Challenges, Opinions, Perspective and Proposals

H. Kopetz

June 2003

What have we Learnt?

- The rate of transient failures of SoCs is on the increase due to the following
  - Single event upsets
  - Signal integrity problems
  - Variations due to manufacturing
  - Degradation problems after shipment
- The single bit-flip model is out
- In ultradependable applications, we need fault-tolerance at the architecture level to achieve the desired level of dependability
- The initial cost of a SoC is so high, that only applications that require millions of chips can afford their own SoC
- Radiation hardened chips can be replaced by fault-tolerant architectures based on commodity SoCs

Some Consequences for Safety-Critical Systems

- We cannot assume that in ultradependable applications a single SoC can contain more than a single Fault-Containment Region (FCR)--not independent enough.
- The physical structure of an ultradependable distributed application is determined by the requirement to support enough independent FCRs such that the required dependability can be achieved by fault-tolerance at the architecture level.
- Since it must be assumed that an SoC can fail in an arbitrary failure mode, the encapsulation and fault isolation mechanisms *within an SoC* are not in the same criticality class as the fault isolation mechanisms *between SoCs*.

Independence of Fault Containment Regions

The diversity of Fault Containment Regions (FCRs) that are located on a single chip is compromised by:

- Same Physical Space (Physical Proximity Failures)
- Same Mask (Mask Alignment Issues)
- Same Bulk Material
- Same Wafer Production Process
- Same Power Supply
- Same Earthing
- Same Timing Source
-

Although some of these dependencies can be eliminated, others cannot.

Independence of FCRs

There are two basic mechanisms that compromise the independence of FCRs

- Missing fault isolation
- Error propagation

The independence of failures of different FCRs is the most critical issue in the design of an ultra-dependable system:

- Is it justified to assume that a single silicon die contains two independent FCRs?--**NO**
- Can we assume that the failure modes of a single silicon die are well-behaved (e.g., fail-silent) to the required level of probability?-- **NO**
- How can we make sure that FCR failures are not correlated, even at a very low level of correlation (e.g., 1 in 1000)?

Architecture based Fault Isolation

TTP/C-C1-based hardware prototype with XILINX 600k FPGA (tested by IST project FIT):

Heavy Ion Experiments (at Chalmers):

Bus topology: 37036 faults--78 error propagations (0.21 %)

Star topology: 26600 faults-- 0 error propagation

Software Implemented Fault Injection (Vienna):

Bus topology: 562122 faults--14 error propagations (0.02 %)

Star topology: 541744 faults-- 0 error propagation

Published at DSN, San Francisco, June 2003

Formal Verification using Model Checking (SAL, UPPAAL2k) and Theorem Proofing (PVS) is ongoing in the NEXT TTA Project.

Open Questions:

- Is it possible to build ultradependable control systems out of commodity SoCs?„
- Can we use the architecture-based fault tolerance to cover the transient failures of commodity SoCs as well?
- What are the constraints for integrating safety-critical and non safety critical functions in the same SoC in order to reduce the overall cost?
- What is the best way to deal with state-corruption that is caused by transients?

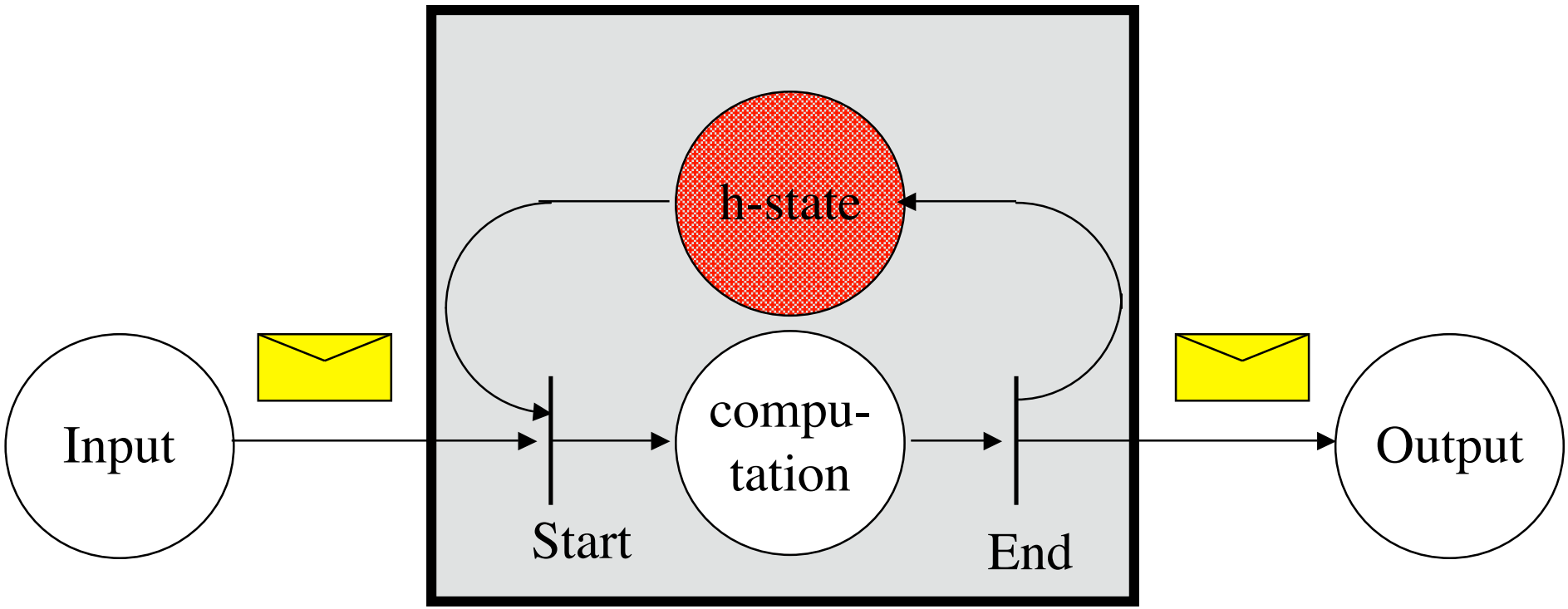
What is State?

"The state enables the determination of a future output solely on the basis of the future input and the state the system is in. In other word, the state enables a "decoupling" of the past from the present and future. The state embodies all past history of a system. Knowing the state "supplants" knowledge of the past. Apparently, for this role to be meaningful, the notion of past and future must be relevant for the system considered.,, (Taken from Mesarovic, Abstract System Theory, p.45)

A system-wide consistent notion of a discrete time is a prerequisite for a consistent notion of state, since the notion of *state* is introduced in order to separate the *past* from the *future*.

Fault-masking by voting (TMR) requires a consistent notion of state in distributed Fault Containment Regions (FCRs) and thus a consistent notion of gobal time.

The Concept of Critical State



Critical-State-Aware System Design

- The *critical state of an SoC* is the part of the state that is essential for the future safe behavior of the SoC.
- After a transient SoC failure, a relevant version of the *critical state* must be reloaded into the SoC as fast as possible.
- The part of the *critical state* that cannot be recovered from the sensors in the environment must be *stored* in distinct fault containment regions.
- The recovery problem is reduced, if the *stored critical state* is small in size.
- We must identify and reduce the size of the *stored critical state* of a distributed system by an application specific analysis of the *stored critical state requirements.*

Conclusion

- The upcoming SoCs pose a number of challenging research issues, driven by concerns of dependability and economics.
- How can we build ultradependable systems out of commodity SoCs?
- We need a new approach to system design at the application level: *critical-state-aware* system design.