CPU Testing & Testable Design

- Problems in CPU Testing
- Test Strategies for CPU
- Testable CPU Architectures
- Testable Design Flow

Problems in CPU Testing

- Large number of registers
- Large number of small buffers or queues
- Different sizes of memories
- Complex random logic (control path & data path)
- Cell library
- Board level testing
- Test control
- Test integration & Scheduling
- CAD tool supporting

Test Strategies

- Design hierarchy & Circuit partitioning
- BIST for large memories / arrays
- Special BIST for small buffers
- Scan for random logic
- Shadow registers where necessary
- Boundary Scan for test control and board level testing
- Functional testing
- Testable design rules
- CAD tool usage

Design Hierarchy & Circuit Partitioning

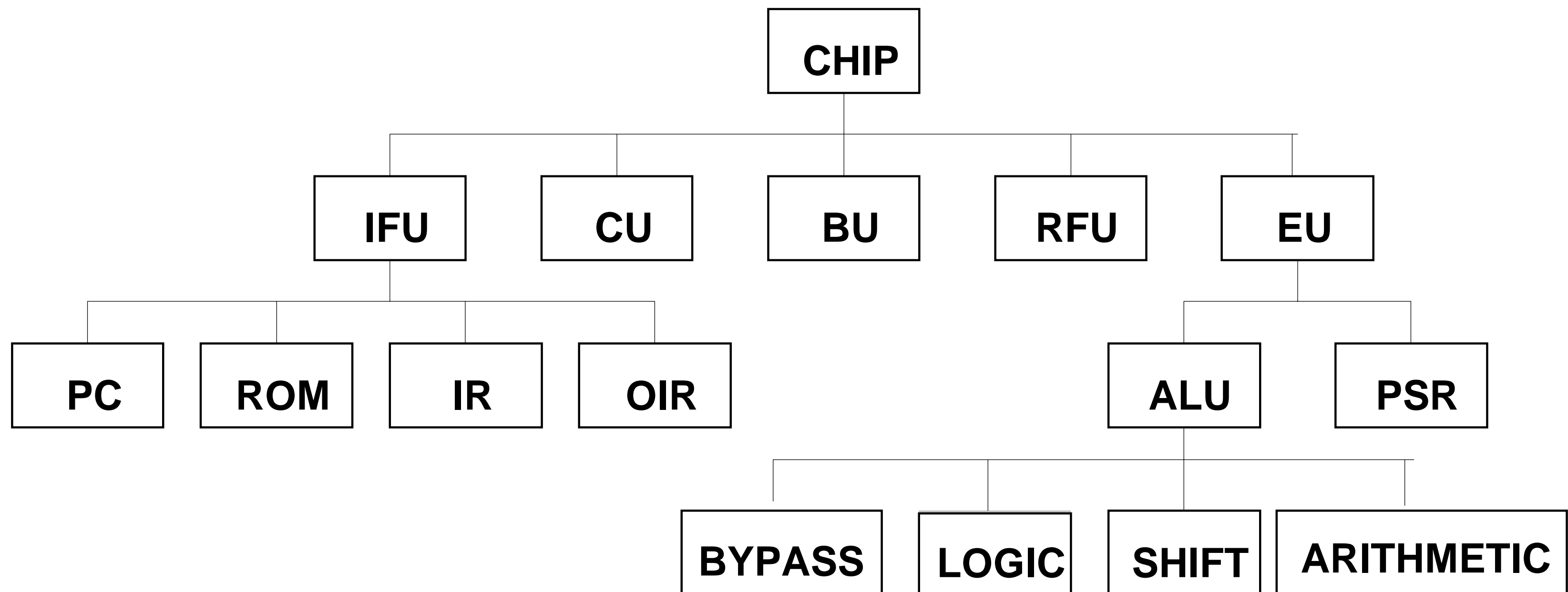
- Represent the design in a hierarchy fashion
- Try to identify all the bottom components
- Identify test strategy for each bottom component
- Go up the hierarchy from the bottom component, determine test strategy for the circuitry between components

Example: a simple pipelined CPU

- 4-level hierarchy
- 5 functional units

  - IFU: instruction fetch unit
  - CU: control unit
  - BU: bus unit
  - RFU: register file unit
  - EU: execution unit

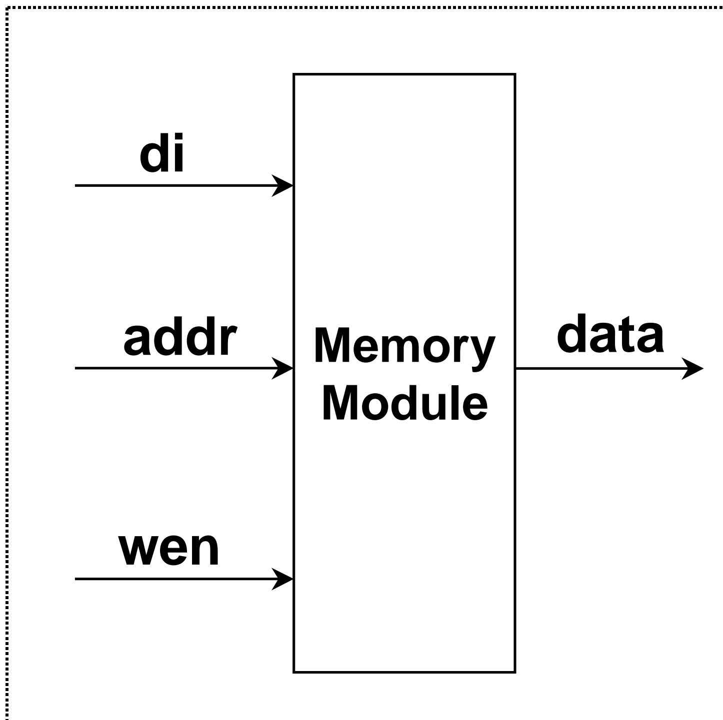
Example: a simple pipelined CPU (cont.)

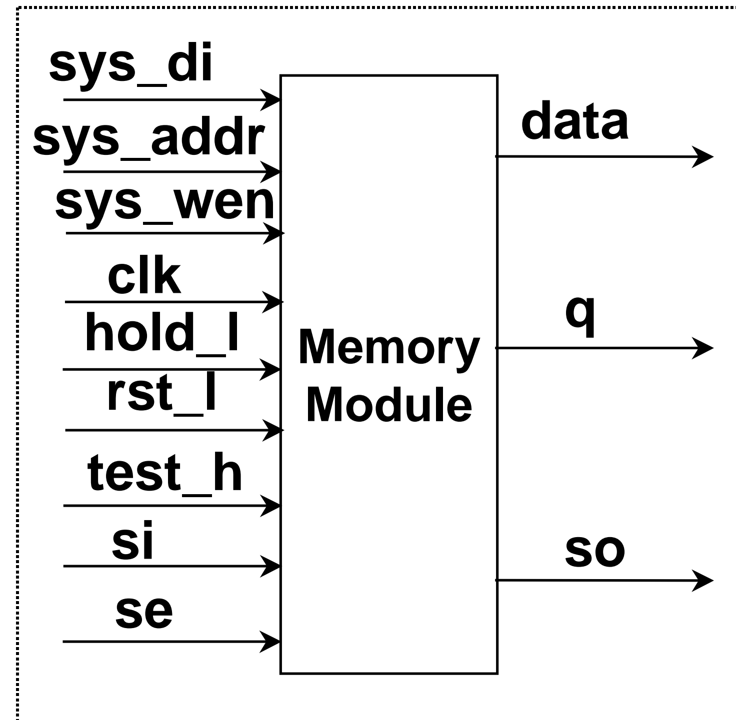


BIST for Large Memories / Arrays

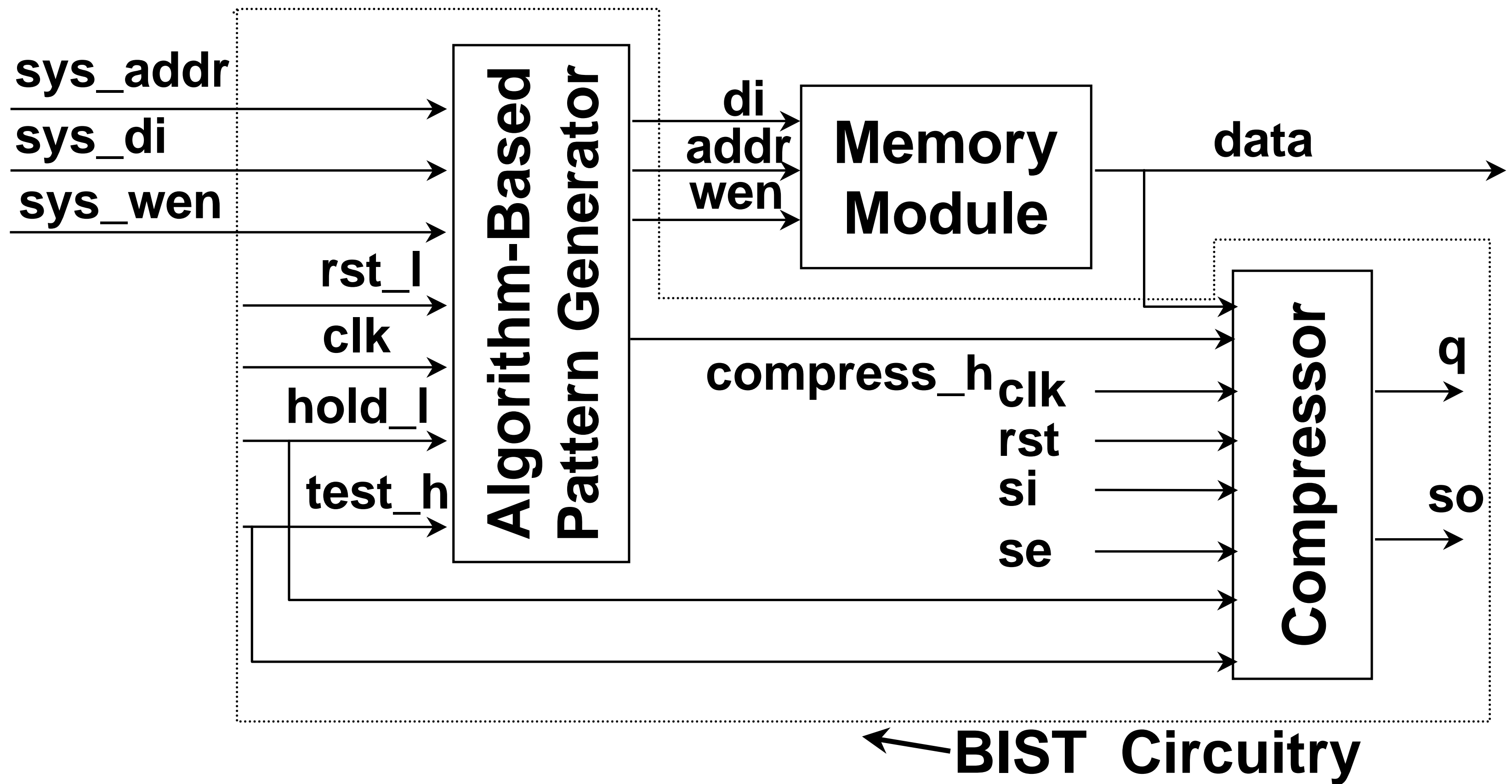
- Use BIST for all large memories
- Test all memories in parallel
- Deal with memories with different sizes
- Deal with memories with different widths of cells
- Use one test controller

Memory BIST Architecture with a Compressor

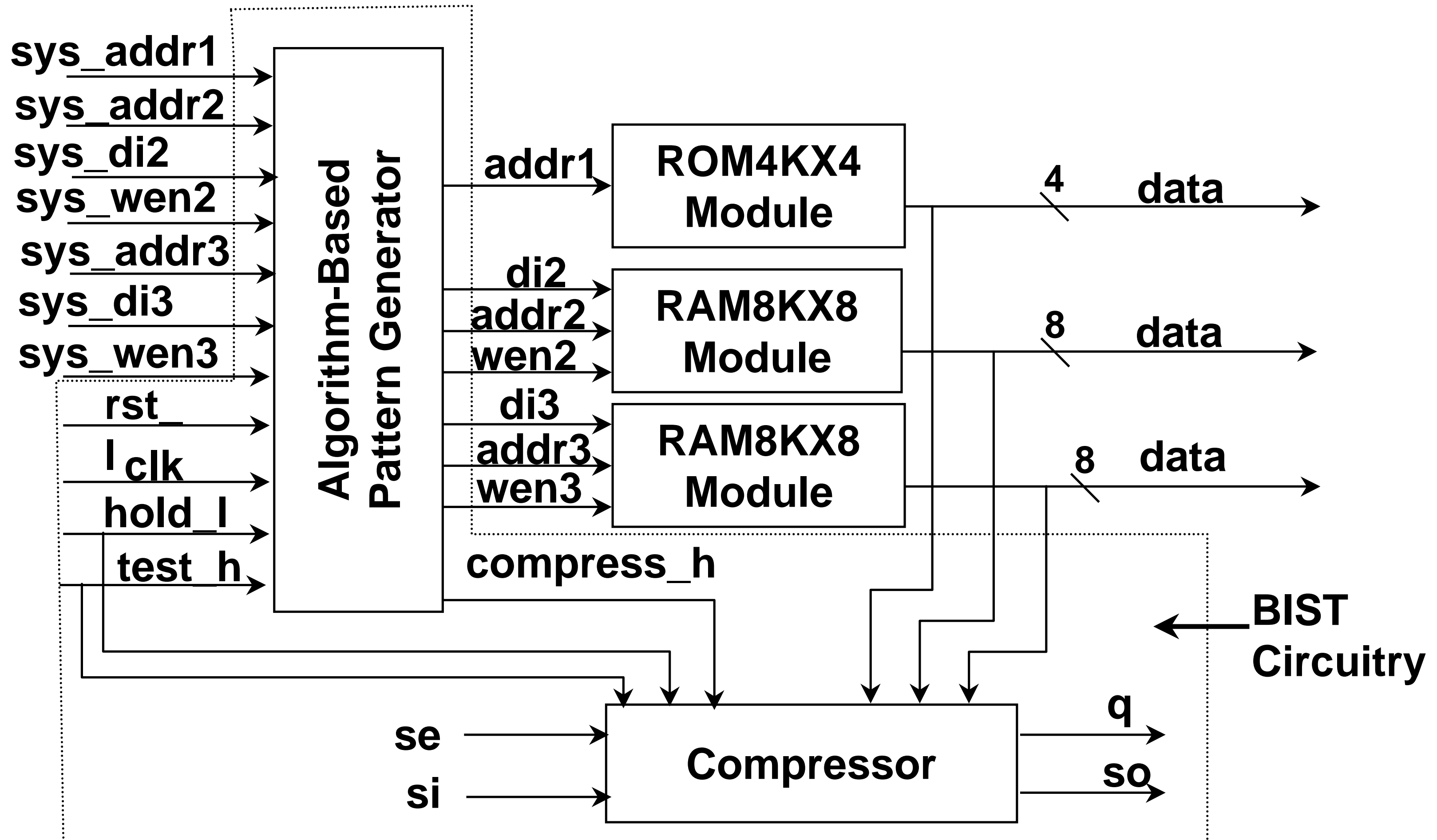




Memory BIST Architecture with a Compressor (cont.)



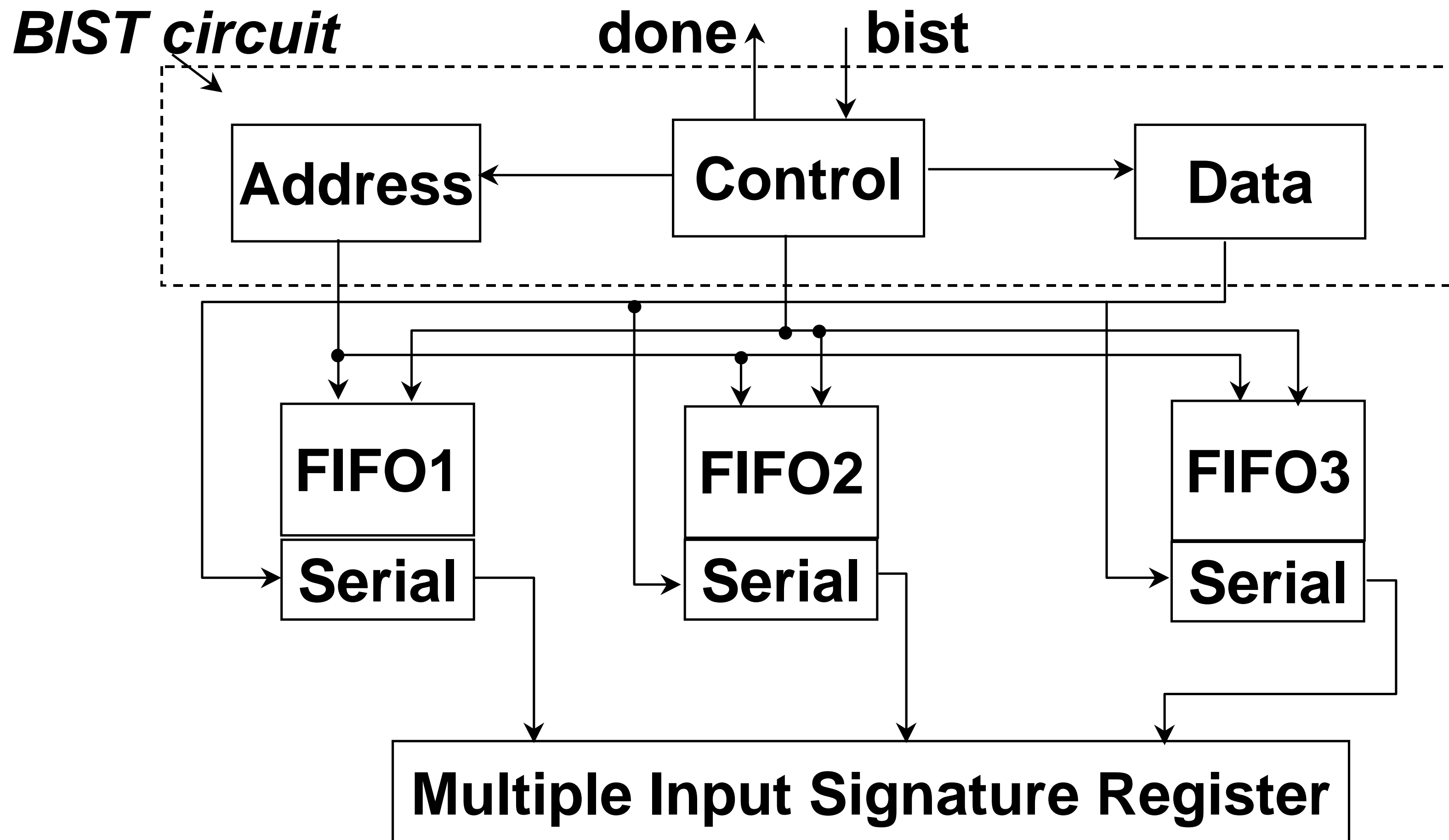
Three Memories and One Compressor



Small Buffer Testing

- Several FIFOs reside in a superscalar CPU
  - Instruction queue
  - Reorder buffer
  - Reservation station
- Memory cells as well as the control logic for these cells must be tested
- Need a special test architecture to test these buffers

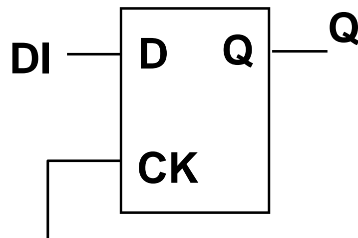
Parallel Test Architecture for Small Buffers

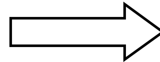


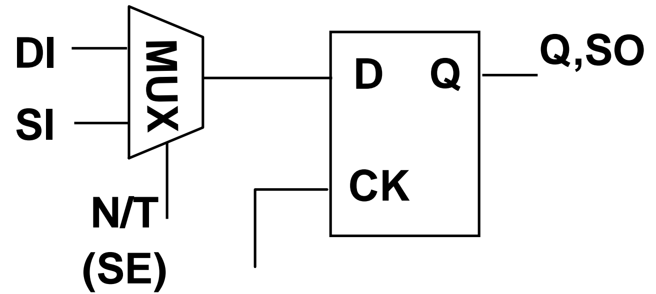
Scan for random logic

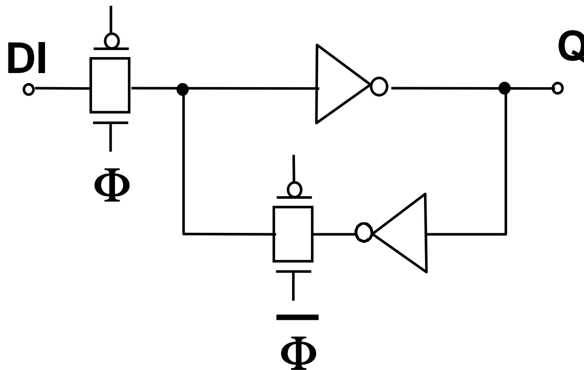
- Scan cell design
- Full scan v.s. partial scan
- Single scan chain v.s. multiple scan chain
- Test control and scheduling

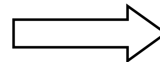
Scan Cell Design

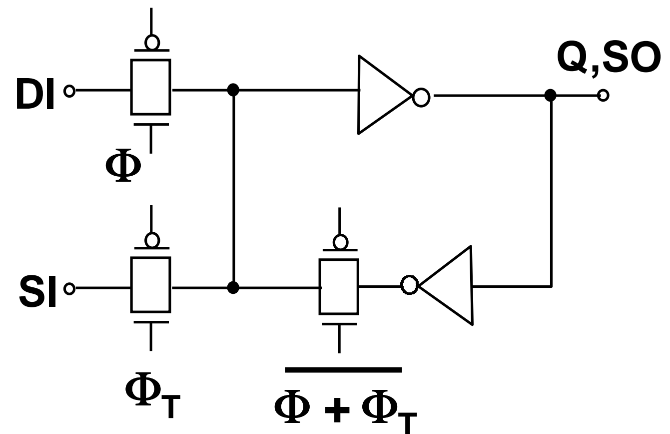




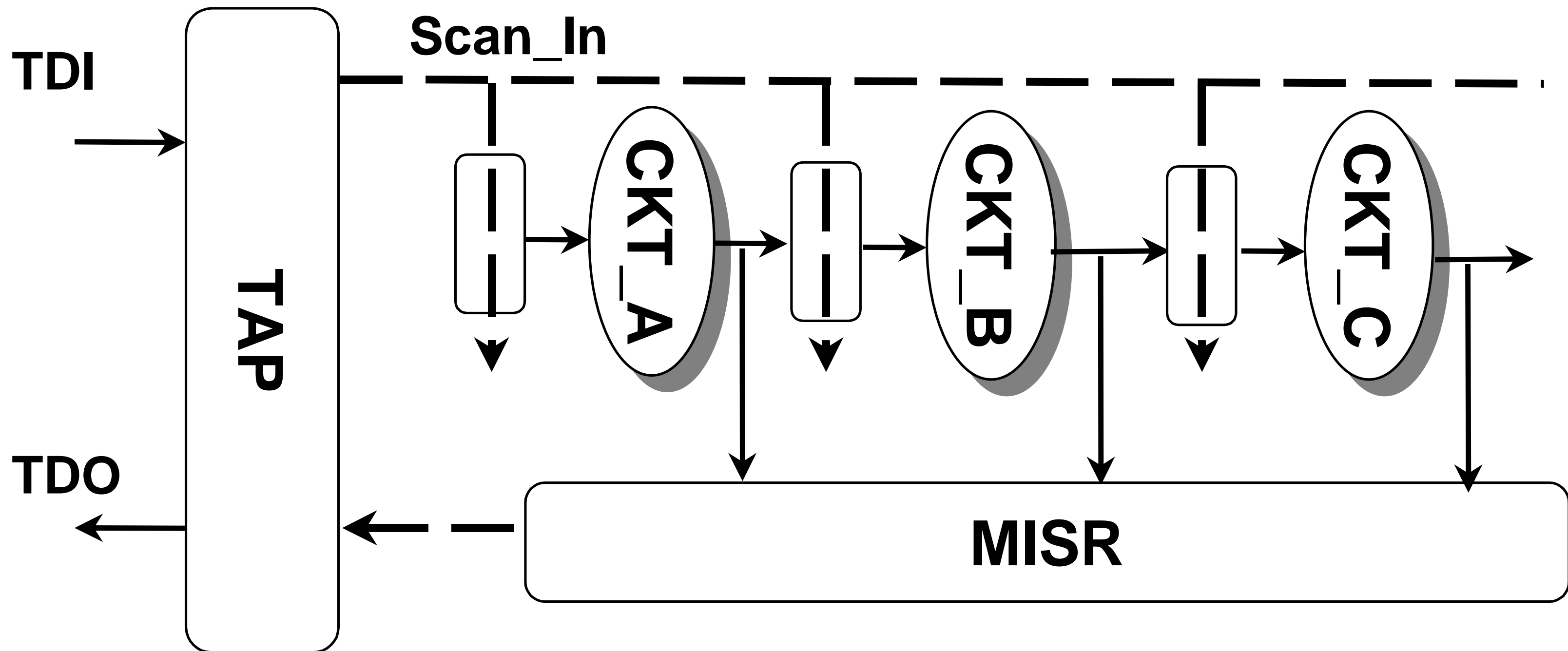








A possible method to drive multiple scan chains using a data input



Boundary scan for test control and board level testing

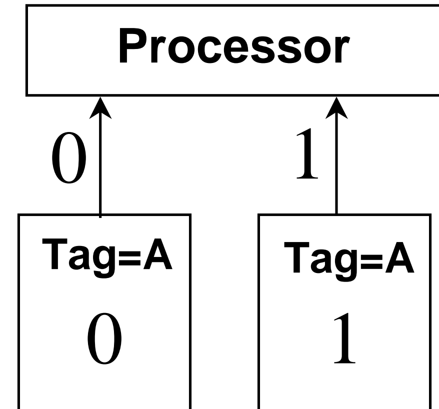
- Must provide test control signals and test data for
  - Scan testing (including shadow registers)
  - BIST
  - Small buffer testing
  - Any ad hoc techniques
- Must define some new Boundary Scan instructions
- Must determine required TMS signals

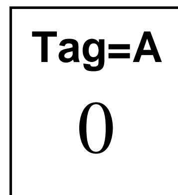
Functional testing

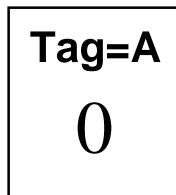
- Several circuitry, especially some interconnections are not tested by component testing
  - interconnections network
  - Interface between register file and ALU or other logic
  - Bus Interface unit
  - Cache control logic
  - Global chip function
- Functional testing is carried out by executing a sequence of instructions

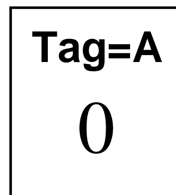
An example of functional testing for cache control

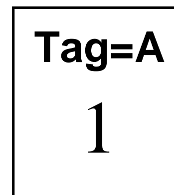
For functional fault
- Cache always miss











Cache (enable) | Main Memory | Cache (disable) | Main Memory (write 1 of tag A) | Cache (enable) | Main Memory

(read different data into processor)

Initial state | **Fault activation** | **Fault detection**

Testable design rules and requirements

- Important for full-custom design
- All custom design must have a corresponding Verilog RTL code
- If DFT circuitry is added by tools, then usually the rules are satisfied

Some design rules

(1) Synchronous design ONLY except reset

(2) Avoid gated clocks

(3) Must be able to break global feedback loop

(4) SET/RESET must not be driven by other FF/s

(5) ROM and RAM must be isolatable during testing

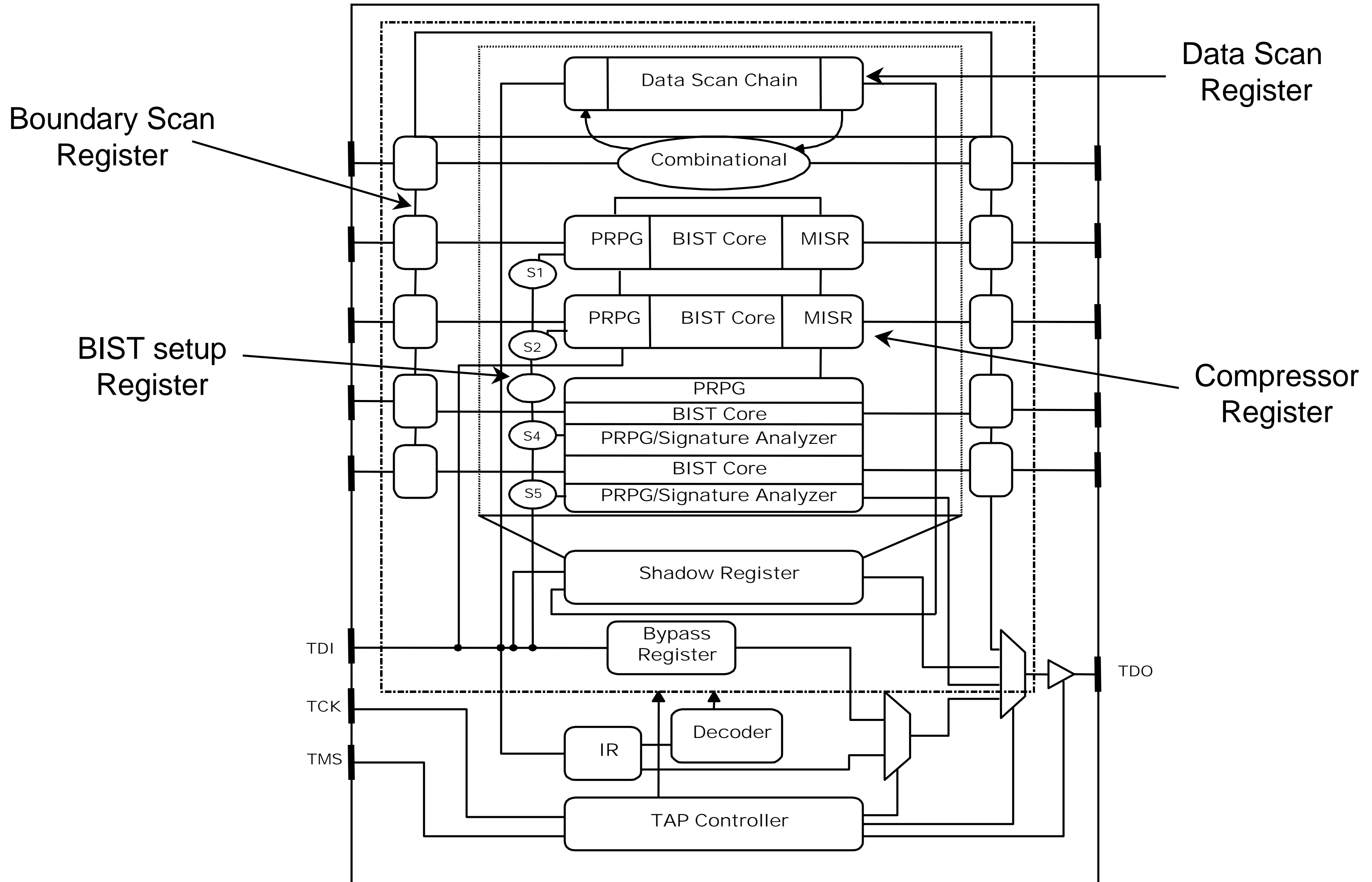
CAD Tools Usage

- Should try to use tools wherever possible
- Comparison between
  - Syntest
  - Mentor Graphics
  - Synopsys
- Usually tools are not turn-key solutions, much human work has to be done to generate
  - Data files
  - Control files
  - Interface between tools

Testable CPU architecture

- A generic global CPU test architecture
- A simple pipelined CPU
- Test control architecture for the pipelined CPU

A Global Test Control Architecture



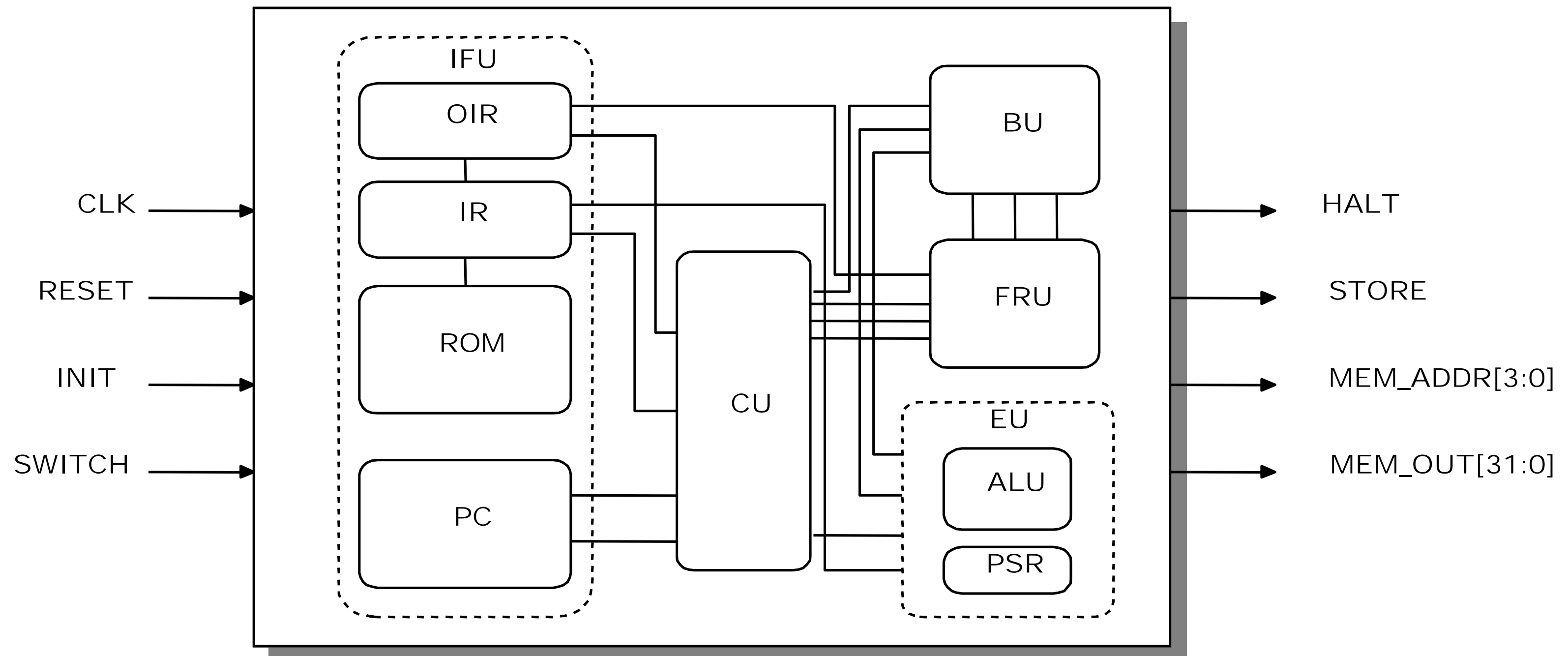
Boundary Scan Instruction Set

- 2 classes, 9 instructions

Class	Group	Instruction	Opcode	Target register
IEEE 1149.1 Instruction	Mandatory	BYPASS	0000	Bypass register
		SAM./PRELOAD	0001	Boundary scan register
		EXTEST	1111	Boundary scan register
	Optional	CLAMP	0010	Bypass register
		HIGHZ	0011	Bypass register
User defined instruction	SHADOW	SHASCAN	0100	Shadow register
	SCAN	INSCAN	0101	Data scan chain
	BIST	SETBIST	0110	Setup scan chain
		RUNBIST	0111	Compressor register

A Simple Pipelined CPU

- Input pins: CLK, RESET, INIT, SWITCH
- Output pins: HALT, STORE, MEM_ADDR, MEM_OUT



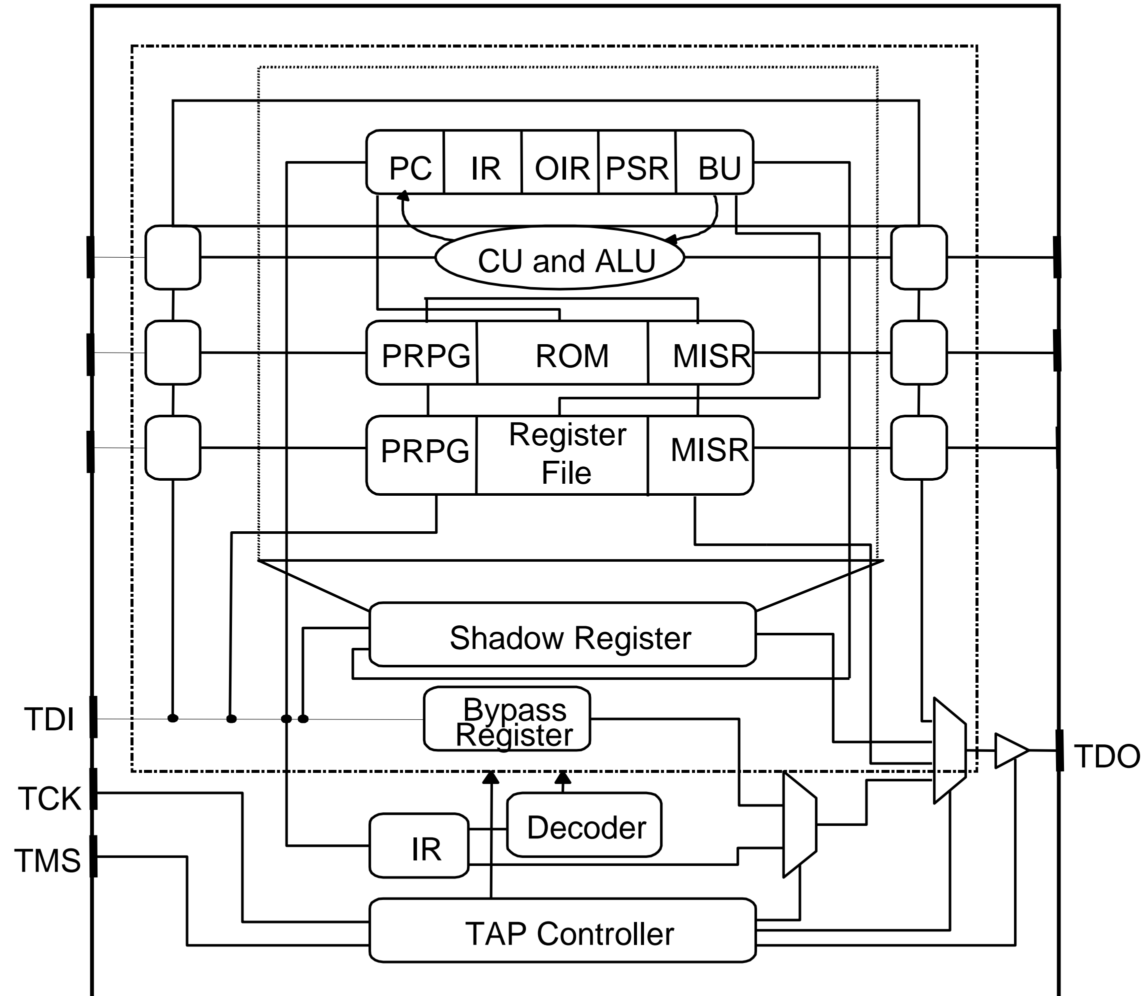
Instruction Set

Class	Inst.	Opcode	Format	Description
CTL	NOP	0000	NOP	No operation
	HALT	1111	HALT	Halt system
LD/ST	LOAD	0010	LOAD DST, #n	Load constant to register
		0010	LOAD DST,[mem]	Loadmem data to register
	STORE	0011	STORE mem, #n	Store constant tomemory
		0011	STORE mem, SRC	Store register data to memory
LOGIC	AND	1100	AND DST, SRC	SRC & DST to DST
	OR	1101	OR DST, SRC	SRC | DST to DST
	XOR	1110	XOR DST, SRC	SRC ^ DST to DST
	LSF	1000	LSF DST, SRC	SRC << DST to DST
	RSF	1001	RSF DST, SRC	SRC >> DST to DST
BRANCH	BRA	0001	BRA LABEL	Branch to label always
	BRN	0001	BRN LABEL	Branch to label if negative
	BRZ	0001	BRZ LABEL	Branch to label if zero
	BRP	0001	BRP LABEL	Branch to label if parity
	BRE	0001	BRE LABEL	Branch to label if even
	BRC	0001	BRC LABEL	Branch to label if carry
ARITH.	ADD	0100	ADD DST, SRC	SRC + DST to DST
	SUB	0101	SUB DST, SRC	SRC - DST to DST
	MUL	0110	MUL DST,SRC	SRC * DST to DST

Test Strategy

Blocks	Component	Architecture	Test strategy
IFU	PC	Random logic gates	Scan
	ROM	Memory elements	BIST
	IR	Register	Scan
	OIR	Register	Scan
EU	ALU	Combinational gates	Scan
	PSR	Register	Scan
FRU	R_FILE	Memory elements	BIST
CU	CU	Combinational gates	Scan
BU	BU	Random logic gates	Scan
Shadow Register	Shadow Register	Register	Shadow Sampling
			Scan
CHIP	CPU	Complex circuit design	Boundary Scan
			Functional testing

CPU Test Control Architecture



Boundary Scan Instruction Set

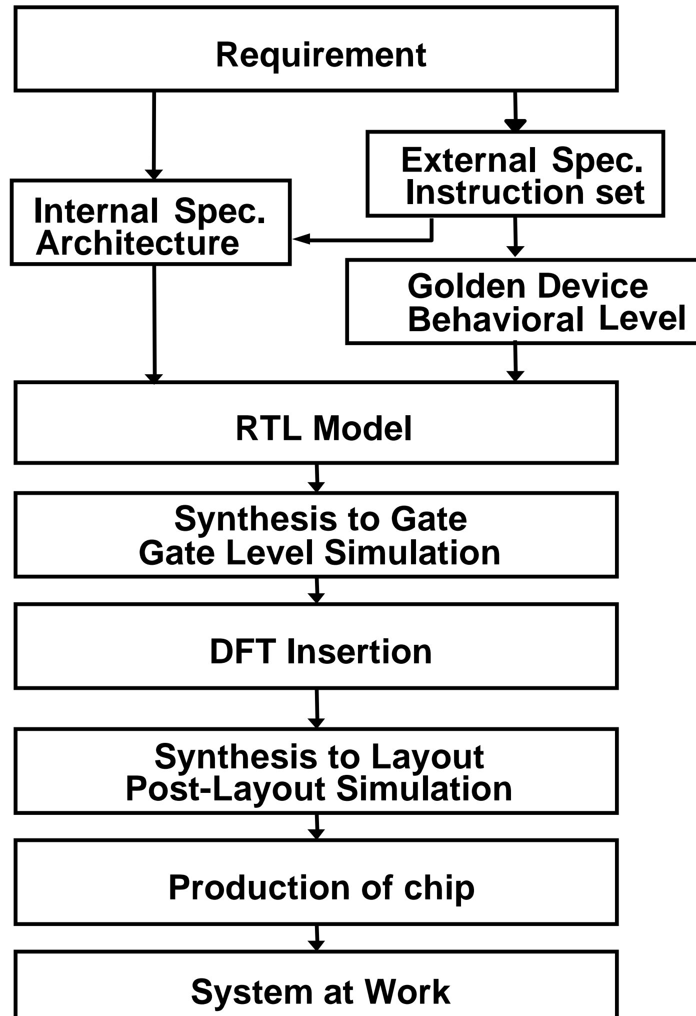
- **2 classes, 8 instructions**

Class	Group	Instruction	Opcode	Target register
IEEE 1149.1 Instruction	Mandatory	BYPASS	0000	Bypass register
		SAM./PRELOAD	0001	Boundary scan register
		EXTEST	1111	Boundary scan register
	Optional	CLAMP	0010	Bypass register
		HIGHZ	0011	Bypass register
User defined instruction	SHADOW	SHASCAN	0100	Shadow register
	SCAN	INSCAN	0101	Data scan chain
	BIST	RUNBIST	0110	Compressor register

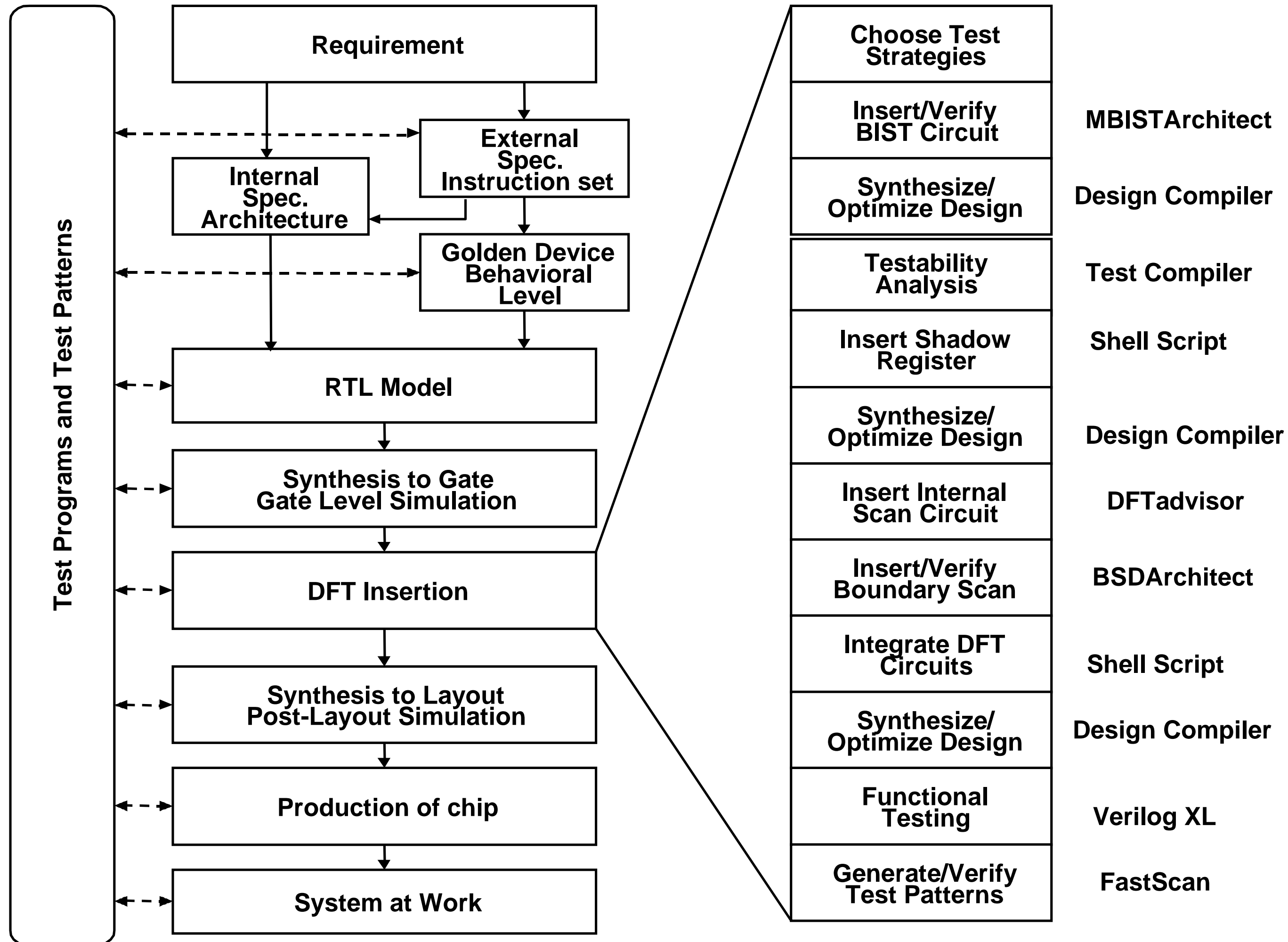
Testable Design Flow

- Traditional design flow
- Testable design flow at gate-level
- DFT instruction phases
- Testable design flow at RTL/gate level
- Automatic DFT insertion
- Test Scheduling

Traditional Design Flow (cell-based)



Testable Design Flow (gate-level)

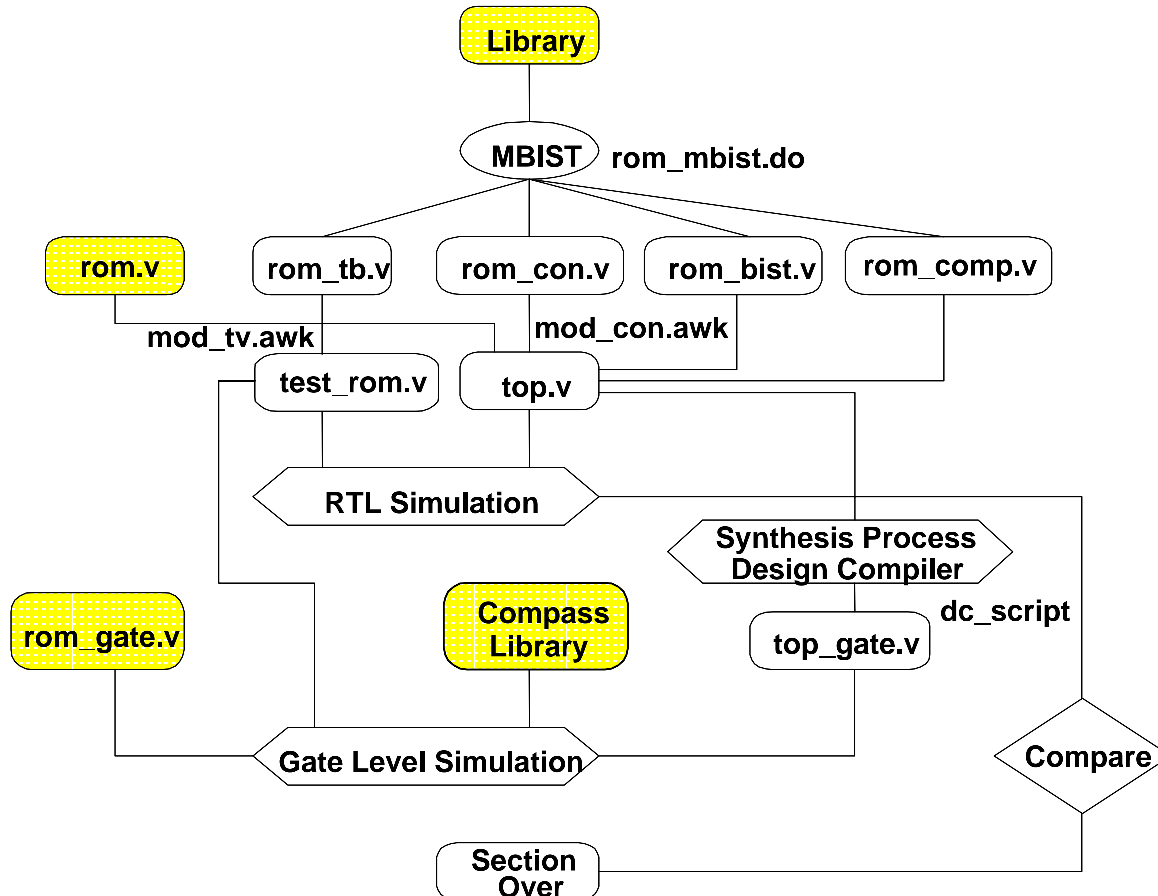


DFT Insertion Phase

- Choose test strategy
- Insert/Verify BIST circuit
- Synthesize/optimize design
- Testability analysis
- Insert shadow register
- Insert internal scan circuit
- Synthesize/optimize design
- Insert/Verify Boundary Scan
- Integrate DFT components
- Synthesize/optimize design
- Functional testing
- Generate/verify test patterns

Memory BIST Insertion

- Automatic RTL BIST insertion
- MBISTArchitect and batch program



Shadow Register Insertion

- Test points selection
  - ⇨ Testability analysis
    - - Gate level
    - - Test Compiler

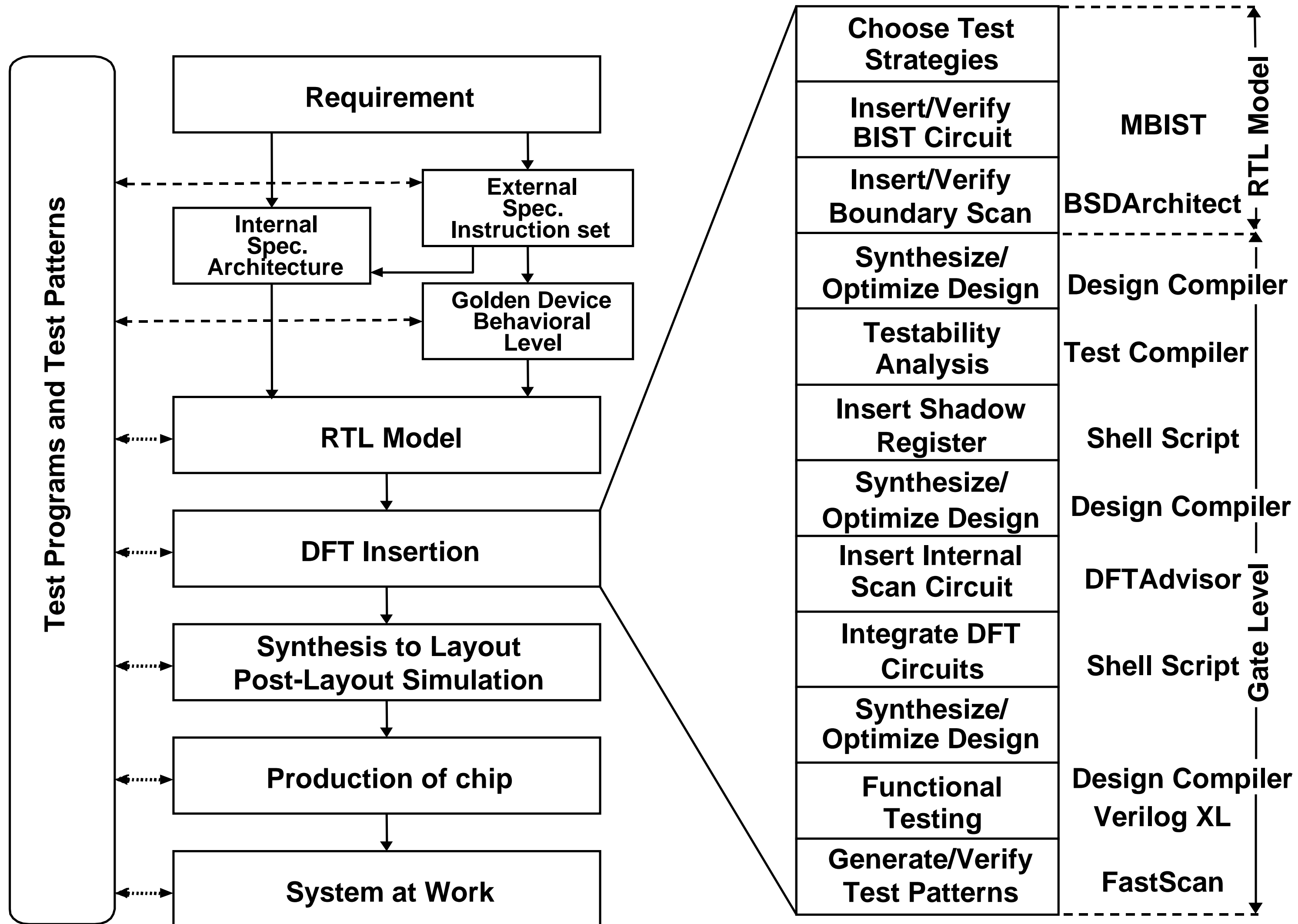
    Circuit characteristic
- Automatic shadow register insertion
- Shadow register generator

    C-shell script
- RT-level insertion

Scan Chain Insertion

- Automatic scan chain insertion
  - Gate level
  - Scan cell identification
  - Scan cell connection
- DFTAdvisor or Test Compiler
  - 328 scan cells

Testable Design Flow (RTL/Gate)

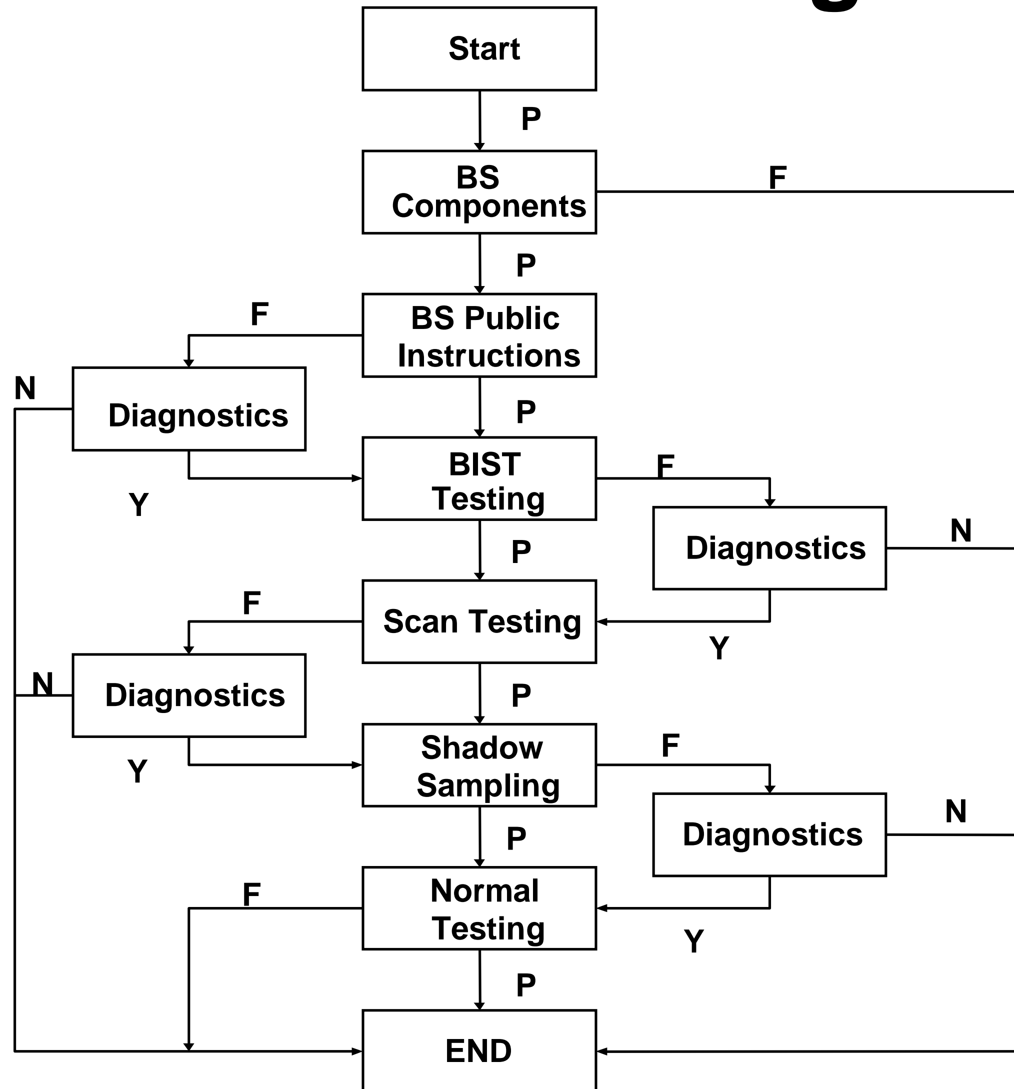


Automatic DFT Insertion

- **Programs list**

Step	Programs	Functions
Test Program	program_gen	Code translation NOP insertion
Pipelined CPU Synthesis	Design Compiler	Synthesis/Optimization
	Batch Files	Batch execution
BIST Insertion	MBISTArchitect	BIST synthesis
	mon_tn.awk	Test fixture modification
	mon_con.awk	Top module modification
	Design Compiler	Synthesis/Optimization
	Batch files	Batch execution
Testability analysis	Test Compiler	Testability analysis
Shadow Register Insertion	shadow_gen	Shadow Register synthesis
Scan Chain Insertion	DFTAdvisor	Scan chain synthesis
	Batch file	Batch execution
Boundary Scan Insertion	BSDArchitect	Boundary Scan synthesis
	Batch file	Batch execution
DFT Integration	dft_gen	DFT integration
Test Fixture Generation	fixture_gen	Test fixture generation
Test Patterns Generation	FastScan	Test patterns generation
	Batch file	Batch execution
Tutorial Demo Program	dft_tutorial	Tutorial demo program

Test Scheduling



Conclusions

Most important tasks in CPU testing

- **Design partition - RTL codes**
- **Test training course**
- **Design rules / constrains**
- **Test architecture decision**
- **Close link between design team and test team**
- **Good management for coordination**
- **Test integration**
- **Functional testing**