EEDG/CE 6303: Testing and Testable Design

Mehrdad Nourani

Dept. of ECE
Univ. of Texas at Dallas

Session 07

Path Delay Testing

Definitions & Concept

Key Issues

- Delay test definition
- Circuit delays and event propagation
- *Path-delay tests*
  - *Non-robust test*
  - *Robust test*
  - Five-valued logic and test generation
- *Path-delay fault* (PDF) and other fault models
- Test application methods
  - Combinational, enhanced-scan and normal-scan
  - Variable-clock and rated-clock methods
- Advanced Techniques

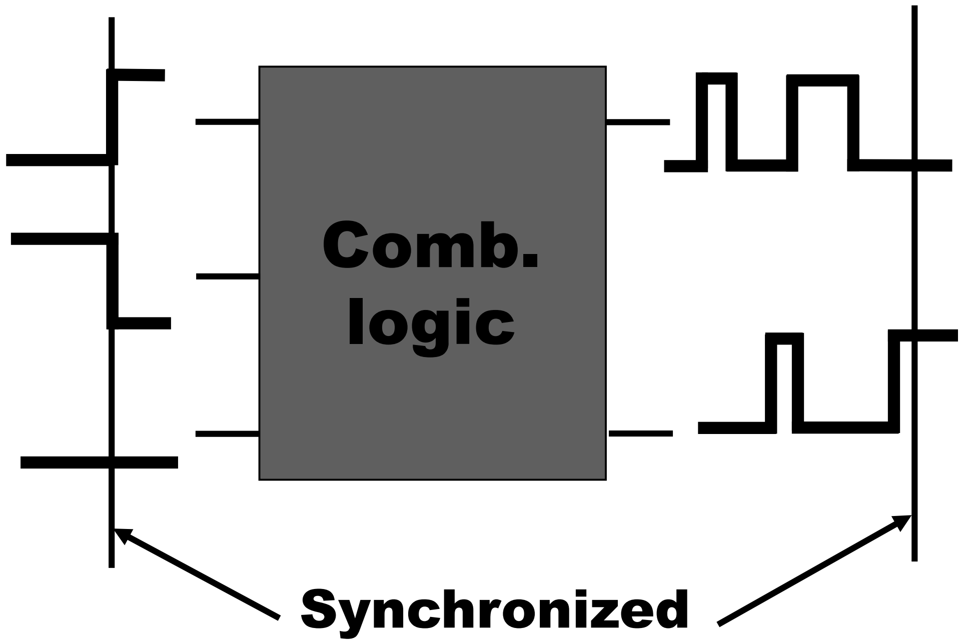
Delay Test Definition

- A circuit that passes delay test must produce correct outputs when inputs are applied and outputs observed **with specified timing**.
- For a combinational or synchronous sequential circuit, delay test verifies the limits of delay in combinational logic.
- Delay test problem for asynchronous circuits is complex and not well understood.

Digital Circuit Timing

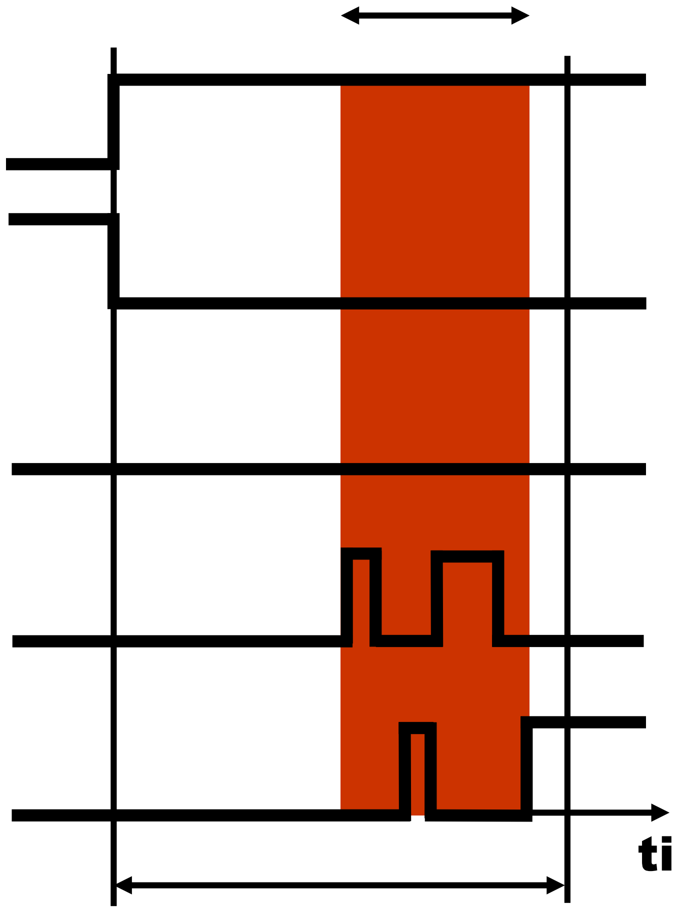
Input Signal changes

Output Observation instant



Synchronized With clock

Transient region

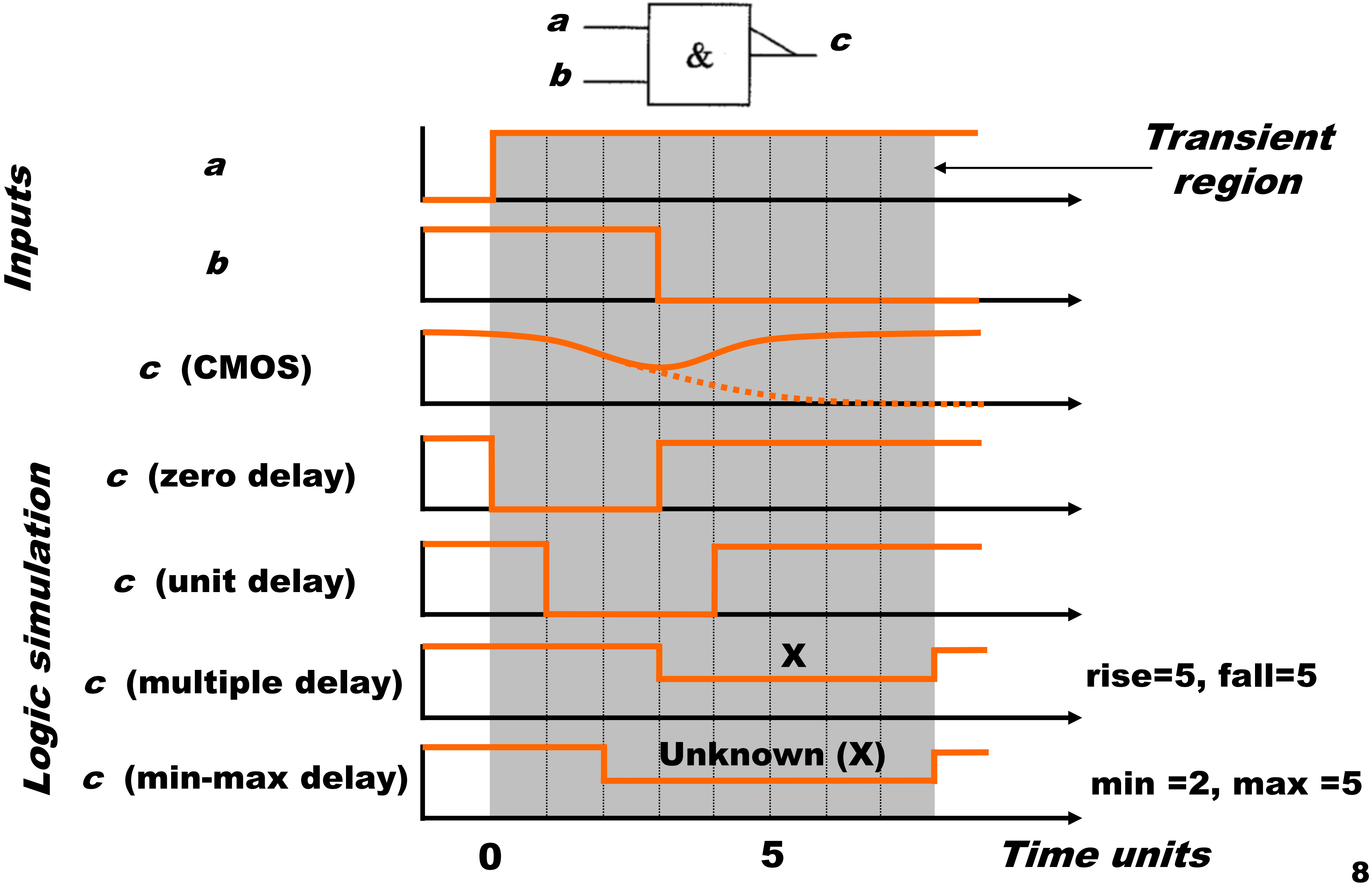


Clock period

Circuit Delays

- Switching or inertial delay is the interval between input change and output change of a gate:
  - Depends on input capacitance, device (transistor) characteristics and output capacitance of gate.
  - Also depends on input rise or fall times and states of other inputs (second-order effects).
  - Approximation: fixed rise and fall delays (or min-max delay range, or single fixed delay) for gate output.
- Propagation or interconnect delay is the time a transition takes to travel between gates:
  - Depends on transmission line effects (distributed *R*, *L*, *C* parameters, length and loading) of routing paths.
  - Approximation: modeled as lumped delays for gate inputs.
- See next slide for some timing models.

Options for Inertial Delay (e.g. NAND)



a

b

&

c

Inputs

a

b

Logic simulation

c (CMOS)

c (zero delay)

c (unit delay)

c (multiple delay)

c (min-max delay)

Transient region

X

Unknown (X)

rise=5, fall=5

min =2, max =5

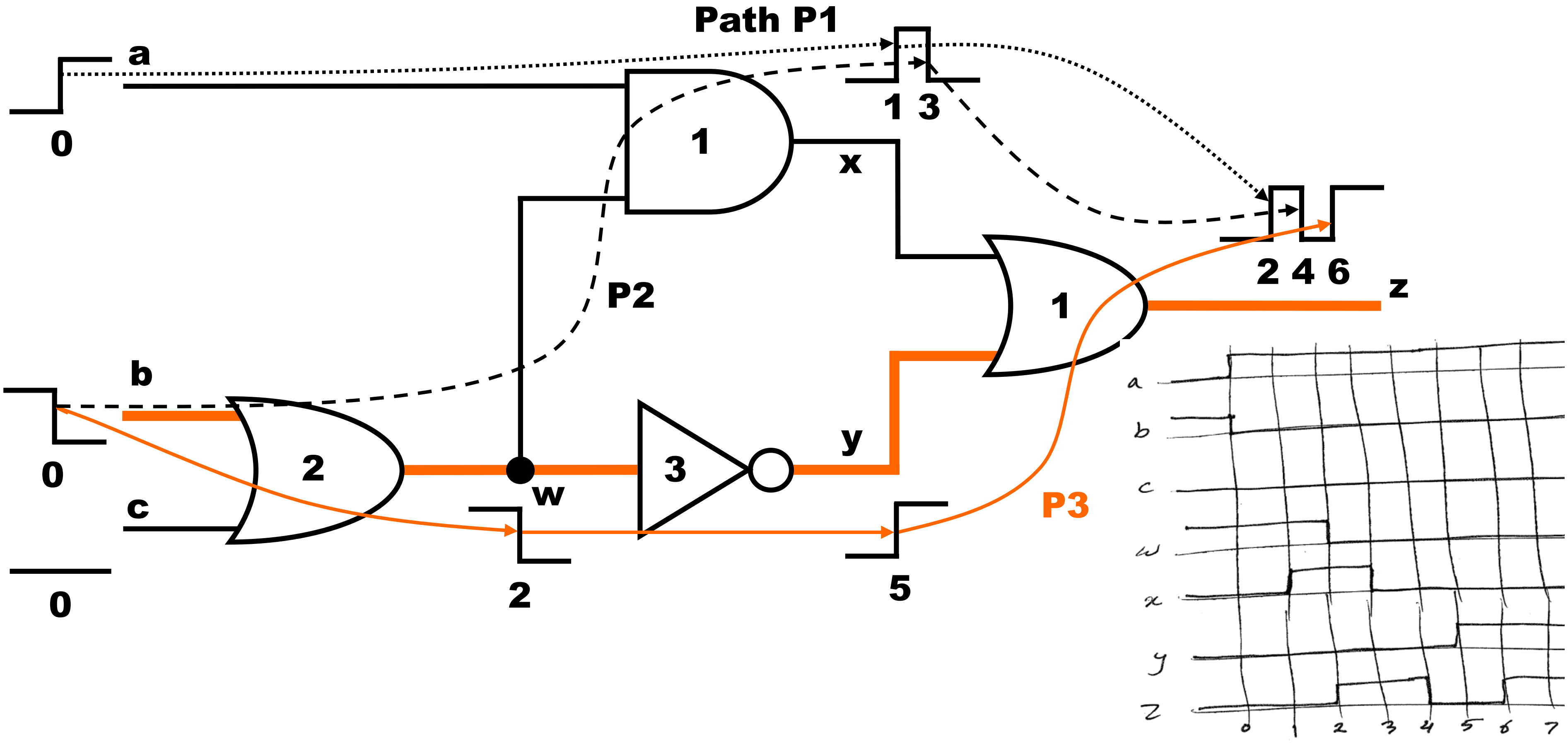
0

5

Time units

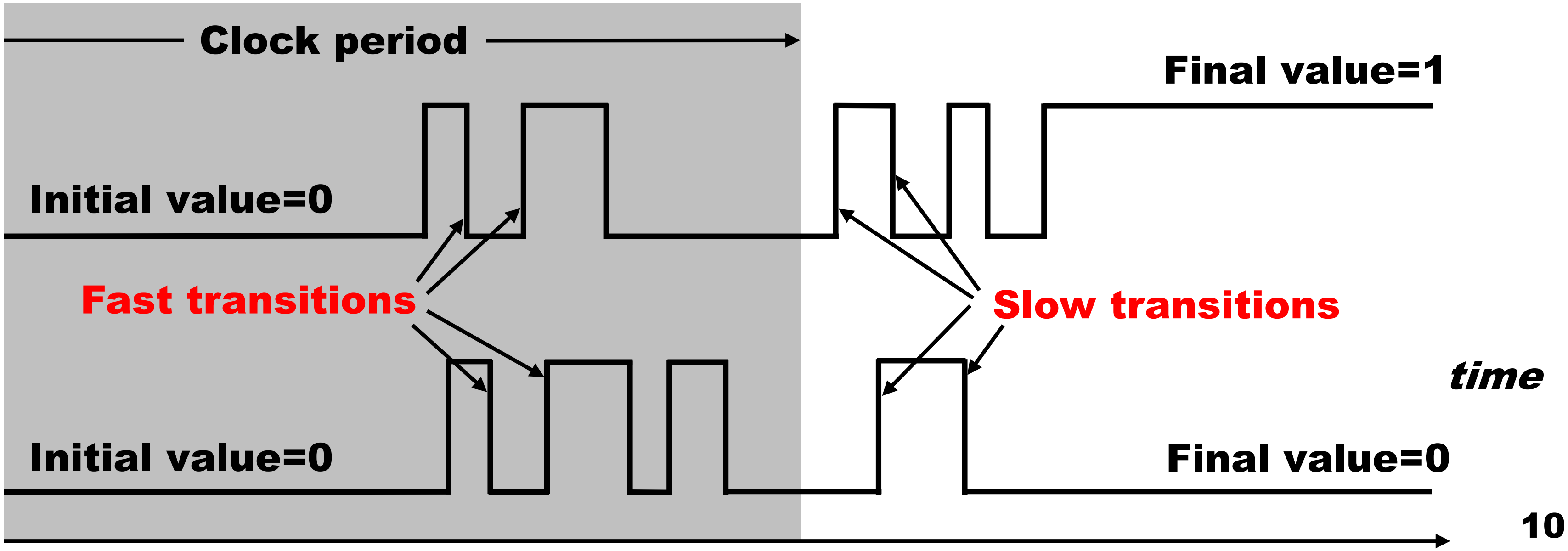
Event Propagation Delays

- Single lumped inertial delay modeled for each gate
- PI transitions assumed to occur without time skew



Circuit Outputs

- Each path can potentially produce one signal transition at the output.
- The location of an output transition in time is determined by the delay of the path.
  - Fast transitions: transitions with delay <= T_{ck}
  - Slow transitions: transitions with delay > T_{ck}
- If delay of a path increases/decreases, the corresponding output transitions move to right/left.



Key Assumption

- We usually assume the gate propagation delay is fixed and independent of input logic values. Therefore, it is assumed that **if the circuit passes the test for a given fault, then that fault will not cause any incorrect circuit operation for any other sequence of input patterns.**
    - This may not be strictly what happens in a circuit but we need this assumption to make path-delay testing tractable.

Robust and Non-Robust Test

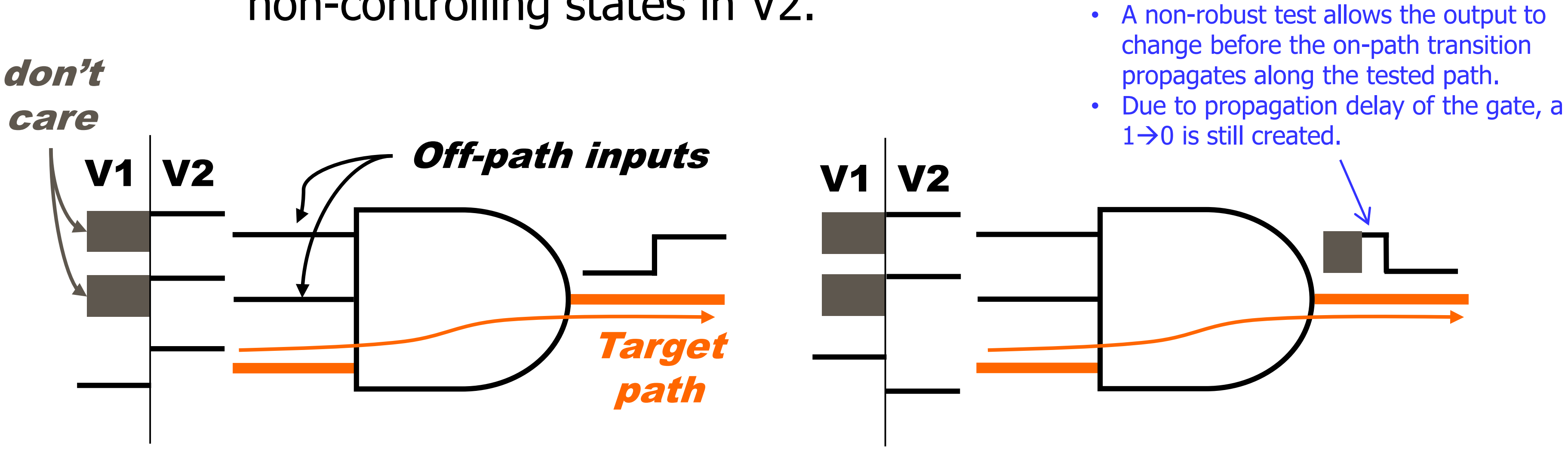
Robust vs. Non-Robust Test

- A two-pattern test (V1,V2) is a **non-robust** test for a path-delay fault (PDF), if and only if it satisfies the following conditions:
  1. It launches the desired logic transition in PIs of the targeted path
  2. All off-path (side) inputs of the targeted path *settle* **on the non-controlling values** under V2.

- A test for delay fault is said to be **robust** if it can detect a PDF independent of the delays in the rest of the circuit. It must satisfy:
  1. The conditions for the non-robust test
  2. Whenever the logic transition at an on-path input is from a non-controlling to a controlling, each corresponding off-path input should maintain a *steady* **non-controlling value**.

Singly-Testable Paths (Non-Robust Test)

- The delay of a target path is tested if the test propagates a transition via path to a path destination.
- Delay test is a combinational vector-pair, (V1,V2), that:
  - Produces a transition at path input.
  - Produces static sensitization -- All off-path inputs assume non-controlling states in V2.



- A non-robust test allows the output to change before the on-path transition propagates along the tested path.
- Due to propagation delay of the gate, a 1→0 is still created.

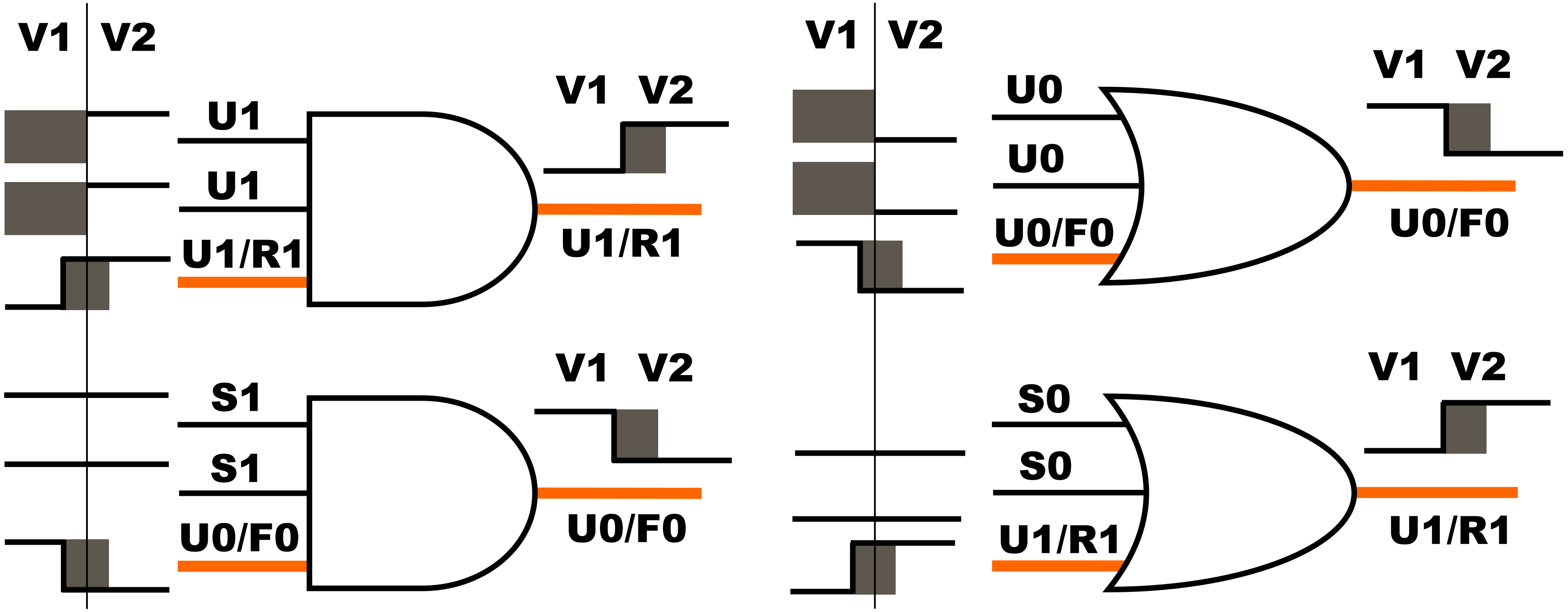
Static sensitization guarantees a test when the target path is the only faulty path. The test is, therefore, called *non-robust.* It is a test with minimal restriction. A path with no such test is a *false path.*

Robust Test

- A robust test guarantees the detection of a delay fault of the target path, irrespective of delay faults on other paths.
- A robust test is a combinational vector-pair, (V1,V2), that satisfies following conditions:
  - Produce *real events* (different steady-state values for V1 and V2) on all on-path signals.
  - All on-path signals must have *controlling events* arriving via the target path.
- A robust test is also a non-robust test.
- Concept of robust test is general – robust tests for other fault models can be defined.

Robust Test Conditions

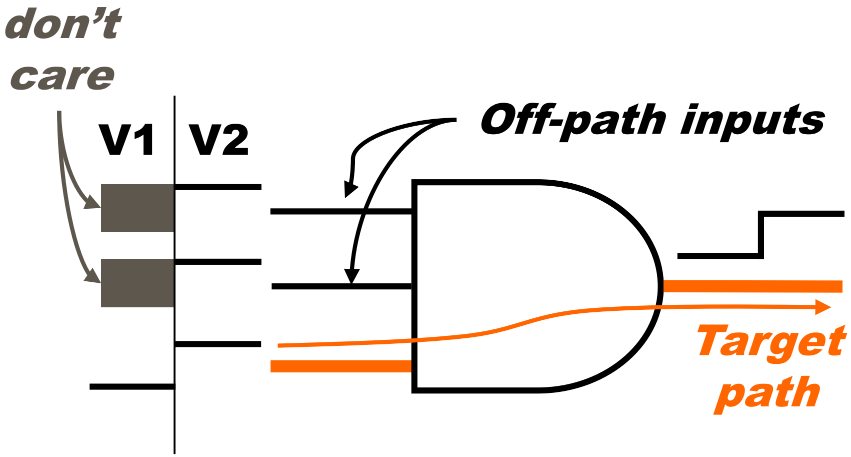
- Real events on target path.
- Controlling events via target path.



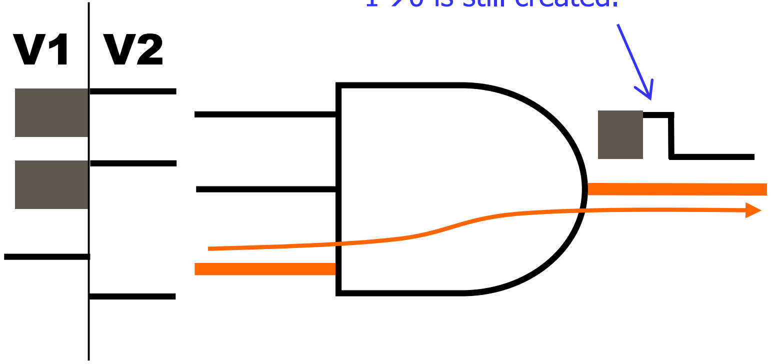
Robust vs. Non-Robust Test

- Another repetition of figures for an AND gate to show robust versus non-robust test better.

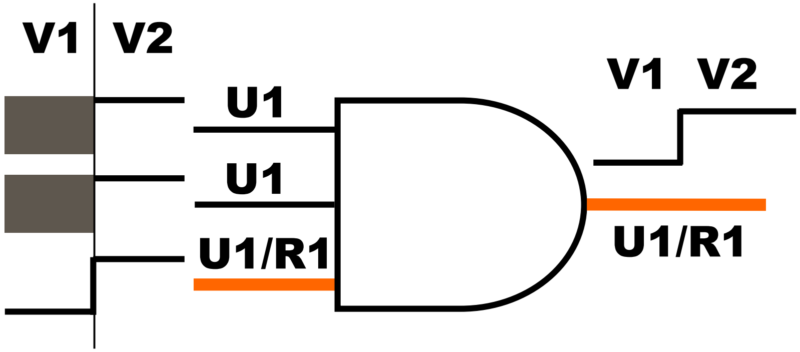
Non-Robust

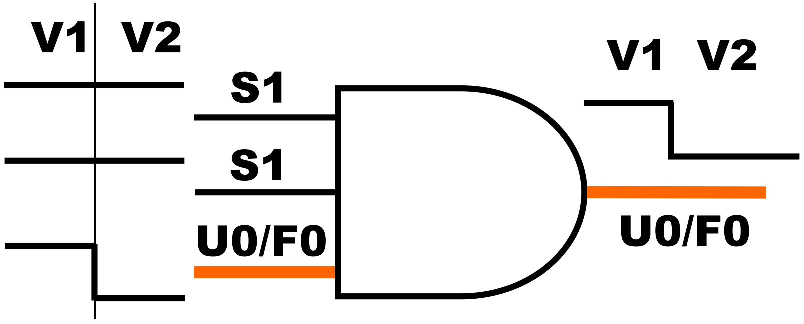


- A non-robust test allows the output to change before the on-path transition propagates along the tested path.
- Due to propagation delay of the gate, a 1→0 is still created.



Robust





A Five-Valued Algebra (Robust & Non-Robust)

- Signal States: S0, U0 (F0), S1, U1 (R1), XX.
- On-path signals: U0=F0 (1→0) and U1=R1 (0→1).
- Off-path signals: U0 and U1.

				Input 1		
AND		S0	U0	S1	U1	XX
Input 2	S0	S0	S0	S0	S0	S0
	U0	S0	U0	U0	U0	U0
	S1	S0	U0	S1	U1	XX
	U1	S0	U0	U1	U1	XX
	XX	S0	U0	XX	XX	XX

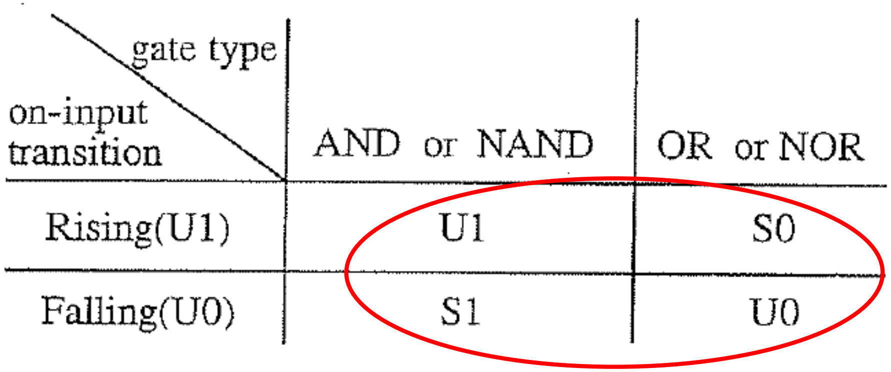
				Input 1		
OR		S0	U0	S1	U1	XX
Input 2	S0	S0	U0	S1	U1	XX
	U0	U0	U0	S1	U1	XX
	S1	S1	S1	S1	S1	S1
	U1	U1	U1	S1	U1	U1
	XX	XX	XX	S1	U1	XX

			Input		
NOT	S0	U0	S1	U1	XX
	S1	U1	S0	U0	XX

Ref.:
Lin-Reddy
***IEEETCAD*-87**

Sensitizing Input Values

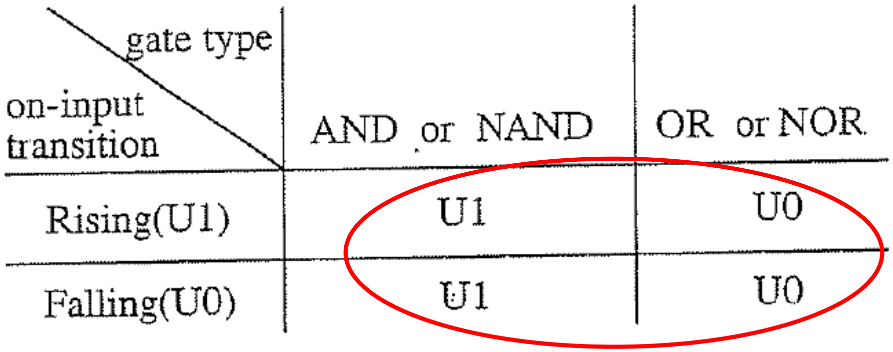
- Robust



gate type on-input transition	AND or NAND	OR or NOR
Rising(U1)	U1	S0
Falling(U0)	S1	U0

Off-path inputs needed

- Non-Robust



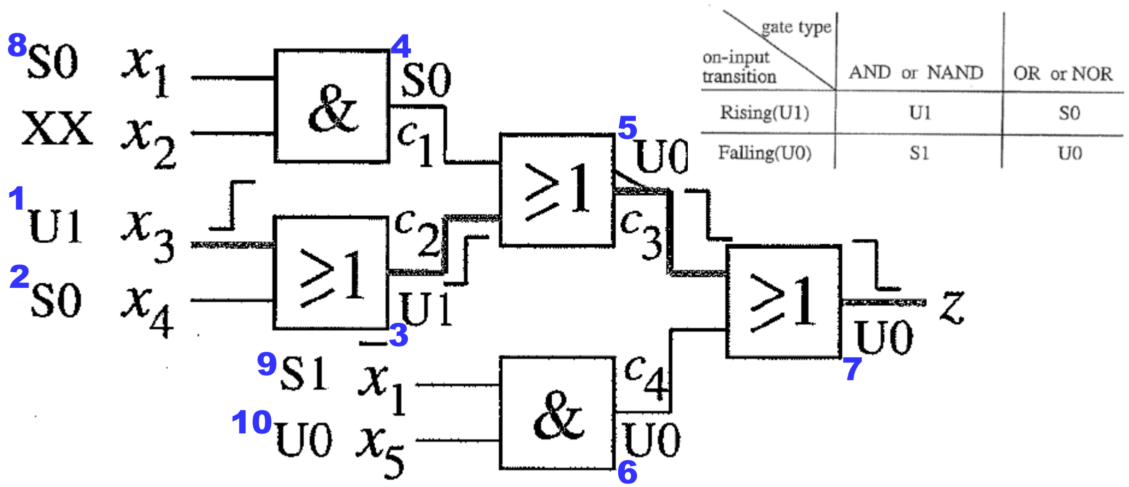
gate type on-input transition	AND or NAND	OR or NOR
Rising(U1)	U1	U0
Falling(U0)	U1	U0

Off-path inputs needed

- In backward propagation
  - In robust: S0 and U0 (also S1 and U1) are considered a conflict.
  - In non-robust, S0 and U0 are not a conflict and we will not have any S0 or S1 at the end.

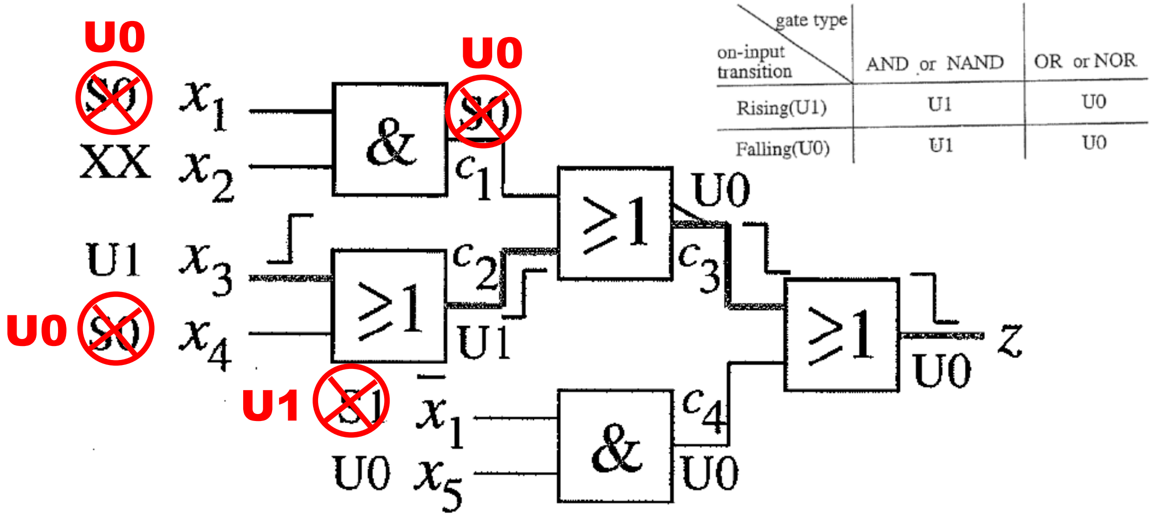
Example (PDF: Rising on $x_3 \rightarrow c_2 \rightarrow c_3 \rightarrow z$)

- Robust
  - Test Vector ($x_1x_2x_3x_4x_5$): (V_1,V_2)={(0x00x),(0x100)} OR (V_1,V_2)=(S0,XX,U1,S0,U0)



on-input transition \ gate type	AND or NAND	OR or NOR
Rising(U1)	U1	S0
Falling(U0)	S1	U0

- Non-Robust
  - S0 becomes U0 for off-path inputs, requiring non-controlling only for the second vector

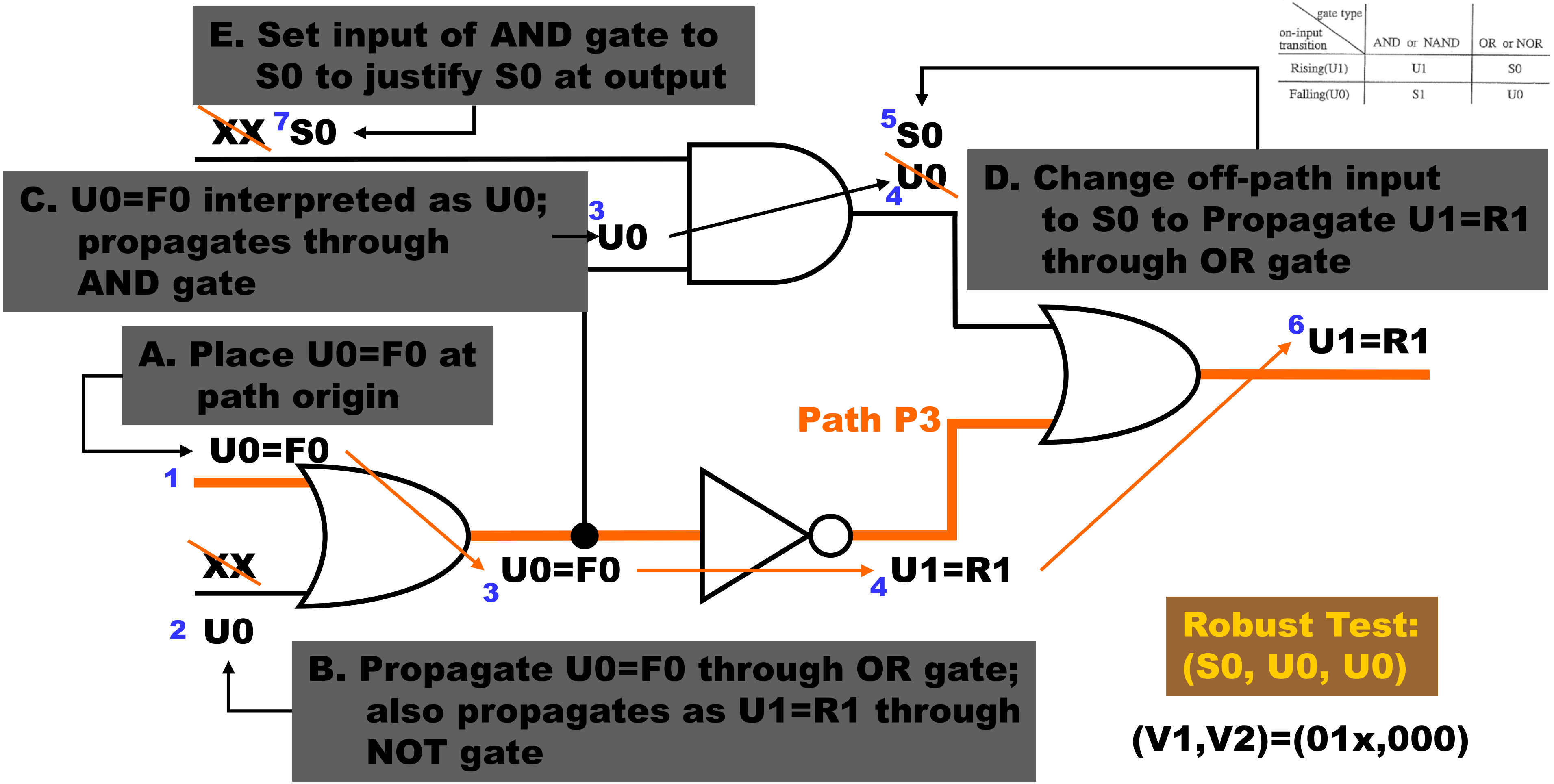


on-input transition \ gate type	AND or NAND	OR or NOR
Rising(U1)	U1	U0
Falling(U0)	U1	U0

  - Test Vector ($x_1x_2x_3x_4x_5$): (V_1,V_2)={(xx0xx),(0x100)} OR (V_1,V_2)=(U0,XX,U1,U0,U0)

Robust Test Generation (Path P3)

Test for ↓ P3 – falling transition through path P3: Steps A through E



gate type / on-input transition	AND or NAND	OR or NOR
Rising(U1)	U1	S0
Falling(U0)	S1	U0

E. Set input of AND gate to S0 to justify S0 at output

D. Change off-path input to S0 to Propagate U1=R1 through OR gate

C. U0=F0 interpreted as U0; propagates through AND gate

A. Place U0=F0 at path origin

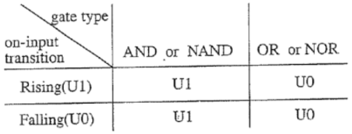
B. Propagate U0=F0 through OR gate; also propagates as U1=R1 through NOT gate

Robust Test: (S0, U0, U0)

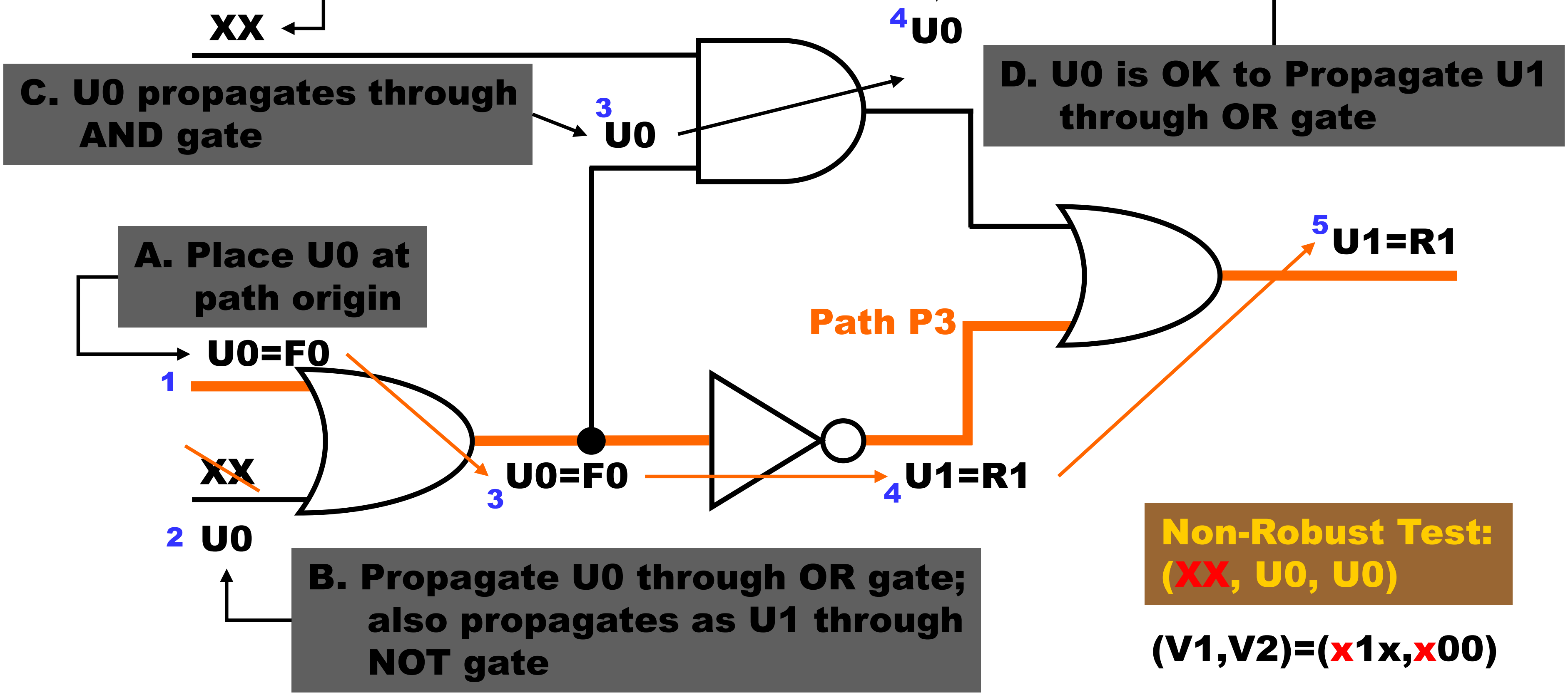
(V1,V2)=(01x,000)

Non-Robust Test Generation (Path P3)

Test for ↓ P3 – falling transition through path P3: Steps A through E



gate type / on-input transition	AND or NAND	OR or NOR
Rising(U1)	U1	U0
Falling(U0)	U1	U0



E. XX is OK

C. U0 propagates through AND gate

D. U0 is OK to Propagate U1 through OR gate

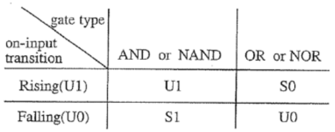
A. Place U0 at path origin

B. Propagate U0 through OR gate; also propagates as U1 through NOT gate

Non-Robust Test: (XX, U0, U0)

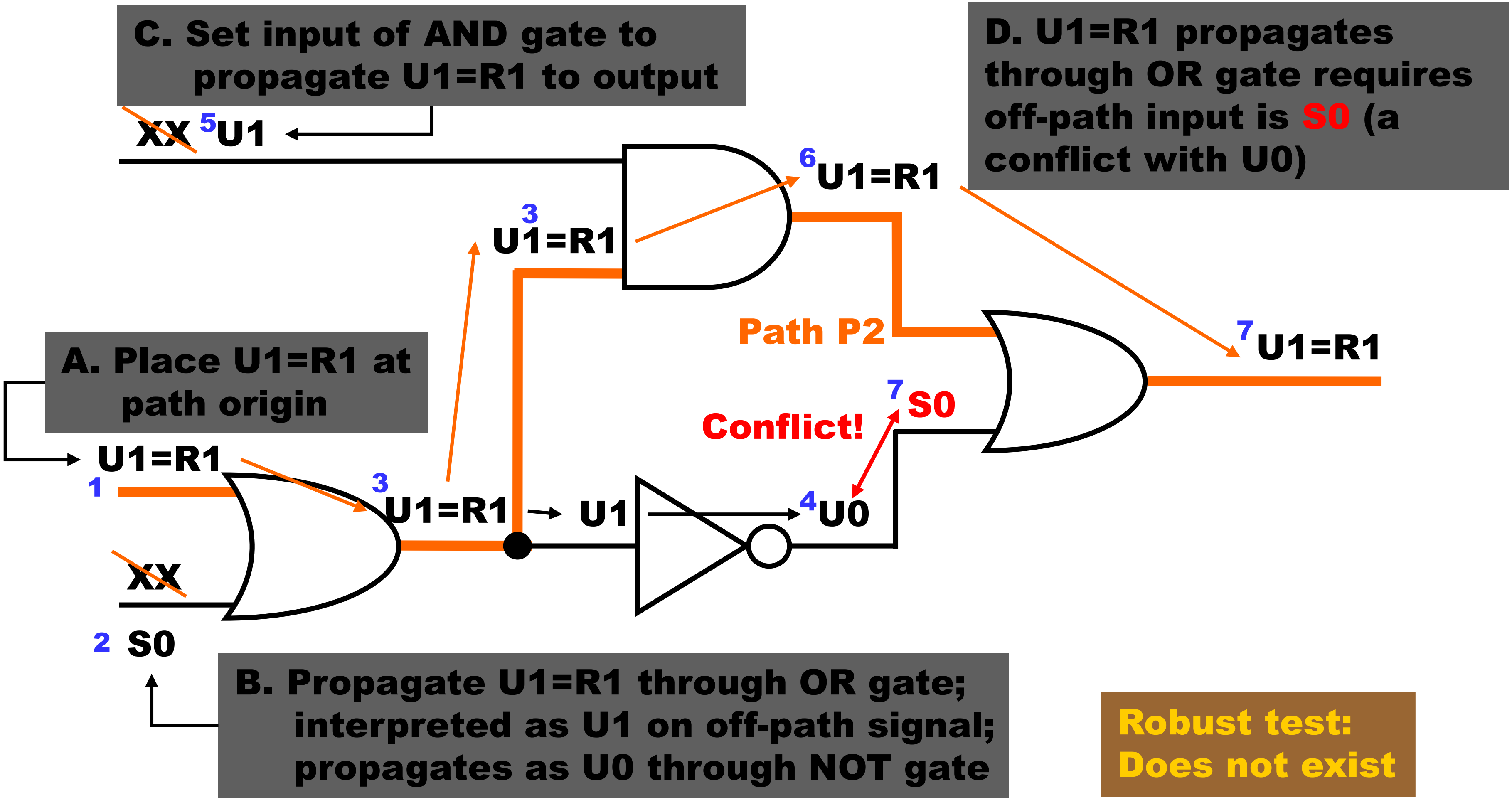
(V1,V2)=(x1x,x00)

Robust Test Generation Fails (Path P2)



gate type / on-input transition	AND or NAND	OR or NOR
Rising(U1)	U1	S0
Falling(U0)	S1	U0

Fault ↑P2 – rising transition through path P2.



C. Set input of AND gate to propagate U1=R1 to output

D. U1=R1 propagates through OR gate requires off-path input is S0 (a conflict with U0)

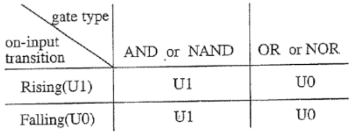
A. Place U1=R1 at path origin

B. Propagate U1=R1 through OR gate; interpreted as U1 on off-path signal; propagates as U0 through NOT gate

Robust test: Does not exist

Non-Robust Test Generation (Path P2)

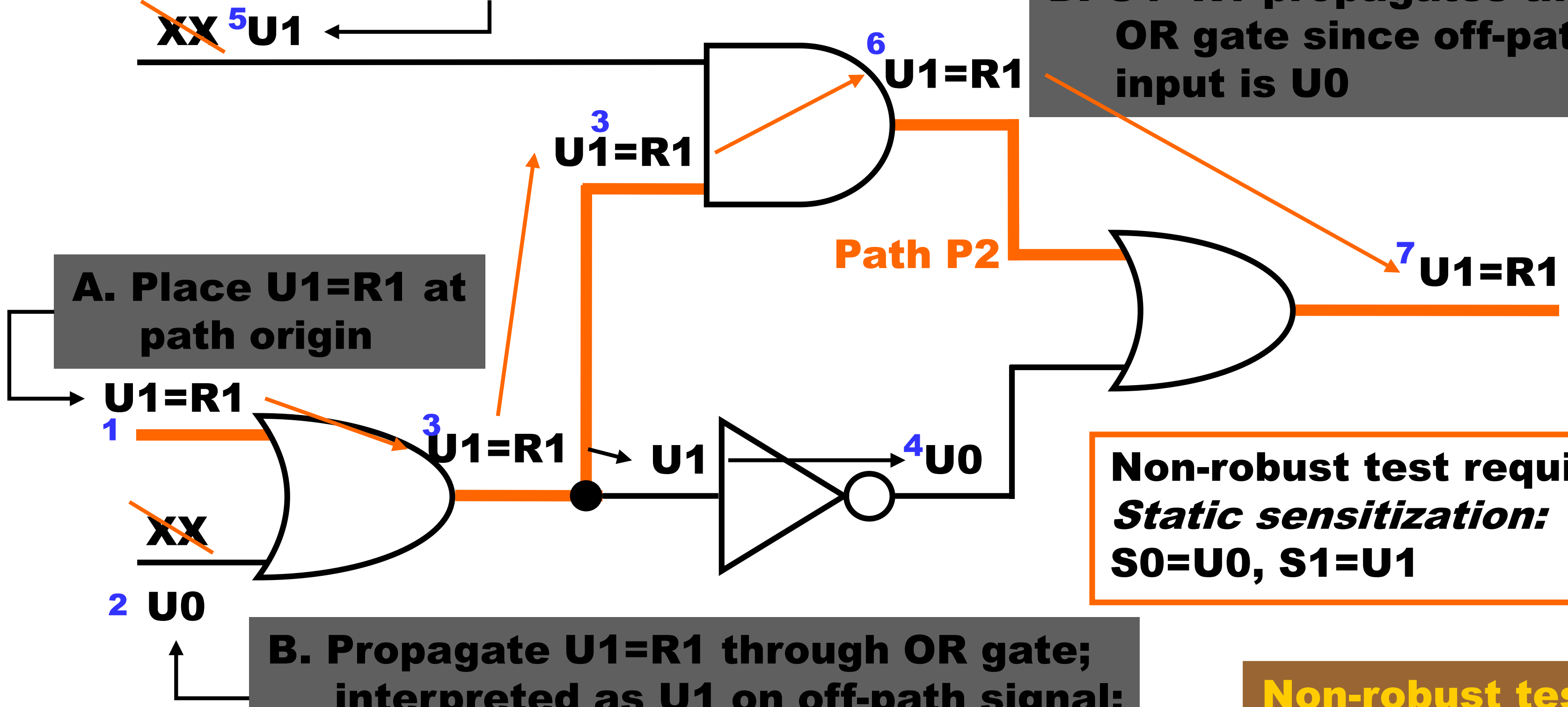
Fault ↑P2 – rising transition through path P2 has no robust test.



gate type / on-input transition	AND or NAND	OR or NOR
Rising(U1)	U1	U0
Falling(U0)	U1	U0

C. Set input of AND gate to propagate U1=R1 to output

D. U1=R1 propagates through OR gate since off-path input is U0



A. Place U1=R1 at path origin

B. Propagate U1=R1 through OR gate; interpreted as U1 on off-path signal; propagates as U0 through NOT gate

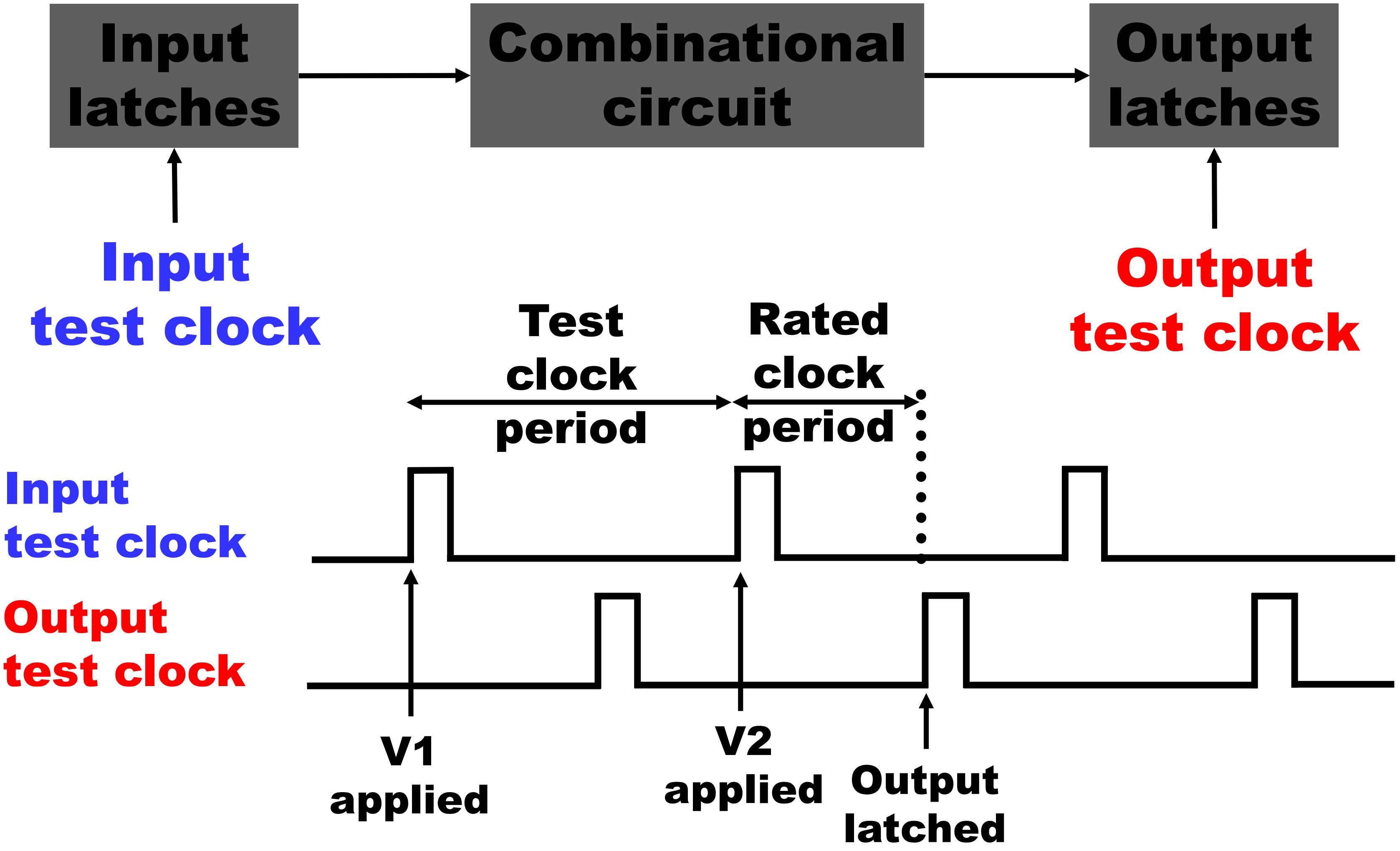
Non-robust test requires *Static sensitization:* S0=U0, S1=U1

Non-robust test: (U1, U1, U0)

(V1,V2)=(x0x,110)

Advanced Techniques

Slow-Clock Test



Input latches

Combinational circuit

Output latches

Input test clock

Output test clock

Test clock period

Rated clock period

Input test clock

Output test clock

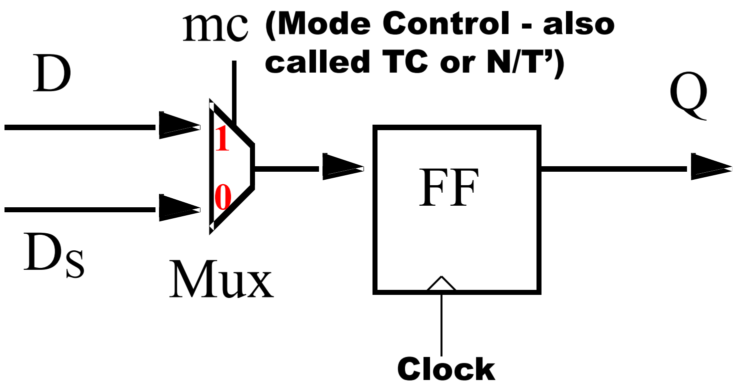
V1 applied

V2 applied

Output latched

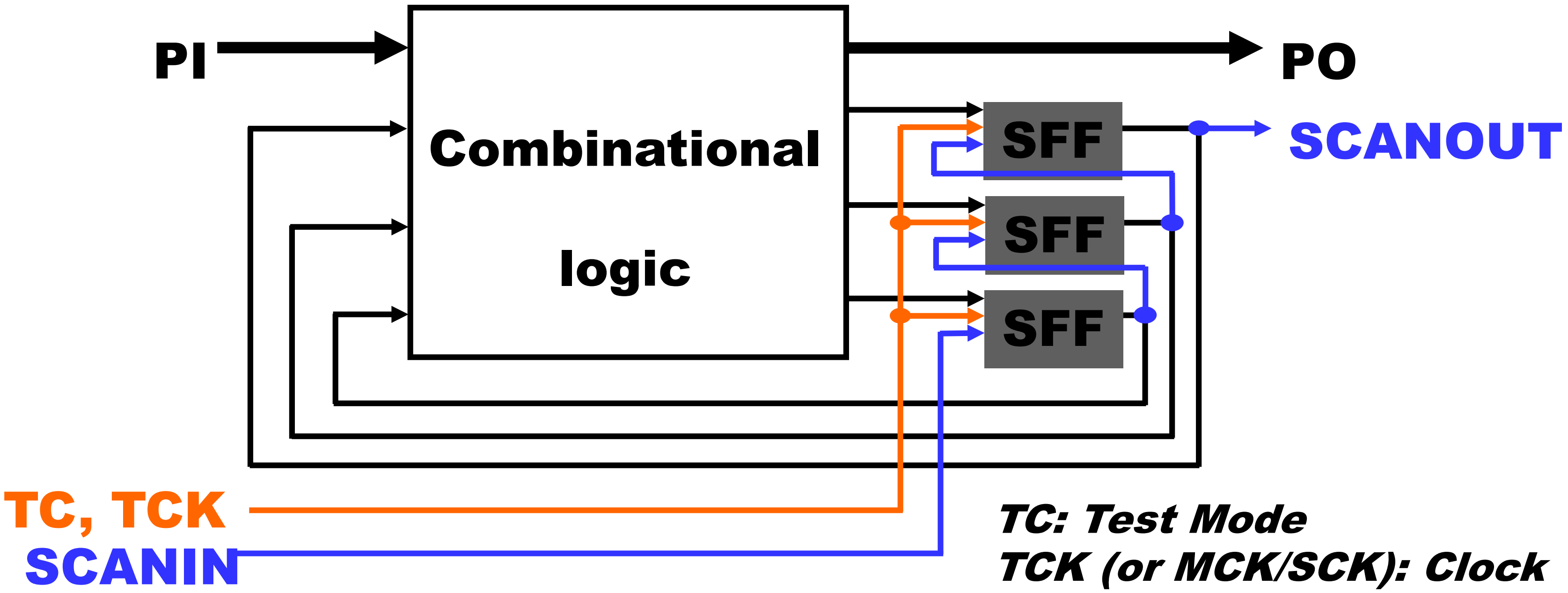
Review - Scan as a DFT Method

- In scan DFT (design for testability) methodology, flip-flops (or latches) designed to support two modes
  - —**Normal mode**: Flip-flops configured as in the original circuit
  - —**Test mode**: Flip-flops configured as one or more shift-registers, called **scan registers** or **scan chains**
- Most Common SSF cell



Review - Scan Structure

- Testing using scan
  - In normal mode, responses at state outputs captured in flip-flops
  - Circuit then configured in the test mode
    - Scan registers clocked
    - The output of the last flip-flop in scan chain observed
  - At the same time, values to be applied at state inputs in the subsequent test shifted into flip-flops



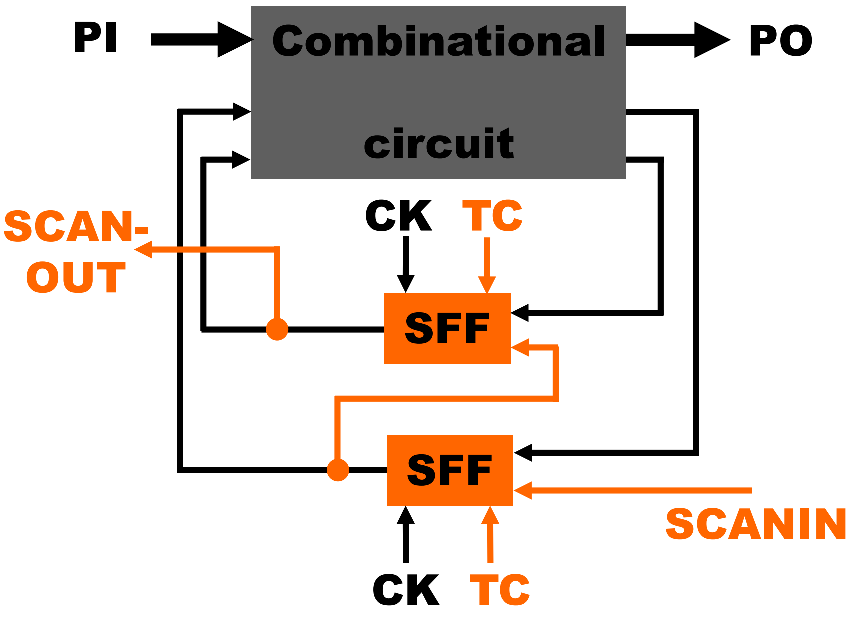
TC: Test Mode
TCK (or MCK/SCK): Clock

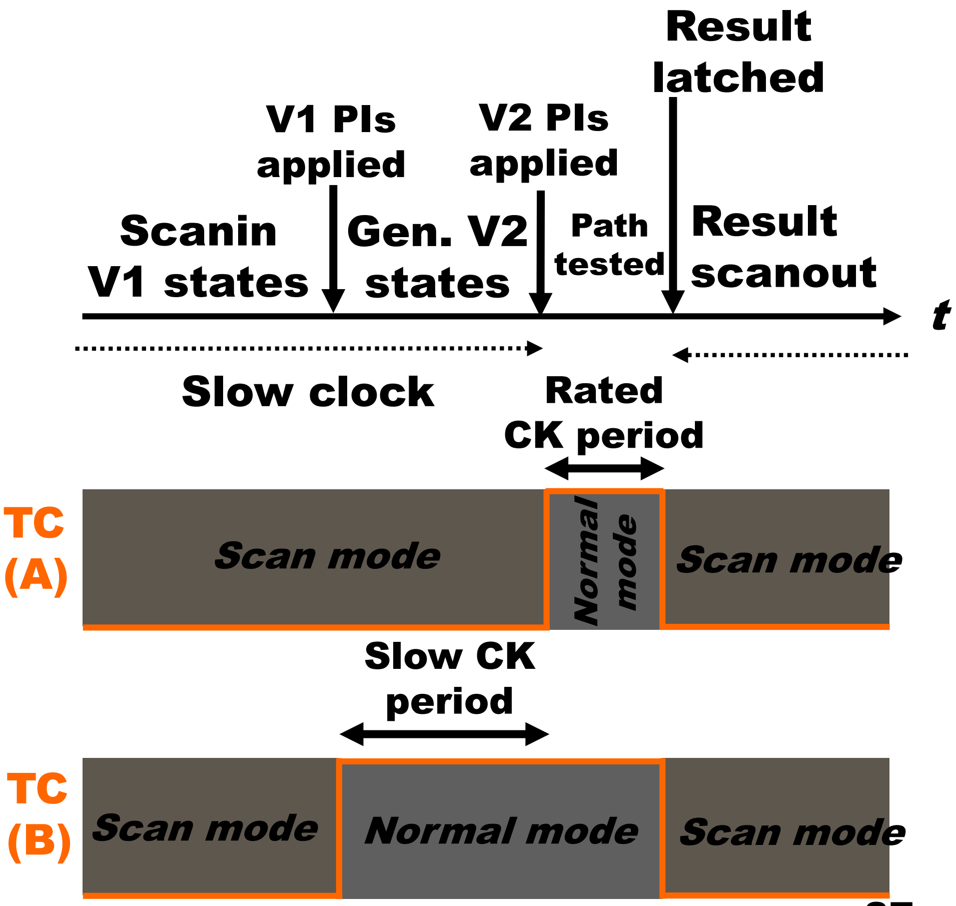
Normal-Scan Test – Two Special Cases

- **V2 state/vector is generated**

  (A) scan-shift delay test: V2 = one-bit scan shift of V1, or

  (B) broad-side delay test: V2 = f(V1) = V1 applied in functional mode.

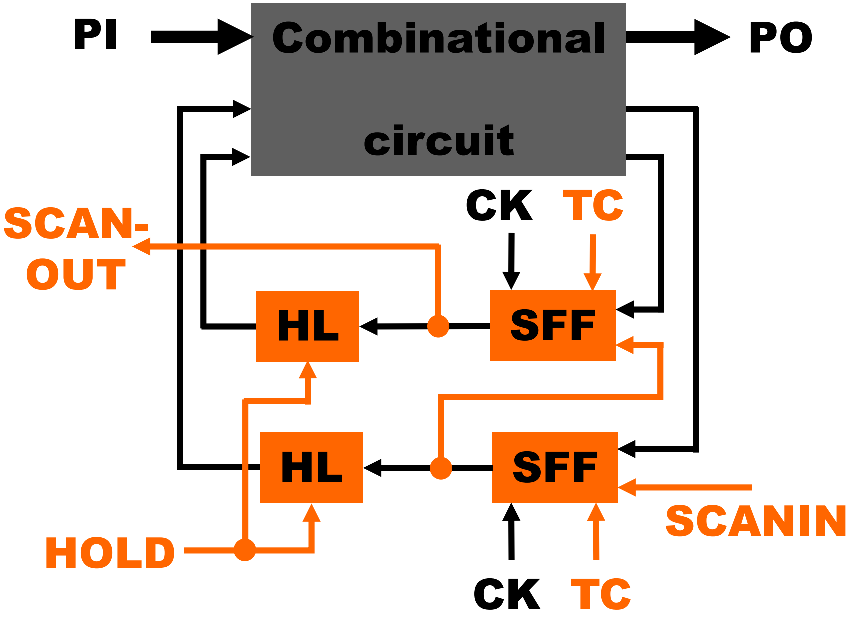




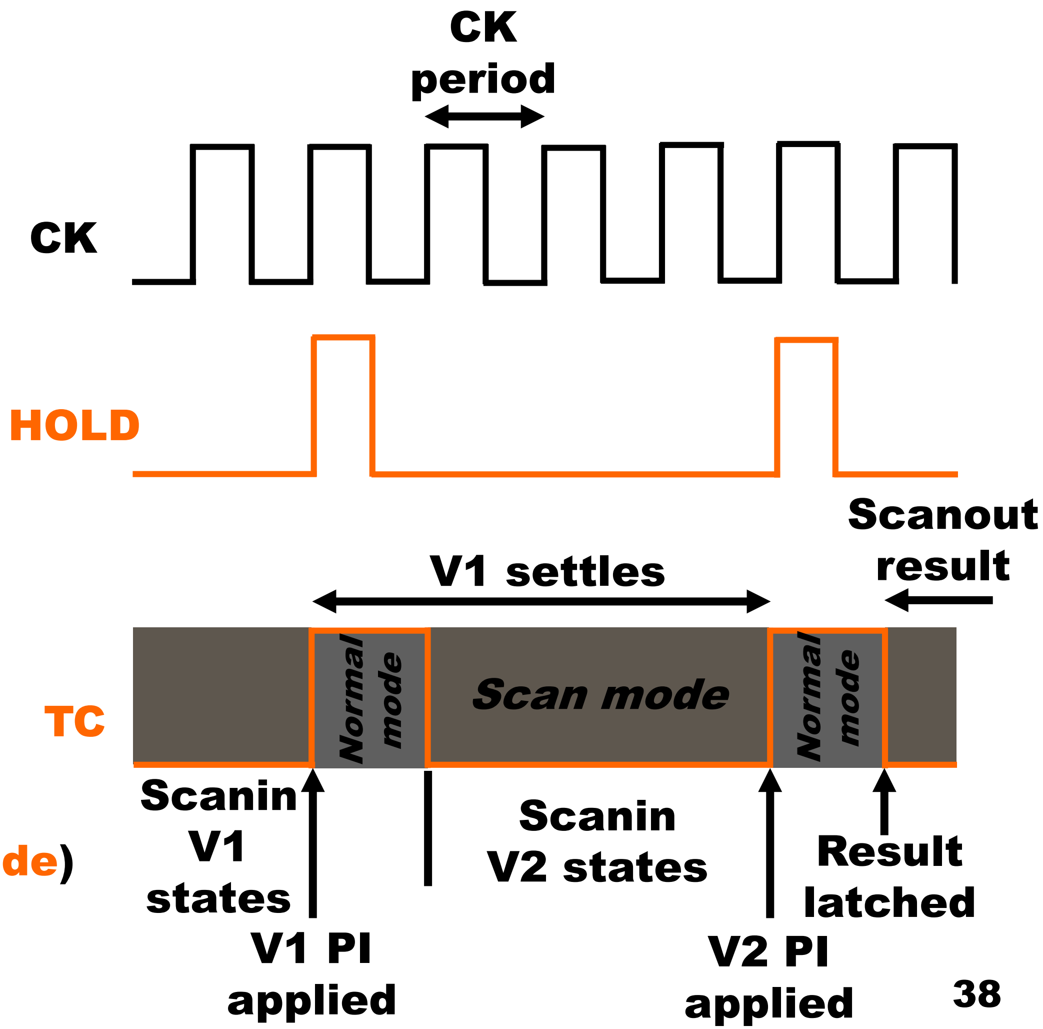
CK: system clock
TC: test control (TC=1 normal mode)
SFF: scan flip-flop

Enhanced-Scan Test – General Case

- Shift the state bits of V1 serially (TC=0 and apply CK). Use slow clock to minimize power dissipation.
- Transfer V1 to hold latches by activating HOLD while the PI bits of V1 are applied at PI.
- Shift the state bits of V2. Activate HOLD and apply PI bits of V2. This makes V1→V2

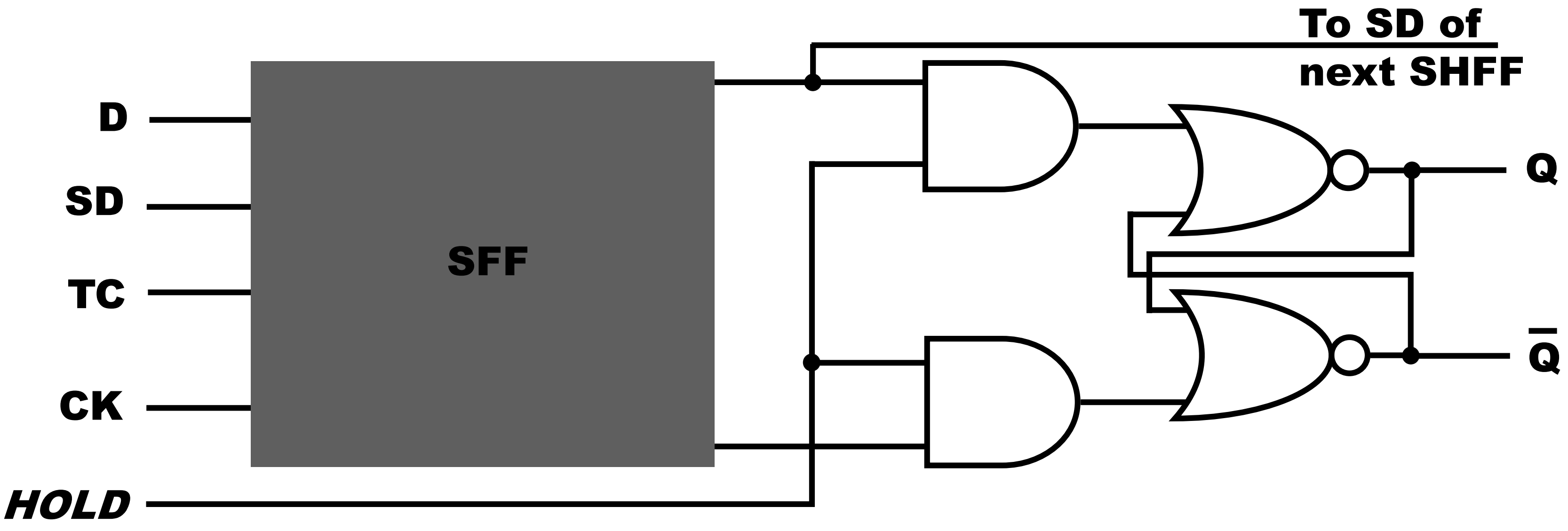


CK: system clock
TC: test control (TC=1 normal mode)
HOLD: hold signal
SFF: scan flip-flop
HL: hold latch



Enhanced-Scan Test (cont.)

- Structure and operation of scan-hold flip-flop (SHFF):
  - The control input *HOLD* keeps the output steady at previous state of flip-flop.
  - Applications:
    - Reduce power dissipation during scan
    - Isolate asynchronous parts during scan test
    - Delay testing (applying V1→V2 pair)



At-Speed Test

- At-speed test means application of test vectors at the rated-clock speed.
- Two methods of at-speed test.
- External test:
  - Vectors may test one or more functional critical (longest delay) paths and a large percentage (~100%) of transition faults.
  - High-speed testers are expensive.
- *Built-in self-test* (BIST):
  - Hardware-generated random vectors applied to combinational or sequential logic.
  - Only clock is externally supplied.
  - Non-functional paths that are longer than the functional critical path can be activated and cause a good circuit to fail.
  - Some circuits have initialization problem.

Timing Design & Delay Test

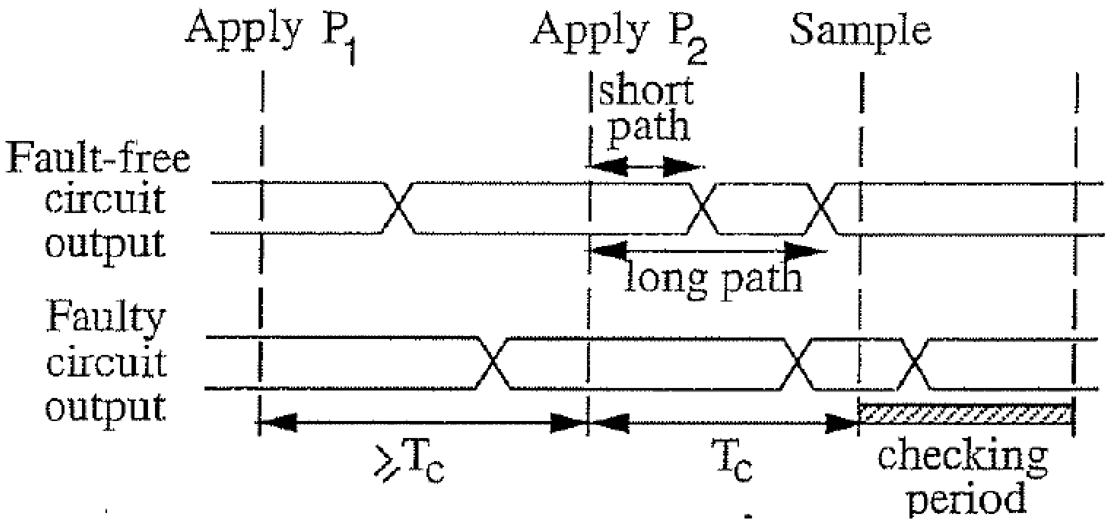
- Timing simulation:
  - Critical paths are identified by static (vector-less) timing analysis tools like *Primetime* (Synopsys).
  - Timing or circuit-level simulation using designer-generated functional vectors verifies the design.
- Layout optimization: Critical path data are used in placement and routing. Delay parameter extraction, timing simulation and layout are repeated for iterative improvement.
- Testing: Some form of at-speed test is necessary. PDFs for critical paths and all transition faults are tested.

Simplifications in Path-Delay Test

- Nodes unrelated to the path may affect the path delay (e.g. due to capacitive couplings, etc.)
- Different gate inputs are not equivalent from the electrical point of view. E.g. in CMOS NAND, delay of 11→00 can be 60-90% higher than 11→01 (or 11→10).
- Transitions at off-path fanouts may result in considerably longer (or shorter) delays on the path under test.

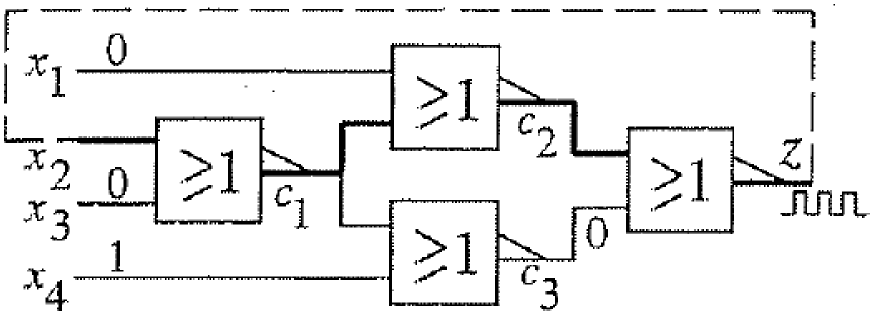
Output Waveform Analysis

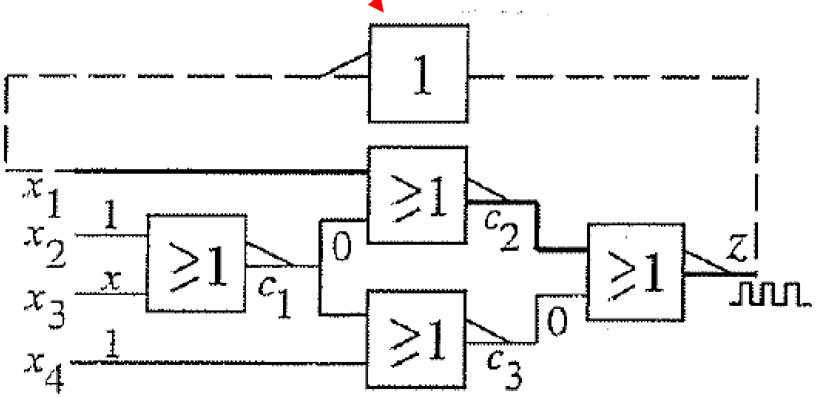
- A post-sampling waveform analysis is performed to conclude the test result.
  - The output of a fault-free circuit will be stable after the sampling time
  - The output of a faulty circuit will have transitions which will be caught.
- When (V1,V2) is applied, after clock period (T_c), if an incorrect value is latched then the PDF is detected. A stability checker can detect hazards or faulty transitions even if the correct value is latched.



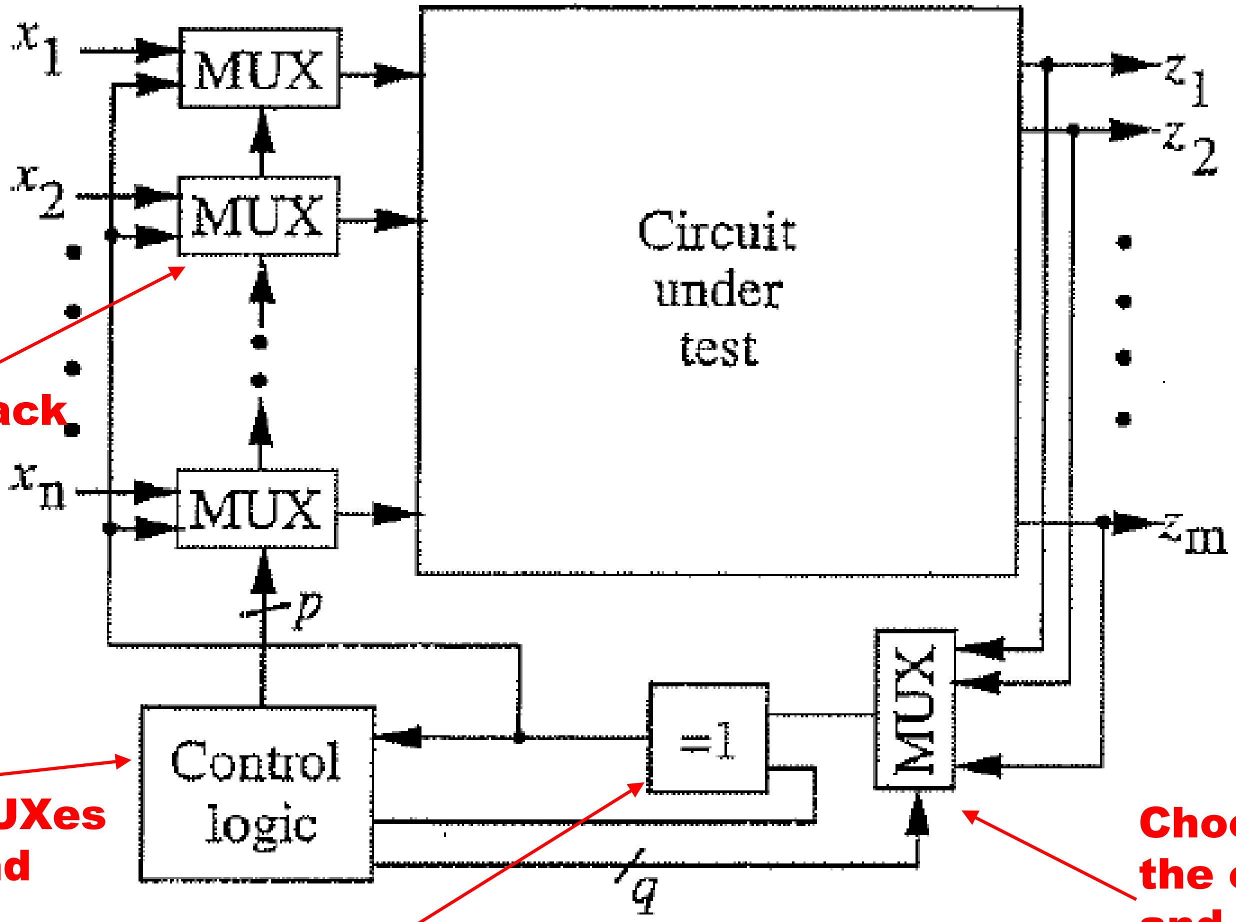
Digital Oscillation Testing

- Make sure there are an odd number of inverters in the test loop. DF and SAF manifest themselves by causing a deviation from the fault-free oscillation frequency.
  - The $x_2c_1c_2z$ path is sensitized and converted to an oscillator by test vector $(x_1,x_2,x_3,x_4)=(0,z,0,1)$. Note that SA0 or SA1 on this path will not cause any oscillation.
  - To sensitize x_1c_2z, use an external inverter. The vector that sensitize this path is: $(x_1,x_2,x_3,x_4)=(z,1,x,1)$.





Generalization of Oscillation Testing



Chooses the feedback loop

Controls MUXes and XOR and measures f_{osc}

Establishes an inverting/non-inverting loop

Chooses the output and establish the feedback