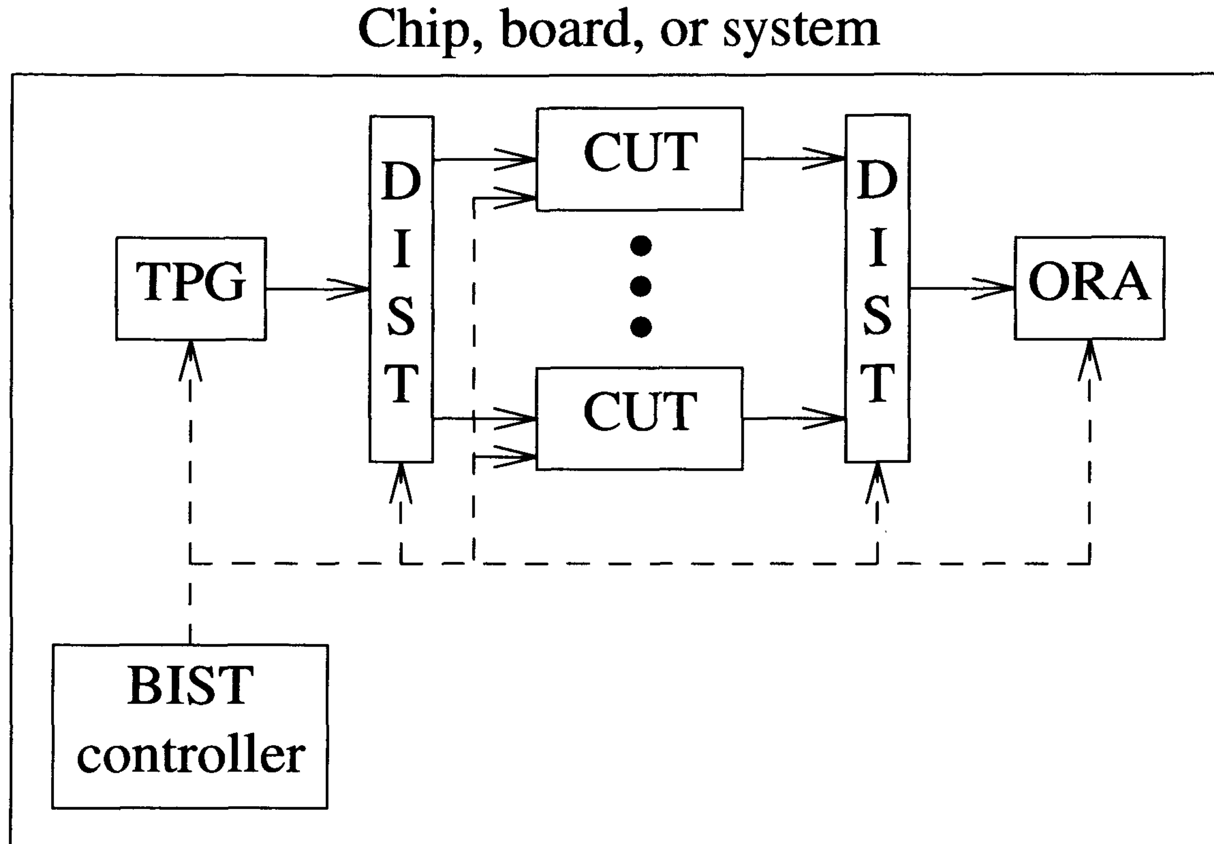
CIS 4930 Digital System Testing
Built-In Self Test (BIST)

Dr Hao Zheng
Comp. Sci. & Eng.
U of South Florida

Introduction

Built-In Self-Test (BIST)

- BIST is the capability of a circuit (chip, board, or system) to test itself



Forms of Built-In Self-Test (BIST)

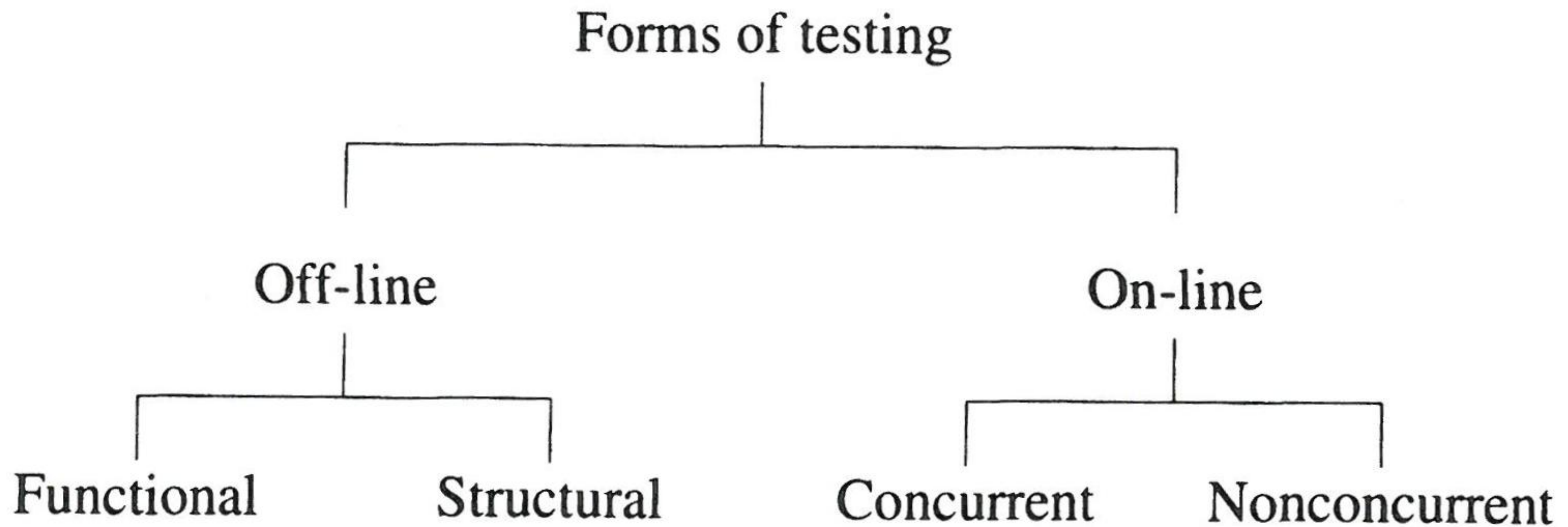


Figure 11.1 Forms of testing

On-line BIST

- Testing occurs during normal functional operating conditions
  - Circuit Under Test (CUT) is *not* put in test mode
- **Concurrent online BIST**
  - Testing occurs *simultaneously with normal functional operation*
- **Non-concurrent online BIST**
  - Testing while system is in **idle state**
  - Executing diagnostic software
  - Test process can be interrupted so that normal operation can resume

Off-line BIST

- Testing a system when the it is not carrying out its normal functions
- Systems, boards, and chips can be tested
- Applicable at the manufacturing, field, depot, and operational stages
- Usually employs test-pattern generators (TPGs) and output response analyzers (ORAs)
- Errors cannot be detected in real time

Off-line BIST – cont'd

- **Functional off-line BIST**
  - Test based on functional description
  - Employs a functional fault model
- **Structural off-line BIST**
  - Explicit structural fault model may be used
  - Fault coverage based on **structural fault detection**
  - Usually tests are generated and responses are compressed

Our discussion is primarily on Structural Off-line BIST

Glossary of key BIST Architectures

BILBO — built-in logic block observer (register)

LFSR — linear feedback shift register

MISR — multiple-input signature register

ORA — (generic) output response analyzer

PRPG — pseudorandom pattern generator, often referred to as a pseudorandom number generator

SISR — single-input signature register

SRSG — shift-register sequence generator; also a single-output PRPG

TPG — (generic) test-pattern generator

Hardcore

- Parts of circuit that must be operational (correct) to execute a self-test
- At a minimum it consists of Power, Ground, and Clock Distribution
- Easy to detect, but hard to diagnose
  - Faults may be in CUT or hardcore
- Usually tested by external test equipment
- Designer attempts to minimize complexity of hardware

Levels of Test

- **Production Test**
  - Newly manufactured components
  - Performed at Chip, Board, System levels
  - Reduces the need for expensive ATE (Automated Test Equipment)
- **Field Testing**
  - Eliminates the need for expensive special test equipment.
  - Improve maintainability,
  - Reduce life-cycle costs.

Test-Pattern Generation for BIST

Test Pattern Generation for BIST

Assume CUT = *n*-input, *m*-output combinational circuit

- **Exhaustive Testing**
  - Exhaustive test-pattern generators – expensive
- **Pseudo-random Testing**
  - Weighted test generator
  - Adaptive test generator
- **Pseudo-exhaustive testing (cf. 8.3)**
  - Syndrome driver counter
  - Constant-weight counter
  - Combined LFSR and shift register
  - Combined LFSR and XOR gates
  - Condensed LFSR
  - Cyclic LFSR

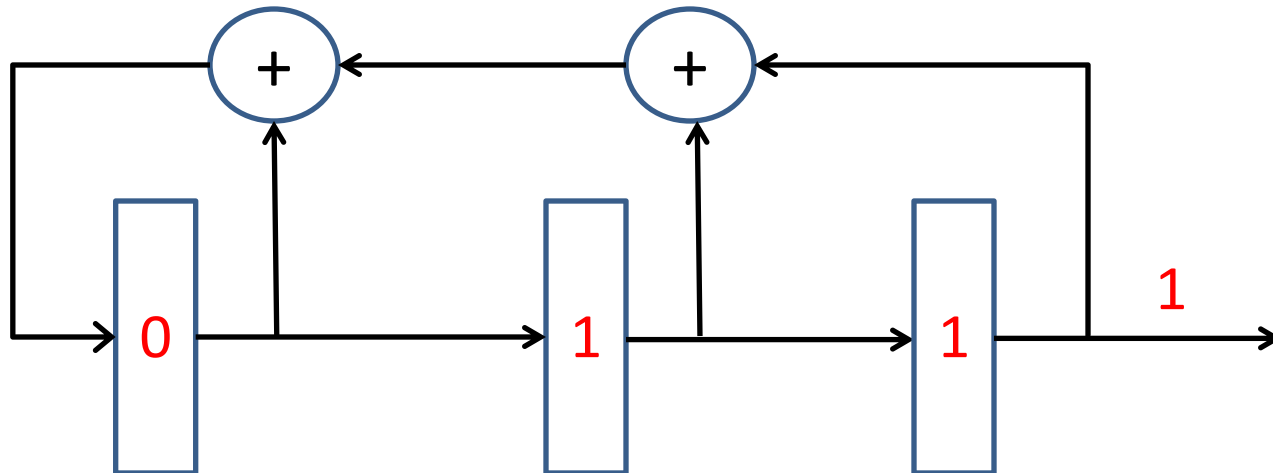
Linear Feedback Shift Register (LFSR)

- LFSRs used for pseudo-random test vector generation and signature analysis

State		
S0	1	0
S1	0	1
S2 = S0	1	0
:		
:		

- Two states
- Cycles through states 01 and 10 repeatedly
- No inputs except clock

XOR Gates in Feedback



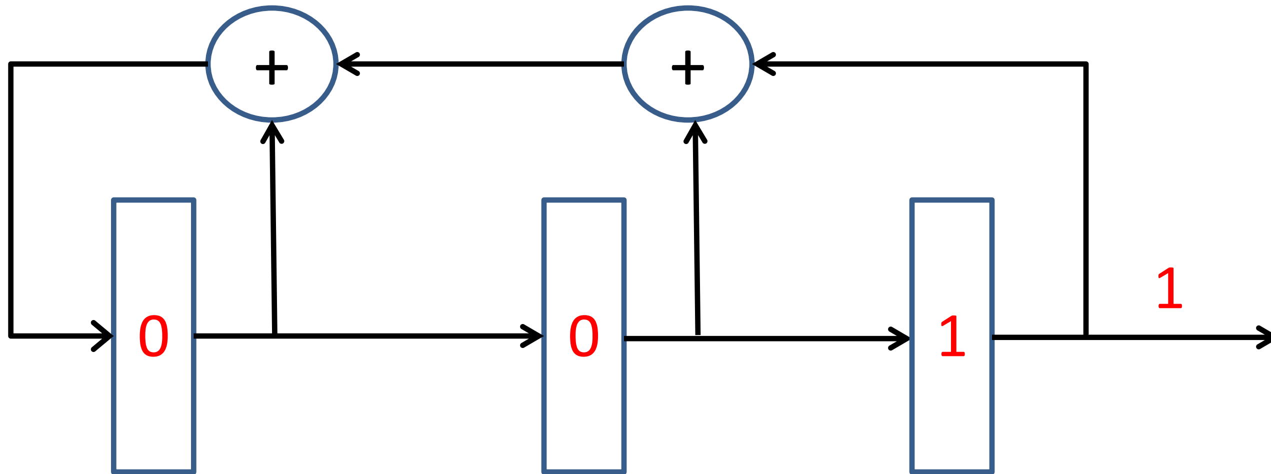
S0	0	1	1
S1	0	0	1
S2	1	0	0
S3	1	1	0
S4 = S0	0	1	1

:

:

Cycles through 4 states

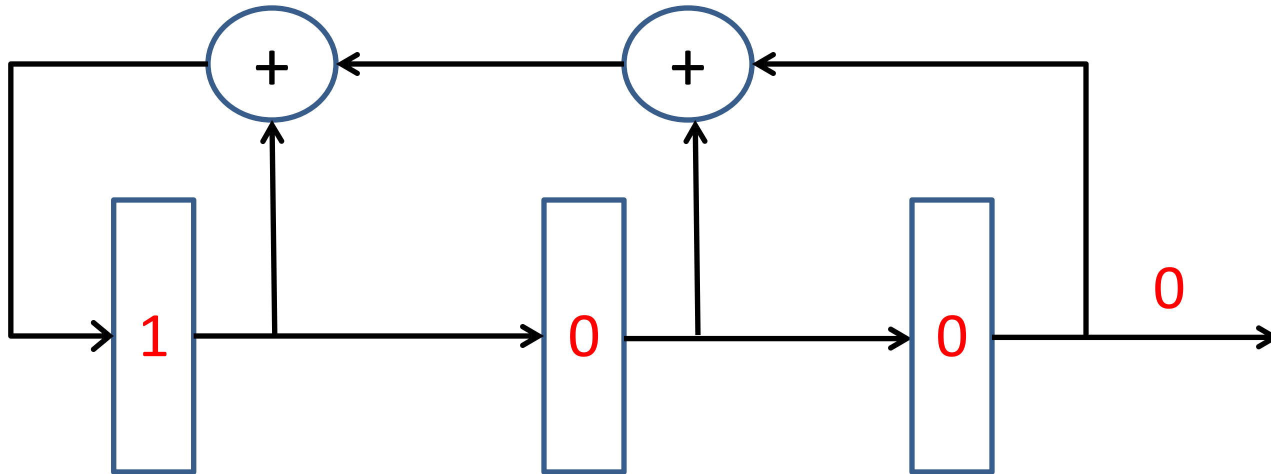
XOR Gates in Feedback



S0	0	1	1
S1	0	0	1
S2	1	0	0
S3	1	1	0
S4 = S0	0	1	1
:			
:			

Cycles through 4 states

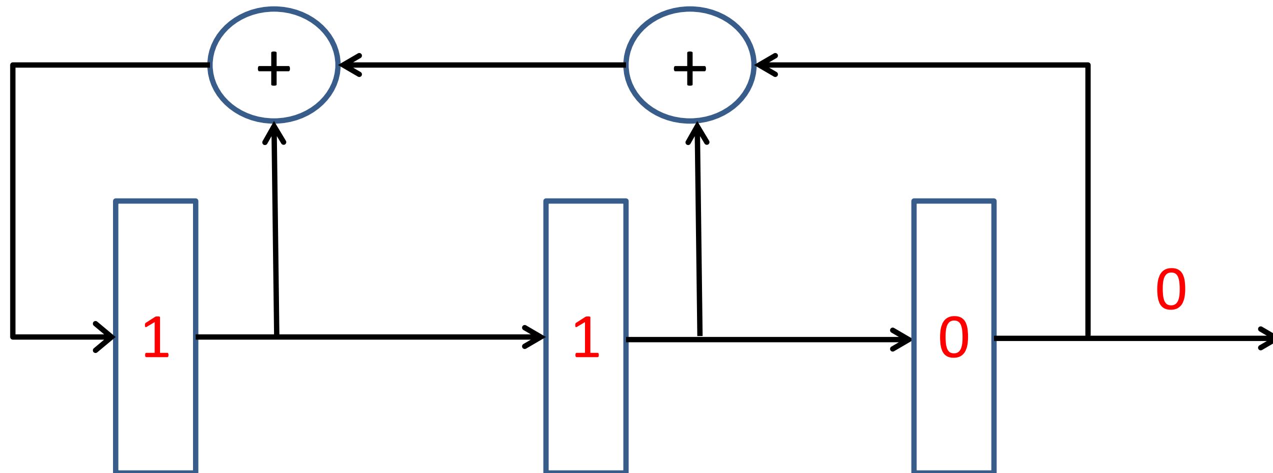
XOR Gates in Feedback



S0	0	1	1
S1	0	0	1
S2	1	0	0
S3	1	1	0
S4 = S0	0	1	1
:			
:			

Cycles through 4 states

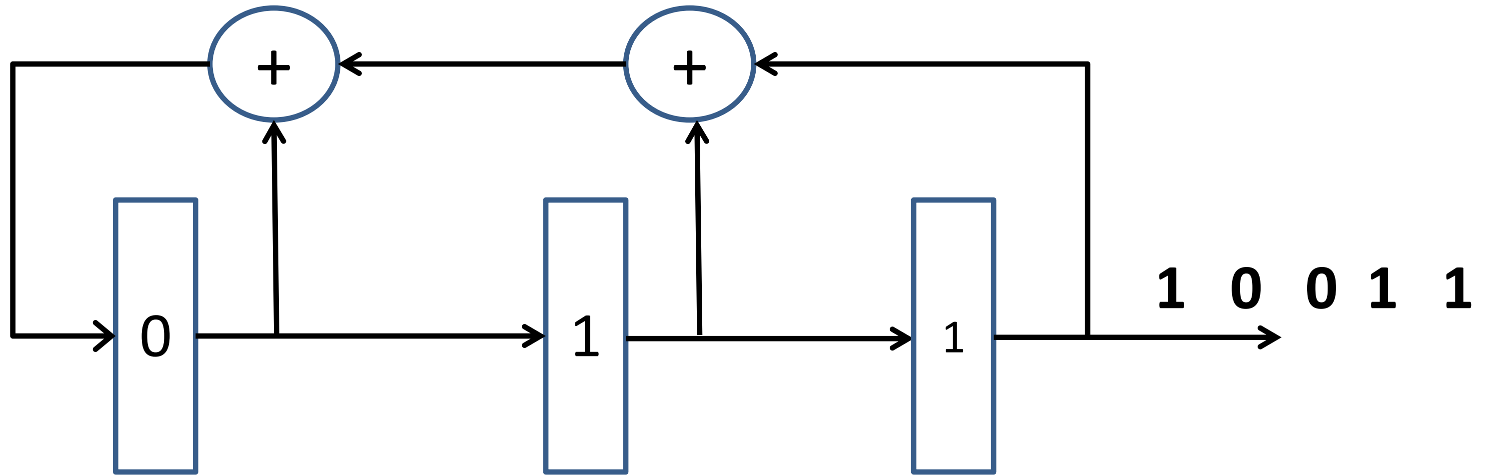
XOR Gates in Feedback



S0	0	1	1
S1	0	0	1
S2	1	0	0
S3	1	1	0
S4 = S0	0	1	1
:			
:			

Cycles through 4 states

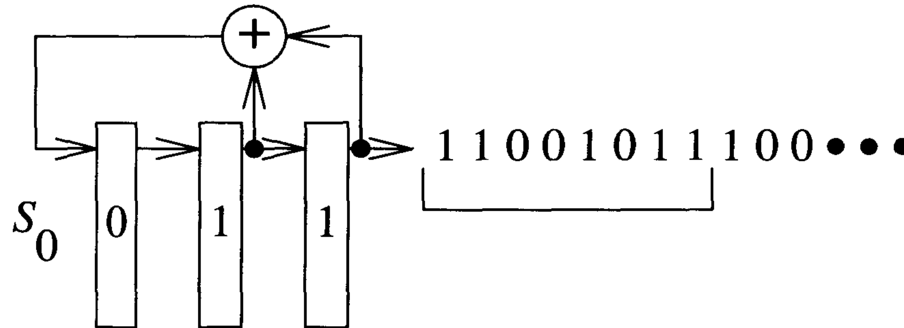
XOR Gates in Feedback



S0	0	1	1
S1	0	0	1
S2	1	0	0
S3	1	1	0
S4 = S0	0	1	1
:			
:			

Cycles through 4 states (S0–S3)

Maximal Length LFSR



S_1	0	0	1
S_2	1	0	0
S_3	0	1	0
S_4	1	0	1
S_5	1	1	0
S_6	1	1	1
$S_7 = S_0$	0	1	1
⋮			

Generates a cyclic sequence of length $2^n - 1$

All-0 initial state leads to a sequence of length 1.

Exhaustive Testing

- Test the *n*-input comb. circuit with 2^n inputs
- Binary counter can be used as TPG.
- Autonomous LFSR can also be used.
- Guarantees that all detectable faults that *do not* introduce sequential behavior will be detected
  - i.e. no bridging faults.
- Depending on clock rate, n > 22 is impractical
- Not used for sequential circuits

Pseudo-Random Testing

- Many characteristics of random patterns
- Generated deterministically => Repeatable
- With or without replacement
- **With replacement** = patterns can repeat
- **Without replacement** = unique patterns (autonomous LFSR can be a source)
- Applicable to both comb. and seq. circuits

Bias in Pattern Generation

- Autonomous LFSR: 0's and 1's balanced in the output
- Sometimes we want a bias (say more 1's than 0's)
- Example: 4-input AND gate
  - Probability of an input set to 0 is 15/16
  - With random inputs, hard to test other input s-a-0 or s-a-1 fault

Weighted & Adaptive Test Generation

- **Weighted Test Generator**
  - Distribution of 0s an 1s -> not uniform
  - Can be constructed by LFSR + a combl. Circuit
  - When testing a circuit using WTG, preprocessing is carried out to determine weights
  - Therefore, each part of circuit can be tested with different distributions
- **Adaptive Test Generator**
  - Uses a WTG
  - Results of fault simulation used to modify weights
  - Efficient in terms of test length
  - Requires complex TPG hardware

Pseudo-Exhaustive Testing

- Achieves benefits of exhaustive testing *but with far fewer test patterns*
- Relies on circuit segmentation
- A segment = subcircuit of the CUT
- Attempts testing each segment exhaustively
- Segments need not be disjoint
- Forms of Segmentation
  1. Logical Segmentation
     a. Cone Segmentation
     b. Sensitized Path Segmentation
  2. Physical Segmentation

Cone Segmentation

- Cone segmentation of a *m* output circuit is logically segmented into *m* cones
- Cone = all logic associated with one output
- Each cone tested exhaustively
- All cones tested concurrently

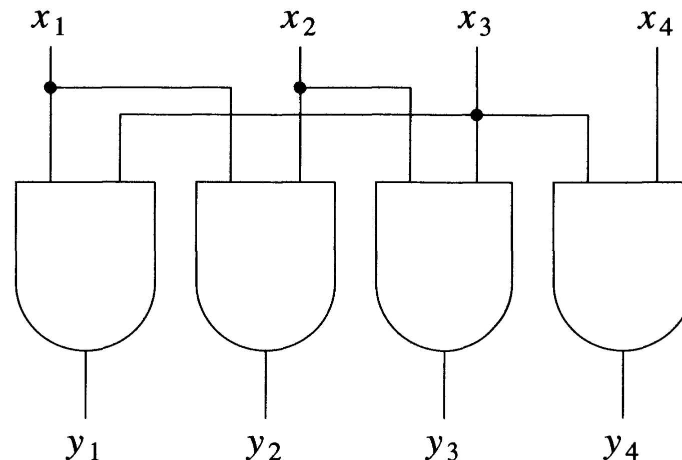
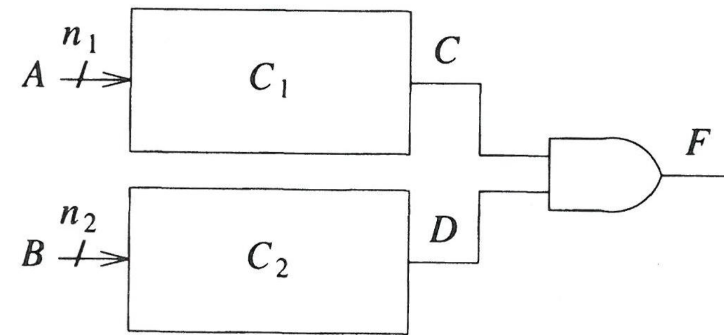


Figure 11.3 A (4,2)-CUT

Sensitized Path Segmentation

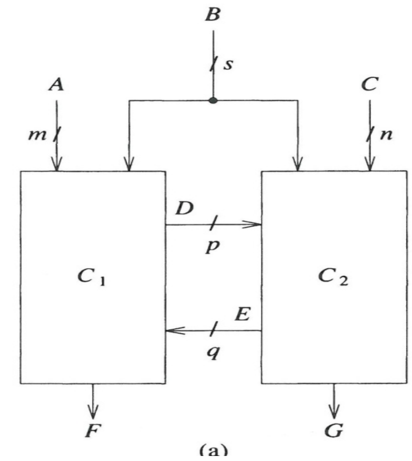
- Example:
  - C partitioned into C_1 and C_2
  - Set inputs to B such that D=1 and apply 2^{n1} patterns to test C_1
  - Similarly test C_2
  - Need $2^{n1} + 2^{n2} + 1$ patterns instead of $2^{n1 + n2}$



Physical Segmentation

- In large circuits, pseudo-exhaustive testing leads to large test sets
- Can employ physical segmentation
  - Partitioning: Circuit is divided into sub-circuits
  - Bypass Storage Cell
    - Normal mode: acts as a wire
    - Test mode: part of an LFSR

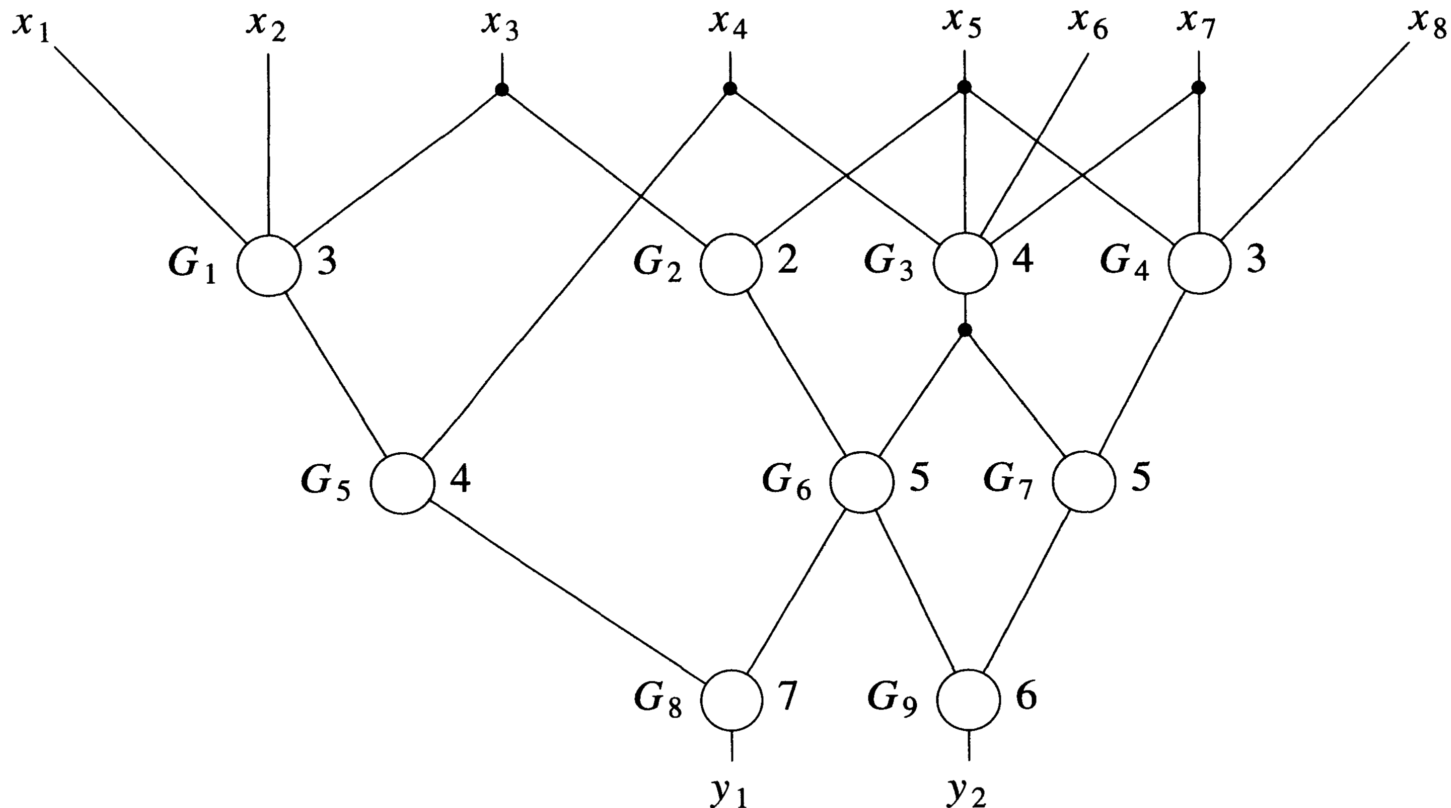
Physical Segmentation by Partitioning

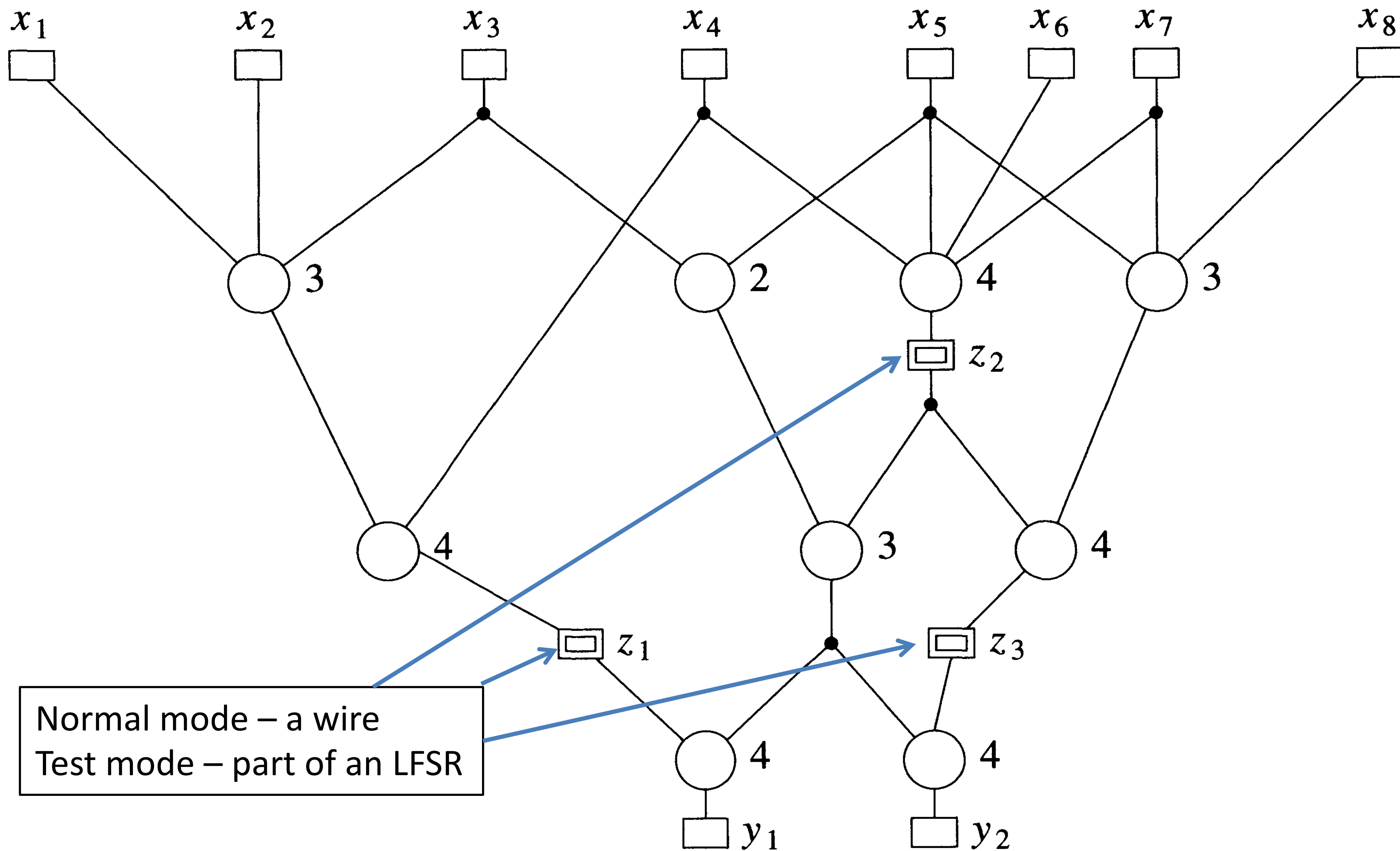


T_1	T_2	Mode
0	0	normal
0	1	test C_1
1	0	test C_2

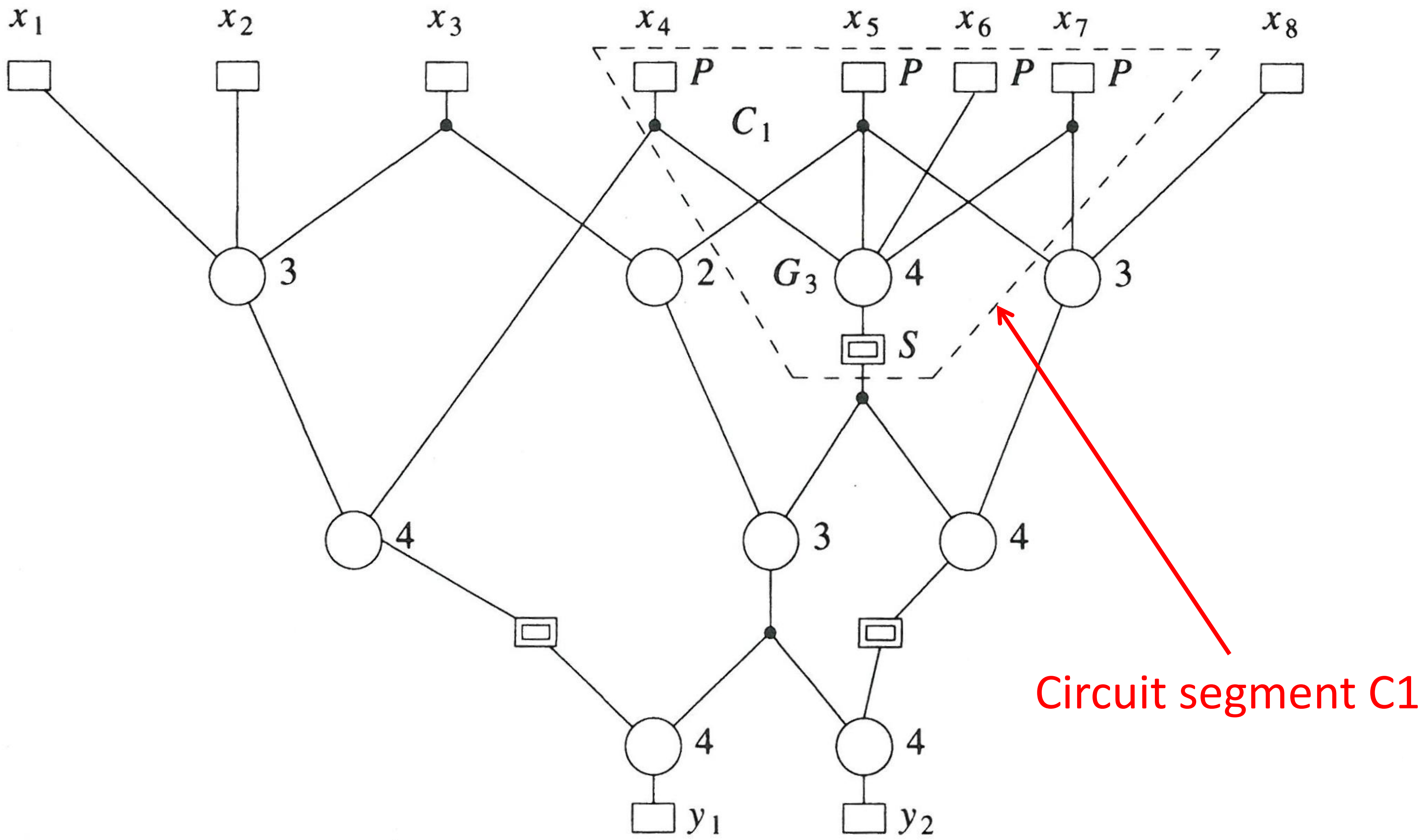
Physical Segmentation by Storage Cells

- Let us say we want to segment the following such that no signal is a function of more than 4 variables

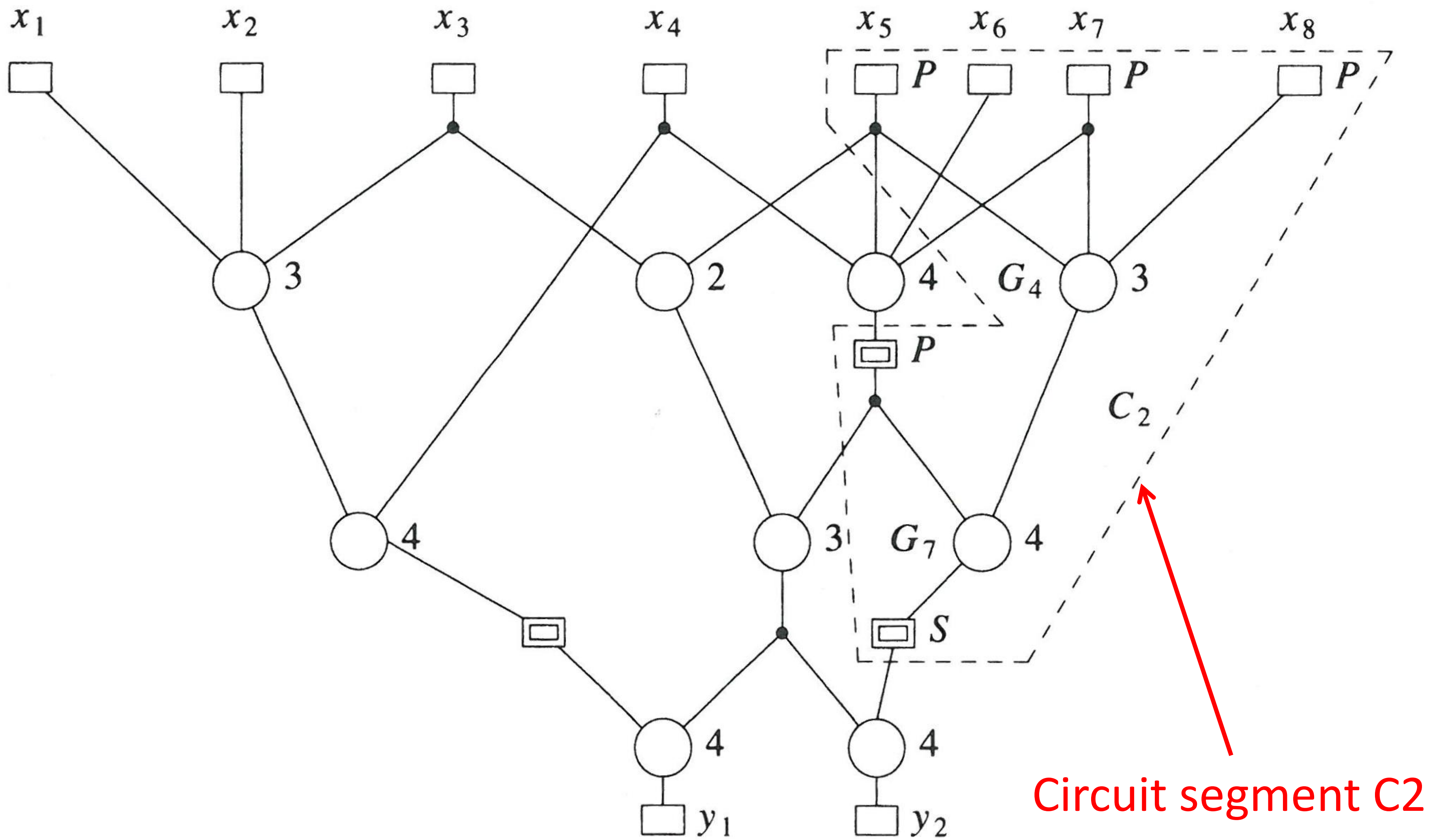




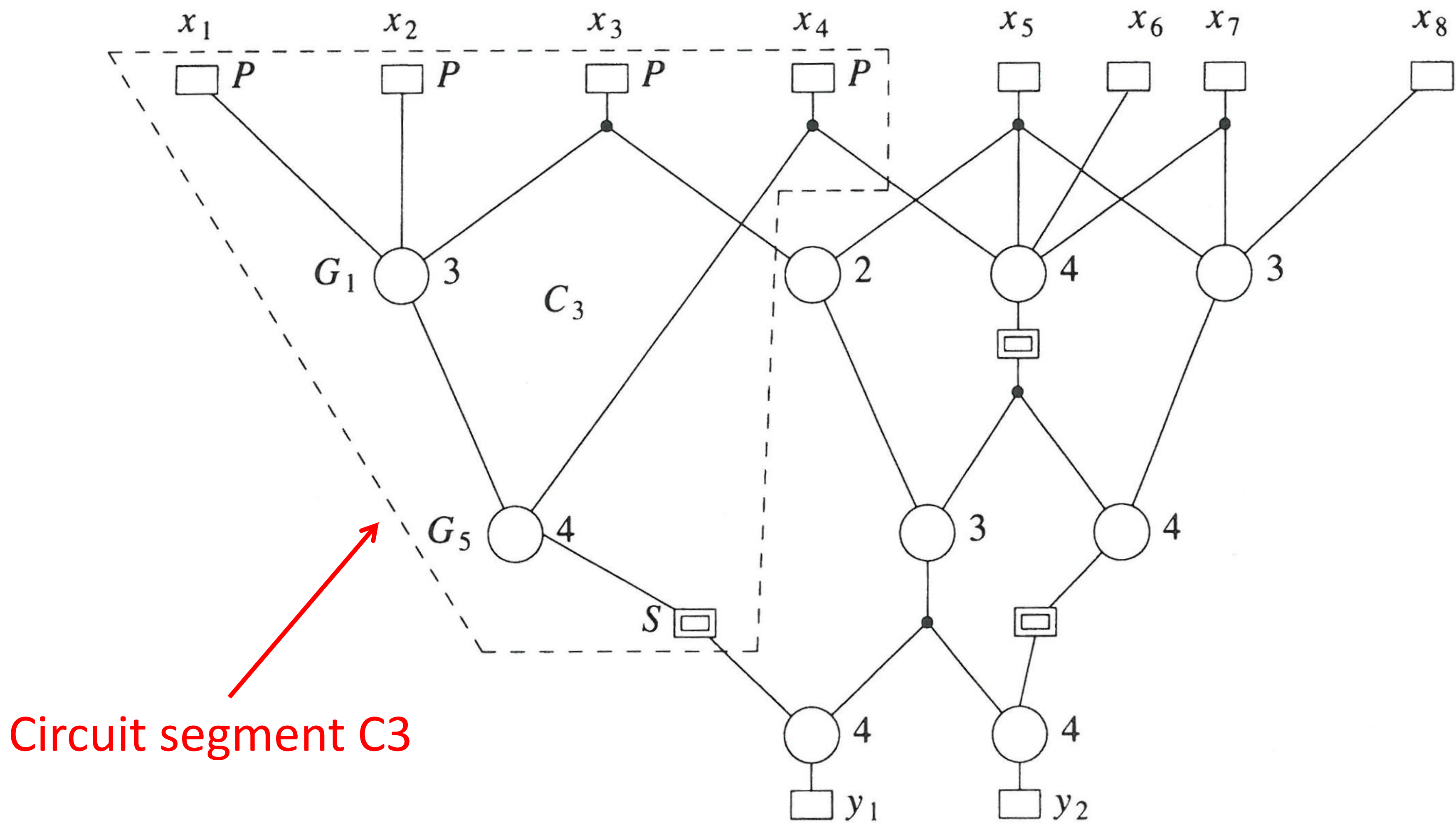
x_1
x_2
x_3
x_4
x_5
x_6
x_7
x_8
3
2
4
3
z_2
4
3
4
z_1
z_3
4
4
y_1
y_2
Normal mode – a wire
Test mode – part of an LFSR



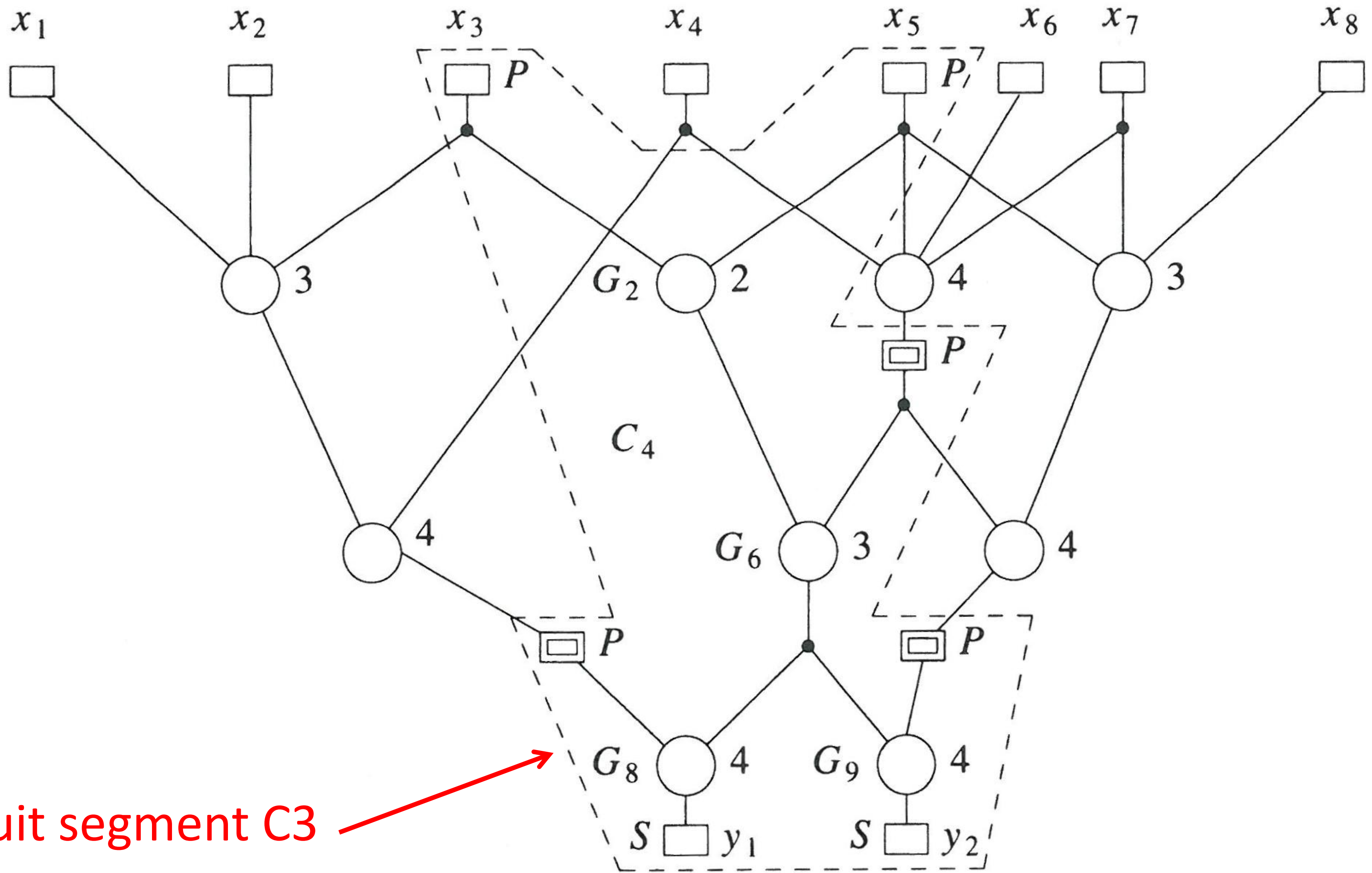
(a)



(b)

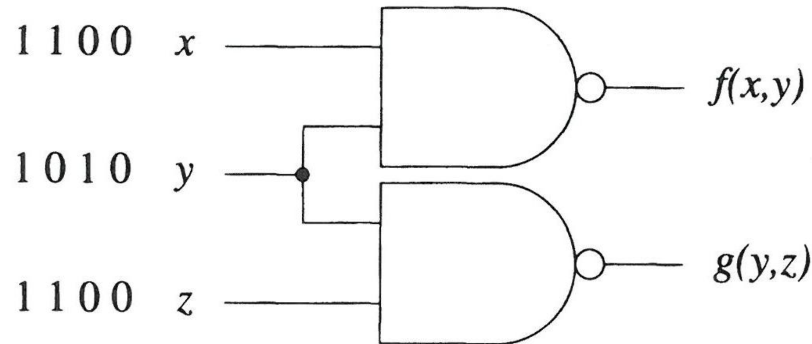


(c)



(d)

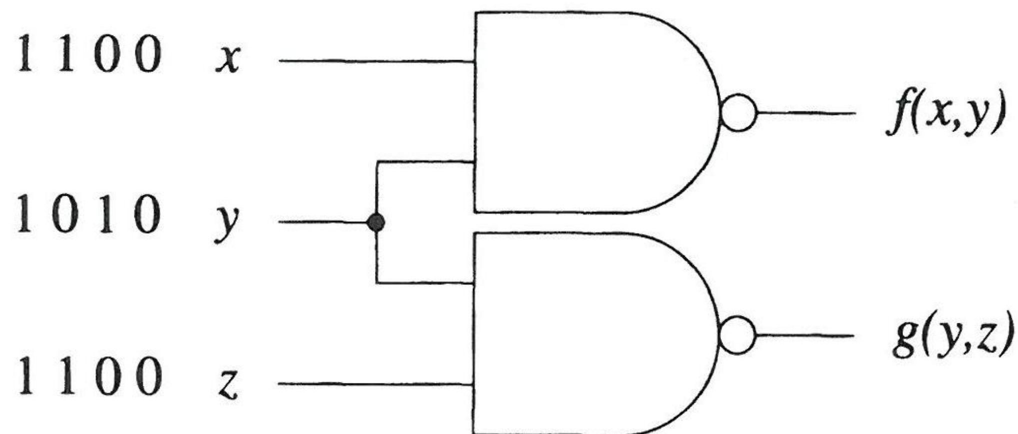
Identification of Test Signal Inputs



- *f* and *g* are functions of only two inputs each
- To exhaustively test the multiple function (*f, g*), we need 8 vectors
- Since no output is function of both x and z, same test data can be applied to both these lines
  - 2 test signals
  - 4 test vectors are sufficient

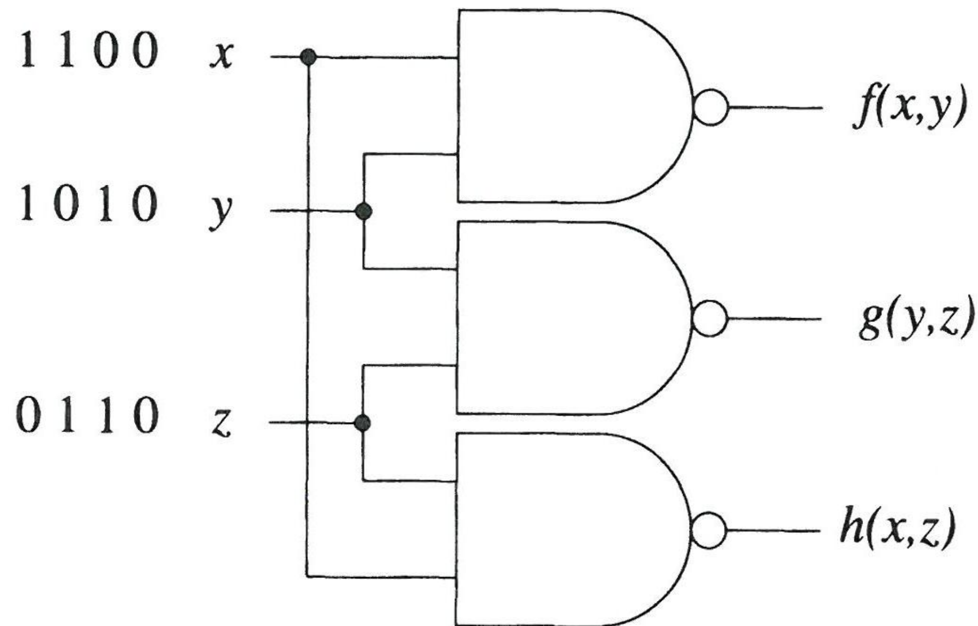
Maximal-Test-Concurrency (MTC) circuit

- A circuit is said to be a *maximal-test-concurrency* (MTC) circuit, if the minimal number of required test signals is equal to the maximum number of inputs upon which any output depends.



Non-MTC circuit

- All three signals are required, can still be tested exhaustively by just four test patterns



ıre 11.6 A nonmaximal-test-concurrency circuit with verification test inpu

TPG – Syndrome-Driver Counter

- If (*n*-*p*) input share test signals with *p* other inputs, at most 2^p tests are required.
  - *n*: # of inputs
- $n = 4$, p = 3, w=2
  - w = # of inputs of a segment
- At most 8 tests are needed.
- 0000 & 1111 are not needed

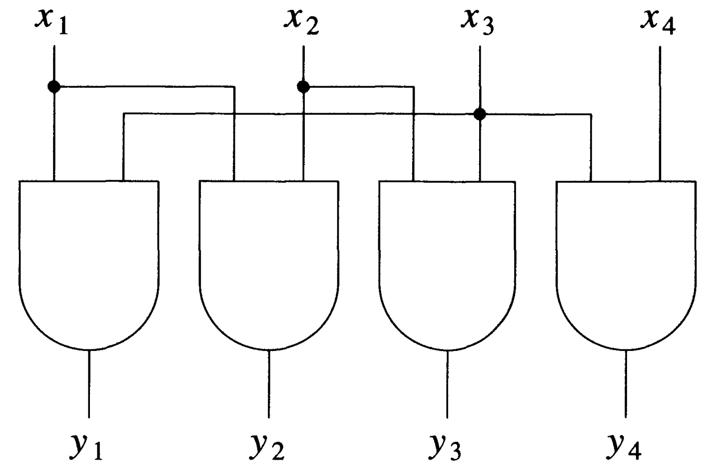


Figure 11.3 A (4,2)-CUT

TPG – Constant-Weight Counter

- A (n,w) circuit can be tested by a counter implementing by *w-out-of-K*
- Complexity of the counter can be high for large w

1100
1010
1001
o110
010 1
o011

TPG – Combined LFSR/SR

- (n, w) circuit
- Lower cost
- May generate more tests
- # of tests near minimal when w << n/2

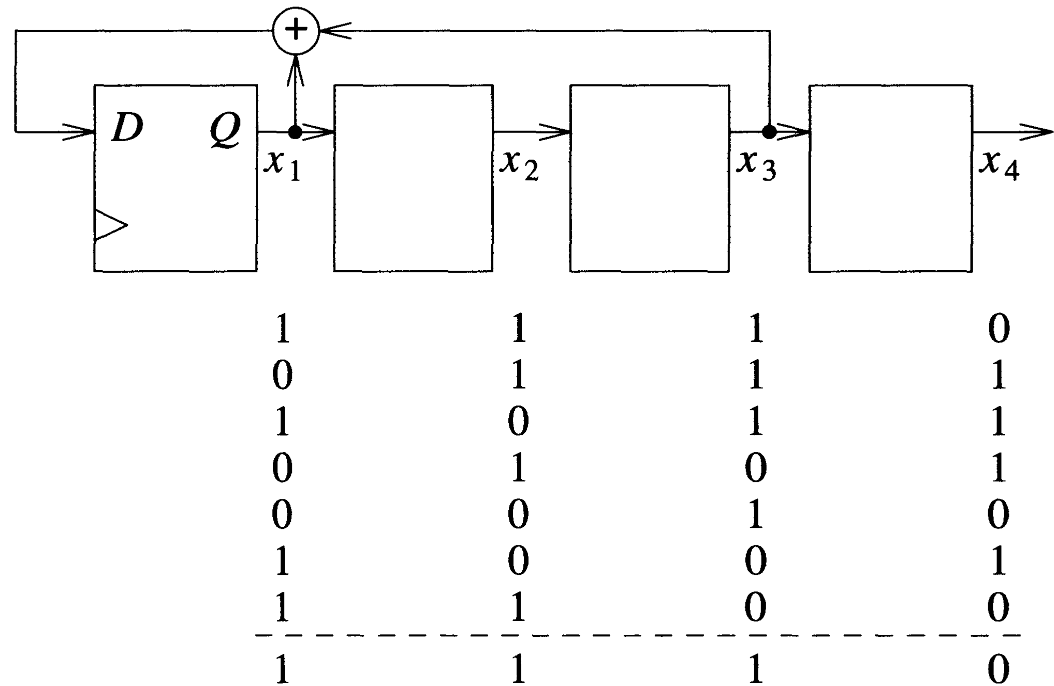


Figure 11.12 A 4-stage LFSR/SR for a (4,2)-CUT

TPG – Condensed LFSR

- (n, w) circuit
- Can produce efficient test set when w >= n/2
- But produce more test than combined LFSR/SR
- What patterns does it generate?

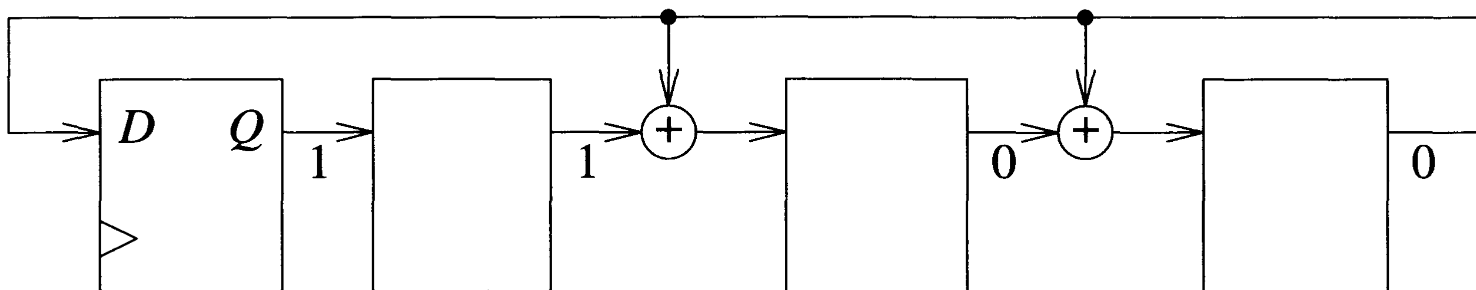


Figure 11.15 A condensed LFSR for a (4,2)-CUT

Generic Off-line BIST Architectures

- Off-line BIST Architectures
  1. Centralized or Distributed
  2. Embedded or Separate
- BIST architecture elements:
  1. Test pattern generators
  2. Output response analyzers
  3. Circuit under test
  4. Distribution system (DIST) for transmitting date from TPGs to CUTs and from CUTs to ORAs
  5. BIST Controller

BIST Controller

During testing BIST Controller can carry out one or more functions:

1. **Single-step** the CUTs through some test sequence
2. **Inhibit** system clocks and control test clocks
3. **Communicate** with other test controllers
4. **Control** the operation of self-test (seeding of registers, number of test patterns processed, etc.)

Centralized and BIST Architecture

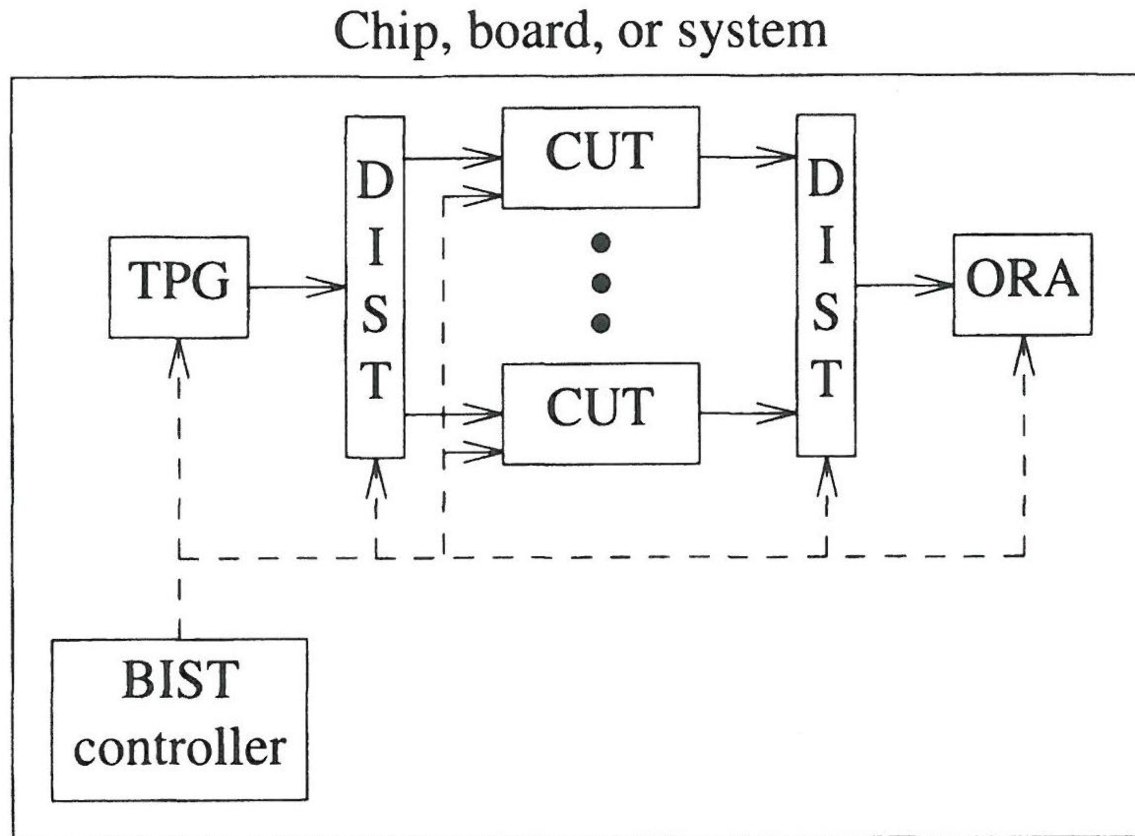


Figure 11.18 Generic form of centralized and separate BIST architecture

Distributed and Separate BIST

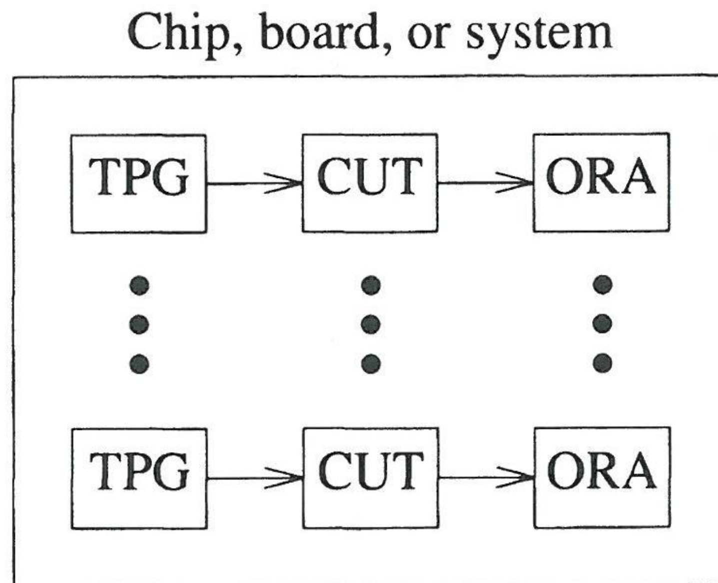


Figure 11.19 Generic form of distributed and separate BIST architecture

Distributed and Embedded BIST

- TPG and ORA configured from within CUT
- Complex design to control

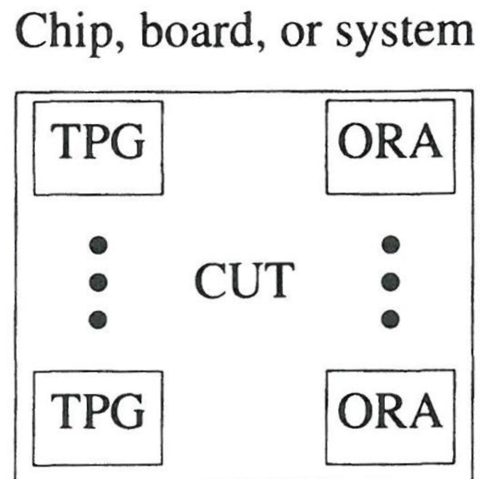


Figure 11.20 Generic form of distributed and embedded BIST architecture

BIST Architecture

When choosing BIST architecture, following factors need to be considered:

1. Degree of test parallelism
2. Fault coverage
3. Level of packaging
4. Test time
5. Physical constraints
6. Complexity of replaceable units
7. Factory and field test-and-repair strategy
8. Performance degradation

Some Example BIST Architectures

1. Centralized and Separate Board-Level BIST

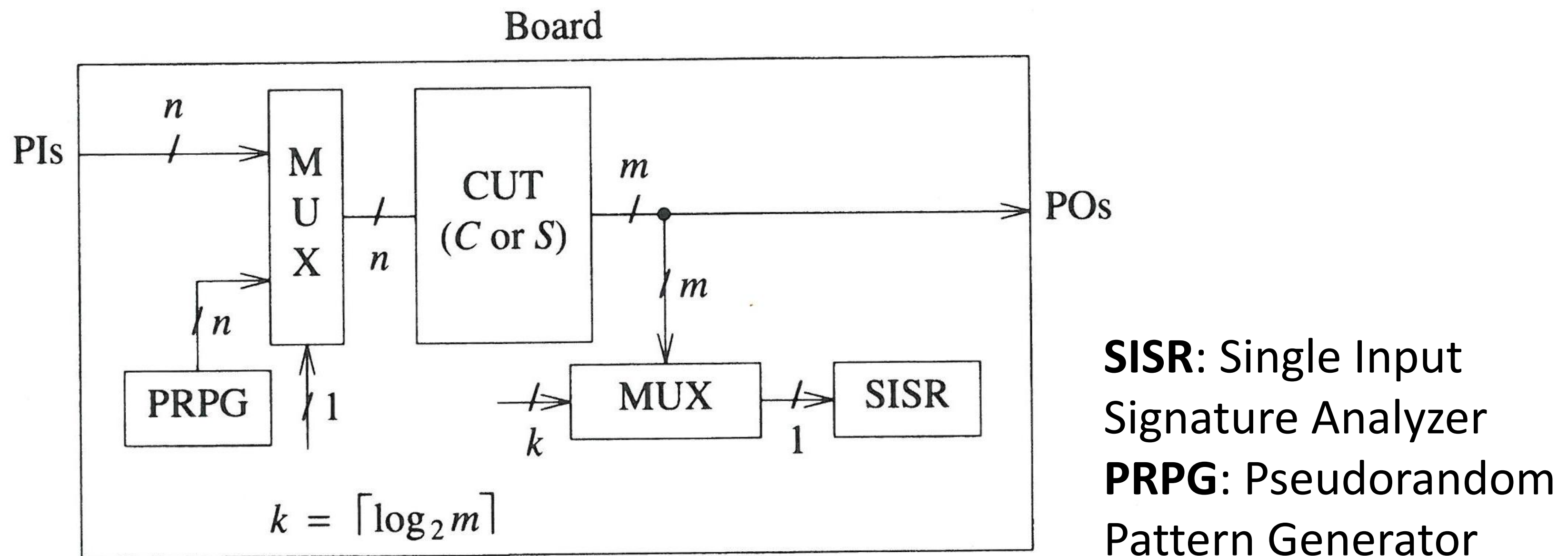


Figure 11.21 A centralized and separate BIST architecture (CSBL)

SISR: Single Input Signature Analyzer
PRPG: Pseudorandom Pattern Generator

Some Example BIST Architectures

2. LSSD On-Chip Self-Test (LOCST)

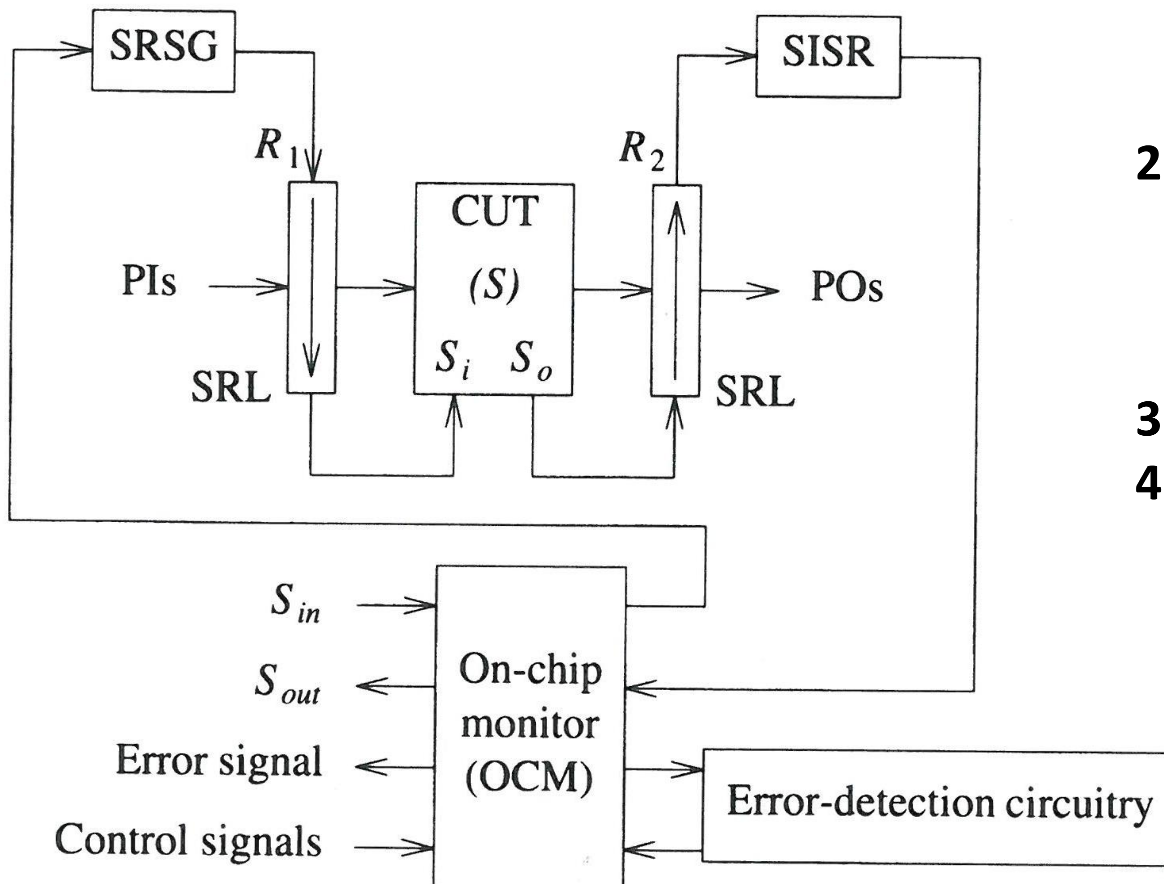


Figure 11.25 The LOCST architecture

Test Process

1. **Initialize**
   Scan path loaded with seed via S_{in}
2. **Activate Self-test mode**
   a) Disable sys clks on R1 and R2
   b) Enable LFSR operation
3. **Execute Self-test**
4. **Check Result**
   Compare final value of SISR with known good signature

LSSD: Level Sensitive Scan Design
SRSG: Shift Register Sequence Generator

Some Example BIST Architectures

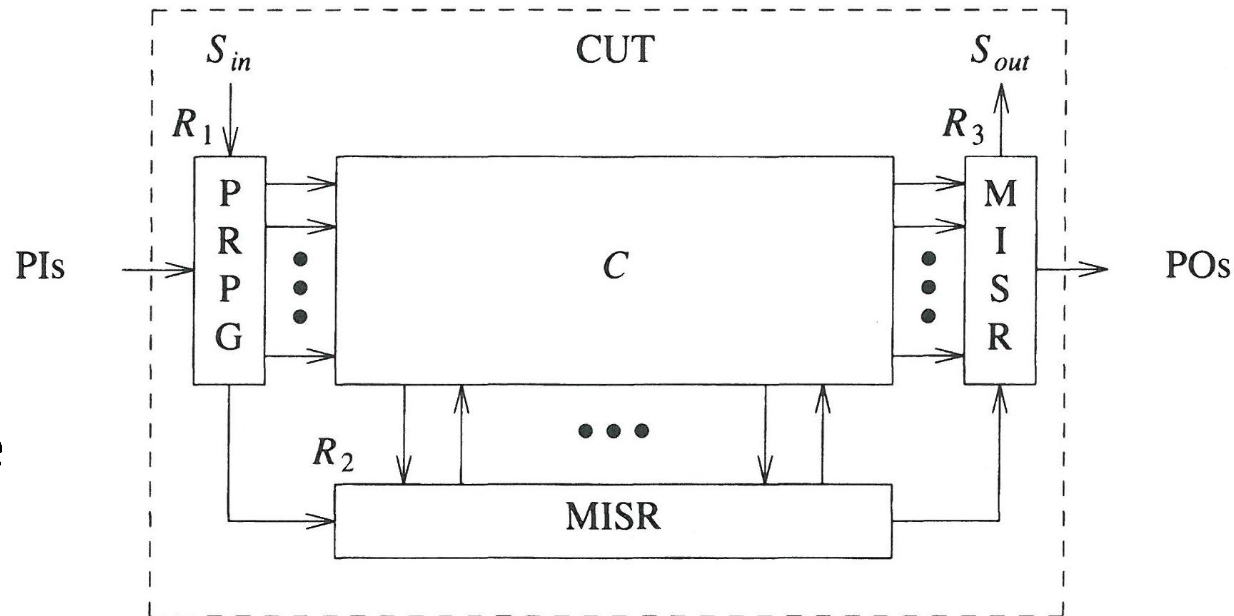
3. Random Test Data (RTD) BIST

- Previous archs – entire scan path be loaded with new data to apply a single test pattern to CUT; RTD overcomes this
- Test process:

a) R1, R2, and R3 set to scan mode and a seed pattern is loaded

b) Registers put to test mode and held while circuit is tested

c) For each clock cycle, R1 and R2 generate a new test pattern, and R2 and R3 operate as a MISR



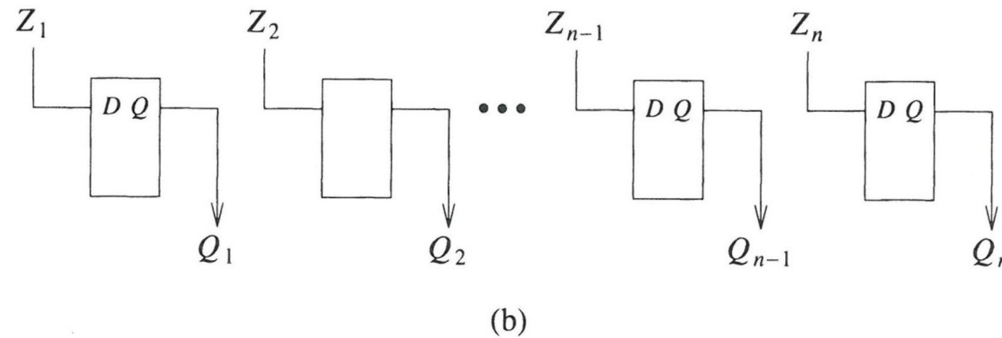
Built-In Logic Block Observation (BILBO) Register

Operates in four modes:

- B1 = B2 =1 - Normal Mode (parallel load register)
- B1 = B2 = 0 – Shift Register Mode
- B1 = 1, B2 = 0 -- LFSR (test) mode
- B1 = 0 , B2 = 1 -- all storage cells - reset

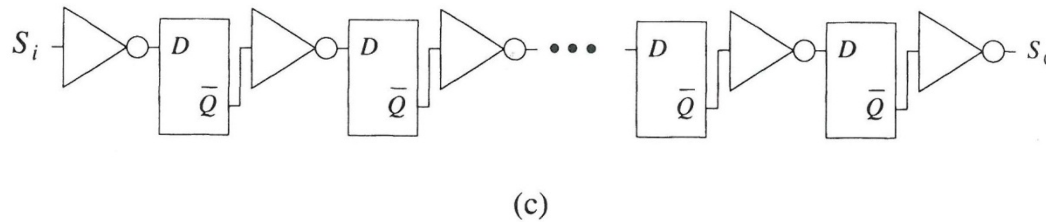
BIBLO Register Modes

B1 = B2 =1
Normal Mode



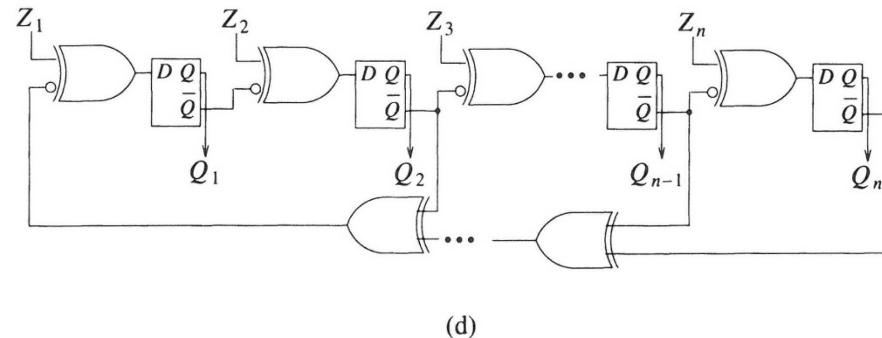
(b)

B1 = B2 = 0
Shift Register Mode



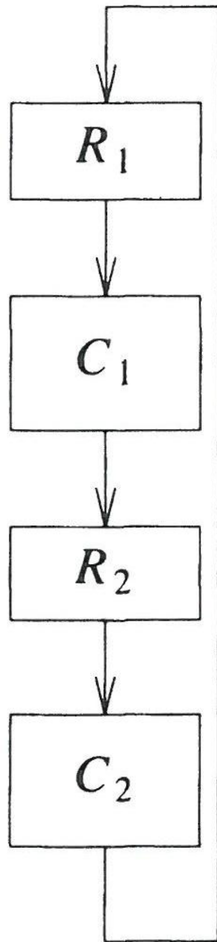
(c)

B1 = 1, B2 = 0
LFSR Mode



(d)

BIST Design with BILBO Registers



(a)

- To test C_1
  1. R_1 and R_2 are seeded
  2. R_1 into PRPG mode, R_2 into MISR mode
  3. Hold inputs to R_1 to value 0 so that LFSR (R_1) acts as a PRPG
  4. Run for N clock cycles
- If C_1 is not too large, C_1 can be tested exhaustively (except for all-zero pattern)
- At the end of test session, R_2 scanned out and signature checked
- Need two test sessions, one for C_1 and other for C_2

Summary

- Built-In Self Test – can be offline or online
- Needs test pattern generators (TPGs) and output response analyzers (ORAs)
- Linear Feedback Shift Registers (LFSRs) can be used as both as a TPG and as an ORA
- Offline BIST architectures can be centralized or distributed, embedded or separate