

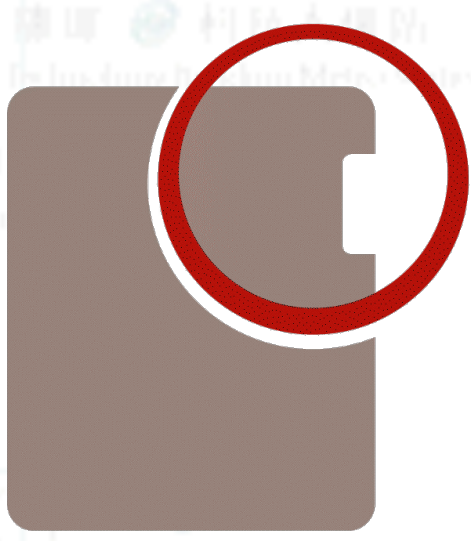
Design-for-Test

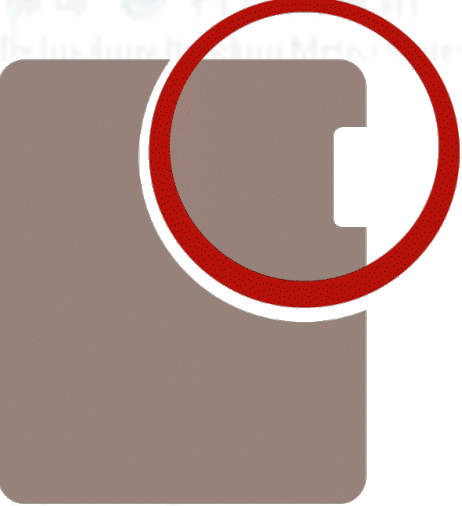
Jiun-Lang Huang

GIEE/ICDA, National Taiwan University

Evolution of Design-for-Test

- As the design complexity keeps growing, design without testing in mind eventually leads to prohibitively high test costs or poor test quality.
- It is difficult to set the circuit under test to desired internal states.
- Low sequential test pattern generation (TPG) efficiency.
- ***Ad hoc approaches*** were developed to improve controllability and observability.
 - For example, test point insertion.
 - Still suffer poor sequential TPG efficiency — difficult to reach 90% fault coverage for large designs.




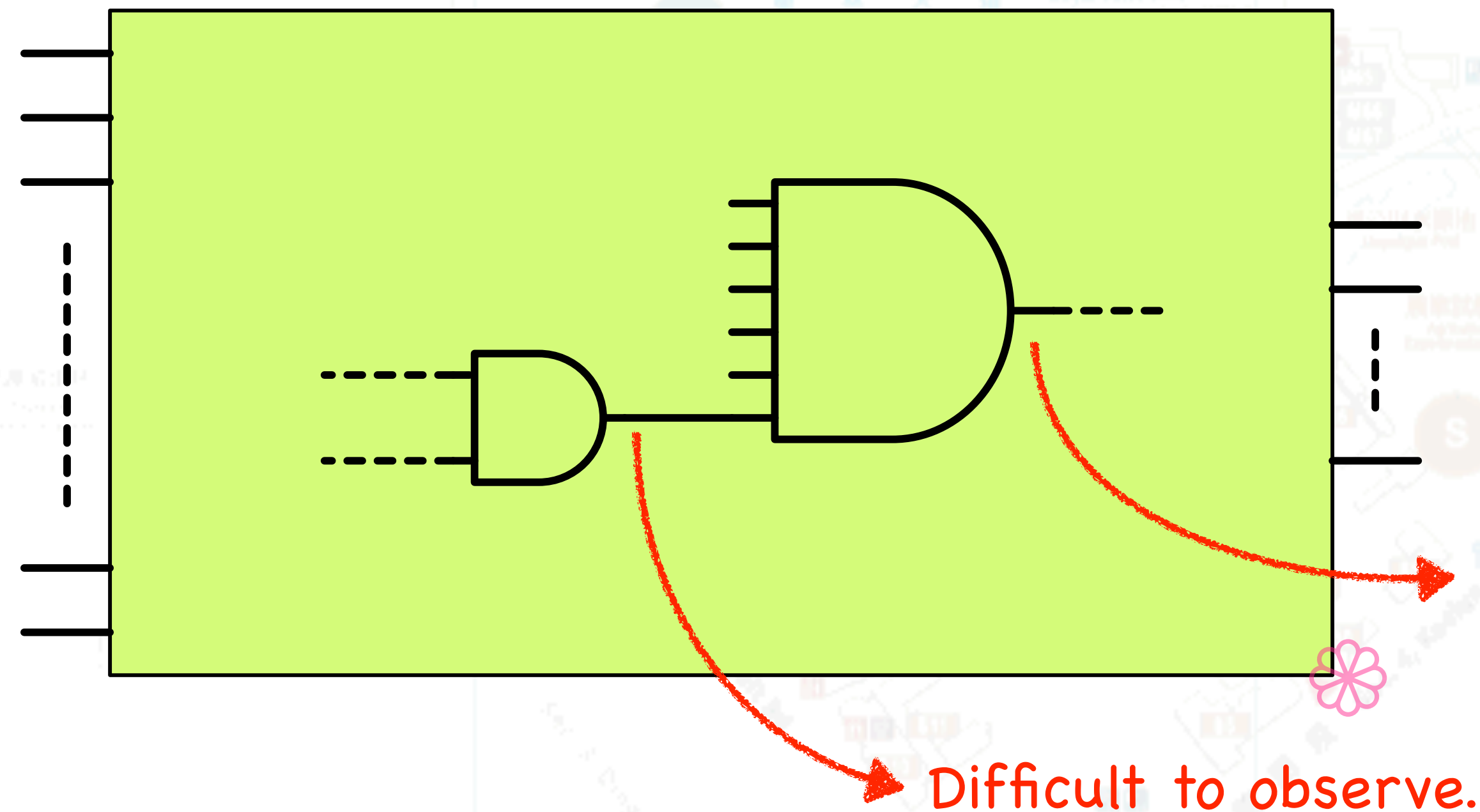
- 
- ***Scan design*** was developed to exploit the more efficient combinational TPG technologies.
 - With full-scan, flip-flops are completely controllable and observable.
 - Variants of scan design:
 - full-scan, almost full-scan,
 - partial scan (w/ sequential TPG), and
 - pipelined/feedforward/balanced partial-scan design (w/ combinational TPG).
 - Based on scan infrastructure, ***built-in self-test (BIST)*** and ***test compression*** techniques are developed to lower the test cost.



Testability Analysis

Testability

- A relative measure of the effort or cost of testing a logic circuit.
- Assuming that only PIs can be controlled and only POs can be observed.
- Reflect on the efforts required to set and observe internal signals. 



Testability Analysis

- The process of assessing the testability of a logic circuit by calculating a set of numerical measures for each signal.
- Controllability and observability.

| | patterns | computation efforts | accuracy | TPG guidance |
|------------------|---------------|---------------------|----------|--------------|
| topology based | no (static) | low | low | yes |
| simulation based | yes (dynamic) | high | high | no |

A tradeoff between accuracy and complexity.

SCOAP

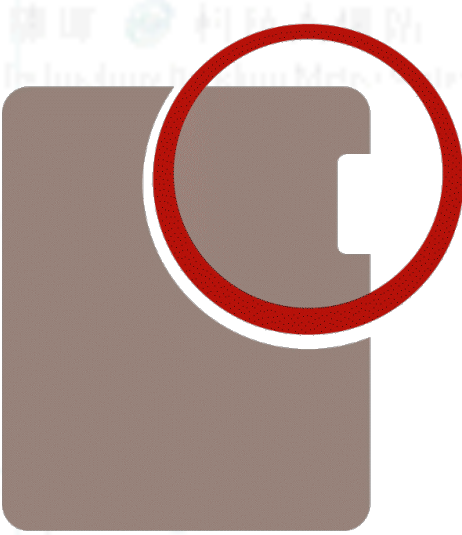
- Sandia Controllability/Observability Analysis Program [Goldstein 79].
- SCOAP computes six integer values for each signal s in a circuit.

| | 0-controllability | 1-controllability | observability |
|---------------|-------------------|-------------------|---------------|
| combinational | CC0(s) | CC1(s) | CO(s) |
| sequential | SC0(s) | SC1(s) | SO(s) |

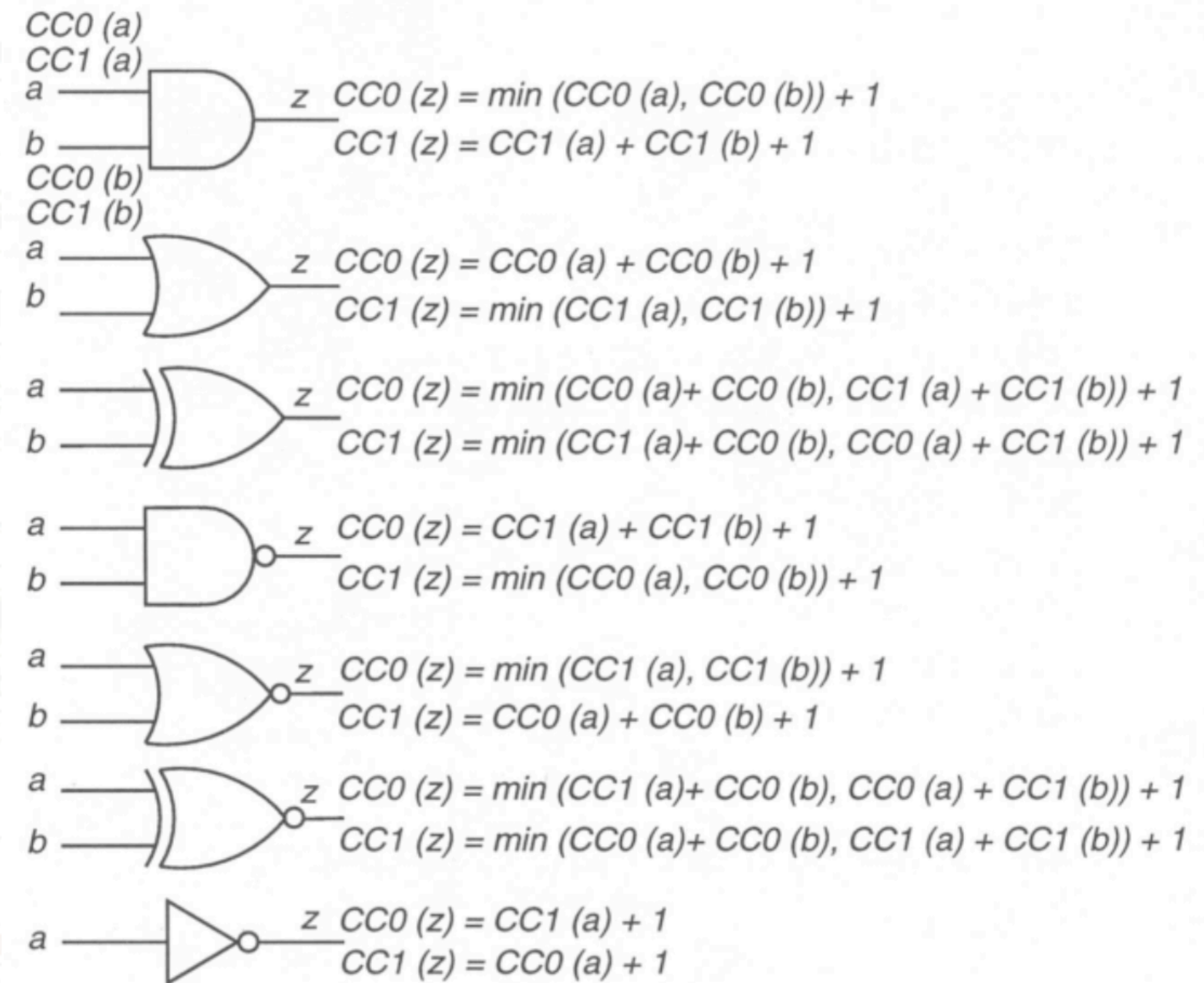
- $1 \leq CC0, CC1, SC0, SC1 < \infty$

- $0 \leq CO, SO < \infty$

SCOAP Combinational Controllability Measures



- Performed from PIs to POs according to the topological order.
- Process a signal only after all the inputs of its driving gates have been processed.
- Rules for controllability calculation:
 - 1 account for the logic depth.
 - Assume independent, uncorrelated gate inputs.
 - Fanout branches inherit controllability measures from the fanout stem.



[M. L. Bushnell and V. D. Agrawal, *Essentials of Electronic Testing*, Kluwer Academic Publishers, 2000.]



| | CC0 | CC1 |
|----------------------|--|--|
| primary input | 1 | 1 |
| AND | $\min(\text{input CC0's}) + 1$ | $\text{sum}(\text{input CC1's}) + 1$ |
| OR | $\text{sum}(\text{input CC0's}) + 1$ | $\min(\text{input CC1's}) + 1$ |
| NOT | $(\text{input CC1}) + 1$ | $(\text{input CC0}) + 1$ |
| XOR* | $\min(\text{CC0}(a) + \text{CC0}(b), \text{CC1}(a) + \text{CC1}(b)) + 1$ | $\min(\text{CC0}(a) + \text{CC1}(b), \text{CC0}(a) + \text{CC1}(b)) + 1$ |
| BRANCH | $\text{CC0}(\text{stem})$ | $\text{CC1}(\text{stem})$ |
| NAND | $\text{sum}(\text{input CC1's}) + 1$ | $\min(\text{input CC0's}) + 1$ |
| NOR | $\min(\text{input CC1's}) + 1$ | $\text{sum}(\text{input CC0's}) + 1$ |
| BUFFER | $(\text{input CC0}) + 1$ | $(\text{input CC1}) + 1$ |
| XNOR | $\min(\text{CC0}(a) + \text{CC1}(b), \text{CC0}(a) + \text{CC1}(b)) + 1$ | $\min(\text{CC0}(a) + \text{CC0}(b), \text{CC1}(a) + \text{CC1}(b)) + 1$ |

* a and b are the gate inputs.

Combinational Observability Measures

- Performed from POs to PIs after calculating controllability measures in a reverse pass.
- No distinction between logic 0 and 1 observability.
- Rules for observability calculation:
 - 1 account for the logic depth.
 - The assumption of independent observation via fanout branches incurs errors.

$$CO(a) = CO(z) + CC1(b) + 1$$

$$CO(b) = CO(z) + CC1(a) + 1$$

$$CO(a) = CO(z) + CC0(b) + 1$$

$$CO(b) = CO(z) + CC0(a) + 1$$

$$CO(a) = CO(z) + \min(CC0(b), CC1(b)) + 1$$

$$CO(b) = CO(z) + \min(CC0(a), CC1(a)) + 1$$

$$CO(a) = CO(z) + CC1(b) + 1$$

$$CO(b) = CO(z) + CC1(a) + 1$$

$$CO(a) = CO(z) + CC0(b) + 1$$

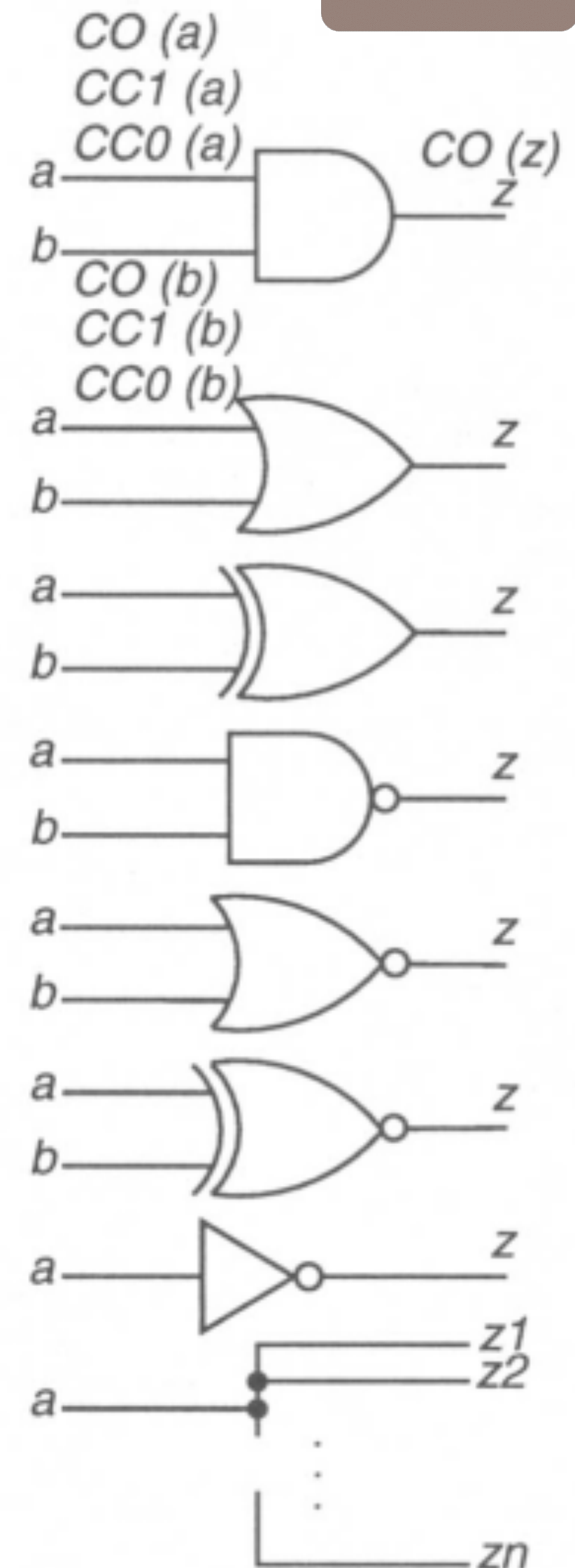
$$CO(b) = CO(z) + CC0(a) + 1$$

$$CO(a) = CO(z) + \min(CC0(b), CC1(b)) + 1$$

$$CO(b) = CO(z) + \min(CC0(a), CC1(a)) + 1$$

$$CO(a) = CO(z) + 1$$

$$CO(a) = \min(CO(z1), CO(z2), \dots, CO(zn))$$

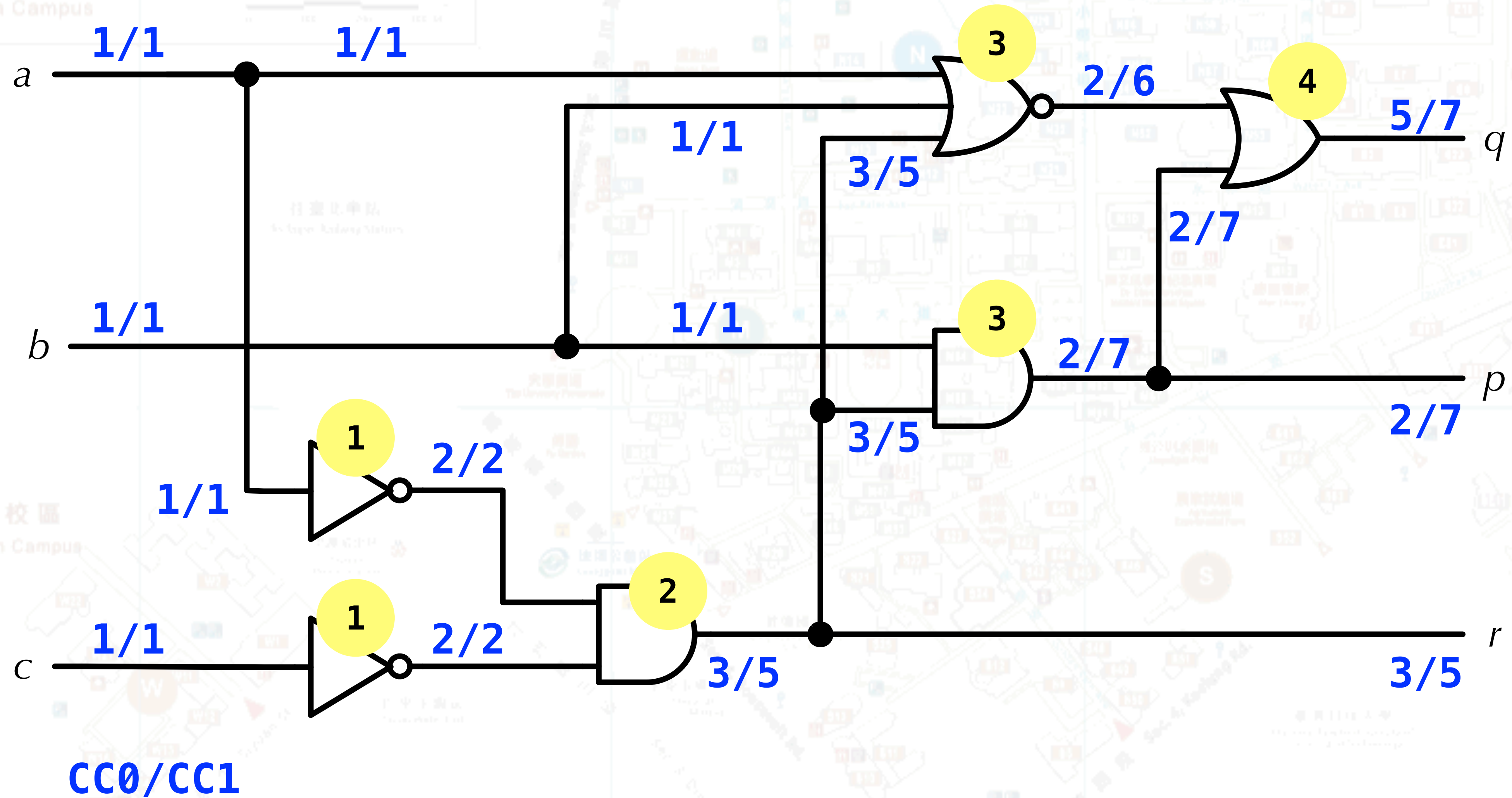




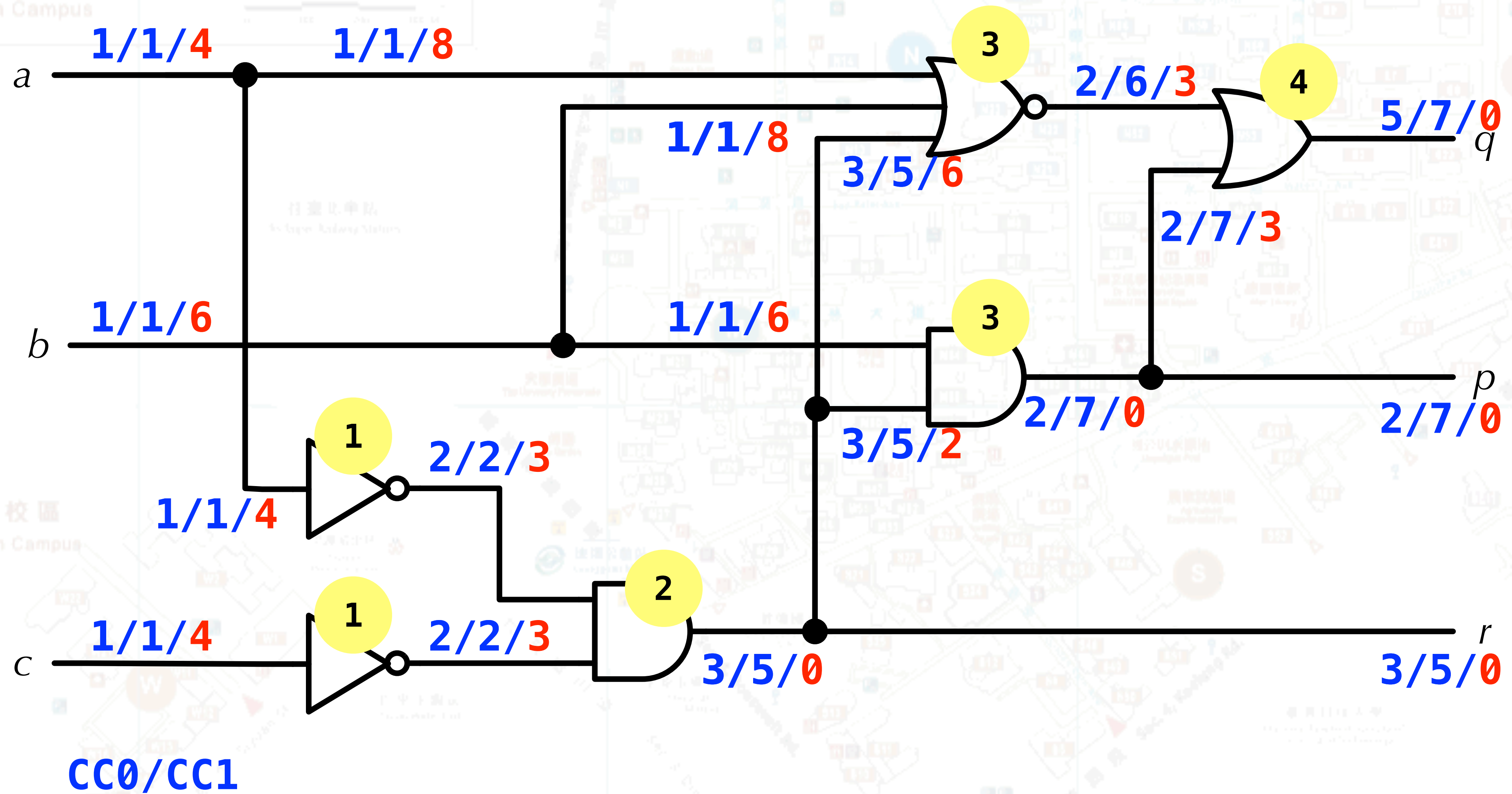
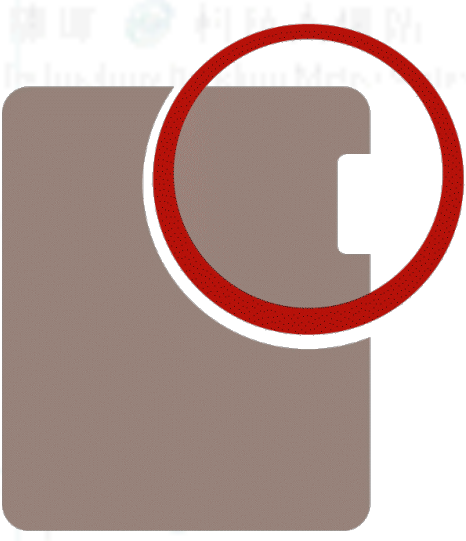
| | CO |
|------------------|---|
| primary output | 0 |
| AND/NAND input | $\text{sum}(\text{output CO, other inputs' CC1}) + 1$ |
| OR/NOR input | $\text{sum}(\text{output CO, other inputs' CC0}) + 1$ |
| NOT/BUFFER input | $(\text{output CO}) + 1$ |
| XOR/XNOR input* | $(\text{output CO}) + \min(\text{CC0}(x), \text{CC1}(x)) + 1$ |
| stem | $\min(\text{branch CO's})$ |

* x is the other input.

SCOAP Example — CC0/CC1

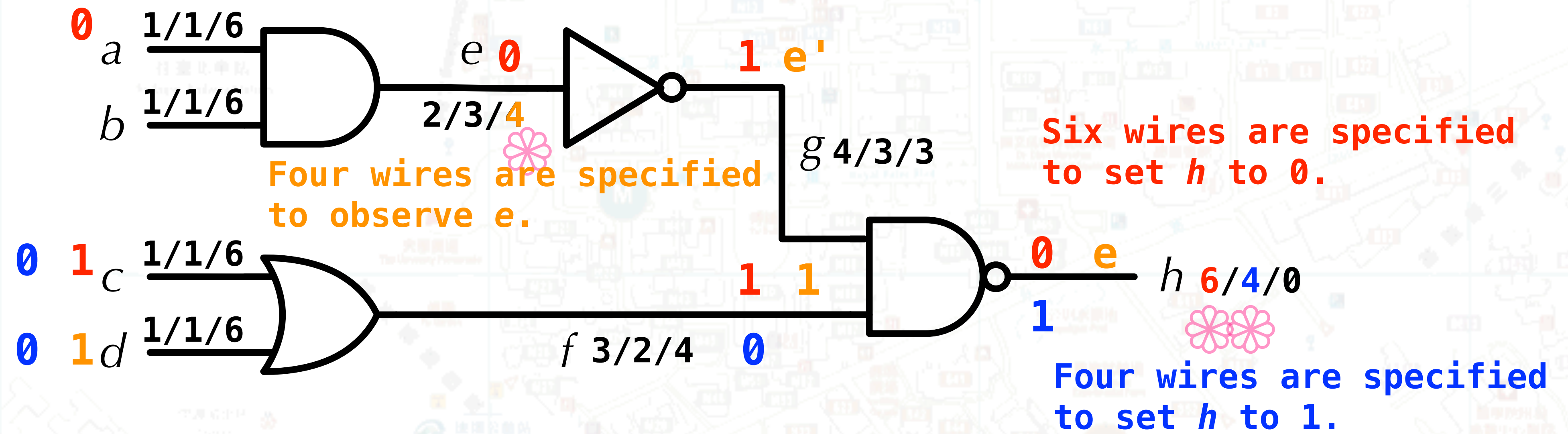


SCOAP Example — CO

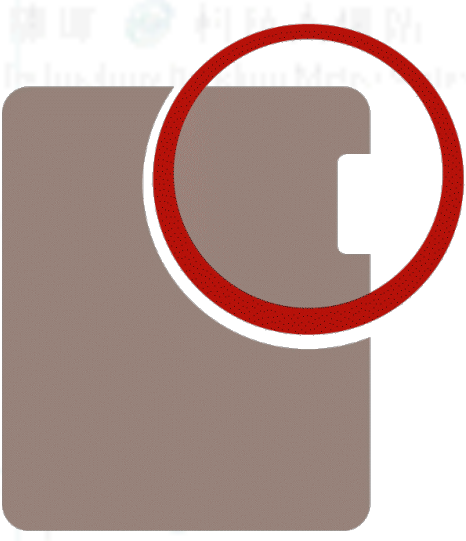
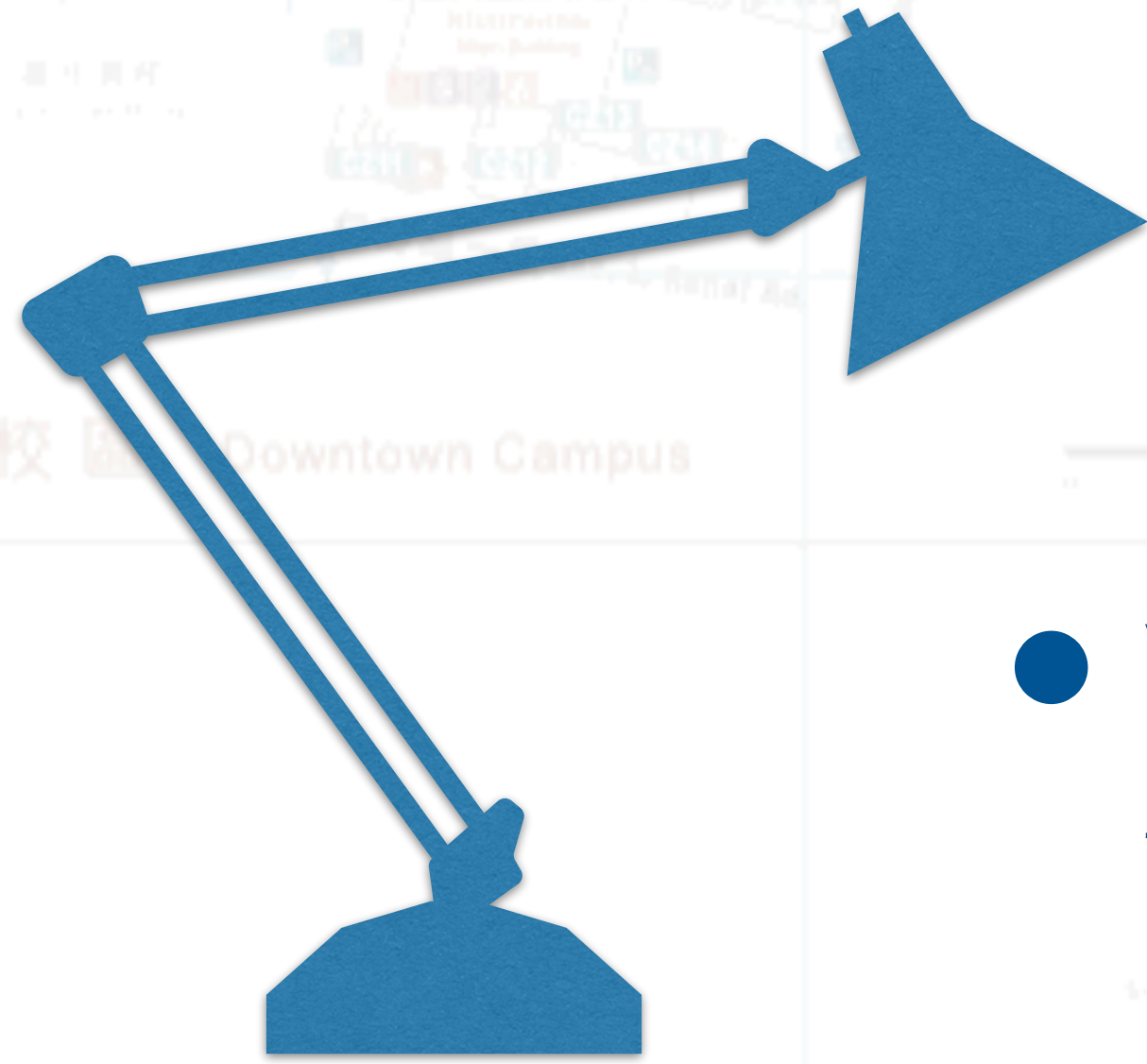


Remarks on SCOAP

- The controllability and observability numbers approximate the number of circuit lines that must be set to control or observe a given circuit line.



- Larger CC/CO values indicate higher difficulty controlling/observing a signal's value.



- What's the rule for calculating SCOAP combinational testability measures for 3-input XOR gates?
- Give examples where the CC0/CC1/CO rules overestimate or underestimate.

Probability-Based Testability Analysis

- SCOAP is extremely helpful in TPG.
- For BIST applications that utilize pseudo-random patterns, probability-based testability measures can be used to derive *random testability*.

| | 0-controllability | 1-controllability | observability |
|--|-------------------|-------------------|---------------|
| probability-based testability measures | $C0(s)$ | $C1(s)$ | $O(s)$ |

- $0 \leq C0, C0, O \leq 1$
- $C0(s) + C1(s) = 1$



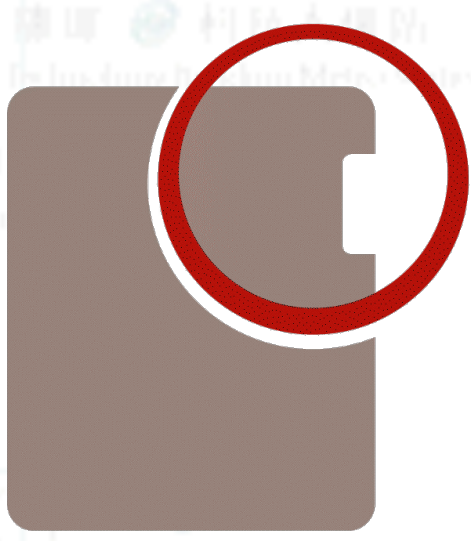
- Probability-based controllability calculation rules:
 - The derivation process is similar to SCOAP, with different rules.
 - Assume that gate inputs are independent.
 - A smaller $C1(s)/C0(s)$ indicates that it is harder to set s to 1/0.

| | C0 | C1 |
|----------------|--------------------------------|--------------------------------|
| primary input* | p_0 | $p_1 = 1 - p_0$ |
| AND | $1 - \prod(\text{input C1's})$ | $\prod(\text{input C1's})$ |
| OR | $\prod(\text{input C0's})$ | $1 - \prod(\text{input C0's})$ |
| NOT | (input C1) | (input C0) |
| XOR | $C0(a)C0(b) + C1(a)C1(b)$ | $C0(a)C1(b) + C1(a)C0(b)$ |
| BRANCH | (stem C0) | (stem C1) |

* p_0 is the PI's 0-probability, $0 \leq p_0 \leq 1$.

- Probability-based observability calculation rules:

- A smaller $O(s)$ indicates that it is harder to observe the value of s .



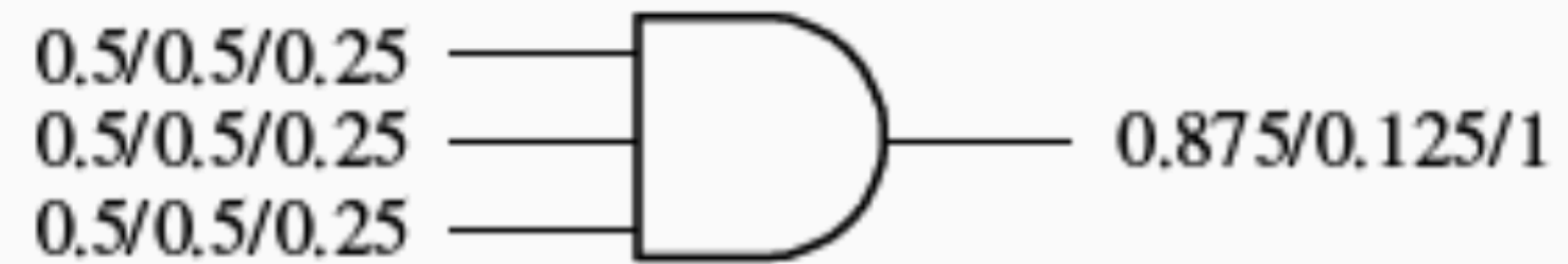
| | O |
|-------------------------|--|
| primary output | 1 |
| AND/NAND input | $\prod(\text{output } O, \text{ other inputs' } C1)$ |
| OR/NOR input | $\prod(\text{output } O, \text{ other inputs' } C0)$ |
| NOT/BUFFER input | (output O) |
| XOR/XNOR* | $\prod(\text{output } O, \max(C0(x), C1(x)))$ |
| stem | $\max(\text{branches' } O)$ |

* x is the other input.

- Example:



(a) SCOAP combinational measures



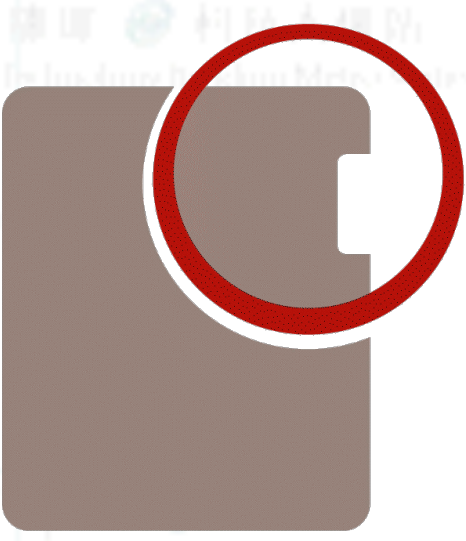
(b) Probability-based measures

[L.-T.Wang, C.-W.Wu, and X.Wen, *VLSI Test Principles and Architectures*, Morgan Kaufmann Publishers, 2006]

- Faults on signal lines with poor probability-based testability measures are often called ***random-pattern-resistant*** (PR-resistant).
- For BIST applications that utilize pseudo-random pattern generation, insert test points or adjust PI weights to address this issue.

Remarks on Topology-Based Testability Measures

- In general, linear complexity.
- Reconvergent circuit structures degrade accuracy.



Simulation-Based Testability Analysis

- More accurate than topology-based approaches, at the cost of higher computation efforts.
- They are derived through logic or fault simulation with given patterns.
- Statistical sampling to reduce the number of simulated patterns.
- Collect, for all or interested signals, the numbers of occurrences of 0's, 1's, 0-to-1 transitions, and 1-to-0 transitions.
- Generally used to guide testability enhancement, e.g., test point insertion, to achieve very high fault coverage.



Conclusions

- Testability analysis can be performed at different circuit abstraction levels (gate-level or RTL) and different strategies (static or dynamic).
- TPG guidance: gate-level + static — SCOAP
- Test point insertion for BIST: gate-level + static — probability-based testability measures
 - When very high fault coverage is demanded, use simulation-based testability measures.
- High-level testability improvement: RTL testability analysis




Ad Hoc Design-for-Test Techniques

Ad Hoc DfT Approaches

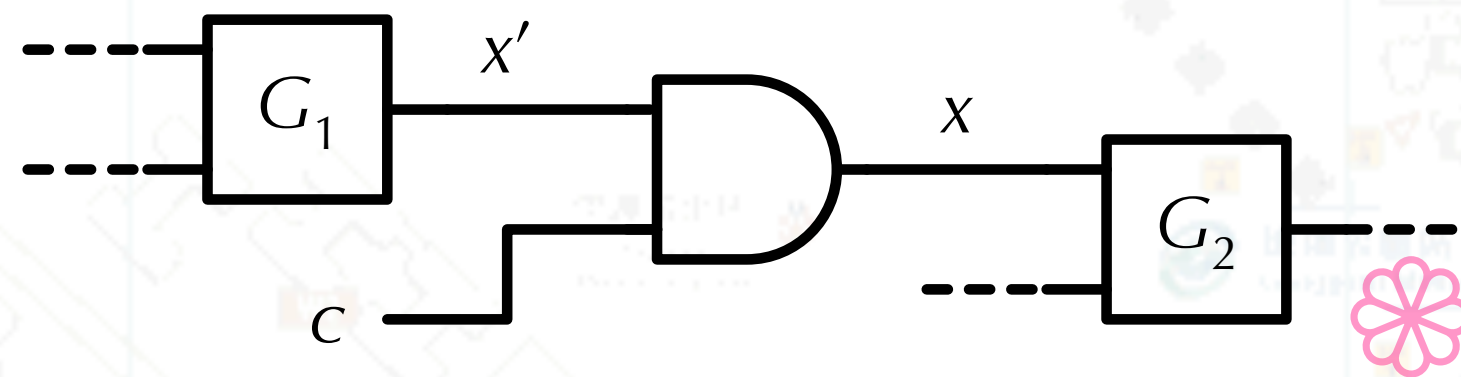
- Rely on design practice and modification guidelines for testability improvement.
- Initially proposed to deal with the difficulty of testing sequential circuits.
- Soon run out of steam:
 - Not scalable — human efforts are required.
 - Not too many testability experts to consult.
 - We still need sequential ATPG to validate the effectiveness.

| | |
|-----------|---|
| A1 | insert test points |
| A2 | avoid asynchronous set/reset for storage elements |
| A3 | avoid combinational feedback loops |
| A4 | avoid redundant logic |
| A5 | avoid asynchronous logic |
| A6 | partition a large circuit into small blocks |

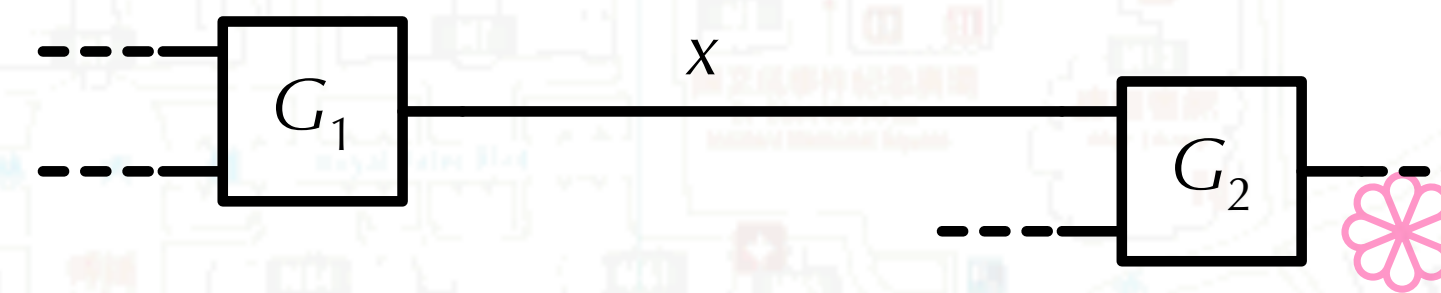
Test Point Insertion

- Add test points to enhance controllability or observability.
- Use testability analysis to determine where test points should be inserted.
- Control points:
Add additional primary inputs to enhance 1 or 0 controllability. 

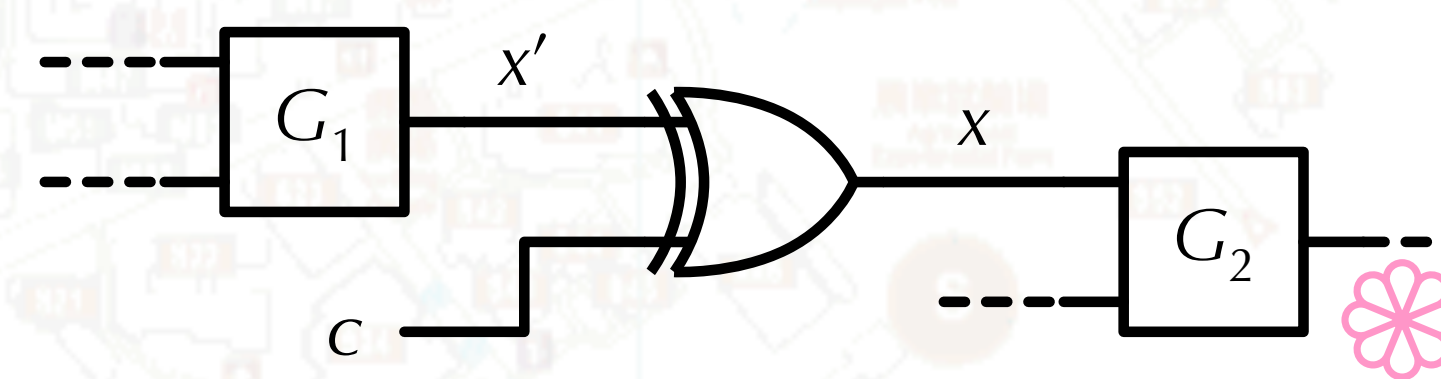
zero-control point



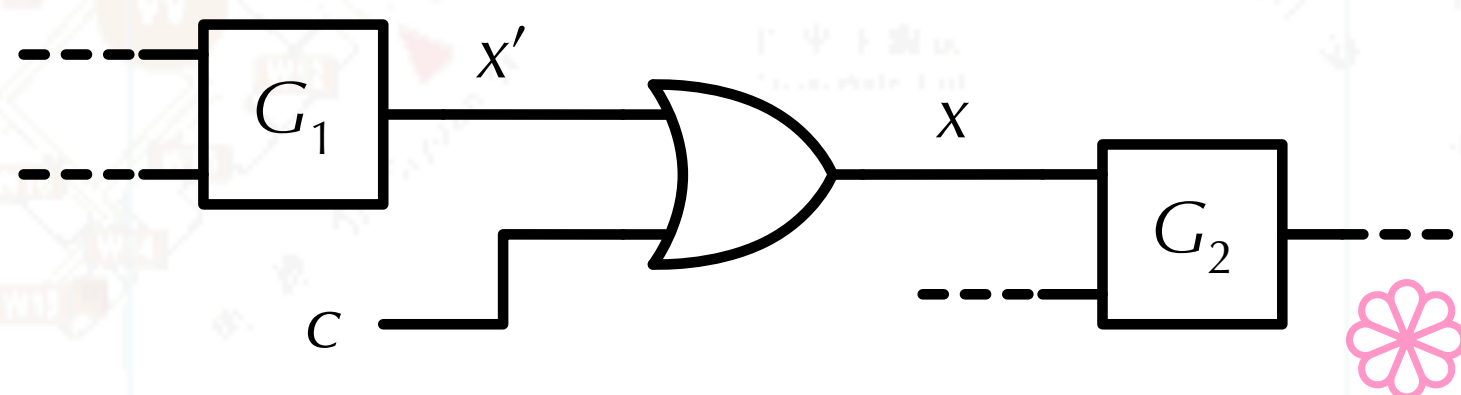
original



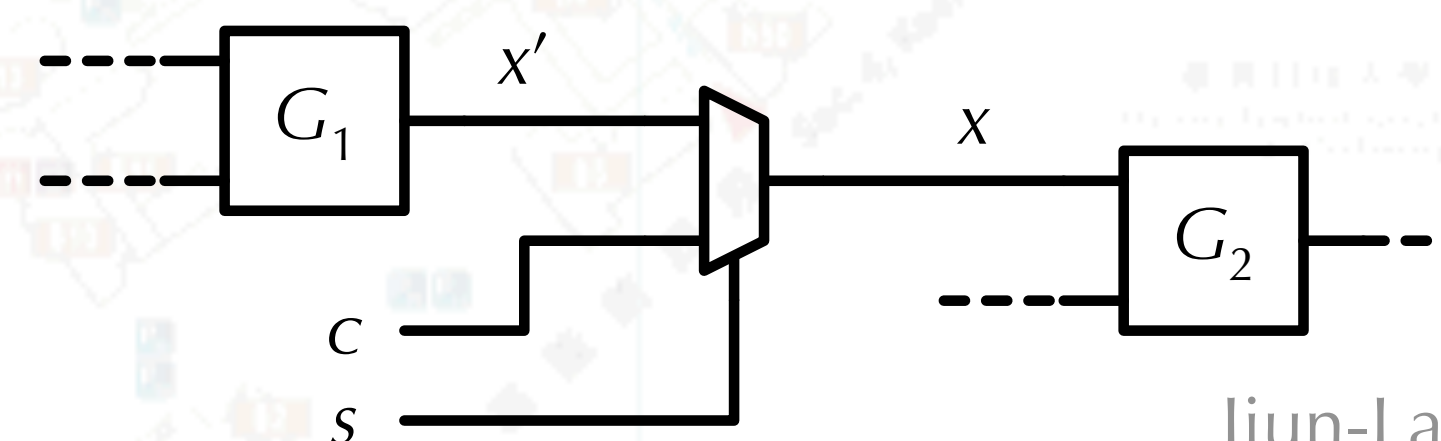
inversion-control point



one-control point

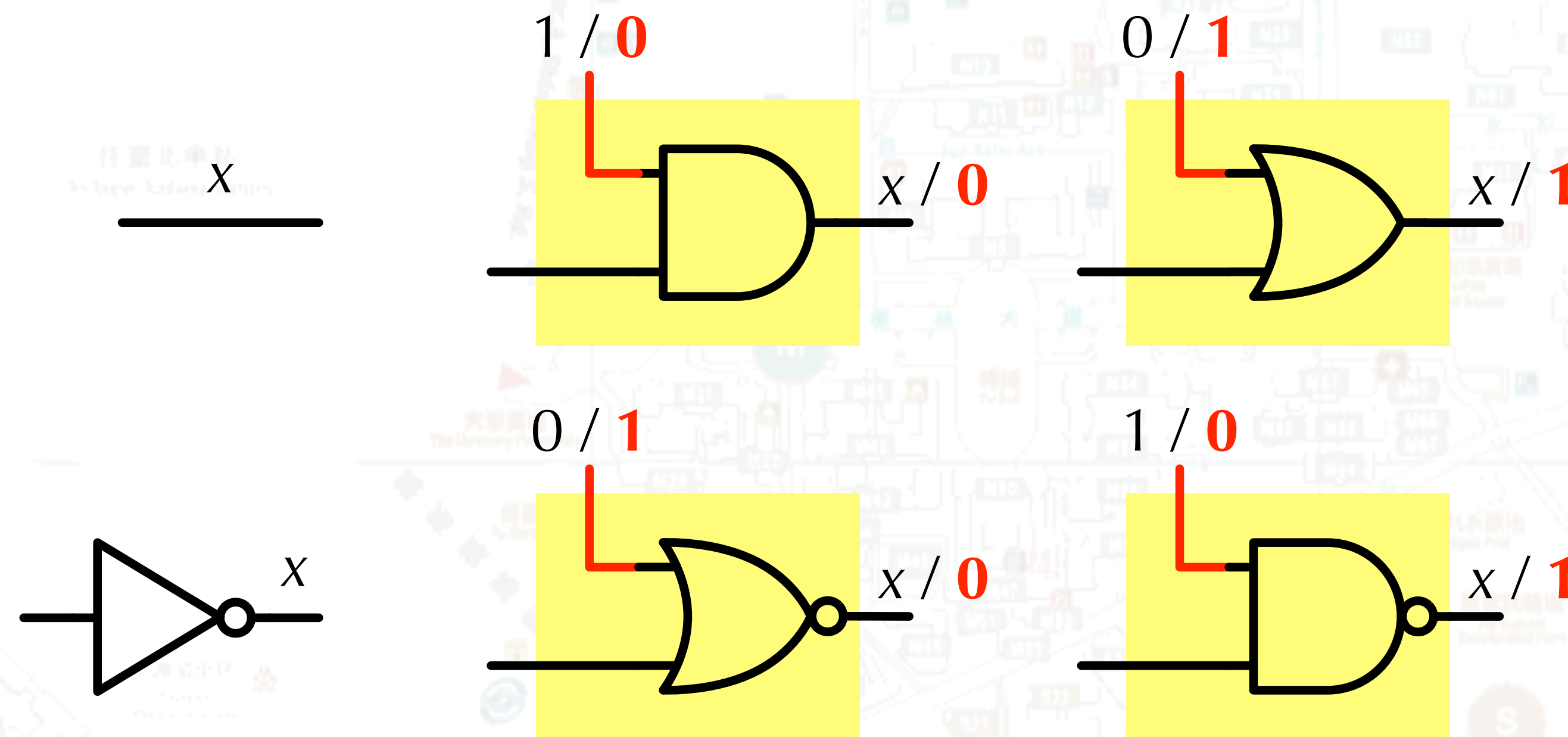


complete-control point

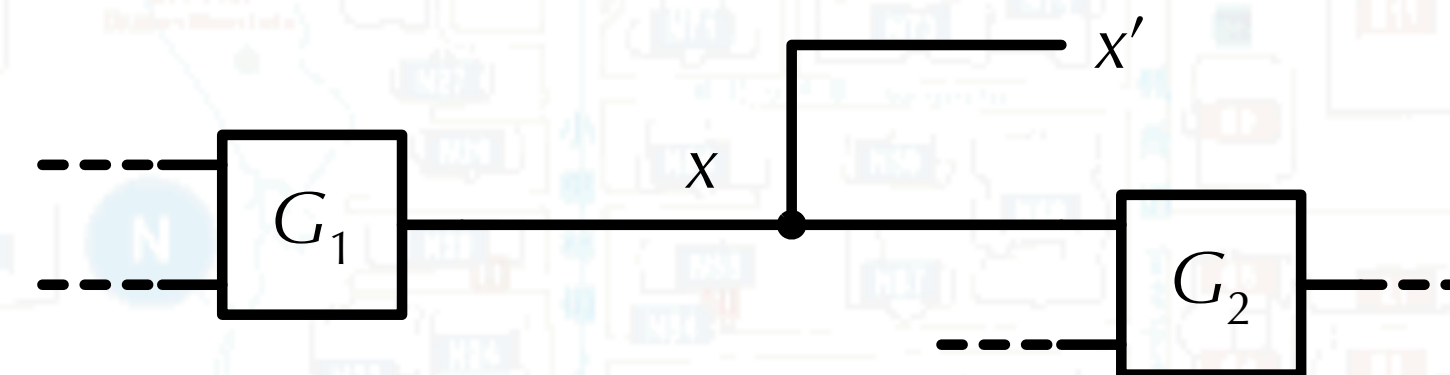
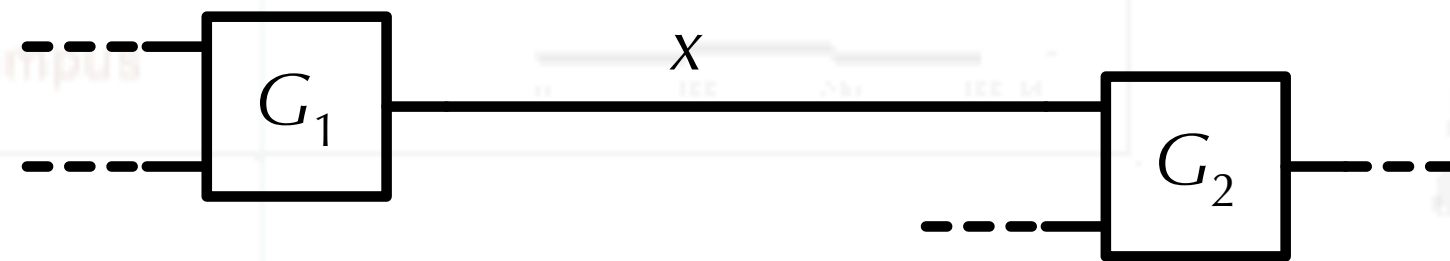


- Control points: (cont'd)

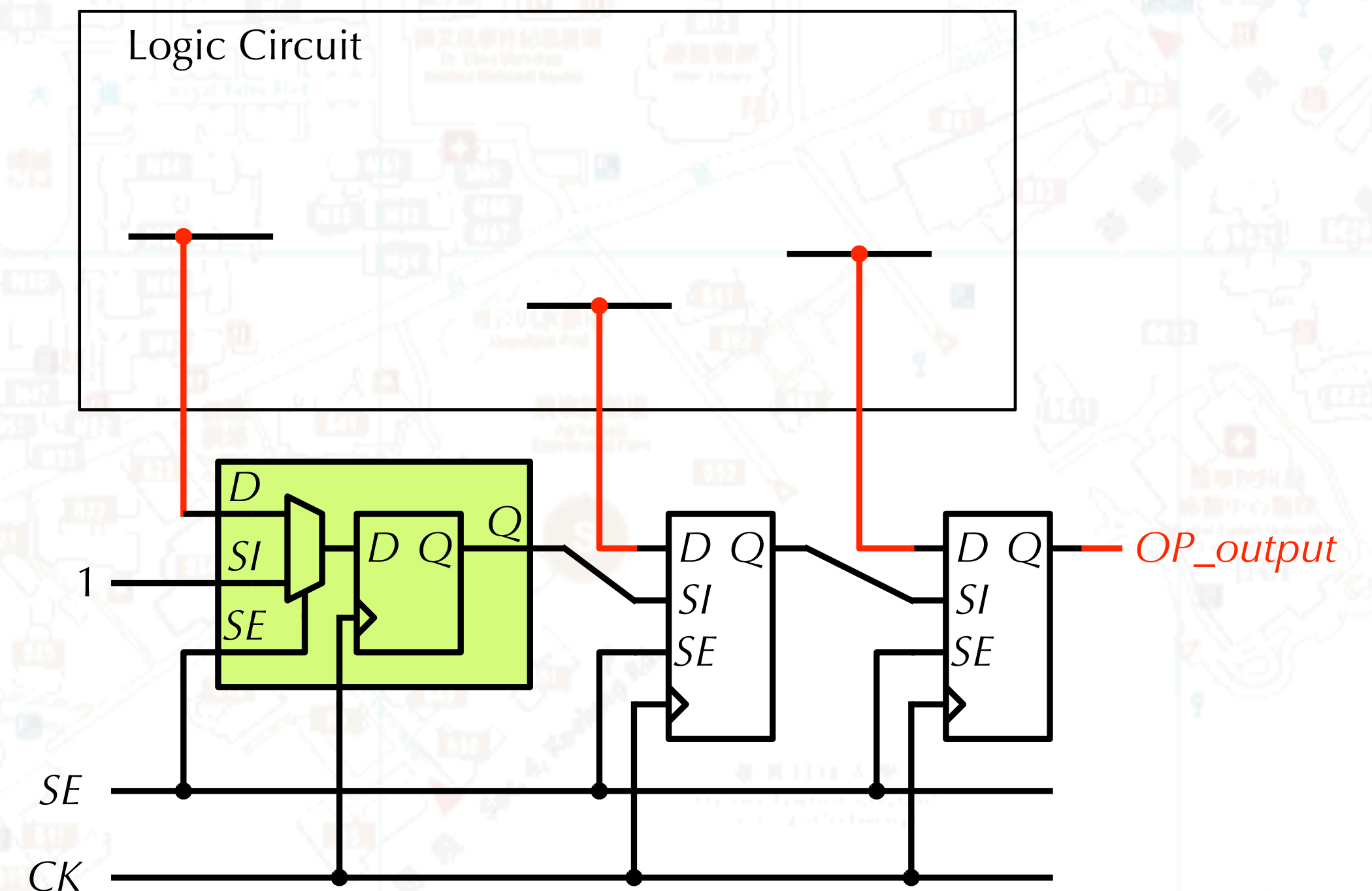
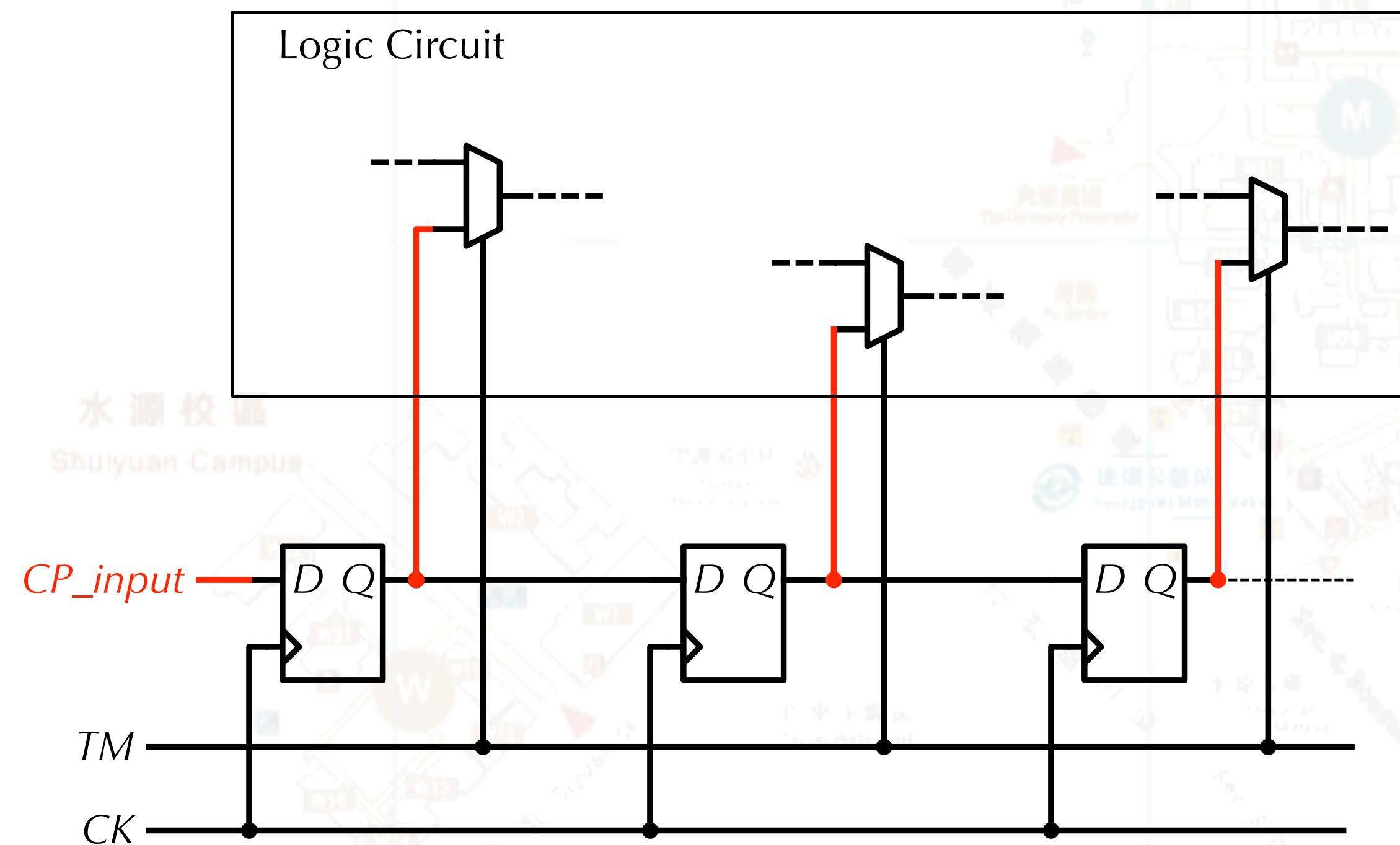
- In practice, a control point may be implemented as a part of one of the gates in the original circuit.

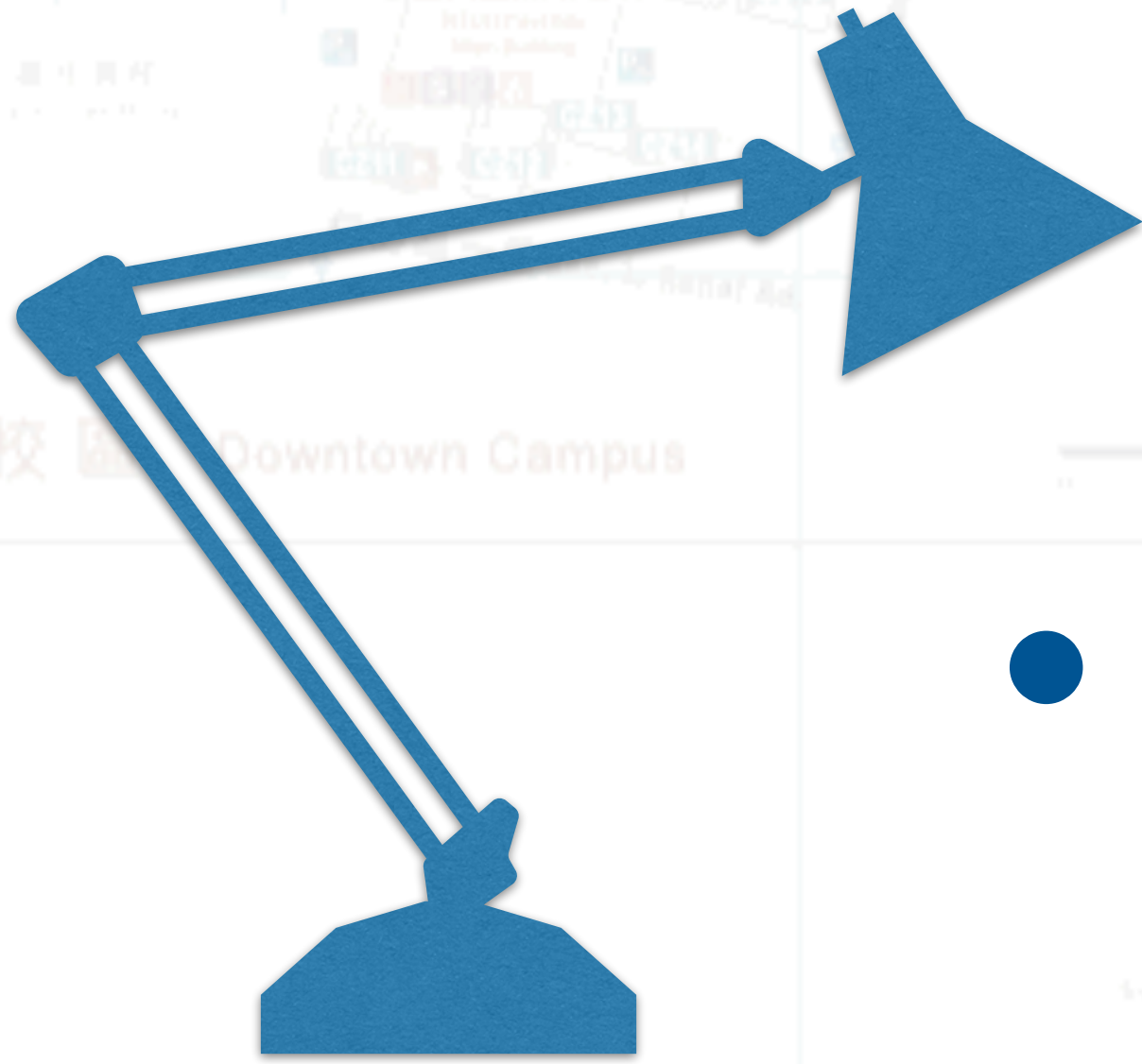


- Observation points: additional primary outputs to enhance observability.

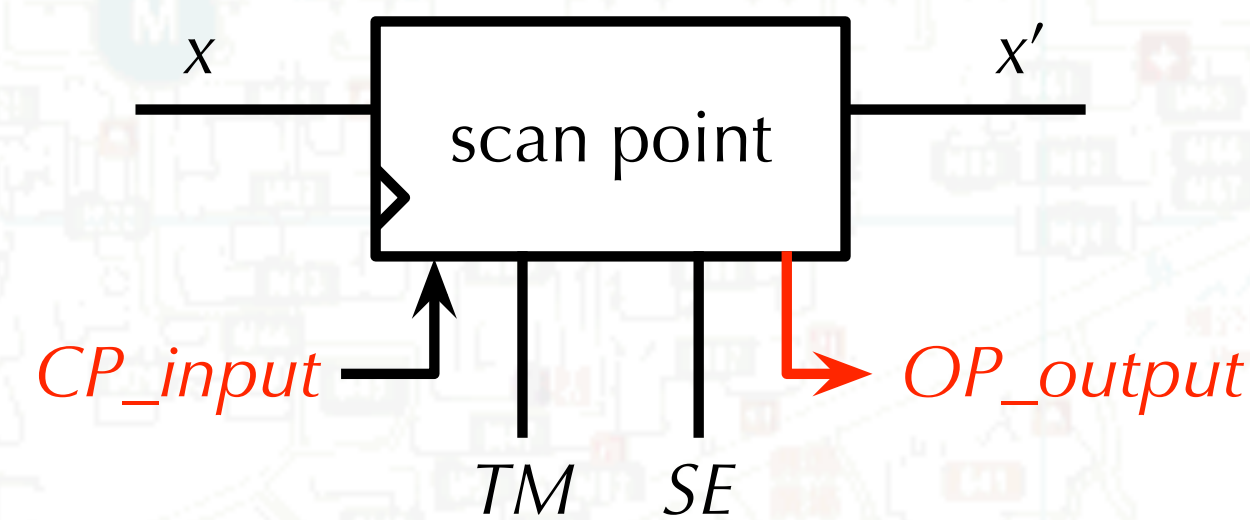


- For pin count reduction, utilize control/observation shift registers.





- Devise a scan point design that allows one to observe a source signal and set it to the desired value simultaneously.
(You may not need both the *TM* and *SE* signals.)





Scan Design

Scan Design — Overview

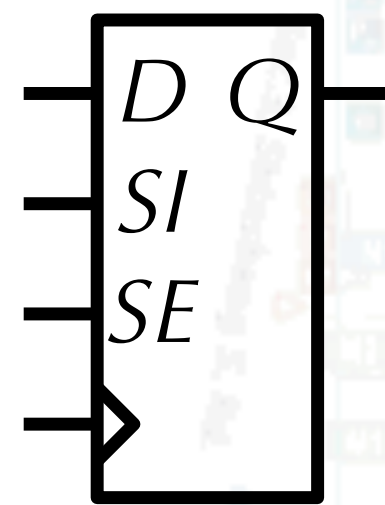
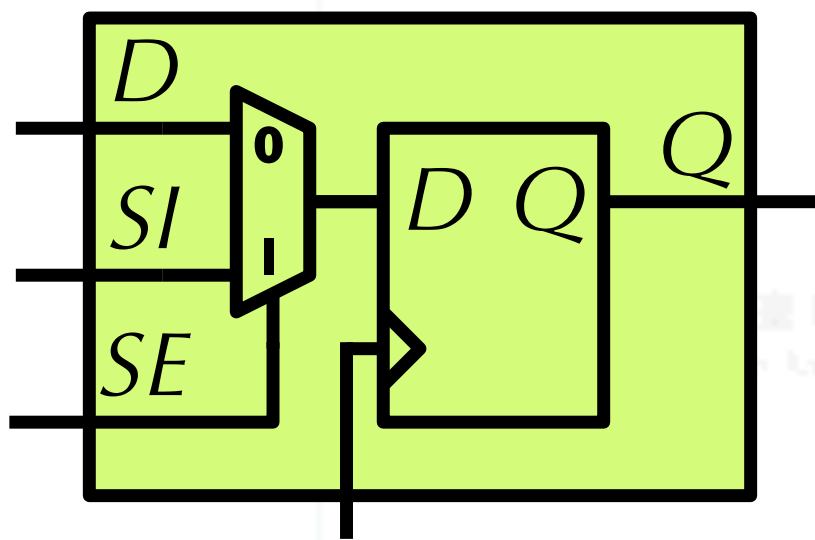


- The most widely used structured DfT methodology.
- Aim to improve the controllability and observability of storage elements in sequential circuits.
- Three operations: normal, shift, and capture.
 - Set $TM = 1$ to turn on test-related fixes and guarantee safe operation.
 - Configure flip-flop operation by setting SE .

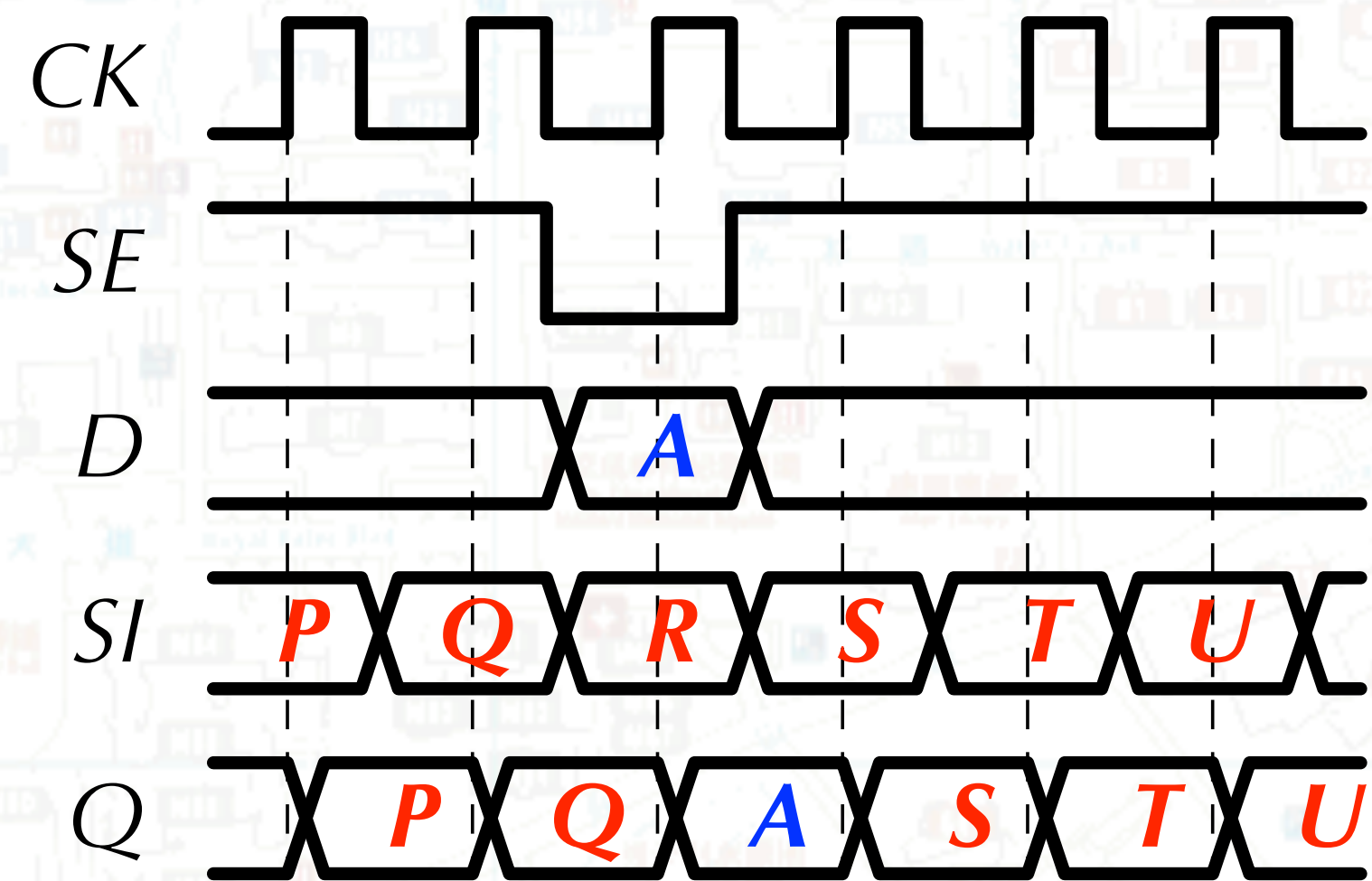
| | TM (test mode) | SE (scan enable) | comments |
|----------------|------------------------------------|--------------------------------------|----------------------------------|
| normal | 0 | 0 | test related circuits turned off |
| shift | 1 | 1 | shift-in and shift-out |
| capture | 1 | 0 | |

Muxed-D Scan Cell

- It is constructed by adding a multiplexer to the D input of a D flip-flop.








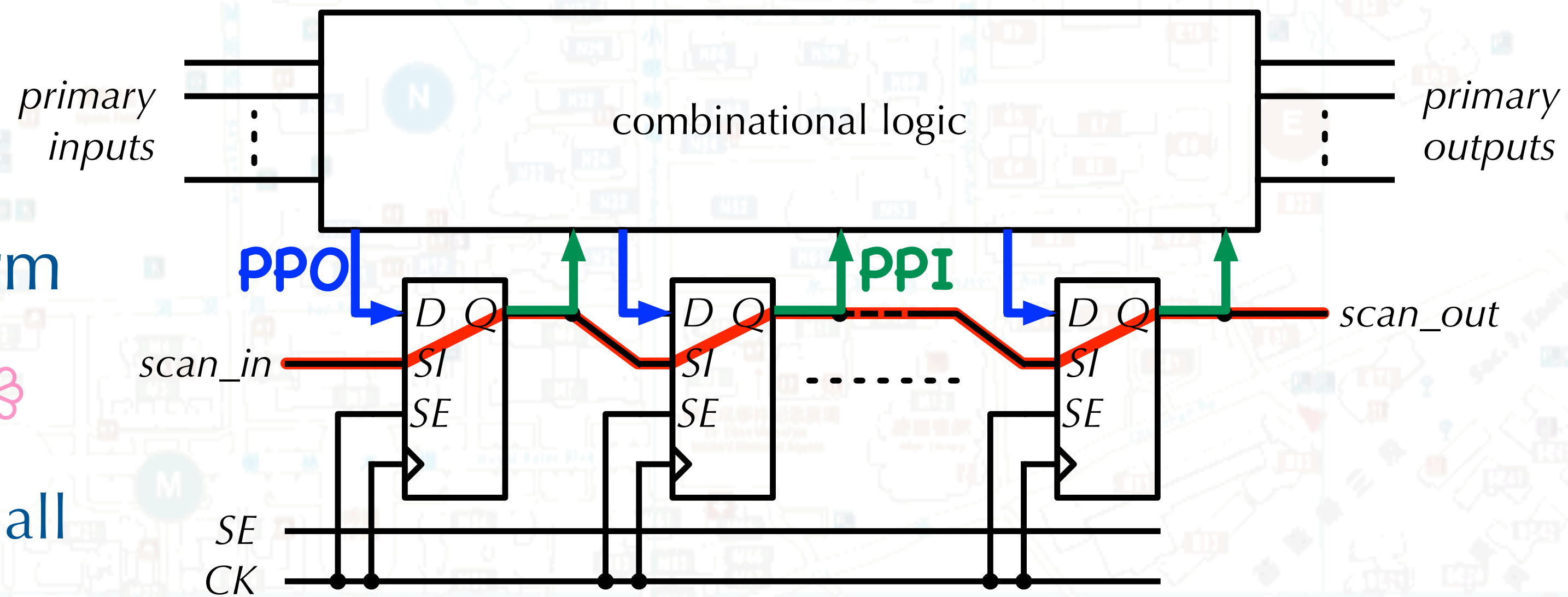
- The SE (scan enable) input selects the DFF input.



- In the normal and capture modes, SE is set to 0 to capture the D input value when triggered.
- In the shift mode, SE is set to 1 to capture the SI input value from when triggered.

Full-Scan Design

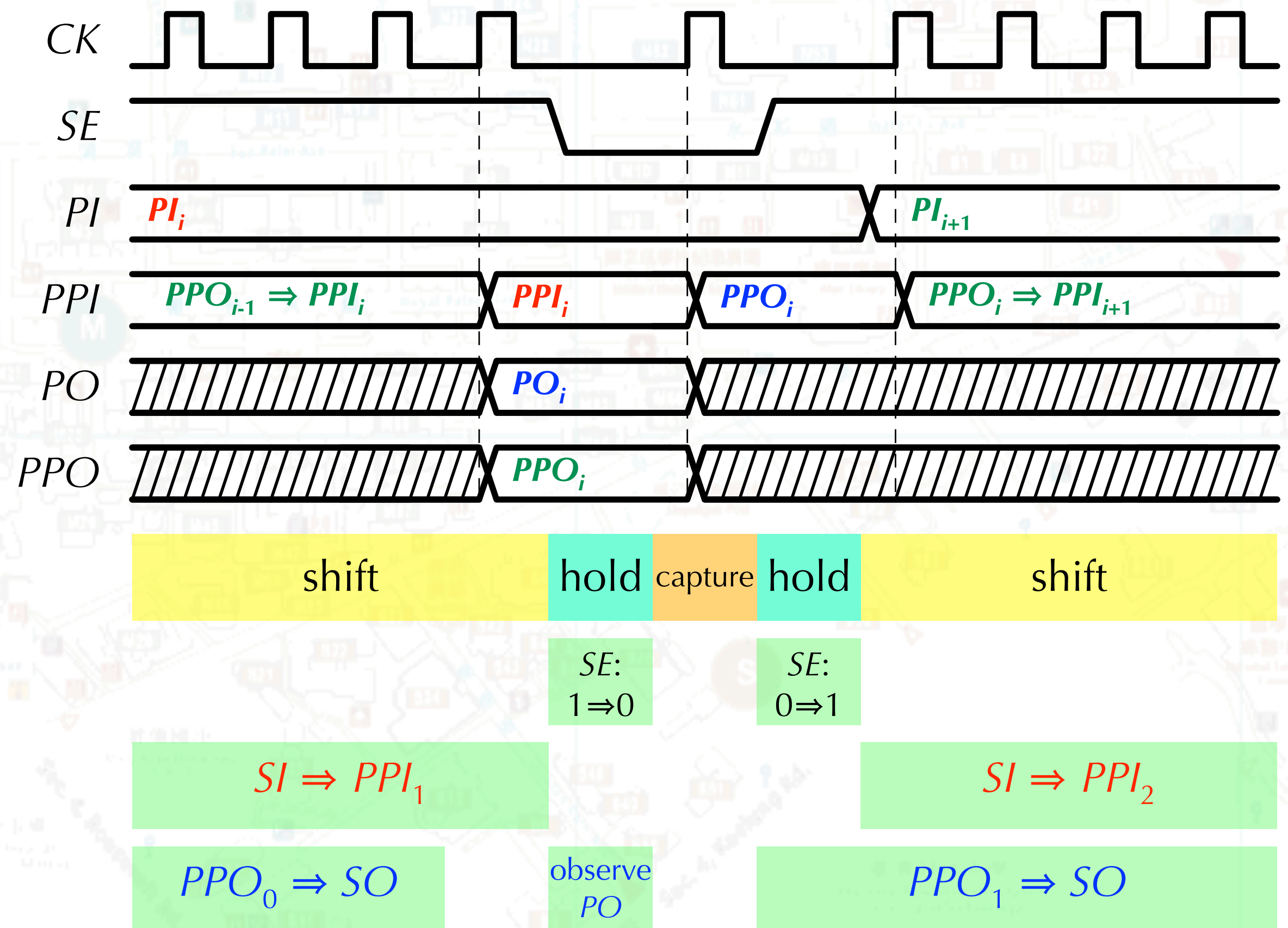
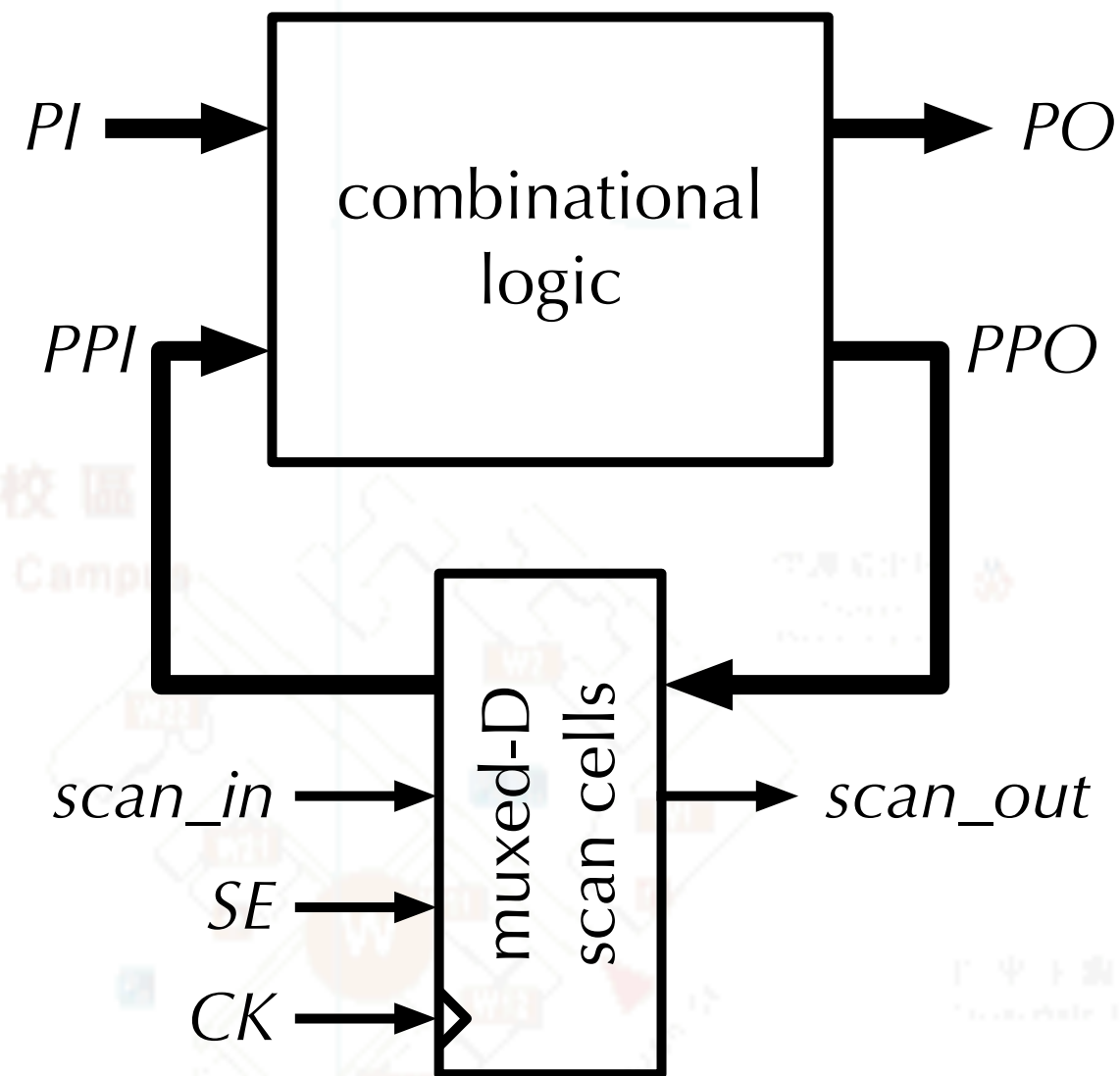
- All storage elements are replaced with scan cells.
- Scan FFs are configured to form one or multiple scan chains. 
- Using the shift operation ($SE = 1$), all flip-flop outputs are controllable. 
- *Pseudo-Primary Inputs (PPIs)* 
- Using the capture operation followed by shift operations, all flip-flop inputs are observable. 
- *Pseudo-Primary Outputs (PPOs)* 



The full-scan design converts the difficult sequential TPG problem to the simpler combinational TPG problem.

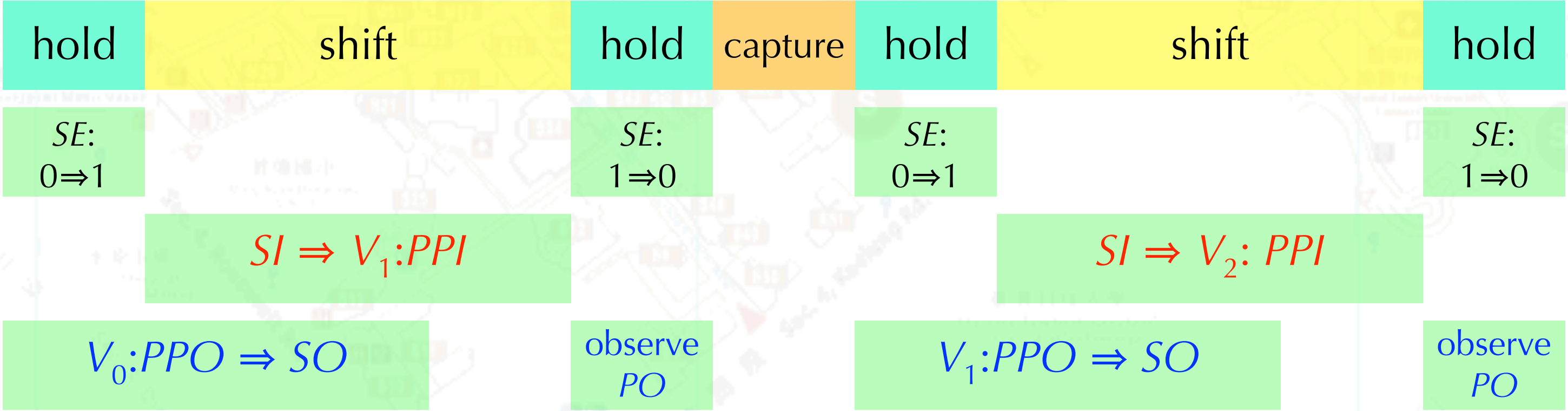
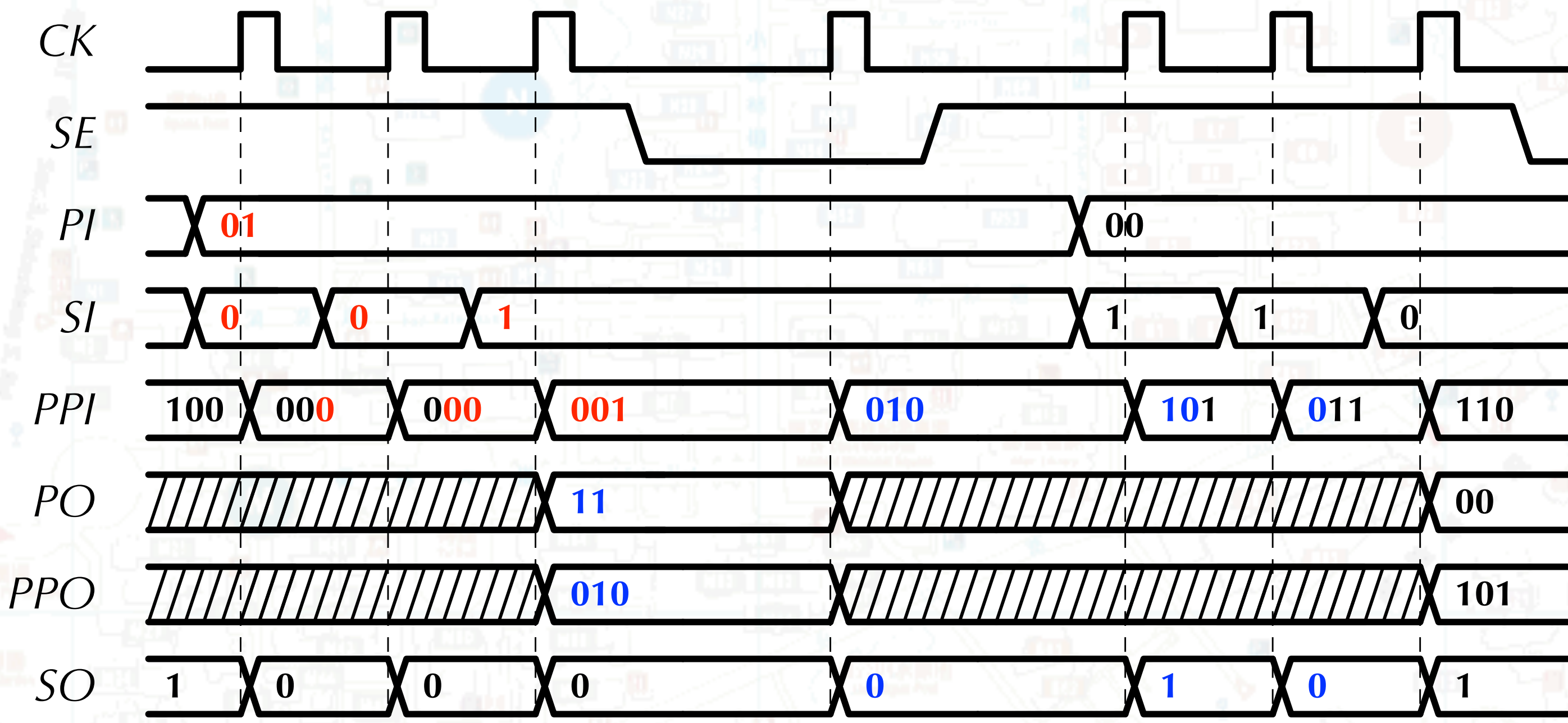
Full-Scan Design Test Application

- A test vector specifies PI and PPI values: $V_i = \langle PI_i, PPI_i \rangle$.
- The expected test response is $R_i = \langle PO_i, PPO_i \rangle$.



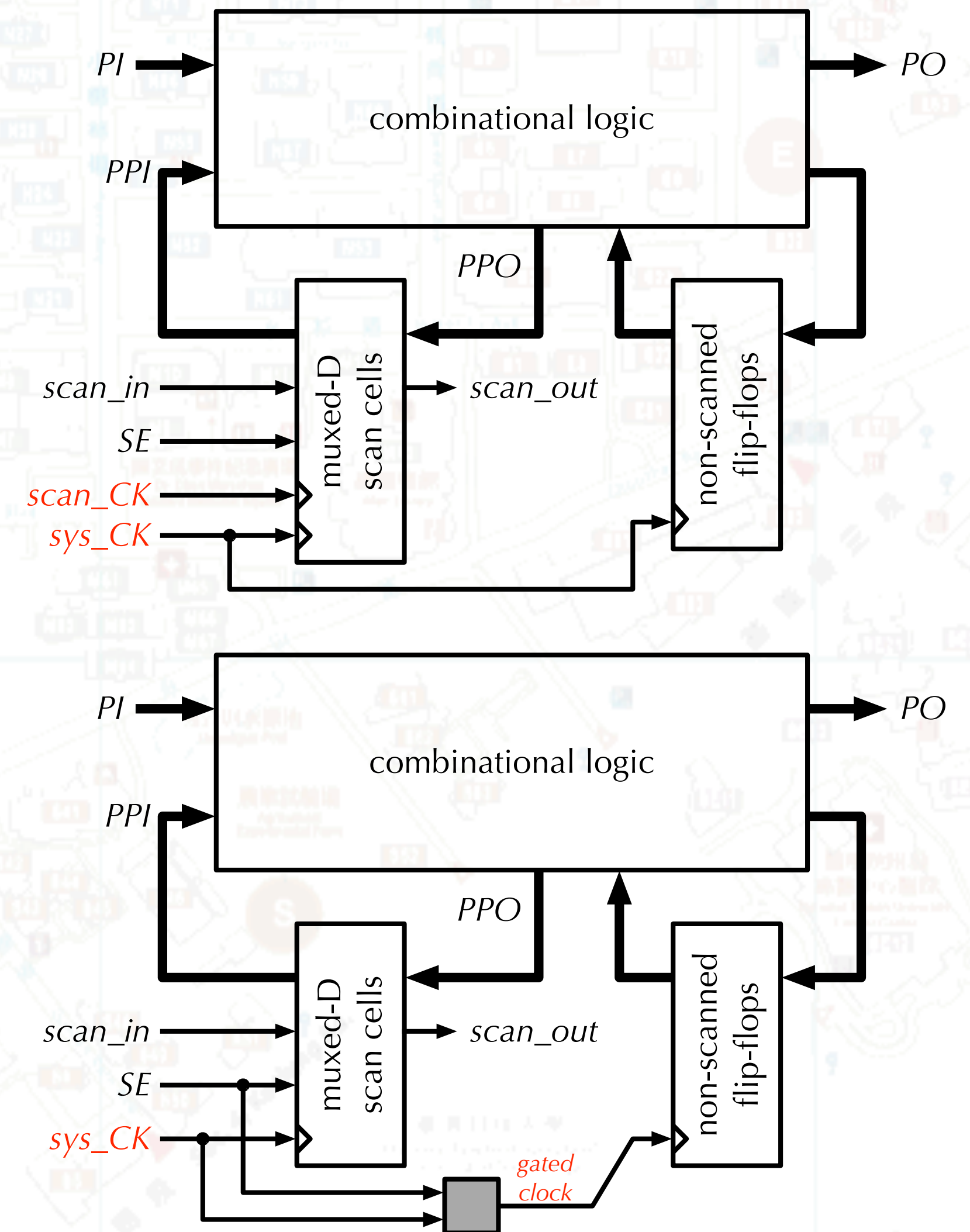
Test Application Example

| | vector (V) | | response (R) | |
|---|------------|-----|--------------|-----|
| | PI | PPI | PO | PPO |
| 1 | - | - | 01 | 100 |
| 2 | 01 | 001 | 11 | 010 |
| 3 | 00 | 110 | 00 | 101 |



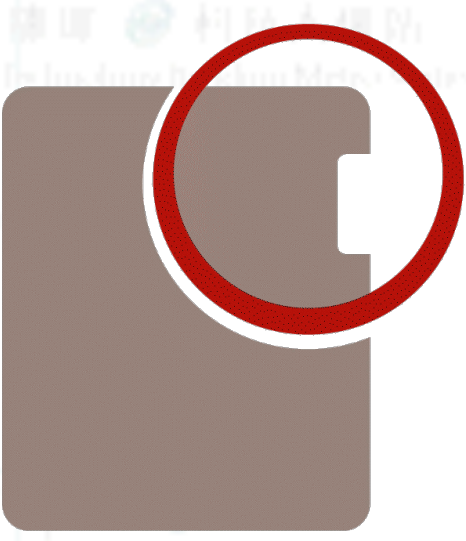
Partial Scan Design

- Replace a subset of flip-flops with scan cells.
- Require sequential TPG to generate patterns to control and observe non-scanned FFs.
- To lower the TPG efforts, disable non-scanned flip-flops during the shift operation.
- Criteria for scan flip-flop selection:
 - Timing and area overhead.
 - TPG complexity: cycle breaking, maximum sequential depth.



Functional Partitioning Based Partial-Scan Design

- Divide the circuit under test into the data path and controller portions.
- Only scan the controller flip-flops.
- Avoid adding extra delay to the timing-critical data path portion.



Pipelined Partial-Scan Design



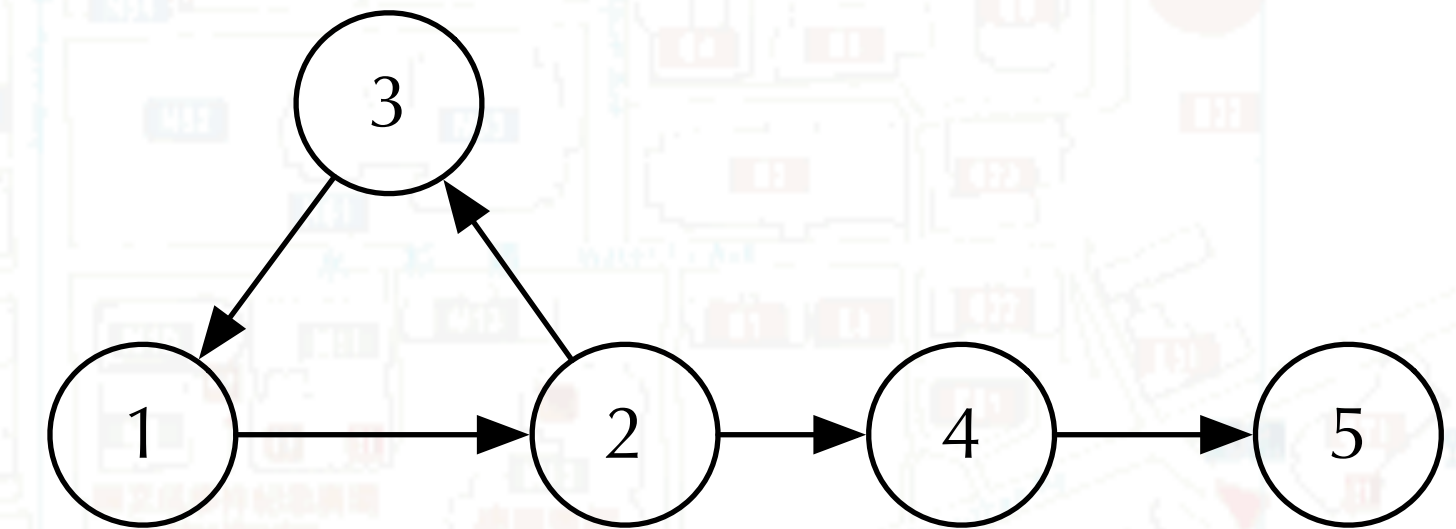
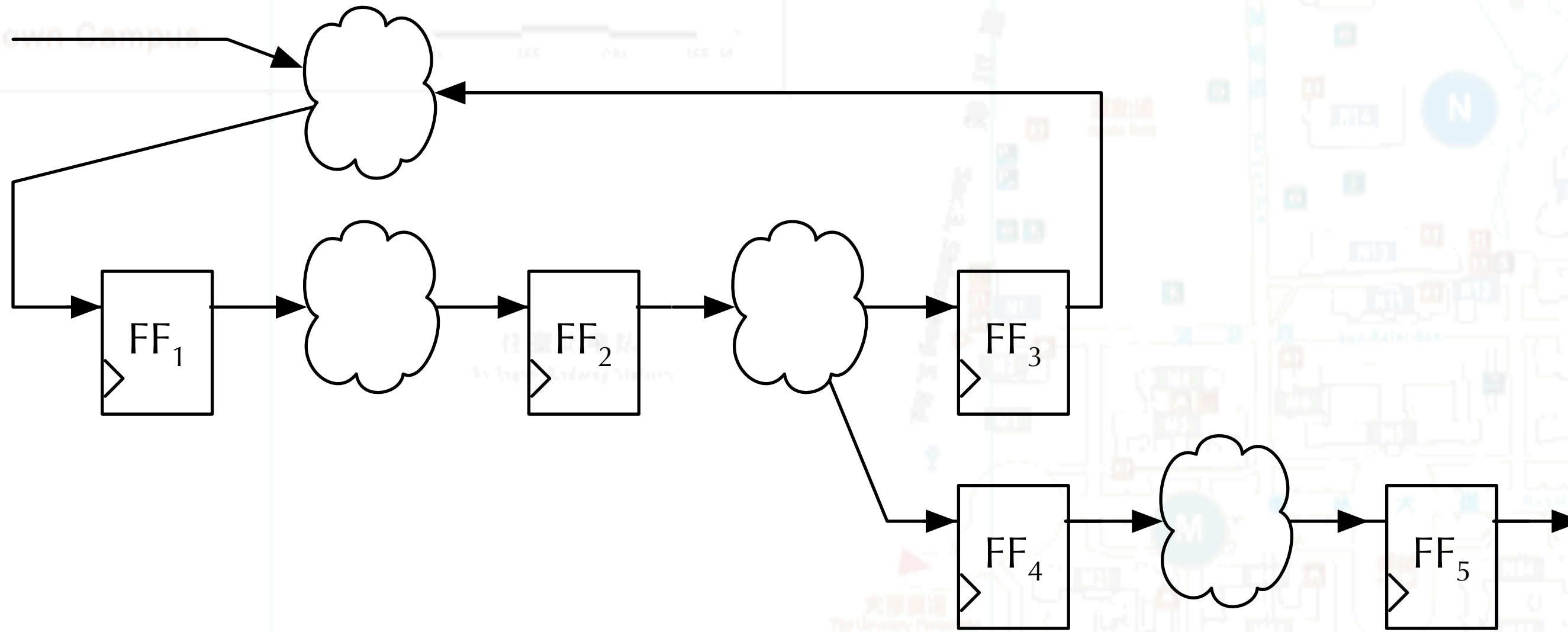
- Select flip-flops to scan in a way such that the partially-scanned circuit is feedback-free.
- Cycle breaking
 1. Construct a structure graph (s-graph).
 - Each node corresponds to a flip-flop; each directed edge indicates a combinational signal path.
 2. Remove vertices until the graph becomes a directed acyclic graph (DAG).
 - The scanned flip-flops.
 - The distance along the longest path is the *sequential depth*.

- TPG complexity is similar to a combinational circuit for a cycle-free partial-scan circuit.

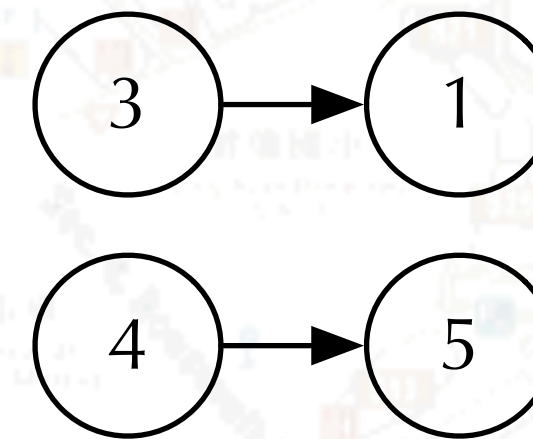
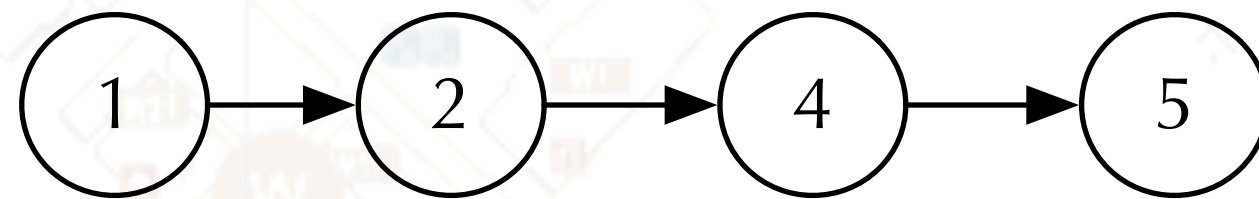


- If the sequential depth is D , any single fault can be tested with at most D vectors.
- One may break only large loops and keep self and small loops to reduce the scan-induced overhead.
- For cyclic s-graph, the test sequence length is about $D \times 2^L$ where L is the maximum length of any cycle.
- Balanced partial-scan design:
Set a limit on the sequential depth, e.g., 3 to 5, to further simplify TPG efforts.

- Example:
The resulting s-graph contains a loop.



- Cycle breaking is possible by scanning FF₃ or FF₂.



Partial-Scan Design Limitations

- Degraded fault coverage.
- Longer TPG time.
- In general, require functional patterns to meet the target fault coverage.
- Less support for debugging, diagnosis, and failure analysis.

