Testing Self-Timed Circuits with MrGO

Swetha Mettala Gilla, Marly Roncken, Ivan Sutherland, Xiaoyu Song
Asynchronous Research Center, Portland State University
(mettalag@cecs.pdx.edu)

MOTIVATION

Why self-timed?
- Self-timed circuits offer modularity
- Self-timed circuits offer energy efficiency
- Self-timed circuits offer speed

What?
- Self-timed networks of state-holding **links** (▭)
- Exchange data at action-capable **joints** (☺)

Wanted:
- A **general test method** to initialize states and control actions for:
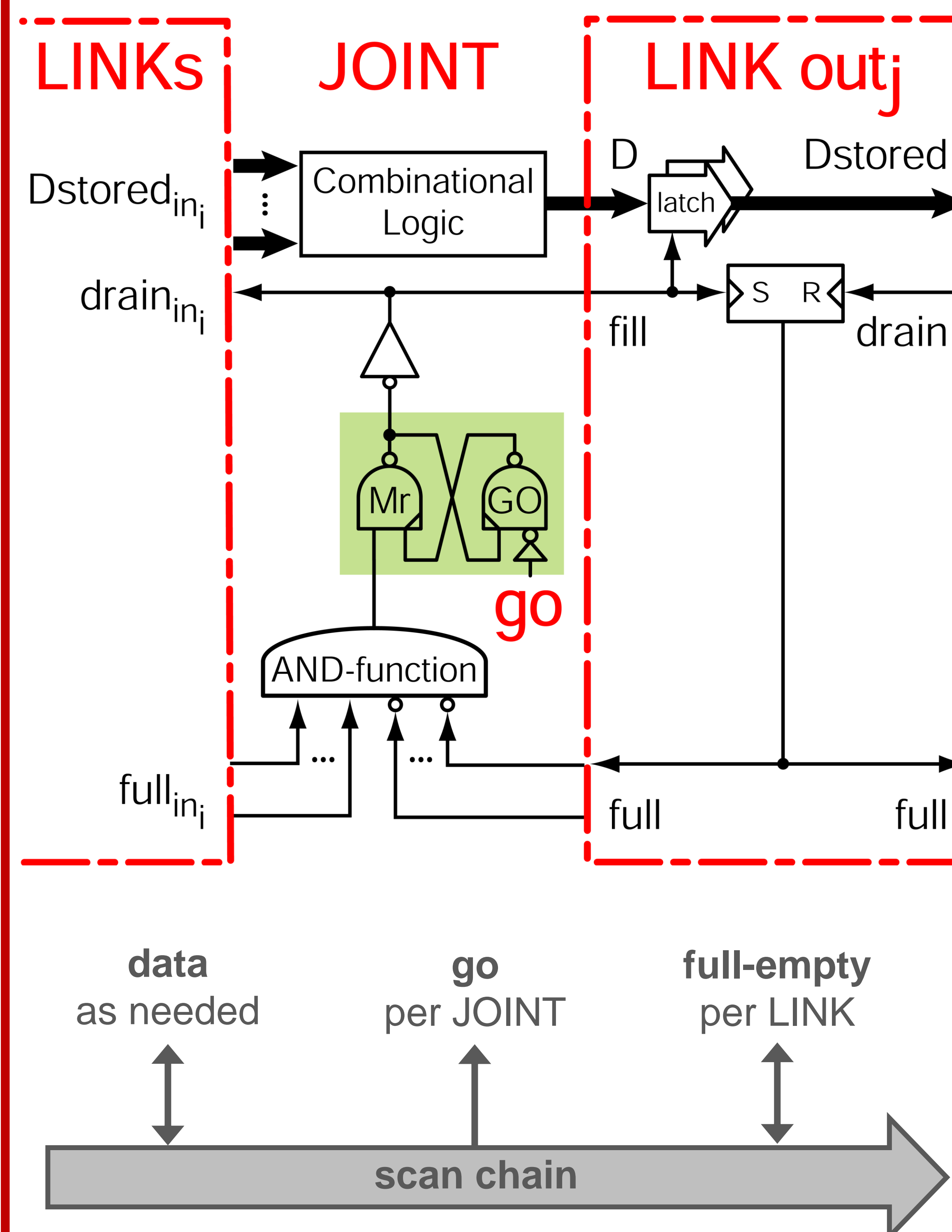 1. structural fault testing,
  2. at-speed testing, and
  3. debug

SOLUTION

- *(Well-known)* **scan chain** to initialize and observe link states

- *(New)* **MrGO** to control individual joint actions
  - go is high (**GO**) – run
  - go is low (🚫) – stop
  - arbiter for safe stop – "proper stopper"
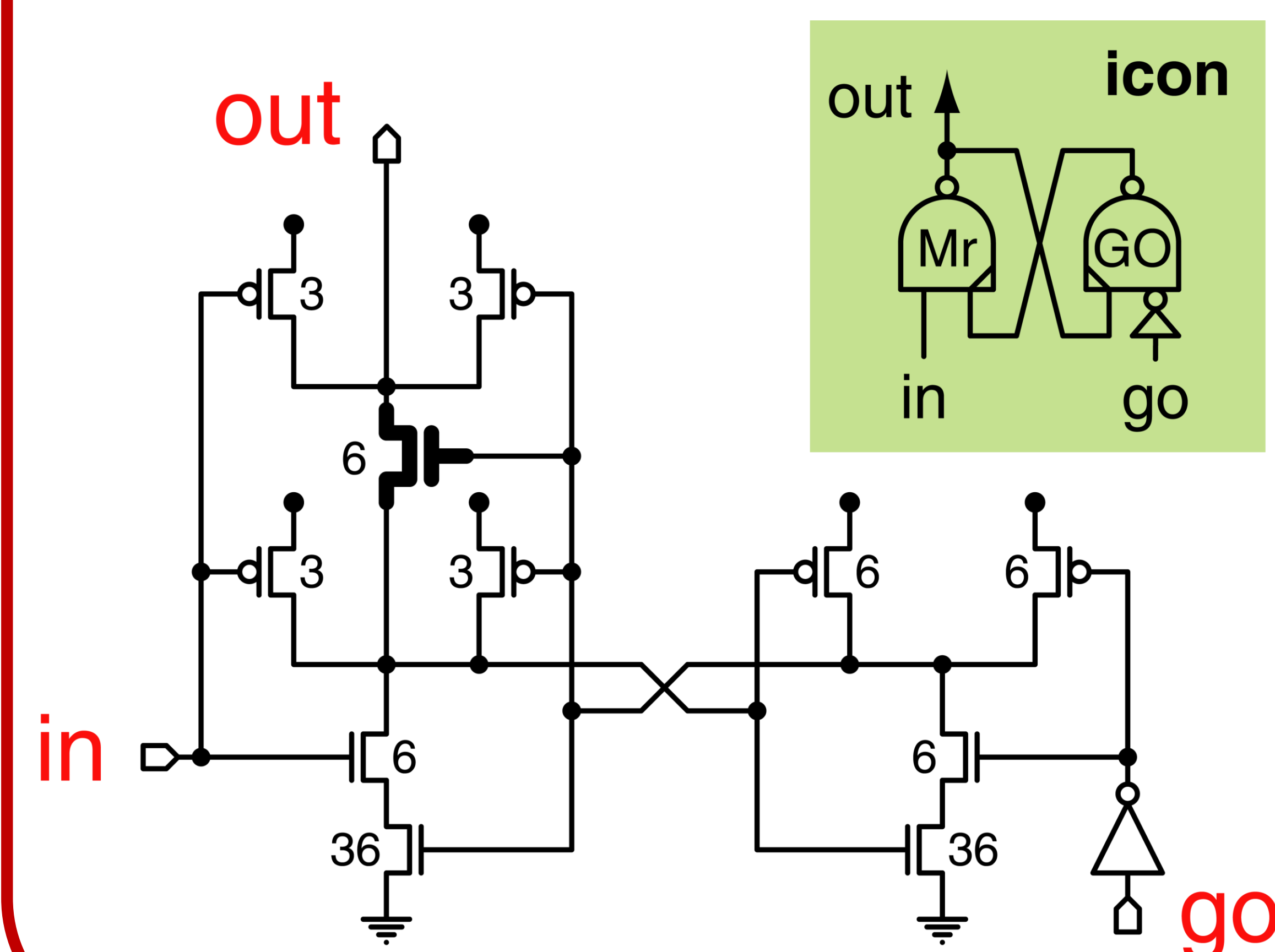 - scan chain delivers go signals

REFERENCES

[1] M. Bushnell and V. Agrawal, *"Essentials of Electronic Testing for Digital, Memory, and Mixed-Signal VLSI Circuits,"* Springer, 2005.
[2] C. Molnar, I. Jones, W. Coates, and J. Lexau, "A FIFO Ring Performance Experiment," *ASYNC*, pp. 279–289, 1997.
[3] M. Roncken, "Defect-Oriented Testability for Asynchronous ICs," *Proceedings of the IEEE*, Vol. 87, No. 2, pp. 363–375, Feb. 1999.
[4] M. Roncken, S. Mettala Gilla, H. Park, N. Jamadagni, C. Cowan, I. Sutherland, "Naturalized Communication and Testing," *ASYNC 2015.*
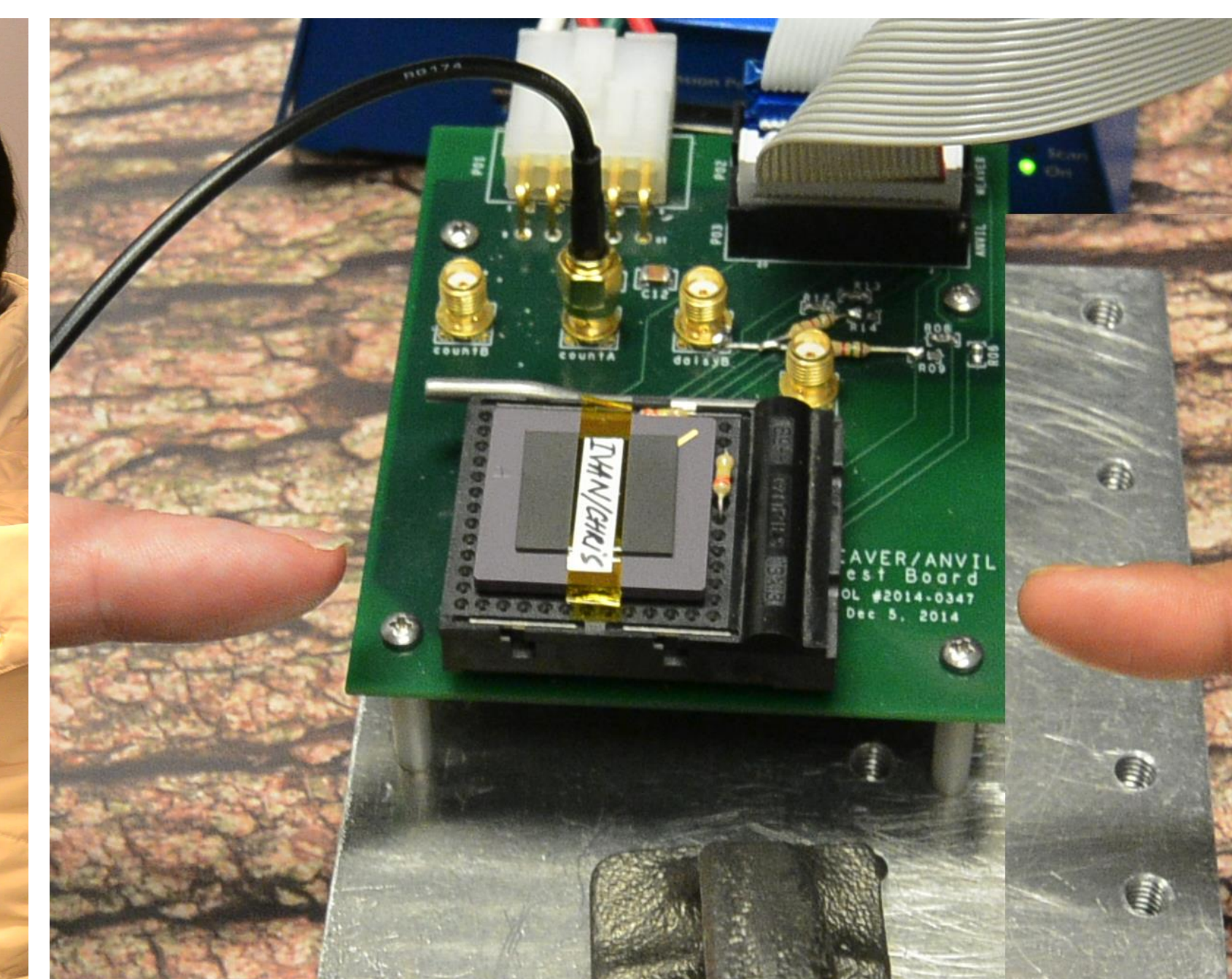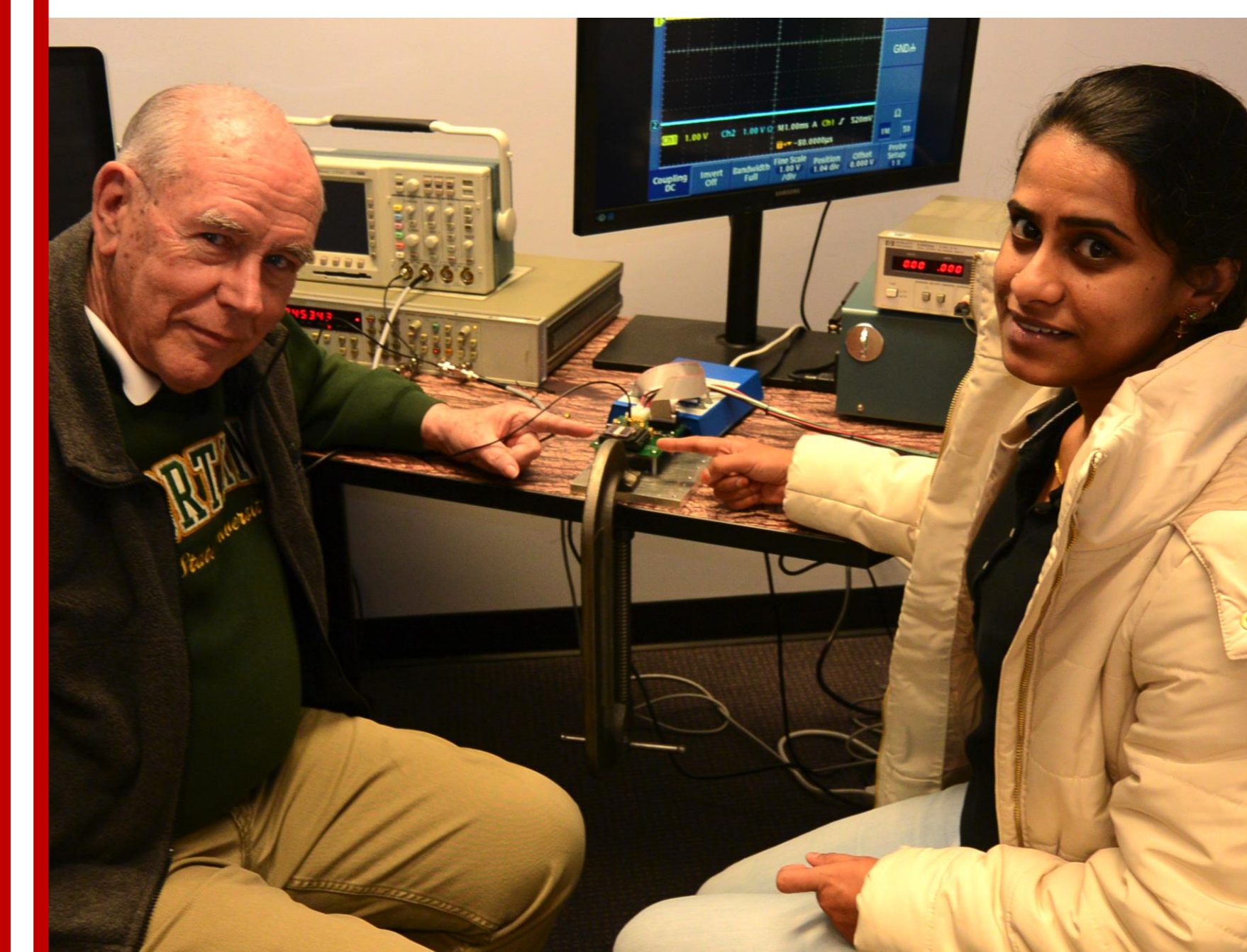
DESIGN FOR TEST

LINKs   JOINT   LINK out$_j$

Dstored$_{in_i}$ Combinational Logic D latch Dstored
drain$_{in_i}$ fill S R drain
Mr GO go
AND-function
full$_{in_i}$ full full

data as needed   go per JOINT   full-empty per LINK

scan chain

MrGO circuit

out   icon
out Mr GO
in go

in   go

TEST EXECUTION

Example: testing a counter at speed

INITIALIZE   joint 1   2   3   4   5
1. freeze all joints
2. set state
   - full-empty links
   - counter data
3. unfreeze "runway" (3, 4)

full empty
GO GO
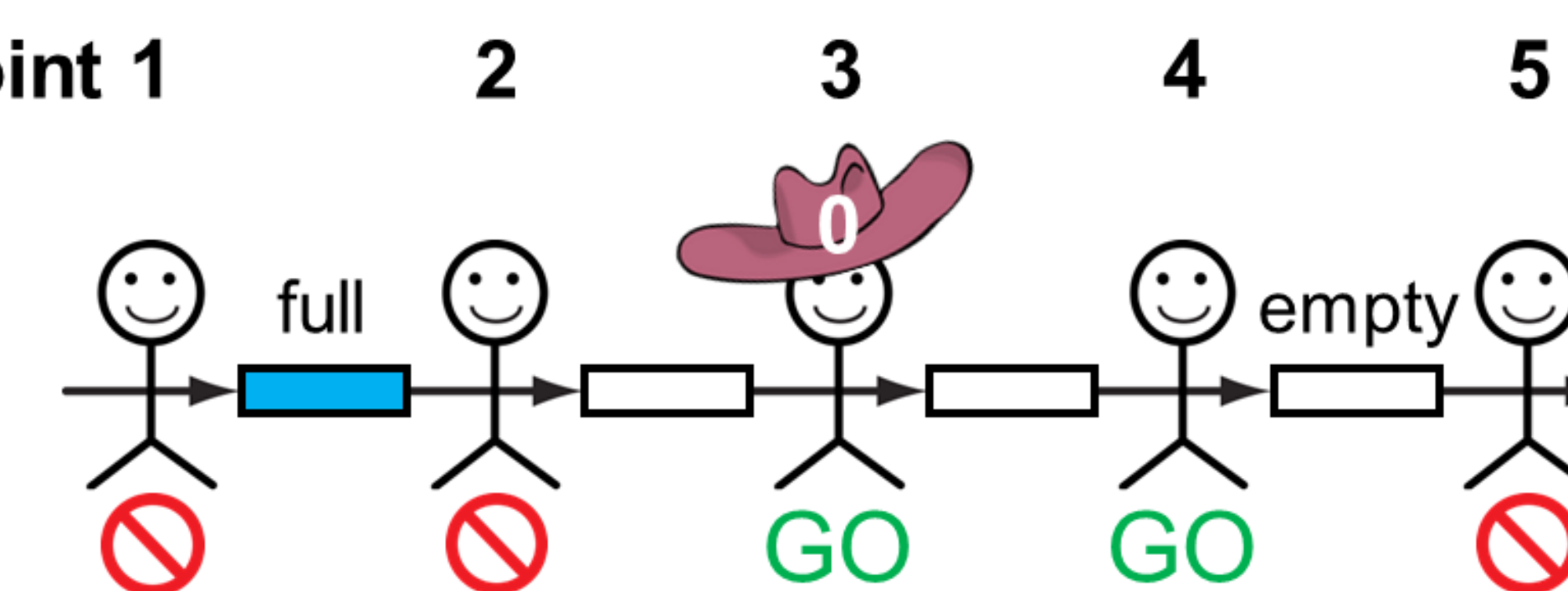
RUN   joint 1   2   3   4   5
1. unfreeze entry (2)
2. wait for action to finish

empty full
GO GO GO
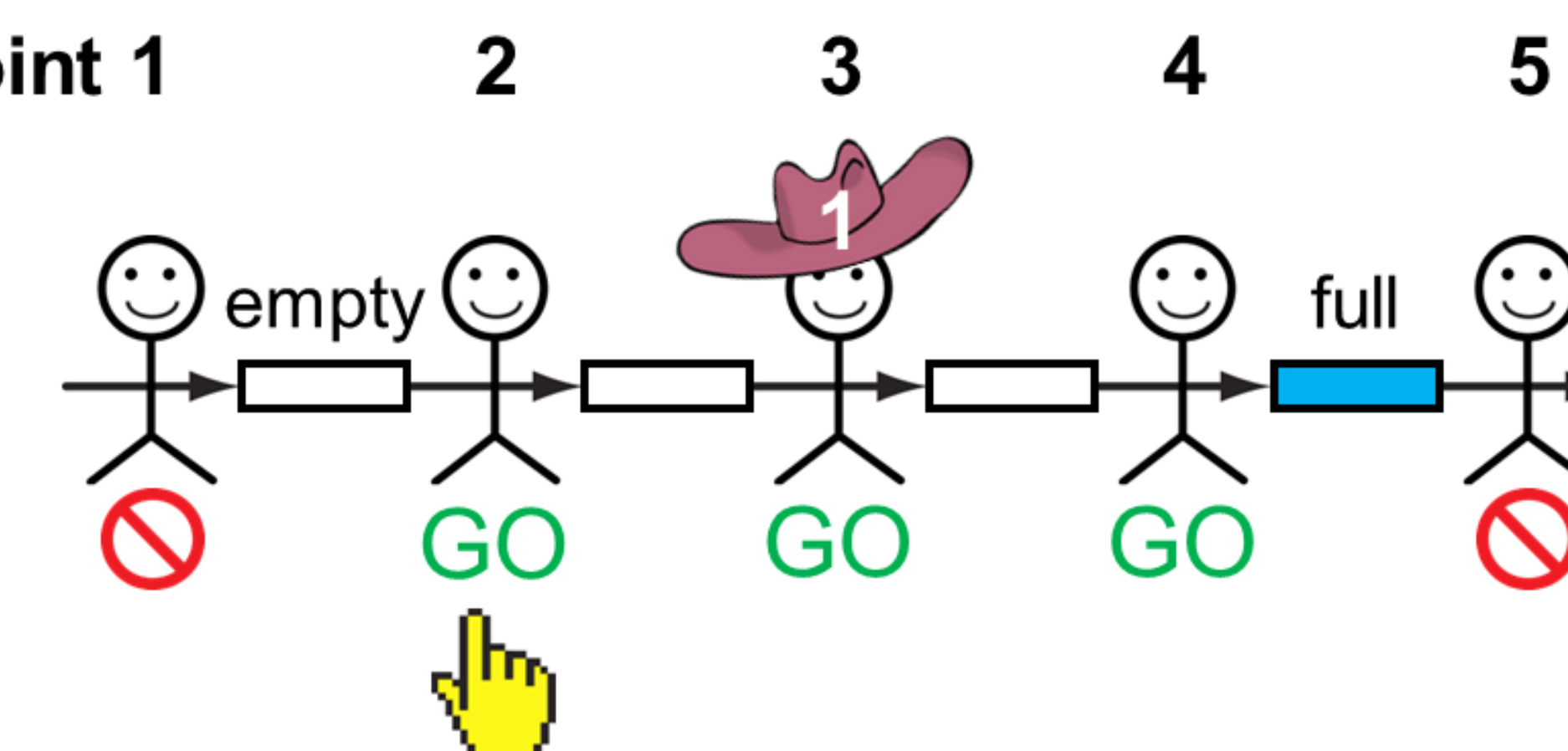
EVALUATE
- read counter data

Supports:
1. Initialization
2. Arbitrated stop from full speed
3. Single- and multi-step operations
4. At-speed testing of sub-systems
5. Canopy graph generation
6. Testing of structural faults like stuck-at

Is built into the latest ARC-Oracle test chip AND IT WORKS !