DFT Reuse Across Silicon Lifetime

From manufacturing test to self-health-aware silicon

Artur Jutman

TALLINN UNIVERSITY OF TECHNOLOGY

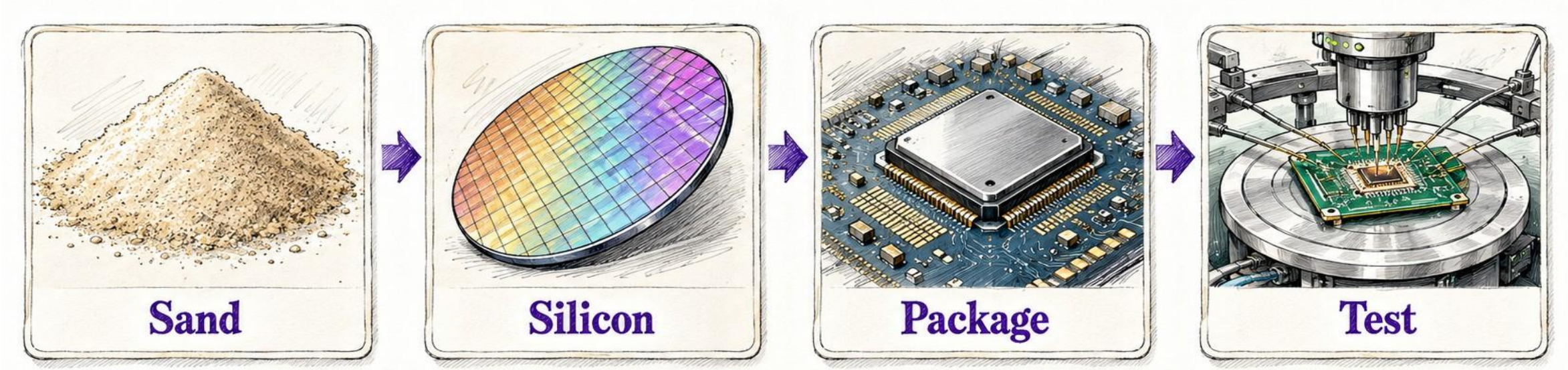
2nd Scientific Workshop on Fault-Tolerant Design
and Real-Time Reliability Monitoring
July 13-15, 2026

Talk Outline

Test → Observe → Diagnose → Recover → Adapt

- Why classical testing is no longer enough
  - HPC/AI complexity, PVT variation, DVFS and lifecycle-dependent failures
- Reusing DFT across the silicon lifecycle
  - Manufacturing test → system-level test → in-field monitoring
- Collecting actionable data on-chip
  - Embedded monitors, sensors and IEEE 1687 IJTAG access
- Turning observations into fault management
  - Filtering, localization, diagnosis, and autonomous recovery
- ESA project case study and lessons learned
  - Architecture, overhead, performance, IC degradation assessment

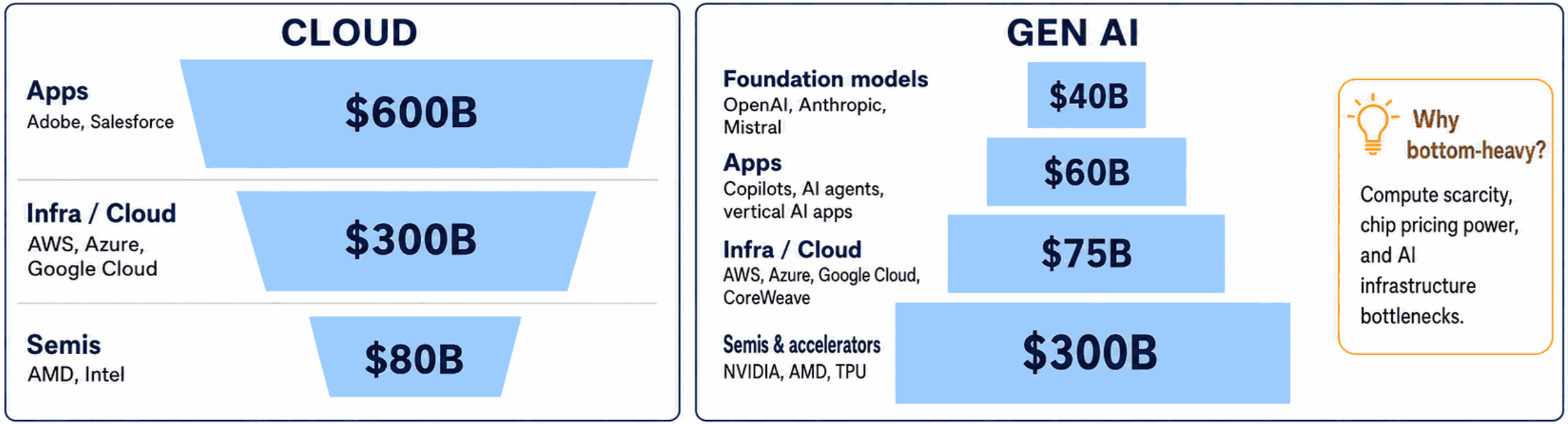
ICs: From Sand to Silicon to Users



Wafer processing: thousands of steps over hundreds of machines

Assembly & Test: hundreds more steps before shipping to users

Global Disruption: Classical Cloud vs. AI Cloud



Source: adapted from Altimeter — Where Does Value Accrue in AI? (estimated annual revenue)

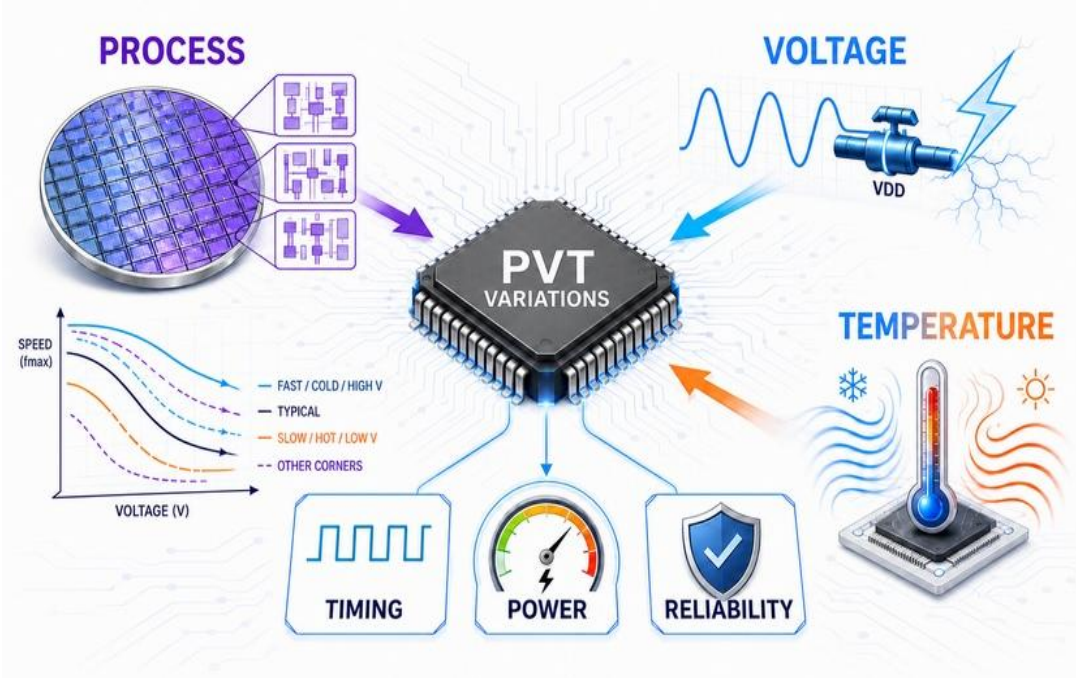
	CLOUD	GEN AI
Fault profile	Memory-dominated SDC	Distributed across HW subsystems
Fault handling	SW-controlled HW redundancy	HW redundancy — too expensive
Test	Classical test methods	Classical + advanced: SLT, SLM, DVFS

Test Methods Become a Critical Asset

- Chiplet / advanced packaging access
  - Known-good-die is not enough after assembly
  - D2D/HBM/interposer faults require package-level diagnosis
- At-speed coverage gaps
  - Many defects appear only under real timing, power, or workload stress
  - Structural tests (even cell-aware) alone miss marginal behavior
- ATE time and cost explosion
  - More cores, memories, SerDes, HBM links, accelerators in one chip
  - Longer test programs collide with high-volume economics

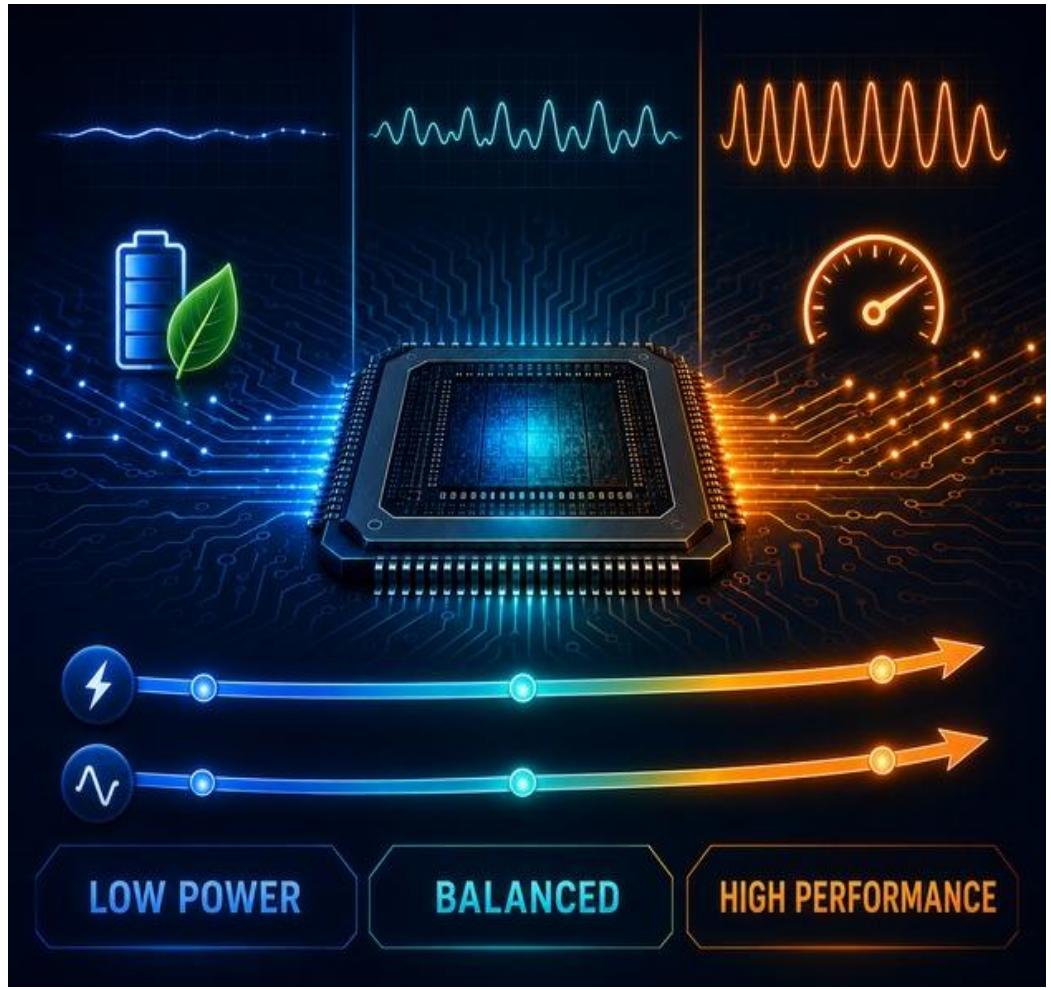
HPC/AI Chips Change Test Assumptions

- Classical DFT = controllability + observability for
  - manufacturing test
  - stuck-at and transition faults
- Critical failures increasingly depend
  - timing, thermal, power
  - workload, aging
- Silicon Lifecycle Management (SLM)
  - controllability + observability + measurability + diagnosability + characterization
  - across manufacturing, System-Level Test (SLT), and field lifetime



Dynamic Voltage and Frequency Scaling

- Tuning of voltage and clock speed to maximize performance while staying within power, thermal, and reliability limits
  - stable specific V/F operating points
  - minimum safe voltage (Vmin)
  - turbo frequencies
- DVFS is often used during System Level Test (SLT) to
  - Check workload-dependent instability, thermal throttling behaviour
  - Create worst-case behaviour to catch faults that classical ATE patterns miss



Change in Test Paradigm

- Test must characterize operational behaviour - not only detect faults

- Observe: PVT, path margin, voltage droop, temperature and aging
  - PVT monitors, ring oscillators, path-margin monitors, embedded sensors
  - functional interface-based high-speed test access
- Exercise: real workloads, interfaces
  - Dynamic Voltage and Frequency Scaling (DVFS) operating points
- Analyse: device-specific margins and degradation
  - adaptive voltage/frequency binning
  - ML or analytics-assisted screening

- A “known-good die” must become a “known-health system”

HPC CHIP LIFECYCLE: FROM MANUFACTURING TEST TO IN-FIELD HEALTH

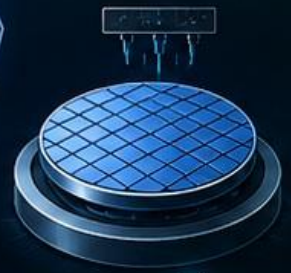
Continuous Silicon Assurance Across the Lifetime

1 DESIGN & DFT

Testability & Observability by Design

2 WAFER TEST (ATE)

Structural & Parametric Test



3 PACKAGE / ASSEMBLY TEST

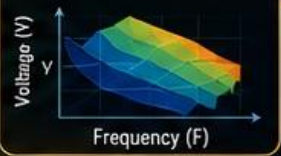
Interconnect & Package Integrity

4 SYSTEM-LEVEL TEST (SLT)

Real Workloads in a Near-Real Environment

V/F PROFILING (DVFS)

Voltage–Frequency (V/F) Profiling under Workloads

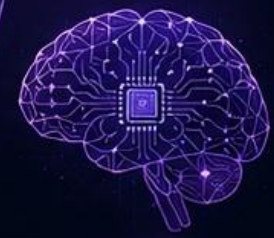


5 DEPLOYMENT & OPERATION

Field Operation (SLM in Action)

6 END-OF-LIFE / NEXT DESIGN

Learning Loop for Future Silicon



PRIMARY OBJECTIVES

- Design for test, debug and lifetime observability
- Defect screening, parametric margins, early binning
- Detect assembly & interconnect defects
- Validate real-world behavior, characterize margins, P/F & V/F profiling
- Ensure reliability, detect degradation, optimize in operation
- Feed insights back to improve next generation

KEY TECHNIQUES & APPROACHES

- DFT / Scan / MBIST
- IJTAG / cJTAG
- Embedded Sensors
- Design-for-Debug
- Instrumentation Planning
- Structural Test (DFT)
- Parametric Test
- PVT Monitors
- IDDQ / Leakage
- Path Margin / At-Speed
- On-Chip Variation Measurement
- Known-Good-Die Test
- D2D Interconnect Test
- HBM / SerDes Test
- Probeless Fault Isolation
- 3D / 2.5D Package Test
- Boundary Scan / IJTAG
- Real Workload Execution
- OS / Firmware / Drivers
- Stress: Power, Thermal, Frequency, I/O
- Functional / Performance Montiors
- V/F Profiling (DVFS)
- Runtime Telemetry (Power, Temp, Errors)
- Health Monitors & Agents
- Degradation Tracking
- Predictive Analytics / ML
- RMA Prevention & Fleet Management
- Failure Analytics
- Root Cause Mining
- Design Feedback
- Test Content Optimization
- Process / Package Improvement

DATA & INSIGHTS FLOW

Test Data & Measurements → Die / Package Health Signature → System-Level Behavior & Margins (SLT) → Field Telemetry & Health Trends → Insights & Learning Loop

CROSS-CUTTING ENABLERS

- Embedded Instrumentation & Sensors
- DFT / IJTAG Infrastructure
- Data Transport (On-Chip / Off-Chip)
- Analytics & Machine Learning
- Security & Root of Trust

OUTCOMES & VALUE

- Higher Yield Lower Cost of Test
- Better Performance Binning
- Lower Field Failure & RMA
- Longer Lifetime & Reliability
- Optimized Power & Thermal Budget
- Continuous Improvement Across Generations

Design & Test Foundation | Manufacturing Test | Packaging & Assembly | System-Level Test (SLT) | In-Field / SLM | Feedback & Learning

DFT – Design for Test | SLT – System-Level Test | V/F – Voltage–Frequency
DVFS – Dynamic Voltage & Frequency Scaling | SLM – Silicon Lifecycle Management
PVT – Process, Voltage, Temperature

LEVERAGING ON-CHIP DFT RESOURCES ACROSS SILICON LIFETIME

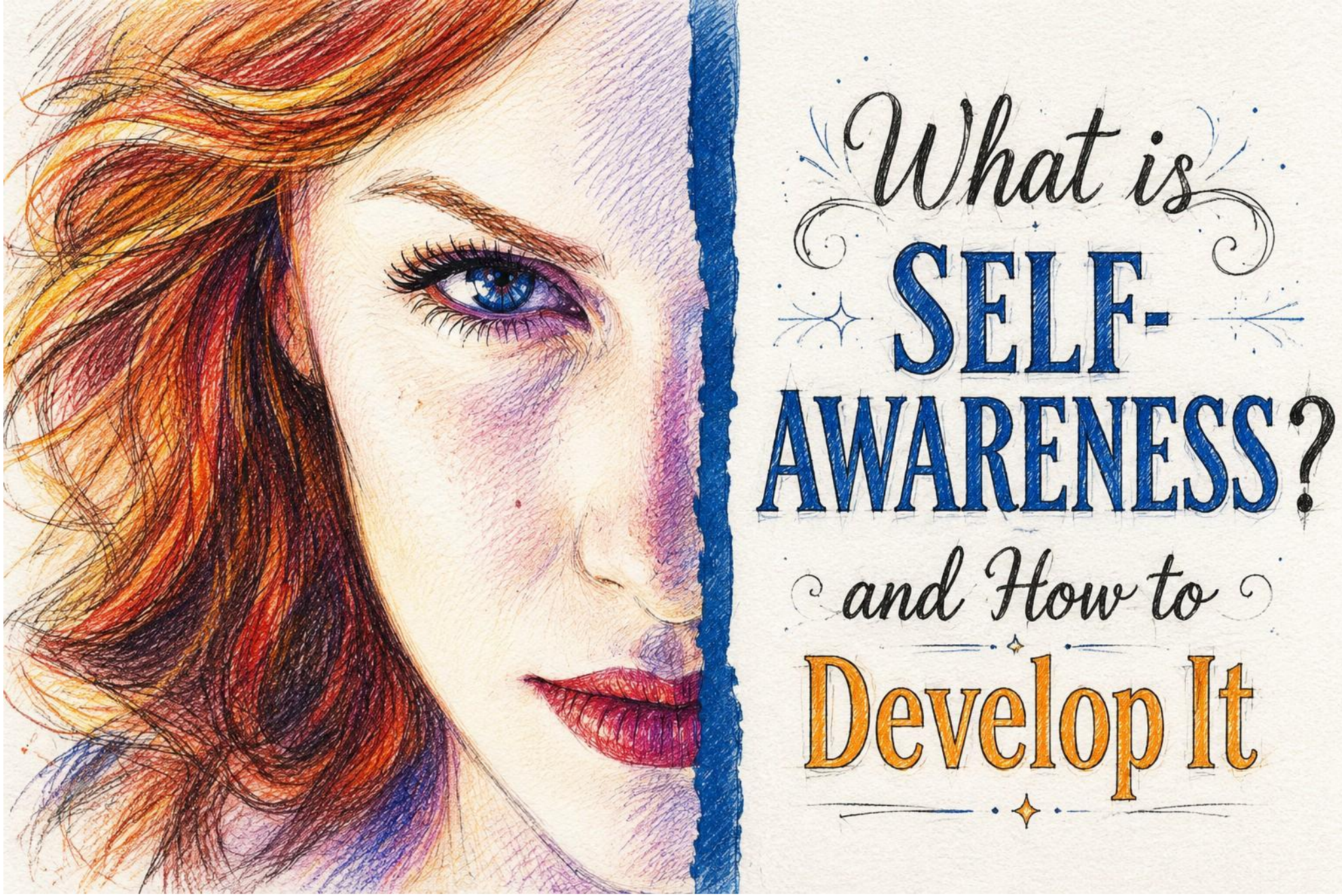
Typical On-Chip DFT Commodity

- Access: JTAG, IJTAG, RSN, TAPs, wrappers, test buses
- Structural Test: Scan, compression, LBIST, ATPG support
- Memory Test: MBIST, BIRA/BISR, ECC, memory diagnosis
- I/O & Package Test: SerDes BIST, loopback, PRBS, D2D/HBM/chiplet test
- Sensors & Parametric Monitors: PVT, droop, thermal, timing margin, aging
- Checkers & Error Detection: ECC, parity, CRC, protocol checkers, watchdogs
- Debug & Trace: trace, counters, embedded logic analyzers, event capture
- SLT / SLM Telemetry: workload monitors, V/F profiling, degradation tracking
- Security: secure debug, scan locking, lifecycle-state access control

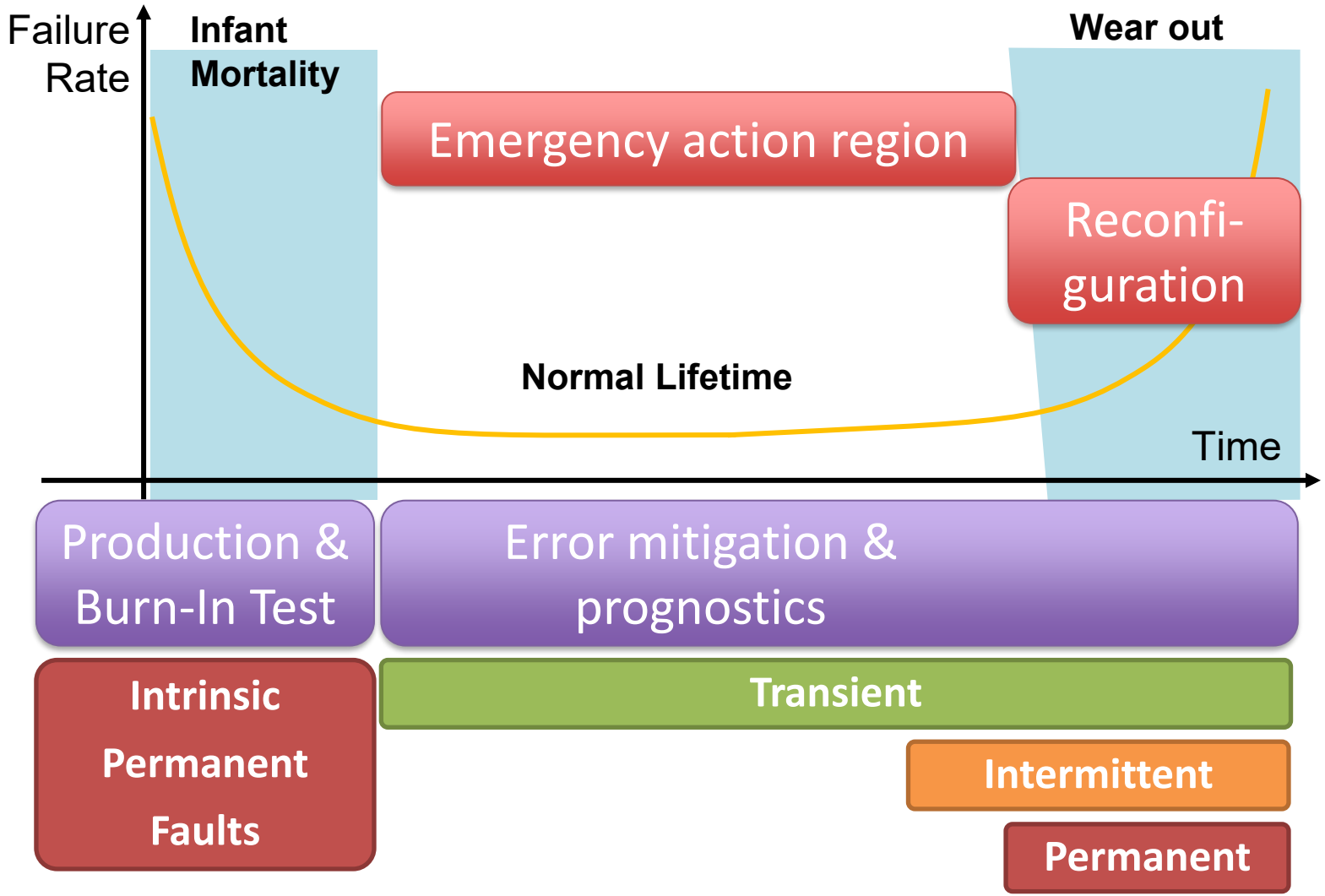
Possible DFT Reuse Across Test Stages

DFT / Embedded Test Infrastructure	Wafer test	Assembly / package test	System-level test	Power-on self-test	Maintenance / repair test	Online monitoring
JTAG/IJTAG	Yes	Yes	Partial	Partial	Yes	Partial
Scan	Yes	Yes	Partial	Partial	Yes	No
LBIST	Yes	Yes	Partial	Yes	Yes	No
MBIST	Yes	Yes	Yes	Yes	Yes	No
SerDes BIST	Partial	Yes	Yes	Yes	Yes	Partial
Loopback	Partial	Yes	Yes	Yes	Yes	No
Parametric monitors	Yes	Yes	Yes	Yes	Yes	Yes
Protocol checkers	No	Partial	Yes	Yes	Yes	Yes
Watchdogs	No	Partial	Yes	Yes	Yes	Yes
ECC	Partial	Partial	Yes	Yes	Yes	Yes
Debug & Trace infra	No	Partial	Yes	Partial	Yes	Partial
Workload monitors	No	No	Yes	Partial	Yes	Yes
V/F profiling	Partial	Partial	Yes	Partial	Yes	Partial
Aging monitors	Partial	No	Partial	Partial	Yes	Yes

TOWARDS IN-SILICON SELF-HEALTH AWARENESS

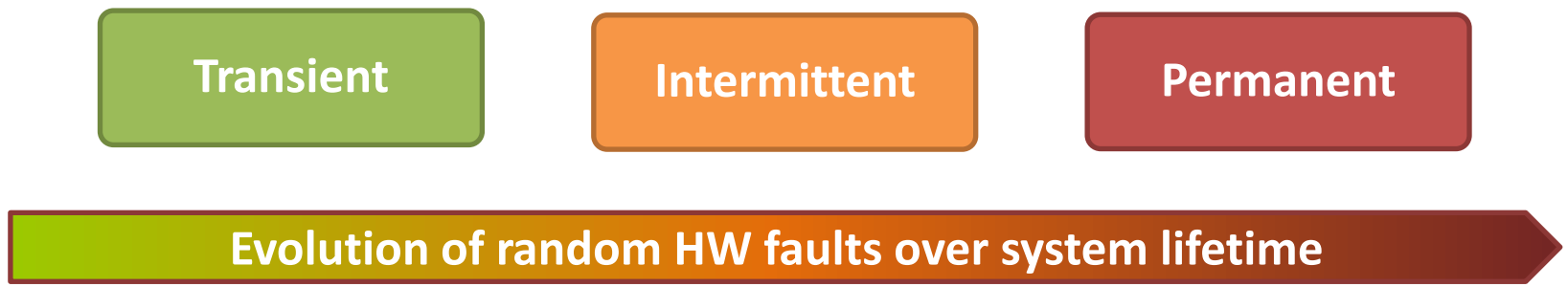

What is
SELF-
AWARENESS?
and How to
Develop It

Silicon Lifecycle and Fault Mitigation



Lifecycle from Functional Safety Perspective

- A system has to
  - remain dependable during in-field operation
  - function as intended even in case of a failure
- In-field failures can be due to
  - systematic design errors both in HW and SW
  - random hardware faults (due to test escapes, radiation or ageing)



ENABLING ON-CHIP DATA COLLECTION

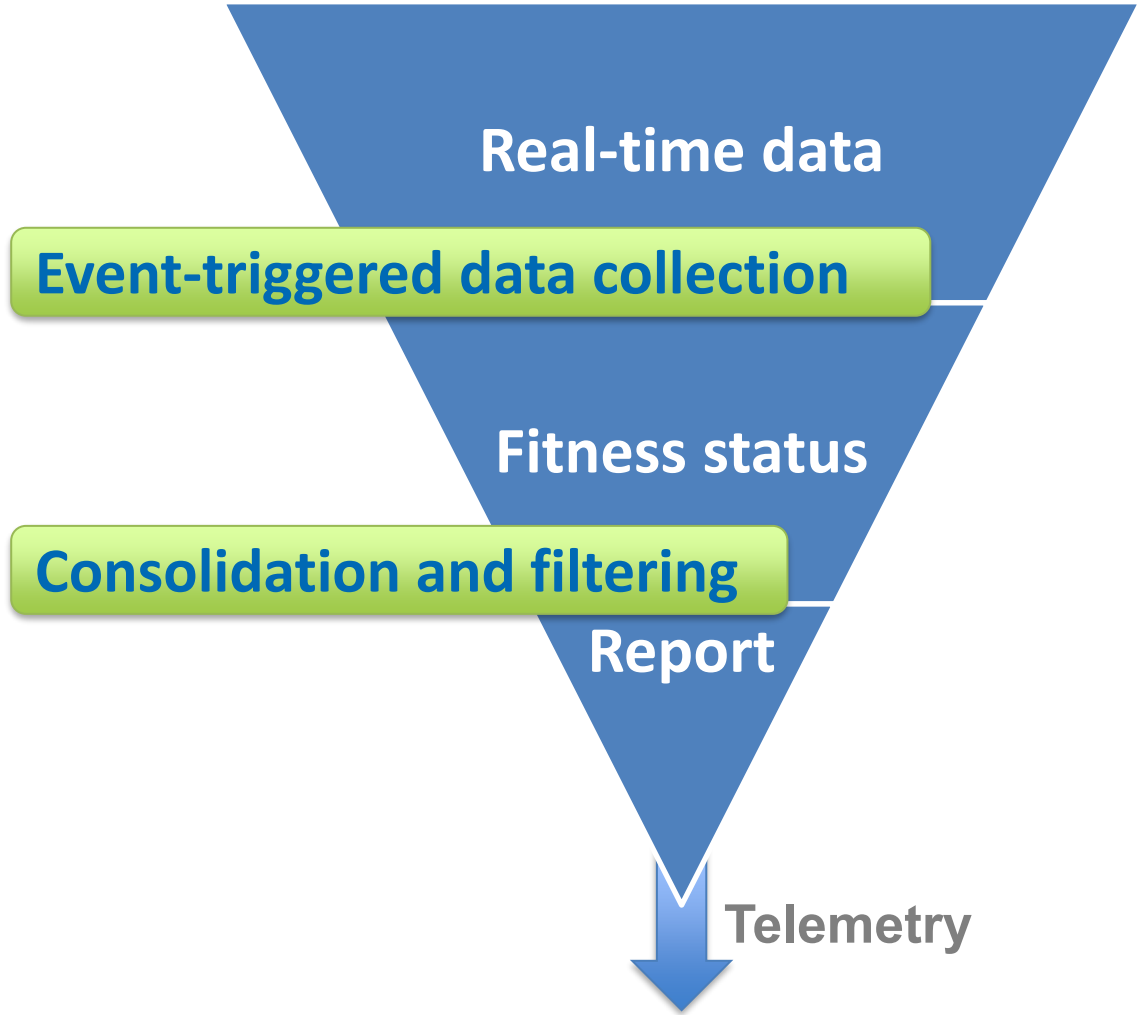
Often Done: Periodic Data Collection

- Power on self-tests
- Key on / key off tests
- Reading system-level sensors
- Periodic tests by execution of Logic BIST / Memory BIST
- Monitoring system parameters during periodic SBST and SLT
- Data often transferred during maintenance sessions *(periodic)*
- Limited real time monitoring and handling

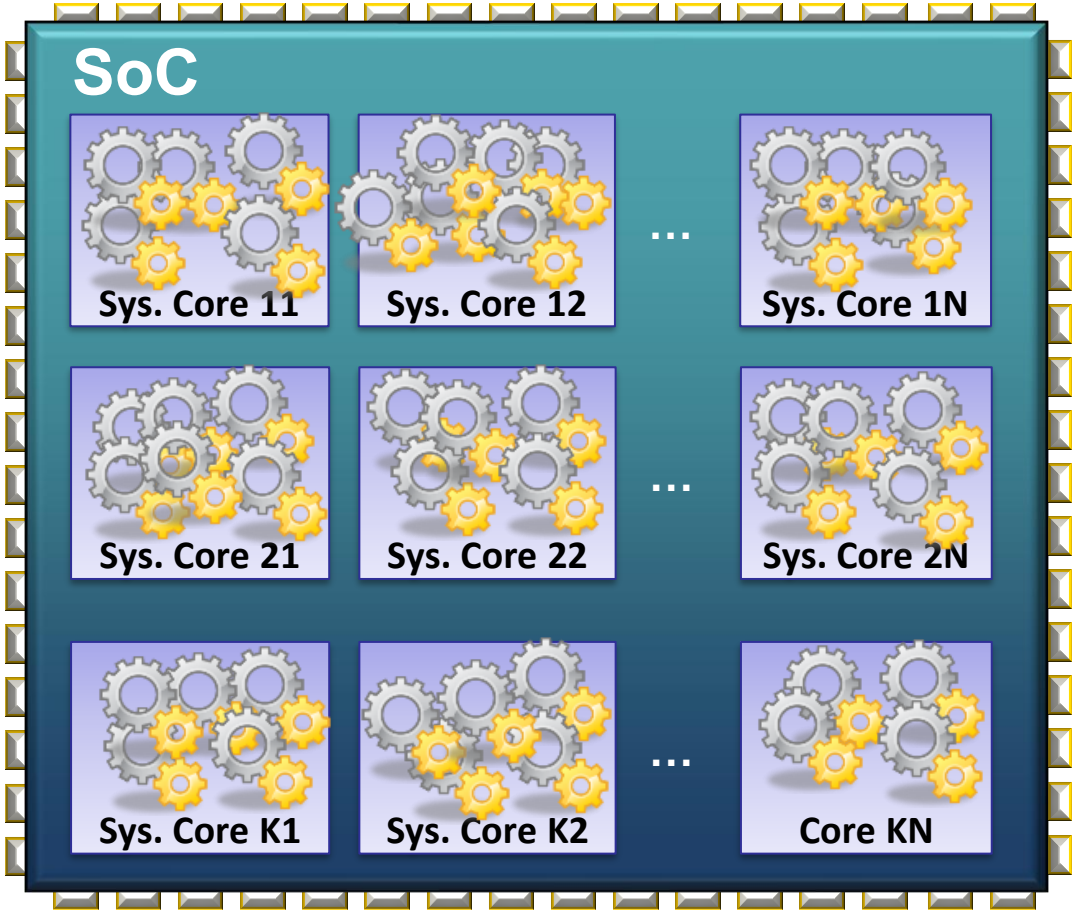
LIMITED value for intelligent ML-based analytics

Current Needs: Non-Stop Data Collection

- Real time monitoring
- Failure detection and recovery
  - Emergency information
  - On-board recovery actions
- Processing and filtering on-board
  - System fitness assessment
  - Selectable filtering
- Reporting via telemetry
  - Current status
  - Detailed fitness information
  - Fitting into limited bandwidth



Collecting Big Data on a SoC



Possible Data Sources

- Voltage and thermal sensors
- IC integrity monitors, aging sensors
- ECC, CRC, arithmetic codes
- TMR voting circuit, SEU guarding
- POST, BIST, BISD, BISR
- Property and assertion checkers
- Debug and diagnosis
- Power supply noise detectors
- Process variation sensors
- Configuration registers
- Status/Signature registers

How to handle thousands of instruments?

IEEE 1687: One Standard to Rule Them All

2004 IJTAG Initiative Group formed

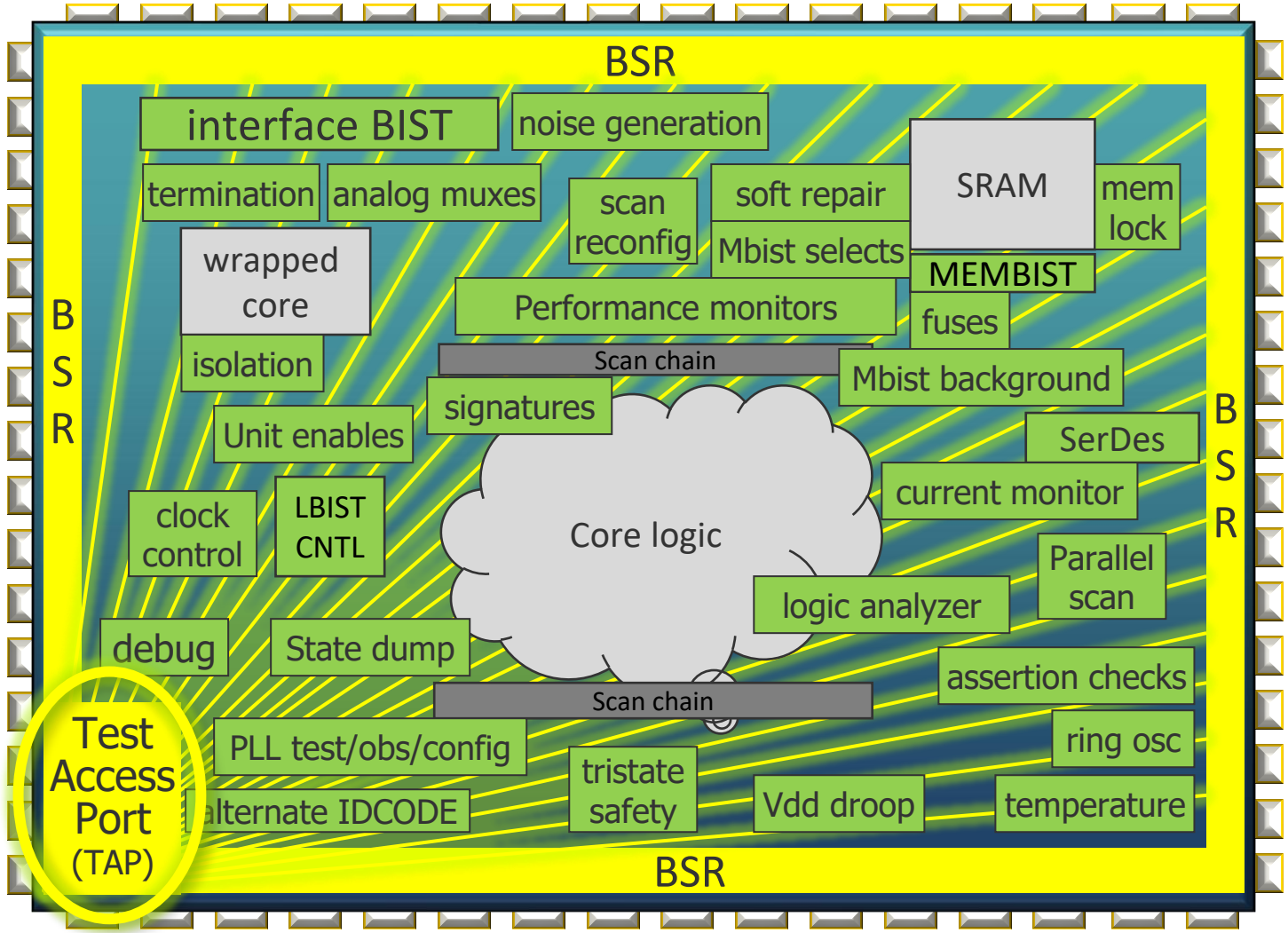
2006 IEEE accepted as a draft project

……. Hard work goes on non-stop …….

2014 IEEE Standard for Access and Control of Instrumentation Embedded within a Semiconductor Device

Applications

- Scan Test incl. BIST
- Configuration & calibration
- Monitoring (sensors)
- IC debug



Adapted from: Jeff Rearick, AMD, ETS 2012 presentation

IEEE Std. 1687 – IJTAG Basics

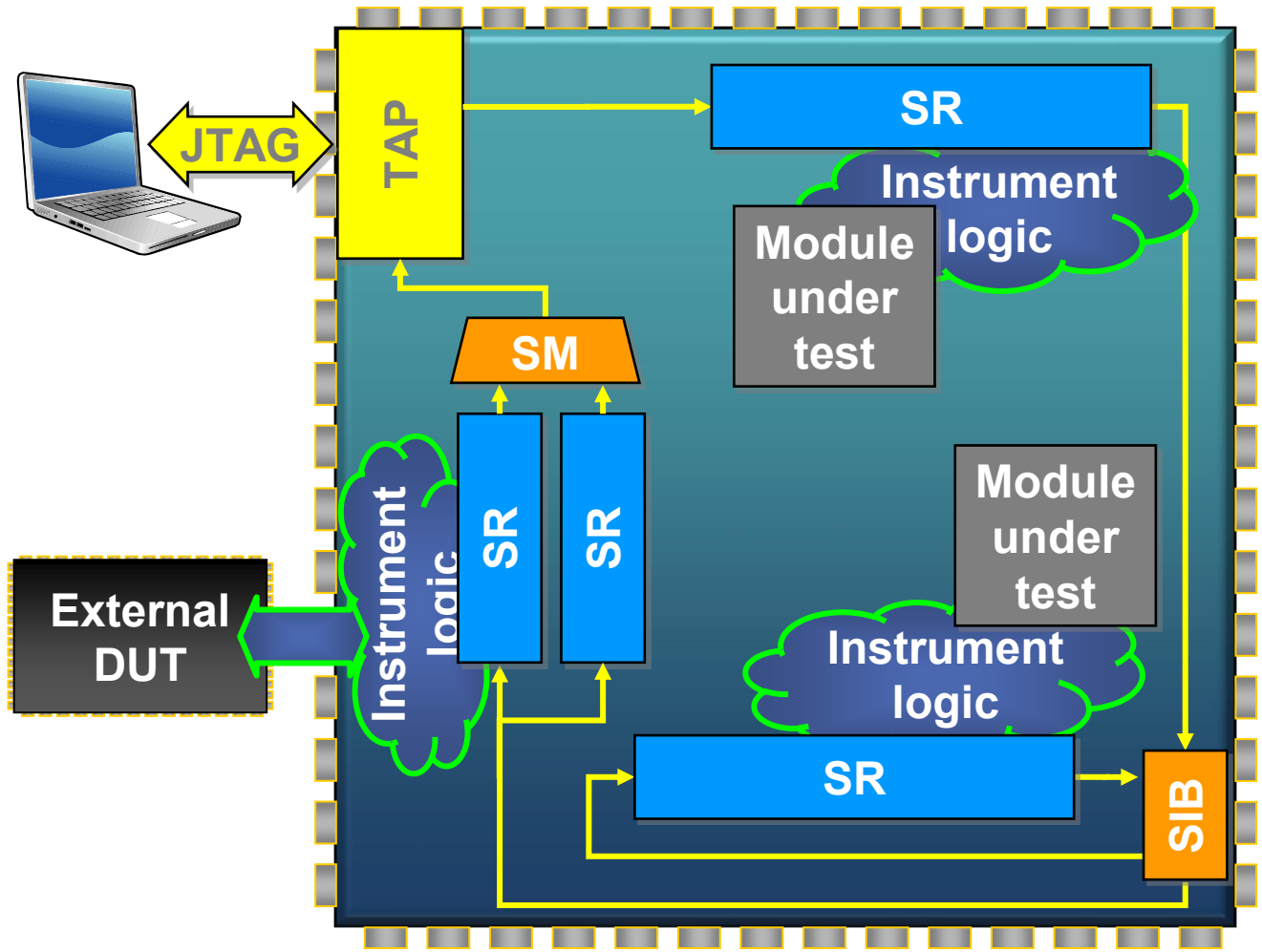
SR – Scan Register
SR – Scan Multiplexer
SIB – Segment Insertion Bit

IEEE 1687 employs **Reconfigurable Scan Networks** to communicate with resources

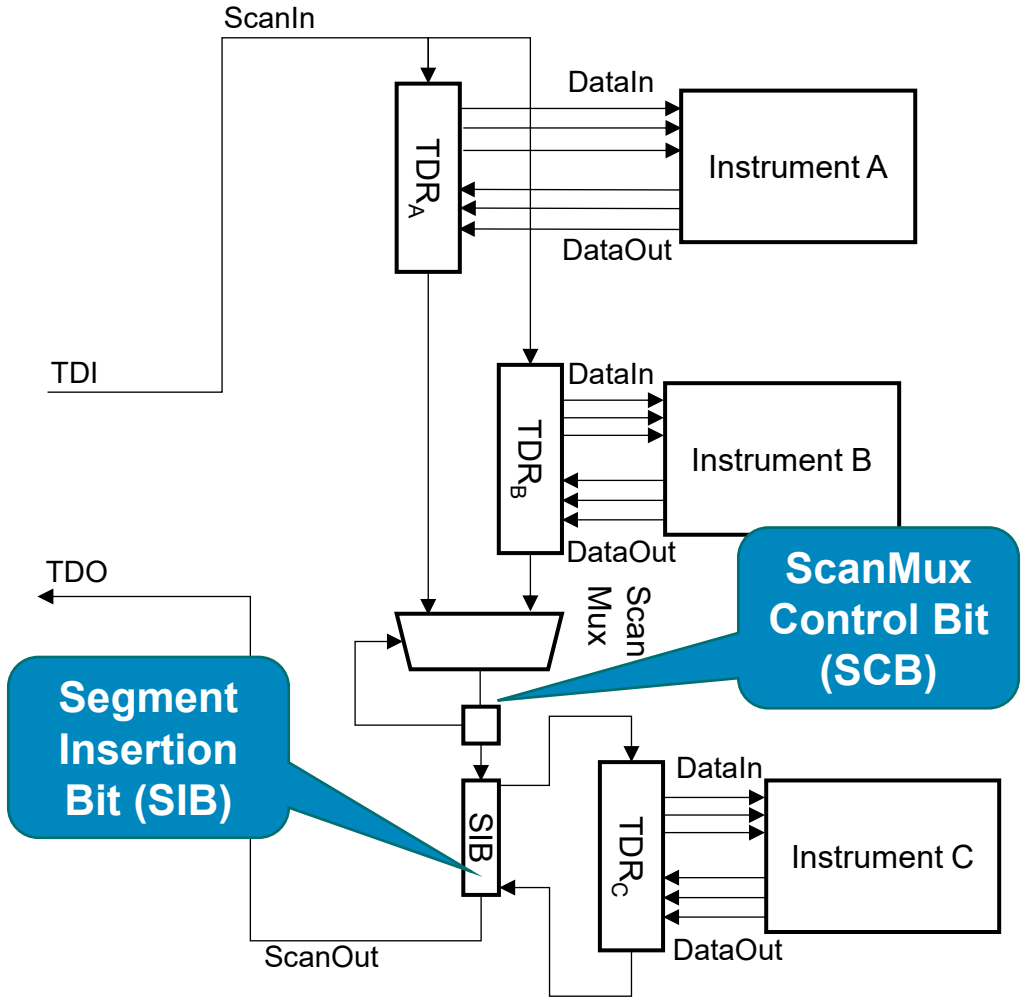
RSN consists of registers (SR, TDR) and multiplexers (SIB, SM, SCB) allowing to exchange data and reconfigure the scan network topology

ScanMuxes support mutually excludable segments by controlling the ScanMux Control Bit (SCB)

SIBs support bypassing segments by toggling the S bit inside the SIB

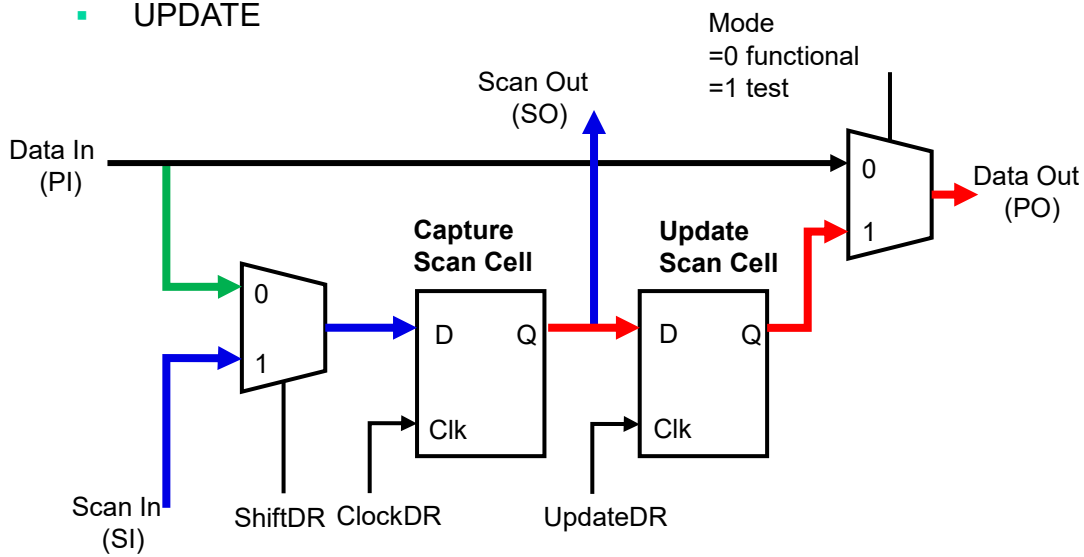


Under The Hood

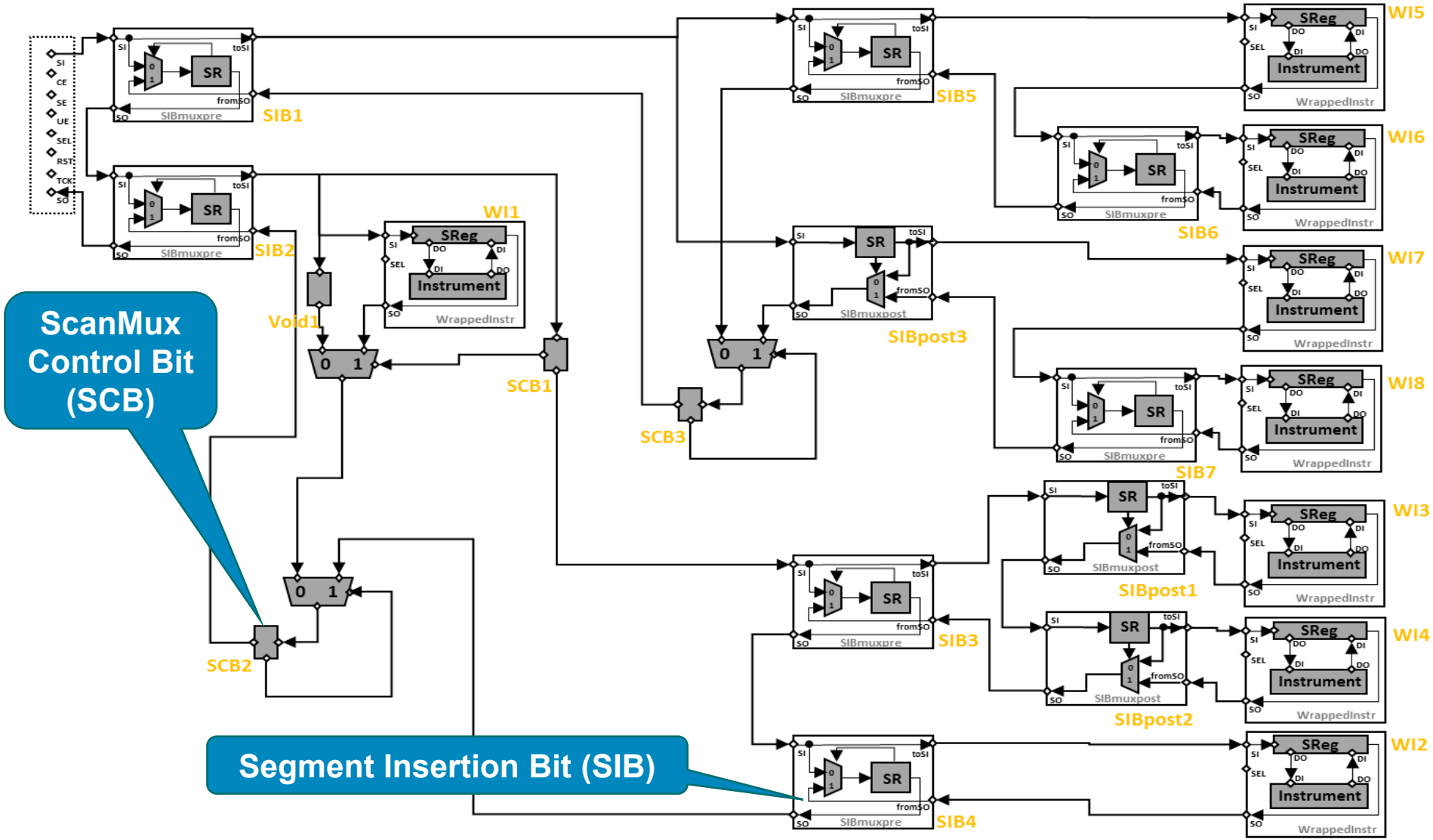


TDRs composed of CSU cells:

- Reading data from instruments
  - CAPTURE
  - SHIFT
- Writing data to instruments
  - SHIFT
  - UPDATE
- Reconfiguring the network
  - SHIFT
  - UPDATE

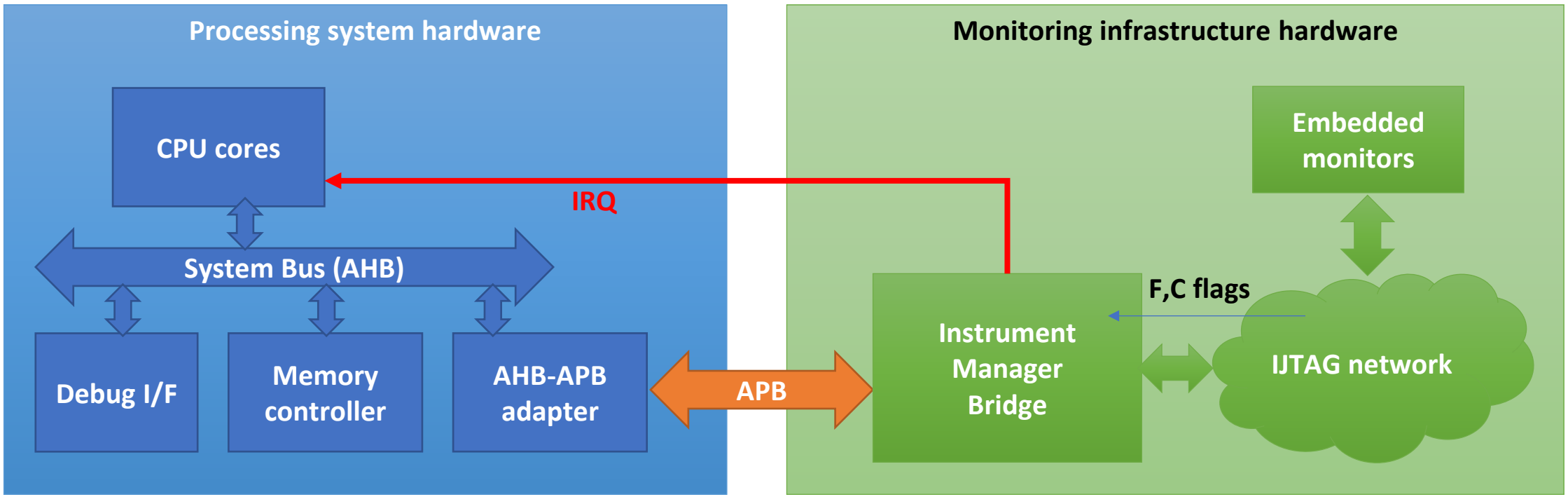


A Typical IJTAG Network Example



Can IEEE 1687 IJTAG be used In-Field?

- Yes, if IJTAG access is NOT disabled after production
- There is a bridge between the IJTAG entry point and the OS



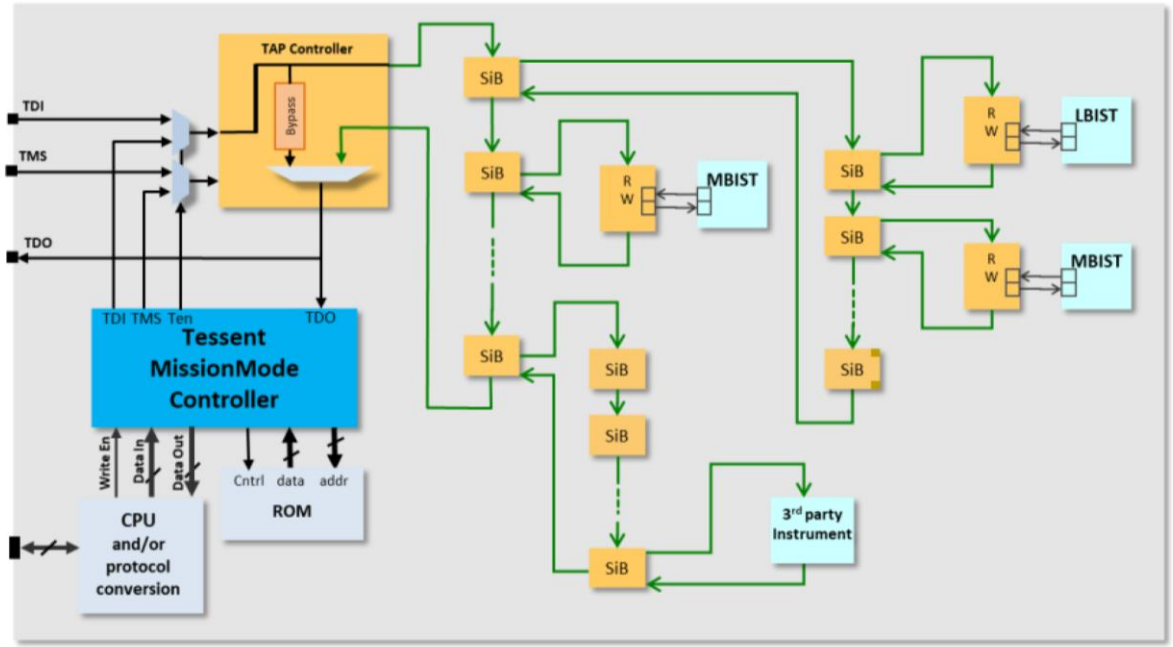
IEEE1687 based On-Chip Monitoring Approaches

- IJTAG extension for fault management has been proposed in
  - A. Jutman, S. Devadze and J. Aleksejev, “Invited paper: System-wide fault management based on IEEE P1687 IJTAG,” in International Workshop on Reconfigurable Communication-centric Systems-on-Chip (ReCoSoC), 2011.
  - E. Larsson and K. Shibin, “Fault management in an IEEE P1687 (IJTAG) environment,” in 15th International Symposium on Design and Diagnostics of Electronic Circuits & Systems (DDECS), 2012.

- Further elaborated by groups from TalTech, Lund Univ., Univ. of Twente, Testonica Lab
  - A. Jutman, S. Devadze and K. Shibin, “Effective Scalable IEEE 1687 Instrumentation Network for Fault Management,” IEEE Design & Test, vol. 30, no. 5, pp. 26-35, 2013.
  - K. Shibin, S. Devadze and A. Jutman, “Asynchronous Fault Detection in IEEE P1687 Instrument Network,” in IEEE North-Atlantic Test Workshop (NATW), 2014.
  - K. Shibin, S. Devadze and A. Jutman, “On-line fault classification and handling in IEEE1687 based fault management system for complex SoCs,” in 17th Latin-American Test Symposium (LATS’2016), 2016.
  - Farrokh Ghani Zadegan, Dimitar Nikolov, Erik Larsson, “A Self-Reconfiguring IEEE 1687 Network for Fault Monitoring,” European Test Symposium, 2016
  - K. Shibin, S. Devadze, A. Jutman, M. Grabmann, R. Pricken, “Health Management for Self-Aware SoCs based on IEEE 1687 Infrastructure”, in IEEE Design & Test: Special Issue on Self-Awareness in SoCs, Vol 34, No. 6, 2017, pp. 27-35.

Industrial Example: Tessent MissionMode

Source: Siemens EDA



Enables system-controlled field test of test instruments

Automates communication between vehicle system and on-chip test resources

- In-system control of on-chip test resources

Complete flow

- Insertion of MissionMode controller IP into the IJTAG infrastructure
- Automated translation between IJTAG serial data and parallel CPU read/write commands
- Verification testbenches

Two modes of operation

- CPU access mode supports parallel read and write to/from CPU bus
- Direct Memory Access (DMA) mode with command data in non-volatile memory

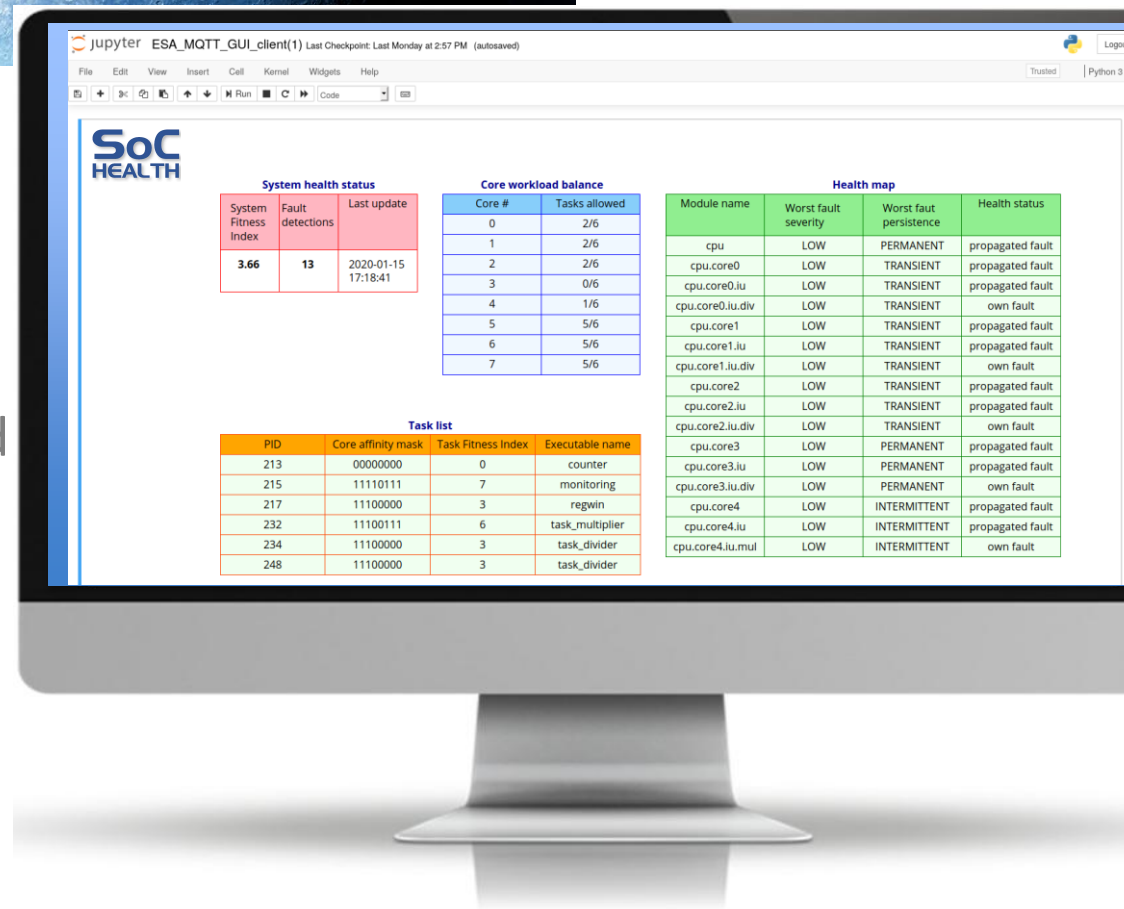
CASE STUDY FOR ESA

SoC-HEALTH: On-Chip Health Monitoring and Fault Management for SoC Health-Awareness in Space Missions

SoC HEALTH

Case study project

- Linux OS running on 8-core CPU
- Health Monitoring hardware
  - 1262 checkers in ALU, registers, pipeline, etc.
- Health Monitoring Software
  - Fault-managed tasks are running on health-managed resources
  - Task scheduling w.r.t. system health and task requirements
- Demonstrator
  - Combination of both HW and SW
  - Human operator readable telemetry

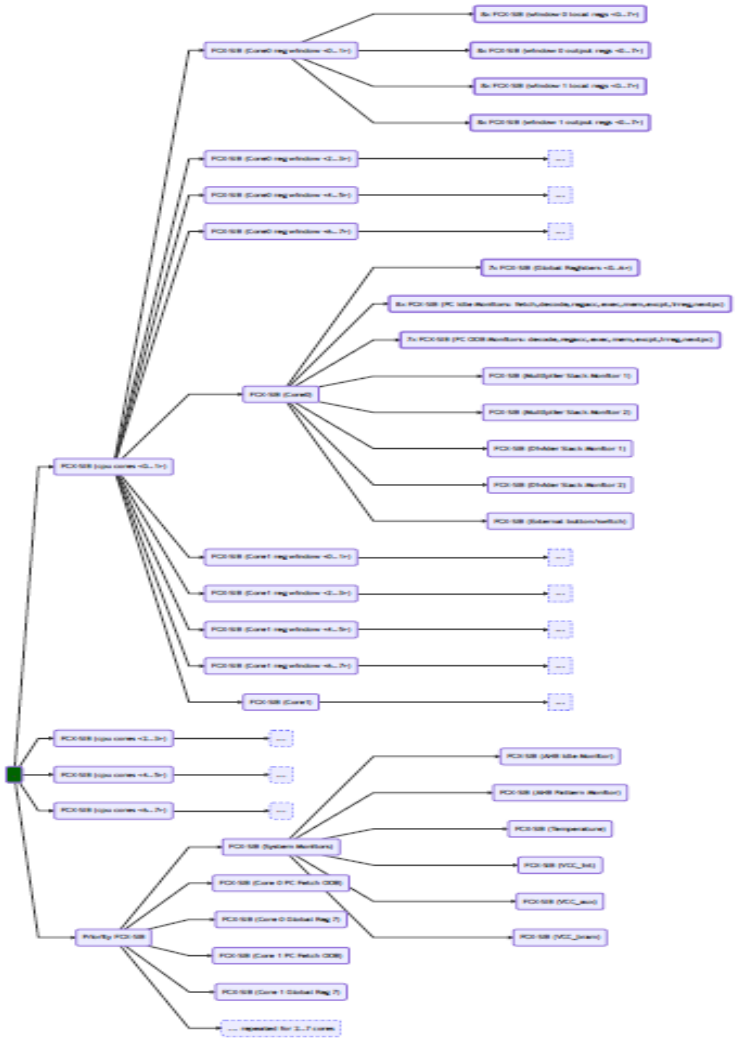


Fleet of Checkers

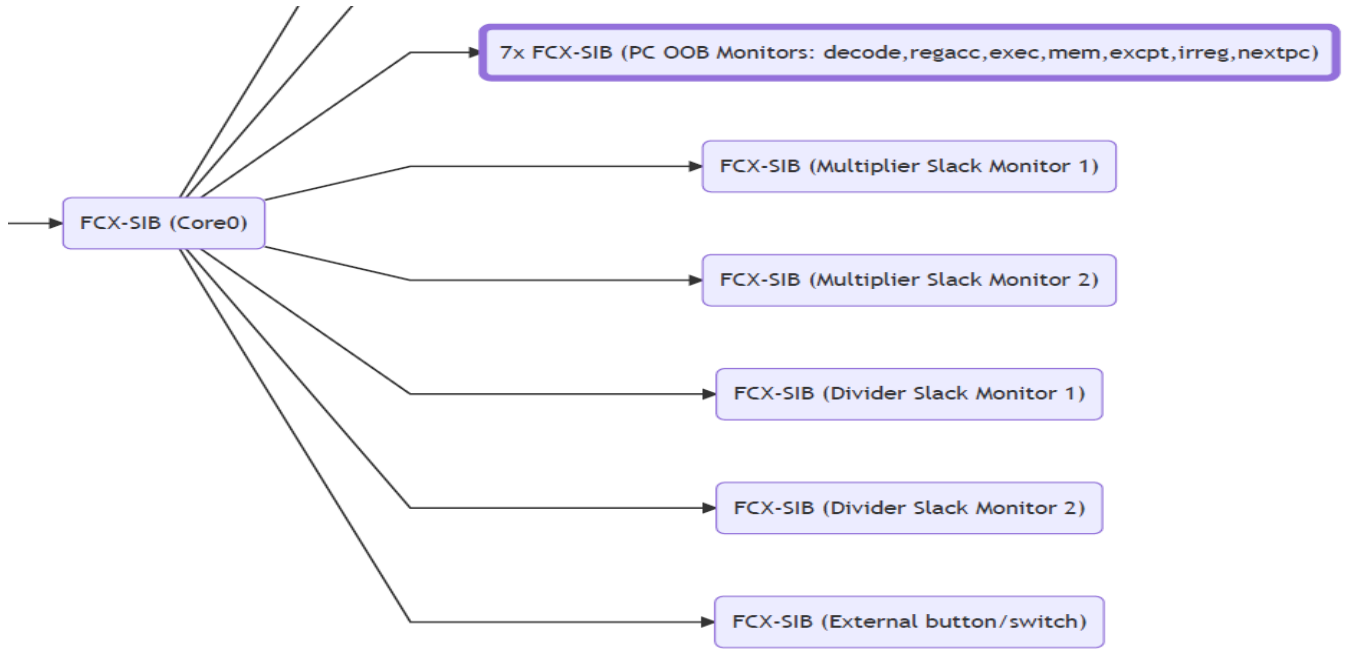
1262 checkers per system are handled

- Property and assertion checkers in larger modules and automata
  - 1088x register checkers 136 per core, which are
  - 8 global register + 16*8 windowed registers
- Watchdogs on I/Os and peripherals
  - 1x AHB/APB bus idle checker (per system)
  - 1x simple address/value pattern checker on AHB/APB bus (per system)
  - 8x push-button linked checkers activating fault signal on button press (1 per core)
- IC integrity and environmental monitors
  - Temperature sensor (per system)
  - 3x voltage sensors (per system)
  - 32x aging sensors that are located near specific core (4 per core)
- CPU specific monitors
  - 64x Program Counter (PC) idle detection (8 per core, one per pipeline stage)
  - 64x PC out of range detection (8 per core, one per pipeline stage)

Instrument Hierarchy

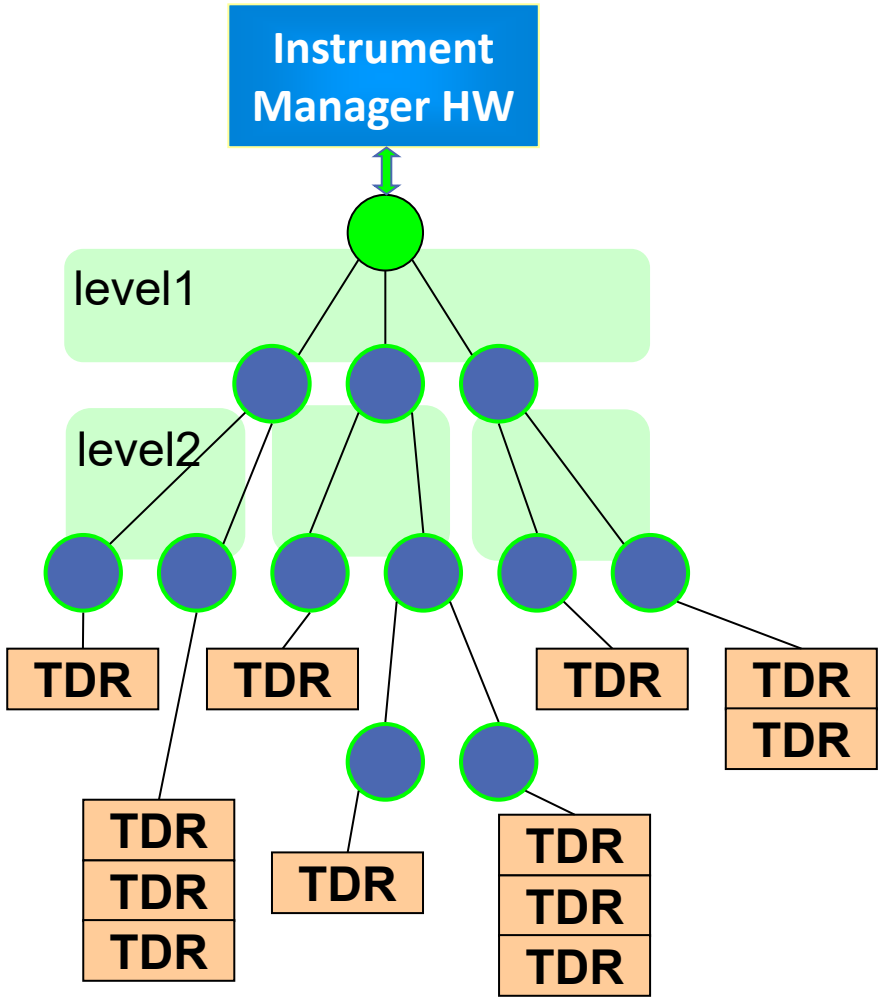


- Hierarchical IEEE 1687 IJTAG network
- Instrument priorities define the hierarchy (4 levels)
- Asynchronous Flag Propagation Network (AFPN)
- Fault detection latency 3 TCK + combinational delay
- 278 TCK (clocks) worst fault localization time
- Both scale logarithmically w.r.t. instrument count



WHAT ABOUT OVERHEAD?

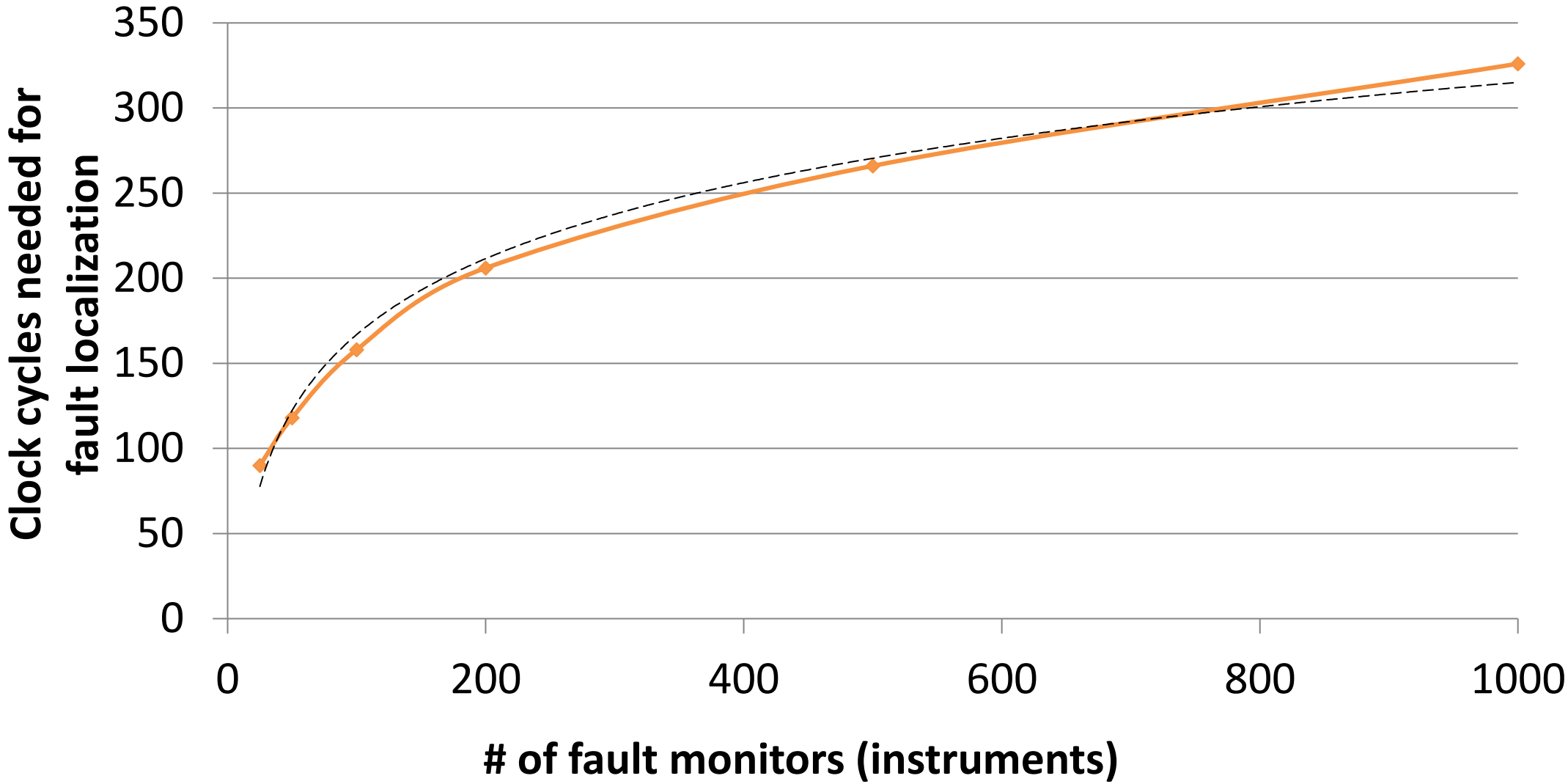
(Balanced) Tree Network Topology



- Balanced tree structure is in general close to optimum
- Network configuration can be based on error probabilities in resources
- Resources with higher error probabilities to be put closer to the top
- Very deep hierarchy is not advisable

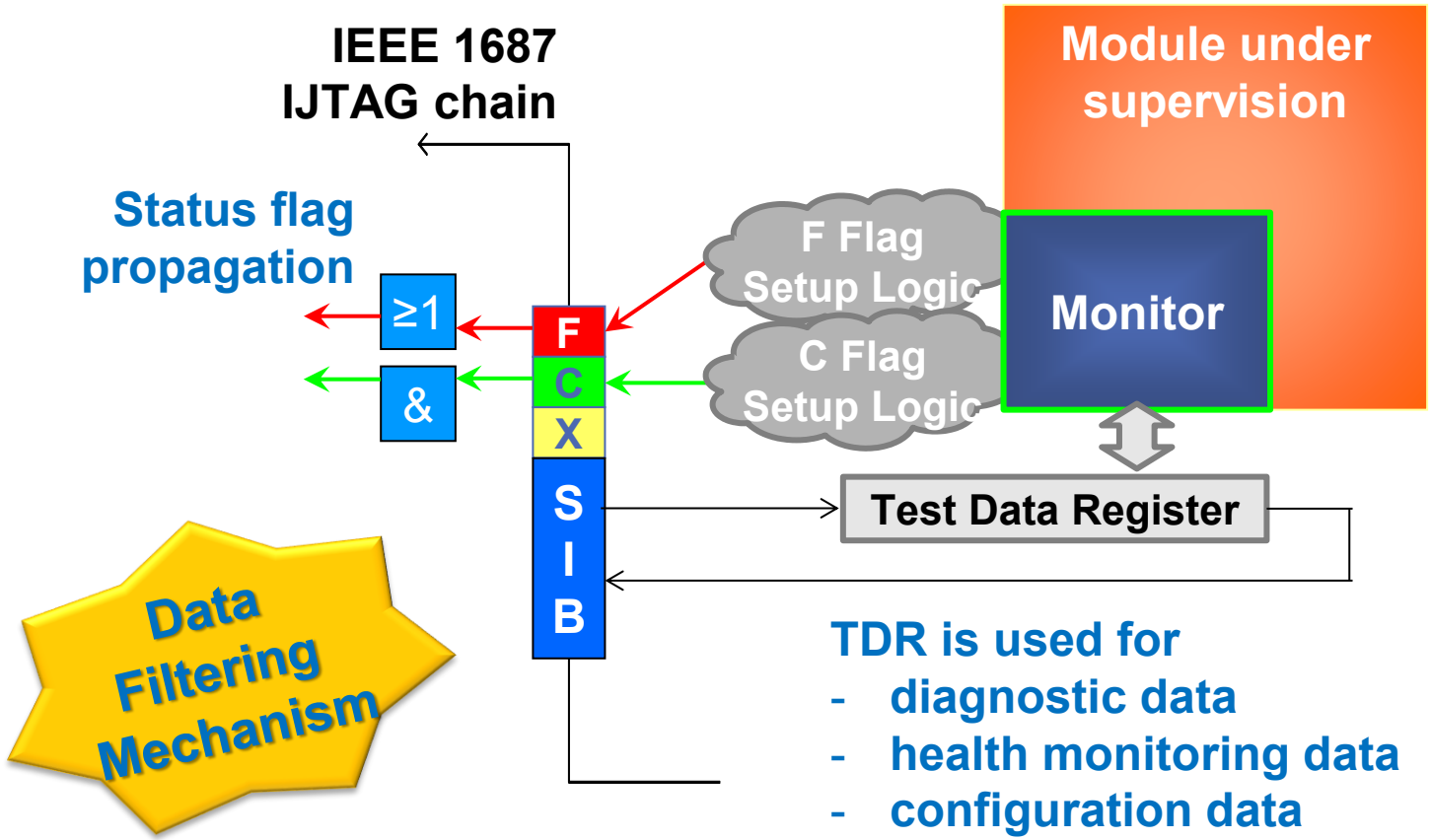
- As a result: the Instrument Manager can be implemented as a simple HW module (FSMD)

Fault Localization Latency



ON-CHIP DATA FILTERING

Sensor With Emergency Flags



F - Fault
- 0 = no fault
- 1 = fault occurred

C - Correction
- 0 = fault not corrected
- 1 = fault corrected

X – Mask (eXclude)
- 0 = fault propagated higher
- 1 = fault is masked, no propagation

Aging Sensor example

- Pre-program aging accumulation values
  - E.g. delay values in aging sensor
  - IMPORTANT: multiple sensor readings (sensor fusion) may be needed to make a conclusion (aging + thermal + voltage)
- Use F=1, C=1 (fault mitigated)
  - When the stress level needs to be reduced
  - When the damage can be recovered (rejuvenated)
- Use F=1, C=0 (not mitigated) for critical alarm
  - When the resource is close to end of life

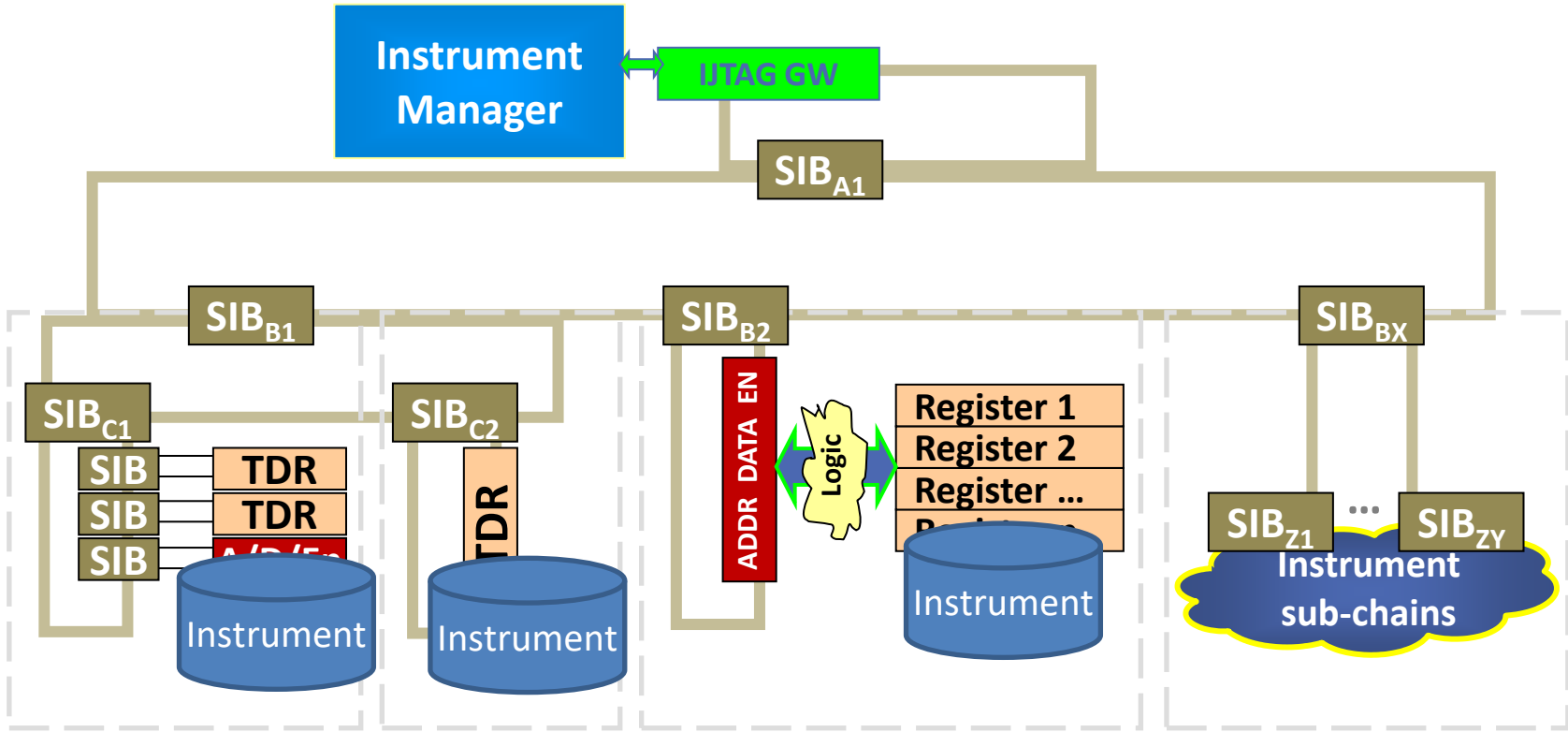
Thermal Sensor example

- Pre-program warning level values
  - Several warning levels are also possible
  - Example: $t^{o}_{w1}=60$; $t^{o}_{w2}=70$; $t^{o}_{e}=80$;
- Use F=1, C=1 (fault mitigated) for warning delivery
- Use F=1, C=0 (not mitigated) for critical alarm

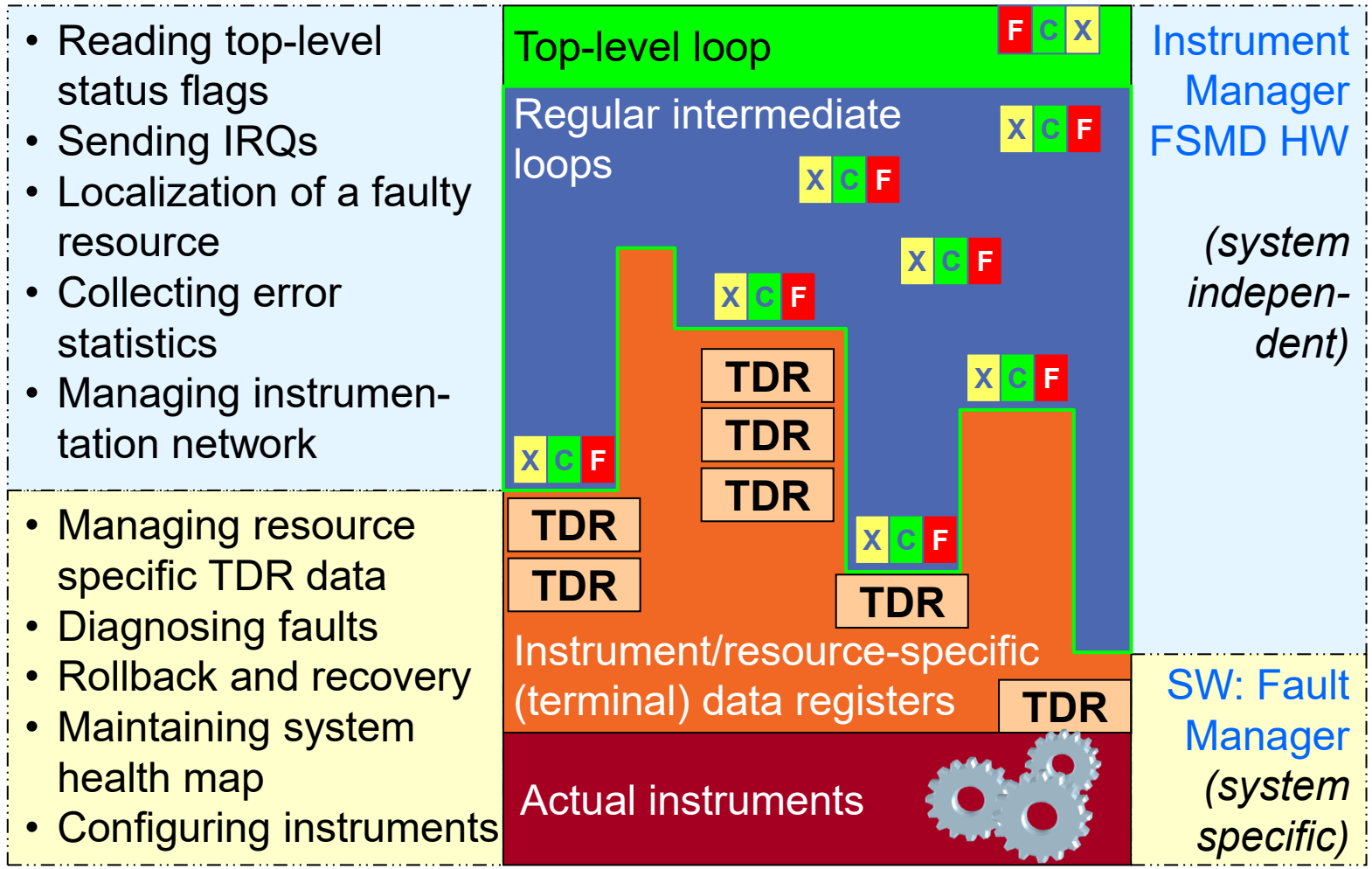
Instruments and IEEE1687 IJTAG

- Detect faults in resources
- Have to be managed

- Connects instruments in flexible and efficient way



Test Access Levels: HW vs. SW



DEGRADATION ASSESSMENT

System lifecycle stages

All cores healthy

- Infrequent diagnostic procedures

Healthy

Some resource faulty

- Avoid scheduling tasks to faulty resources

Damaged

Insufficient healthy resources

- Graceful performance degradation

Faulty

Degradation Assessment

Persistence

- Transient
- Intermittent
- Permanent

Severity

Influence on the resource

Shows the **extent of the damage**

Criticality

Criticality of the **faulty resource** for overall system's operation.

Diagnostic granularity

- Submodule, registers/gates
- Monitor-related
- System module

Fault diagnosis

Overall fault descriptor combining:

- location
- granularity
- severity
- criticality
- persistence

Adaptive fault resilience

- **Adaptive health monitoring and fault resilience**
  - Monitoring effort can be dynamically changed
  - Trade-off between overhead and fault detection probability
- **Examples**
  - Per-task execution redundancy (DMR/TMR)
  - Health monitoring enabled 10% of the time
  - A module had intermittent faults -> less trusted, more monitoring
    - Fault records in health map suggest lower reliability
    - Redundant execution should be scheduled more frequently
  - QoS based on resource health
    - Critical tasks can be scheduled only to healthy modules

Fitness Index Calculation

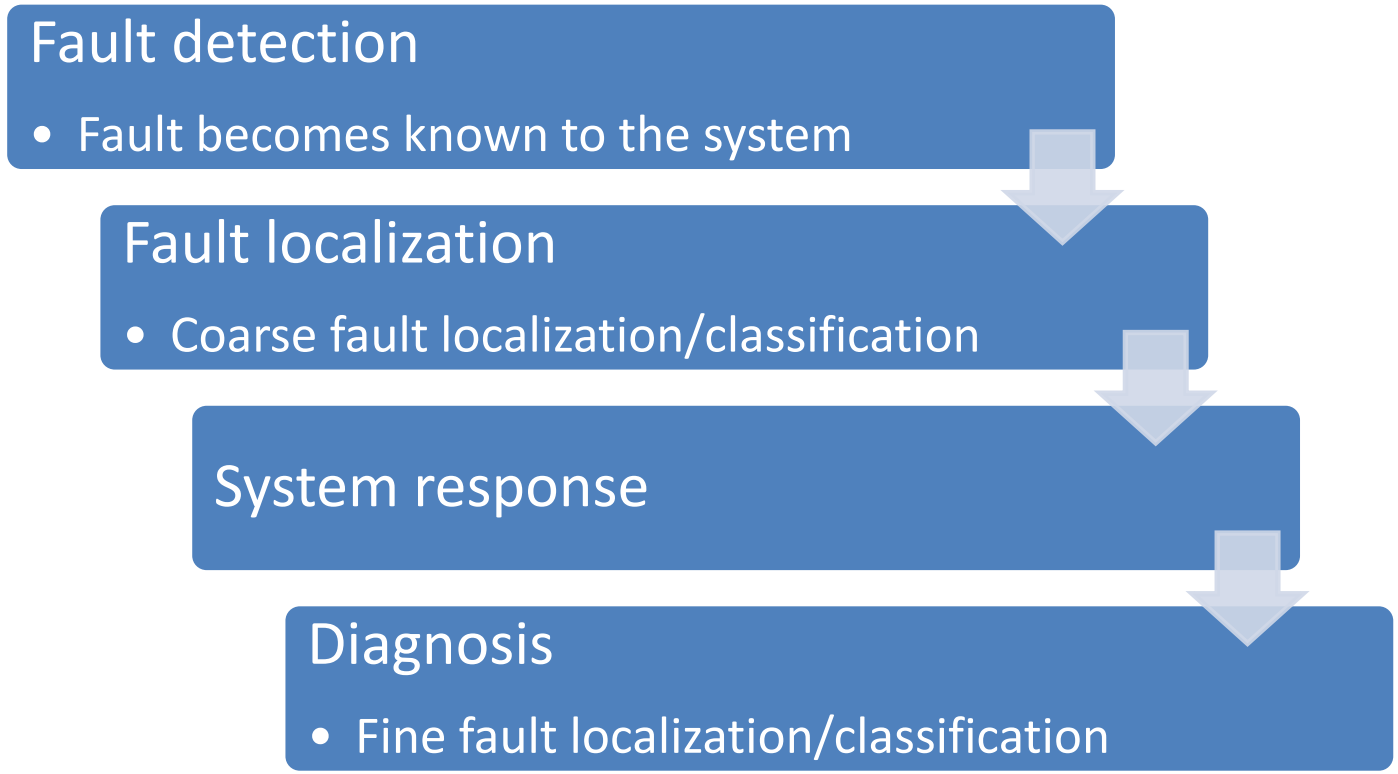
Fitness indices are calculated based on the "redundancy ratio" of top-level system resources needed to execute the tasks

- Task fitness index (TFI) =
  number of available resources / number of required resources
- System fitness index (SFI) = average of all TFIs
- SFI shows the average redundancy of system resources w.r.t. currently running tasks

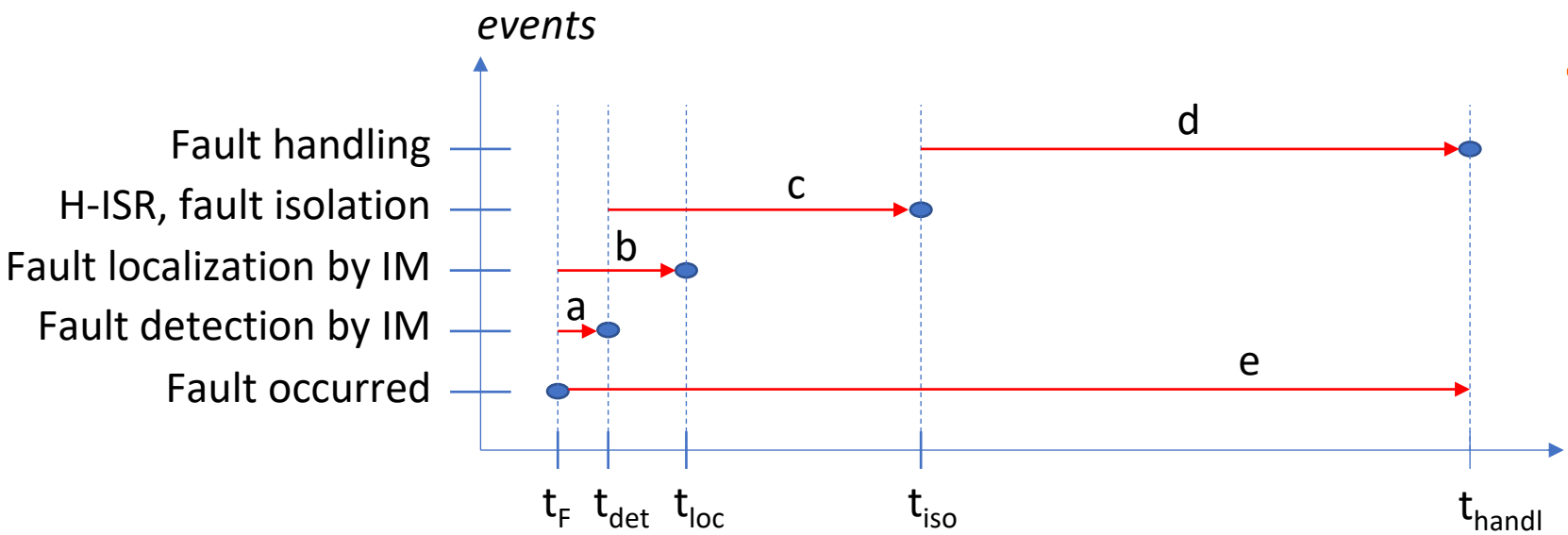
When the TFI reaches 1 for some task, a significant fault in any resource required by this task will lead to inability to run this task and, therefore, a system failure.

FAULT HANDLING DETECTION AND RECOVERY

On-Board Fault Handling



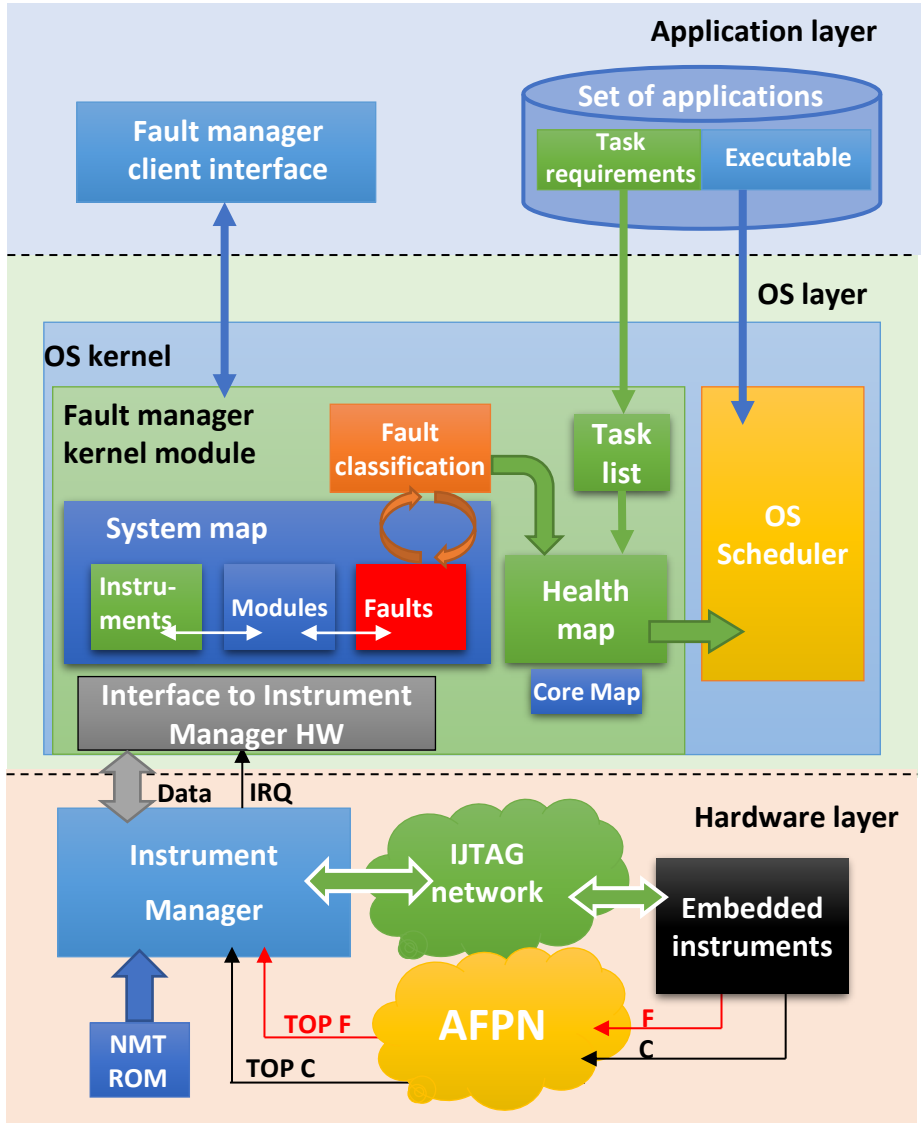
Performance Evaluation



The overall fault handling latency is in the order of 10^{-3}s

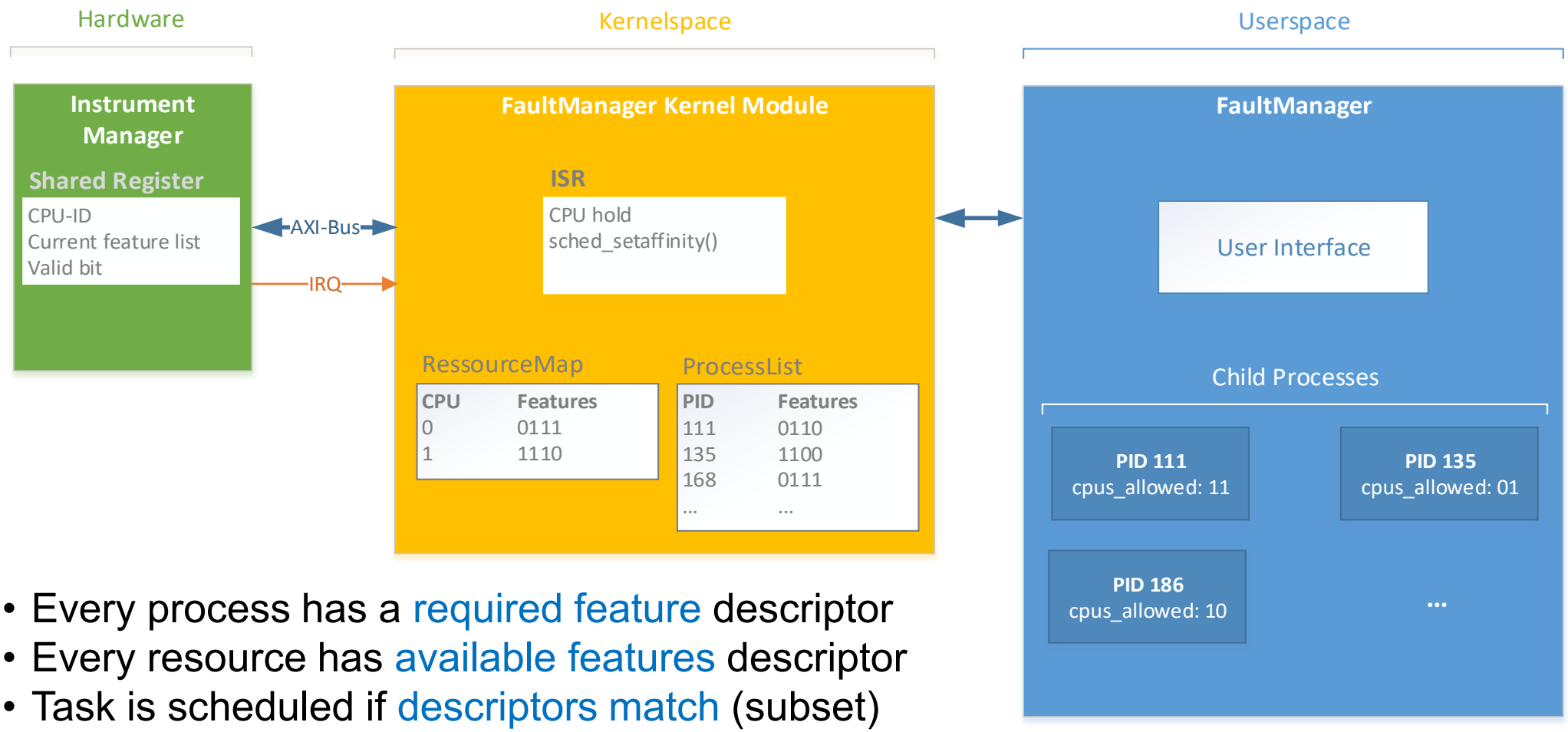
Measurement		Time, clock cycles	Time, μs
a) Fault detection latency	*const*	3	**0.074**
b) Fault localization latency	min–max	37–342	0.97–8.42
c) OS interrupt latency	min–max		38.45–52.75
	average μ_c		**47.39**
	std.dev. σ_c		3.62
d) Fault handling latency ($\mu_e - \mu_c - a$)	average μ_d		**5156**
e) Total time	min–max		4922–5800
	average μ_e		**5203**
	std.dev. σ_e		213

Cross-Layer Data Collection & FDIR



AFPN – Async Flag Propagation Network
NMT – Network Map Table
FDIR – Fault Detection Isolation & Recovery

Descriptor-Based Scheduling



- Every process has a required feature descriptor
- Every resource has available features descriptor
- Task is scheduled if descriptors match (subset)
- Health map contains updated valid descriptors

Scheduling in Asymmetric System

When features of a core become faulty, the system is not symmetric anymore. Processes assigned to cores according

- to the required features of the process
- to the provided (healthy) features of the cores

How to prescribe the set of valid cores for a certain process?

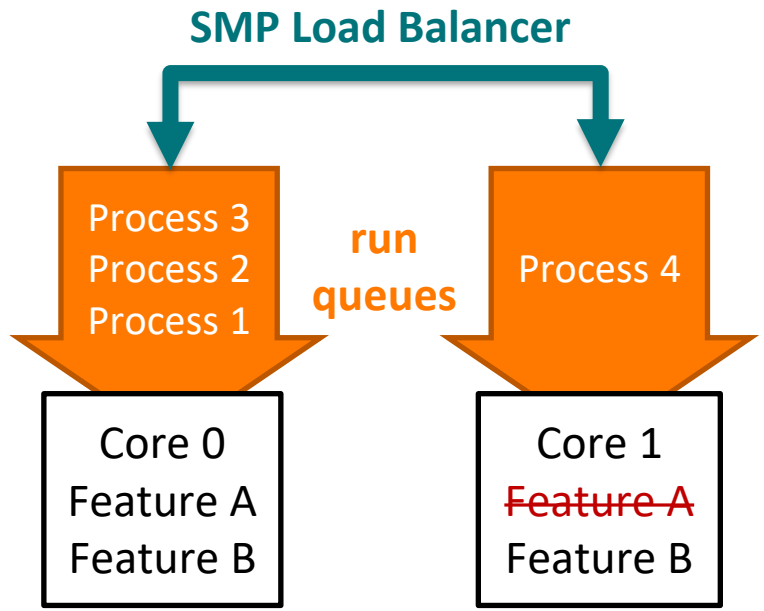
Each process has the affinity bitmask

`cpus_allowed` that specifies allowed cores

→ The system call

`sched_setaffinity()`

is used to modify this affinity mask.



KEY TAKEAWAYS

DFT Reuse Across Silicon Lifetime

Scan chains	Used for wafer/factory diagnosis, but usually too intrusive for continuous online use
LBIST	Can be reused at boot or during maintenance windows; online use is difficult
MBIST	Usually not reused: online use usually means background scrubbing, idle-bank test, or limited destructive/non-destructive modes
JTAG/IJTAG	Technically reusable across the lifecycle, but online/field access might be restricted by security policy
Parametric monitors	Full lifecycle reuse from wafer characterization through field SLM
Protocol checkers, watchdogs, ECC	Highly valuable from SLT onward, but less relevant for wafer structural test
V/F profiling	Valuable during SLT, binning, maintenance diagnosis, and adaptive runtime control
Aging monitors	Read early for baseline calibration, and used across complete lifetime operation

On-Chip Data Collection Enables

- Instant failure recovery (FDIR)
  - Instant fault reporting to OS
  - Task relocation/rescheduling
- Adaptation to degradation (SLM)
  - Damage-aware reconfiguration (healing)
  - Damage-aware scheduling
- “Preuse” of monitoring infrastructure at earlier test stages
  - wafer & assembly test
  - SLT – System Level Test
- Facilitates AI-driven intelligent methods (prognostics)
  - Event-triggered big data filtering by sensors
  - Detailed diagnostics data available