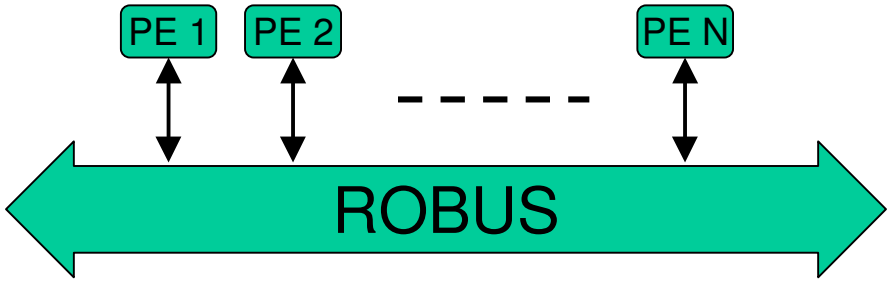
ROBUS 2

A fault-tolerant broadcast communication system



Wilfredo Torres

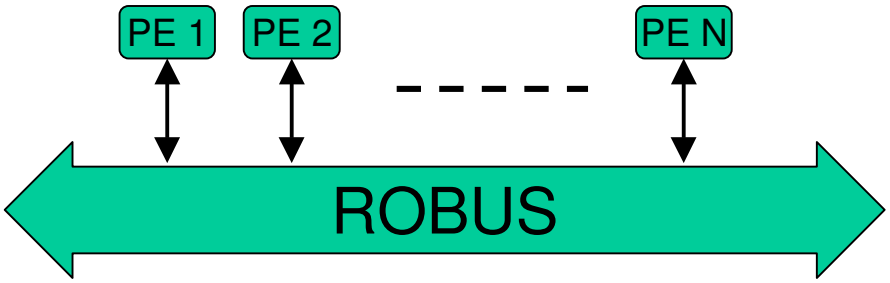
Mahyar Malekpour

Paul Miner

Outline

- What is ROBUS 2?
- Services and features
- System structure
- Distributed coordination
- Redundancy management
- High-level operation
- Implementation

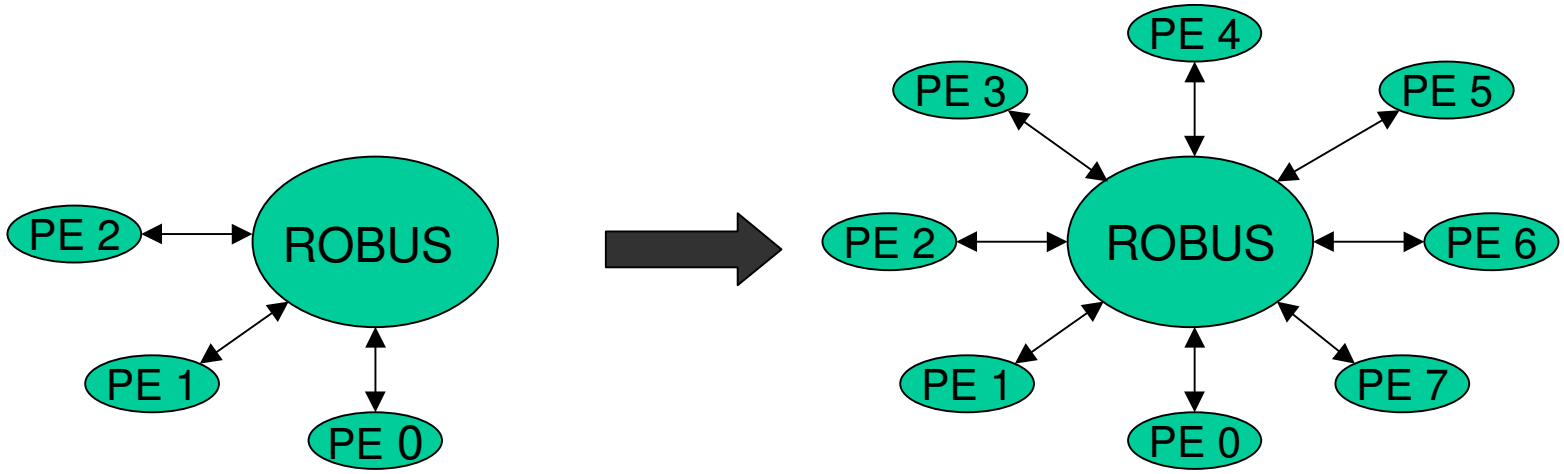
What is ROBUS?



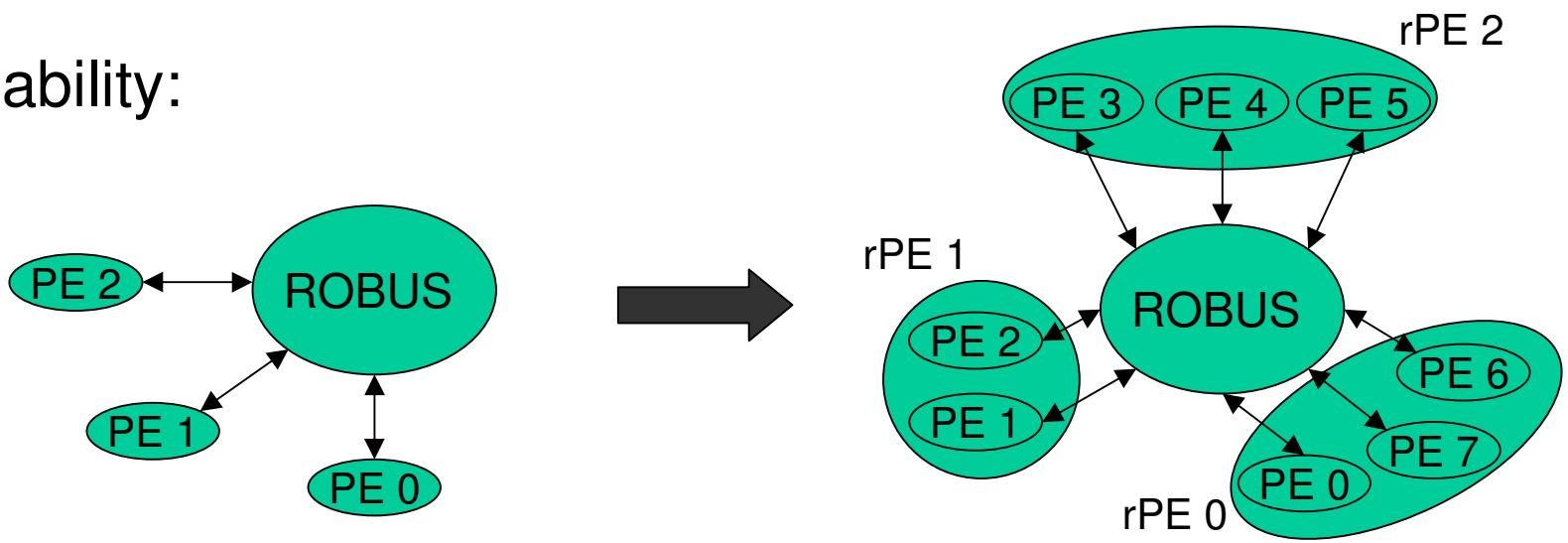
- SPIDER = Scalable Processor-Independent Design for ER
  - Integrated modular avionics (IMA) platform scalable for reliability and performance
- Reliable Optical BUS (ROBUS)
  - Basic communication services
- SPIDER Operating System
  - Communication, process management, and redundancy management at Processing Element (PE) level

Scalability on SPIDER

Performance:



Reliability:



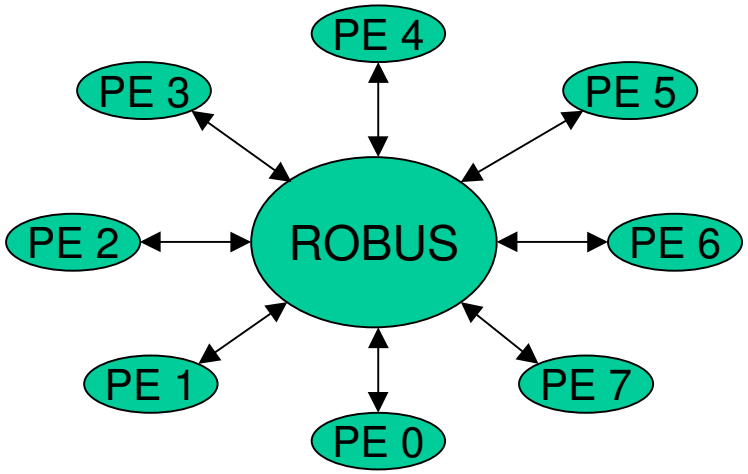
ROBUS Design Evolution

- ROBUS 1
  - Static communication schedule
  - Sequential communication
    - Throughput: 1 message in the system at a time
  - Full-configuration startup only
  - Reconfiguration only through degradation

- ROBUS 2
  - Dynamic schedule update
  - Pipelined communication
    - Maximum throughput: 1 message per tick
  - Enhanced error detection
  - Variable-configuration startup
  - Reconfiguration through degradation and reintegration
  - Automatic restart

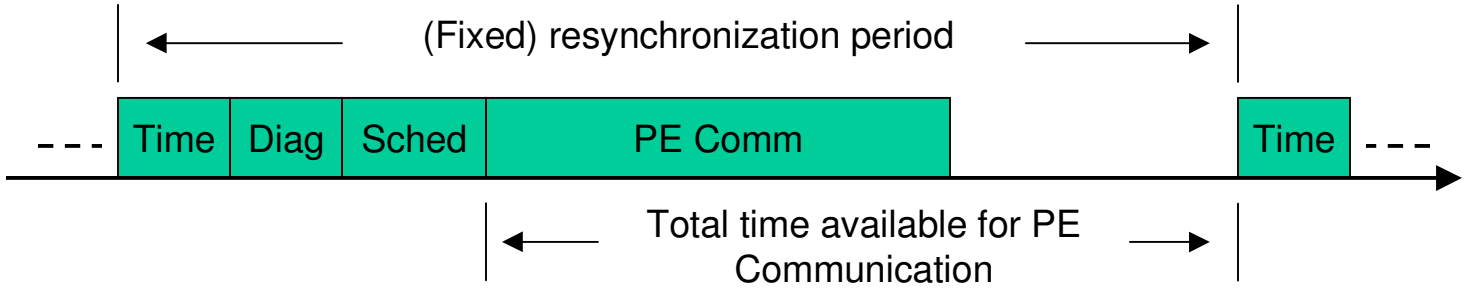
ROBUS 2: Services and Features

- Services (fault-tolerant)
  - Message broadcast
  - Communication schedule update
  - Time reference
  - Self-diagnosis

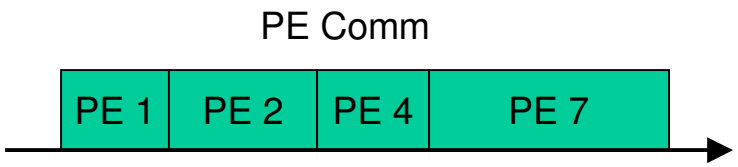


- Other features
  - Time-triggered operation
  - Communication-schedule enforcement
  - Self-reconfiguration (degradation and reintegration)
  - Internal-fault masking
  - Fault-tolerant startup and restart
  - PE-fault tolerance

ROBUS 2: Steady-State Behavior



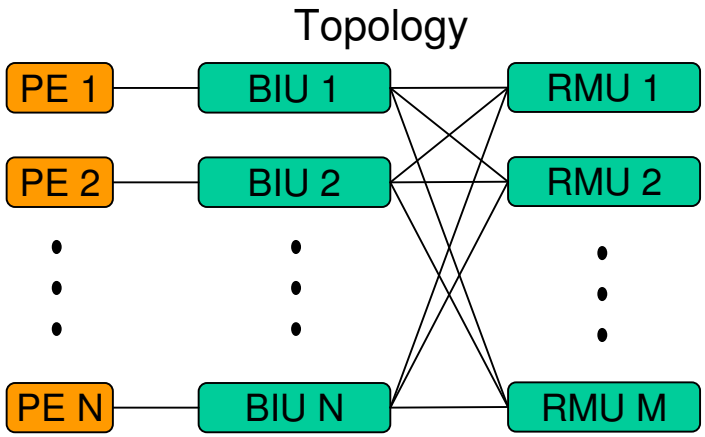
- Round/cycle-based operation
- Time update
  - "Bus time" only; not synchronized to "external time"
  - Common time reference enables the PEs to coordinate their actions
- Self-diagnosis update
  - Used by PEs for SPIDER-level process and redundancy management
- Communication-schedule update
  - Specify number of messages for each PE
  - On bad schedule, switch to default schedule



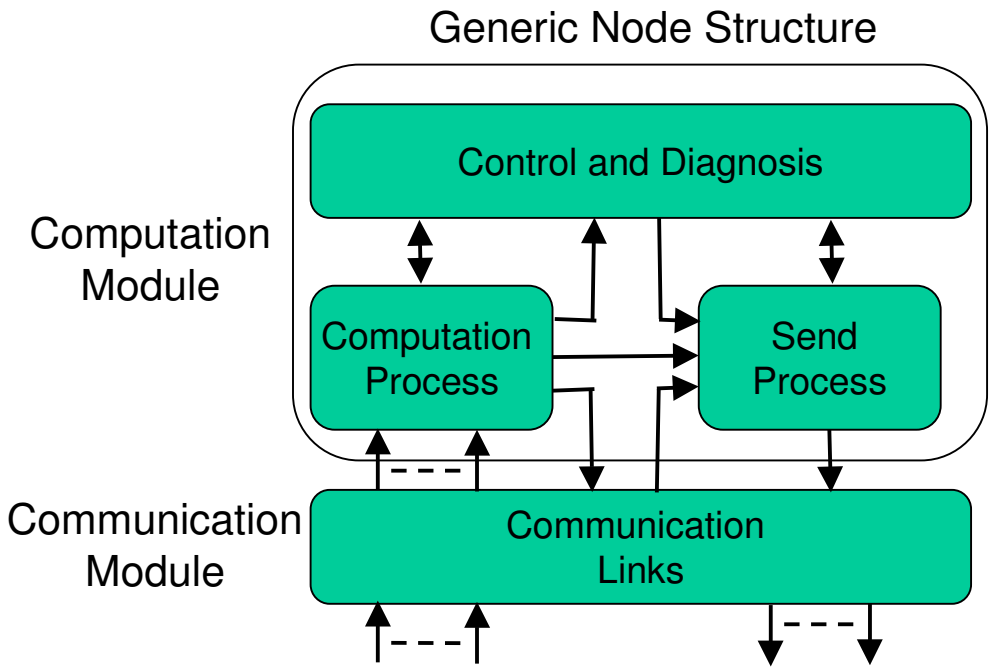
- PE communication
  - Communication pattern: Time-driven, ASAP round-robin

ROBUS Structure

- ROBUS = nodes + links
  - Complete bipartite graph
- ROBUS nodes
  - Bus Interface Units (BIUs) = Ports
  - Redundancy Management Unit (RMU) = Hubs



- Communication Module
  - Generic COTS components
  - One-to-one or one-to-many links
- Computation Module
  - Implements ROBUS-specific functions
    - Low-level protocols
    - Error detection
    - Diagnosis
    - Reconfiguration
    - Mode logic

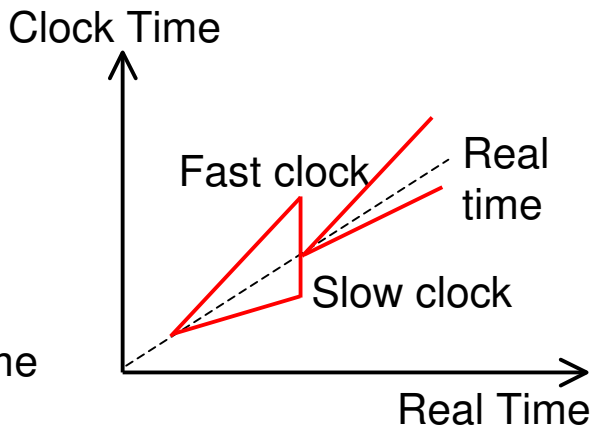


Distributed Coordination

- "What to do?" and "Where to do it?"
  - High-level behavior and low-level protocols are specified to achieve the desired functionality
  - Result: operations, sequencing, and decision rules specified for all nodes
- "When to do it?"
  - Want deterministic timing
  - Some benefits of having a smaller timing uncertainty:
    - Predictable latency and small jitter (required for control applications)
    - Higher time resolution and reduced latency for error detection
    - Higher system performance
    - Refined analysis and design
  - Problems:
    - Ensuring minimal timing variation due to failed nodes
    - Remote communication
  - Our solution:
    - <u>Fault-tolerant Synchronization</u> and <u>Distributed Synchronous Composition</u>

Local-Time Clocks

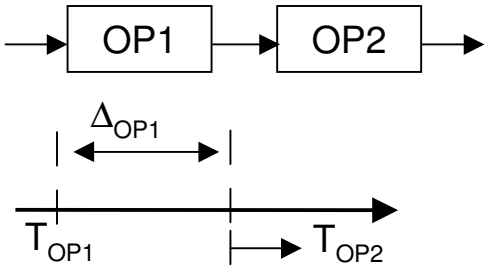
- Every node has a local-time clock
  - Clock is driven locally by an independent physical oscillator
  - Clock indicates "time since reference event"
    - Clocks are themselves "event generators"
  - Clocks drift with respect to each other and real time
    - Precision of the generated time events worsens over time
    - Assuming known bound on drift rate



- Synchronization protocol
  - Protocol delivers a high-precision distributed reference event which is used to reset the local-time clocks
    - Input: Distributed event with (possibly large) imprecision
    - Output: Distributed event with known small imprecision (independent of input precision)
    - Fault tolerant; Good precision and accuracy
  - Protocol also used to deliver time service to the PEs
  - Periodic resynchronization required to limit the relative local-time skew
  - Same basic protocol is used for bus startup and restart
    - Events: Power-On Enable and Failure Detection, respectively

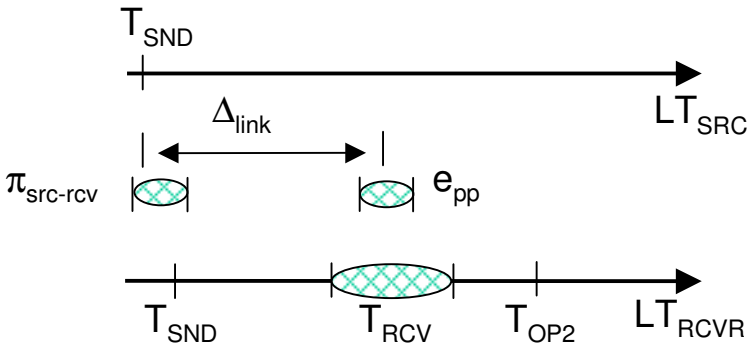
Distributed Synchronous Composition

- Operation scheduling (for non-clock-sync ops)
  - Trigger is a function of only the local time
  - Specify a time trigger that occurs after the input data is ready and the required computation resources are available.



$$T_{OP2} \geq T_{OP1} + \Delta_{OP1}$$

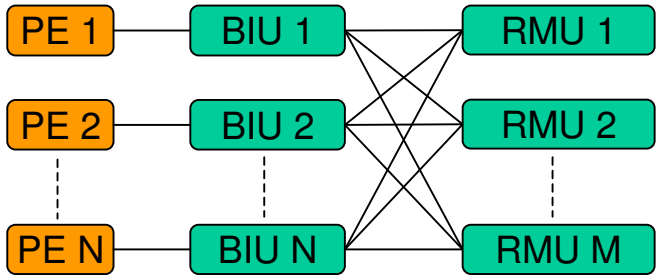
- Point-to-point synchronous communication
  - Send time is predetermined
  - Communication uncertainties:
    - Relative local-time skew ($\pi_{src\text{-}rcv}$)
    - Link delay (e_{pp})
  - Received data is buffered until it is time to process it



$$T_{OP2} \geq T_{SND} + \Delta_{COMM}|_{max}$$

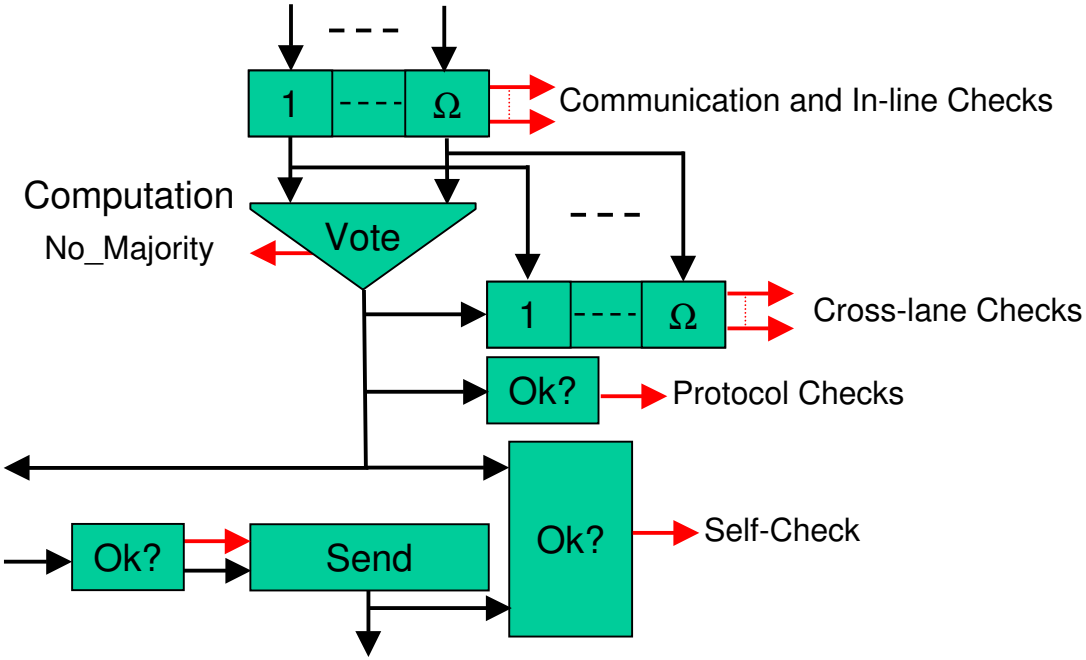
- Synchronous Composition
  - Scheduling based on an system model with fixed operational delays and components driven by a single oscillator and a single local-time clock

ROBUS Redundancy Management (1/3)



- Fault Containment Regions (FCR)
  - Ensure independence of physical faults
  - Each BIU and each RMU in a separate FCR
  - Each PE shares an FCR with its attached PE or is in a separate FCR
  - Issue: correlated faults caused by external threats (e.g., lightning)

- Error detection
  - Communication checks
  - In-line checks
  - Cross-lane checks
  - Protocol checks
  - Self-checks
  - PE-error checks



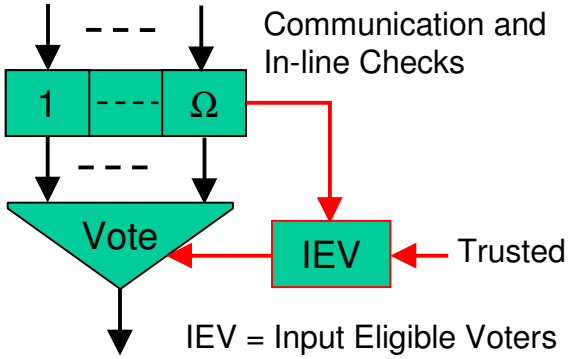
ROBUS Redundancy Management (2/3)

- Diagnosis
  - Trustworthy nodes: can be relied upon to deliver proper services
  - Untrustworthy nodes: Error sources
    - Factors: fault status (good/bad) and state variables (time + diagnosis + schedule)
  - Clique: group of trustworthy BIUs and RMUs that can deliver proper services
    - Mutual trust and agreement on the state variables
  - Local diagnosis: Each node independently diagnoses every node individually and the bus
    - Node blamed for an error => Accused (of being untrustworthy)
      - Local accusations held until end of current diagnostic cycle
    - Detected clique disintegration => Clique failure => Bus failure
  - Collective diagnosis: Distributed protocol used to generate consensus
    - There isn't a majority which does not accuse a node => Convicted
      - Convicted for 1 diagnostic cycle
  - Trust rule:
    - Trust a node only if it is not accused and not convicted

ROBUS Redundancy Management (3/3)

- Reconfiguration
  - Degradation
    - Removal on distrust
    - Trust reassessed for each computation (maximum rate: one-tick)
  - Reintegration
    - Admission on trust re-established
    - Only at diagnostic cycle boundaries

- Error containment
  - Dynamic voting
    - Eligible voter if no new input errors and trusted
  - Node fail-stop
    - On local failure
  - Bus fail-stop
    - On bus failure



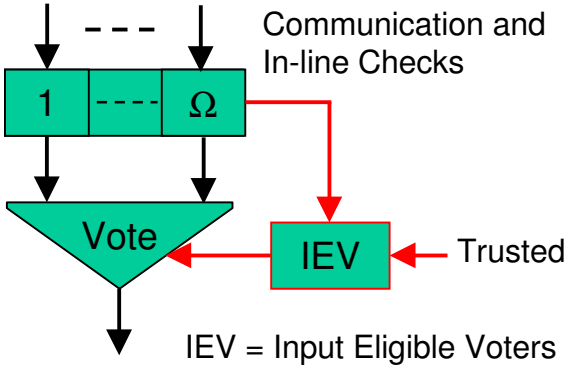
ROBUS Fault Model

- Fault characteristics:
  - Cause, Extent, Latency, Duration, Consistency, Direct detectability
- Critical aspect of ROBUS design:
  - agreement among redundant components
- Concern: threats to agreement generation and preservation
  - Focus on manifestations at the receivers

- Fault classification (for analysis and conceptual design):
  - Manifest:
    - Faulty behavior is detected by all receivers based on direct observation
  - Symmetric:
    - Faulty behavior is not detected through direct observation, but all receivers get the same manifestation
  - Asymmetric:
    - Not all the receivers get the same manifestation and at least one receiver does not detect the faulty behavior through direct observation

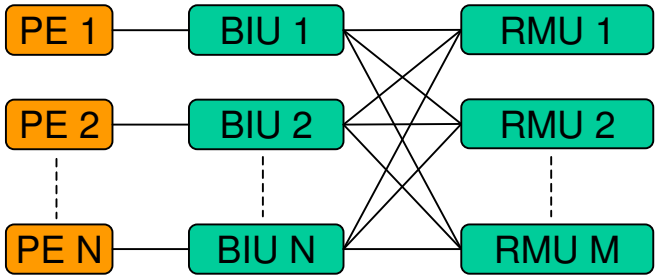
ROBUS Fault Assumptions

- Required for proper operation of a clique
  - Ensure Validity and Agreement on protocol results
  - Preserve Validity and Agreement on state variables

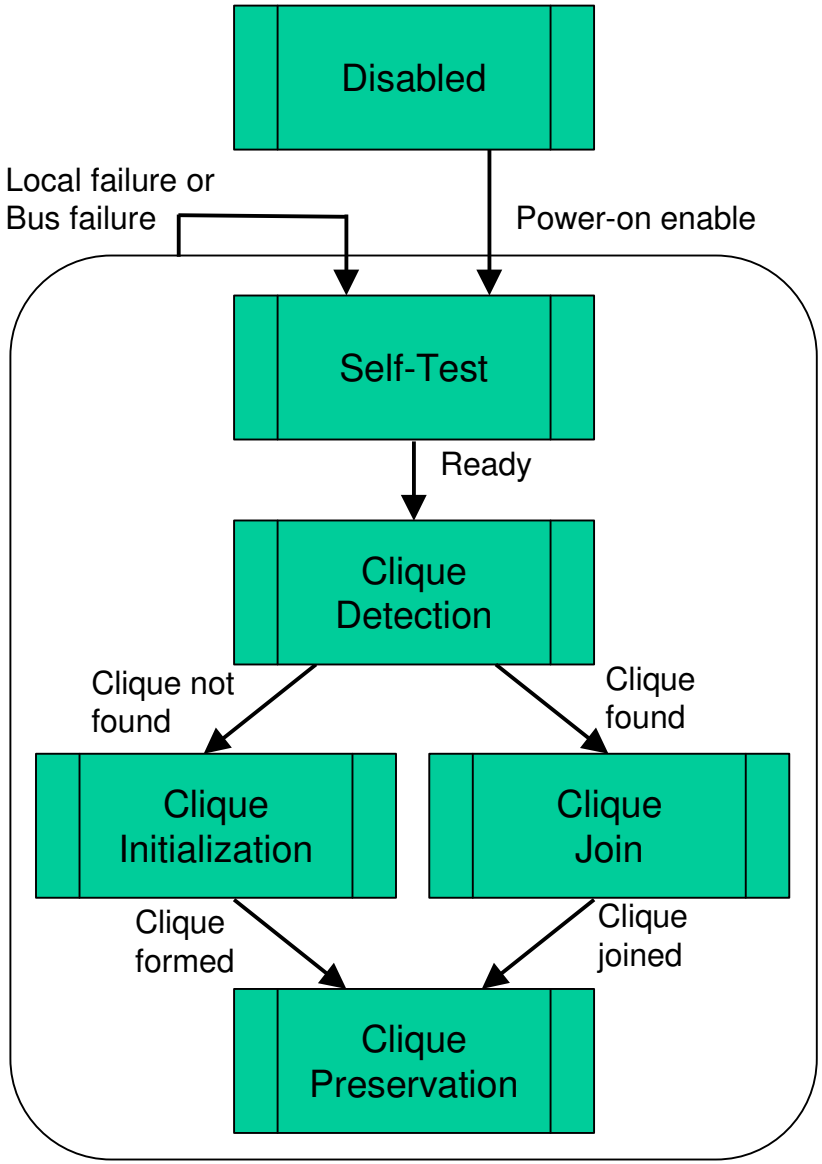


- For each low-level protocol execution:
  - At BIUs: |Trustworthy IEV| > |Untrustworthy IEV|
  - At RMUs: |Trustworthy IEV| > |Untrustworthy IEV|
  - At BIUs: |Asymmetric IEV| = 0 <u>or</u> At RMUs: |Asymmetric IEV| = 0

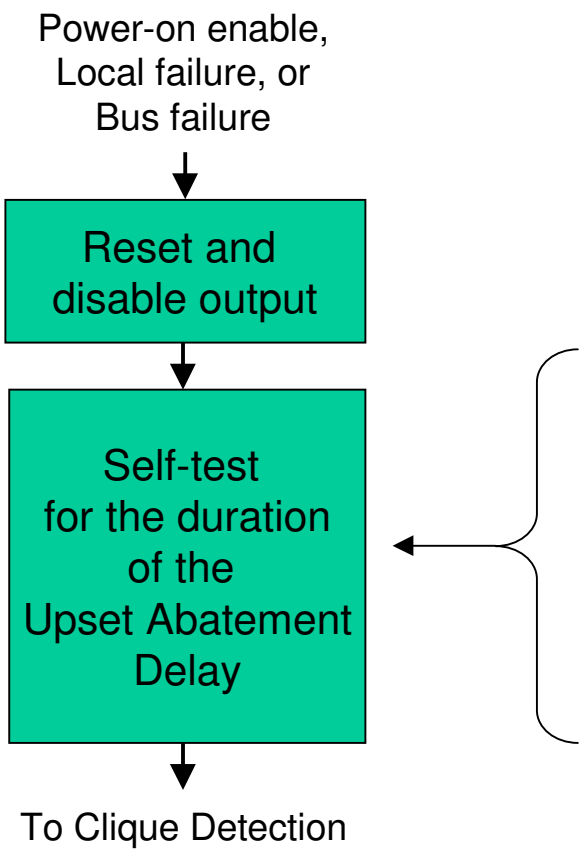
- Note:
  - SPIDER failure ≠ ROBUS failure



High-Level Operation

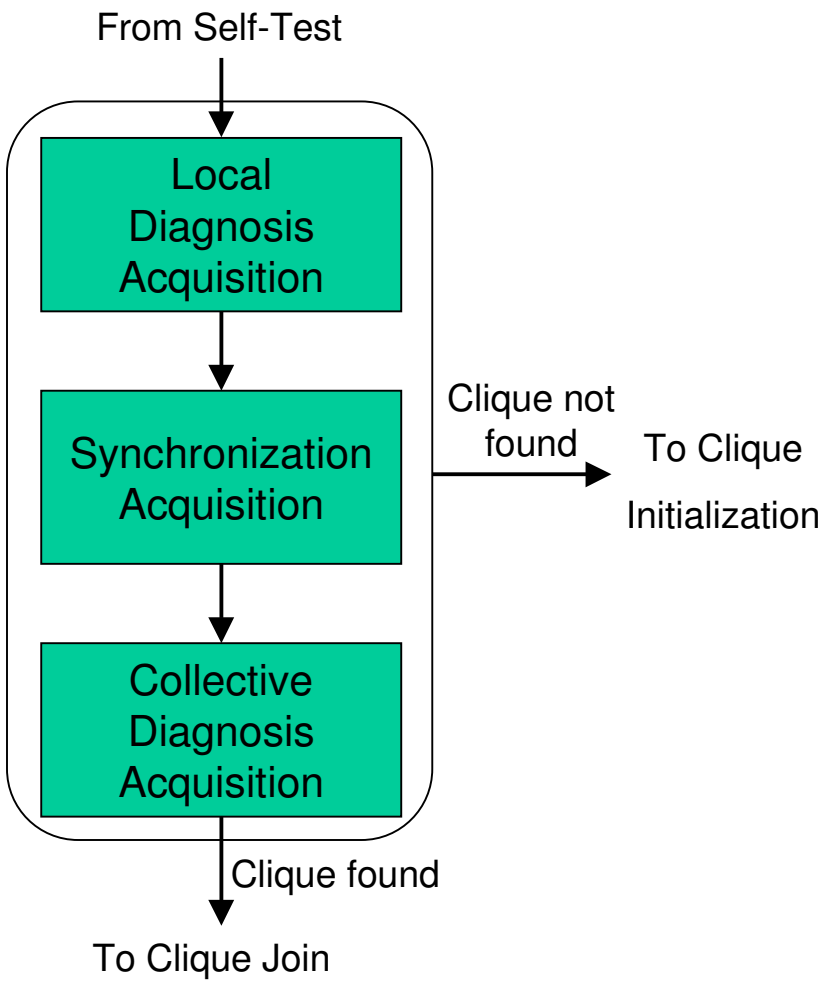


Self-Test

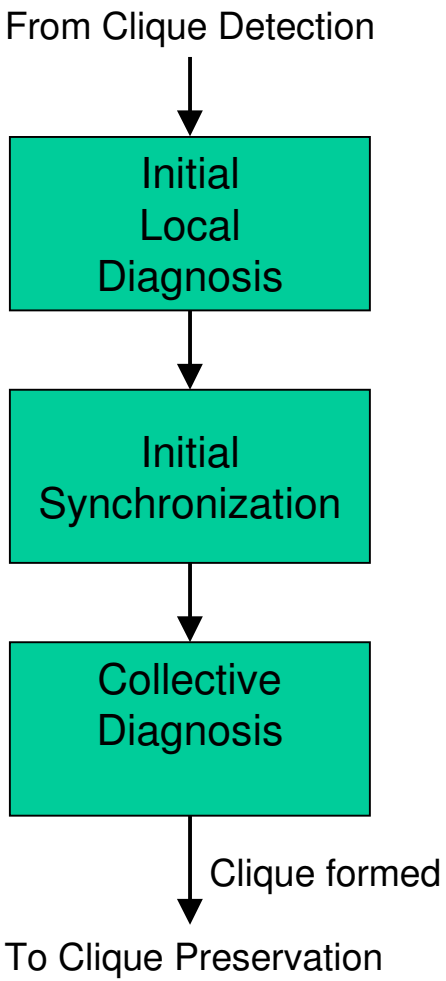


- Safe state
- Self-test failure => Local failure
- Upset Abatement Delay
  - Time for the bus to settle after the arrival of an external fault-causing phenomenon (e.g., lightning)

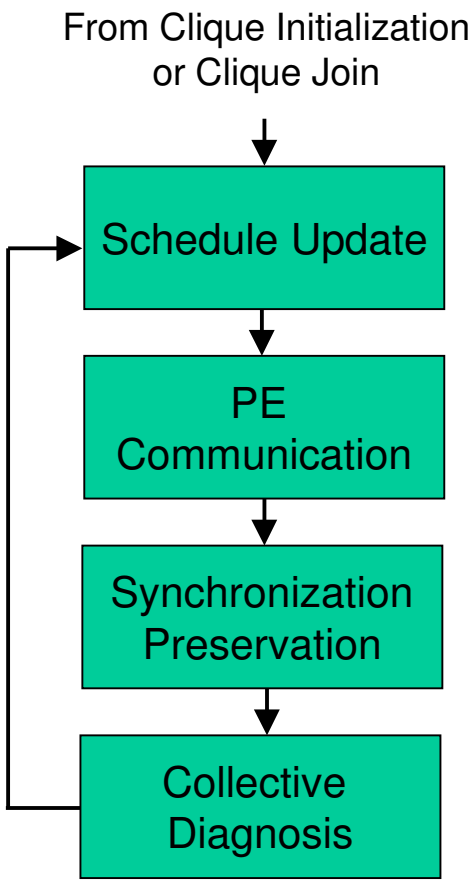
Clique Detection



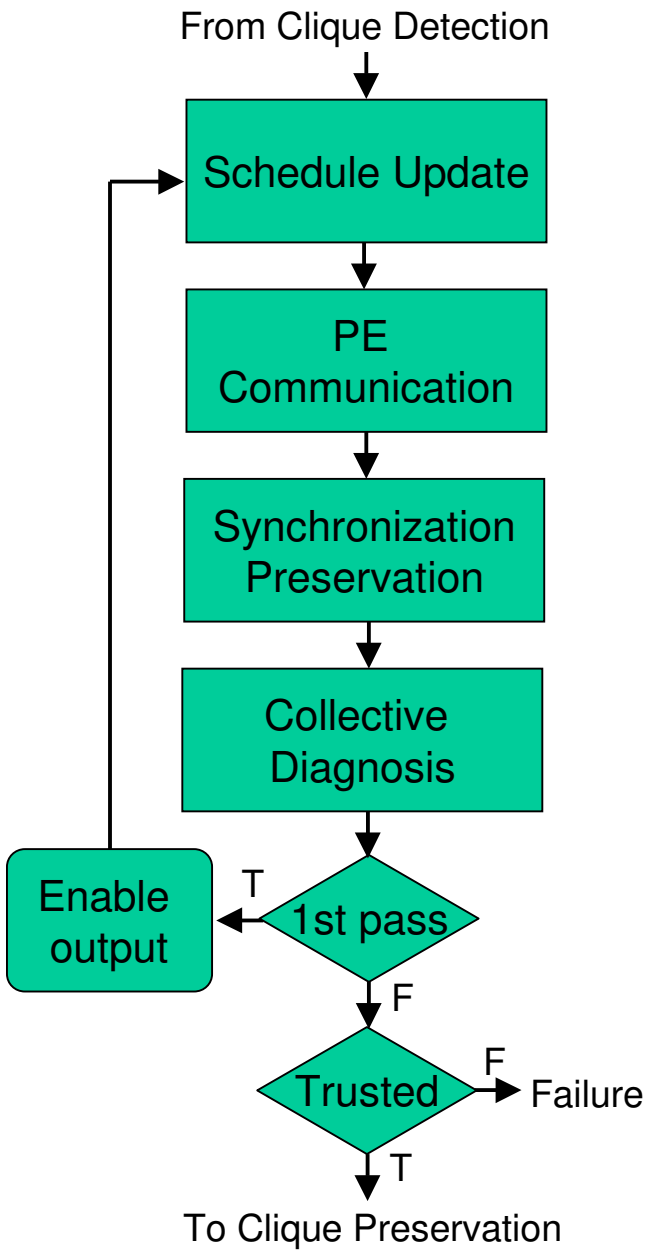
Clique Initialization



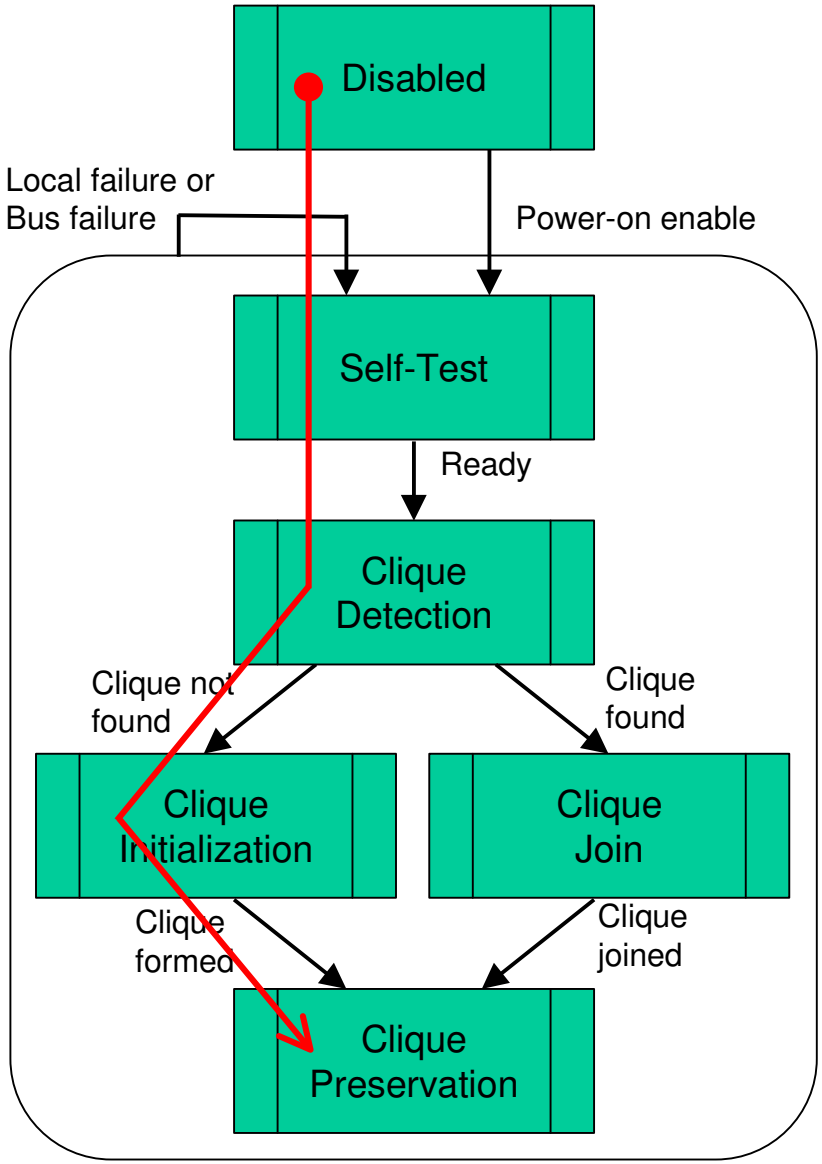
Clique Preservation



Clique Join

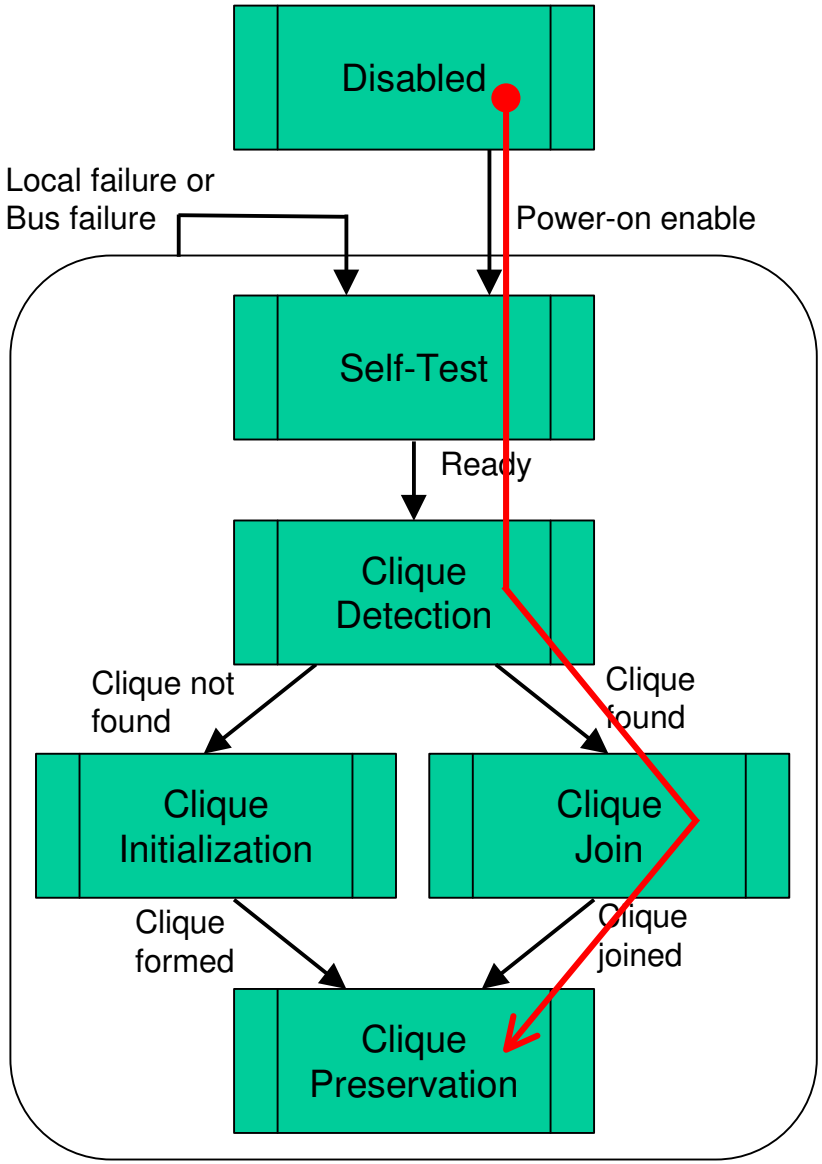


High-Level Operation



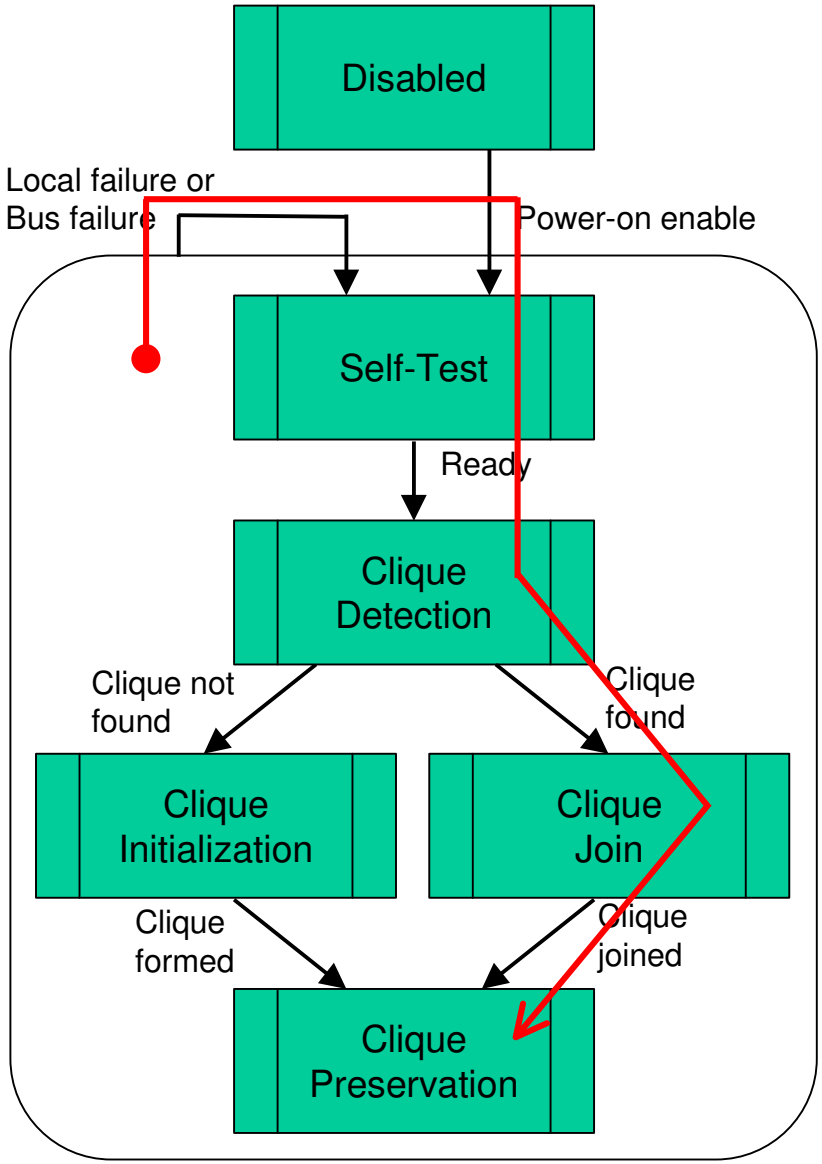
Clique
Initialization

High-Level Operation



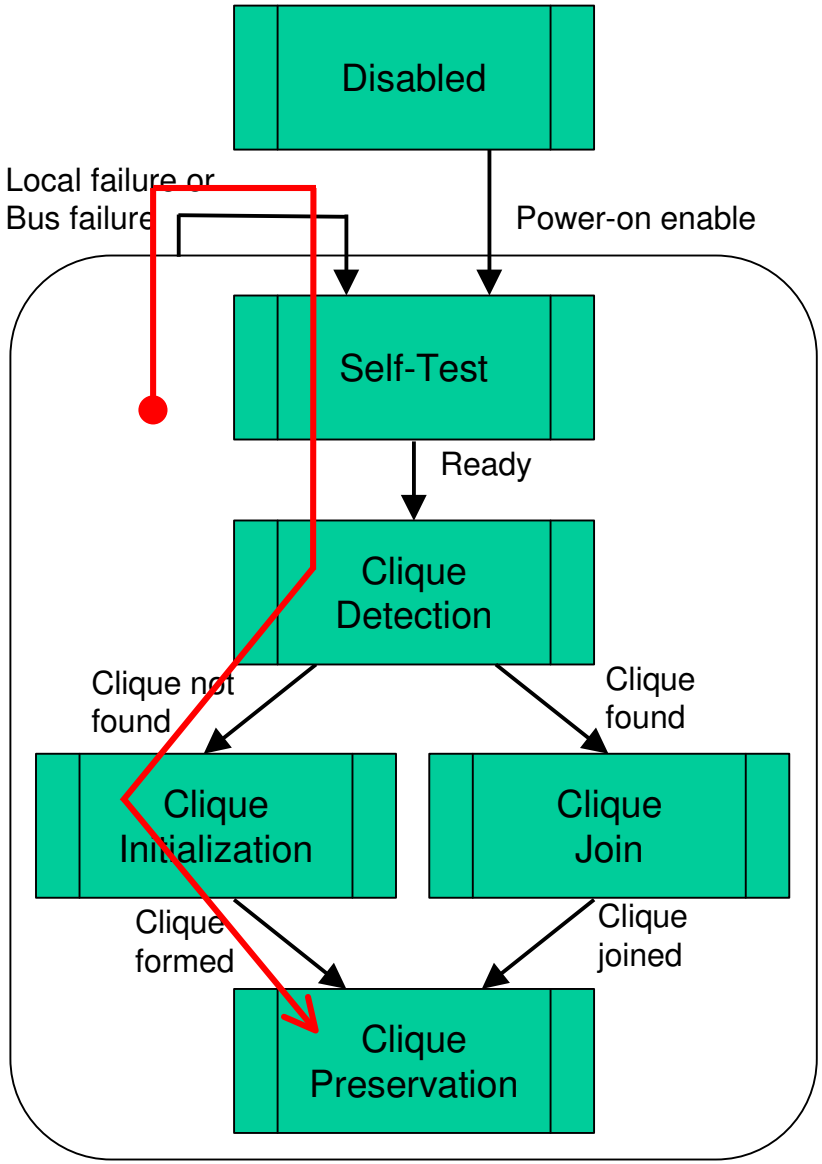
Node
Integration

High-Level Operation



Node
Reintegration

High-Level Operation



Clique
Re-initialization