



Reverse to Reveal: Reversibility as a Fault Defense

- Xhani Marvin Saß, Meng Yuan, Jean-Pierre Seifert

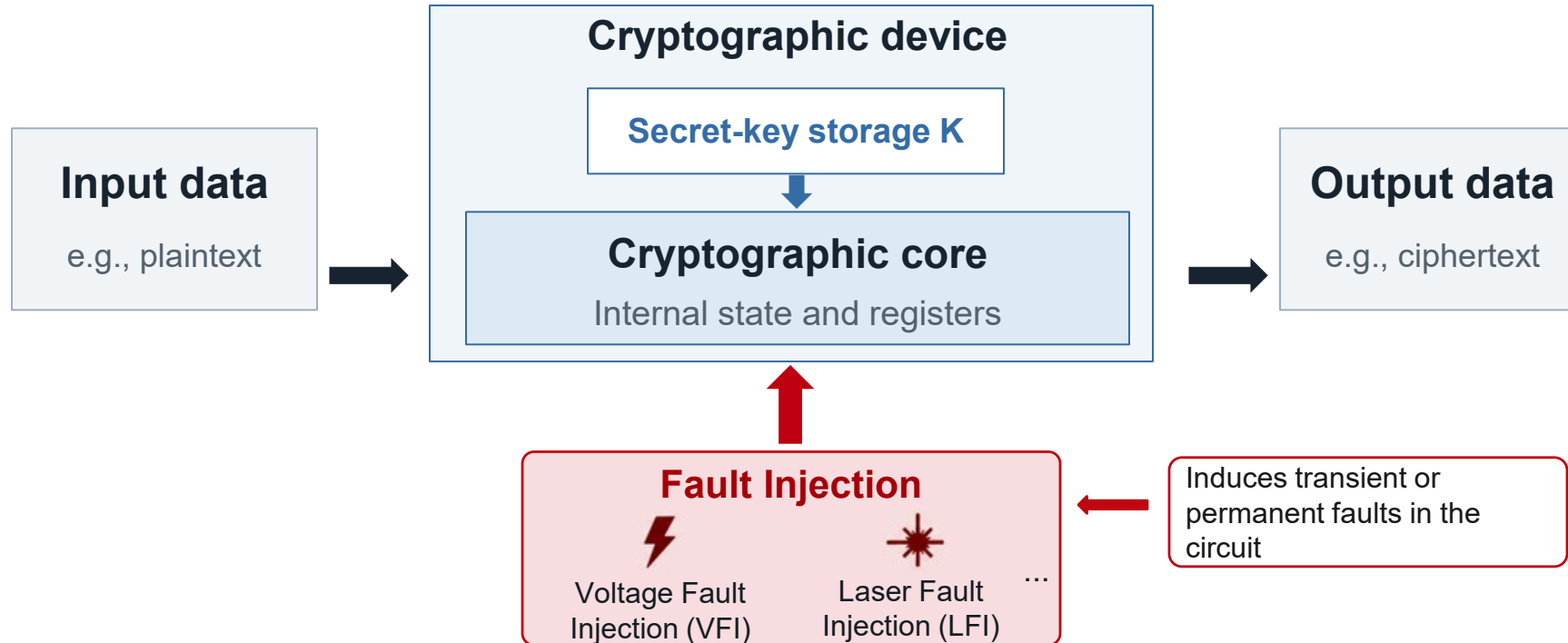
Technische Universität Berlin - Security in Telecommunications (SecT)

Fault Diagnosis and Tolerance in Cryptography (FDTC 2026)

Motivation

Fault injection threatens cryptographic hardware

An attacker perturbs a device and observes its external behavior.



Can we correct errors before they propagate to observable data?



Our Approach

Fault detection & on-the-fly error correction

Integrate fault protection directly into the **structure** of computations.

t: target
c: control

Reversible primitives

FOUNDATIONS

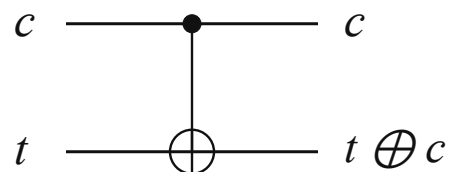
A gate is logically reversible if its output uniquely determines its input.

NOT



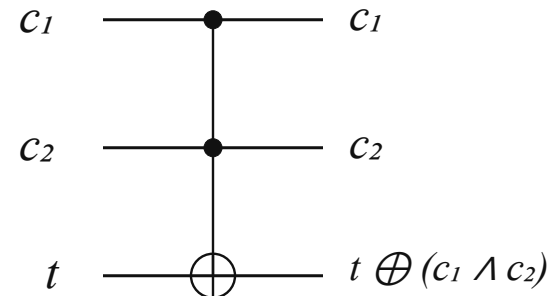
$$t \mapsto \bar{t}$$

CNOT (Feynman)



$$(c, t) \mapsto (c, t \oplus c)$$

CCNOT (Toffoli)



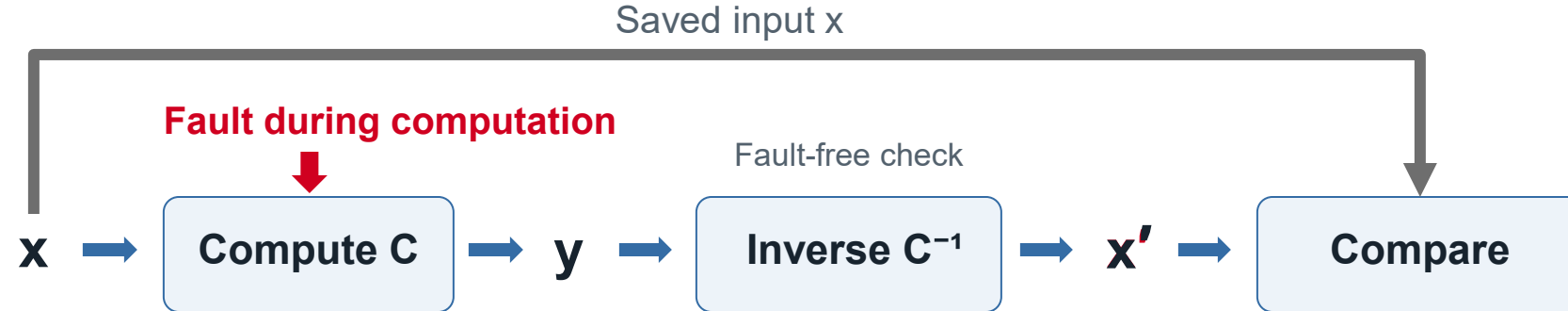
$$(c_1, c_2, t) \mapsto (c_1, c_2, t \oplus (c_1 \wedge c_2))$$

Every gate preserves enough information to reconstruct its inputs.

Reverse the gate order to obtain the inverse circuit.

Two reversibility-based defense schemes

Scheme 1: Bennett uncomputation



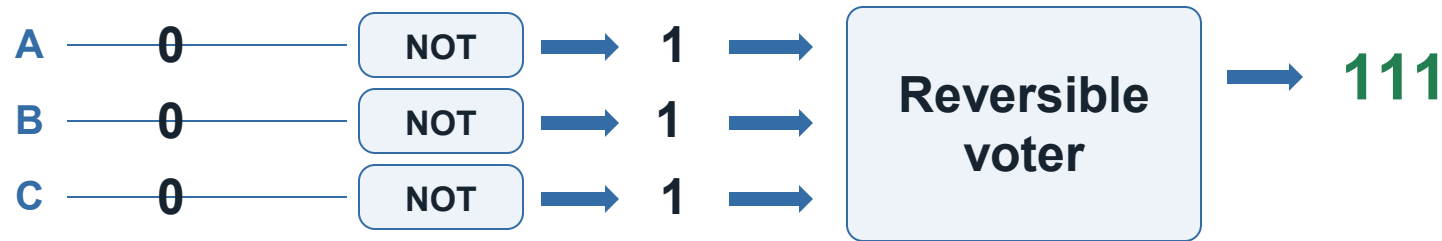
Faulty state $\tilde{y} \rightarrow \text{inverse} \rightarrow \text{reconstructed input } x'$

$x' = x \rightarrow \text{no fault detected}$

Scheme 2: Secure gates: SNOT example

Three NOT gates, followed by a reversible voter.

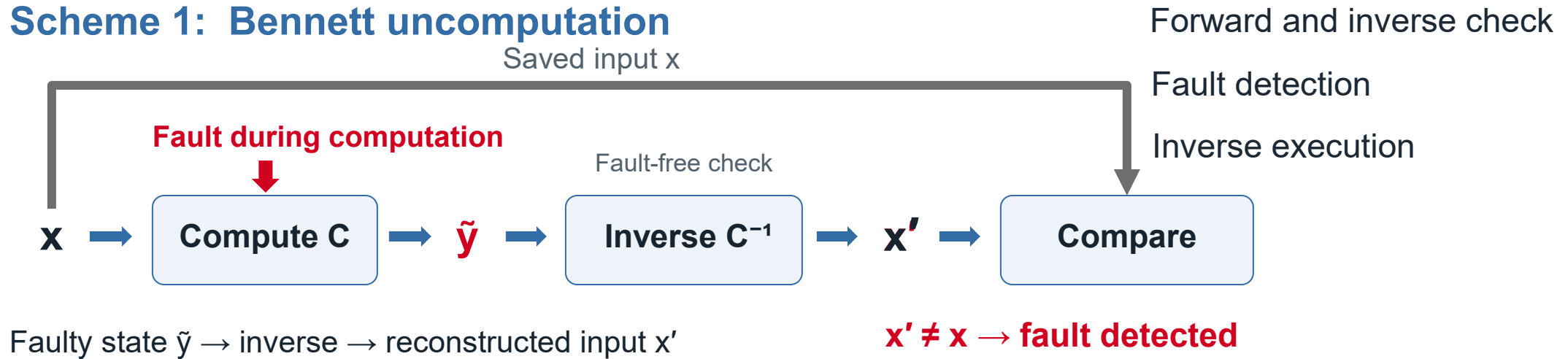
Input: 000



Expected after NOT: 111

Two reversibility-based defense schemes

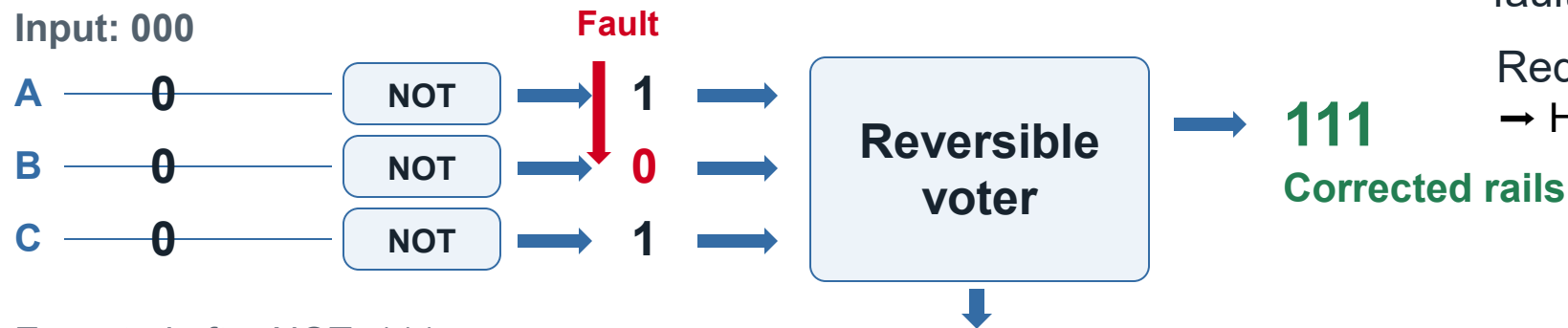
Scheme 1: Bennett uncomputation



Scheme 2: Secure gates: SNOT example

Three NOT gates, followed by a reversible voter.

Input: 000



Expected after NOT: 111

Disagreement retained $\rightarrow$ detected

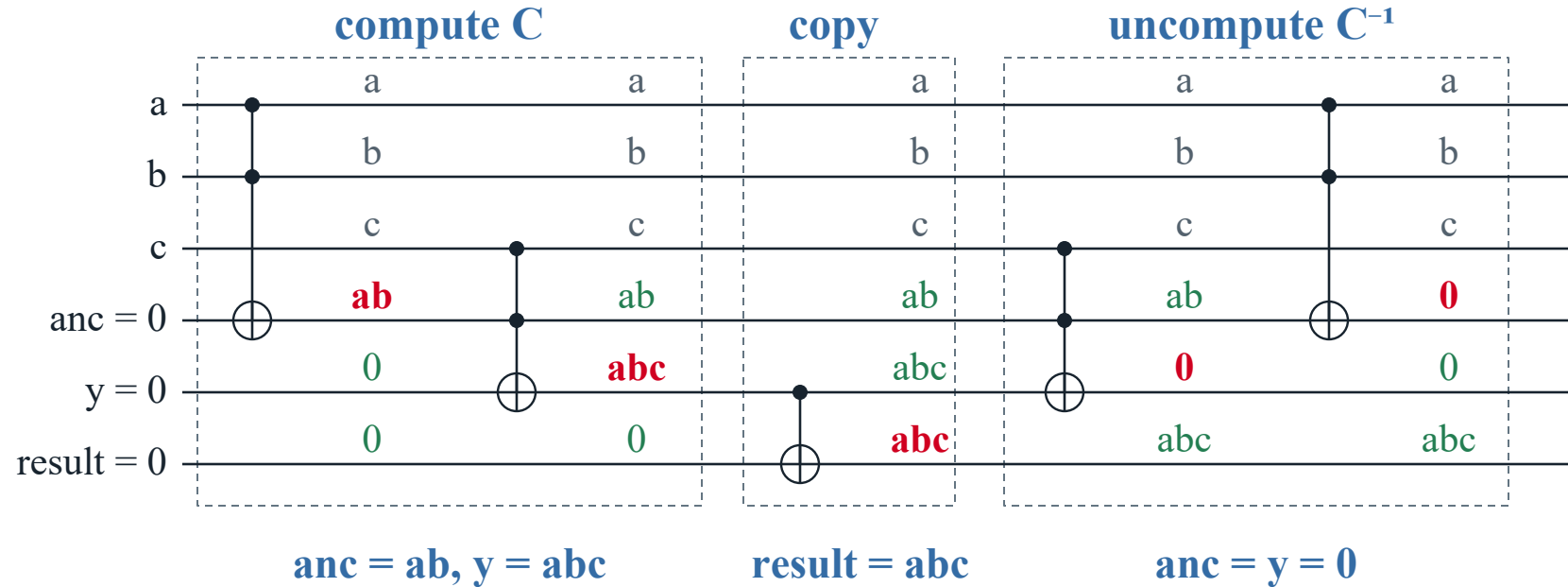
Independent rails and voters

local on-the-fly correction & fault detection

Redundancy at each gate
 $\rightarrow$ High gate cost

Bennett uncomputation

STATE AFTER EACH GATE · NO-FAULT EXECUTION



$$ab = a \wedge b \quad abc = a \wedge b \wedge c$$

Red: value updated by this gate

$$C' = C^{-1} \circ \text{copy} \circ C$$

Working ancillas return to zero. The copied result stays abc.

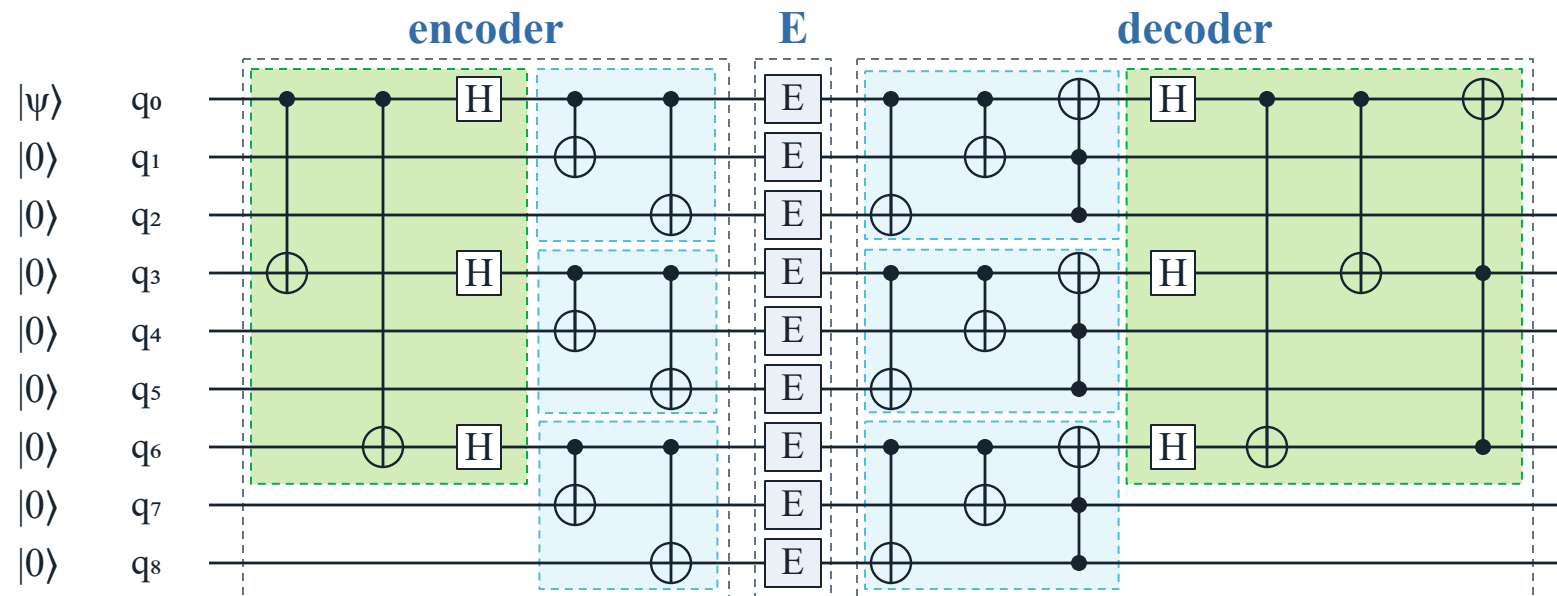
Shor's $[[9,1,3]]$ code

$\boxed{H}$ Hadamard gates

$\boxed{E}$ Error channel

QUANTUM ERROR CORRECTION

Shor's code uses **9** physical qubits to encode **1** logical qubit, with a code distance of **3**. This allows it to correct any single-qubit error.



Outer repetition: phase-flip protection

Inner repetition in each three-qubit block: bit-flip protection

The classical restriction retains one inner bit-flip block.

The connection between quantum gates and classical reversible logic

On computational basis states (such as $|0\rangle$ and $|1\rangle$), NOT, CNOT and Toffoli behave exactly like their classical reversible counterparts.

Quantum description

CNOT: $|c, t\rangle \rightarrow |c, t \oplus c\rangle$

Example: $|1, 0\rangle \rightarrow |1, 1\rangle$

Classical description

CNOT: $(c, t) \rightarrow (c, t \oplus c)$

Example: $(1, 0) \rightarrow (1, 1)$

Restrict to 0 / 1 basis-state inputs

Logical reversibility

The complete output uniquely determines the input.

Our hardware implementation

Conventional CMOS FPGA
No claim of physical reversibility

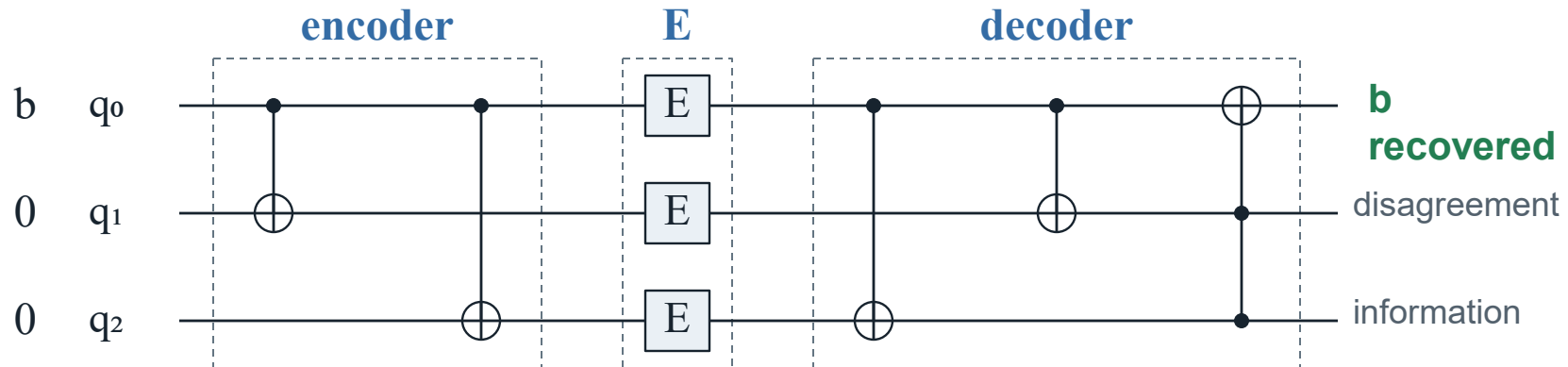
Next: retain the bit-flip repetition block from Shor's code.

Classical Reversible Projection of Shor's code

SHOR $[[9,1,3]]$ AND THE CLASSICAL $[3,1,3]$ REPETITION CODE

Shor $[[9,1,3]]$: one logical qubit in nine physical qubits.

Retain one inner bit-flip repetition block. Omit the phase-protection layer.



Encode: $(b, 0, 0) \mapsto (b, b, b)$

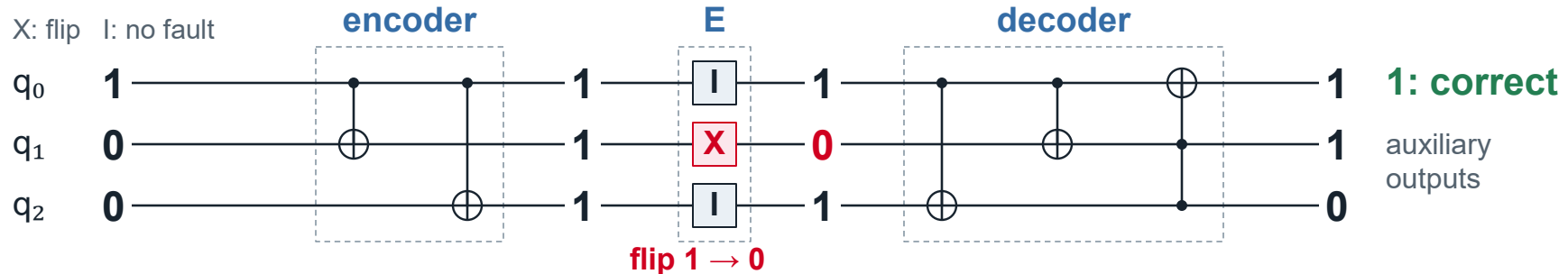
Decode: recover q_0 by majority correction; retain disagreement information.

Corrects one bit flip in E with fault-free encoding and decoding.

Faults beyond the noise channel

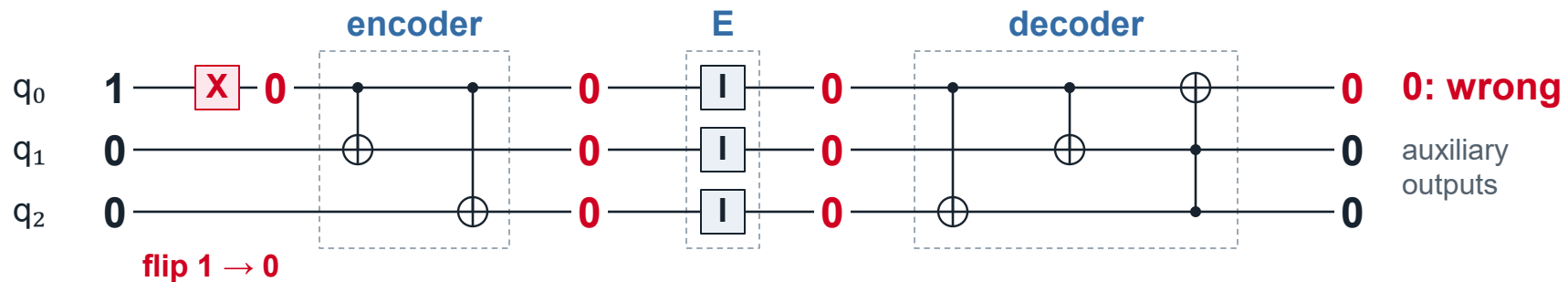
Same circuit as slide 7. Intended bit $b = 1$ in both examples.

A. One bit flips inside E, after encoding



The decoder recovers $q_0 = 1$; the other outputs retain disagreement.

B. The input bit flips before encoding



Three wrong copies agree. Decoding cannot recover the original 1.

Encoder and decoder gates, including the voter, can also fail.

I : No change

X : Bit flip ($0 \leftrightarrow 1$)

The secure-gate library

SCHEME 2

Three rails per logical wire

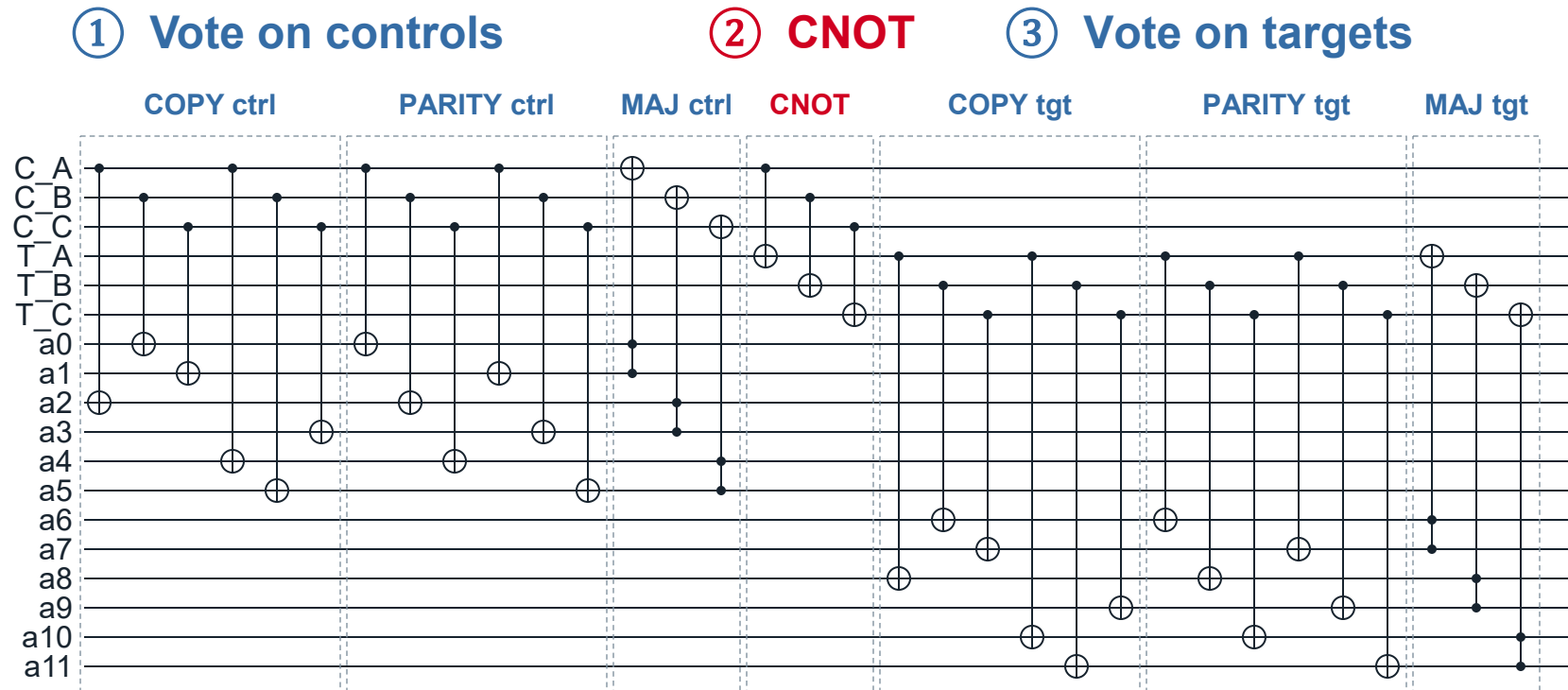
Independent operations, reversible voters and retained ancillas.

	Gates			
Secure gate	NOTs	CNOTs	CCNOTs	Ancillas
SNOT	3	12	3	6
SCNOT	0	27	6	12
SCCNOT	0	36	12	18

Each voter uses 12 CNOTs, 3 Toffolis and 6 ancillas.

SCNOT: control, operation, target

Three control rails · Three target rails · Twelve ancillas initialized to 0



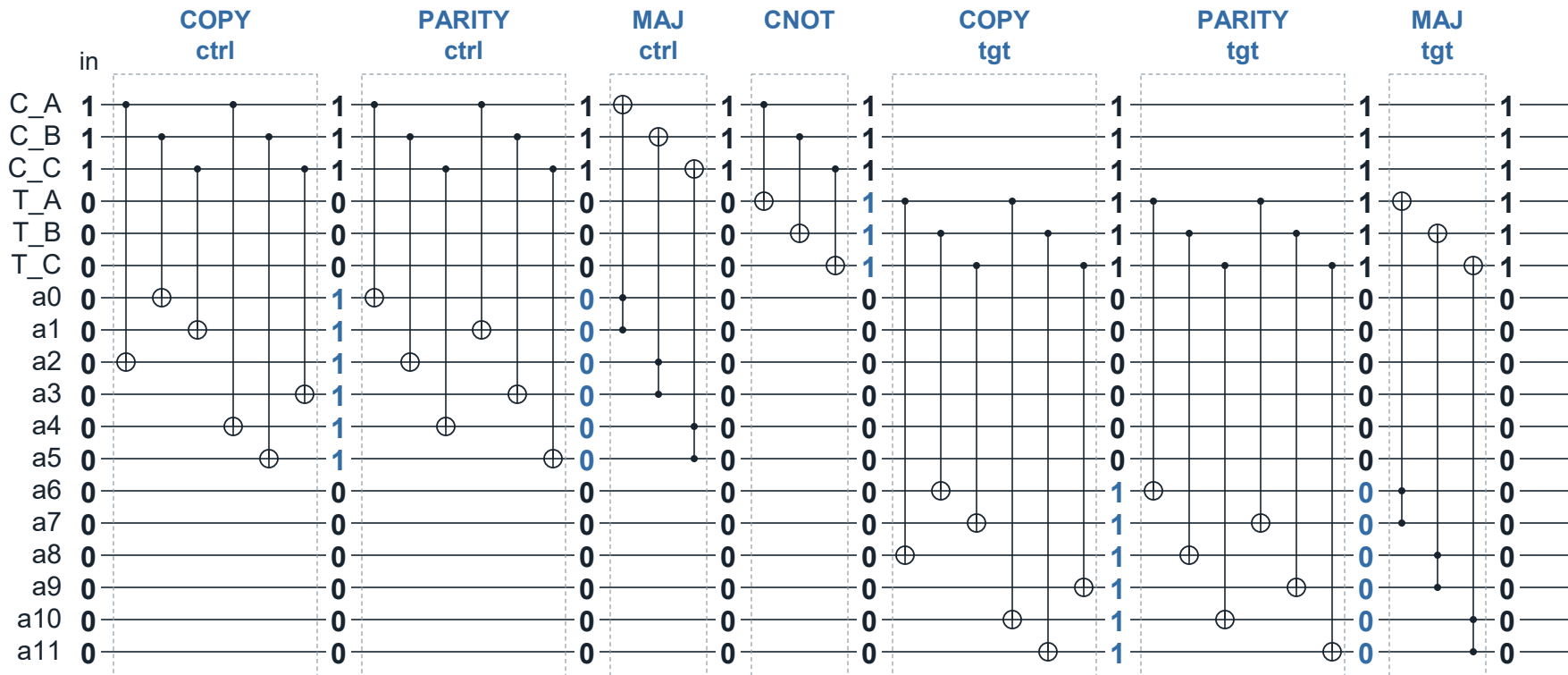
COPY → PARITY → MAJ: record disagreements, then correct by majority.

Ancilla outputs retain disagreement information for detection.

SCNOT without faults: $C = 1, T = 0$

Input: $C = 111, T = 000$, all ancillas = 0

Numbers show each wire after every stage. Blue = changed in that stage.



PARITY: $1 \oplus 1 = 0 \rightarrow$ MAJ makes no correction.

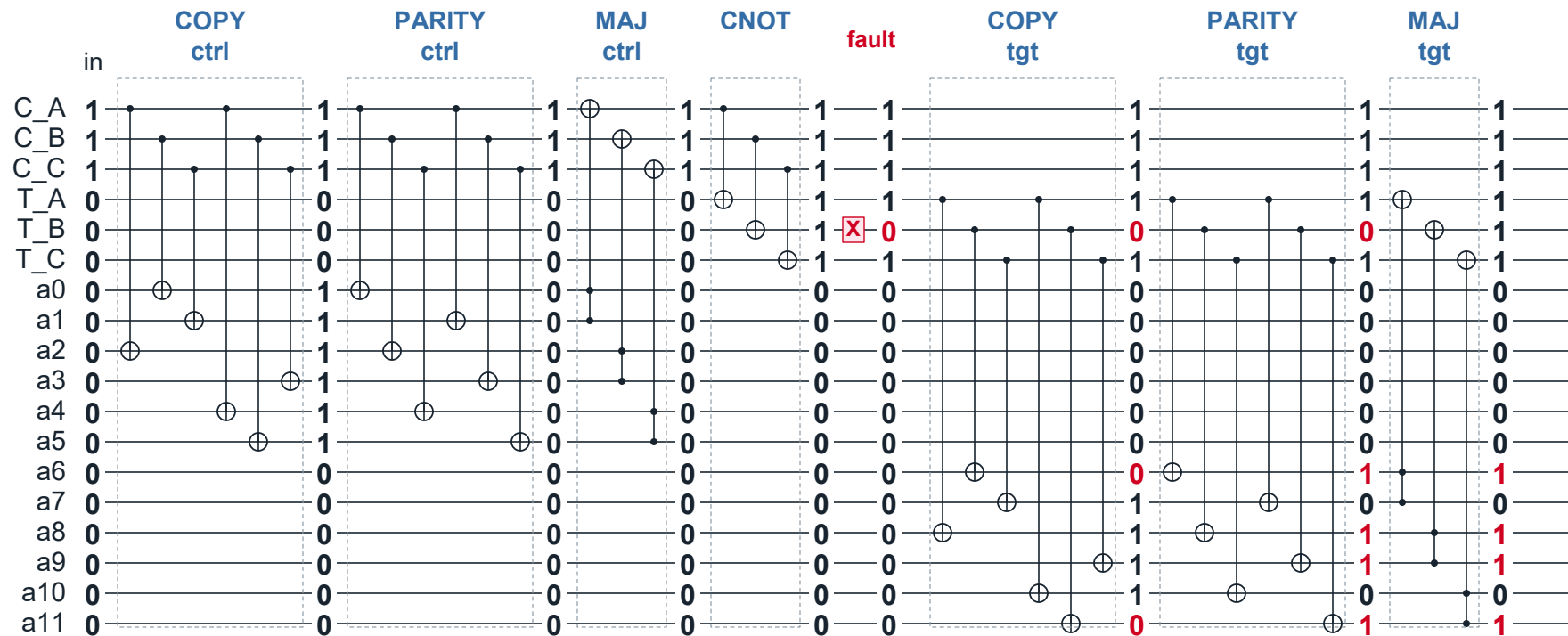
Output: $C = 111, T = 111$; all 12 ancillas = 0.

SCNOT: 1 flipped target rail

Expected T = 111 → fault gives 101 → voter outputs 111

C = 111, T = 000 initially; ancillas = 0. Faults occur before the target voter.

Red bit: differs from the fault-free value. Black bit: matches. Red X: injected flip.



Parity (p, q, r) = 101 Retained a6...a11 = 101101

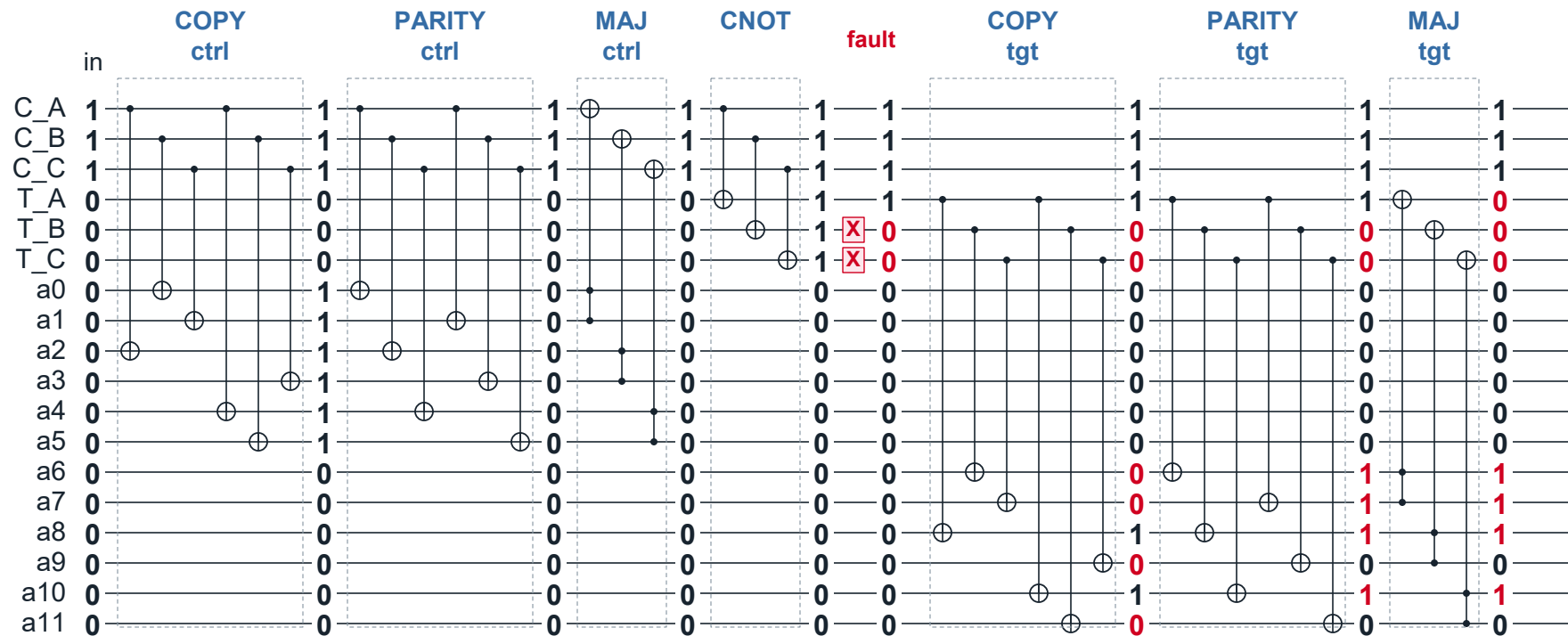
Corrected and detected

SCNOT: 2 flipped target rails in the same rail group

Expected T = 111 → fault gives 100 → voter outputs 000

C = 111, T = 000 initially; ancillas = 0. Faults occur before the target voter.

Red bit: differs from the fault-free value. Black bit: matches. Red X: injected flip.



Parity (p, q, r) = 110 Retained a6...a11 = 111010

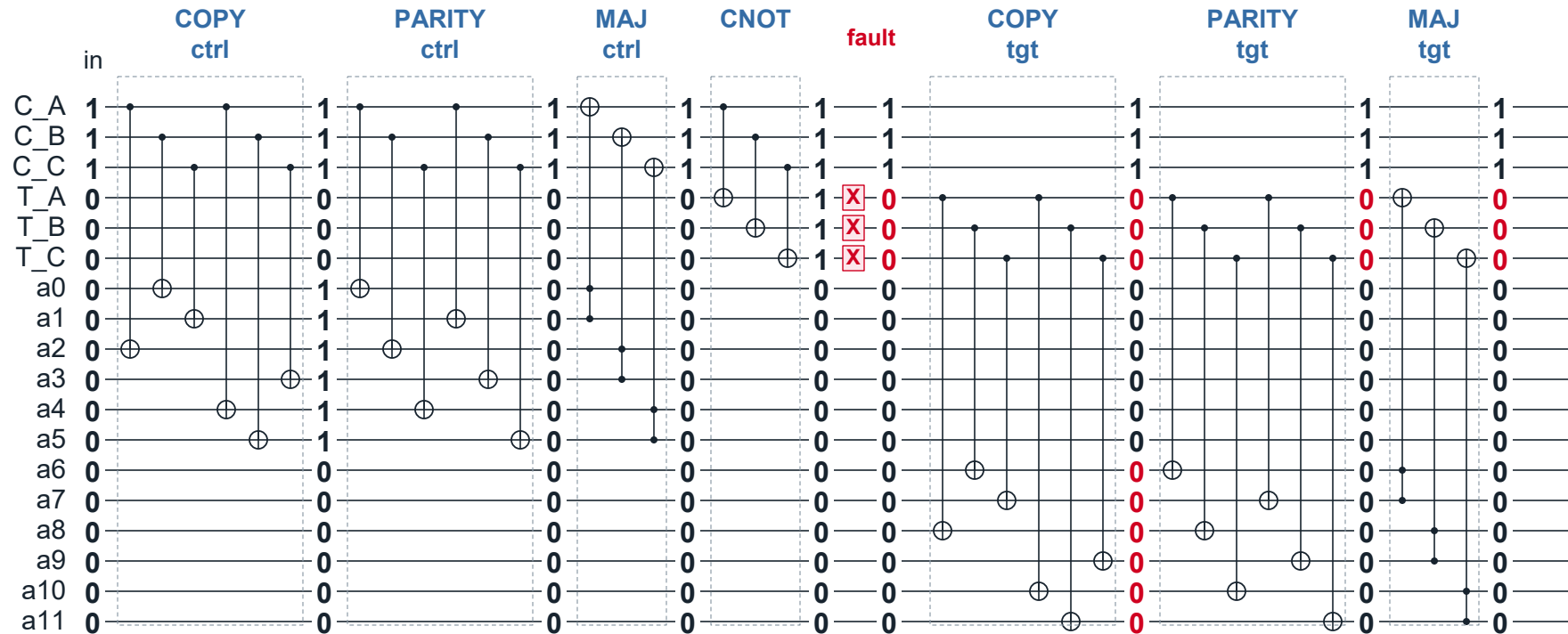
Detected, but not corrected

SCNOT: 3 flipped target rails in the same rail group

Expected T = 111 → fault gives 000 → voter outputs 000

C = 111, T = 000 initially; ancillas = 0. Faults occur before the target voter.

Red bit: differs from the fault-free value. Black bit: matches. Red X: injected flip.



Parity (p, q, r) = 000 Retained a6...a11 = 000000

Wrong output; local parity misses the error

However, a fault-free inverse check reveals a mismatch with the original input.

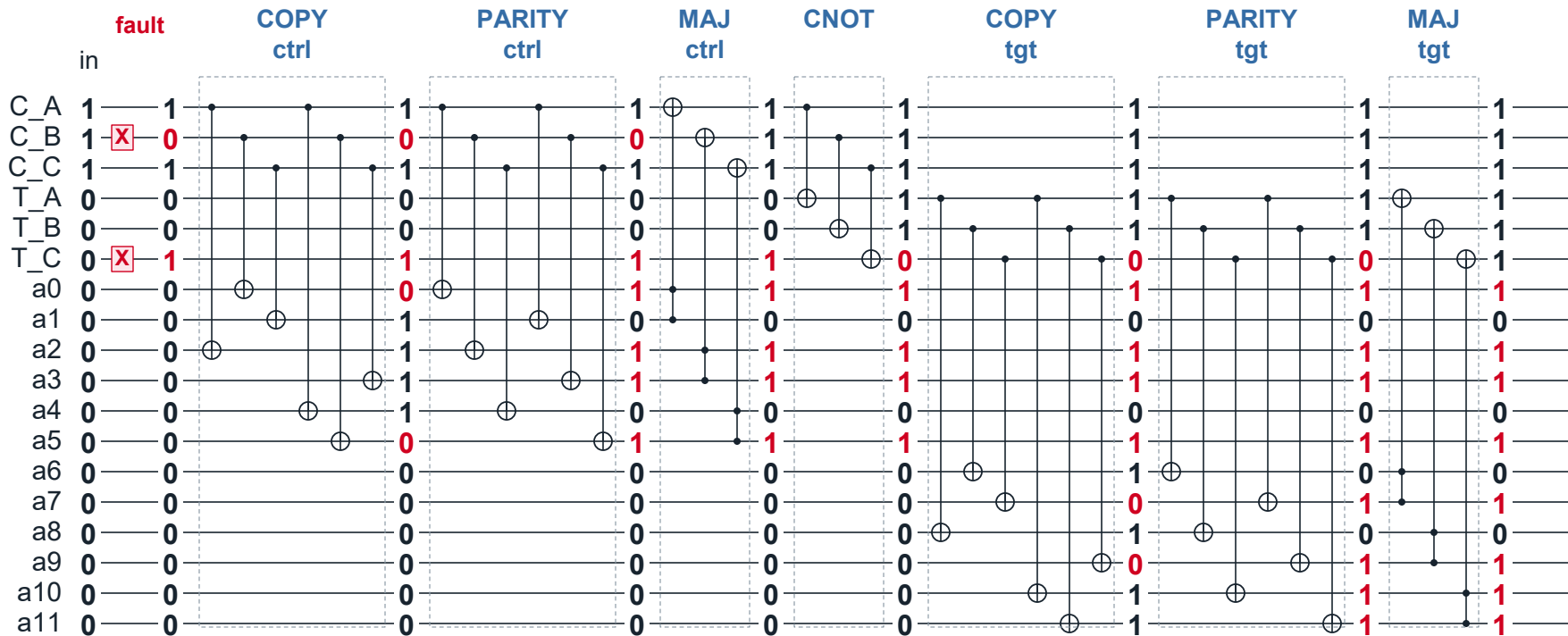
Since errors are uniformly distributed, this case barely happens.

VFI example: one fault in each rail group

Two simultaneous input flips: C_B and T_C

Initial C = 111, T = 000; ancillas = 0. All unmarked gates operate correctly.

Red X: injected flip. Red bit: differs from fault-free value. Black bit: matches.



After MAJ ctrl: C = 111 After MAJ tgt: T = 111

Retained a0...a5 = 101101 a6...a11 = 010111

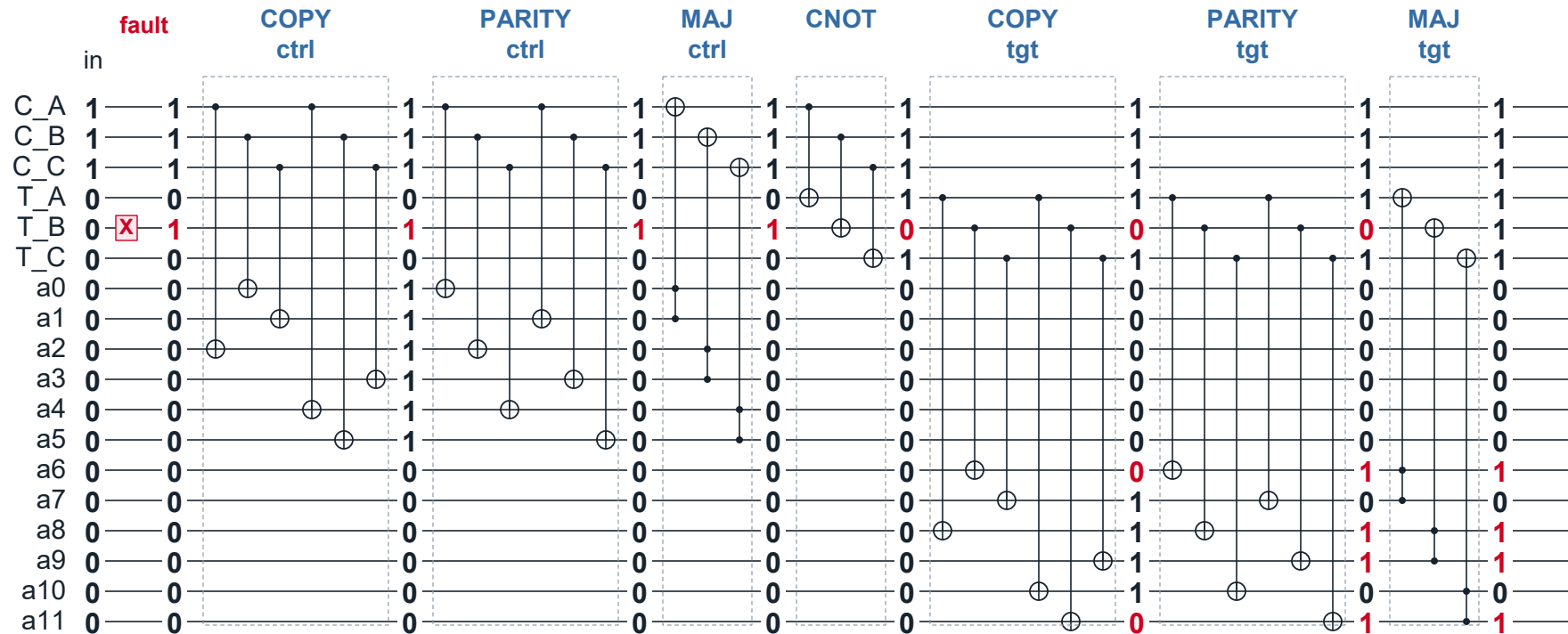
Both faults corrected and detected: one bad rail per voter group.

LFI example: one localized rail fault

One input flip: T_B; the control group is unaffected

Initial C = 111, T = 000; ancillas = 0. All unmarked gates operate correctly.

Red X: injected flip. Red bit: differs from fault-free value. Black bit: matches.



After MAJ ctrl: C = 111 After MAJ tgt: T = 111

Retained a0...a5 = 000000 a6...a11 = 101101

The single rail fault is corrected; nonzero ancillas record it.

Experimental setup and outcome definitions

QVERILOG convert a combinational ASCON design into a reversible netlist.
We then apply Bennett's scheme or replace the gates with secure gates.

ASCON ON XILINX ARTIX-7

Campaign	VFI	LFI
Injector	ChipWhisperer Husky+	Alphanov LMS
FPGA target	XC7A35 on CW312T	XC7A200T on Nexys Video
Fault character	Supply-wide disturbance	Spatially localized laser
Scheme 1 / Scheme 2	ASCON round / secure S-box	ASCON round / secure S-box

Corrected Correct output with evidence of perturbation

Detected only Wrong output with a fault indication

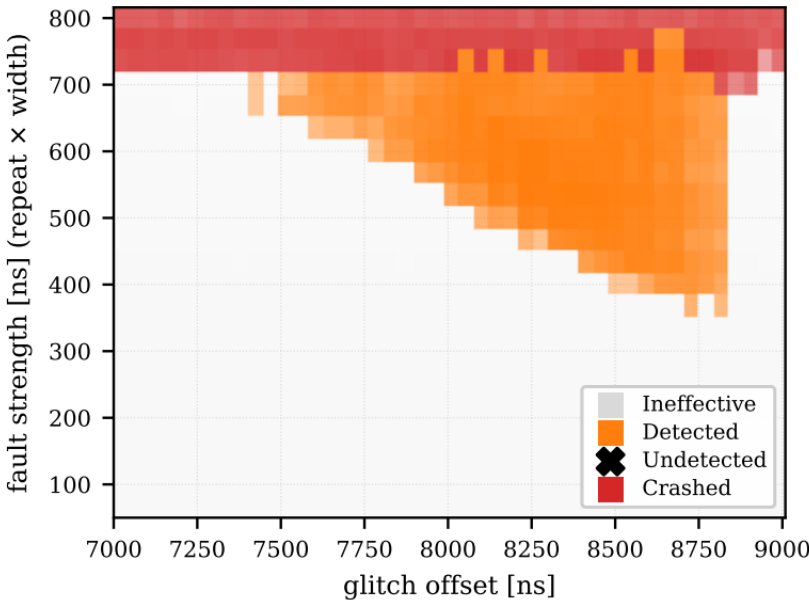
Undetected Wrong output without a fault indication

Correction rate = corrected / effective in-scope faults.

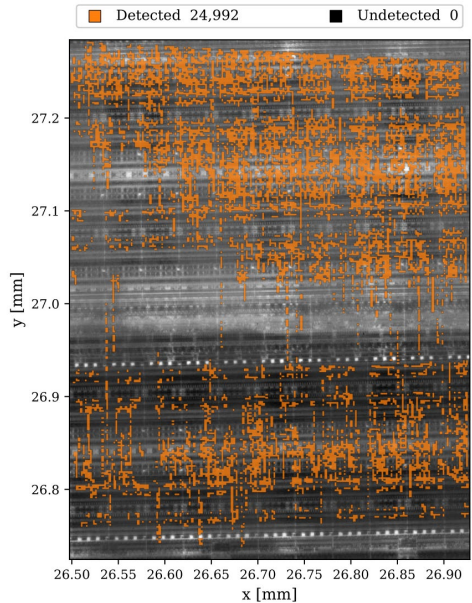
Bennett: fault detection on an ASCON round

No undetected output errors observed in either reported campaign

Voltage fault injection (VFI)



Laser fault injection (LFI)



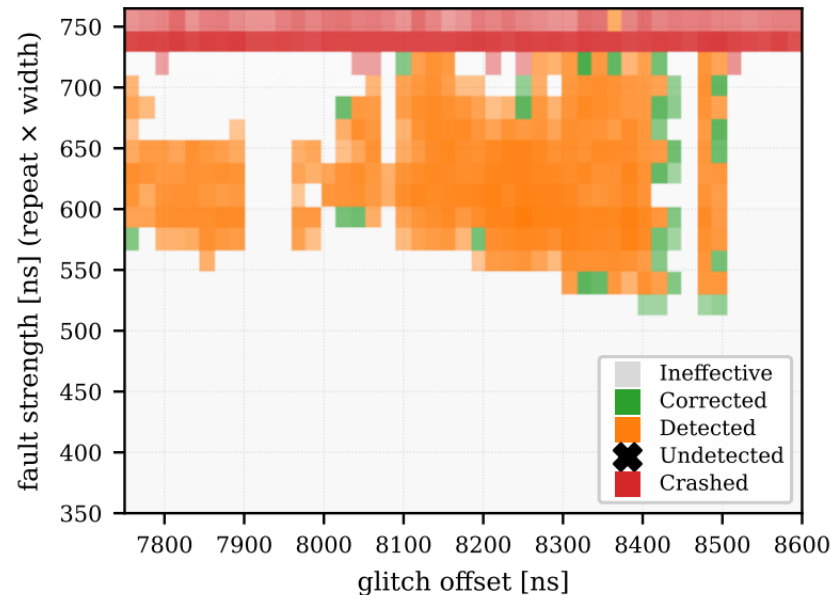
Campaign	Trials	Detected	Undetected
VFI	1,000,000	26,567	0
LFI	870,350	24,992	0

100% detection among the reported effective in-scope faults.

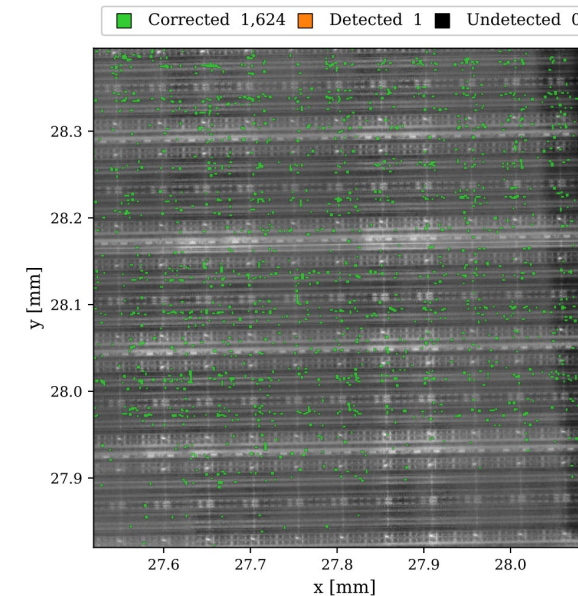
Secure gates: correction depends on fault locality

ASCON S-box protected by SNOT, SCNOT and SCCNOT

VFI: 17.6% corrected



LFI: 99.9% corrected



Campaign / trials	Corrected	Detected only	Undetected
VFI / 1,000,000	1,705	7,960	0
LFI / 214,830	1,624	1	0

No undetected output errors observed in either campaign.

Key Takeaways

Reversibility as a Fault Defense

What we proposed

Bennett forward–inverse checking for fault detection.
SNOT, SCNOT and SCCNOT for fault on-the-fly correction and detection,
using redundant rails, reversible voters and retained ancillas.

What we implemented

We developed QVERILOG to synthesize reversible netlists from
SystemVerilog and apply Bennett and secure-gate transformations.
FPGA implementation and validation against physical fault injection.

What experiments validated

No undetected output errors observed in the reported VFI and LFI campaigns.
Secure S-box correction: 17.6% under VFI and 99.9% under LFI.

Correction rates are among effective in-scope faults in the tested campaigns.

Thank you

Questions?

Meng Yuan

Technische Universität Berlin
Security in Telecommunications

References

BACKUP D

This work

X. M. Saß, M. Yuan, and J.-P. Seifert, “Reverse to Reveal: Reversibility as a Fault Defense,” FDTC 2026.

- [3] C. H. Bennett, “Logical reversibility of computation,” IBM Journal of Research and Development, vol. 17, no. 6, pp. 525–532, 1973.
- [5] T. Toffoli, “Reversible computing,” in International Colloquium on Automata, Languages, and Programming. Springer, 1980, pp. 632–644.
- [7] P. W. Shor, “Scheme for reducing decoherence in quantum computer memory,” Physical Review A, vol. 52, no. 4, p. R2493, 1995.
- [11] P. Aliferis, D. Gottesman, and J. Preskill, “Quantum accuracy threshold for concatenated distance-3 codes,” Quantum Information and Computation, vol. 6, no. 2, pp. 97–165, 2006.

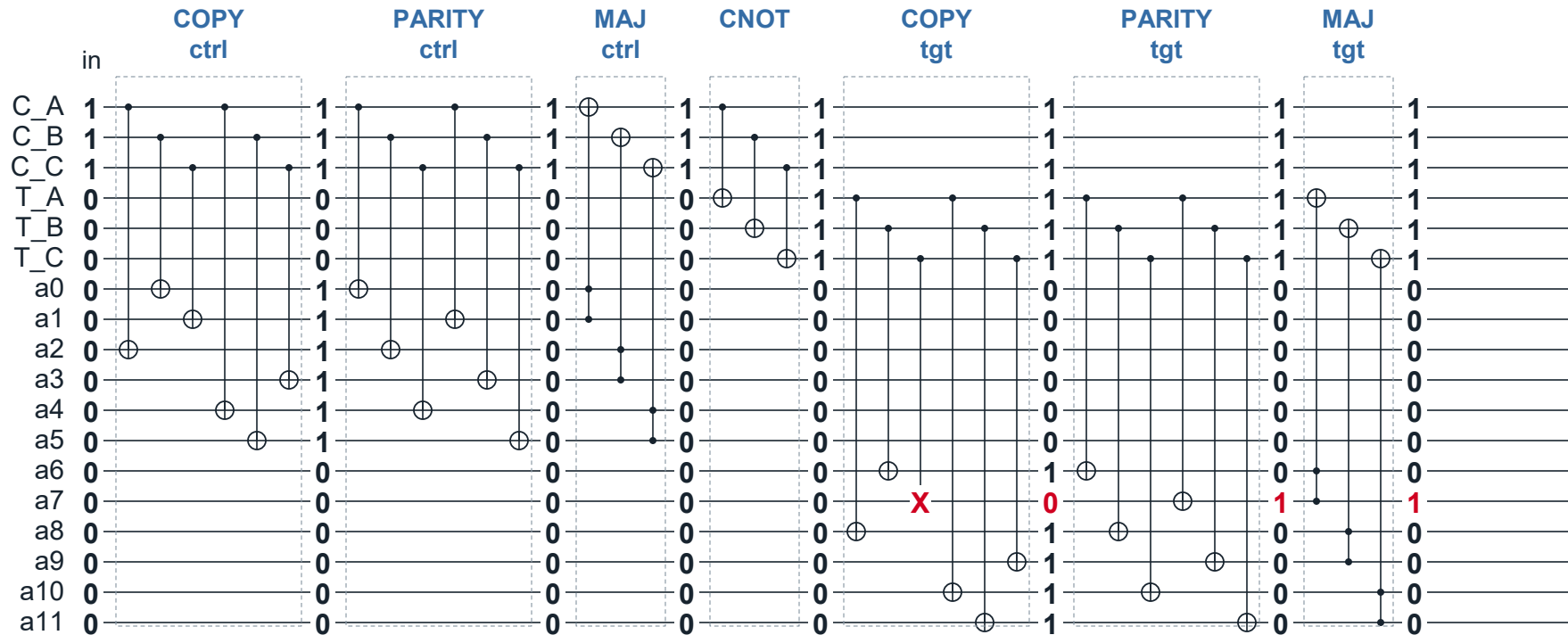
Reference numbers follow the paper.

SCNOT: an a7 fault inside COPY tgt

COPY tgt fault: a7 stays 0 instead of becoming 1

C = 111, T = 000 initially; ancillas = 0. All unmarked gates operate correctly.

Red bit: differs from the fault-free value. Black bit: matches. Red X: failed copy gate (MGF).



a7: faulty COPY 0 → PARITY 1 → MAJ 1

Correct output; ancilla fault detected

MAJ tgt: a6 AND a7 = 0 AND 1 = 0. Final C = 111, T = 111; a6...a11 = 010000.

RECAP: Classical repetition inspired by QEC

FOUNDATION FOR SCHEME 2

Shor $[[9,1,3]]$	Classical bit-flip restriction
Quantum bit and phase errors	Deterministic binary values
Inner three-qubit repetition block	Three rails for each logical bit
Recovery using quantum operations	NOT / CNOT / CCNOT network

Redundancy supplies correction.

Retained ancillas preserve the information needed for reversibility.

The implementation uses classical reversible logic on an FPGA.

Paper §§II-C, III-B. The phase-protection layer is omitted in the classical restriction.