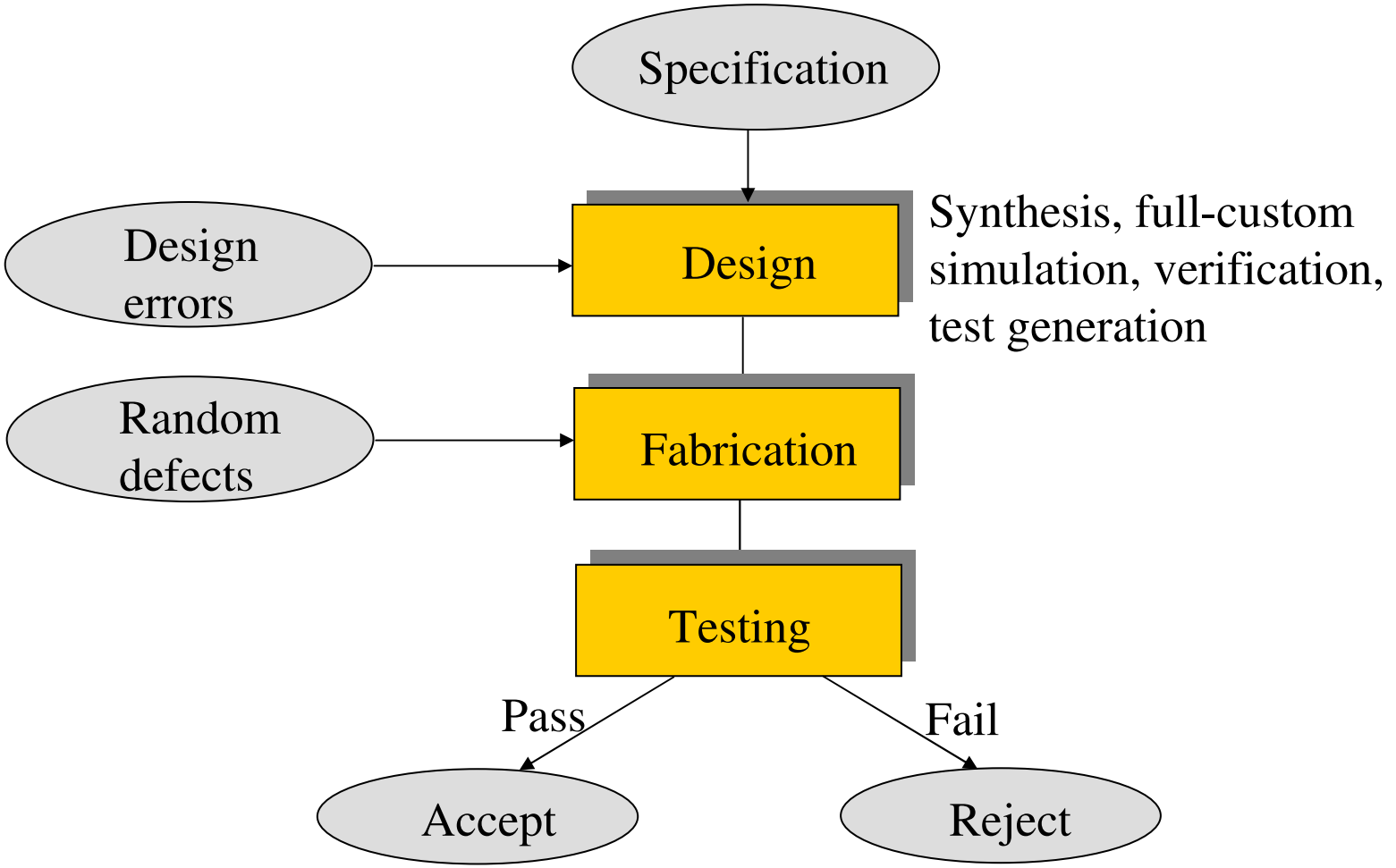
CMOS Testing-2

- Design and test
- Design for testability (DFT)
  - Scan design
- Built-in self-test
- IDDQ testing

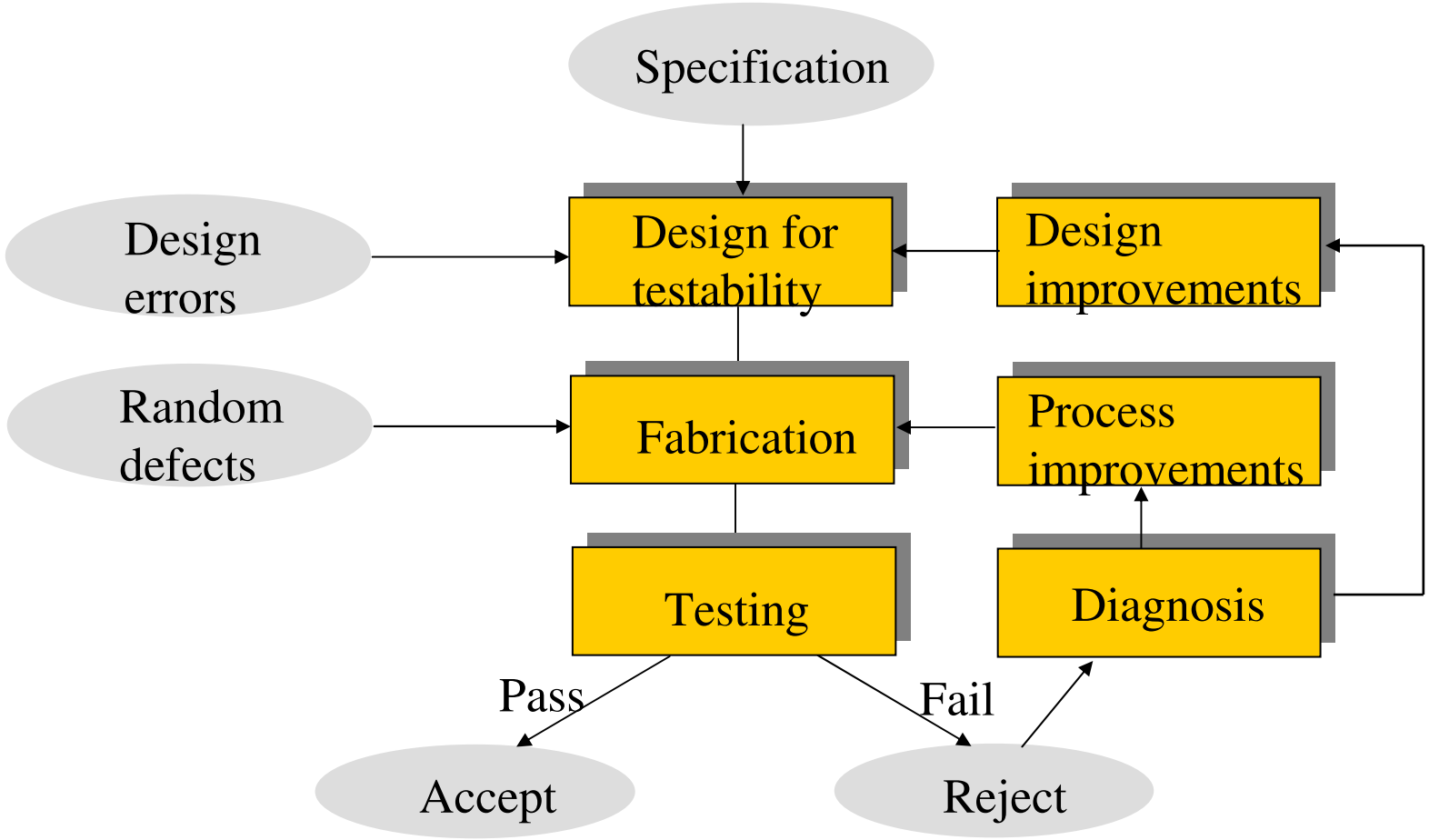
Design and Test Flow: Old View

- Test was merely an afterthought



Design and Test Flow: New View

Design and test are tightly coupled

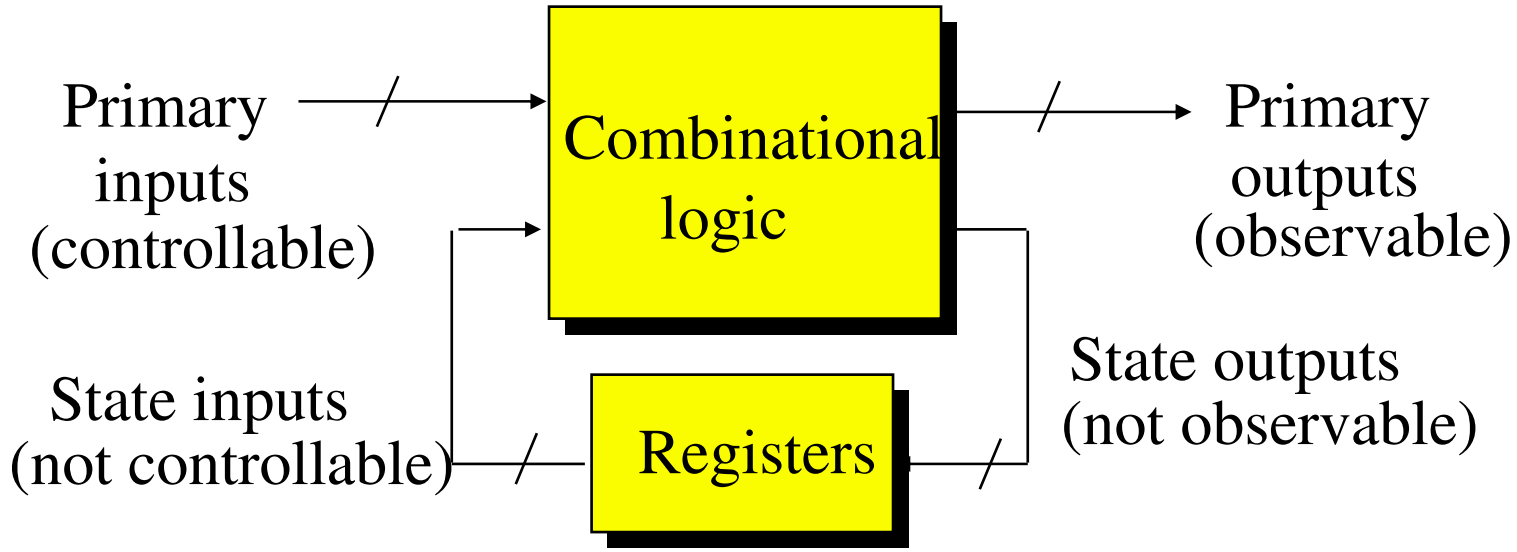


Testing Sequential Circuits

Difficult problem-internal states cannot be directly controlled and observed

Long test sequences are necessary

Solution: Scan design-simplify to combinational circuit testing



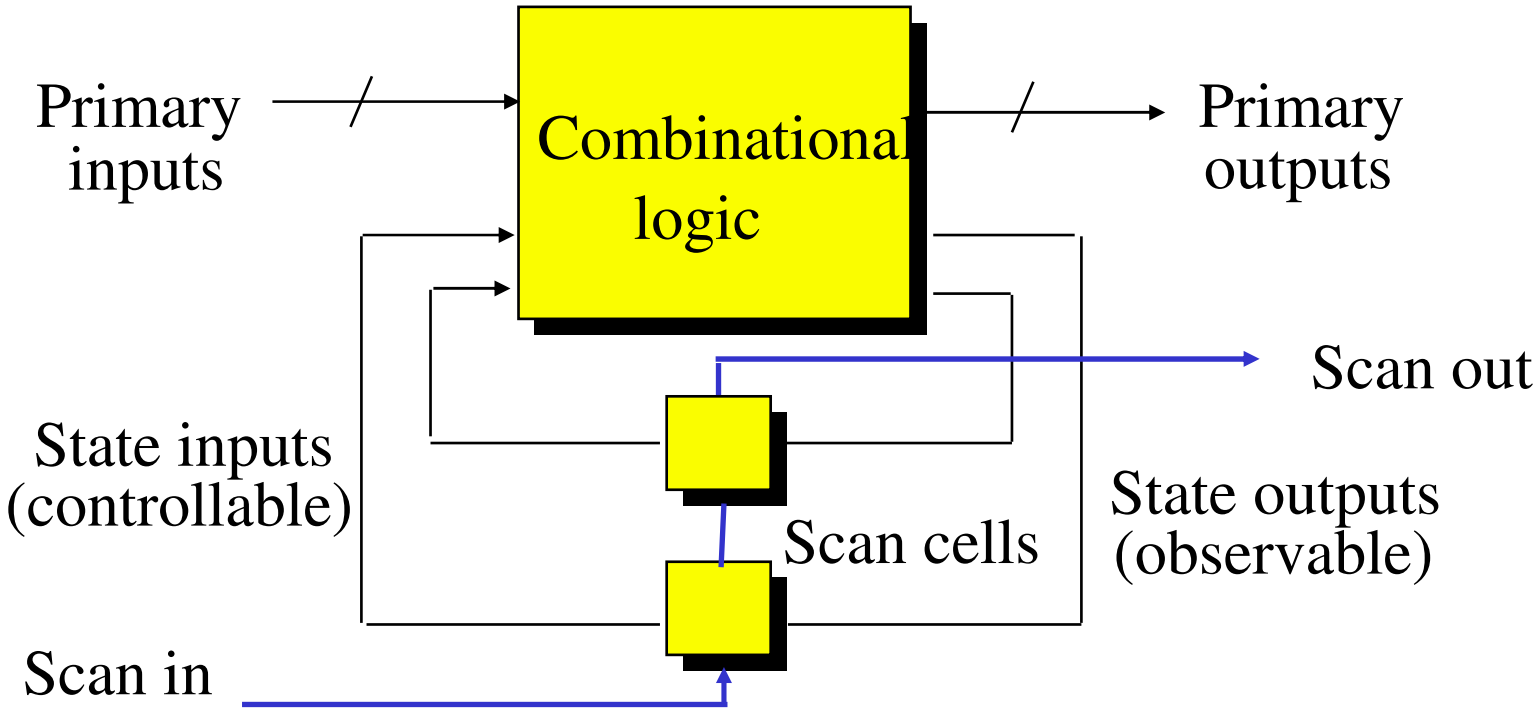
Design for Test

- Design the chip to increase observability and controllability
- If each register could be observed and controlled, test problem reduces to testing combinational logic between registers.
- Better yet, logic blocks could enter test mode where they generate test patterns and report the results automatically.

Scan Design

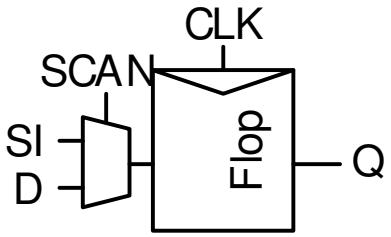
Make all flip-flops directly controllable and observable by adding multiplexers

Popular design-for-test (DFT) technique-circuit is now combinational for testing purposes

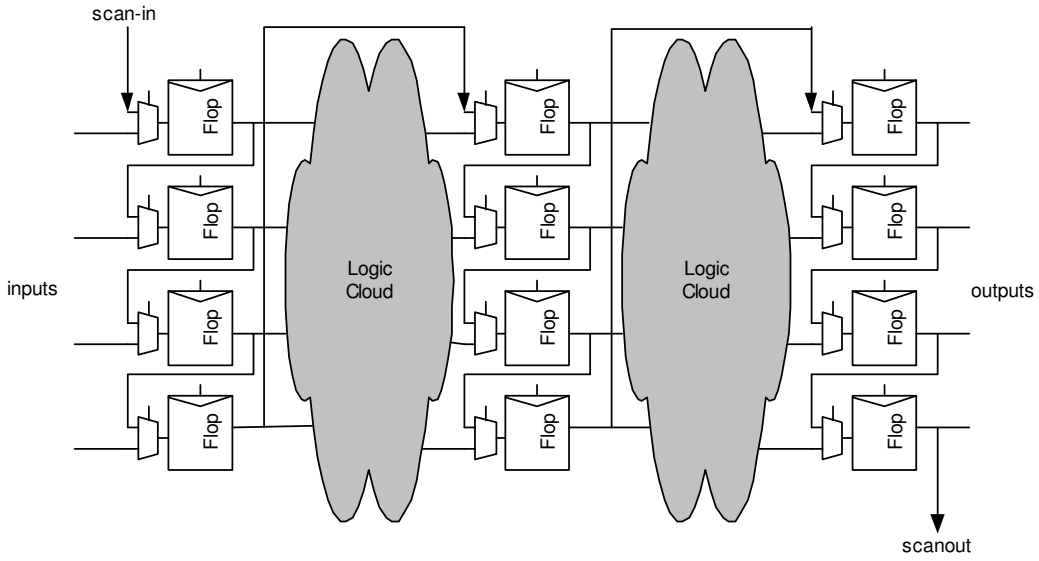


Scan Design

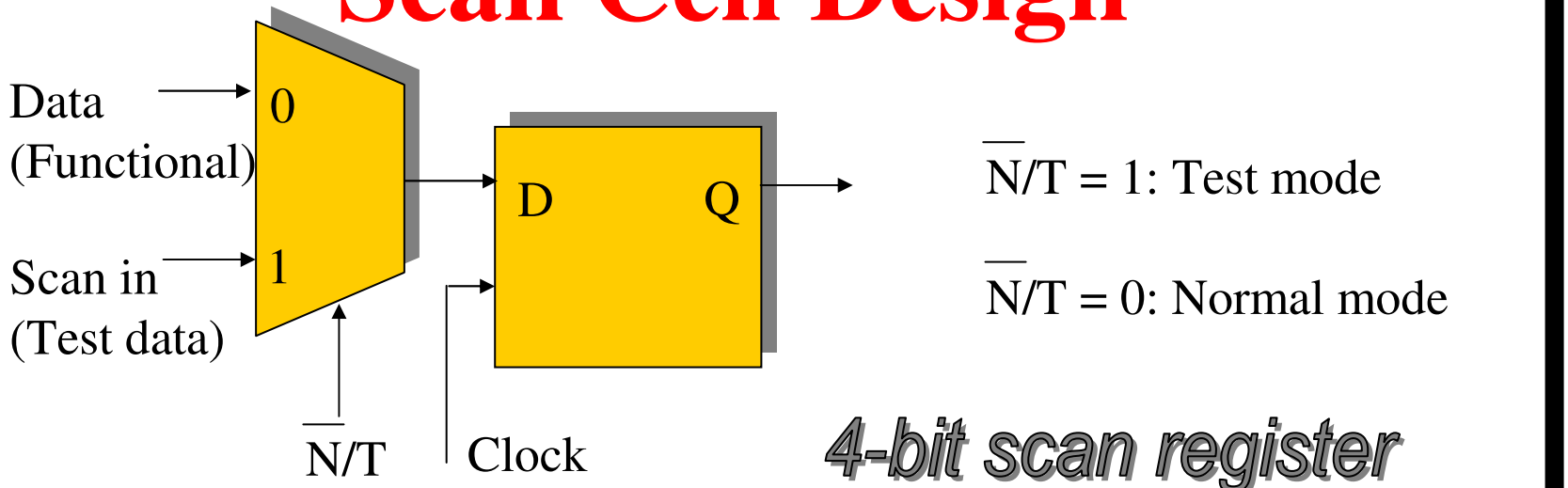
- Convert each flip-flop to a scan register
  - Only costs one extra multiplexer
- Normal mode: flip-flops behave as usual
- Scan mode: flip-flops behave as shift register



- Contents of flops can be scanned out and new values scanned in



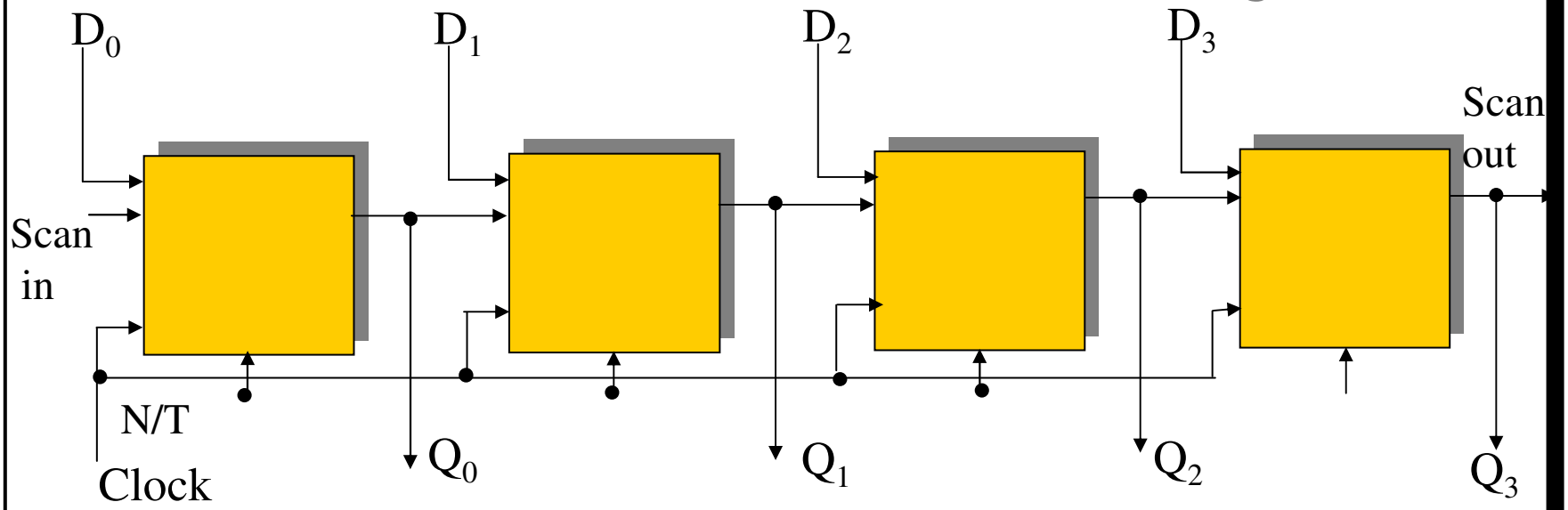
Scan Cell Design



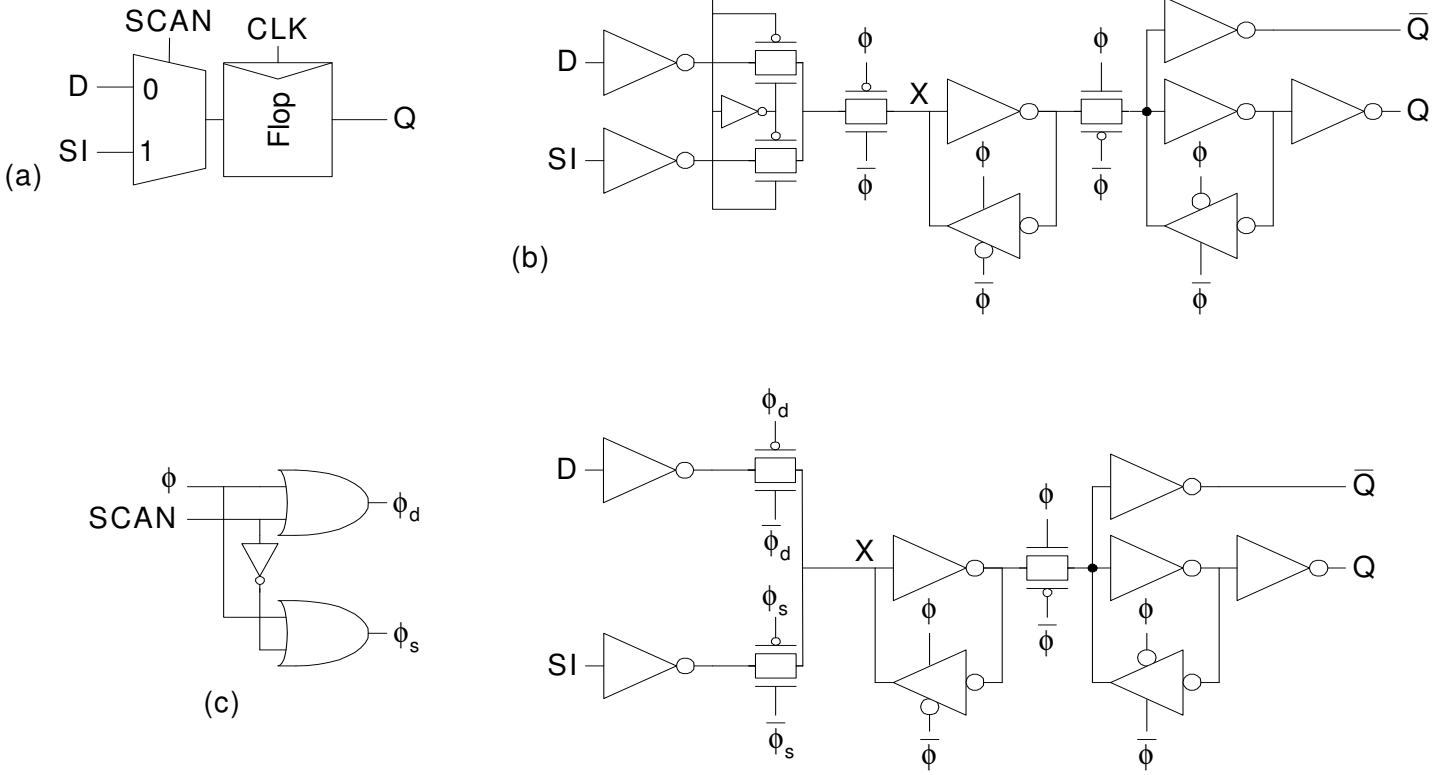
$\overline{N/T}$ = 1: Test mode

$\overline{N/T}$ = 0: Normal mode

4-bit scan register

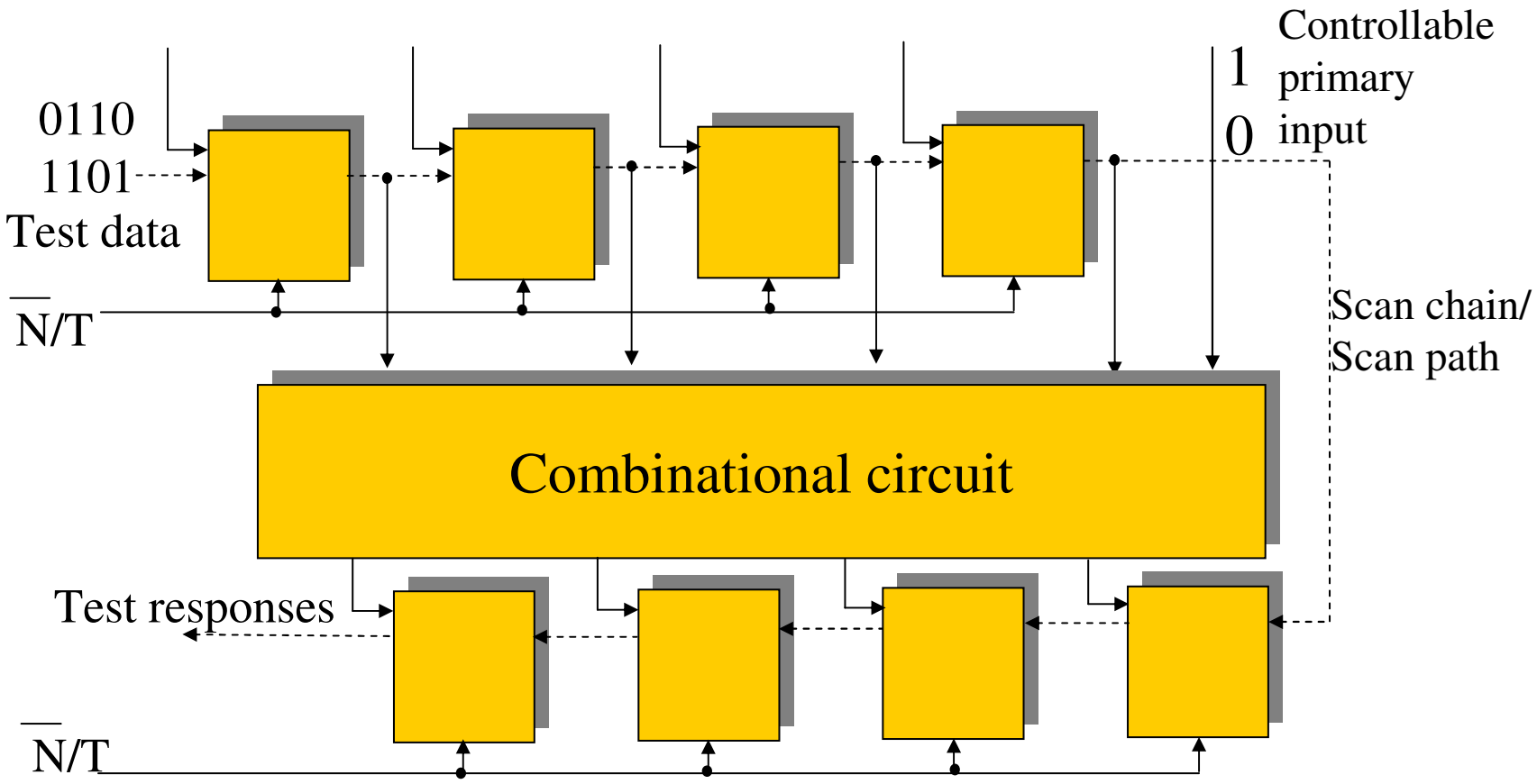


Scannable Flip-flops



Scan Design

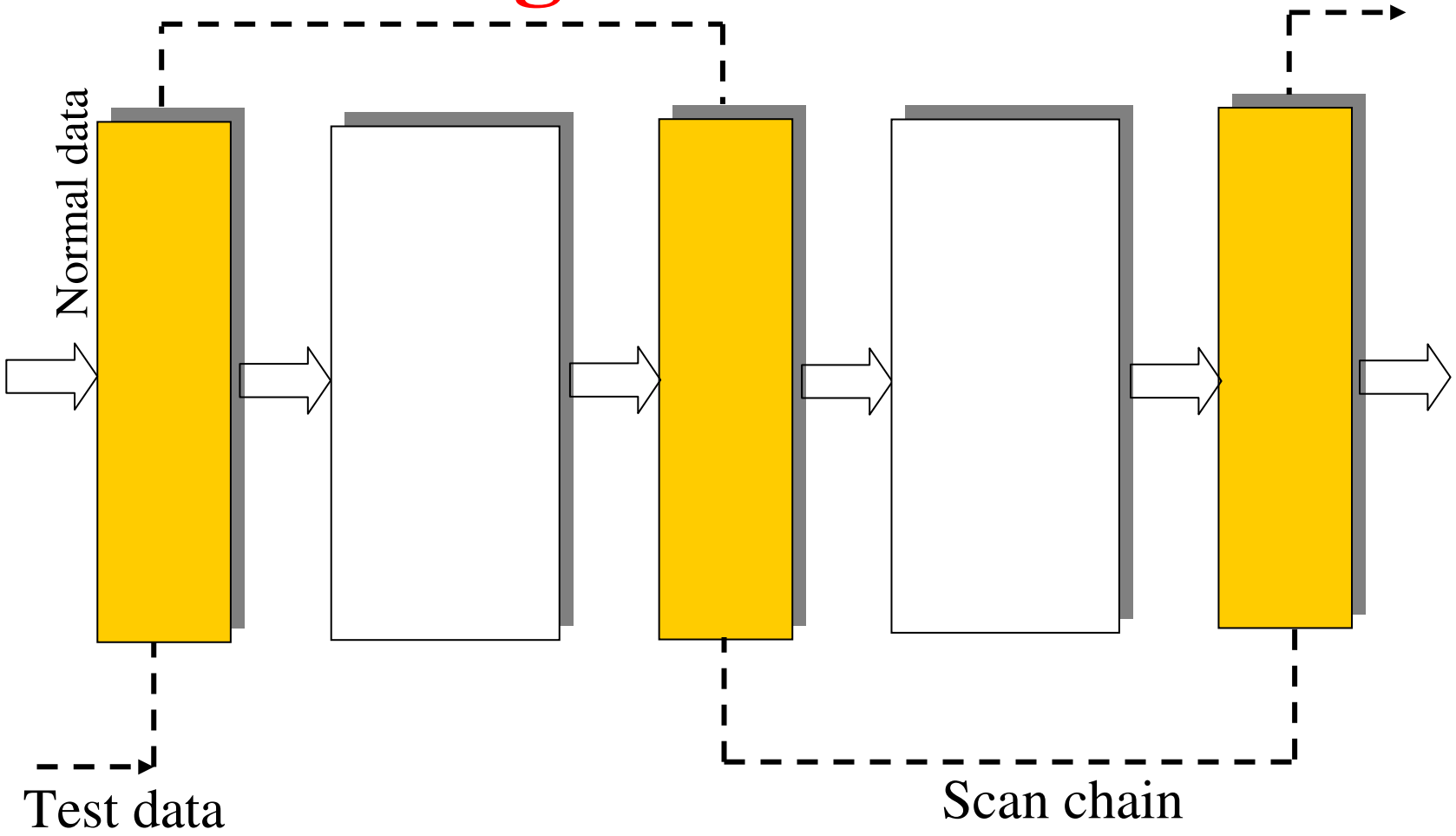
- Separate input and output 4-bit scan registers
- Test sequence: {01100, 11011}, first 4 bits are for flip-flops



Steps in Scan Testing

- $\overline{N}/T = 1$: Scan in test pattern, hold appropriate bit pattern on controllable primary inputs
- $\overline{N}/T = 0$: Apply test pattern to combinational circuit
- $\overline{N}/T=1$: Scan out test responses
- Scan provides complete controllability and observability
- Testing time? How many cycles? How to test scan registers?

Long Scan Chains

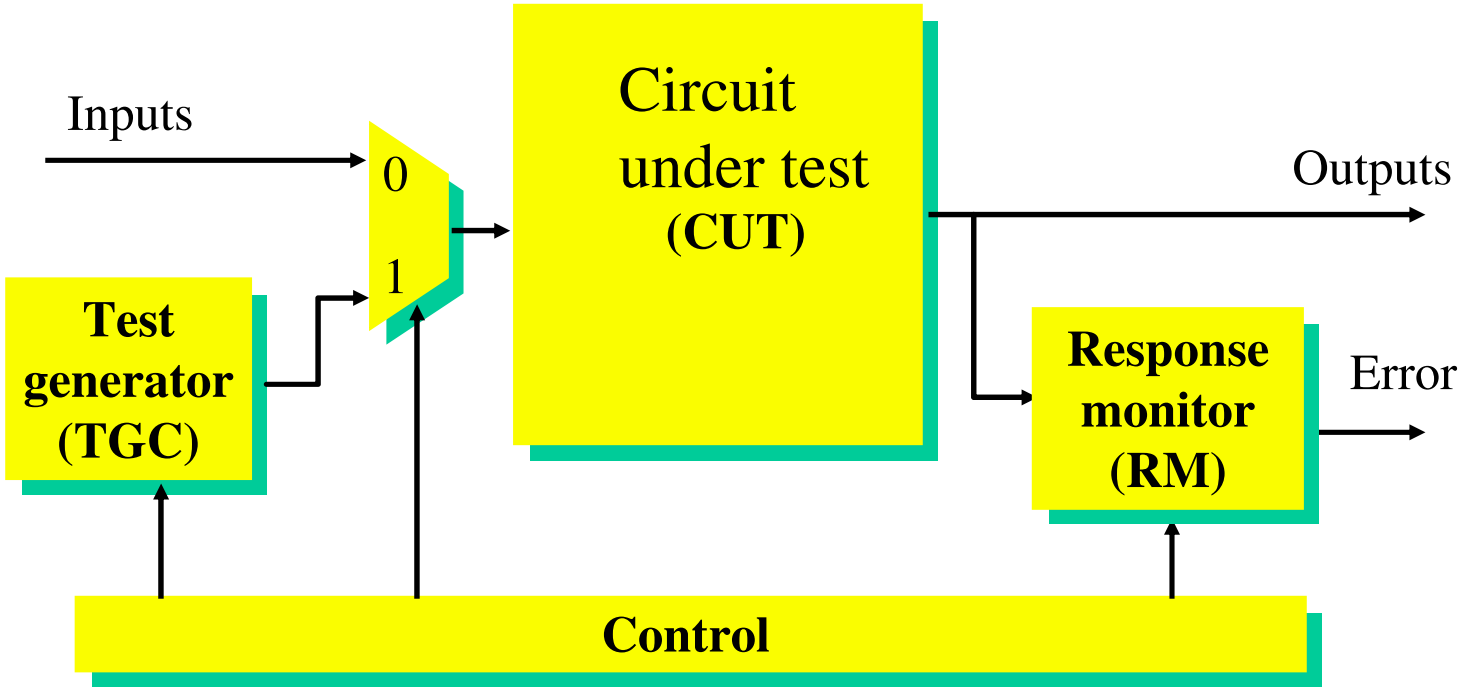


Test vectors need to be translated to scan format

Built-in Self-test

- Built-in self-test lets blocks test themselves
  - Generate pseudo-random inputs to comb. logic
  - Combine outputs into a *syndrome*
  - With high probability, block is fault-free if it produces the expected syndrome

Built-in Self Testing (BIST)



On-chip test generator and response monitor

BIST: Advantages

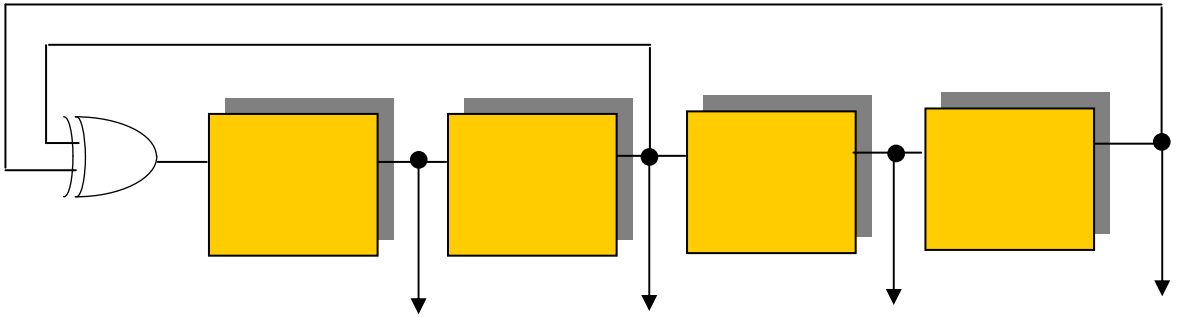
- Lower cost due to elimination of external tester
  - Sematech's projection: 500 MHz tester (400 pins) will cost $50M in 2010, 90% of on-chip testing will be done using BIST
- In-system, at-system, high-quality testing
- Faster fault detection, ease of diagnosis
- Overcomes pin limitations and related interfacing problems
- Reduces maintenance and repair costs at system level

BIST: Issues

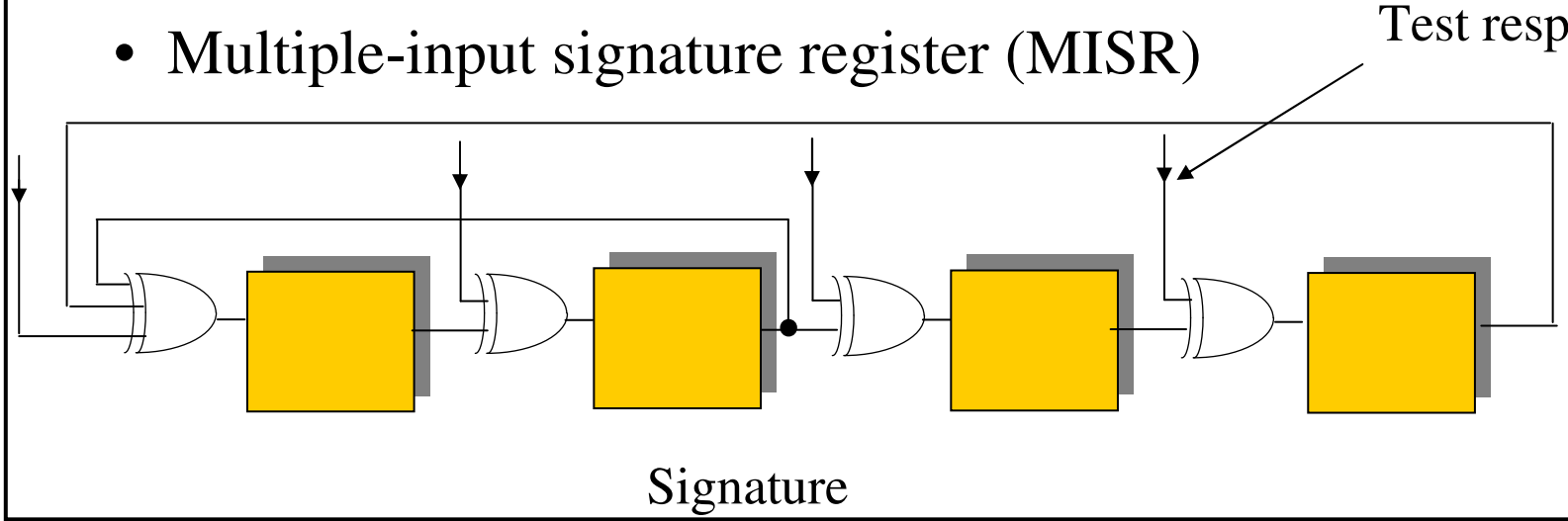
Test strategy (random, exhaustive, deterministic)
Circuit partitioning
Test pattern generation
    Exhaustive: counters
    Random: Linear-feedback shift registers (LFSRs)
    Deterministic: ROM, other methods?
Response analysis
Test control and scheduling

BIST Logic Circuits

- Linear-feedback shift-register (LFSR)

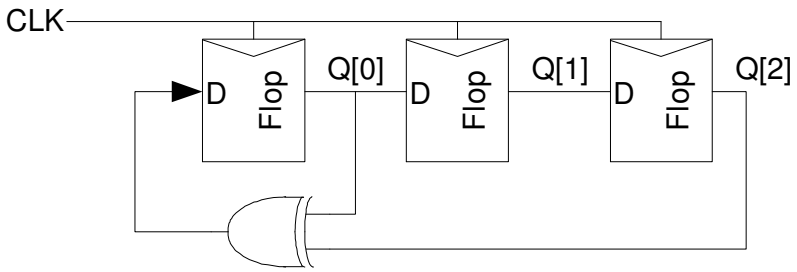


- Multiple-input signature register (MISR)



BIST Pattern Generation

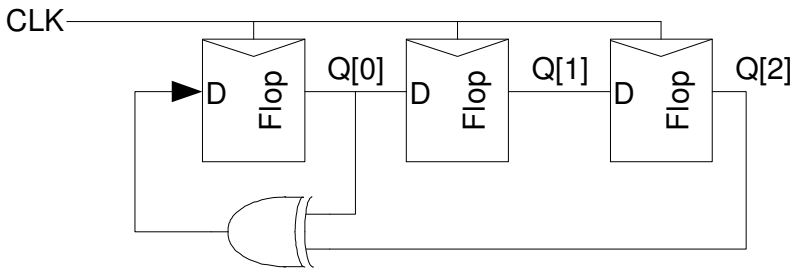
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	
2	
3	
4	
5	
6	
7	

BIST Pattern Generation

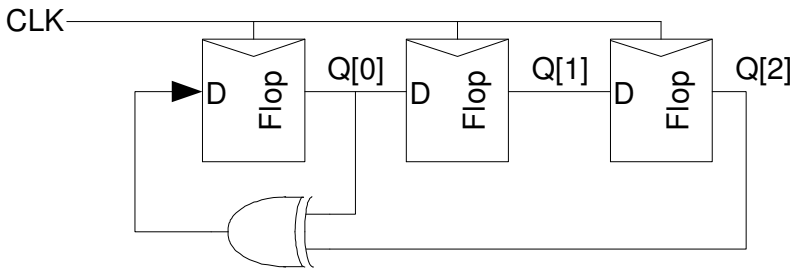
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	
3	
4	
5	
6	
7	

PRSG

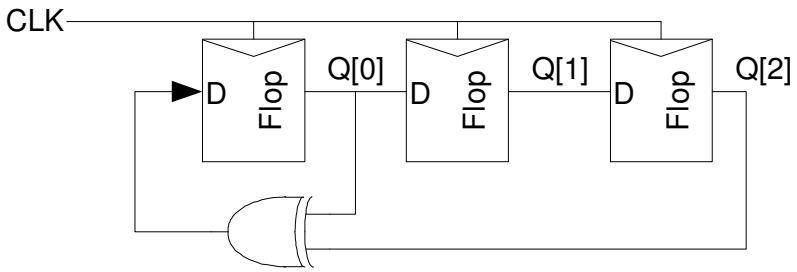
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	
4	
5	
6	
7	

PRSG

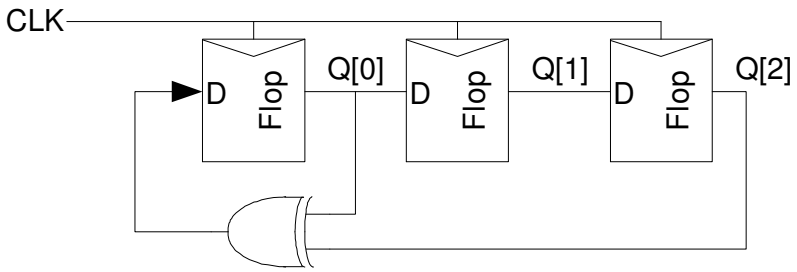
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	
5	
6	
7	

PRSG

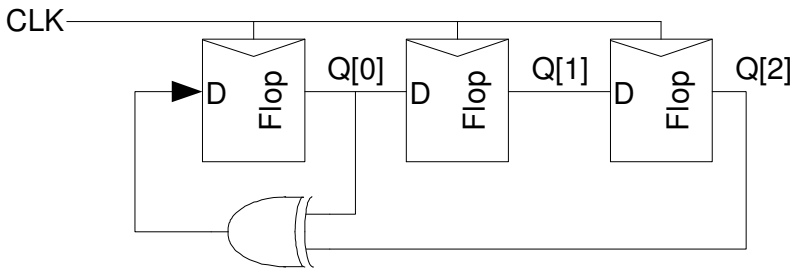
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	100
5	
6	
7	

PRSG

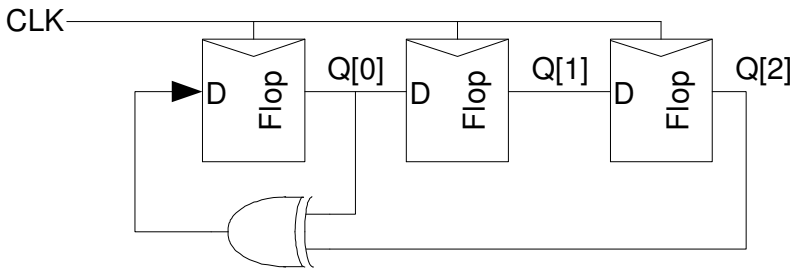
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	100
5	001
6	
7	

PRSG

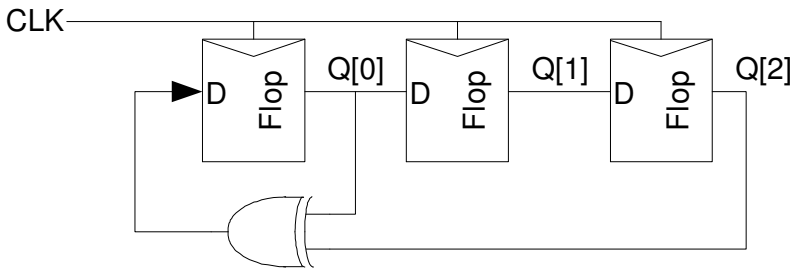
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*



Step	Q
0	111
1	110
2	101
3	010
4	100
5	001
6	011
7	

PRSG

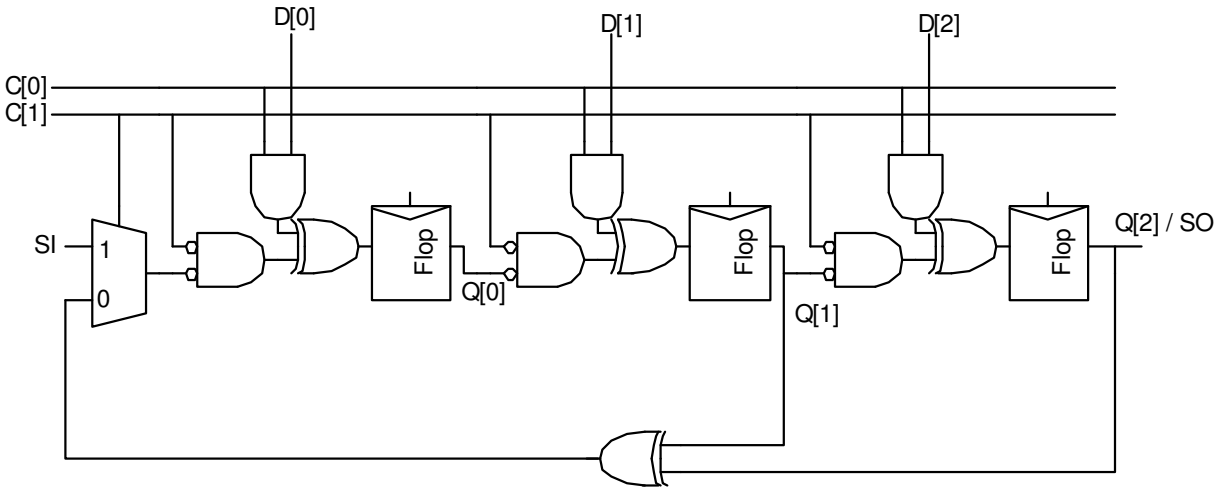
- *Linear Feedback Shift Register*
  - Shift register with input taken from XOR of state
  - *Pseudo-Random Sequence Generator*

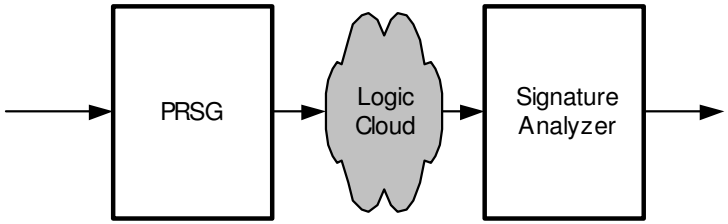


Step	Q
0	111
1	110
2	101
3	010
4	100
5	001
6	011
7	111 (repeats)

BILBO

- Built-in Logic Block Observer
  - Combine scan with PRSG & signature analysis



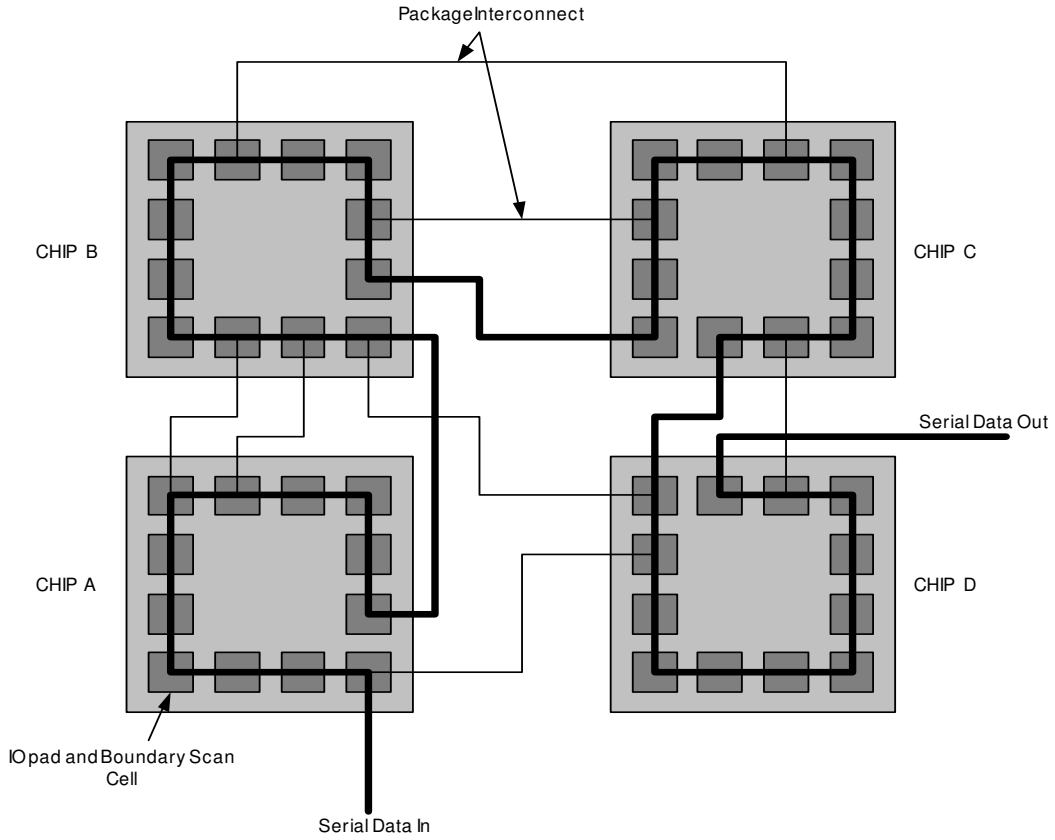


MODE	C[1]	C[0]
Scan	0	0
Test	0	1
Reset	1	0
Normal	1	1

Boundary Scan

- Testing boards is also difficult
  - Need to verify solder joints are good
    - Drive a pin to 0, then to 1
    - Check that all connected pins get the values
- Through-hold boards used “bed of nails”
- SMT and BGA boards cannot easily contact pins
- Build capability of observing and controlling pins into each chip to make board test easier

Boundary Scan Example



Boundary Scan Interface

IEEE 1149.1 JTAG standard

- Boundary scan is accessed through five pins
  - TCK: test clock
  - TMS: test mode select
  - TDI: test data in
  - TDO: test data out
  - TRST*: test reset (optional)
- Chips with internal scan chains can access the chains through boundary scan for unified test strategy.

BIST in Industry

Early days: AT&T (Lucent) incorporated BIST in hundreds of commercial chips

Intel: 80386, Pentium, Pentium Pro

Hardware overhead typically 15% of self-tested portion (around 5% for entire chip, e.g. 6% for the Pentium Pro)

Regular embedded arrays (RAMs, PLAs) almost always tested using BIST: DEC Alpha, PowerPC

BIST for irregular logic not so widespread

IDDQ Testing

- Based on current measurements, not voltage
  - IDDQ = I_{DD} quiescent
- In CMOS technology, quiescent current is very low
- Testing idea: check for faults by detecting current spikes
  - Advantage: Massive observability, good for detecting shorts
  - Disadvantage: slow, leakage current closer to quiescent current for deep submicron