Automatic Time-Redundancy Transformation for Fault-Tolerant Circuits

Dmitry Burlyaev
Pascal Fradet &
Alain Girault

Our Objectives Here

Develop a new *automatic* logic-level transformation for fault-tolerance:

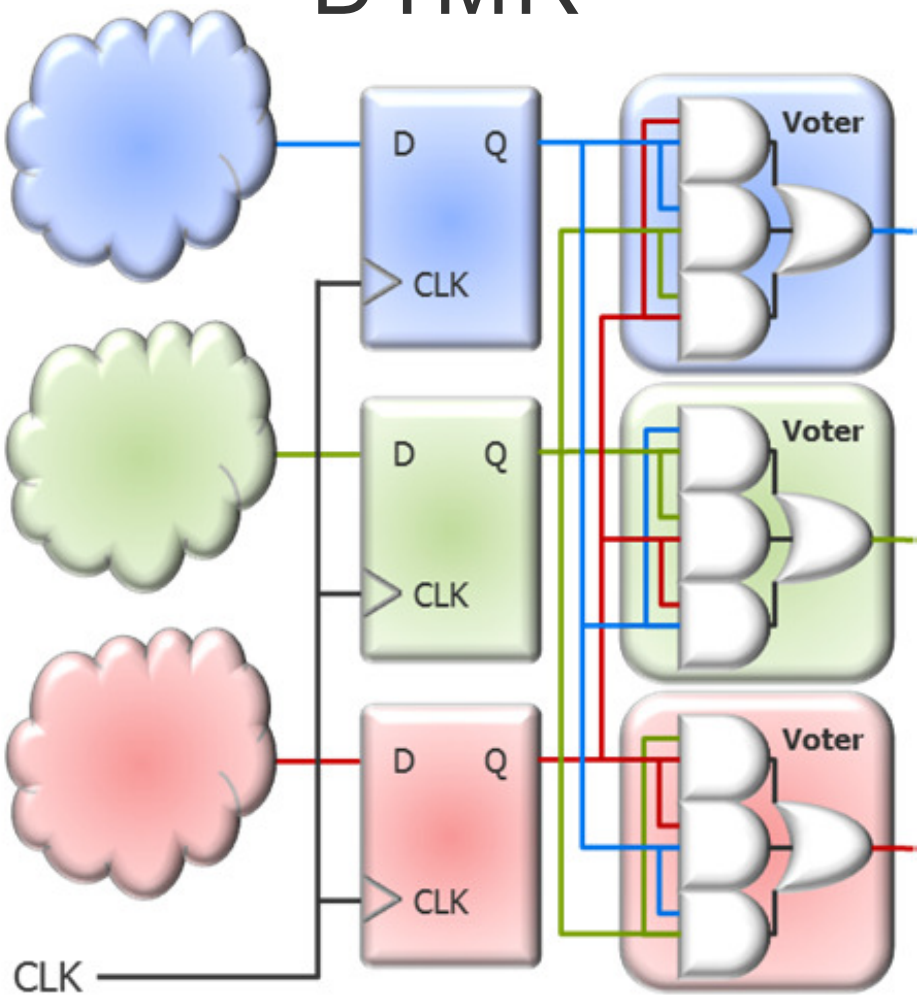
1. technologically independent (*eg*, no clock line control).
2. SET correction with double time redundancy and less HW overhead than in TMR.
3. input/output transparent behavior to SET.
4. formally provable with a proof assistant.

Considered Fault-Models

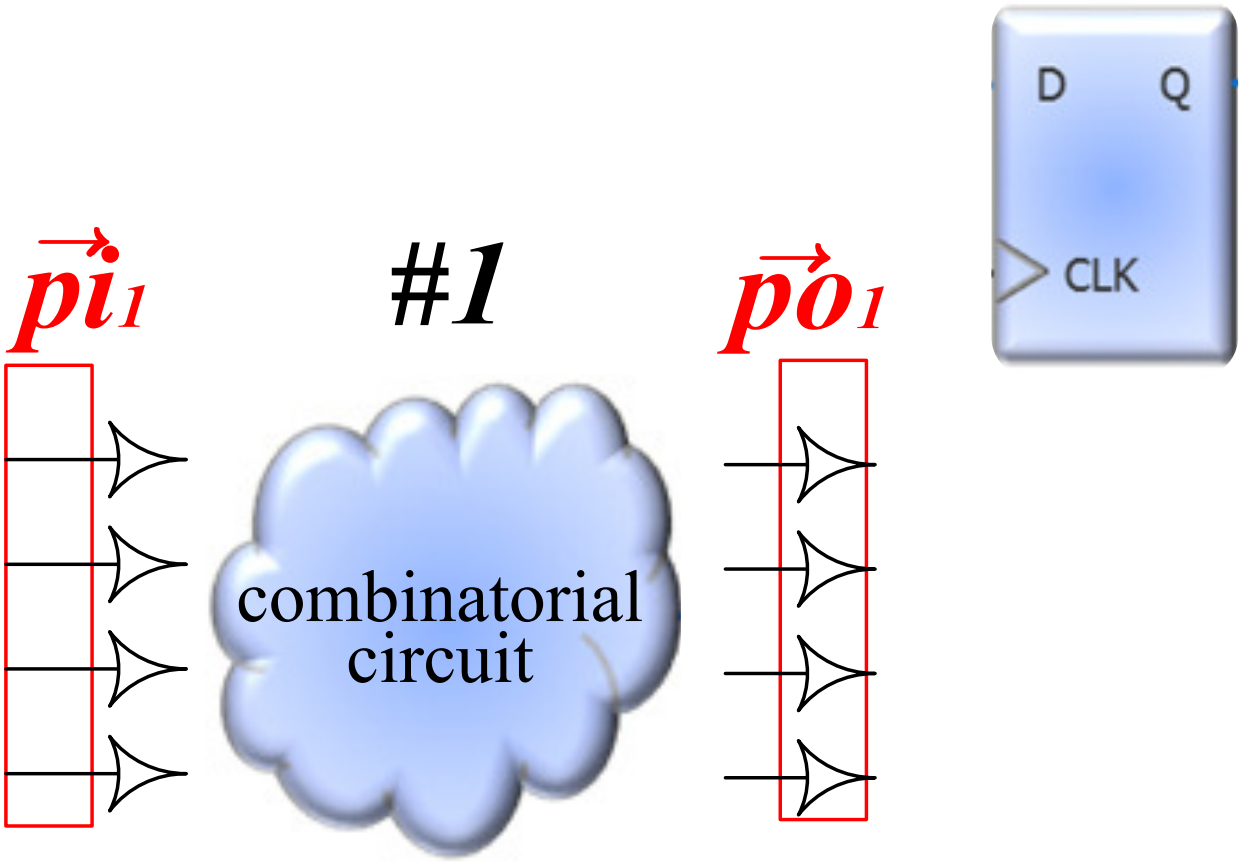
- Single-Event Upset (SEU)
  - a bit-flip
- Single-Event Transient (SET)
  - a current spike on a wire $\rightarrow$ SEU(s)

Automated TMR Schemes

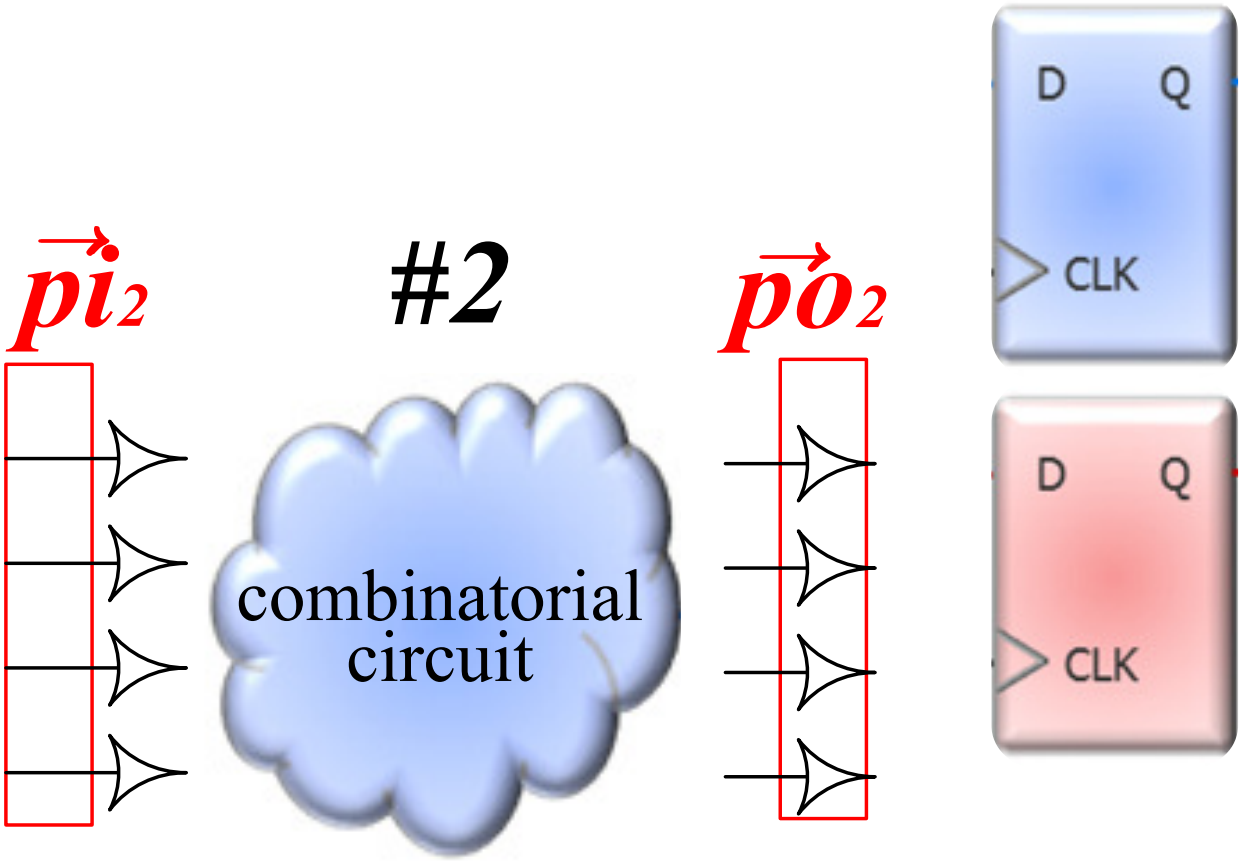
DTMR



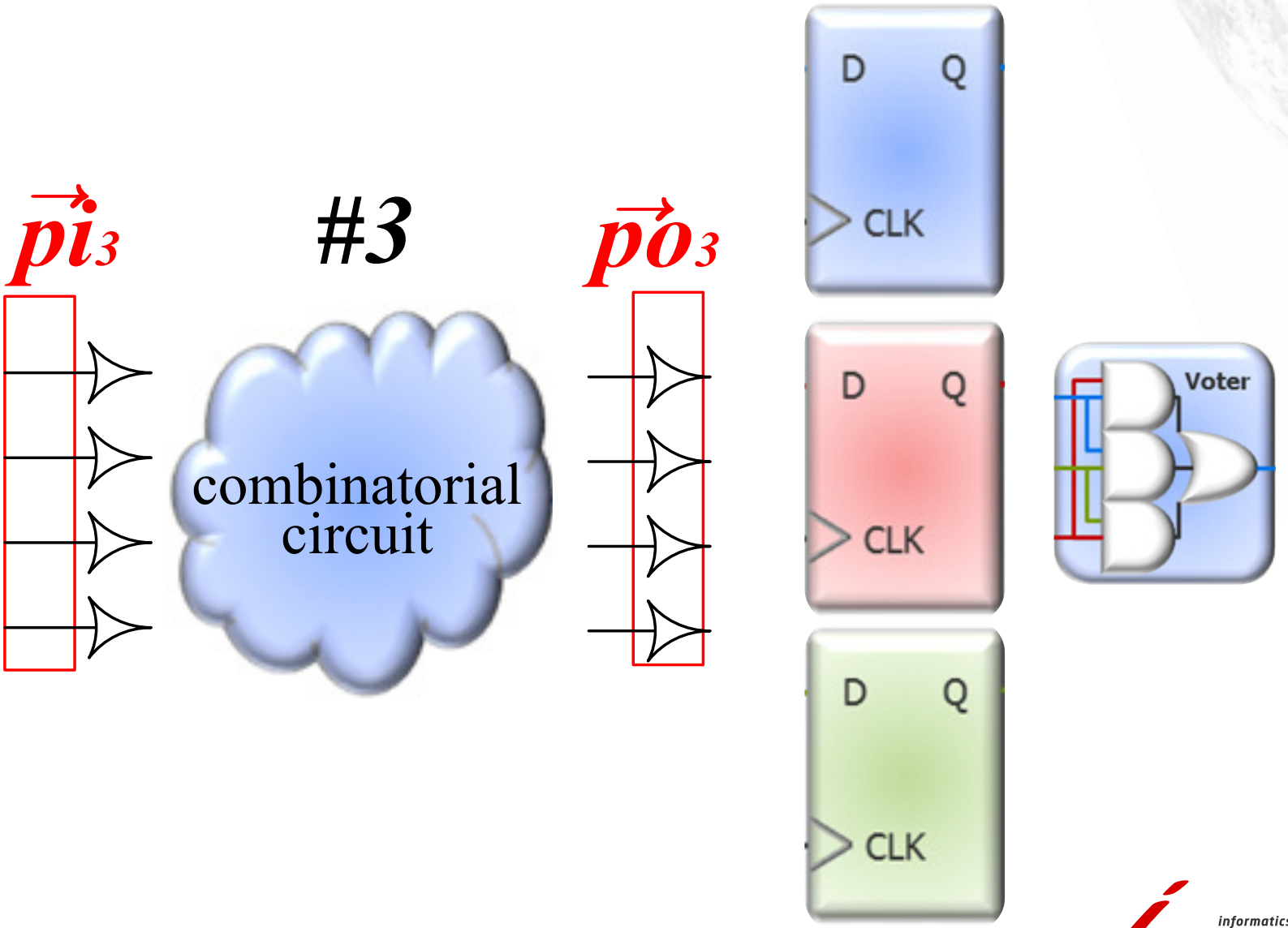
Triple-Time Redundancy



Triple-Time Redundancy



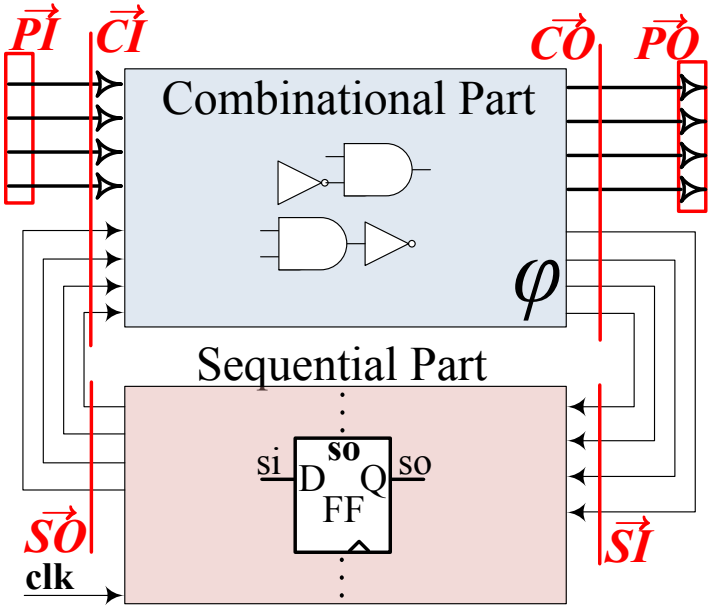
Triple-Time Redundancy

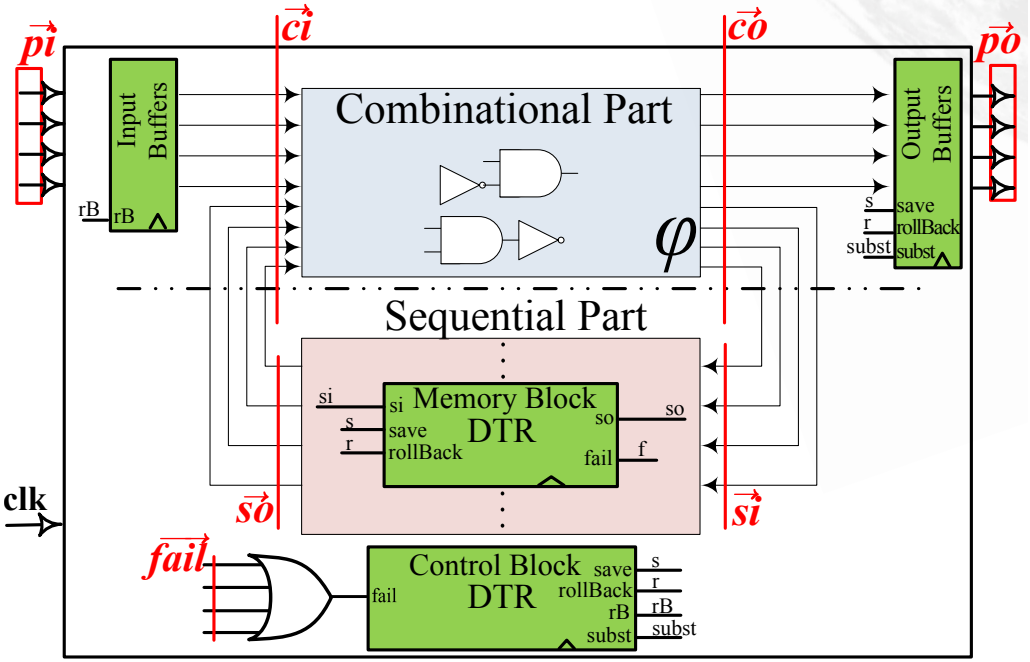


Double Time Redundancy Transformation

- micro checkpointing-rollback
- speed-up mode (switching-off time-redundancy)
- input/output buffers (input/output transparency)
- tolerance to **at most one SET in 10 clock cycles**

Transformation DTR

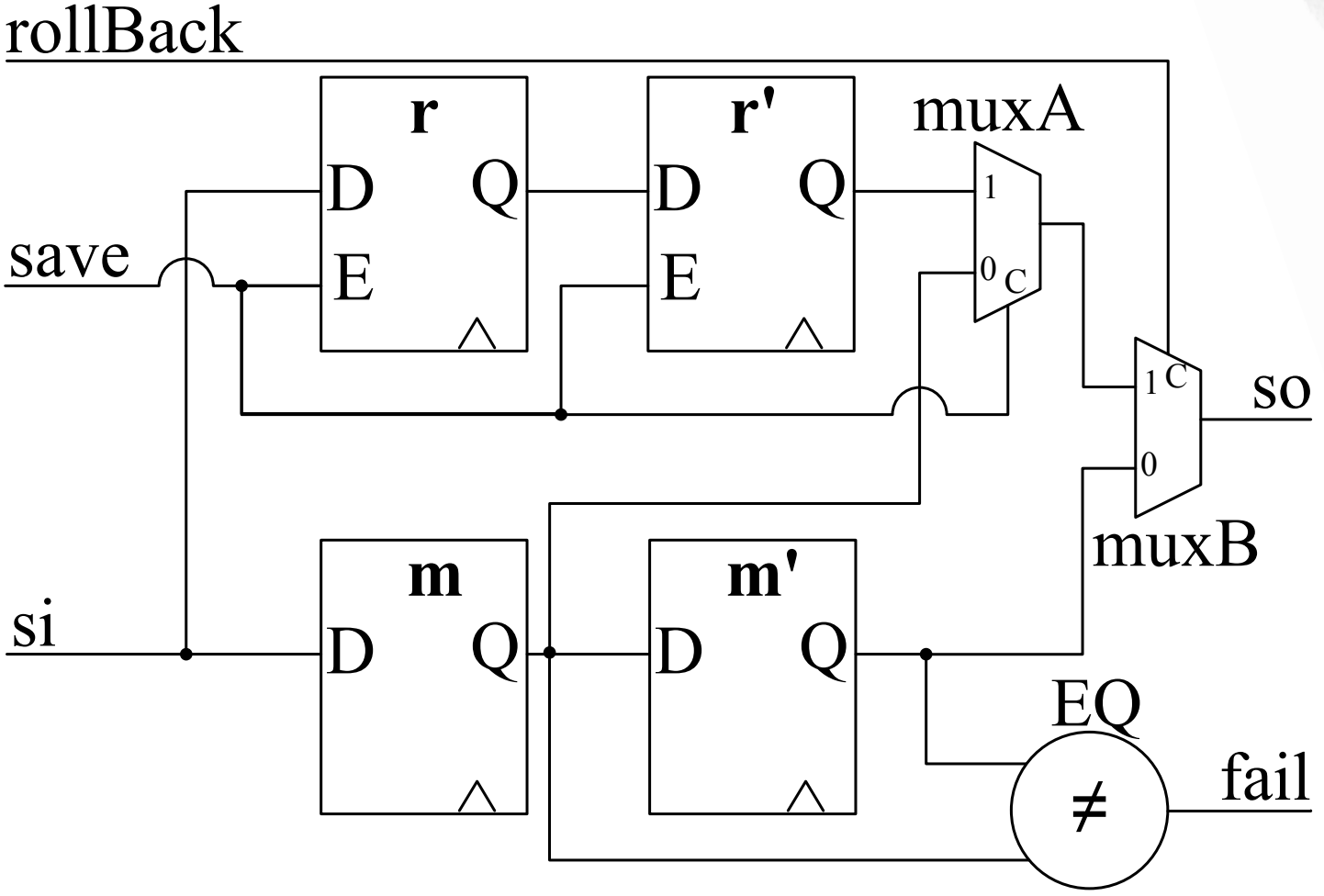




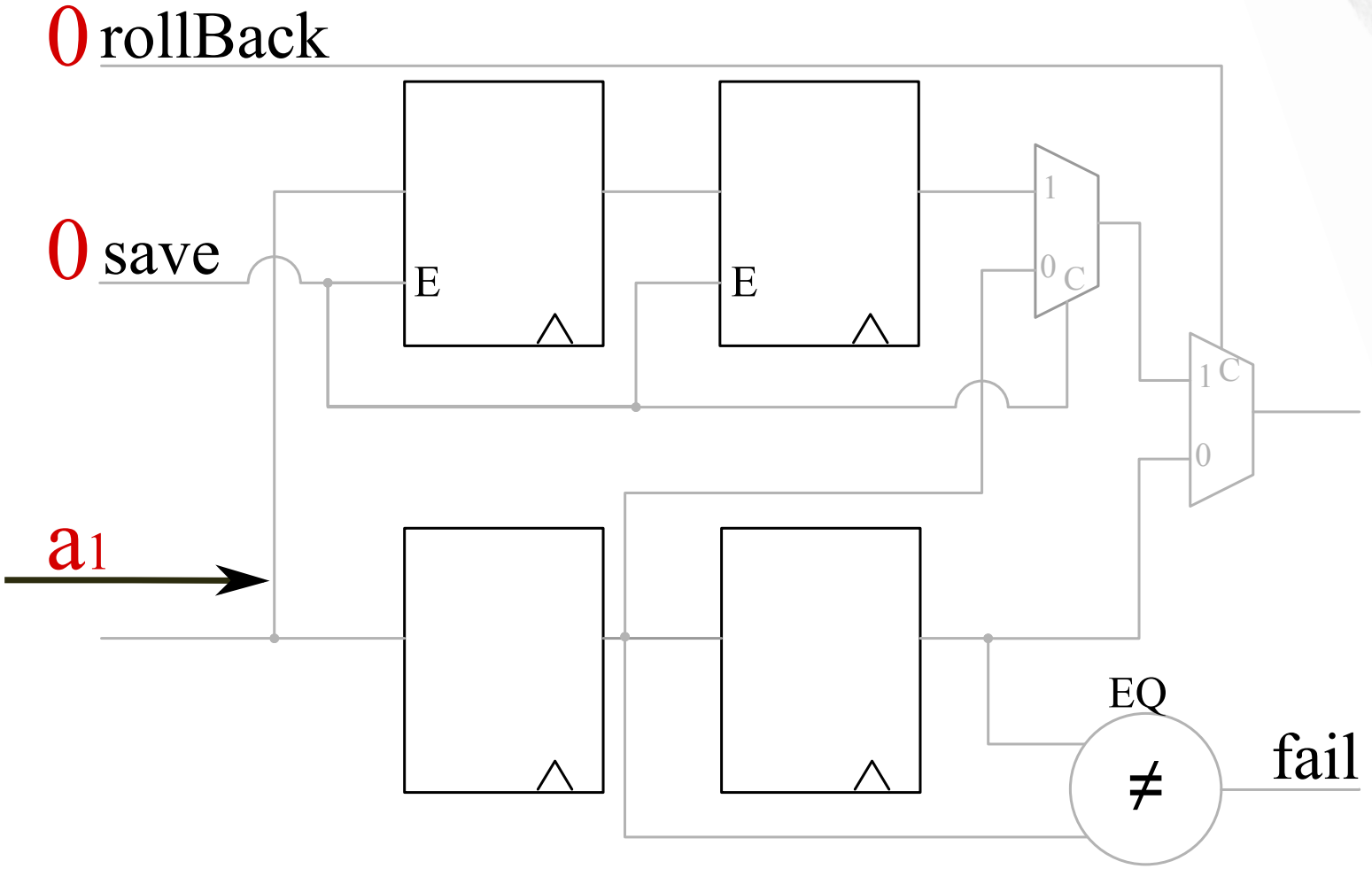
Input stream upsampling **x2** (original throughput/2)

1) Memory Cell ← Memory Block
2) Control Block Introduction
3) Input/Output Buffers Insertion

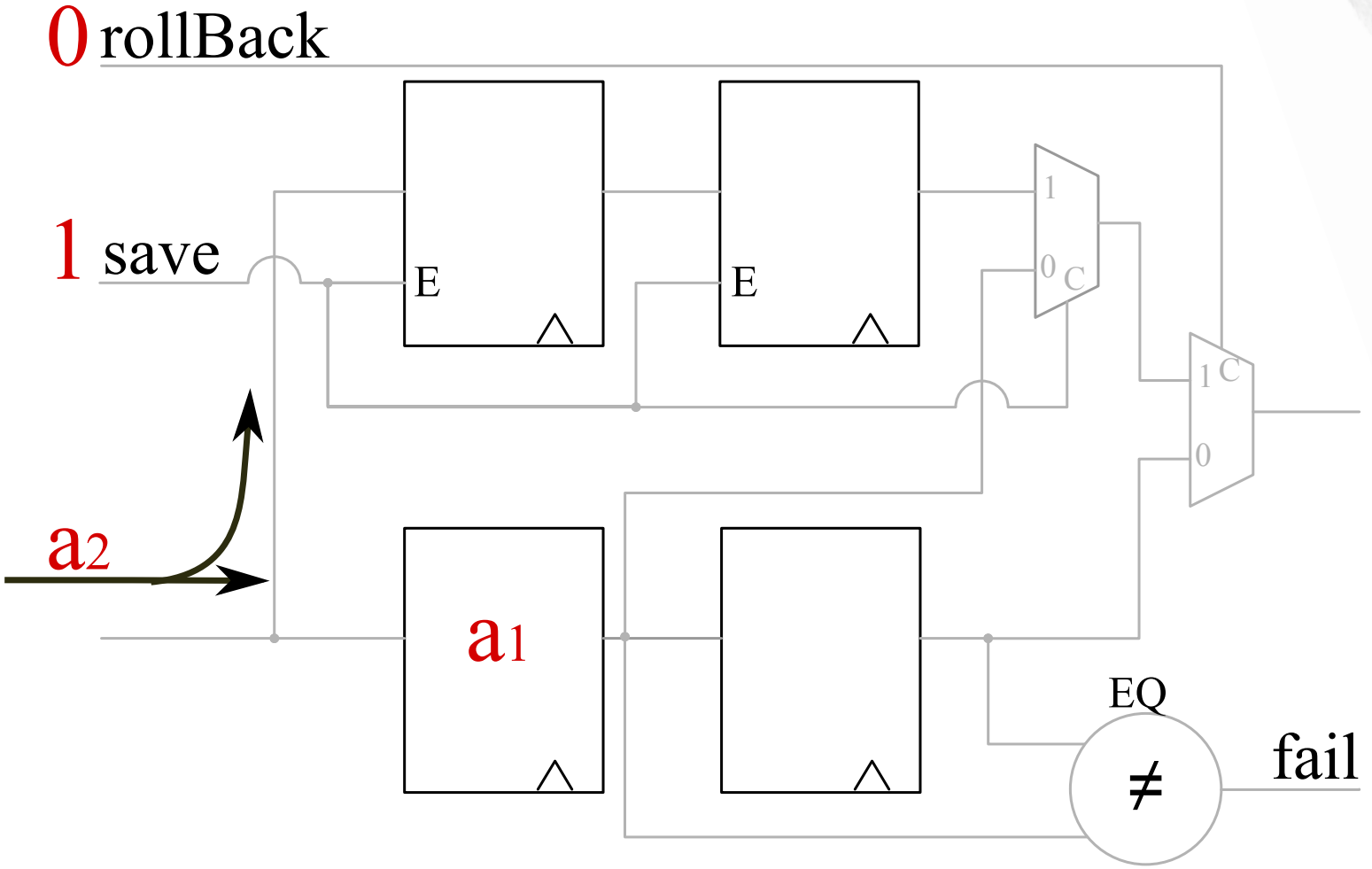
Memory Block: Working Cycle



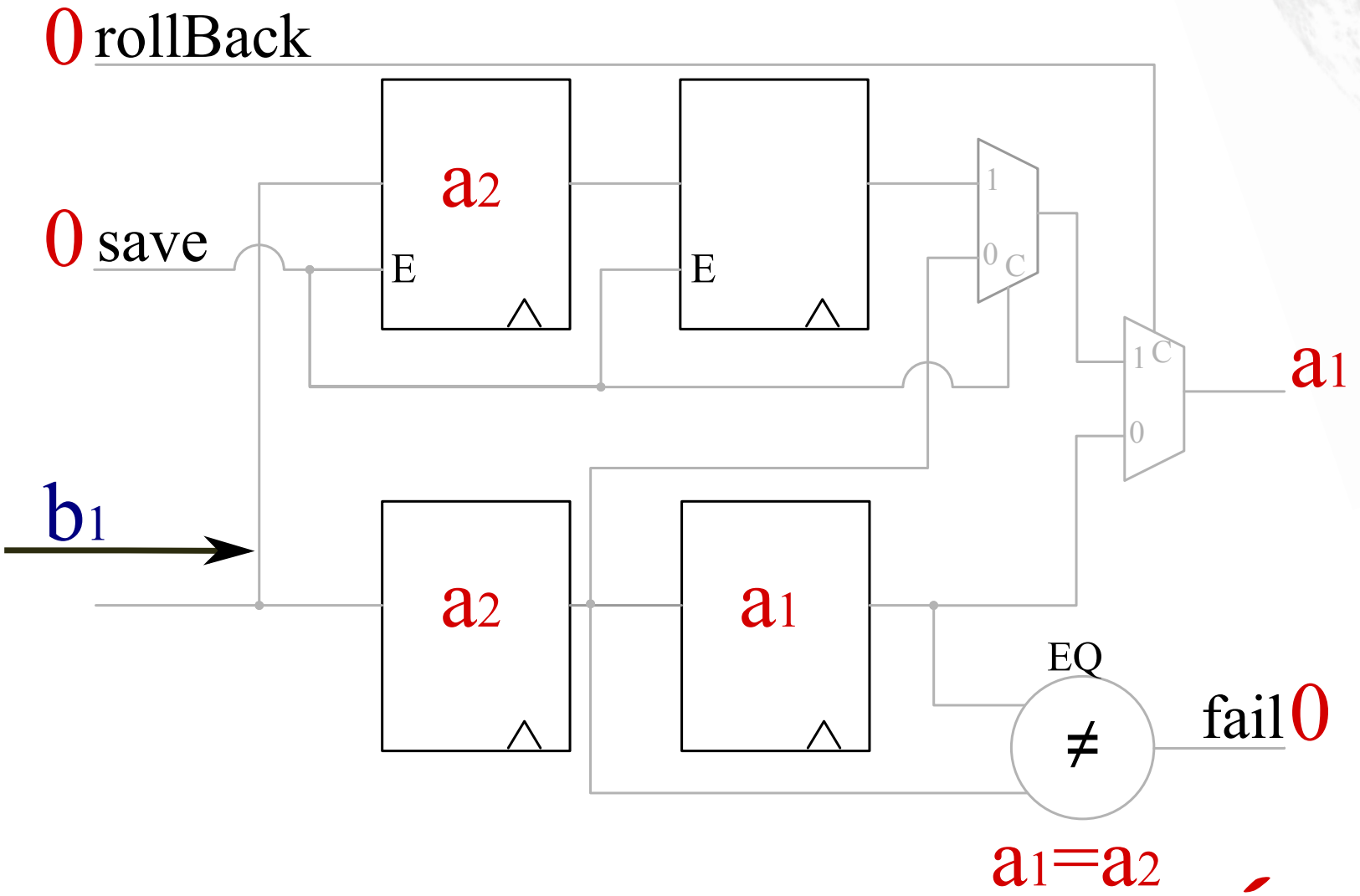
Memory Block: Working Cycle



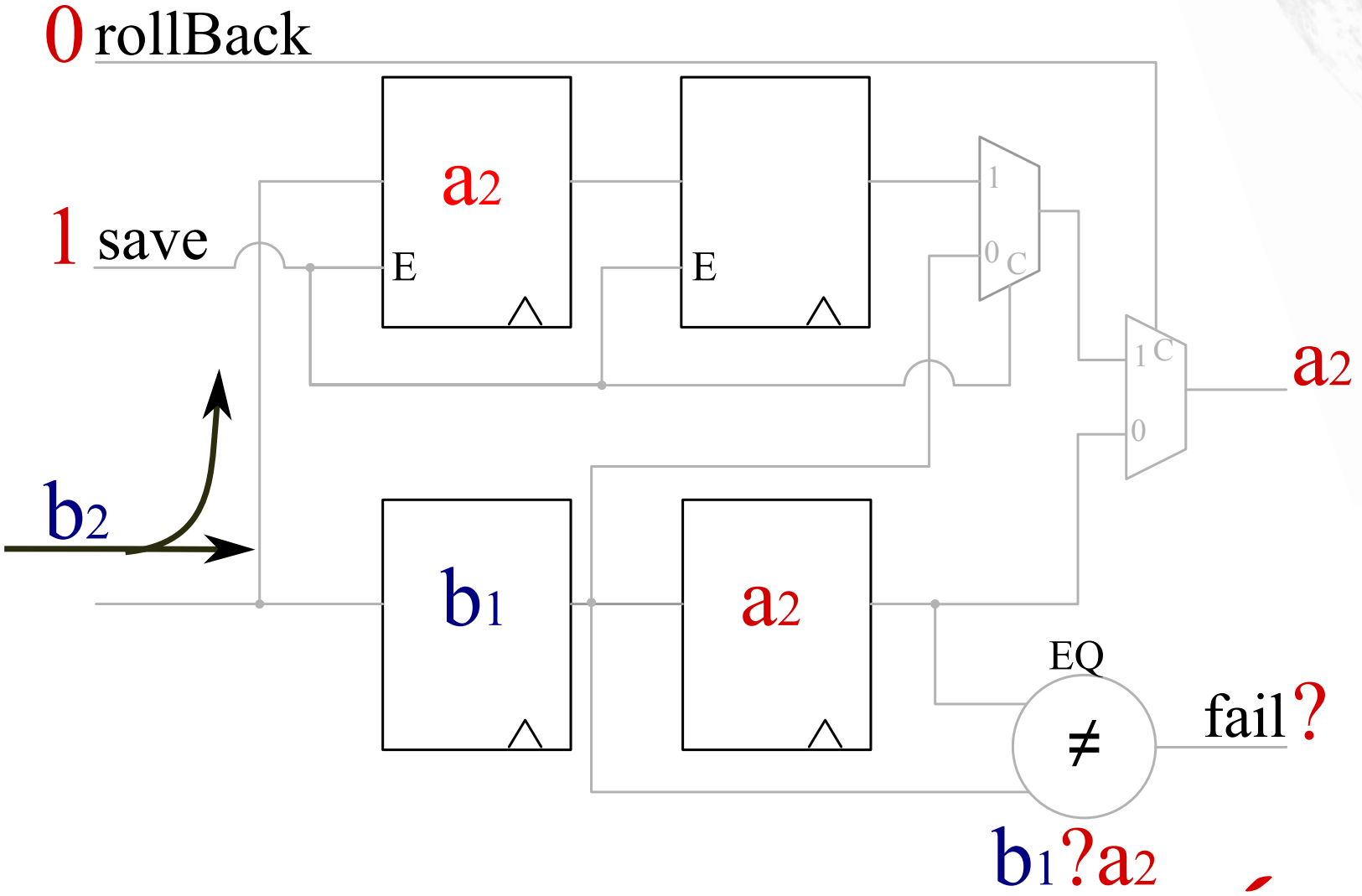
Memory Block: Working Cycle



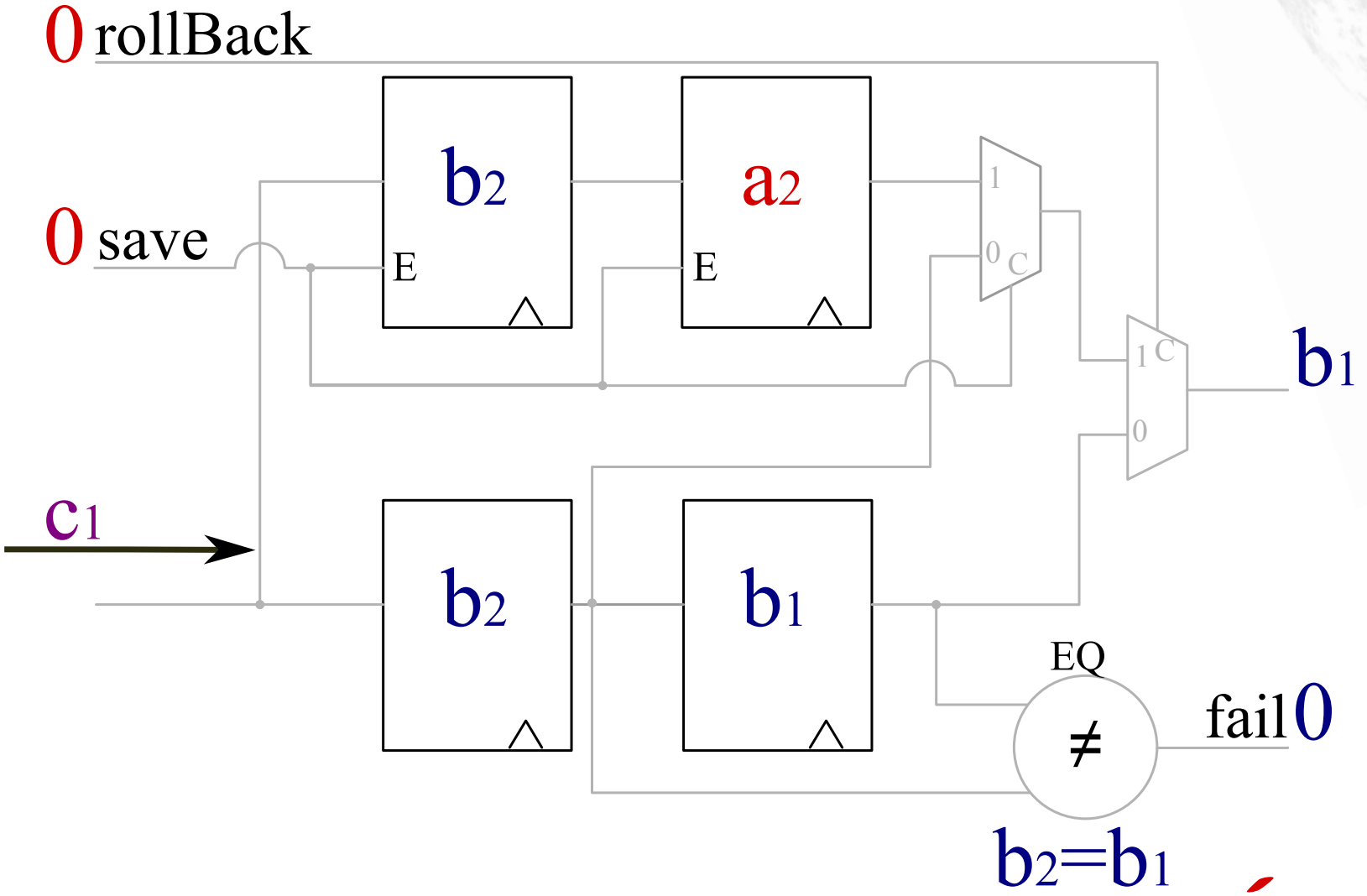
Memory Block: Working Cycle



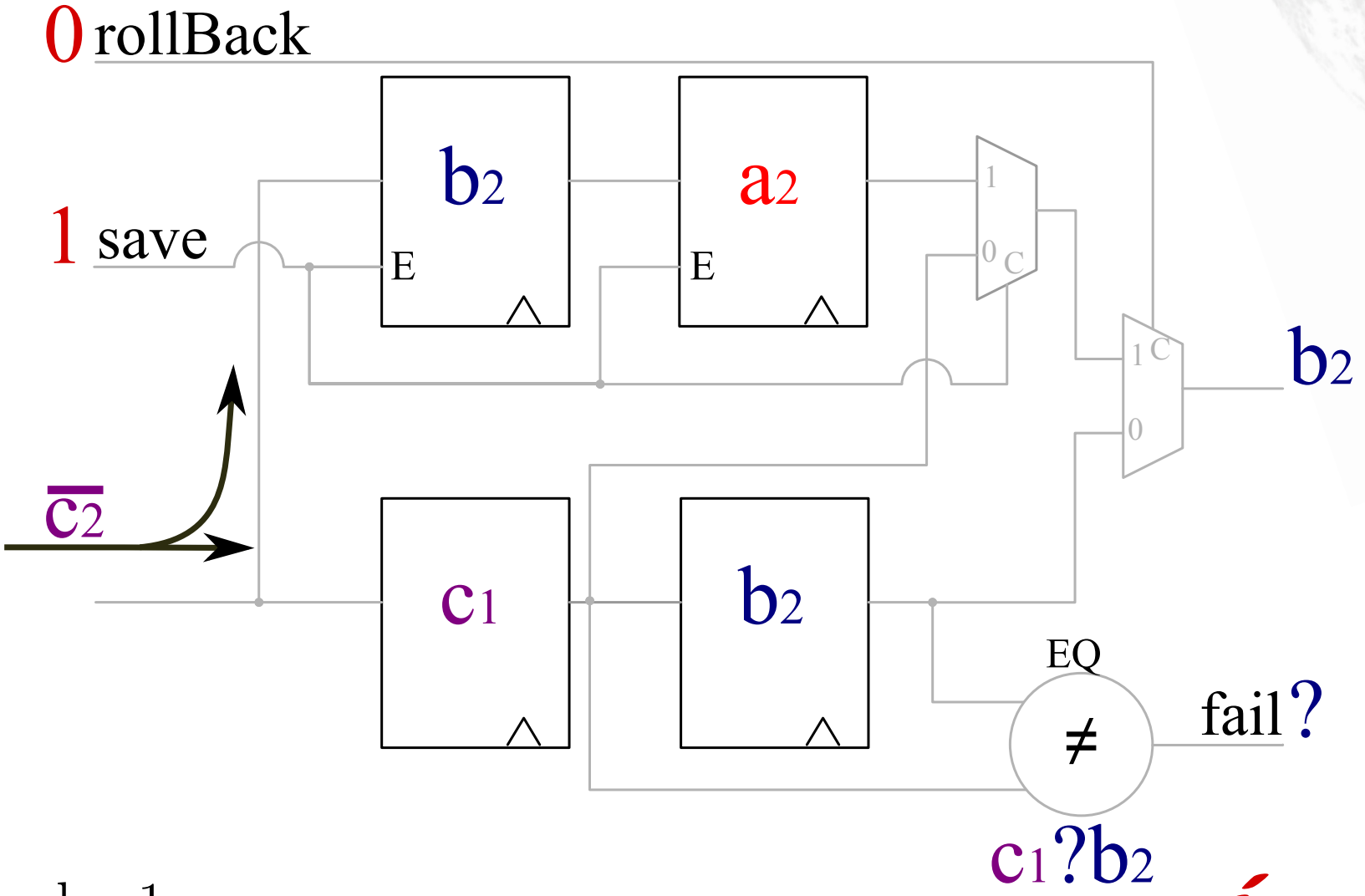
Memory Block: Working Cycle



Memory Block: Working Cycle

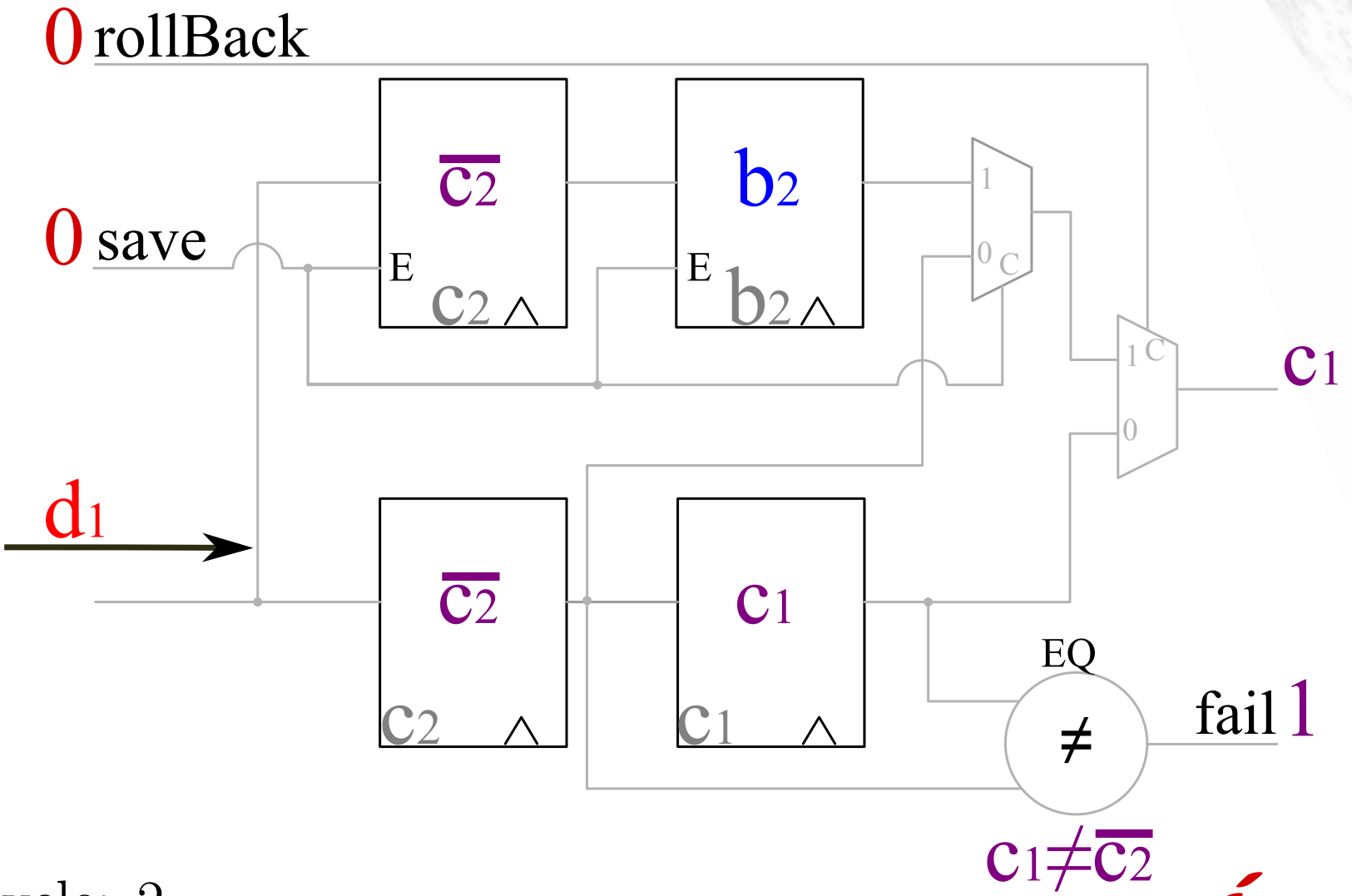


Memory Block: Working Cycle



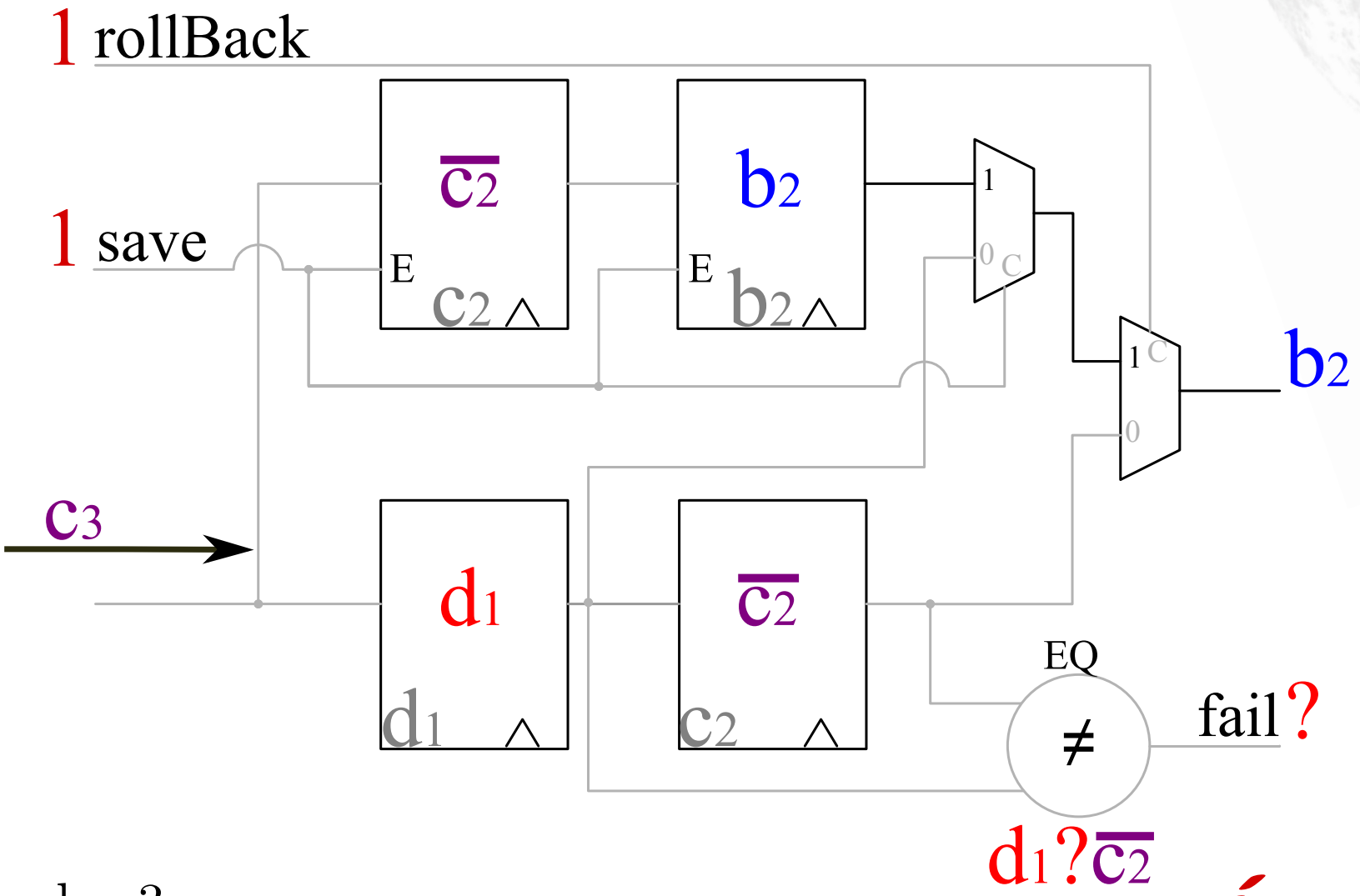
Cycle: 1

Memory Block: Error-detection



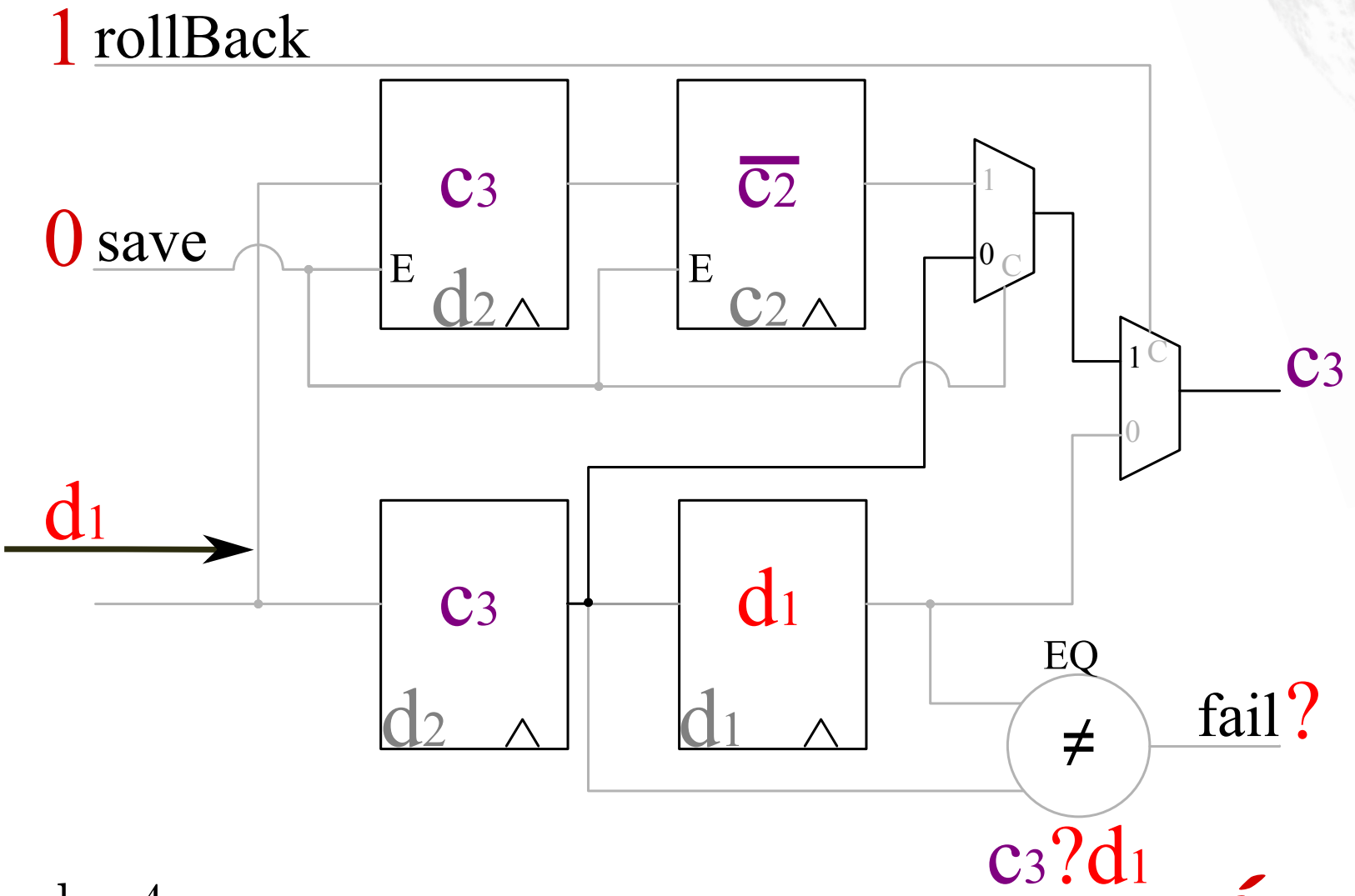
Cycle: 2

Memory Block: Rollback



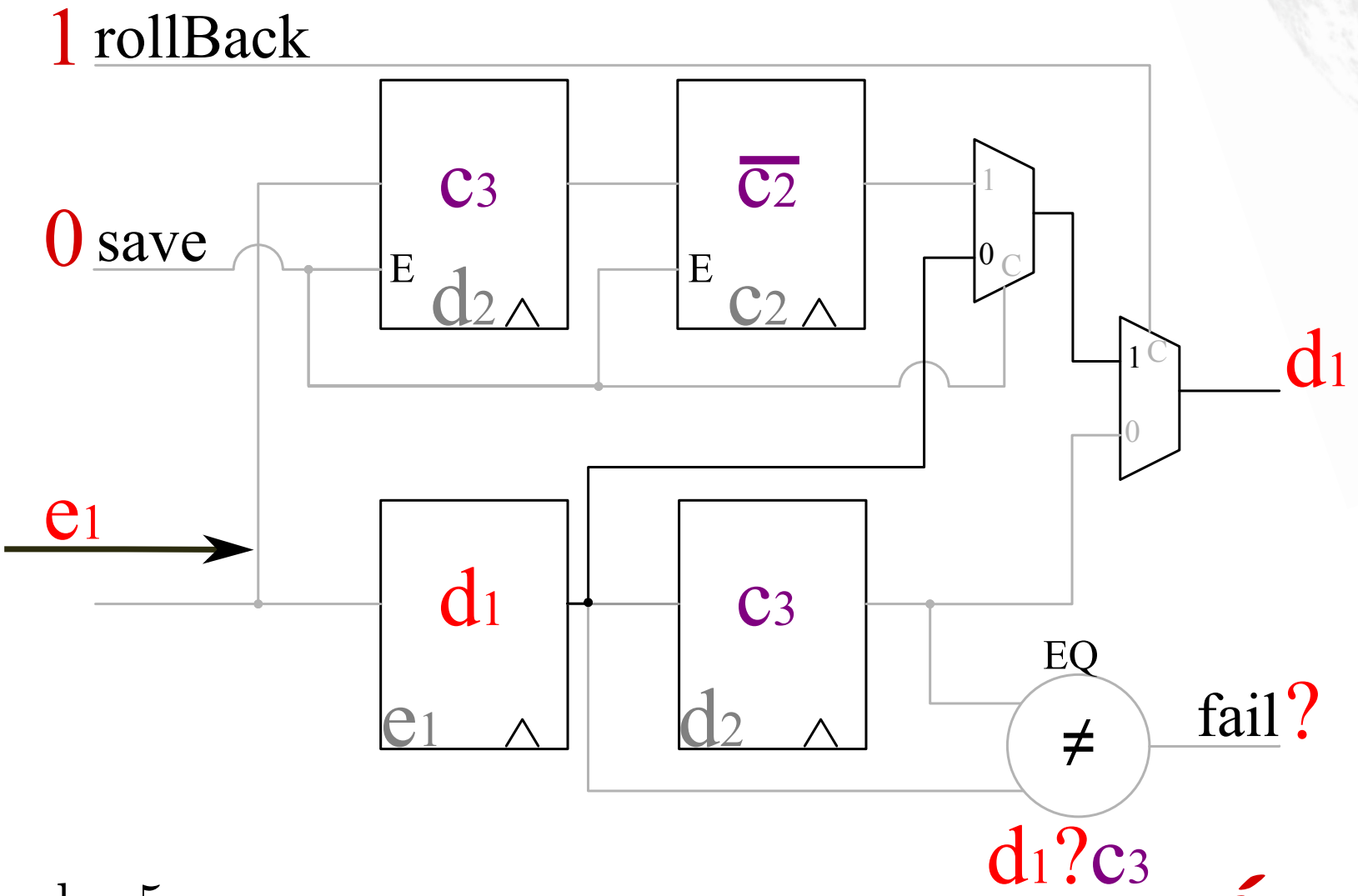
Cycle: 3

Memory Block: Speed-up



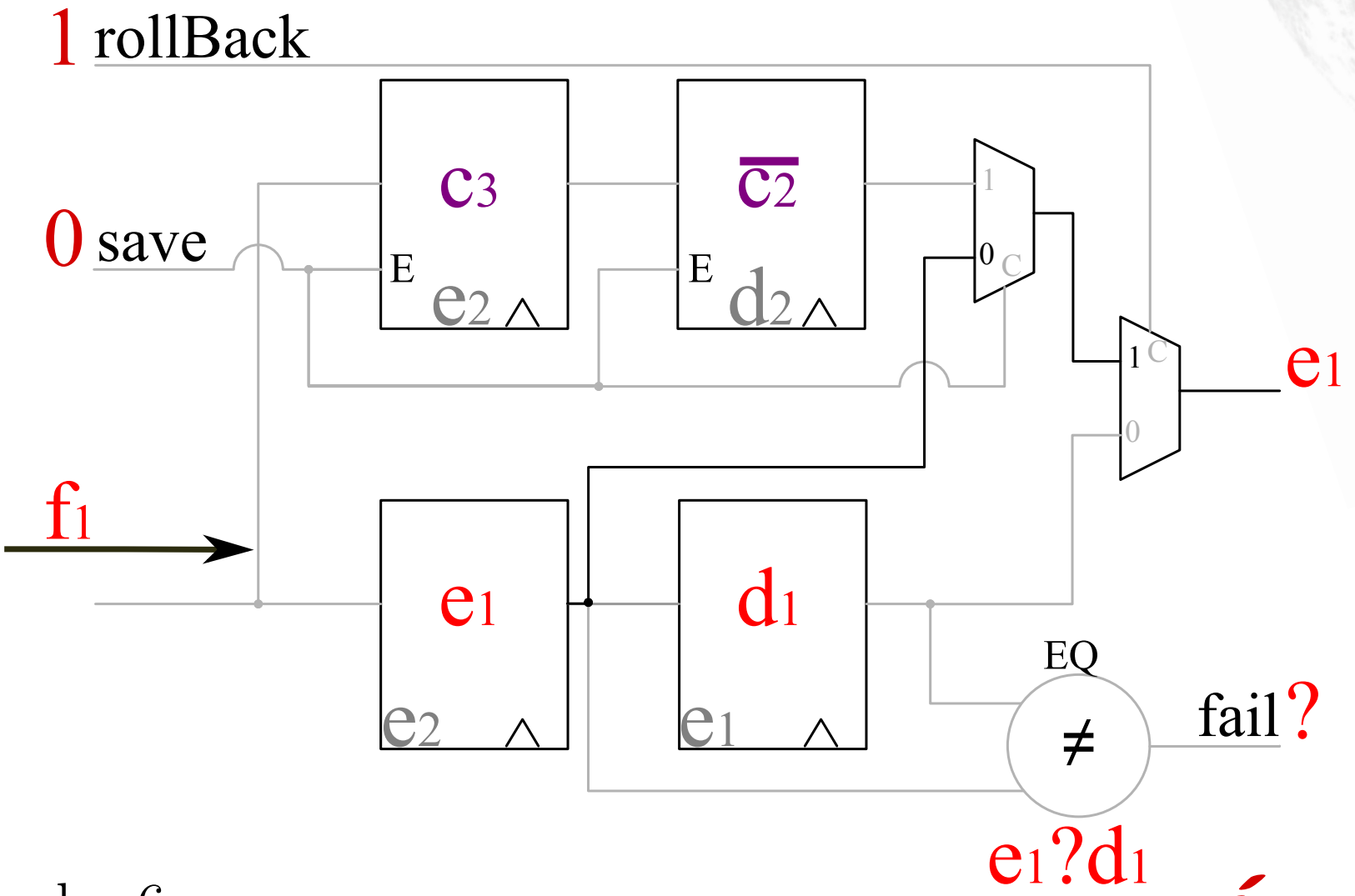
Cycle: 4

Memory Block: Output synchronization



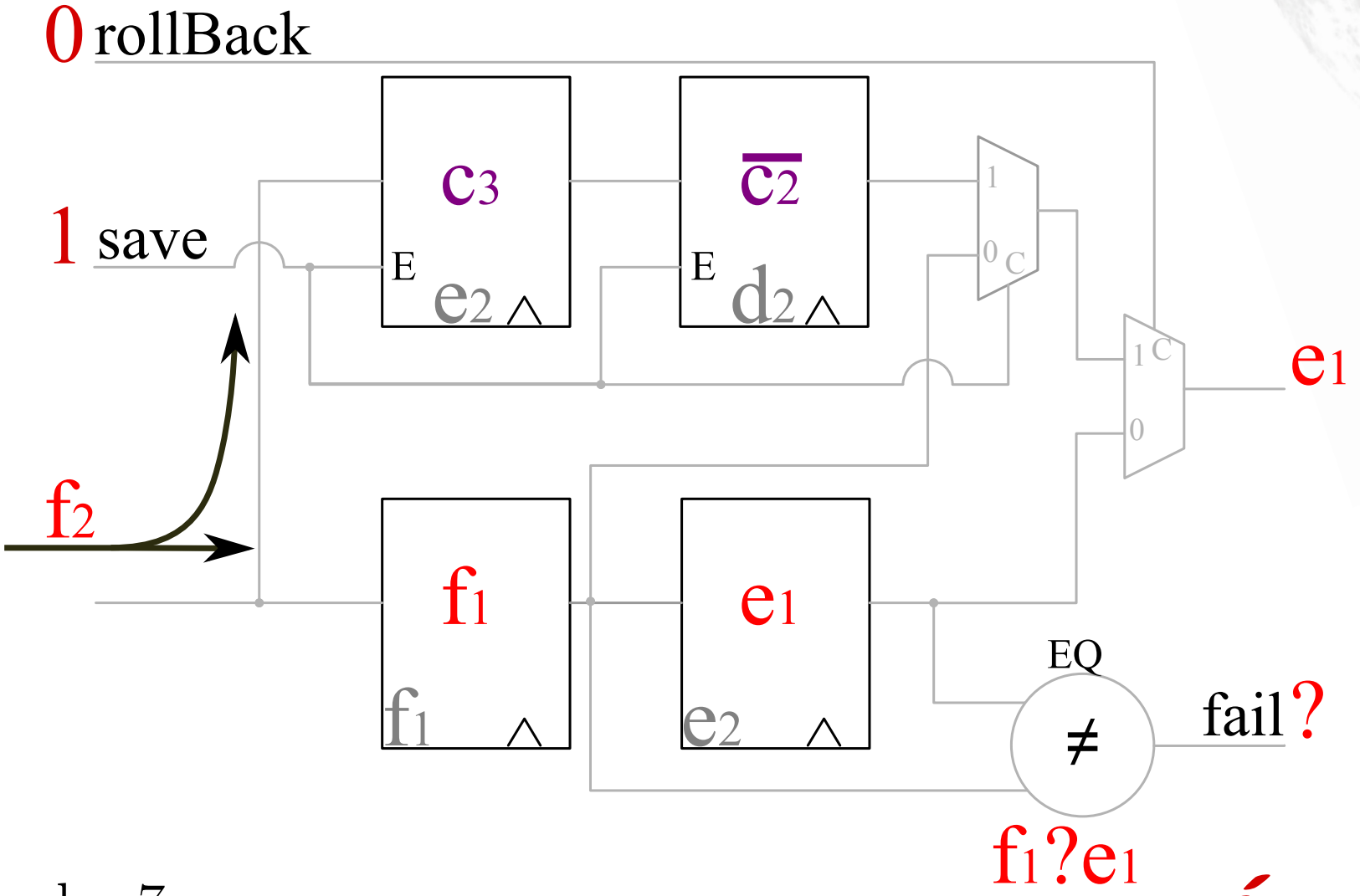
Cycle: 5

Memory Block: Output synchronization



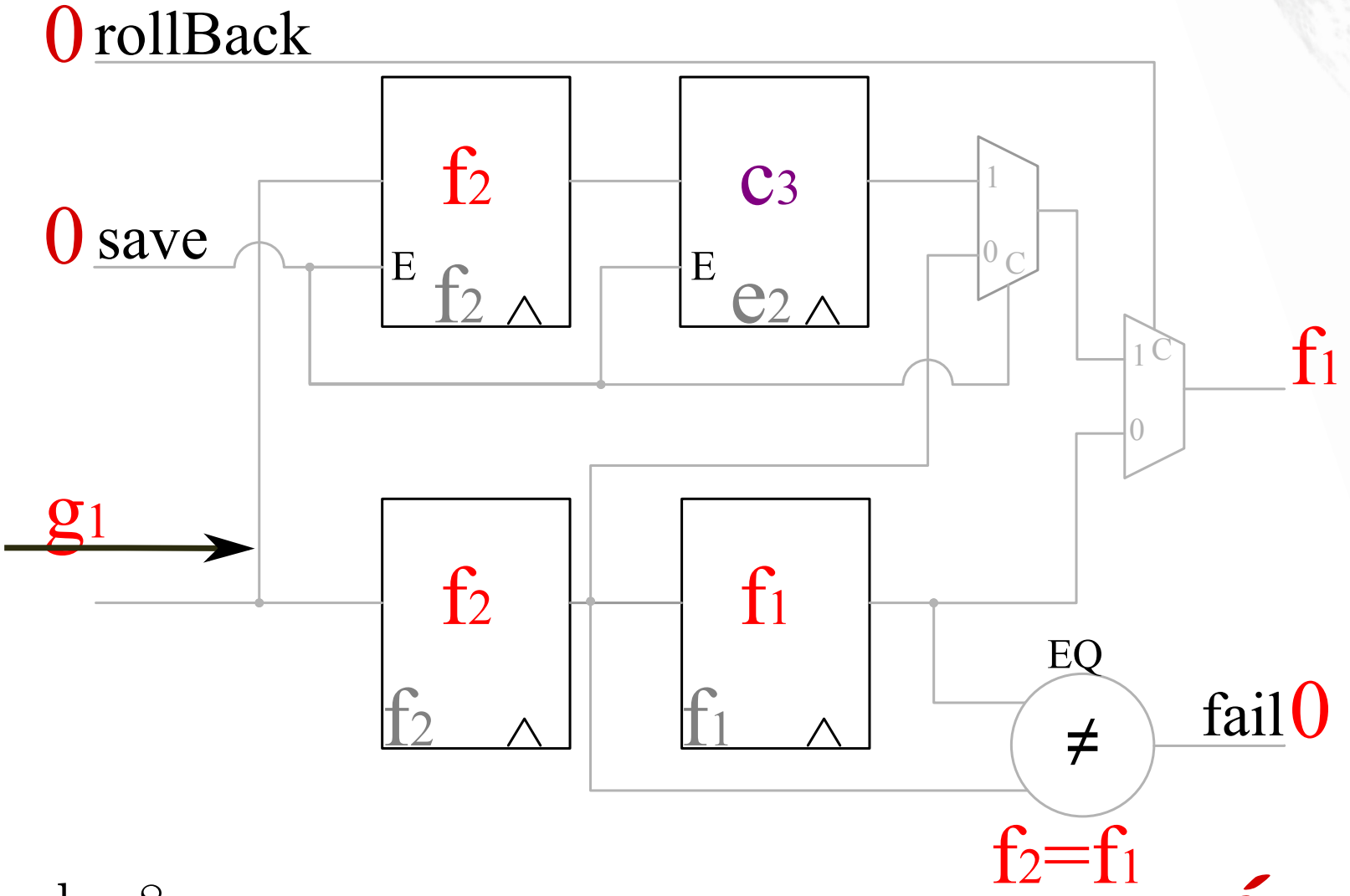
Cycle: 6

Memory Block: Speed-up OFF



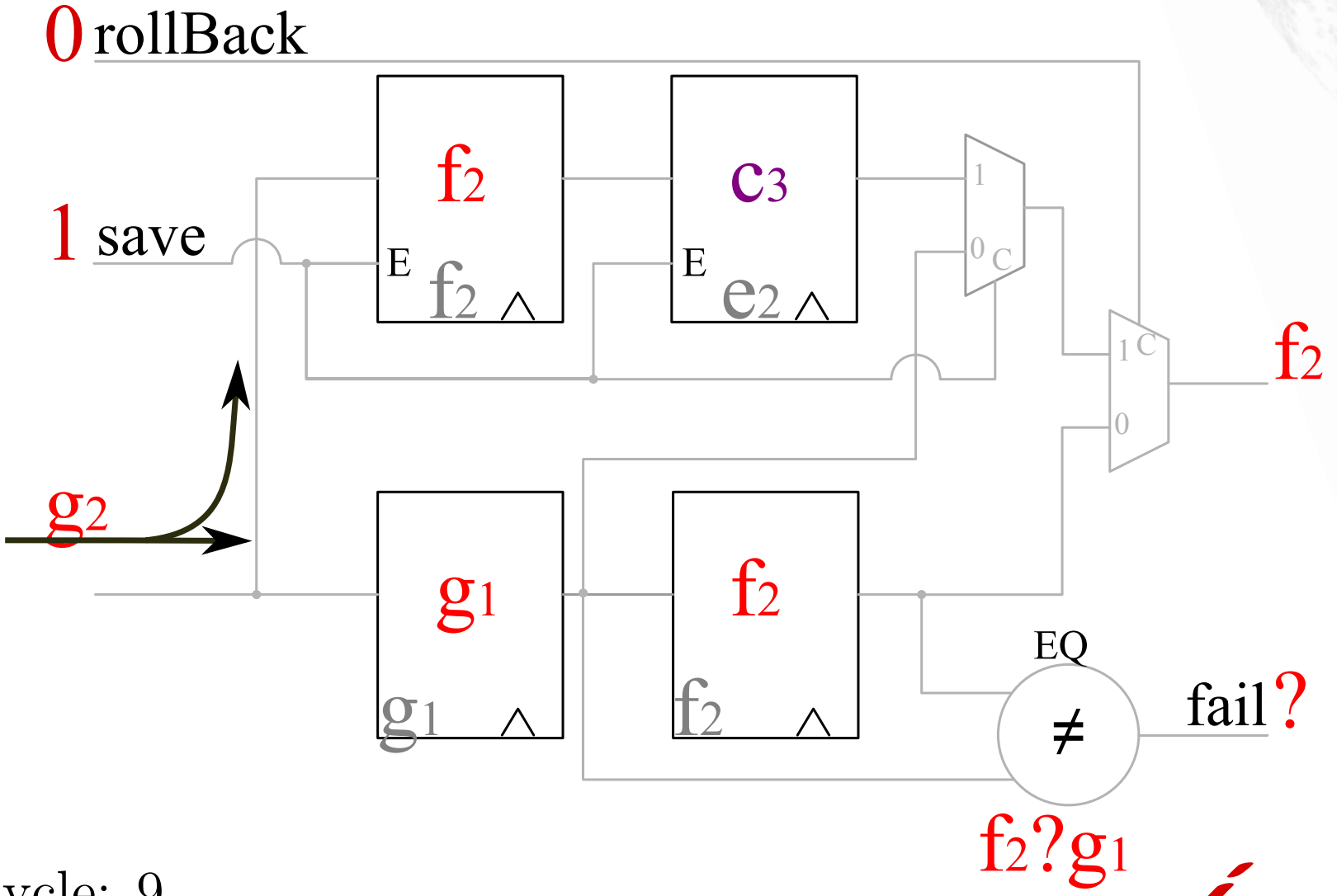
Cycle: 7

Memory Block: Working Cycle- no FT



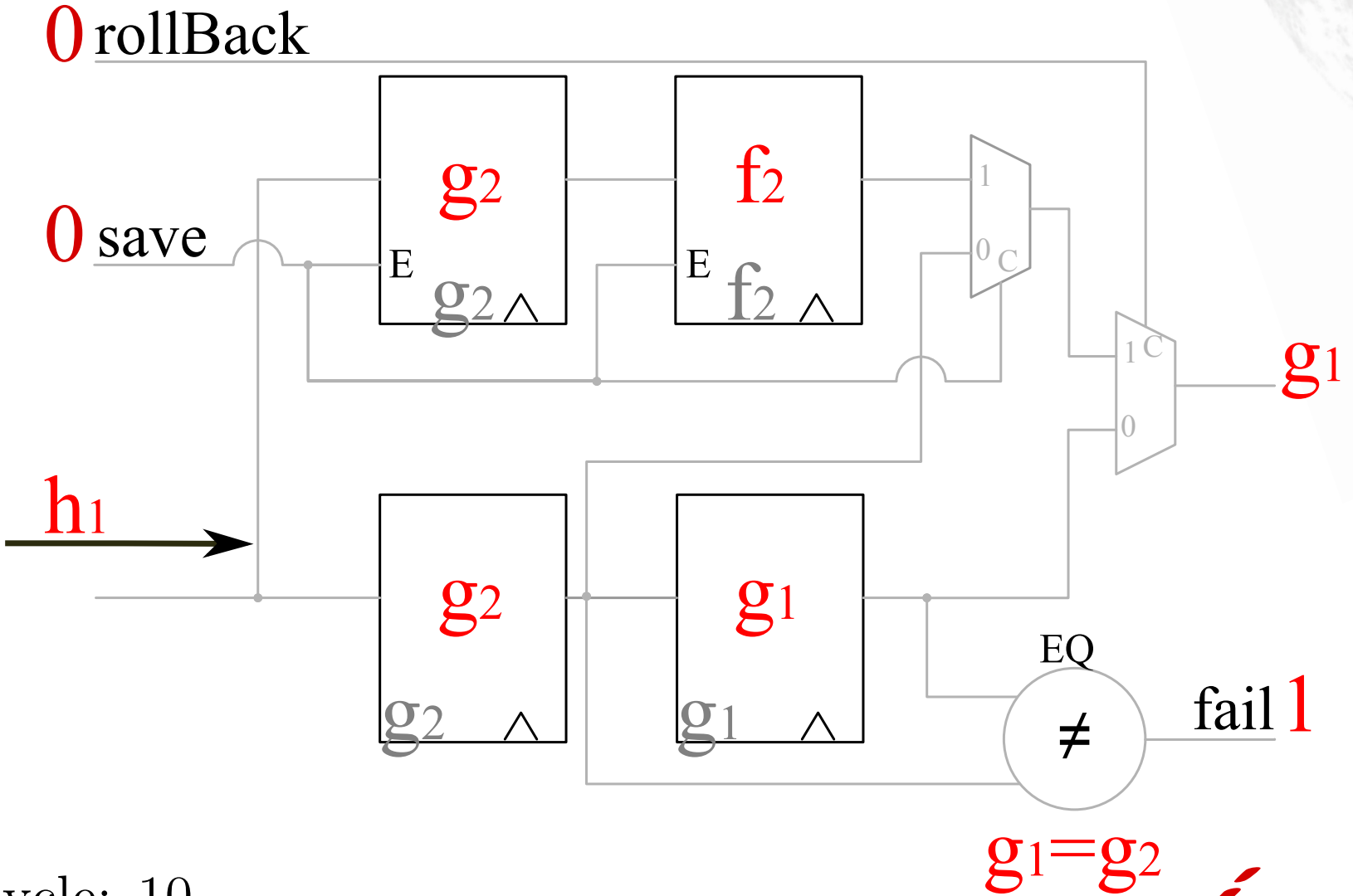
Cycle: 8

Memory Block: Working Cycle- no FT



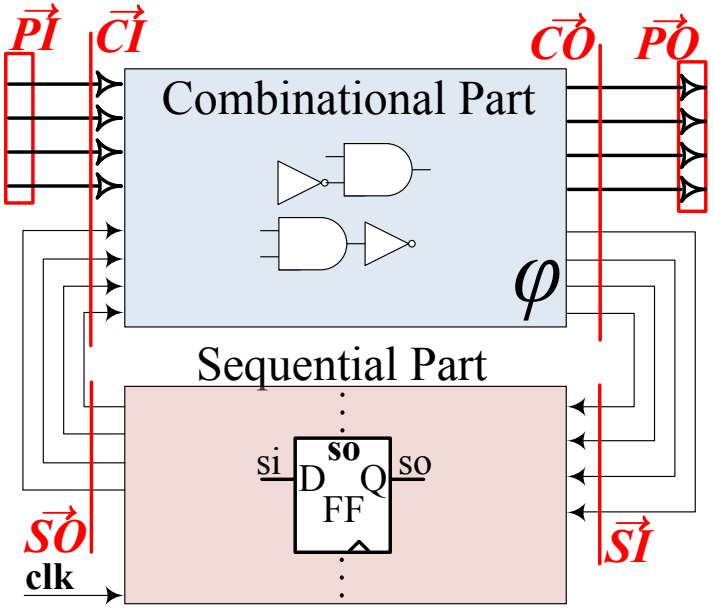
Cycle: 9

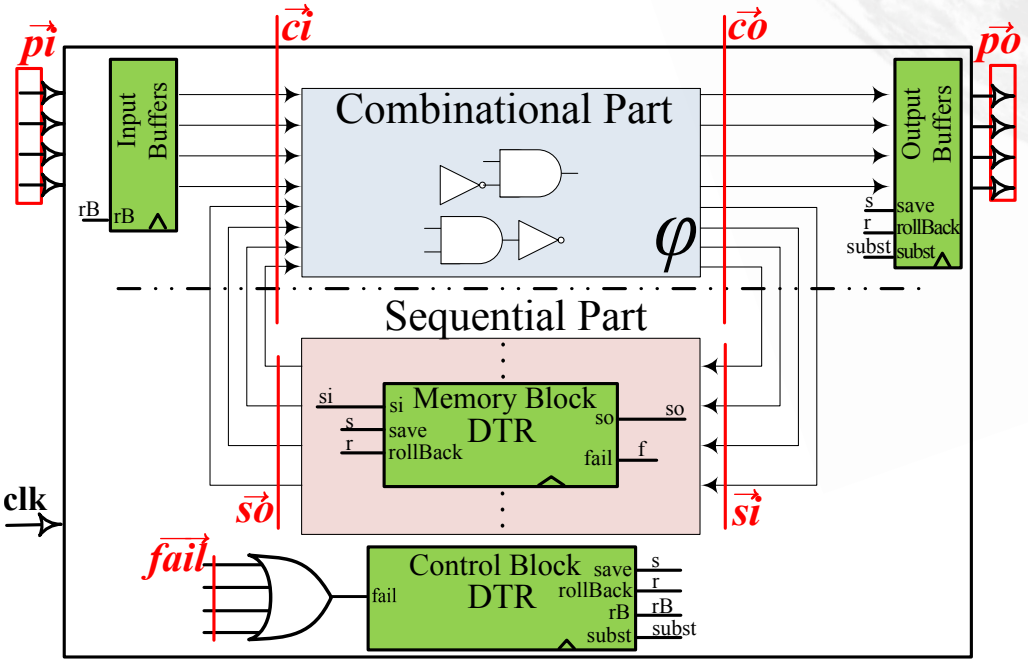
Memory Block: Working Cycle



Cycle: 10

Transformation DTR

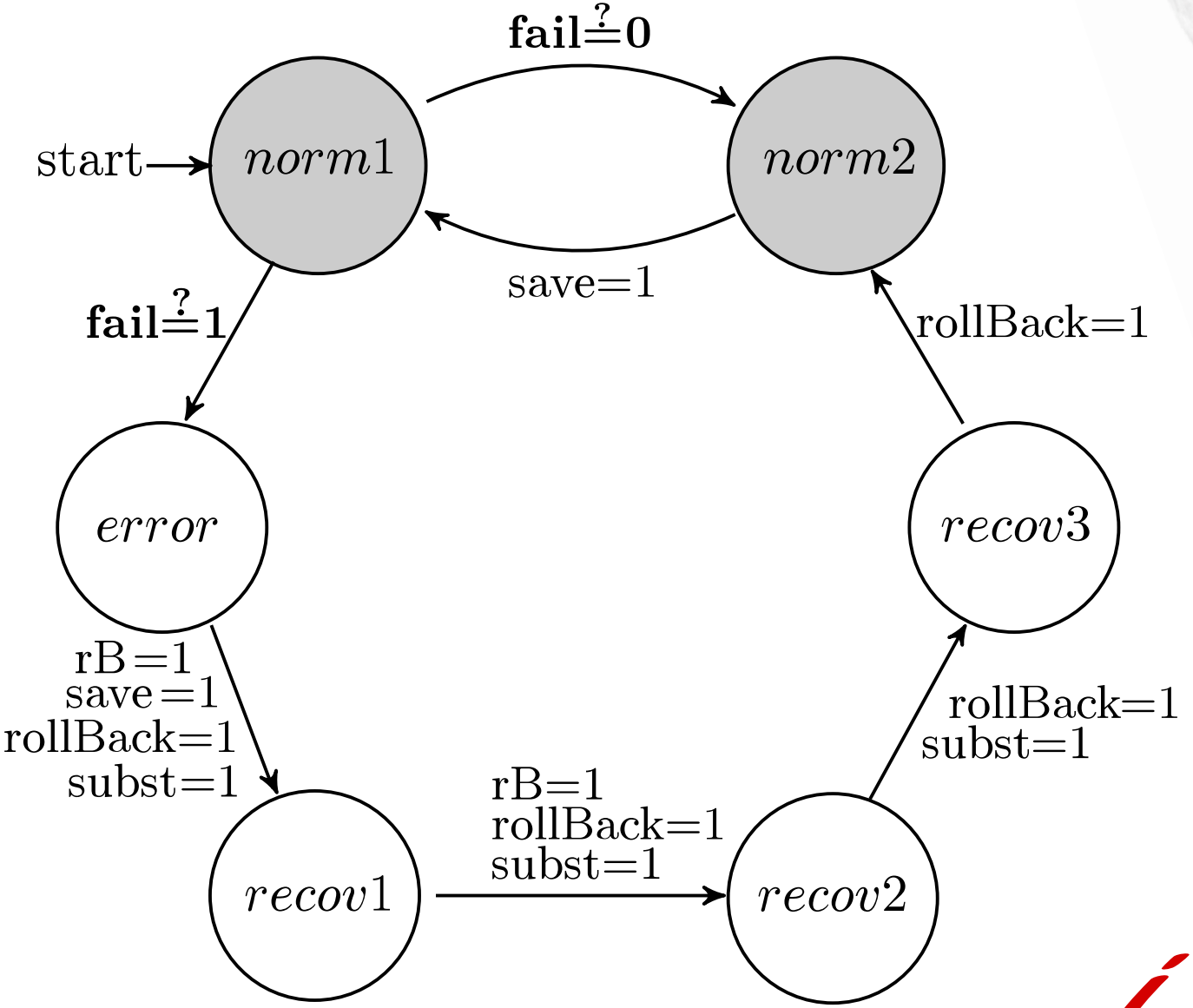




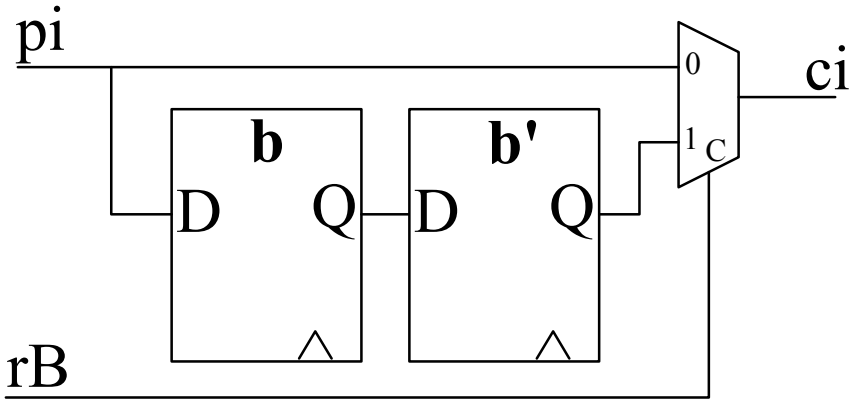
Input stream upsampling **x2** (original throughput/2)

1) Memory Cell ← Memory Block
2) Control Block Introduction
3) Input/Output Buffers Insertion

Control Block

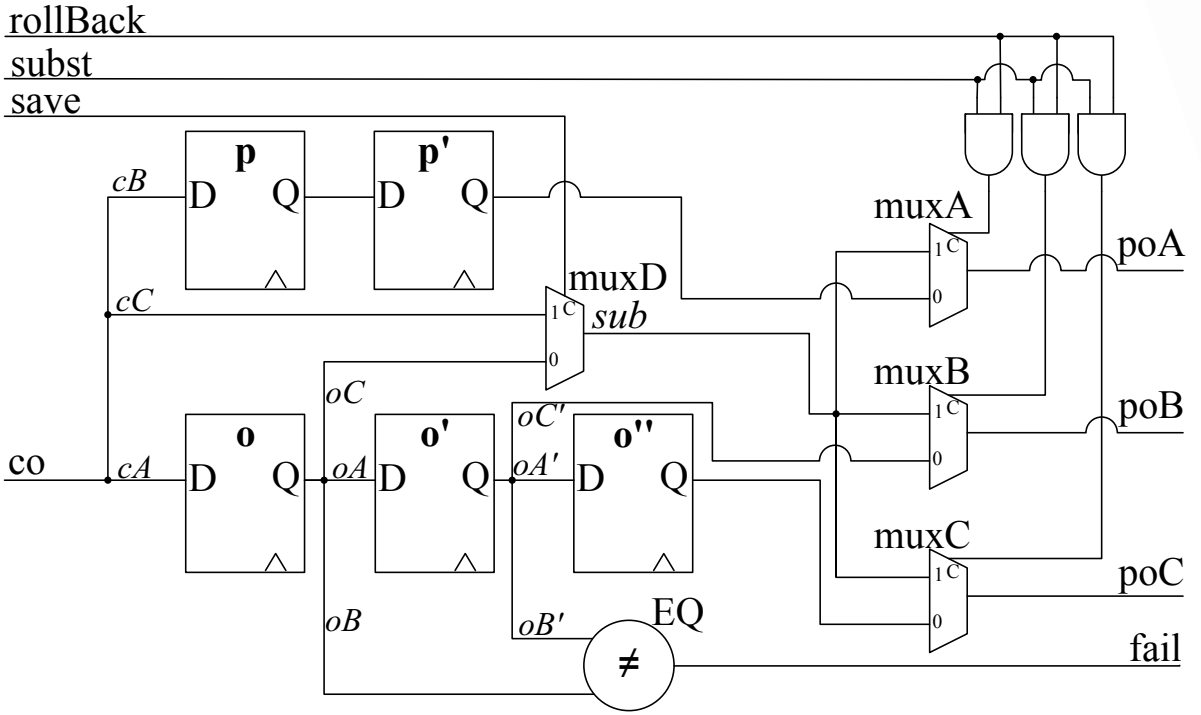


Input Buffer



- added to each primary input
- keeps last 2 bits **For** recovery recalculation
- keeps last 2 bits **During** recovery for transparency

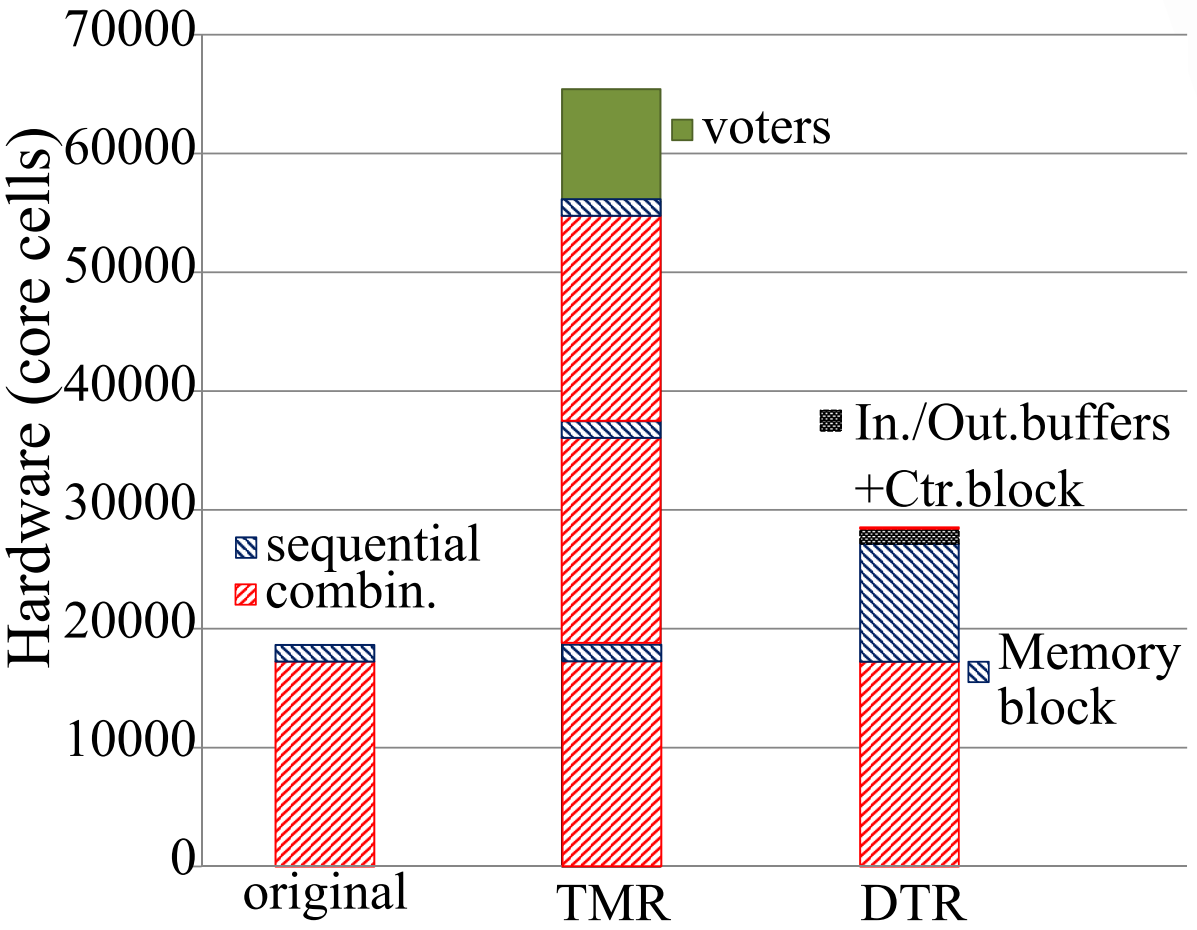
Output Buffer



- added to each primary output
- adds a delay to emit correct values during recovery
- fault-tolerant by itself to SET

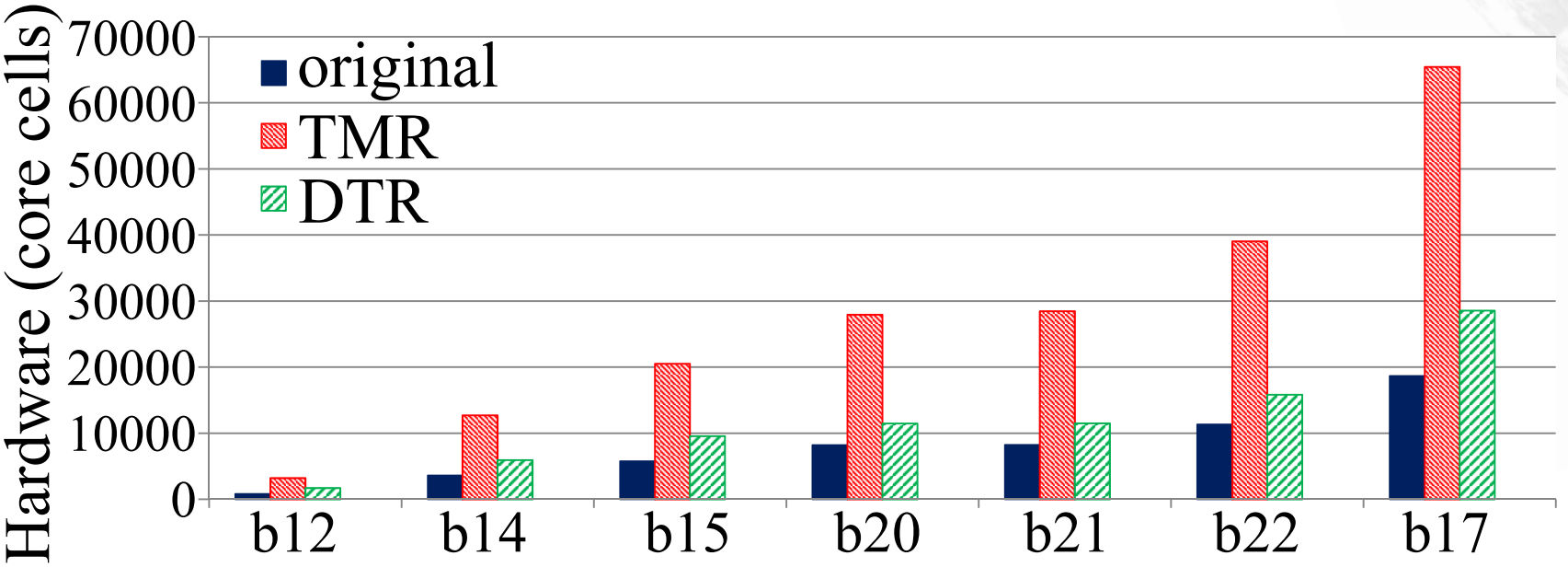
Experimental Results

Synthesis Results: Profiling *b*17, ITC'99



area - in terms of Core Cells for Actel ProASIC3 FPGA
Synthesised in Synopsys Synplify Pro'2009

Synthesis Results: Big Circuits, ITC'99

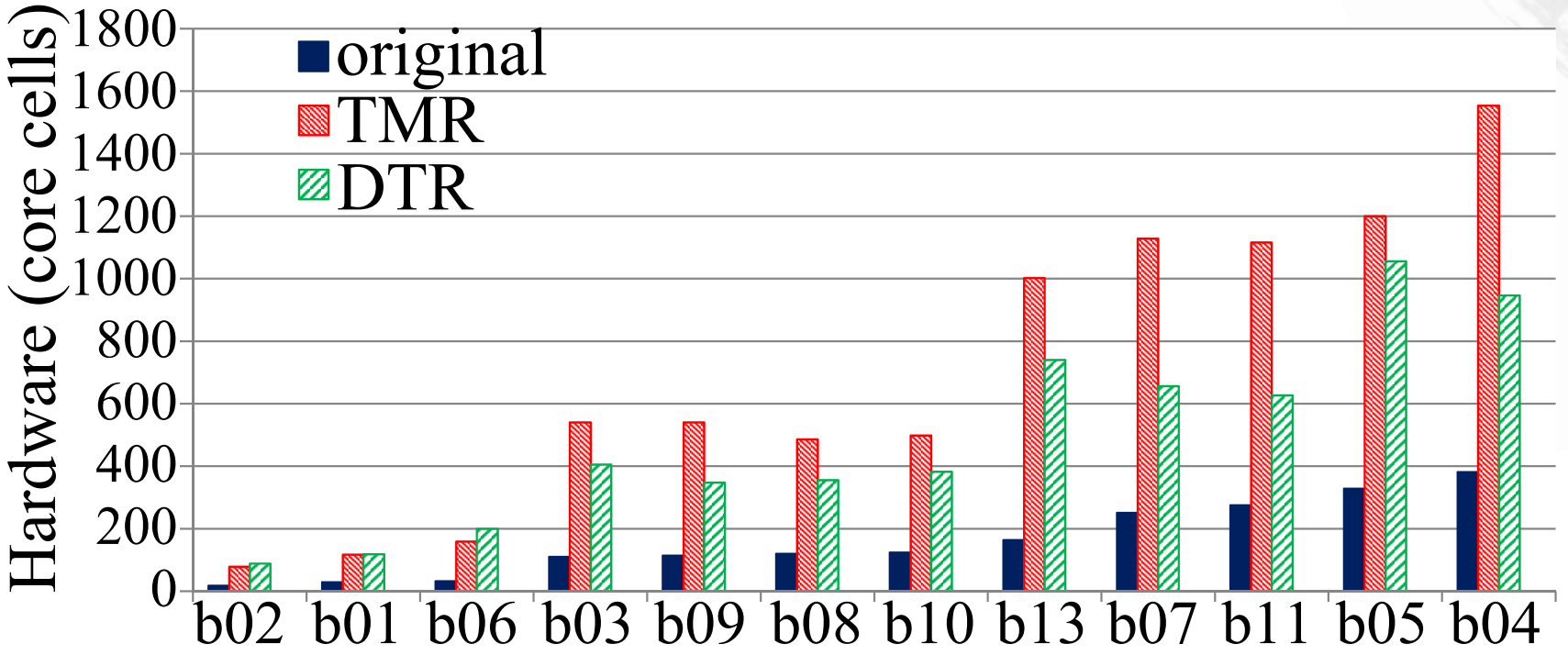


DTR = original x1.39-2.0

TMR = original x3.4- 3.9

TMR = DTR x1.9- 2.5

Synthesis Results: Small <100 mem. cells



The advantage is smaller

Summary

Summary

- Automatic circuits transformation for time-redundancy:
  - can be formally proved:

$$\forall C : circuit, \forall i : inputs, \forall o : outputs, \quad C\ i \longrightarrow o \quad \Rightarrow$$
$$\Rightarrow \quad \mathscr{T}(C)\ \bar{i} \overset{\text{faulty}}{\longrightarrow} \bar{o}$$

  - applicable to existing synthesis tools
    (Mentor Graphics, Xilinx, Synopsys)
  - **1.9-2.5** smaller than TMR
    (with double throughput loss)
  - stream processing, technologically independent, ...

Thank you for your attention!

Your Questions/Feedbacks are
WELCOMED

dmitry.burlyaev @ inria.fr