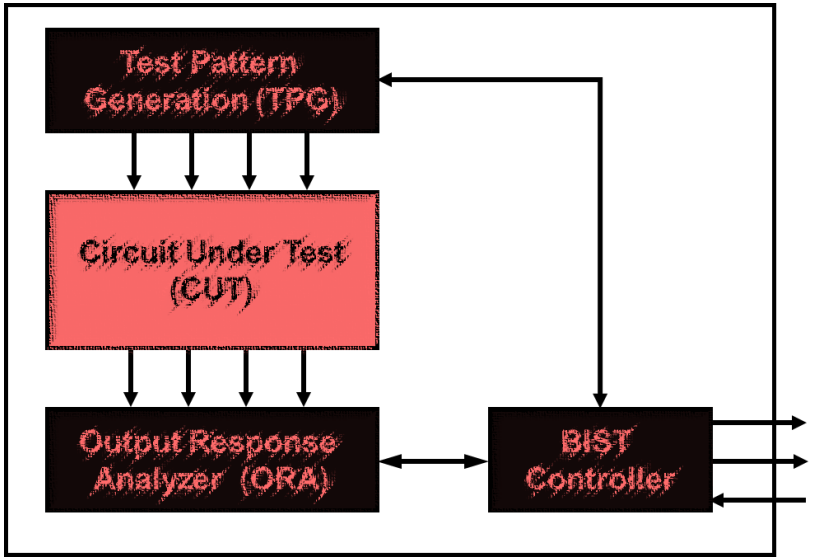
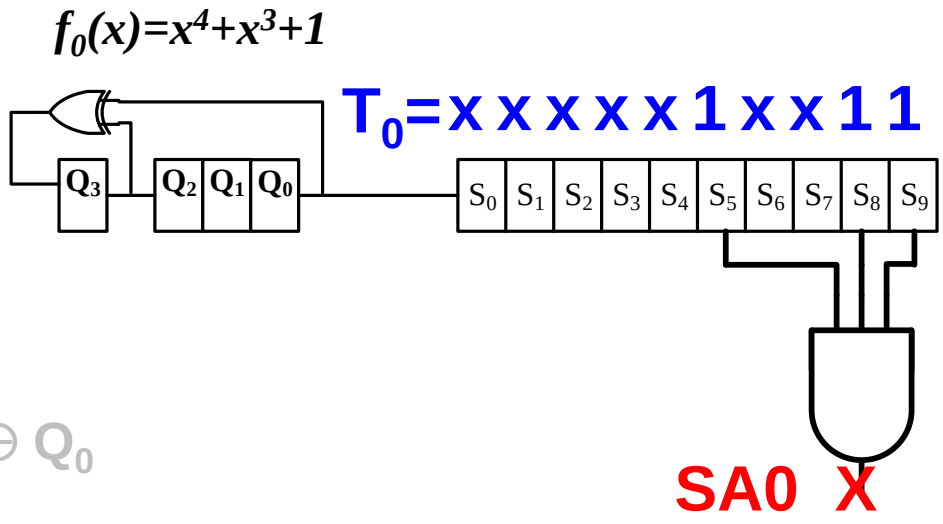
BIST Part 2

- Introduction
- Pattern Generation
- Output Response Analysis
- BIST Architecture
- **Problems and Solutions**
  - **Fault Coverage Not Enough**
    - Structure Dependency
    - **Linear Dependency**
    - **Random Pattern Resistant Fault**
  - **Long Test Length**
  - **"X" Unknown Output Responses**
  - **Diagnosis/Debug**
  - **High Test Power**
- **Concluding Remarks**



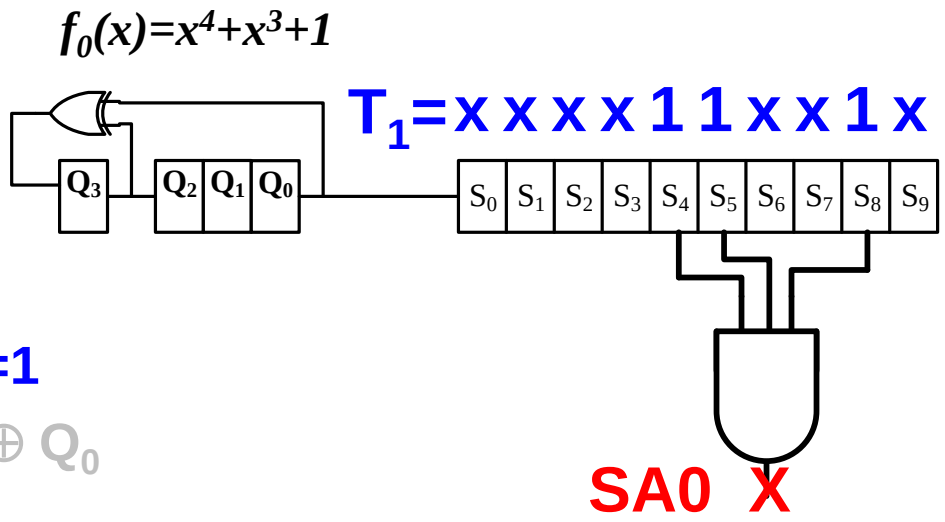
Solve LFSR Seed (Review 13.3)

- **Linear system**
  - $S_9 = Q_0 = 1$
  - $S_8 = Q_1 = 1$
  - $S_7 = Q_2$
  - $S_6 = Q_3$
  - $S_5 = S_9 \oplus S_6 = Q_3 \oplus Q_0 = 1$
  - $S_4 = S_8 \oplus S_5 = Q_3 \oplus Q_1 \oplus Q_0$
  - $S_3 = S_7 \oplus S_4 = Q_3 \oplus Q_2 \oplus Q_1 \oplus Q_0$
  - $S_2 = S_6 \oplus S_3 = Q_2 \oplus Q_1 \oplus Q_0$
  - $S_1 = S_5 \oplus S_2 = Q_3 \oplus Q_2 \oplus Q_1$
  - $S_0 = S_4 \oplus S_1 = Q_2 \oplus Q_0$
- **4 variables, 3 equations**
  - **Solution: seed =$[Q_3\ Q_2\ Q_1\ Q_0]$= 0x11**



Linear Dependency Problem

- **Linear system**
  - $S_9 = Q_0$
  - $S_8 = Q_1$ =1
  - $S_7 = Q_2$
  - $S_6 = Q_3$
  - $S_5 = S_9 \oplus S_6 = Q_3 \oplus Q_0$ =1
  - $S_4 = S_8 \oplus S_5 = Q_3 \oplus Q_1 \oplus Q_0$ =1
  - $S_3 = S_7 \oplus S_4 = Q_3 \oplus Q_2 \oplus Q_1 \oplus Q_0$
  - $S_2 = S_6 \oplus S_3 = Q_2 \oplus Q_1 \oplus Q_0$
  - $S_1 = S_5 \oplus S_2 = Q_3 \oplus Q_2 \oplus Q_1$
  - $S_0 = S_4 \oplus S_1 = Q_2 \oplus Q_0$

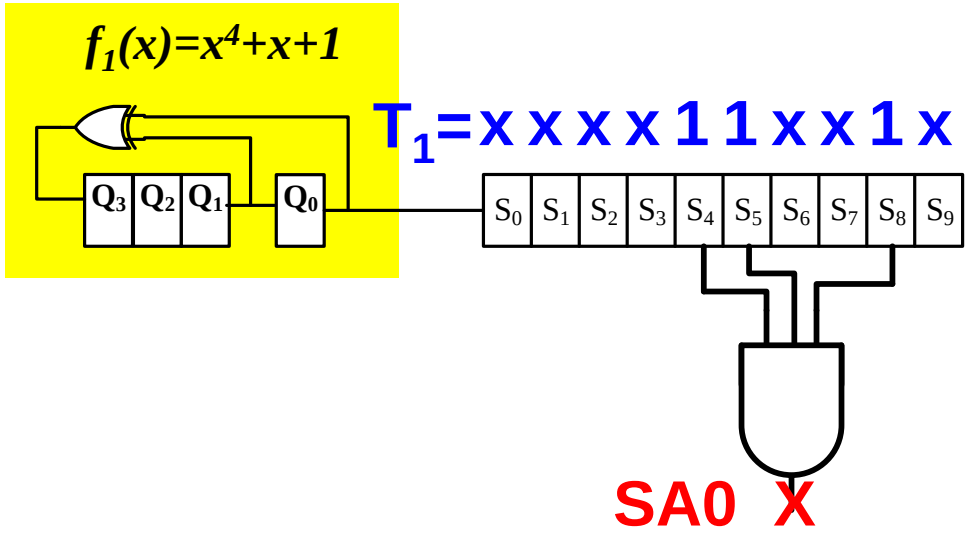


- ***Linear dependency problem:***
  - **No solution due to conflict of linear equations**

What to Do? Change *f(x)*

Change Polynomial x^4+x+1

- **Linear system**
  - $S_9 = Q_0$
  - $S_8 = Q_1$ =1
  - $S_7 = Q_2$
  - $S_6 = Q_3$
  - $S_5 = S_9 \oplus S_8 = Q_1 \oplus Q_0$ =1
  - $S_4 = S_8 \oplus S_7 = Q_1 \oplus Q_2$ =1
  - $S_3 = S_7 \oplus S_6 = Q_3 \oplus Q_2$
  - $S_2 = S_6 \oplus S_5 = Q_3 \oplus Q_1 \oplus Q_0$
  - $S_1 = S_5 \oplus S_4 = Q_2 \oplus Q_0$
  - $S_0 = S_4 \oplus S_3 = Q_1 \oplus Q_3$
- **Solution: seed=[Q_3 Q_2 Q_1 Q_0]= X010**

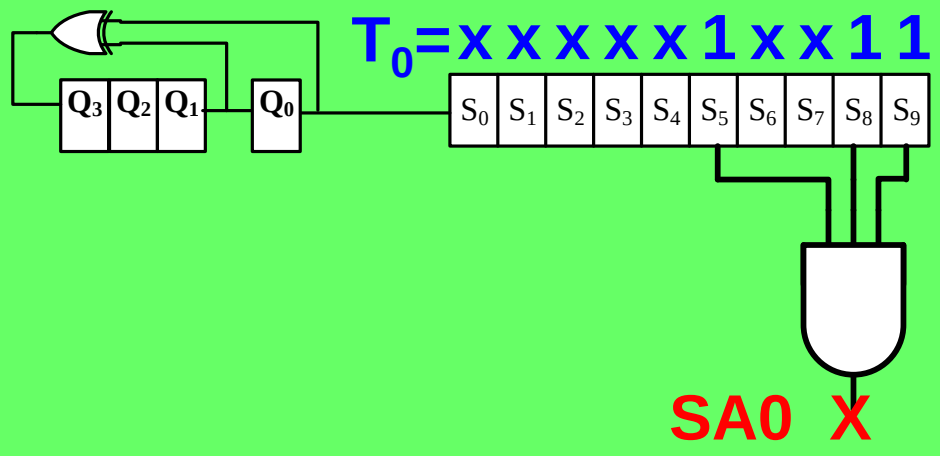


Is $f_1(x)$ Better Than $f_0(x)$?

Quiz

Q: Please show that $f_1(x)=x^4+x+1$ cannot generate T_0='xxxxx1xx11'
ANS:

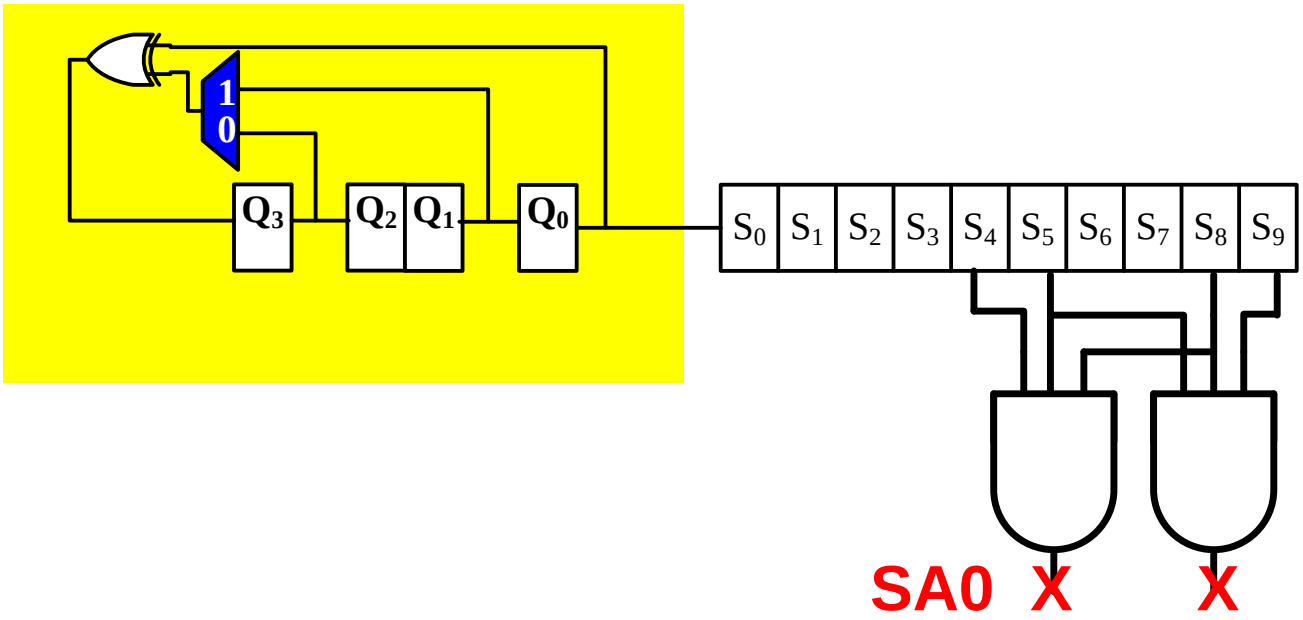
- $S_9 = Q_0$
- $S_8 = Q_1$
- $S_7 = Q_2$
- $S_6 = Q_3$
- $S_5 = Q_1 \oplus Q_0$
- $S_4 = Q_1 \oplus Q_2$
- $S_3 = Q_3 \oplus Q_2$
- $S_2 = Q_3 \oplus Q_1 \oplus Q_0$
- $S_1 = Q_2 \oplus Q_0$
- $S_0 = Q_1 \oplus Q_3$



$f_0(x)$ Generates T_0 but NOT T_1
$f_1(x)$ Generates T_1 but NOT T_0
How to Generate Both T_0 and T_1?

Multiple-polynomial LFSR [Hellebrand 92]

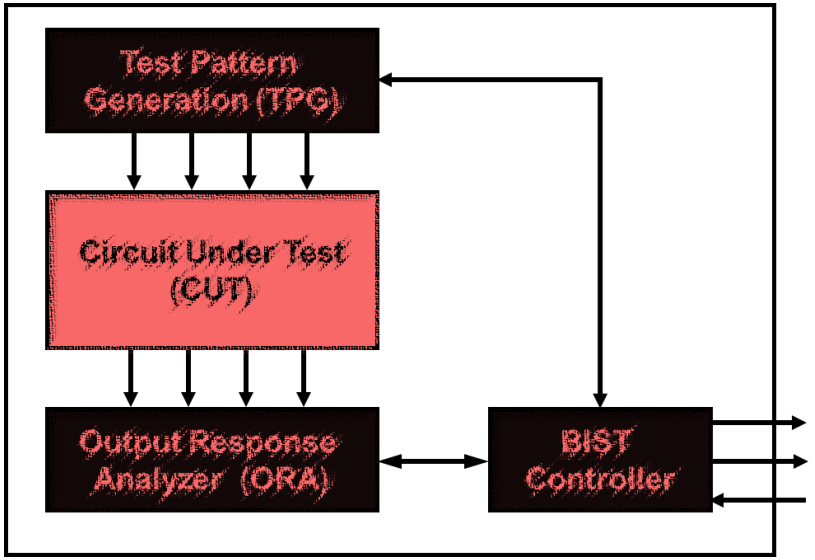
- Polynomial $f_0(x)=x^4+x^3+1$ generates T_0='xxxxx1xx11'
- Polynomial $f_1(x)=x^4+x+1$ generates T_1='xxxx11xx1x'



MP-LFSR Improves Fault Coverage

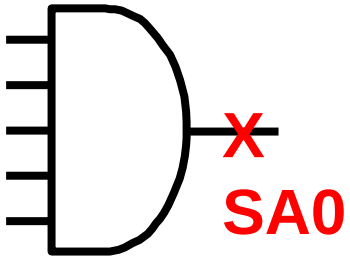
BIST Part 2

- Introduction
- Pattern Generation
- Output Response Analysis
- BIST Architecture
- **Problems and Solutions**
  - **Fault Coverage Not Enough**
    - Structure Dependency
    - **Linear Dependency**
    - **Random Pattern Resistant Faults**
  - **Long Test Length**
  - **"X" Unknown Output Responses**
  - **Diagnosis/Debug**
  - **High Test Power**
- **Concluding Remarks**



Random Pattern Resistant Faults

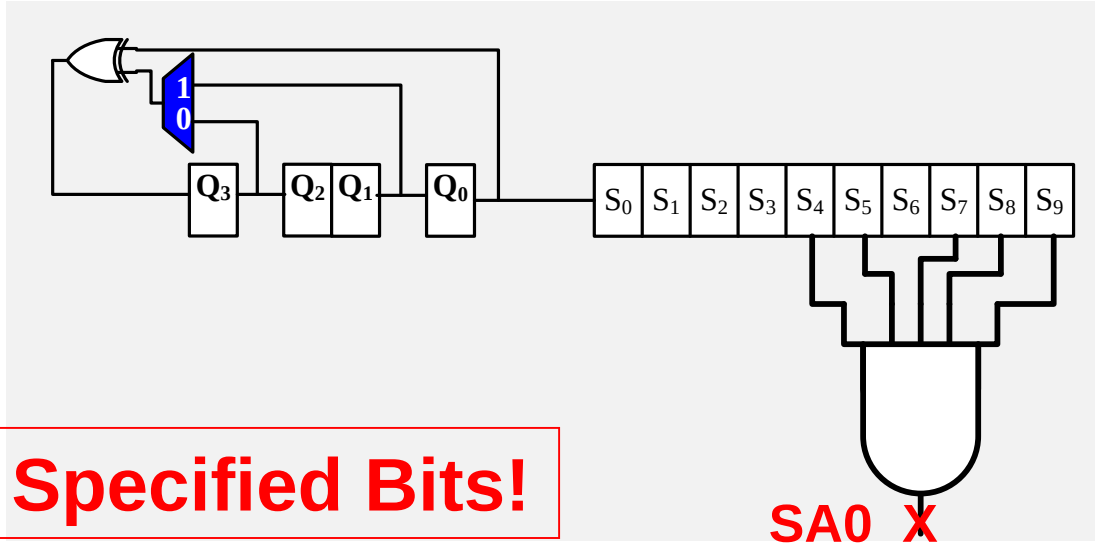
- **RPRF = Faults difficult to be detected by random patterns**
  - **aka. *Difficult faults, hard-to-detect faults***
- **Example: *n*-input AND output SA0 fault**
  - **Probability of detection by random patterns = $1/2^n$**
  - **Very low when *n* is large**
- **Solutions**
  1. **Multiple-polynomial LFSR**
  2. **Top-up ATPG Patterns**
  3. **Weighted Random Patterns**
  4. **Mapping Logic / Bit Flipping**
  5. **Test Point Insertion (see DFT chapter)**



① MP-LFSR

- $f_0(x)=x^4+x^3+1$
  - $S_9 = Q_0$ =1
  - $S_8 = Q_1$ =1
  - $S_7 = Q_2$ =1
  - $S_6 = Q_3$
  - $S_5 = Q_3 \oplus Q_0$ =1
  - $S_4 = Q_3 \oplus Q_1 \oplus Q_0$ =1
- No solution!

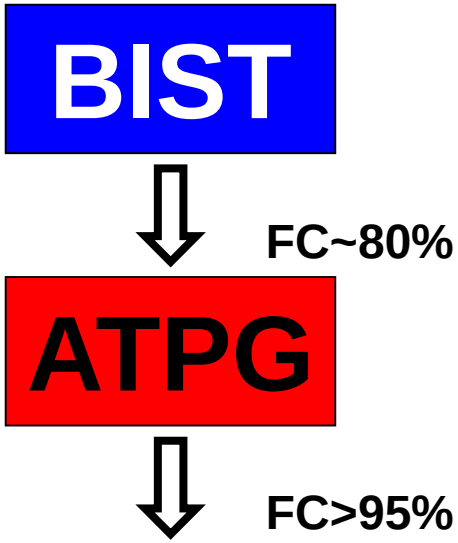
- $f_1(x)=x^4+x+1$
  - $S_9 = Q_0$ =1
  - $S_8 = Q_1$ =1
  - $S_7 = Q_2$ =1
  - $S_6 = Q_3$
  - $S_5 = Q_1 \oplus Q_0$ =1
  - $S_4 = Q_1 \oplus Q_2$ =1
- No solution!

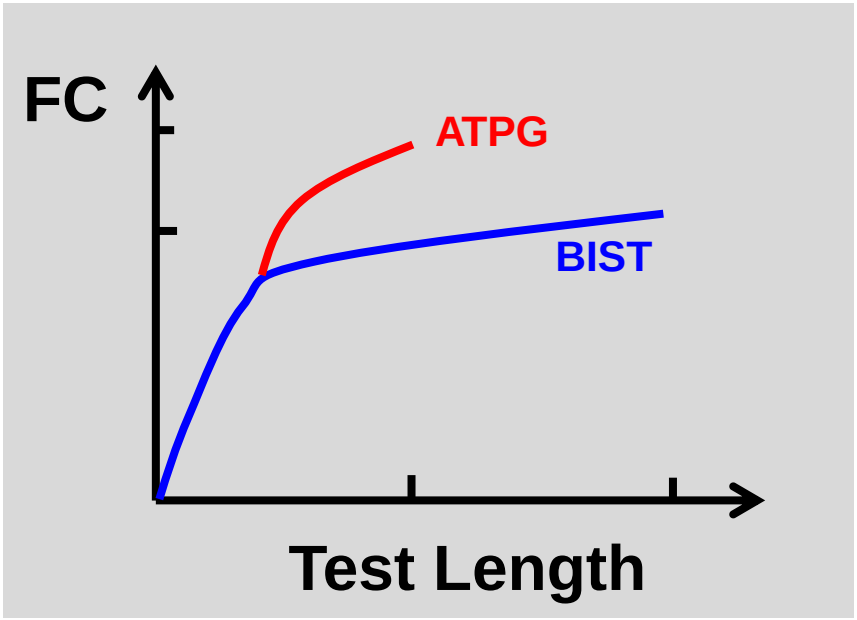


Too Many Specified Bits!

② Top-up ATPG Patterns

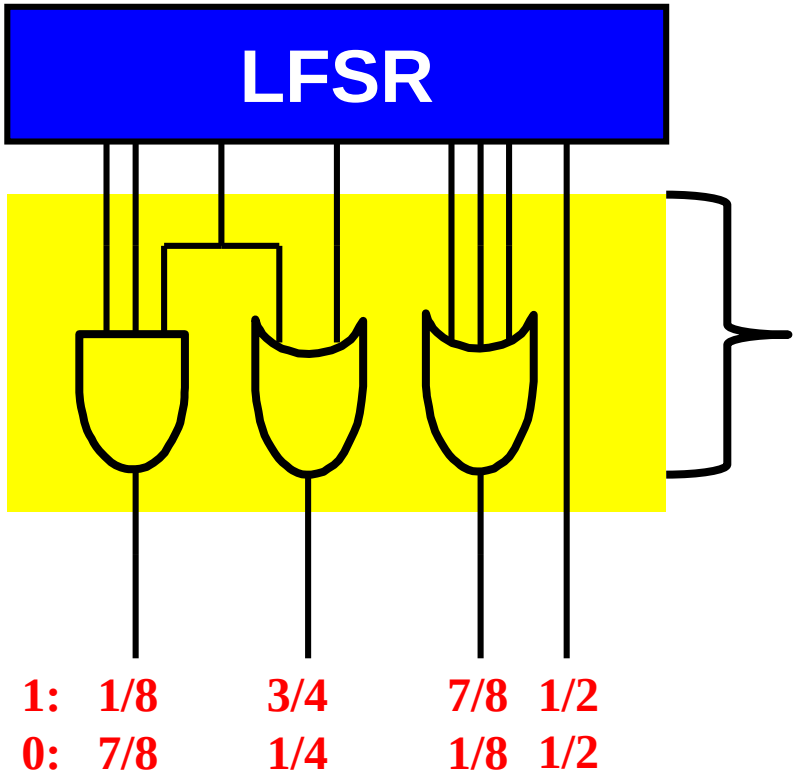
- *Mixed-mode BIST* = mixed BIST with ATPG
  - Run BIST to detect easy-to-detect faults
  - Load top-up patterns from ATE to detect hard-to-detect faults
- Good compromised solution
  - High fault coverage at cost of small ATE memory





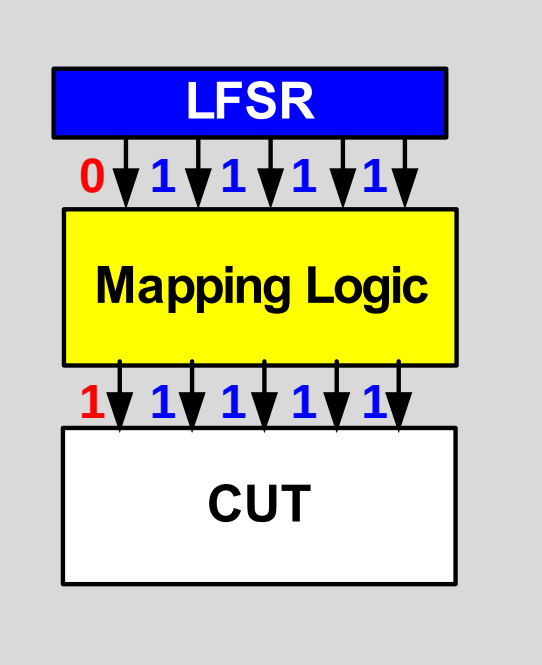
③ Weighted Random Pattern [Schnurmann 75]

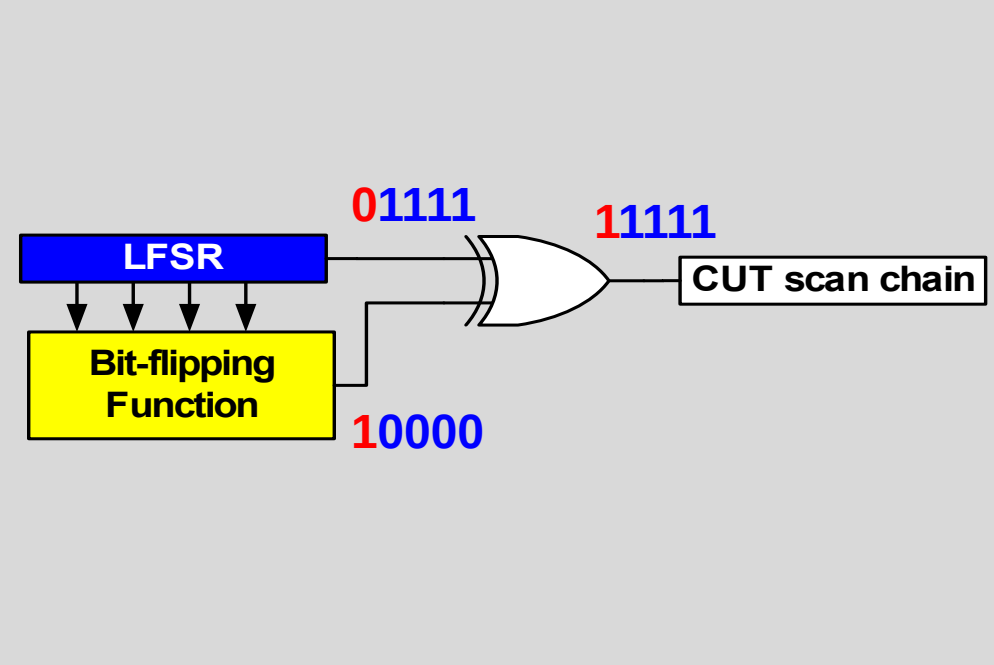
- Add weighting logic to bias probability of zeros and ones



④ Mapping Logic [Touba 95] Bit-Flipping [Wunderlich 96]

- Insert logic that converts
  - original LFSR pattern to desired test patterns

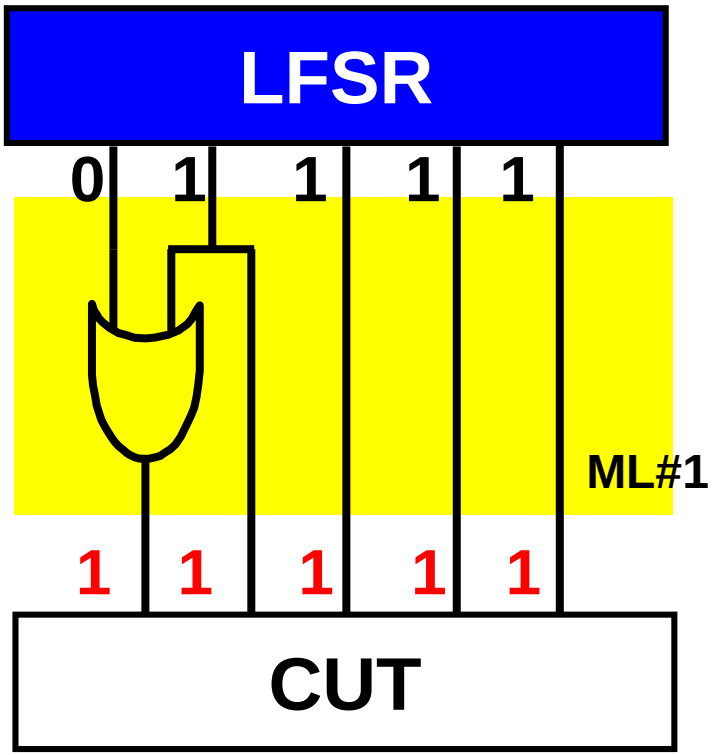


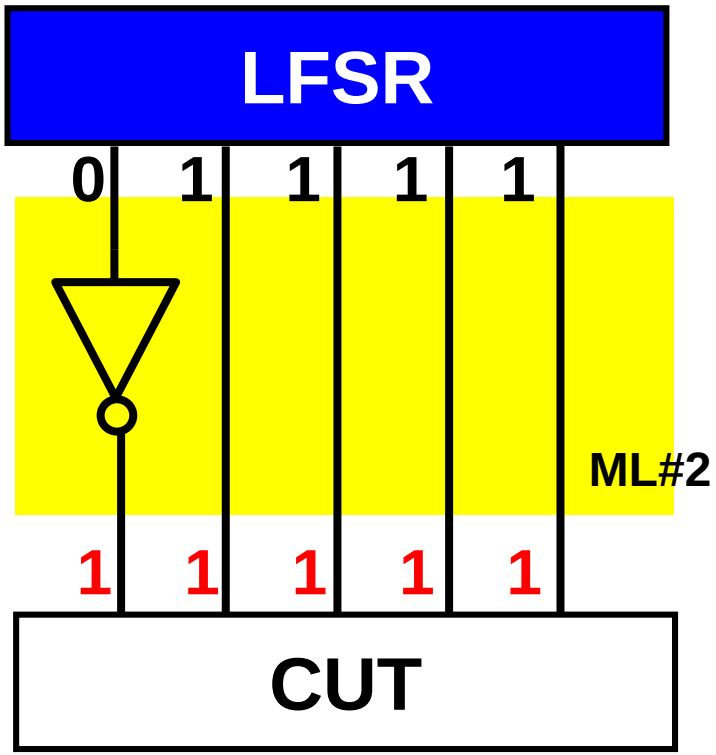


Minimize Area Overhead, Maximize Fault Coverage

Quiz

Q: Two mapping logic to generate '11111'. Which is better?
ANS:



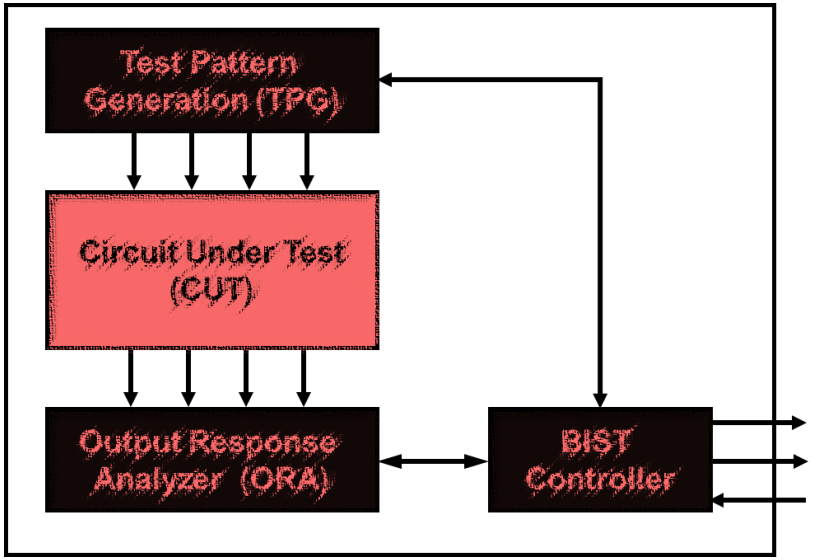


Fault Coverage Problem - Summary

	Structural Dependency	Linear Dependency	Random Pattern Resistant Faults
Reason of FC loss	LFSR phase shift among channels	LFSR feedback polynomials	Too many specified bits required
Location of spec. bits	Different scan chains	Same scan chain	Different or same scan chain
Solutions	Phase shifter Top-up ATPG patterns Map-logic / Bit-flip Test point insertion	MP-LFSR Top-up ATPG patterns Weighted random pat. Map-logic / Bit-flip Test point insertion	Top-up ATPG patterns Weighted random pat. Map-logic / Bit-flip Test point insertion MP-LFSR

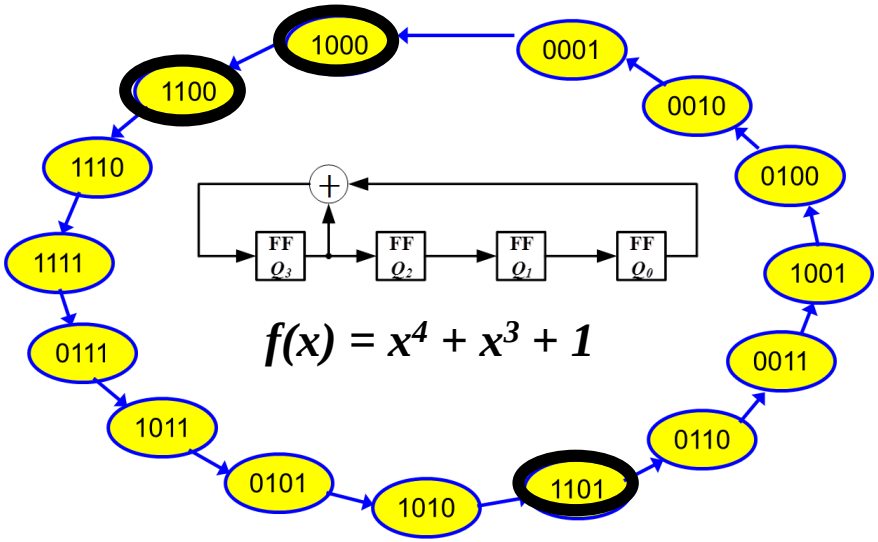
BIST Part 2

- Introduction
- Pattern Generation
- Output Response Analysis
- BIST Architecture
- **Problems and Solutions**
  - **Fault Coverage Not Enough**
  - **Long Test Length**
  - **Unknown Output Responses "X"**
  - **Diagnosis/Debug**
  - **High Test Power**
- **Concluding Remarks**



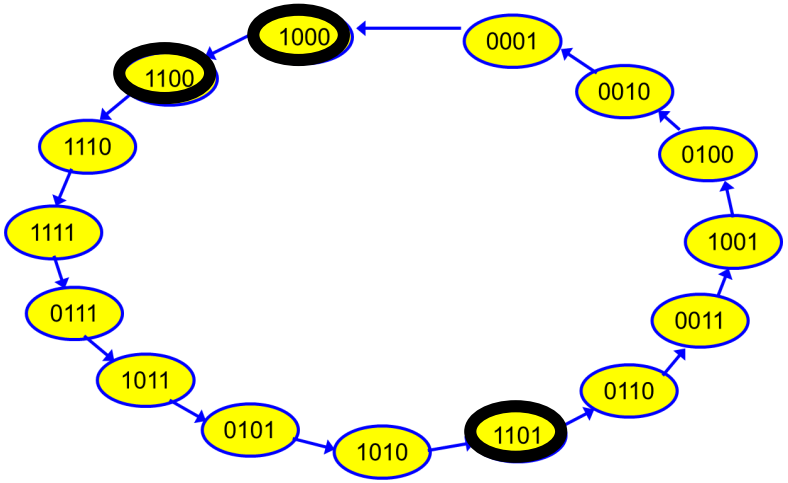
Long Test Length

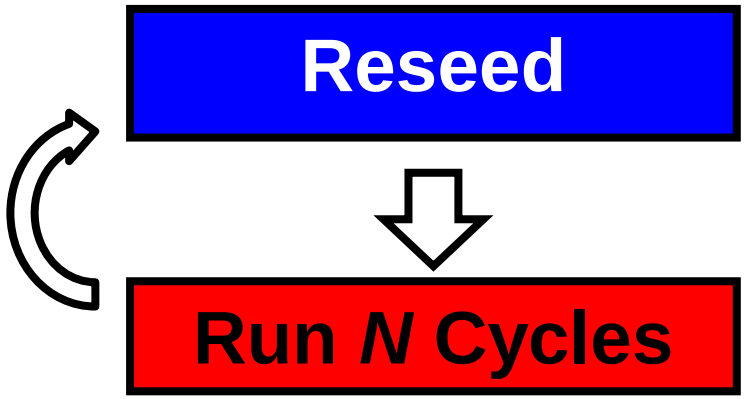
- BIST usually requires more patterns than traditional ATPG
- Example: want three patterns '1000, 1100, 1101'
  - Need 9 cycles. Many useless patterns.
- How to shorten BIST test length?
  - ① MP-LFSR, Top-up ATPG Patterns, Mapping logic …
  - ② Reseeding

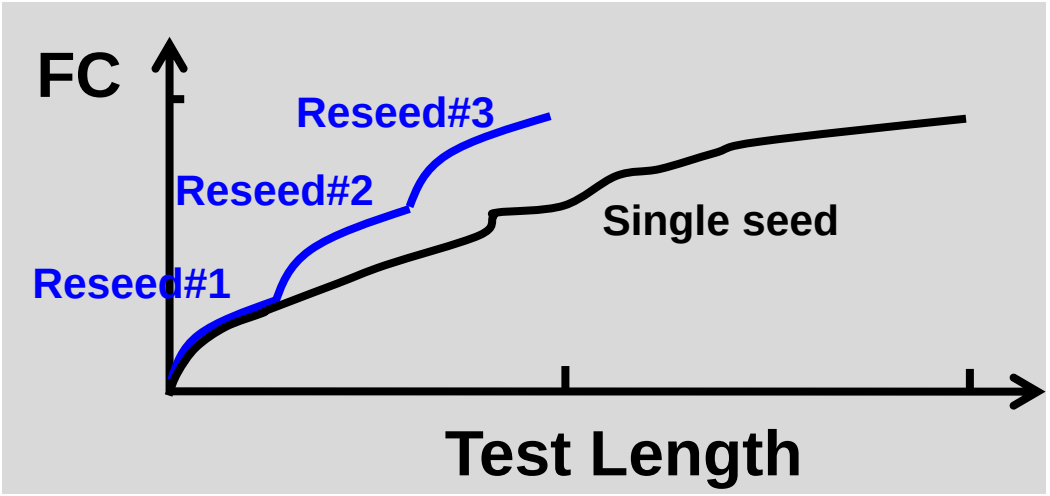


Reseeding [Könemann 91]

- Download new seeds to improve fault coverage in short time
- Example: want three patterns '1000, 1100, 1101'
  - Initial seed '1000', apply 2 patterns
  - reseed '1101', apply 2 patterns
  - Only 4 cycles

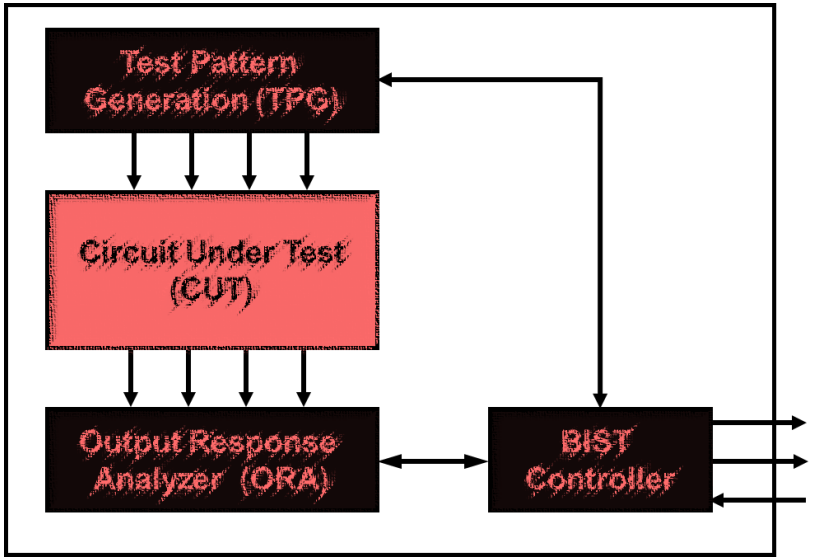






BIST Part 2

- Introduction
- Pattern Generation
- Output Response Analysis
- BIST Architecture
- **Problems and Solutions**
  - **Fault Coverage Not Enough**
  - **Long Test Length**
  - **Unknown Output Responses "X"**
  - **Diagnosis/Debug**
  - **High Test Power**
- **Concluding Remarks**



Unknown Output Response "X"

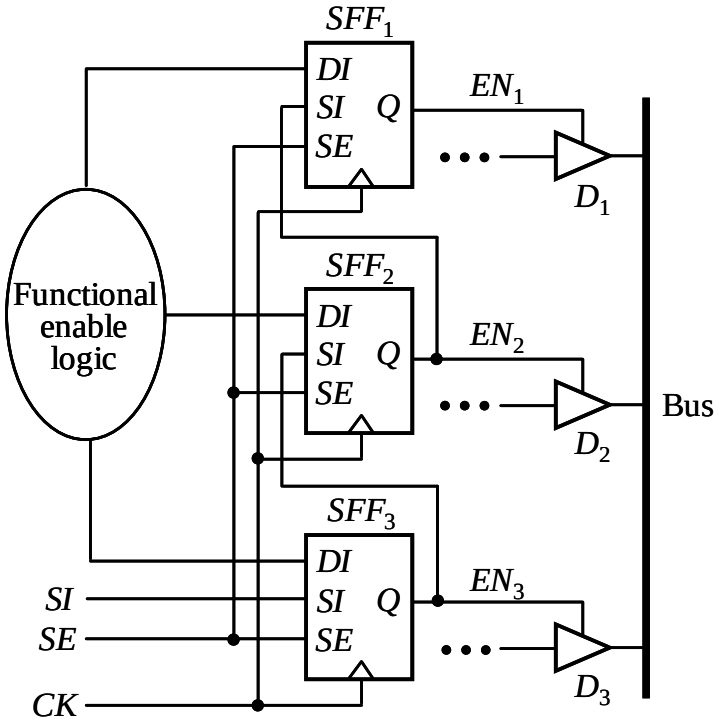
- Sources of unknown CUT output responses
  - ① Tri-state bus contention
  - ② Memory
  - ③ Undriven dangling inputs
  - ④ Uninitialized non-scan FFs
  - ⑤ Combinational loops

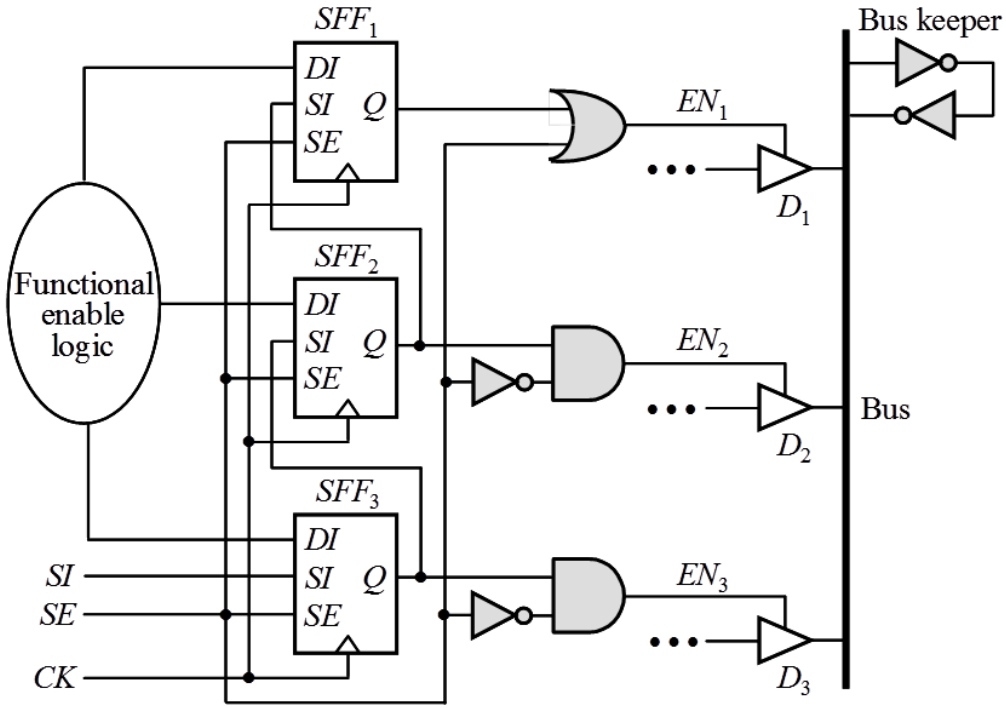
  ... many others
- Solutions
  - ① Control Point Insertion (also see DFT chapter)
  - ② Memory Wrapper
  - ③ Masking Logic

FFT: Why X Should Avoided in BIST?

① Control Point Insertion (Review 11.7)

- **Avoid X from**
  - **1) bus contention during scan shift, 2) undriven bus**





Original design

Testable design

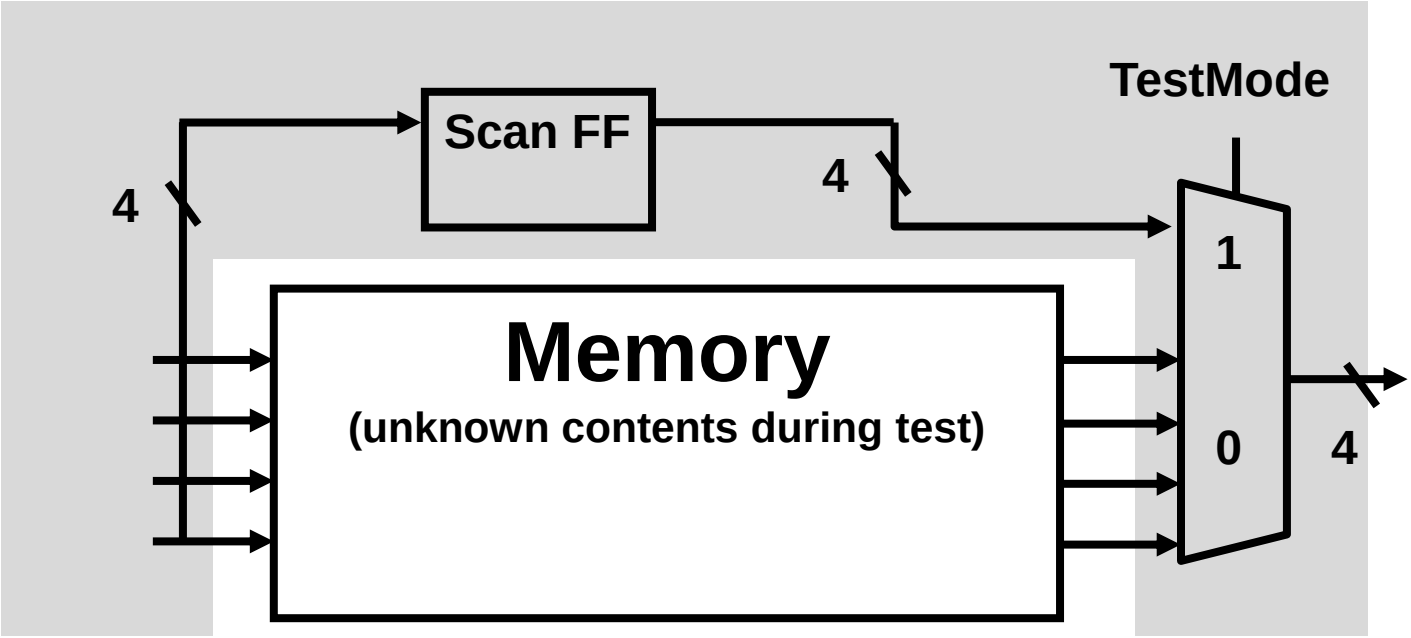
No bus contention when SE=1

No floating bus

(WWW Fig 2.21)

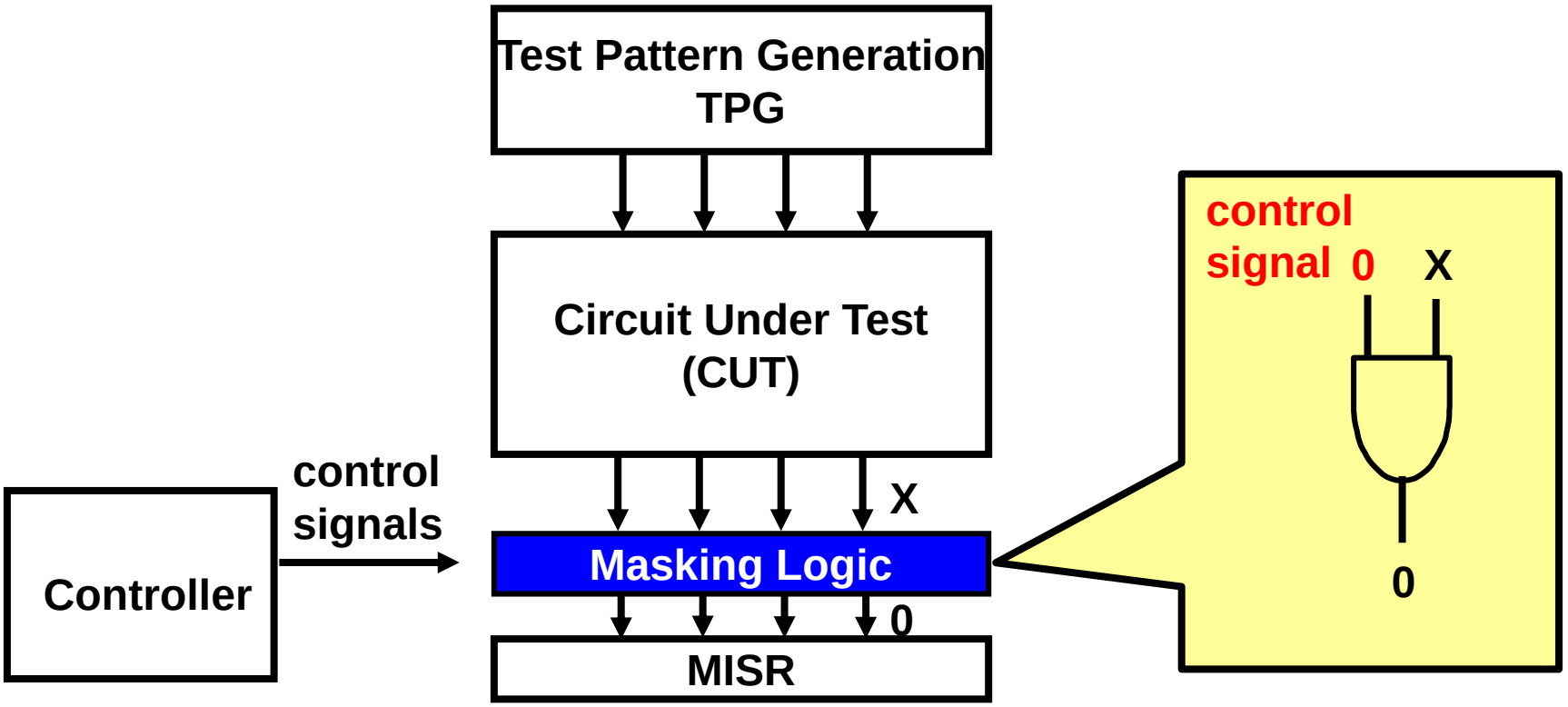
② Memory Wrapper

- Memory inputs are unobservable, output uncontrollable
- What should we do?
  - Add wrapper around memory
  - When TestMode=1, bypass memory



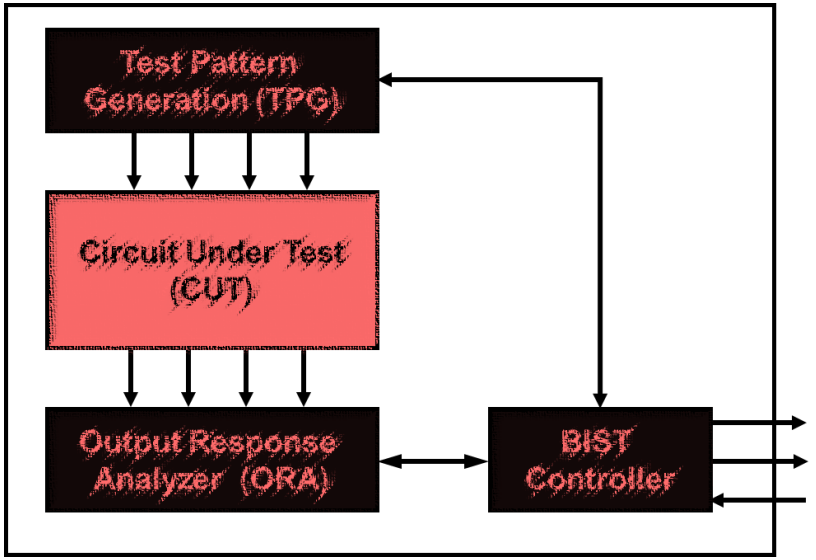
③ Masking Logic

- Add Masking logic before MISR



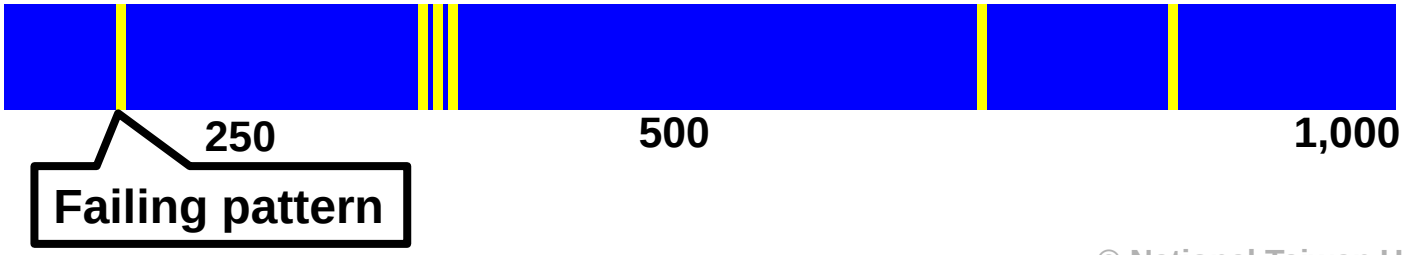
BIST Part 2

- Introduction
- Pattern Generation
- Output Response Analysis
- BIST Architecture
- **Problems and Solutions**
  - **Fault Coverage Not Enough**
  - **Long Test Length**
  - **Unknown Output Responses "X"**
  - **Long Test Time**
  - **Diagnosis/Debug**
  - **High Test Power**
- **Concluding Remarks**



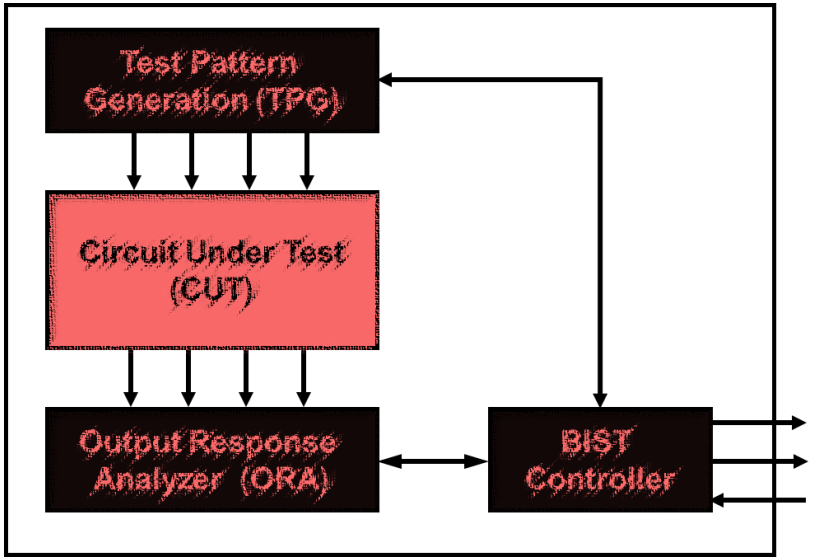
Diagnosis / Debug in BIST Mode

- Difficulty: CUT outputs not directly observable during BIST
  - Where are failing patterns? What are failing outputs?
- Binary search procedure:
  - 1. Run BIST, pause at specified pattern, unload signature
    - if signature is correct, doubles pattern counter
    - if signature is wrong, halves pattern counter
  - 2 Repeat step 1 until only one failing pattern found
  - 3. Run BIST to failing pattern, unload scan outputs
- Problems
  - Needs lots of pre-computed intermediate signatures
  - Very time consuming



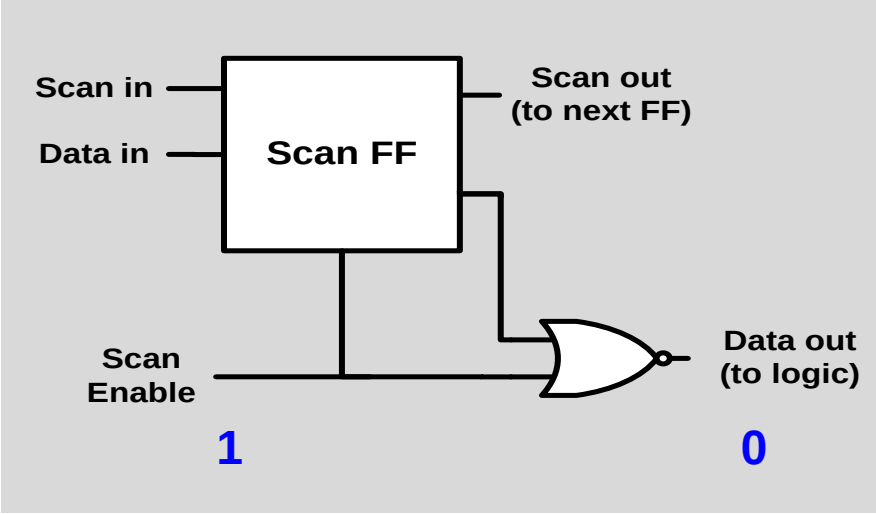
BIST Part 2

- Introduction
- Pattern Generation
- Output Response Analysis
- BIST Architecture
- **Problems and Solutions**
  - **Fault Coverage Not Enough**
  - **Long Test Length**
  - **Unknown Output Responses "X"**
  - **Long Test Length**
  - **Diagnosis/Debug**
  - **High Test Power**
- **Concluding Remarks**



BIST is Too "HOT"!

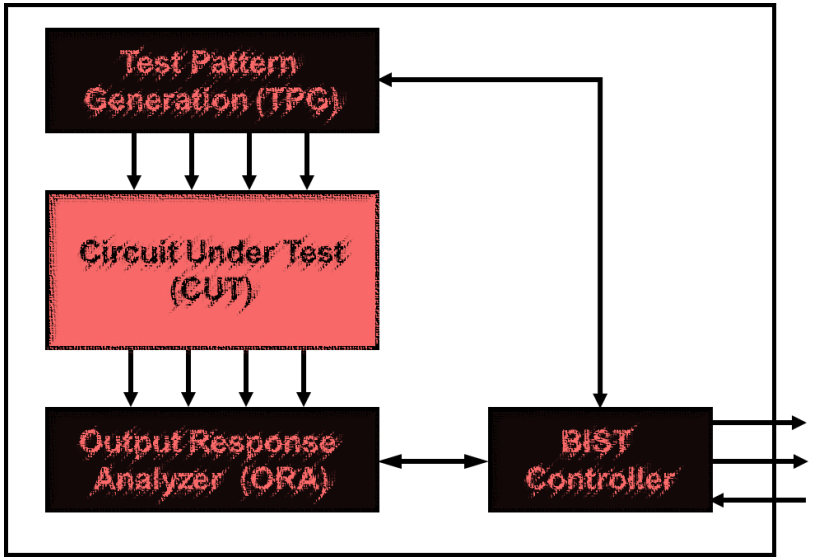
- Power dissipation much higher in BIST mode than normal operation
  - 2x to 10x higher! [industry estimation]
- Solutions
  1. Lower clock frequency during test
  2. Lower test voltage
  3. Reject high power test patterns
  4. Low power DFT, such as *Toggle suppression*



Many Other Solutions to Reduce Test Power

BIST Part 2

- Introduction
- Pattern Generation
- Output Response Analysis
- BIST Architecture
- **Problems and Solutions**
  - **Fault Coverage Not Enough**
  - **Long Test Length**
  - **Unknown Output Responses "X"**
  - **Long Test Time**
  - **Diagnosis/Debug**
  - **High Test Power**
- **Concluding Remarks**



Benefits of BIST

- Less expensive tester
  - Less pattern storage, less pins
  - Slower tester speed
- At-speed testing
- Easier integration of tests
  - Good for system-on-chip (SOC) testing
    - Many *intellectual properties* (IP) from different vendors
- Easier test access to embedded blocks or cores
  - *Embedded memory* testing
- Enables on-line testing

Myths about BIST

- BIST is NOT a replacement for scan DFT
  - BIST is built on top of scan
  - BIST also needs many test points
- BIST does NOT remove need for testers
  - Tester still required to
    - initiate test, apply top-up ATPG patterns,…
- BIST does NOT necessarily reduce test time
  - BIST requires lots of test patterns
- BIST does NOT necessarily reduce test cost

Real BIST Cases [Hetherington 99]

Item	ASIC1	ASIC2	ASIC3	ASIC4
Gate count	180K	356K	550K	748K
No. of Scan Cells	9K	20K	33K	41K
No. scan chains	80	128	128	120
No. Control points added	50	337	444	592
No. Observe points added	70	800	1,200	1,200
BIST area overhead (%)	1.3	1.3	1.0	0.8
Scan insertion time (hr)	0.7	3.3	6.7	14.5
Fault sim. Time (hr)	0.9	3.4	5.2	4.0

*from TI and Mentor Graphics

Real BIST Case (cont'd)

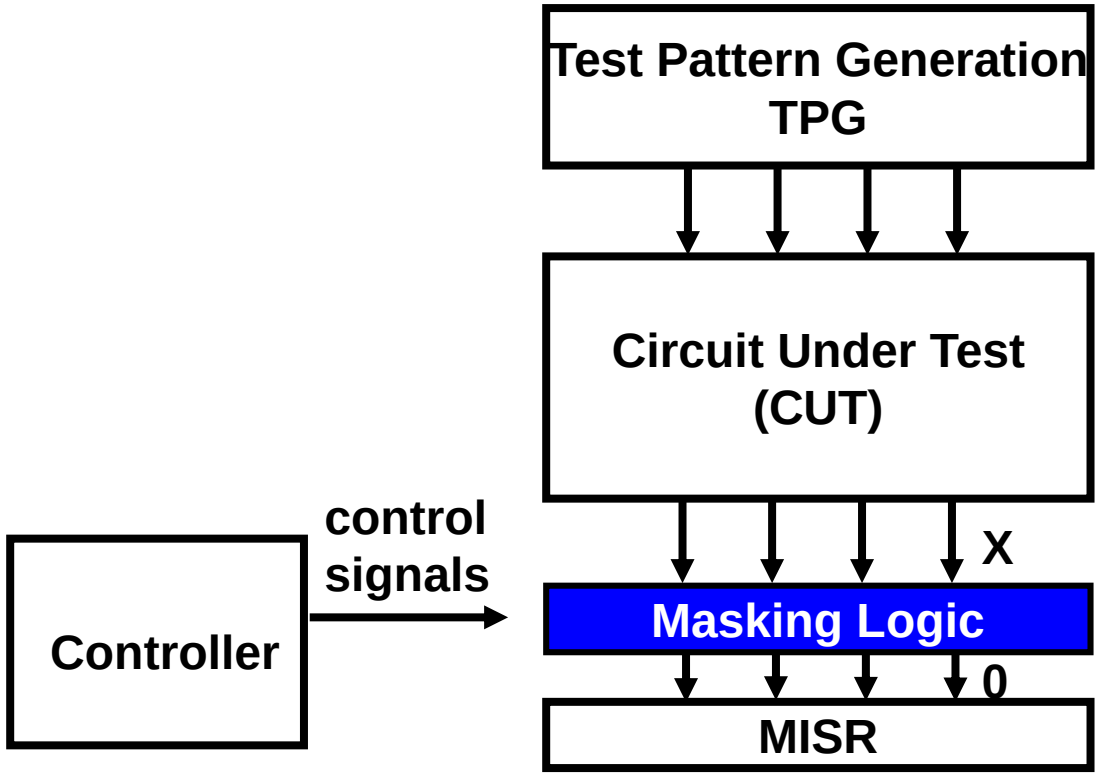
Item	ASIC1	ASIC2	ASIC3	ASIC4
Gate count	180K	356K	550K	748K
BIST pattern count	65K	262K	262K	262K
ATPG pattern count	2.5K	17K	13K	20K
BIST SSF Cov.(%)	96.0	95.7	95.3	95.6
ATPG SSF Cov.(%)	97.8	97.8	97.2	97.9
BIST frequency (MHz)	125	75	75	75
ATPG frequency	50	40	20	50
BIST test time (sec.)	0.06	0.58	0.93	1.2
ATPG test time (sec.)	0.02	0.36	0.94	0.7
BIST volume (MB)	0	0	0	0
ATPG volume (MB)	24	344	451	828

Summary

- **Fault Coverage Not Enough**
  - **MP-LFSR, Top-up ATPG, Weighted random, Map-logic/ Bit-flip**
- **Long Test Length**
  - **Reseeding**
- **Unknown Output Responses "X"**
  - **Control points, Memory wrapper, Masking logic**
- **Diagnosis/Debug**
  - **Binary search**
- **High Test Power**
  - **Low freq., low voltage, low power DFT …**

FFT

- Q: We were not worried about X in ATE,
  - why are we worried about X in BIST?



References

- [Bardell 82] P. H. Bardell and W. H. McAnney, “Self-Testing of Multichip Logic Modules,” Proc. IEEE Int. Test Conf., pp. 200-204, Nov. 1982.
- [Bardell 87] P.H. Bardell, W. H. McAnney, J, Savir, “Built-in Test for VLSI: Pseudorandom Techniques.” Wiely 1987.
- [Könemann 79] B.Könemann, J.Mucha, G.Zwiehoff, “Built-In Logic Block Observation Techniques”, IEEE Int’l Test Conference, 1979.
- [Könemann 91] B Könemann, “LFSR-coded test patterns for scan designs,” European Test Conference, 1991.
- [Hellebrand 92] S. Hellebrand, "GENERATION OF VECTOR PATTTERNS THROUGH RESEEDING OF MUETIPLE-POLYNOMIAL LINEAR FEEDBACK SHIFT REGIST," International Test Conference 1992.
- [Hetherington 99] Hetherington, G and Rajski and et. al., “Logic BIST for Large Industrial Designs: Real Issues and Case Study,” ITC pp.358-367, 1999.
- [McCluskey 85] E. J. McCluskey, "Built-In Self-Test Structures," IEEE Design & Test of Computers, vol. 2, no. 2, pp. 29-36, April 1985.
- [Schnurmann 75] H.D. Schnurmann , “The Weighted Random Test Pattern Generator ,” IEEE Trans. Comput., 1975.
- [Touba 95] N. A. Touba and E. J. McCluskey, "Transformed pseudo-random patterns for BIST," Proceedings 13th IEEE VLSI Test Symposium, Princeton, NJ, 1995.
- [Williams 88]T. Williams, and etl. Al., “Bounds and Analysis of Aliasing Errors in Linear Feedback Shift Registers,” Trans. CAD, Jan. pp. 73-85, 1988.
- [Wunderlich 96] H. J. Wunderlich, G. Kiefer “Bit-flipping BIST,” ICCAD 1996.