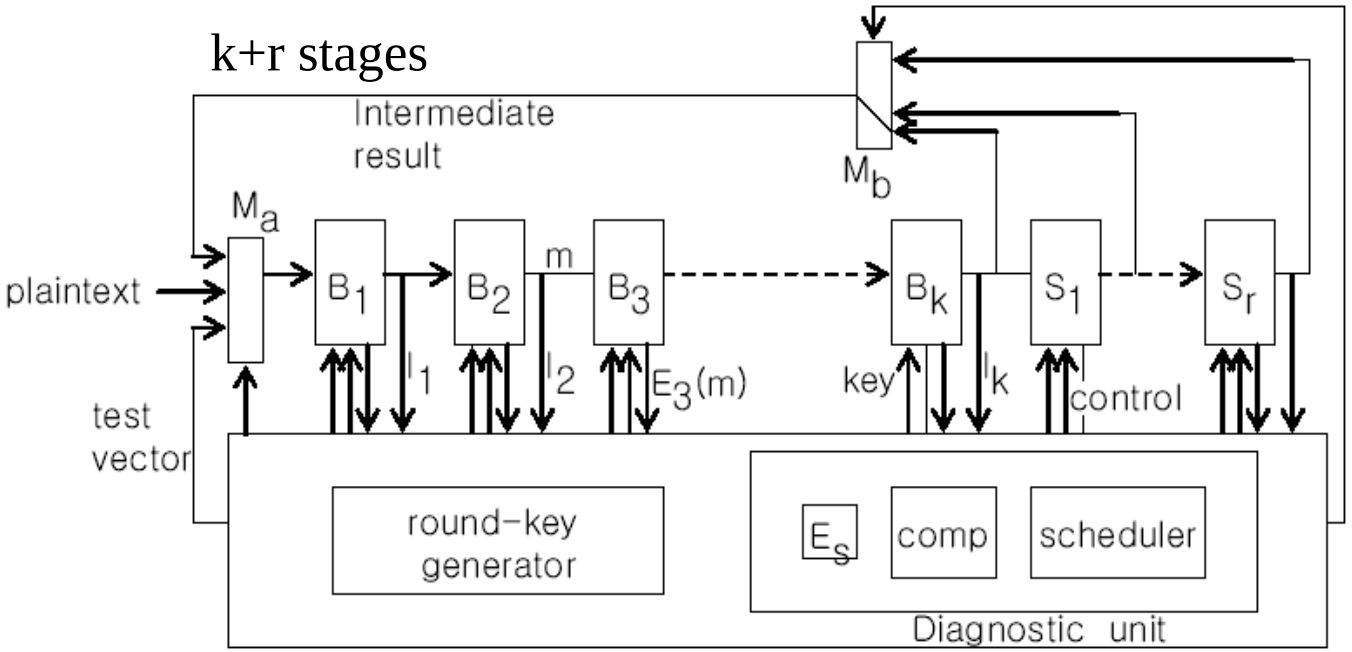
- Exploit parallelism in encryption/decryption by pipelining to compensate for the time overhead inherent in a time redundancy based CED.
- Implement CED and reconfiguration in a simple and unified structure to minimize the hardware overhead.
- Utilize bypass links to detect faults and reconfigure during normal operation.

Fault Model

- A single faulty encryption block in the pipeline
- Bypass links are assumed to be fault-free or pre-tested.
- Comparators are assumed to be fault-free.

Testable and Reconfigurable Pipeline

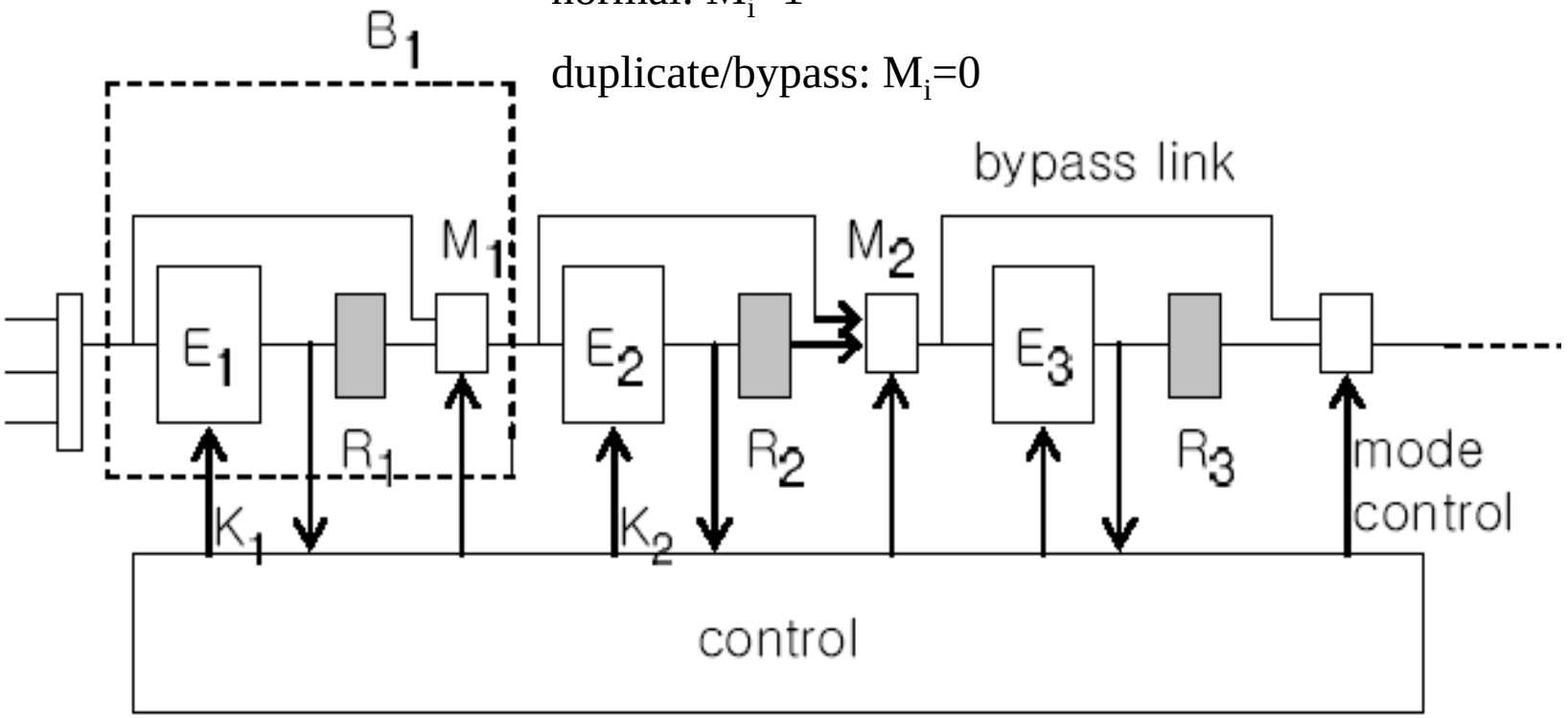
- E_1, E_2, …, E_n: n encryption blocks in the n-stage virtual pipeline, where n rounds are realized using k pipeline stages
- B_1,B_2,…,B_k: k pipeline stages
- S_1,..,S_r: r spare stages



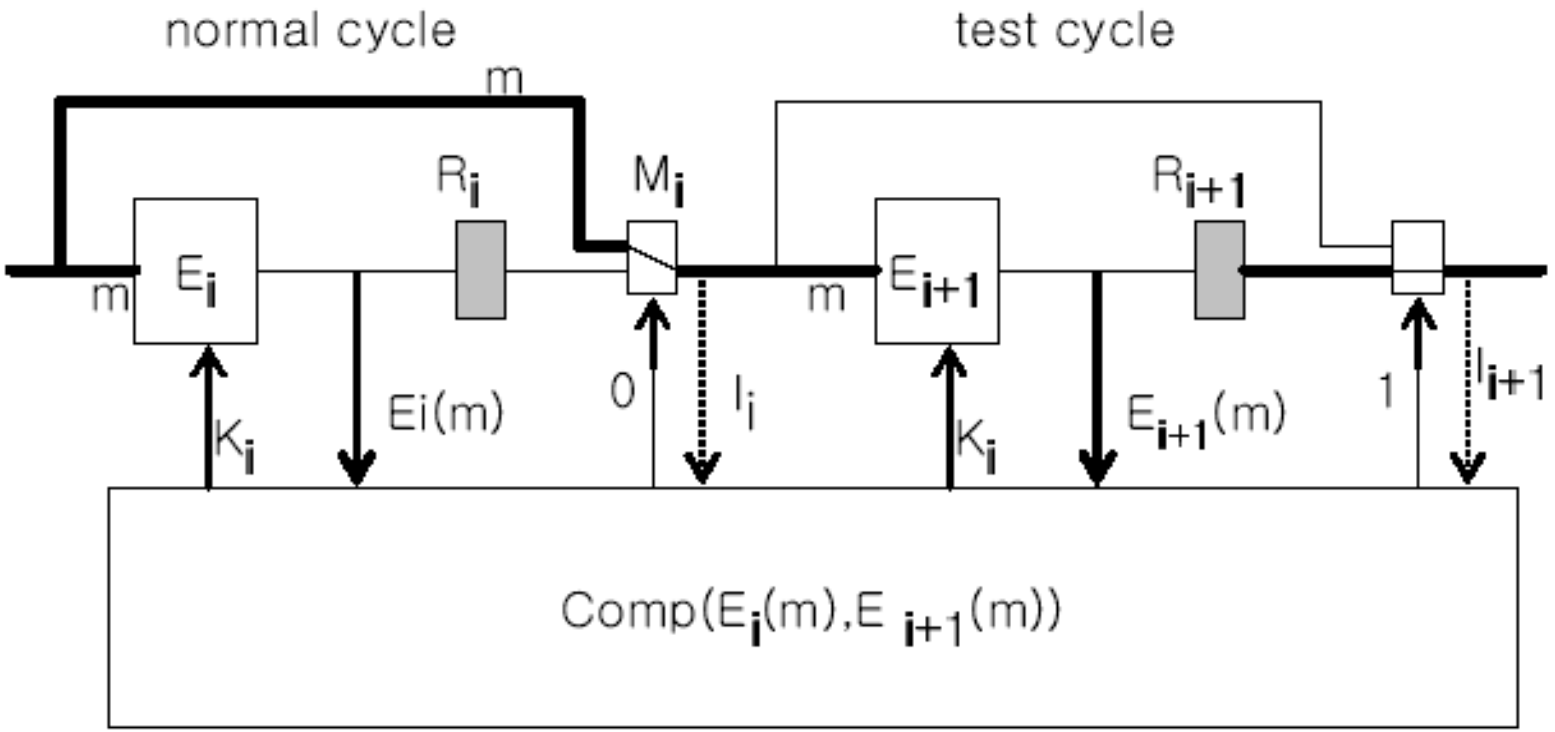
Internal Structure for duplication/bypassing

normal: $M_i=1$

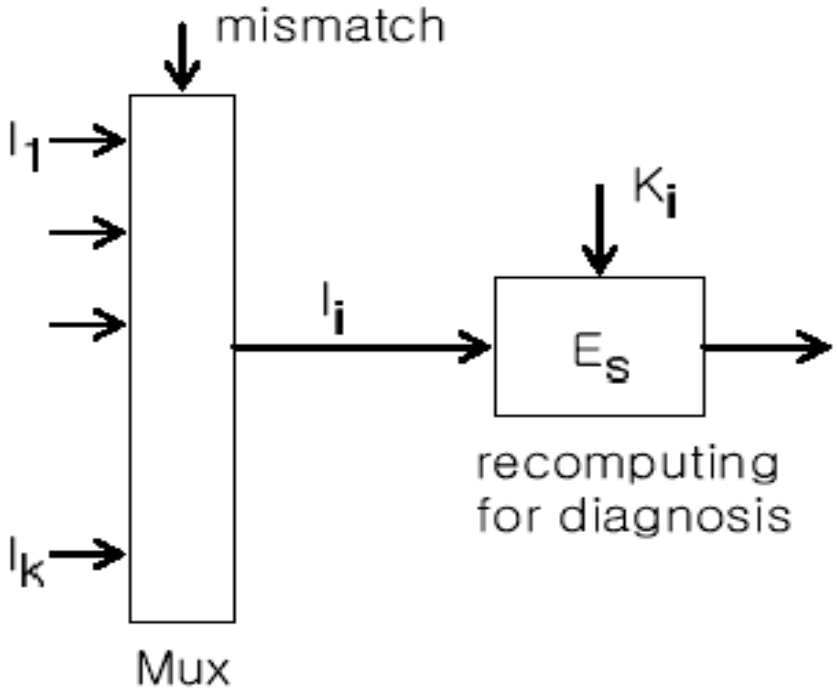
duplicate/bypass: $M_i=0$



Error Detection



Fault Location with Recomputing and Comparison



Scheduling for Error Detection and Reconfiguration

- Three different modes: normal, test, bypass

Table 1. Three different modes of each stage B_i

mode	C_{i-1}	C_i	K_{in}	explanation
normal	1	×	K_i	
test	0	×	K_{i-1}	key of the previous stage
bypass	×	0	×	isolated all the time

Four Strategies

- Complete checking: Every operation is duplicated and compared.
- Periodic checking: A test cycle is inserted periodically.
- Selective checking: Not necessarily periodically
- Checking with test vectors: In the case of **idle**

Controlling Multiplexers

- CED involves one idle cycle for duplicate computation along the pipeline

time	M_1	M_2	M_3	M_4	$-$	$-$	M_k	M_{s1}	$-$
t=1	0	1	1	1	1	1	1	1	1
t=2	1	0	1	1	1	1	1	1	1
t=3	1	1	0	1	1	1	1	1	1
t=4	1	1	1	0	1	1	1	1	1
t=$-$	1	1	1	1	0	1	1	1	1
t=$-$	1	1	1	1	1	0	1	1	1
t=k	1	1	1	1	1	1	0	1	1

Test Scheduling for Complete/Selective Checking

Table 3. Test scheduling for complete checking

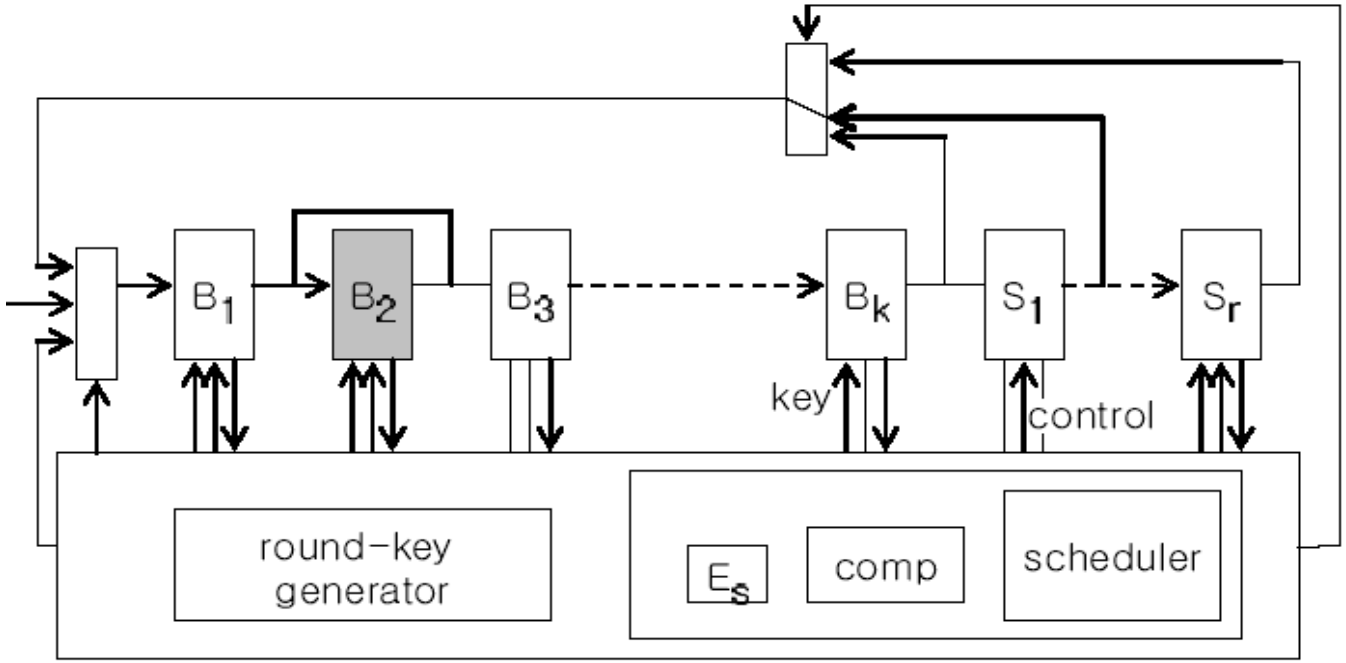
stage	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	–
B_1	1	-	2	-	3	-	4	-		
B_2	$\tilde{1}$	1	$\tilde{2}$	2	$\tilde{3}$	3	$\tilde{4}$	4		
B_3		$\tilde{1}$	1	$\tilde{2}$	2	$\tilde{3}$	3	$\tilde{4}$	4	
B_4			$\tilde{1}$	1	$\tilde{2}$	2	$\tilde{3}$	3	$\tilde{4}$	4
S_1				$\tilde{1}$		$\tilde{2}$		$\tilde{3}$		$\tilde{4}$

Table 4. Test scheduling for selective checking

stage	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	–
B_1	1	2	3	4	-	5	6	7		
B_2		1	2	3	4	$\tilde{5}$	5	6	7	
B_3			1	2	3	4	$\tilde{5}$	5	6	7
B_4				1	2	3	4	$\tilde{5}$	5	6
S_1									$\tilde{5}$	

Reconfigured Pipeline

- One of the spare stages has to participate in forming a new pipeline with k stages



Performance

- We insert a test cycle for every q normal cycles on average
- If q = 1(complete checking) and k = 2(two-stage pipeline), the throughput is 1/n (the same as the non-pipelined design)

	non-pipelined	time-redundancy	proposed
Error detection	no	transient	transient/permanent
Fault tolerance	no	no	yes
No. of encryption blocks	1	1	$k+1$
Error detection time	n/a	1 cycle delayed	immediately
Time overhead	n/a	100%	$\frac{1}{q} \times 100\%$
Throughput	$\frac{1}{n}$	$\frac{1}{2n}$	$\frac{k}{n}(\frac{q}{1+q})$

Throughput

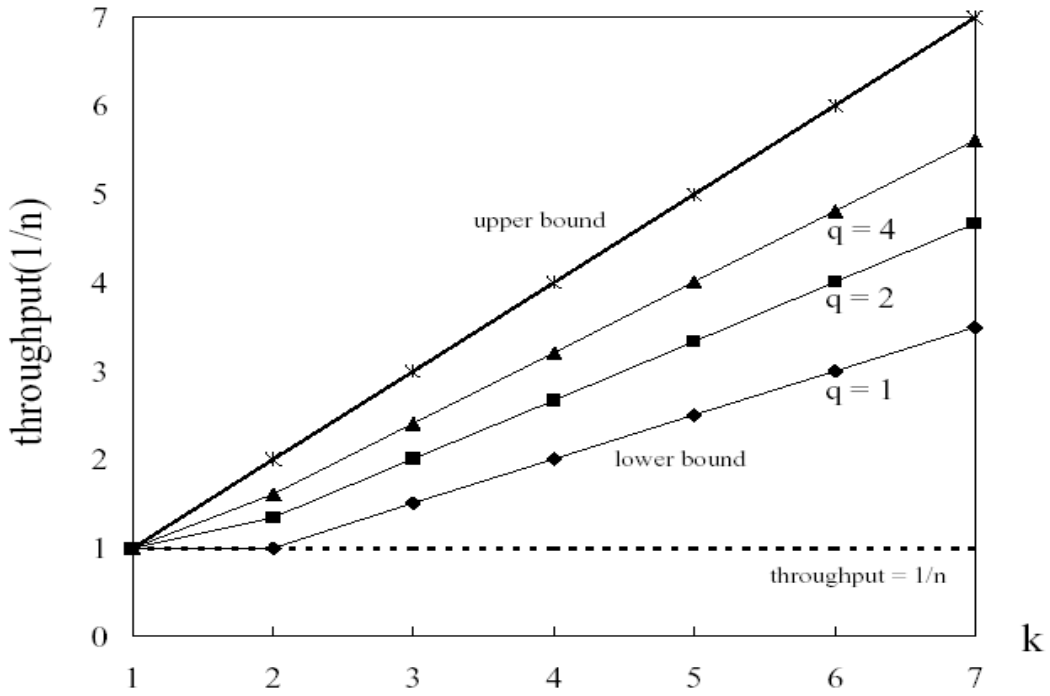


Fig. 6. Throughput for various values of k and q

Conclusions

- We have presented an easily testable and reconfigurable pipeline for block ciphers.
- Errors are detected by duplicate computations using bypass links.
- A faulty encryption block is isolated by activating the same bypass links for duplication.
- Time overhead for error detection has been controlled by dynamically inserting test cycles.
- The proposed technique can best be used for block ciphers, where throughput and availability are of the utmost importance.