Design for Partially Strong Testability of RTL Data Paths to Guarantee Complete Fault Efficiency

Hiroyuki Iwata

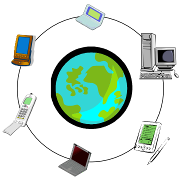
Computer Design & Test Laboratory

Outline

- Background
- Partially Strongly Testability
- Experimental Results
- Summary

Importance of VLSI testing

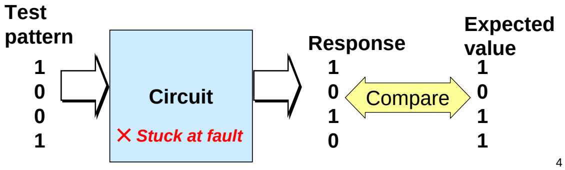
- **Ubiquitous computing**
  - Various tasks are performed anytime and anywhere



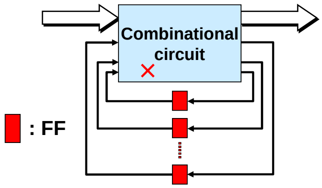
- **Testing of VLSI circuits**
  - Essential technology to realize dependable ubiquitous systems

VLSI Testing

- **VLSI Testing**
  - Check whether faults exist or not
- **Target fault**
  - Stuck at fault
- **Fault efficiency**
  - $$\frac{\text{\# of total faults}}{\text{\# of detected faults + \# of redundant faults}} \times 100$$



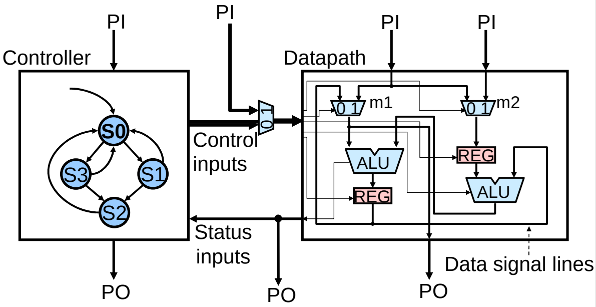
Test Generation



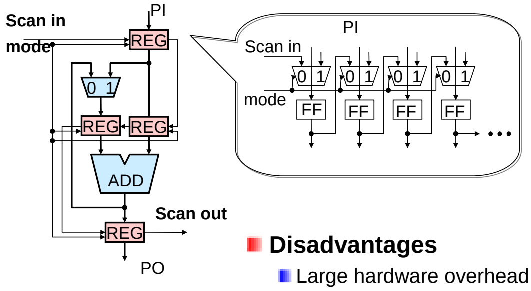
- The complexity of test generation is too difficult

Design for testability (DFT) is important to ease the complexity of the test generation

Register Transfer Level (RTL) Circuit



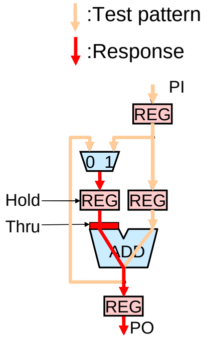
Full Scan Design



- **Disadvantages**
  - Large hardware overhead
  - Can not allow at-speed test

DFT based on Strong Testability [Wada at el. 1999]

- Conditions for Strong Testability
  - Any value can be set to inputs of hardware element *M* along a simple path
  - Any response of *M* is observable along a simple path
- Based on hierarchical test generation
  1. Test pattern generation
  2. Test plan generation



Research objectives

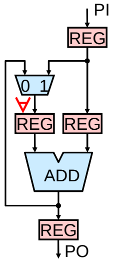
- Objective
  - 100% fault efficiency
  - At speed testing
- Strong testability [Wada et al. 1999]
  - Any value can be propagated to all the hardware elements

    ⇒Room for reduction of the hardware overhead
- Proposed method：partially strong testability
  - Any value can be propagated to some hardware elements

    ⇒To reduce the hardware overhead

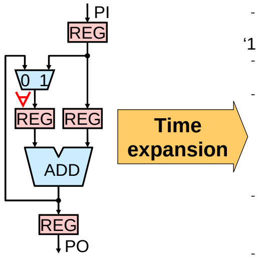
Partially strong testability (1/2)

- Conditions for Partially strong testability
  - Any value can be set to any loop along a simple path
  - Any value that appears in normal operation can be set to inputs of hardware element *M* along a simple path
  - Any value that appears in normal operation can be propagate from *M* to PO along a simple path

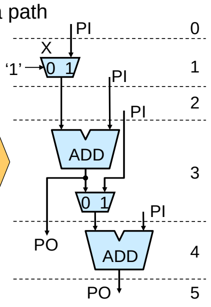


Partially Strongly Testability(2/2)

- If a data path is partially strongly testable, every fault in the data path can be detected on the time expansion model of the data path

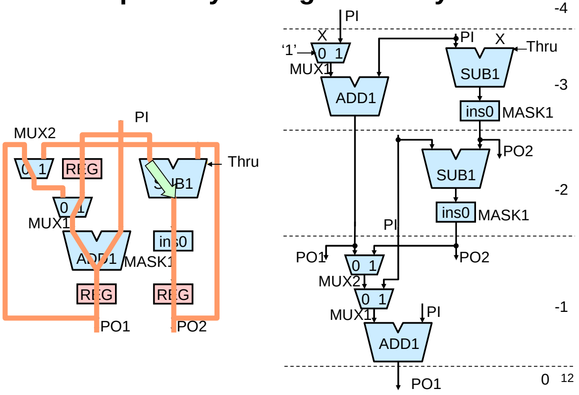


An example of partially strongly testable data path



Time expansion model

How to make a data path partially strong testability



Experiments

- Comparison methods
  - Original Circuit (ORG)
  - Full scan method (FS)
  - Strongly testable method [Wada et al. 1999](ST)
  - Proposed method (PST)
- Benchmark circuits
  - GCD, LWF, JWF, Paulin, Tseng, RISC, MPEG
- Synthesis tools
  - AutoLogic II (MentorGraphics)
- Test generation tools
  - TestGen (Synopsys)

Hardware overheads

	Hardware overhead [%]		
Circuits	FS	ST	PST
GCD	16.5	3.8	0.0
LWF	20.4	9.6	0.0
JWF	18.5	3.1	0.0
Paulin	9.0	3.5	1.8
Tseng	10.5	8.4	2.4
RISC	16.9	4.3	0.0
MPEG	14.0	8.0	0.8

Test generation results(1/2)

	Fault efficiency [%]				Test generation time [sec]			
Circuits	ORG	FS	ST	PST	ORG	FS	ST	PST
GCD	100.00	100.00	100.00	100.00	3.18	0.20	0.83	0.39
LWF	100.00	100.00	100.00	100.00	0.33	0.21	0.62	0.24
JWF	100.00	100.00	100.00	100.00	3.63	0.60	0.62	1.74
Paulin	99.23	100.00	100.00	100.00	297.50	0.83	0.86	4.69
Tseng	99.01	100.00	100.00	100.00	703.90	0.34	1.08	0.87
RISC	99.37	100.00	100.00	100.00	12210.81	55.17	76.56	60.02
MPEG	88.30	100.00	100.00	100.00	68947.90	1.84	1.18	13.34

Test generation results(2/2)

	Test application time [clock]			
Circuits	ORG	FS	ST	PST
GCD	133	2351	387	197
LWF	56	2674	250	70
JWF	244	14849	769	608
Paulin	147	2824	875	526
Tseng	590	4752	633	523
Risc	2271	621284	5520	3420
Mpeg	1216	185183	107359	8448

Summary

- Design for Partially Strong Testability of RTL Data Paths
  - 100% fault efficiency
  - At speed testing
  - Lower hardware overhead compared to strong testable method
  - Shorter test application time compared to strong testable method
- Future works
  - DFT method for the whole RTL circuit including a controller