Contract-based design, model checking, and model-based safety assessment

An integrated view

Alessandro Cimatti

Fondazione Bruno Kessler, Trento, Italy

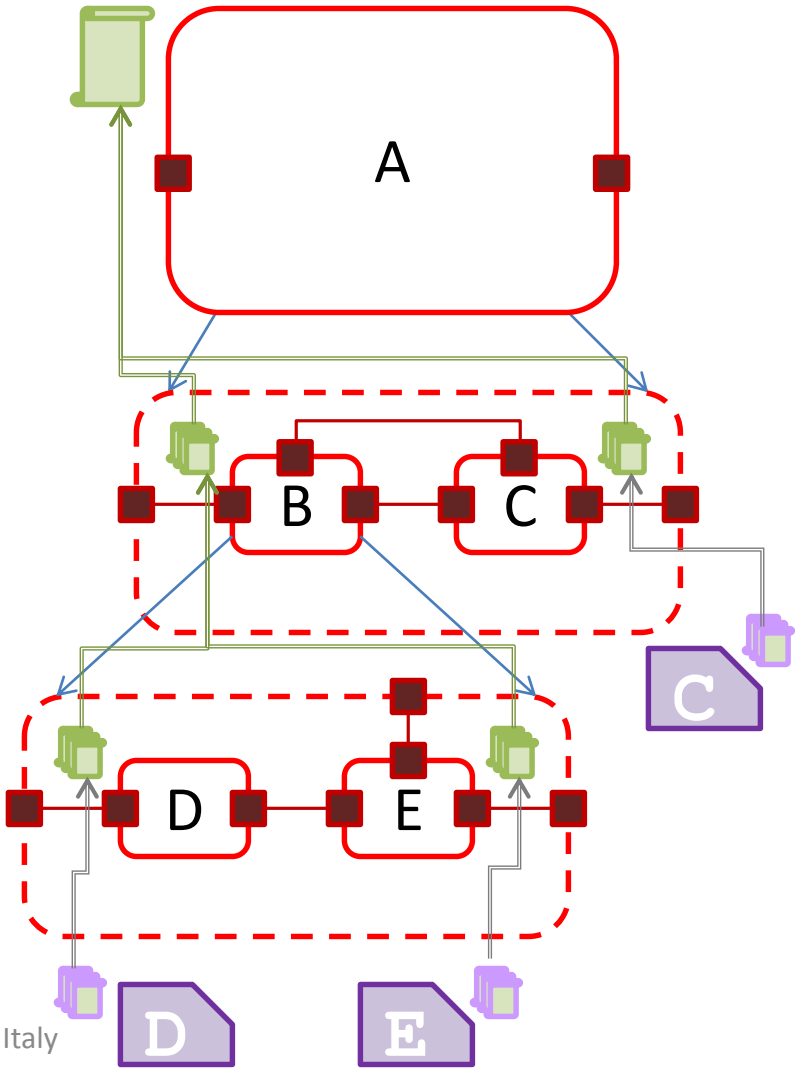
Take away message

- Beyond model checking: new generation of verification techniques
- Tools integrated into structured flow
- May provide novel support for CPSoS design challenges

- From model checking to …
  - Contract-based design
    - architectural decomposition + refinement of requirements
  - Safety analysis
    - Extend nominal model to include faulty behaviours
    - Fault Tree construction: detect all fault combinations causing loss of desirable property

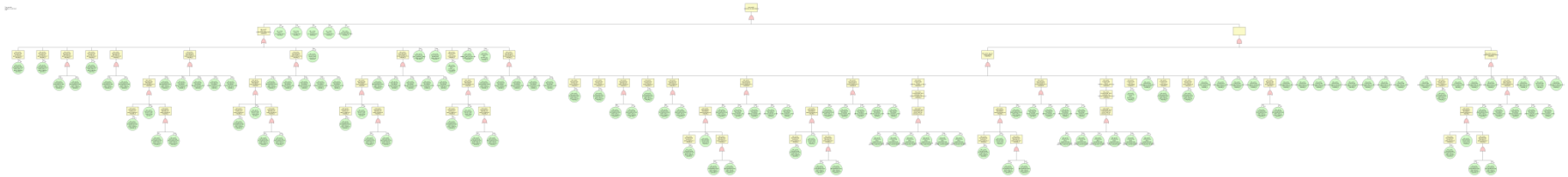
From architectural decomposition to contract-based design

- Hierarchical decomposition
  - Component to subcomponents
  - Implementation of leaf components
- Component associated with contracts
  - Assumptions / guarantees
  - Temporal logic
- Contracts refinement
  - Contract ensured by contract of subcomponents
- Correct implementations ensure correctness of composition



Model-based safety assessment

- Safety assessment
  - Analyze behaviour of system under faults
  - Artifacts: Fault Trees, FMEA tables
  - Qualitative and quantitative arguments



- Model-based Safety Assessment
  - Extend nominal model with faults
    - Symbolic fault injection
    - Valve stuck open, stuck closed, …
  - Analyze extended model
    - Automated production of FT

Formal Verification, Validation, and Safety Assessment

Model Checking

$$\mathcal{M} \models \varphi$$

Verification & Validation → **Safety Assessment**

Formal Verification, Validation, and Safety Assessment

Model Checking

$$\mathcal{M} \models \varphi$$

Fault Injection

$$\mathcal{M} \Longrightarrow \mathcal{M}_{[\mathcal{F}]}$$

Verification & Validation → **Safety Assessment**

Formal Verification, Validation, and Safety Assessment

Model Checking

$$\mathcal{M} \models \varphi$$

Fault Injection

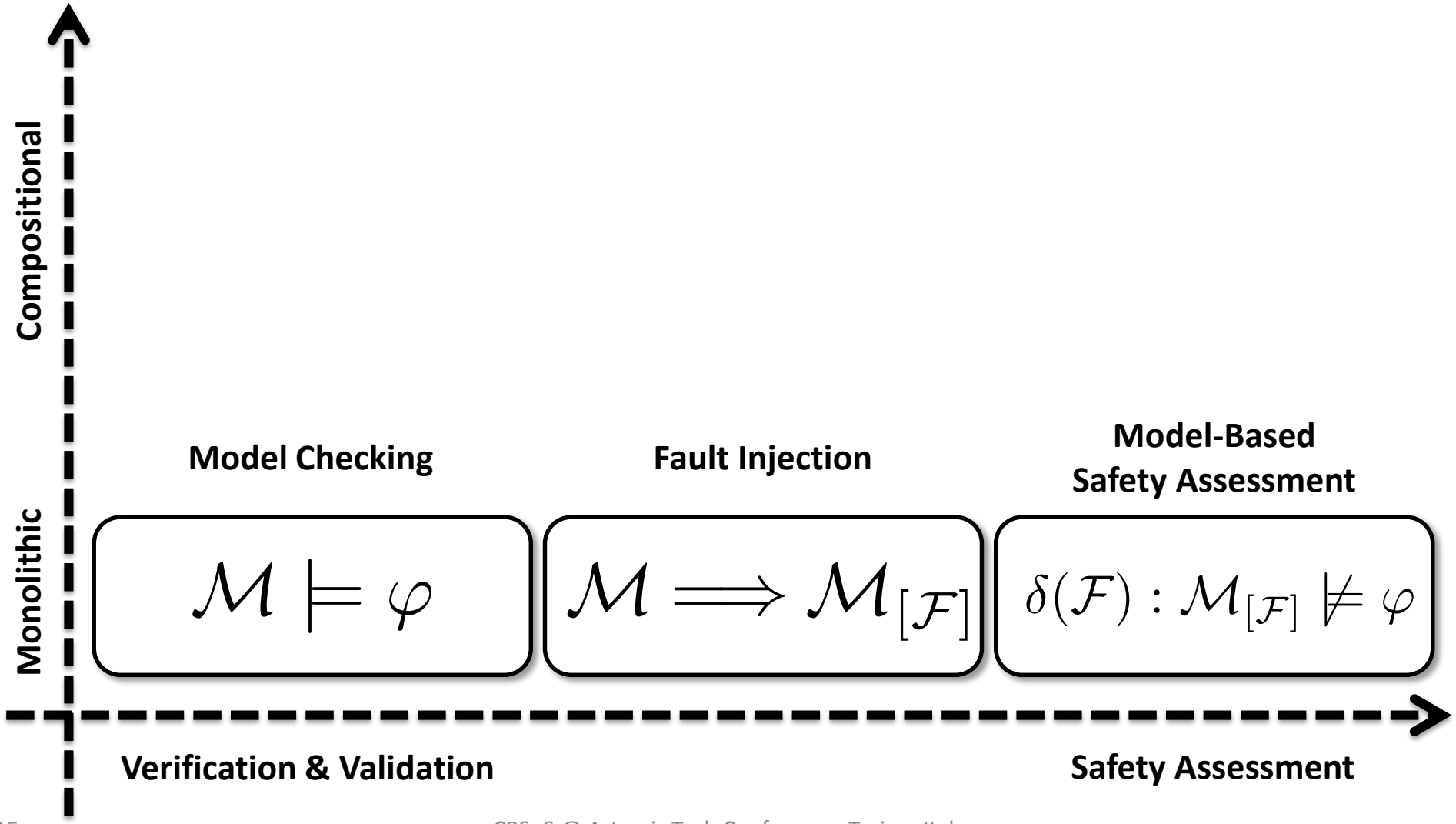
$$\mathcal{M} \Longrightarrow \mathcal{M}_{[\mathcal{F}]}$$

Model-Based Safety Assessment

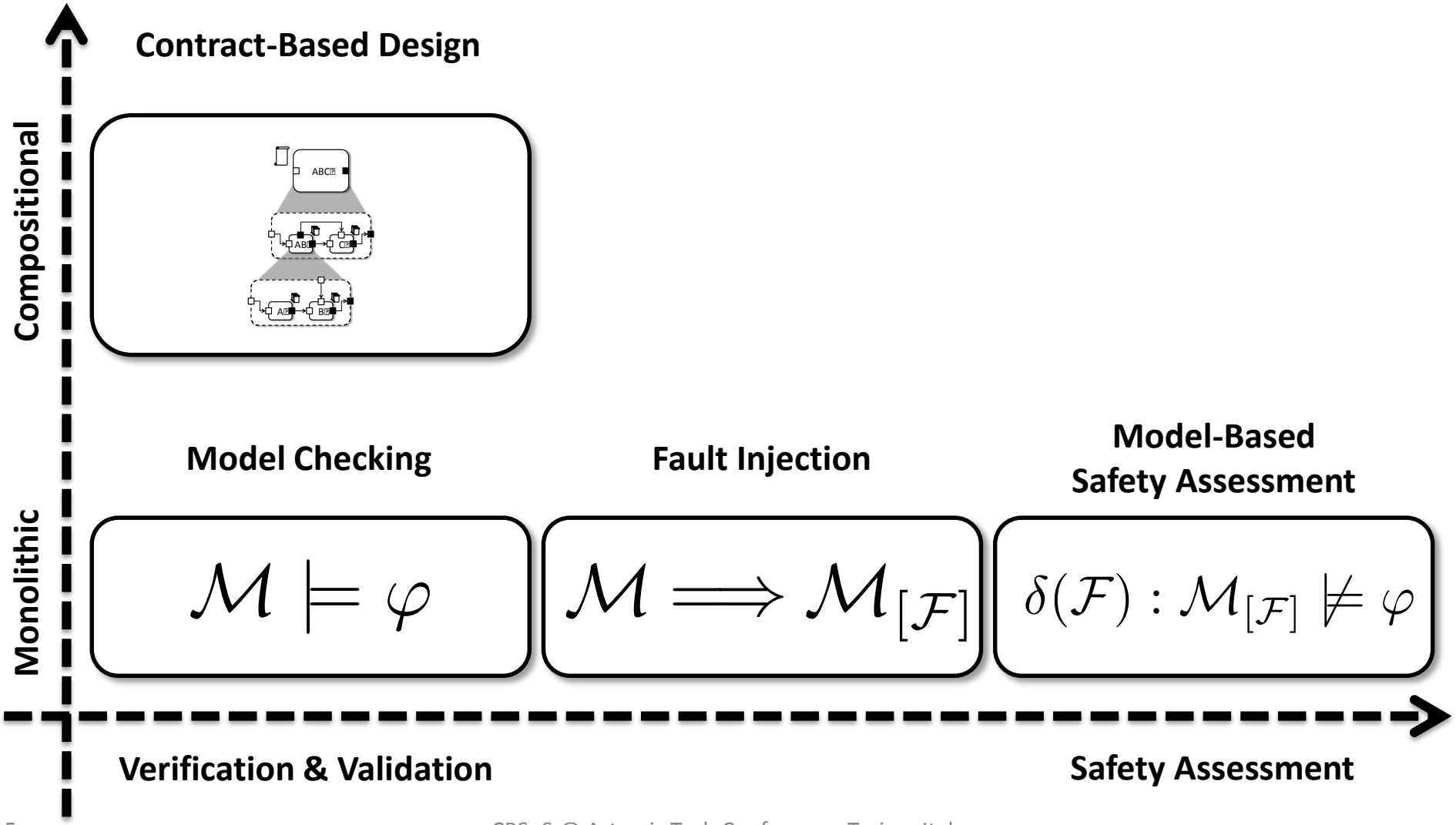
$$\delta(\mathcal{F}) : \mathcal{M}_{[\mathcal{F}]} \not\models \varphi$$

Verification & Validation → **Safety Assessment**

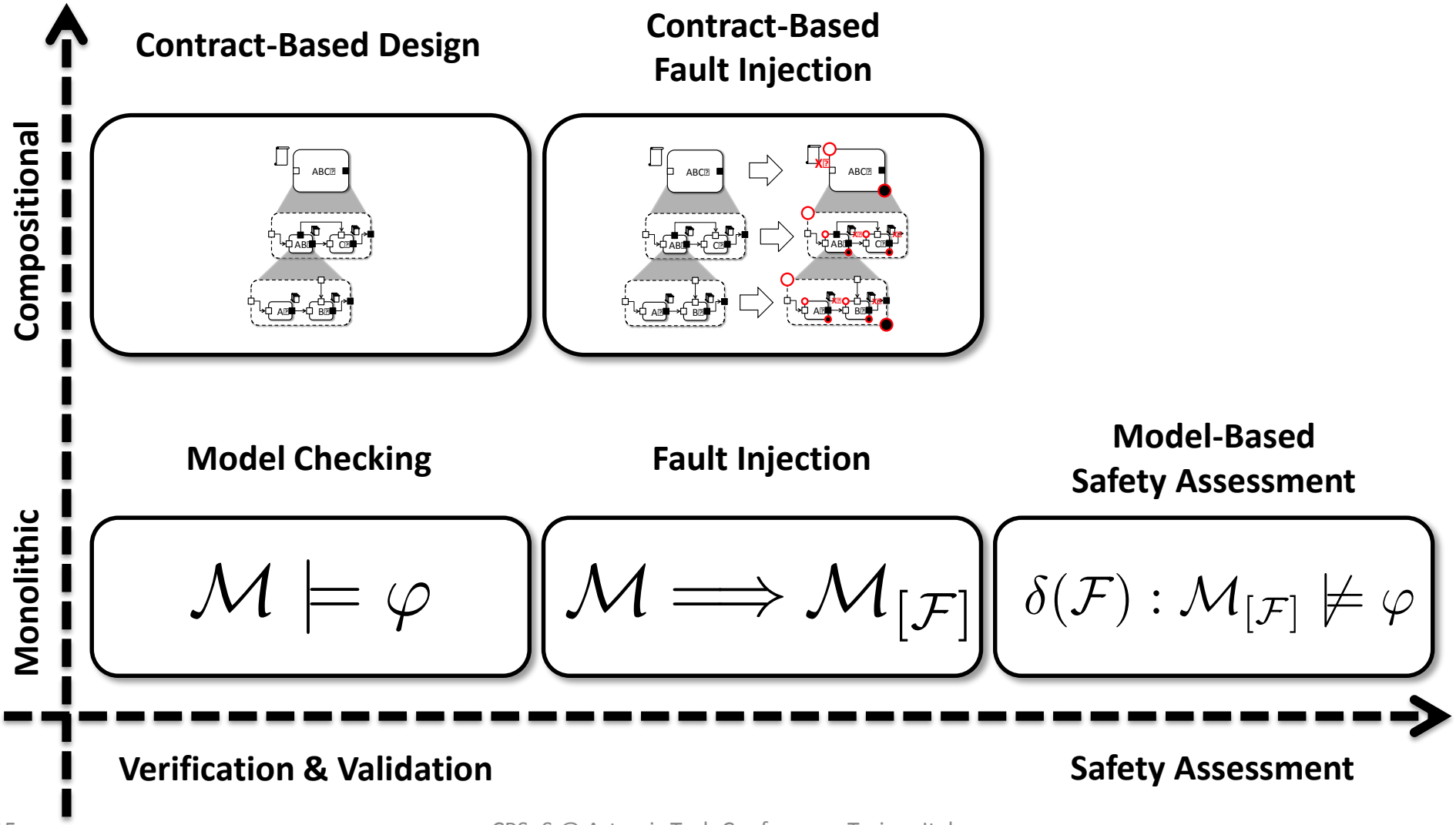
Formal Verification, Validation, and Safety Assessment



Formal Verification, Validation, and Safety Assessment



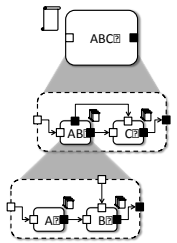
Formal Verification, Validation, and Safety Assessment



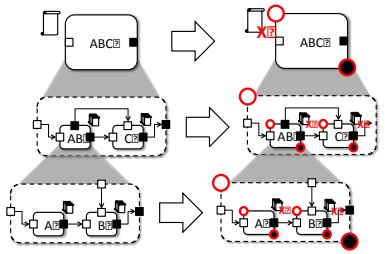
Formal Verification, Validation, and Safety Assessment

Compositional

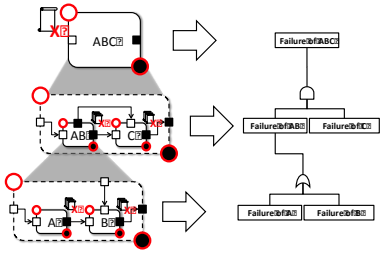
Contract-Based Design



Contract-Based Fault Injection



Contract-Based Safety Assessment



Monolithic

Model Checking

$$\mathcal{M} \models \varphi$$

Fault Injection

$$\mathcal{M} \Longrightarrow \mathcal{M}_{[\mathcal{F}]}$$

Model-Based Safety Assessment

$$\delta(\mathcal{F}) : \mathcal{M}_{[\mathcal{F}]} \not\models \varphi$$

Verification & Validation → **Safety Assessment**

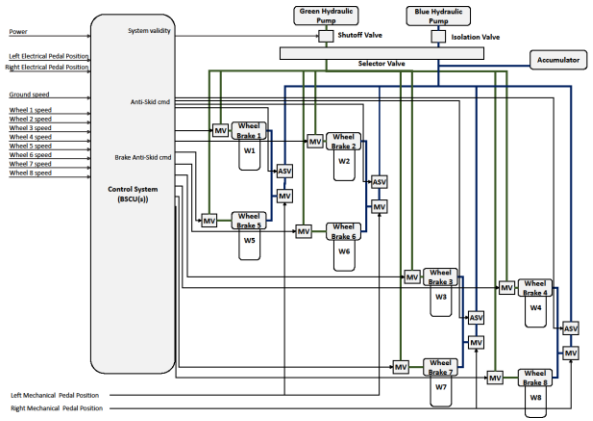
Tool chain

- Infinite-state transition systems
  - The **OCRA** tool for contract-based design
    - http://ocra.fbk.eu/
  - The **nuXmv** model checker
    - http://nuxmv.fbk.eu/
  - The **xSAP** platform for safety analysis
    - http://nuxmv.fbk.eu/

- Hybrid systems
  - **HyCOMP** as a model checker
    - http://hycomp.fbk.eu/

A Wheel Brake System

- Control brake for aircraft wheels
- Redundancy
  - Multiple BCSU
  - Hydraulic plants
- Functions
  - Asymmetrical braking
  - Antiskid
    - Single wheel/coupled
    - depending on control mode



Applications

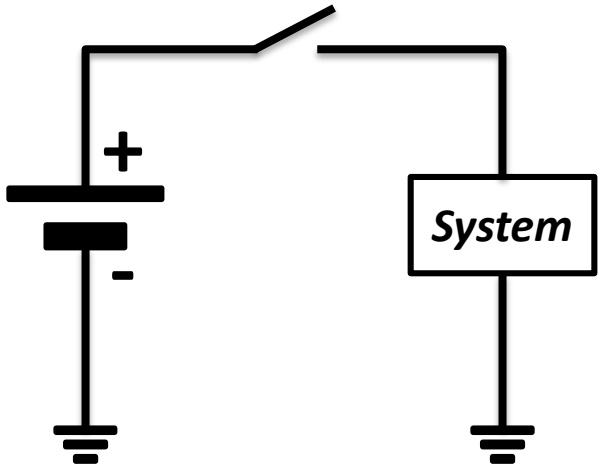
- Joint project with Boeing on MBSA
  - Formal Design and Safety Analysis of AIR6110 Wheel Brake System [CAV'15]
- Adopted in NASA project on analysis of NextGen
  - Comparing Different Functional Allocations in Automated Air Traffic Control Design [FMCAD'15]
- The COMPASS tool chain
  - AADL modeling language
  - Several projects funded by the European Space Agency
  - Specific design technique for FDIR

Automated Formal Analysis of Architectures for Reliability

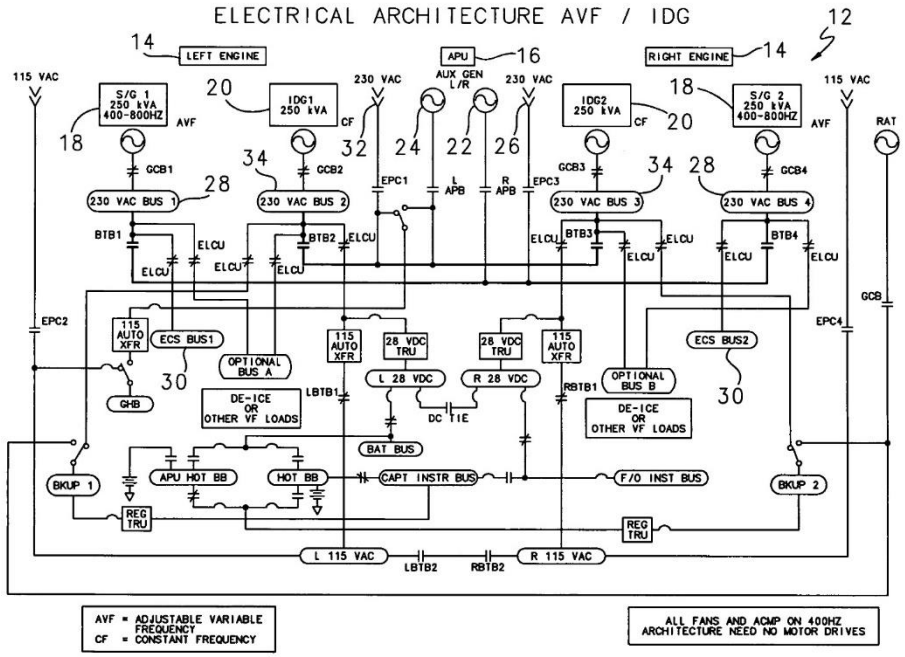
Architectures for Reliability

Power System Example

in a perfect world



…in the real world



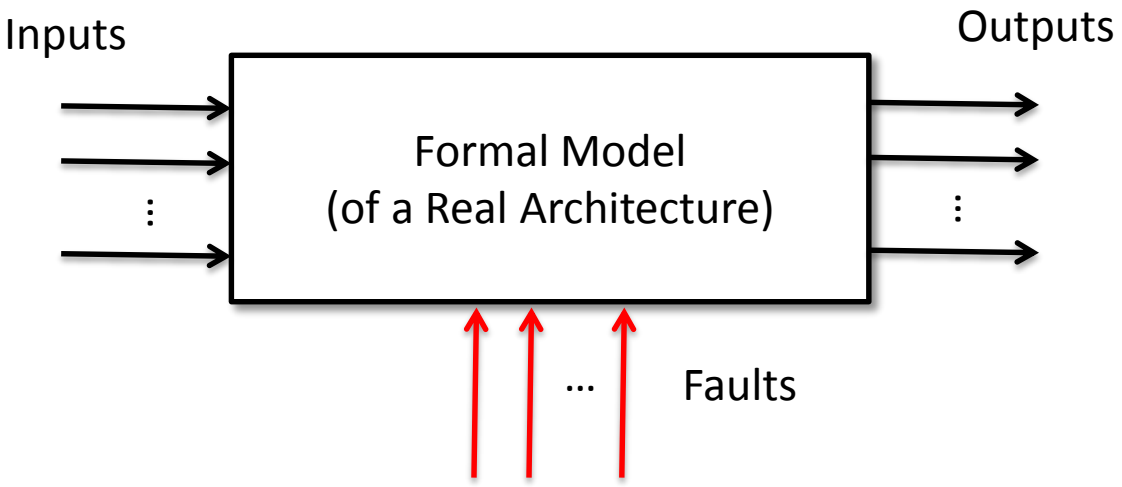
Automated Analysis of Architectures for Reliability

- In 1996 (PFC 777 Paper):

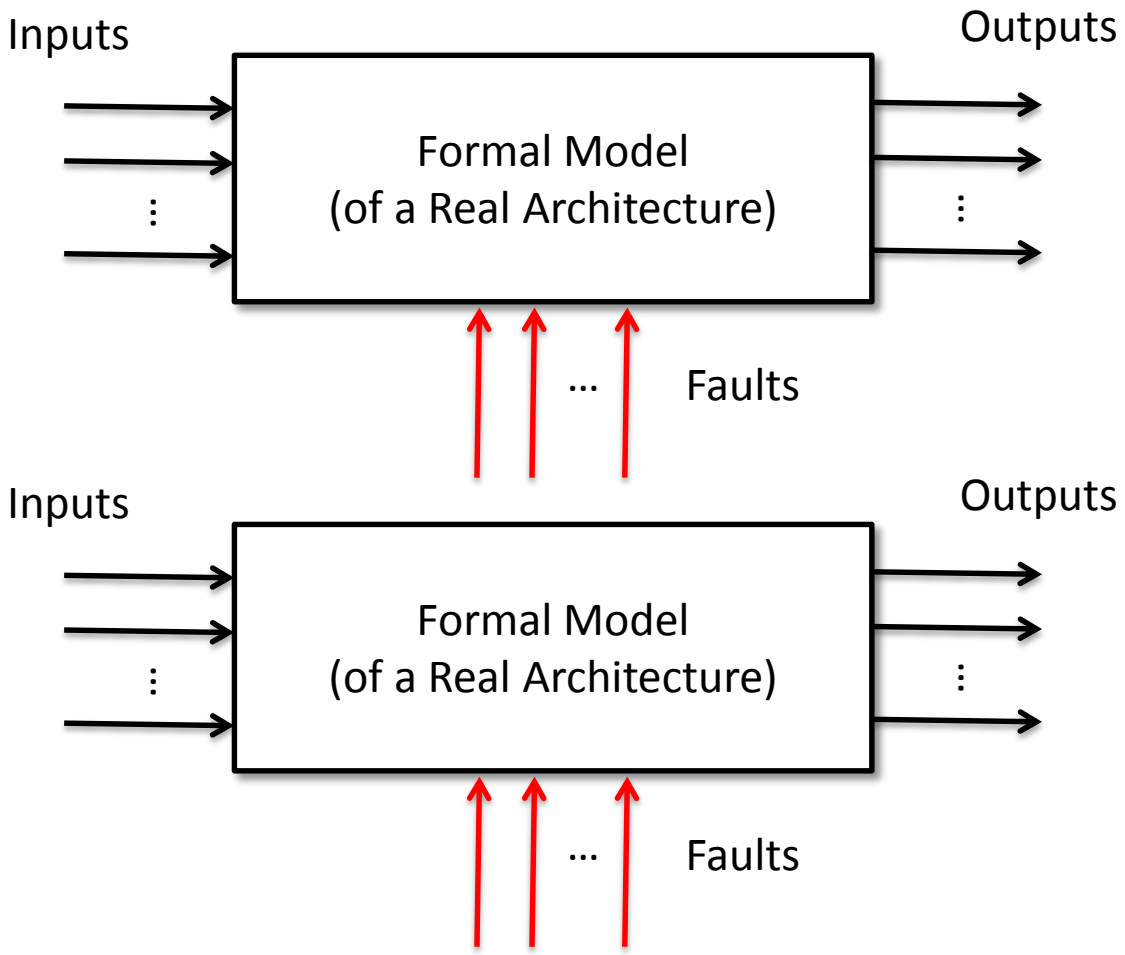
> Rigorous mathematical proof of algorithms to cope with the Byzantine generals problem is not possible for any triple redundant system [5].

- Current techniques:
  - M. Hamamatsu, T. Tsuchiya, and T. Kikuno, “On the reliability of cascaded TMR systems” PRDC, 2010
  - T. Lanfang, T. Qingping, and L. Jianli, “Specification and verification of the triple-modular redundancy fault tolerant system using CSP” DEPEND 2011
  - M. Bozzano, A. Cimatti, and C. Mattarei “Automated Analysis of Reliability Architectures”, ICECCS 2013
  - M. Bozzano, A. Cimatti, and C. Mattarei “Efficient Analysis of Reliability Architectures via Predicate Abstraction”, HVC 2013

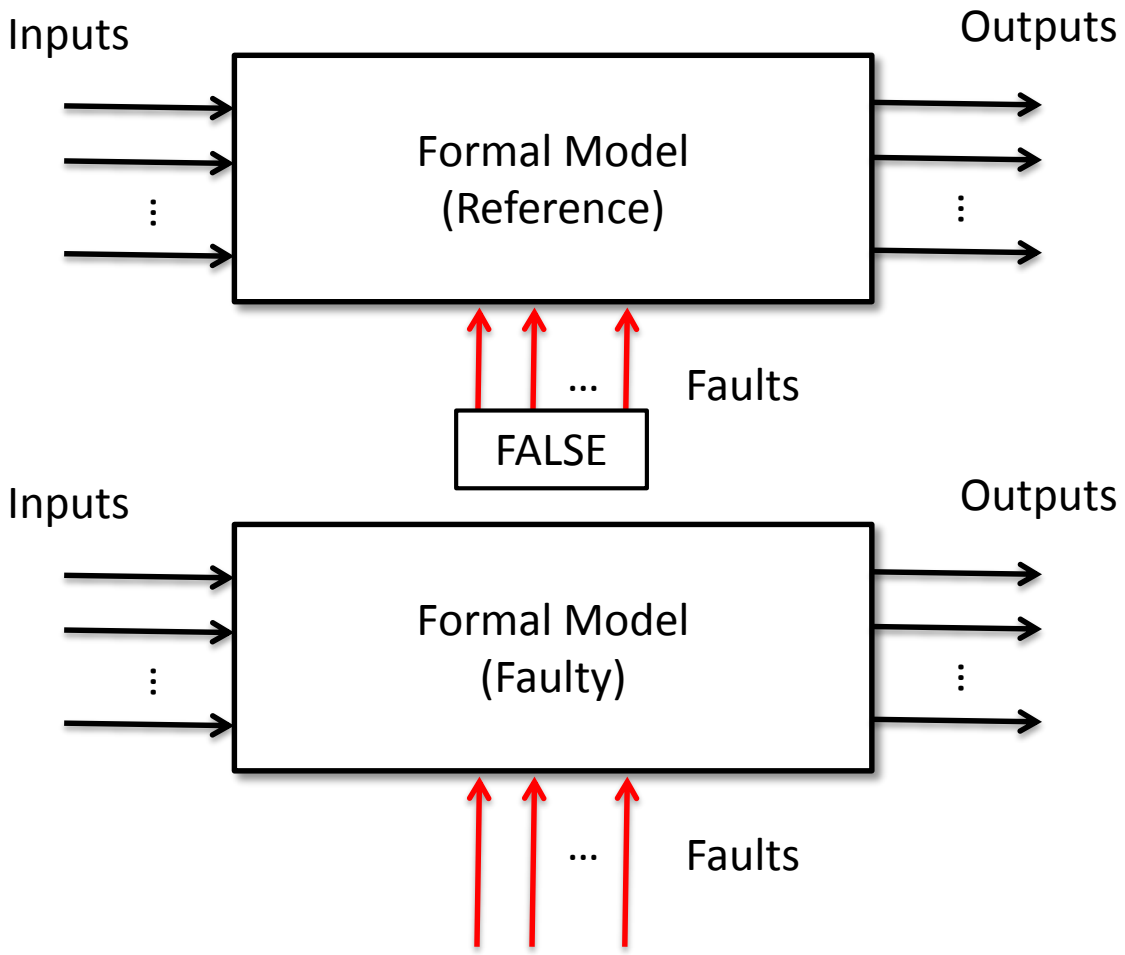
Automated Analysis of Architectures for Reliability



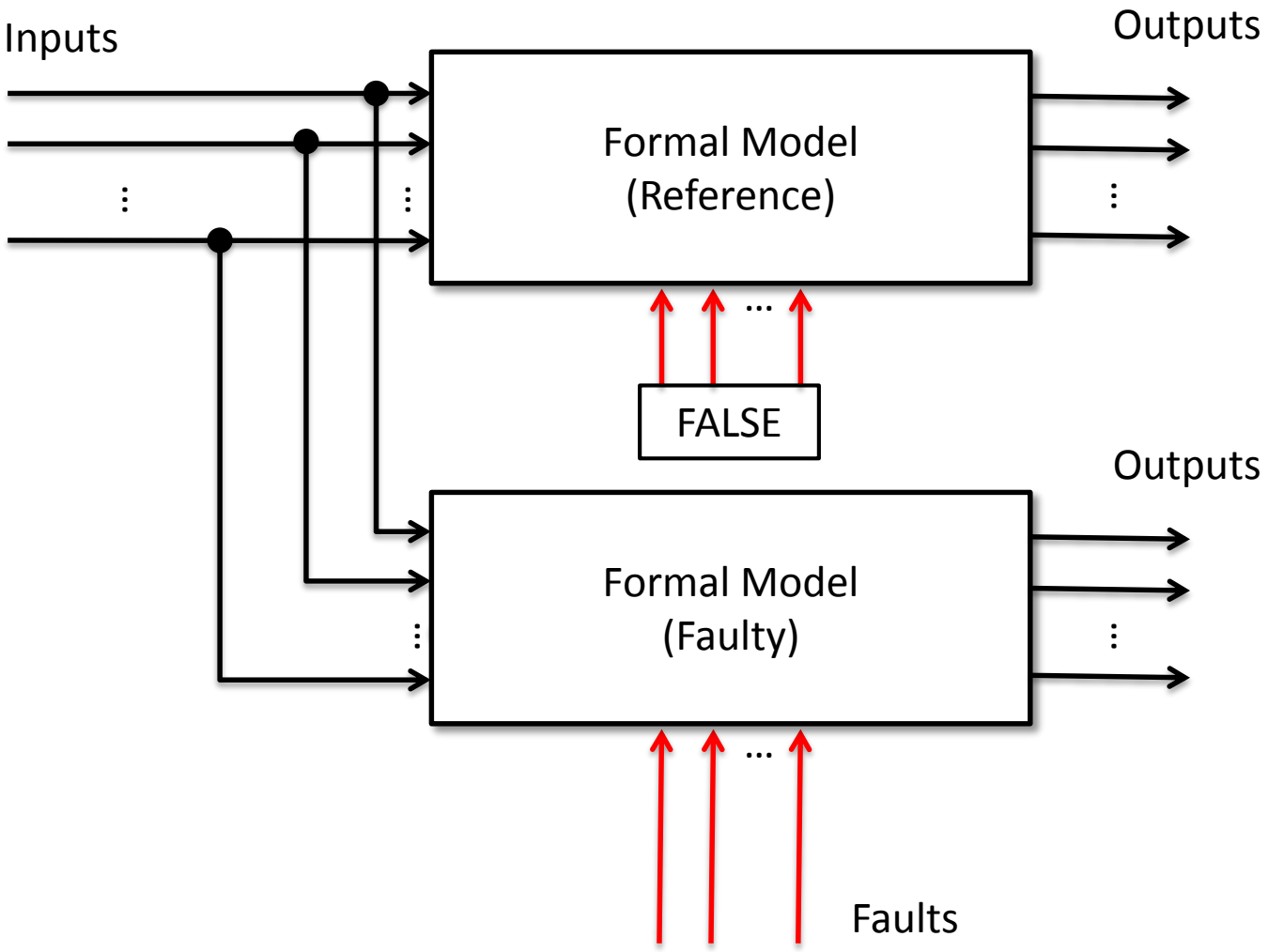
Automated Analysis of Architectures for Reliability



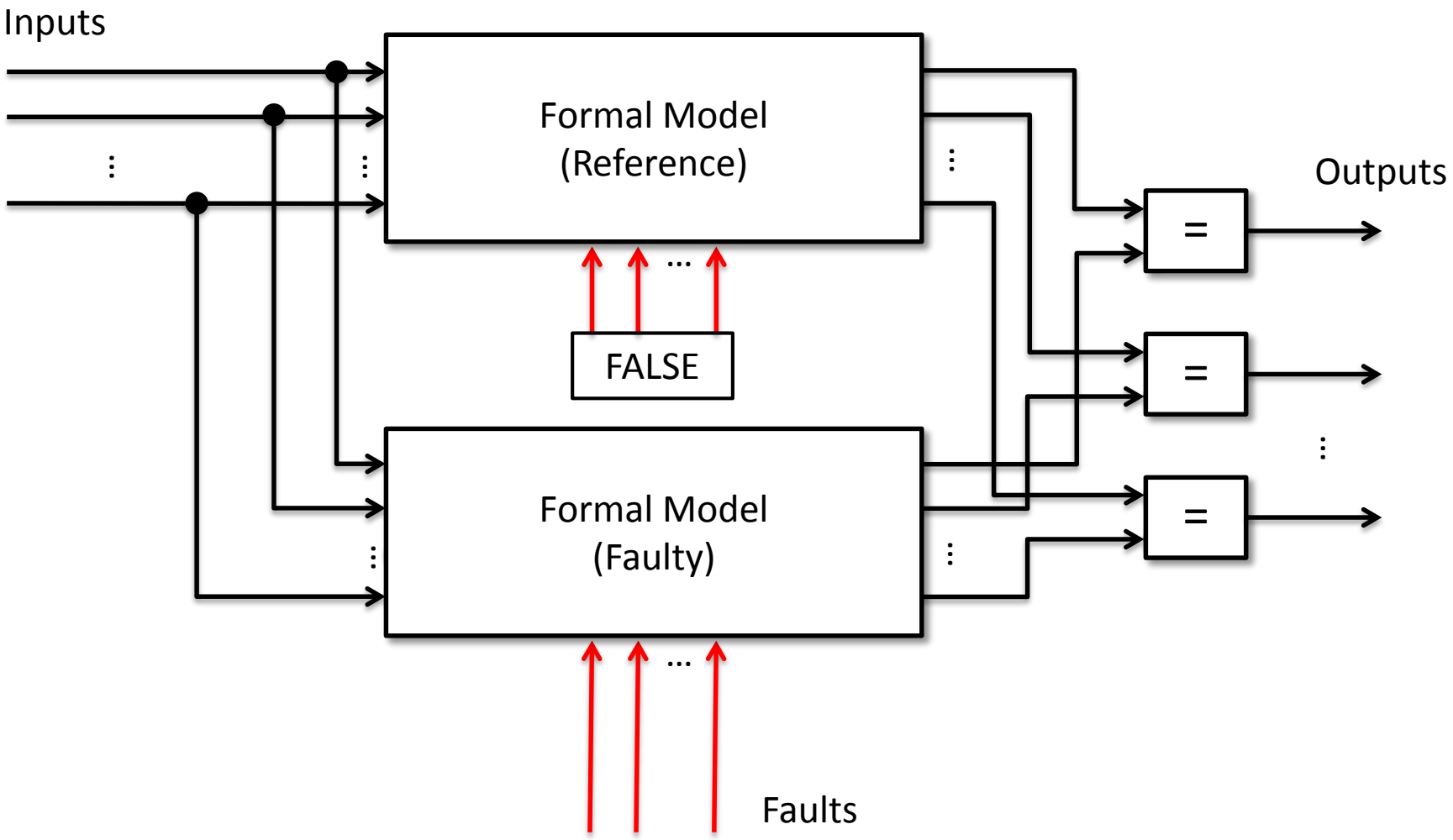
Automated Analysis of Architectures for Reliability



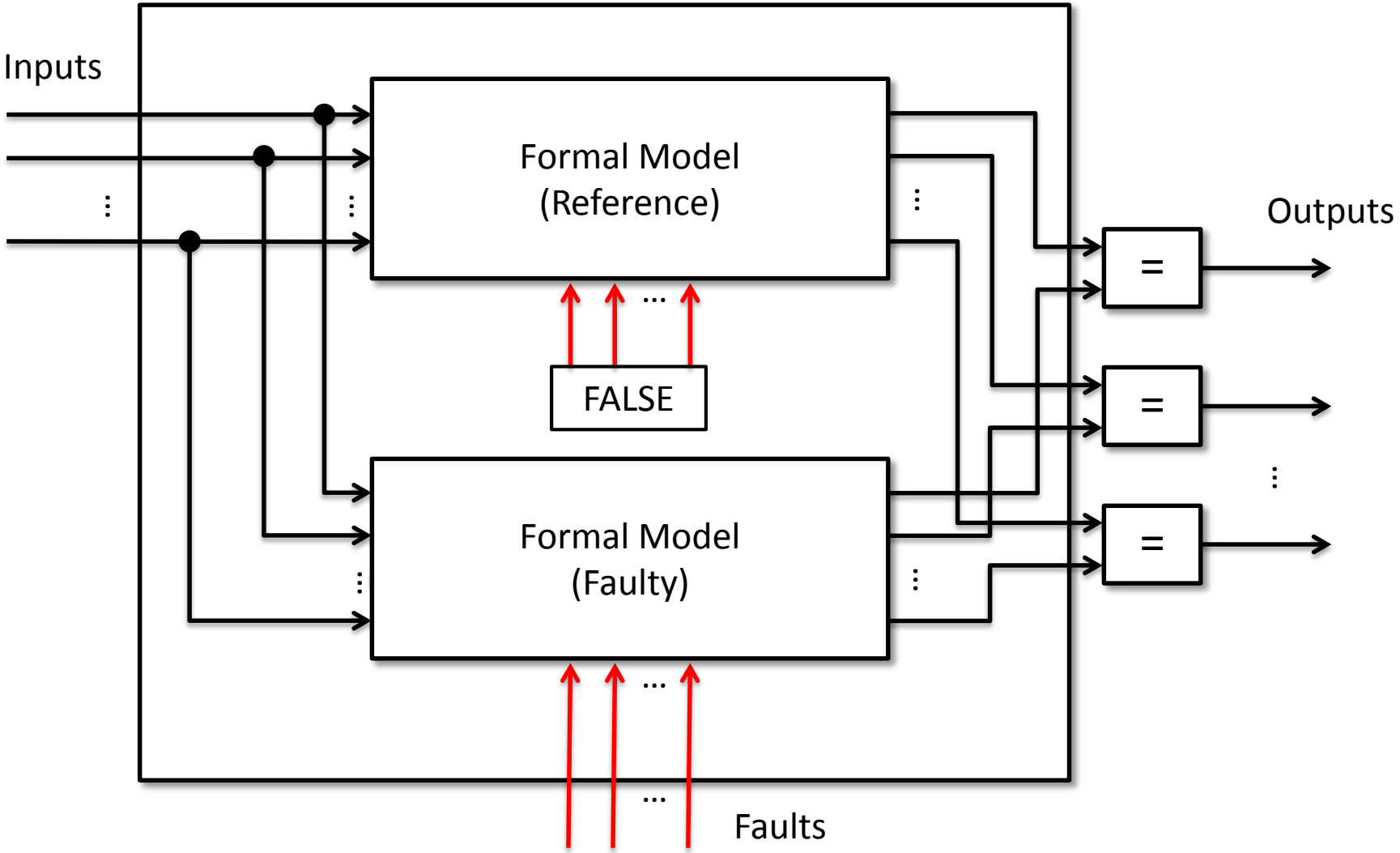
Automated Analysis of Architectures for Reliability



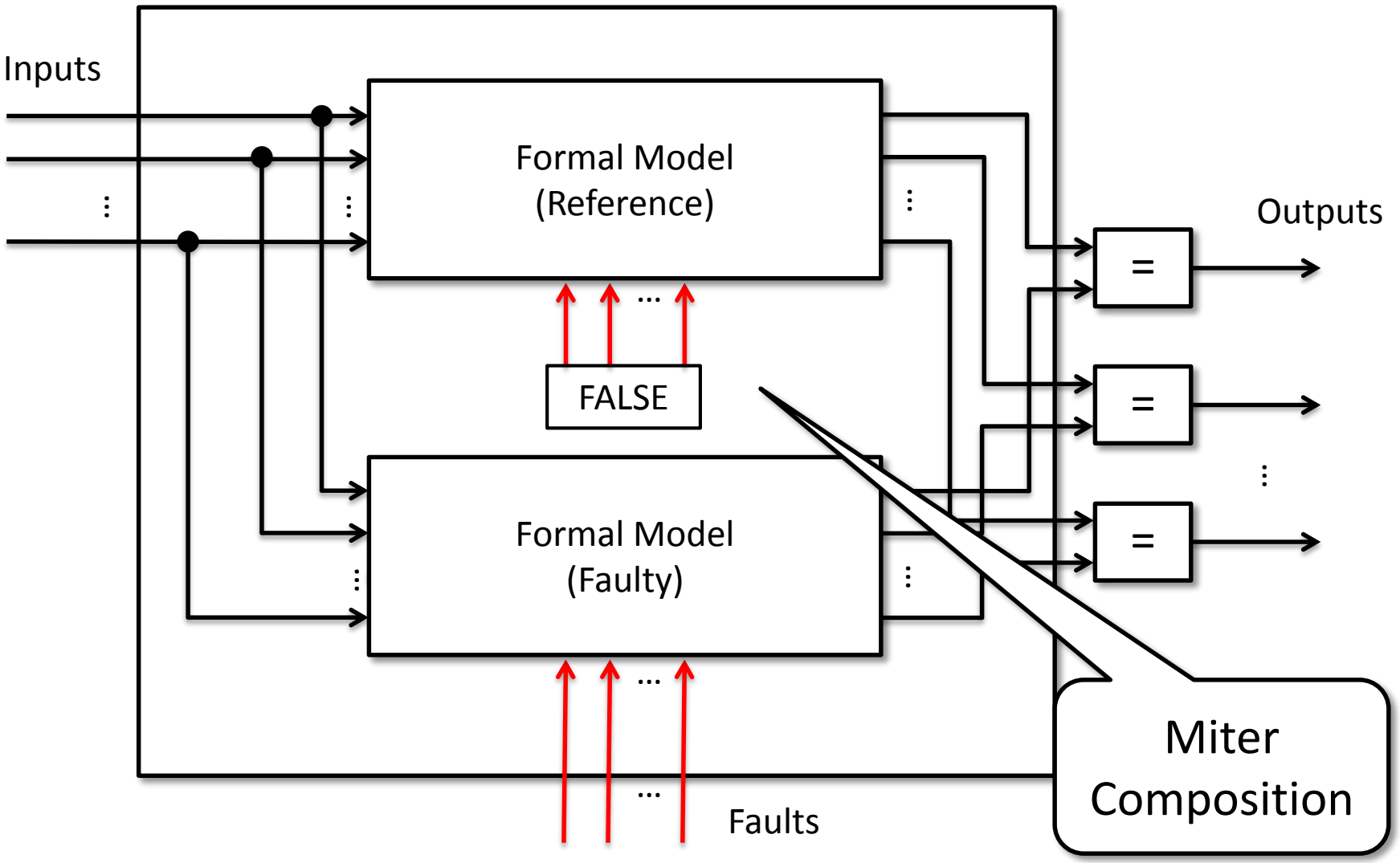
Automated Analysis of Architectures for Reliability



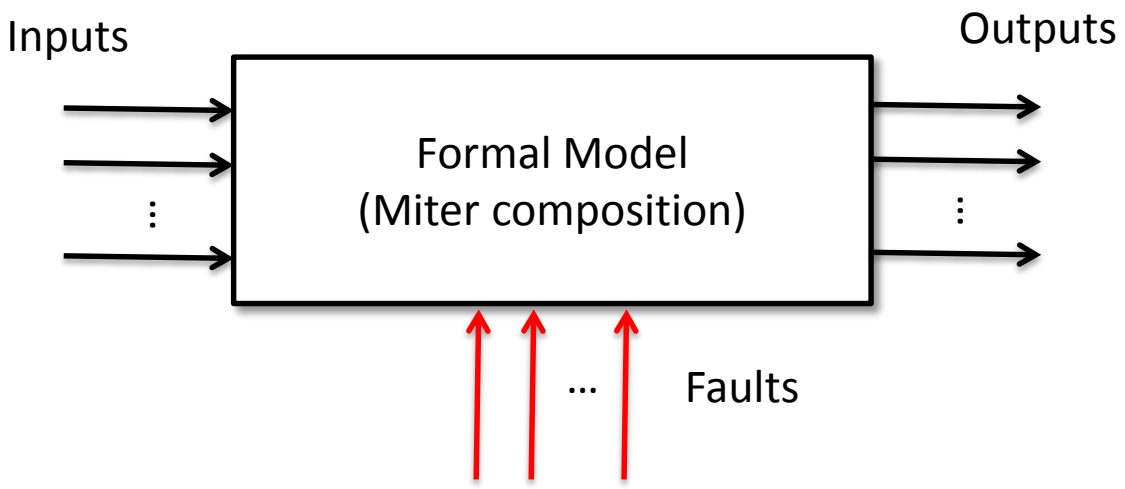
Automated Analysis of Architectures for Reliability



Automated Analysis of Architectures for Reliability



Miter composition: Formal Safety Assessment

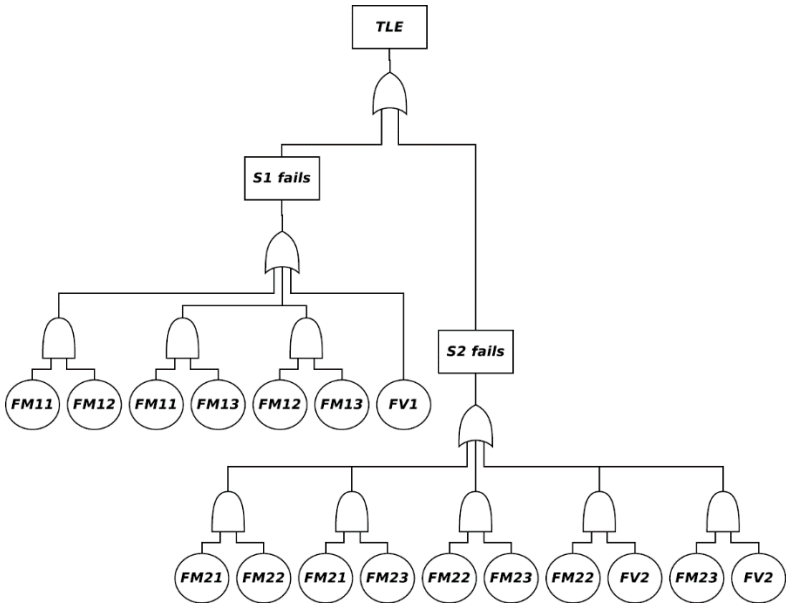


?? All assignments to Faults such that

- At least one output = FALSE
- All but one output = FALSE
- At least two outputs = TRUE
- …

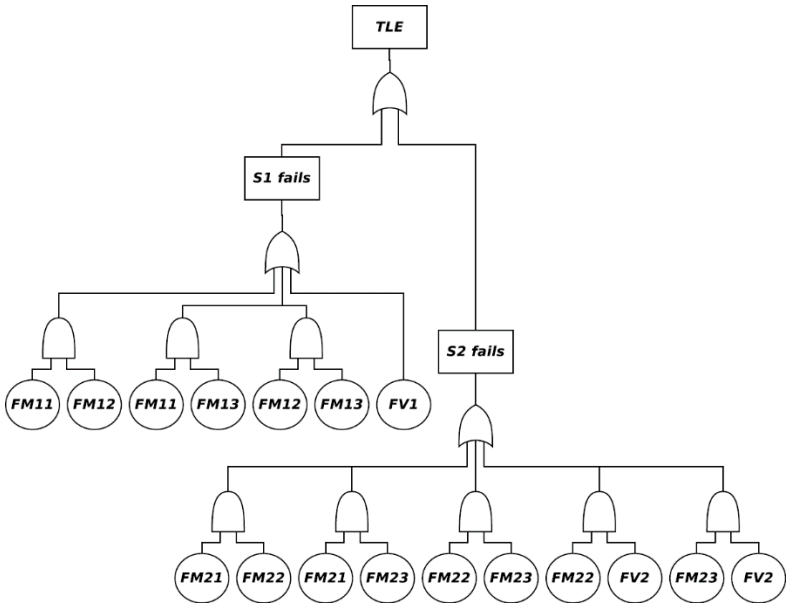
Formal Safety Assessment: Possible Artifacts

Fault Tree



Formal Safety Assessment: Possible Artifacts

Fault Tree



Reliability Function

$$F_{sys} : \mathbb{R}^{[0,1]} \times \ldots \times \mathbb{R}^{[0,1]} \mapsto \mathbb{R}^{[0,1]}$$

$$F_{sys}(F_v, F_{m1}, F_{m2}, F_{m3}) =$$

$$F_v +$$

$$+(1 - F_v) * F_{m1} * F_{m2} +$$

$$+(1 - F_v) * F_{m1} * (1 - F_{m2}) * F_{m3} +$$

$$+(1 - F_v) * (1 - F_{m1}) * F_{m2} * F_{m3} +$$

Triple-Triple Redundant 777 PFC

Y.C. (Bob) Yeh, AAC 1996

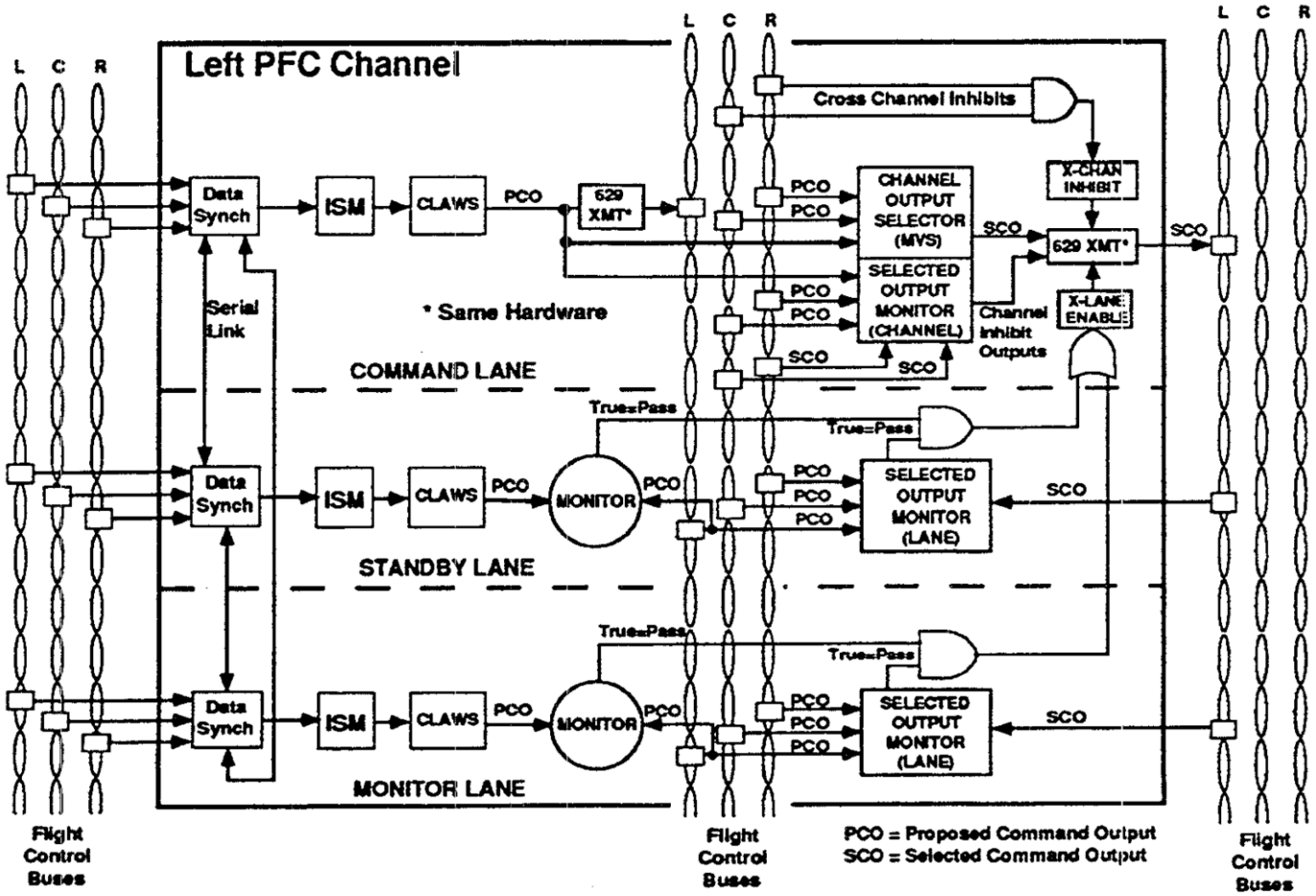


Figure 10 PFC Lane Redundancy Management (Output Signal Monitoring)

Conclusions and Perspective

- Conclusions
  - New generation of verification techniques
  - Tools integrated into comprehensive process
  - Production of interesting artifacts from unique model

- Perspectives
  - Support to design space exploration
    - Design space as parameterized system
  - Comparison based on safety artifacts
    - More fine grained
  - Analysis of reliability architectures