Fault Isolation and Error Containment in the TT-SoC

H. Kopetz
TU Wien

July 2007

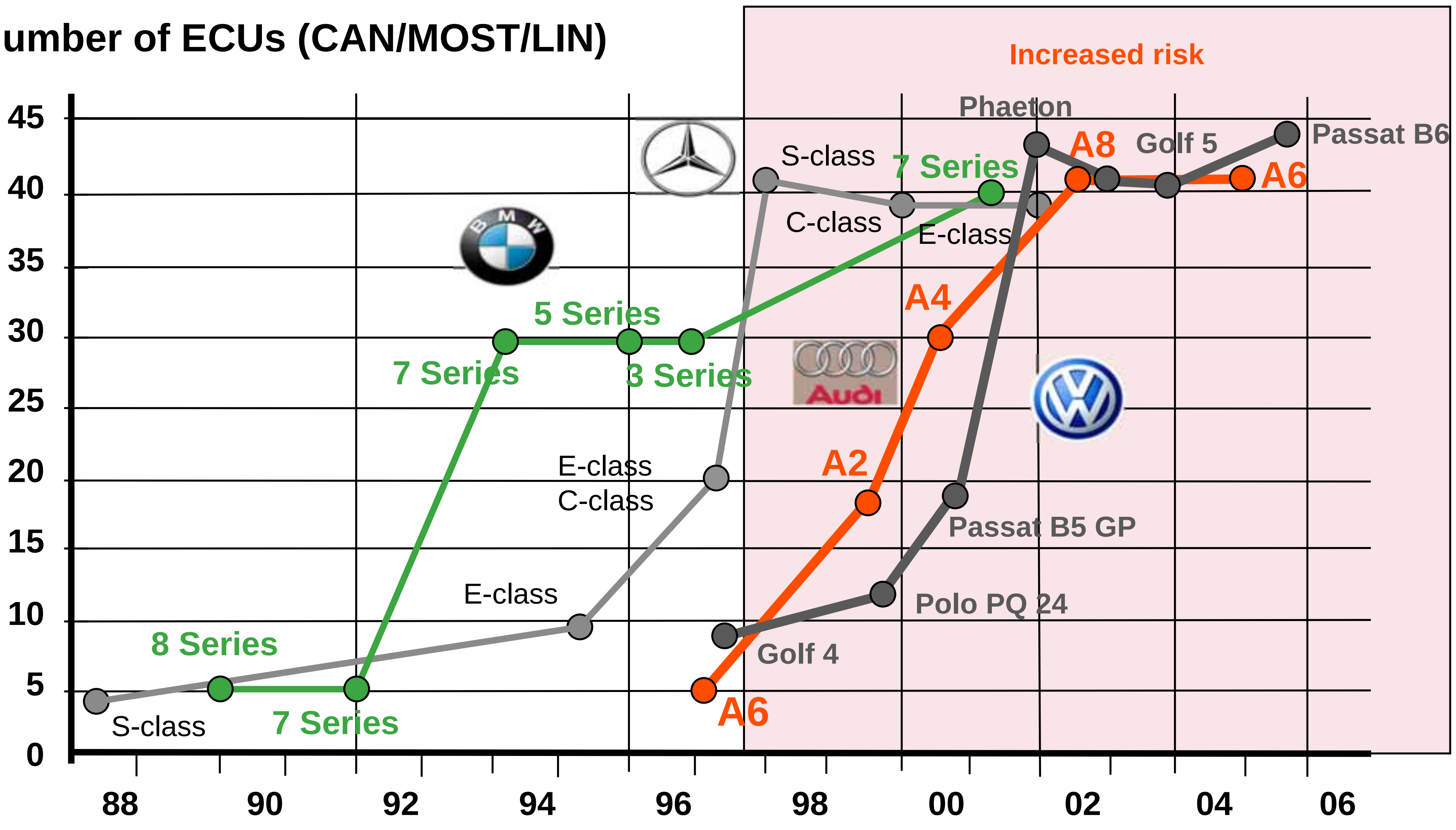
This is joint work with C. El.Salloum, B.Huber and R.Obermaisser

Outline

- Introduction
- The Concept of a Distributed Application Subsystem (DAS)
- The new time-triggered MPSoC
- Fault Isolation and Error Containment
- Conclusion

We are Getting Overwhelmed by Boxes and Wires

Number of ECUs (CAN/MOST/LIN)



Source: Prof. Dr. Jürgen Leohold, TU Vienna Summer School 2004 on *Architectural Paradigms for Dependable Embedded Systems*

Key Obstacles on the Way to Integration

In our opinion the key obstacles that hinders the move to an integrated architecture are the ***limited encapsulation***, the ***weak fault-isolation*** and the ***poor error containment*** capabilities of the existing hardware/software architectures that partition the services of a single CPU to a set of nearly independent distributed application subsystems.

Distributed Application Subsystem (DAS)--*JOB*

A large RT control system (e.g., the computer system onboard a car or an airplane) can be partitioned into a set of nearly autonomous subsystems, we call them Distributed Application Subsystems (DAS).

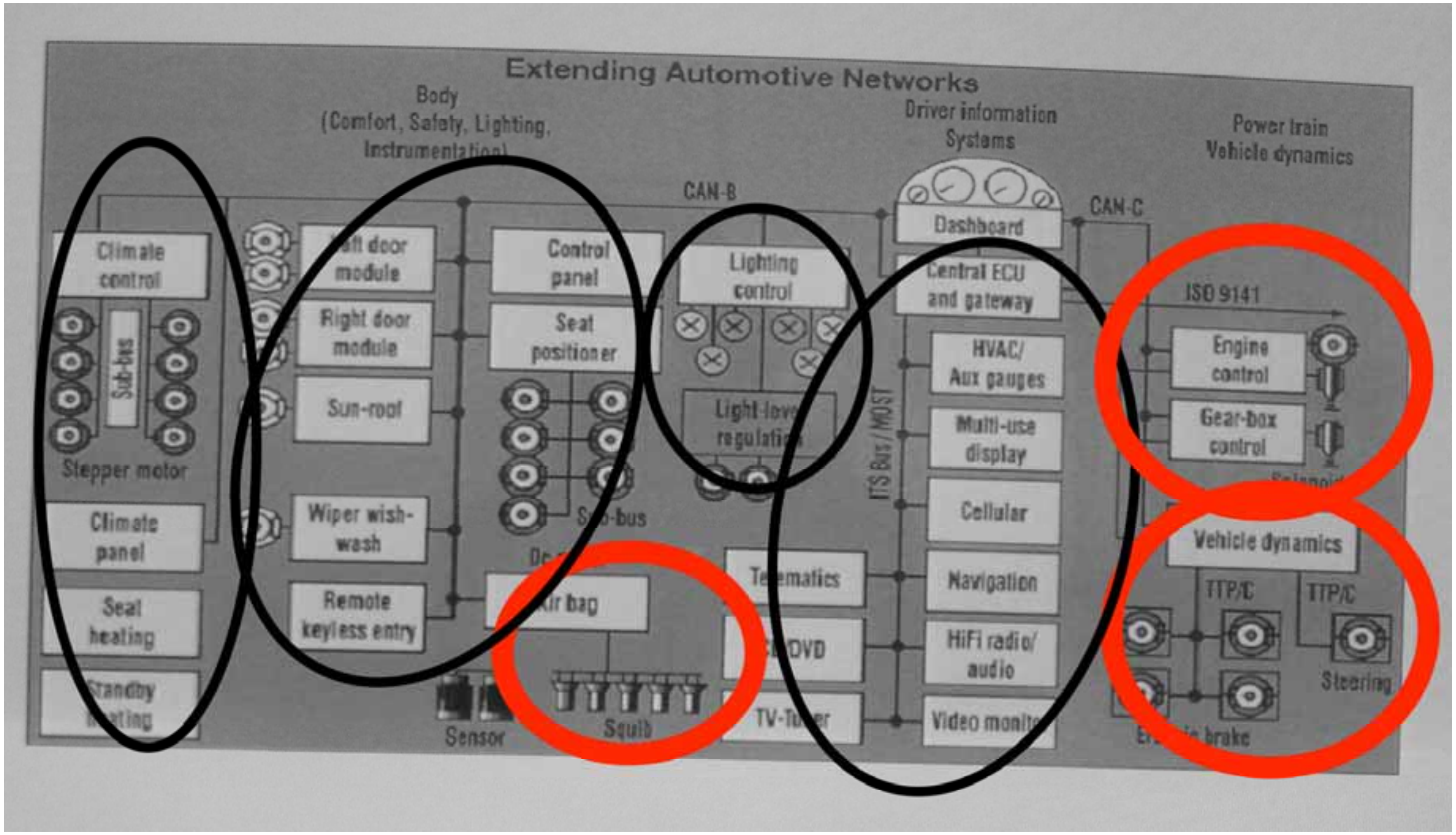
Examples of a DAS onboard a car are

- ♦ The body electronics DAS (doors, lights, etc.)
- ♦ The power train control DAS
- ♦ The multi-media DAS

A DAS consists of a number of ***JOBs*** that are executed at different nodes (today's ECUs) of the control system. (A *Job* is a well-defined Software Unit of distribution).

Large systems do not have a single top.

Automotive Example



DAS-Distributed Application Subsystem

Further Characteristics of *DASes*

Different DASes

- ♦ may be developed by *different organizational entitities* (e.g. by different companies)
- ♦ may be of *different criticality*, e.g. Vehicle Dynamics Control versus Multimedia System in an automobile
- ♦ may use *different platform services* e.g. different communication protocols different real-time operating systems
- ♦ may be maintained by different specialists

Most of the time, each DAS is developed in isolation.

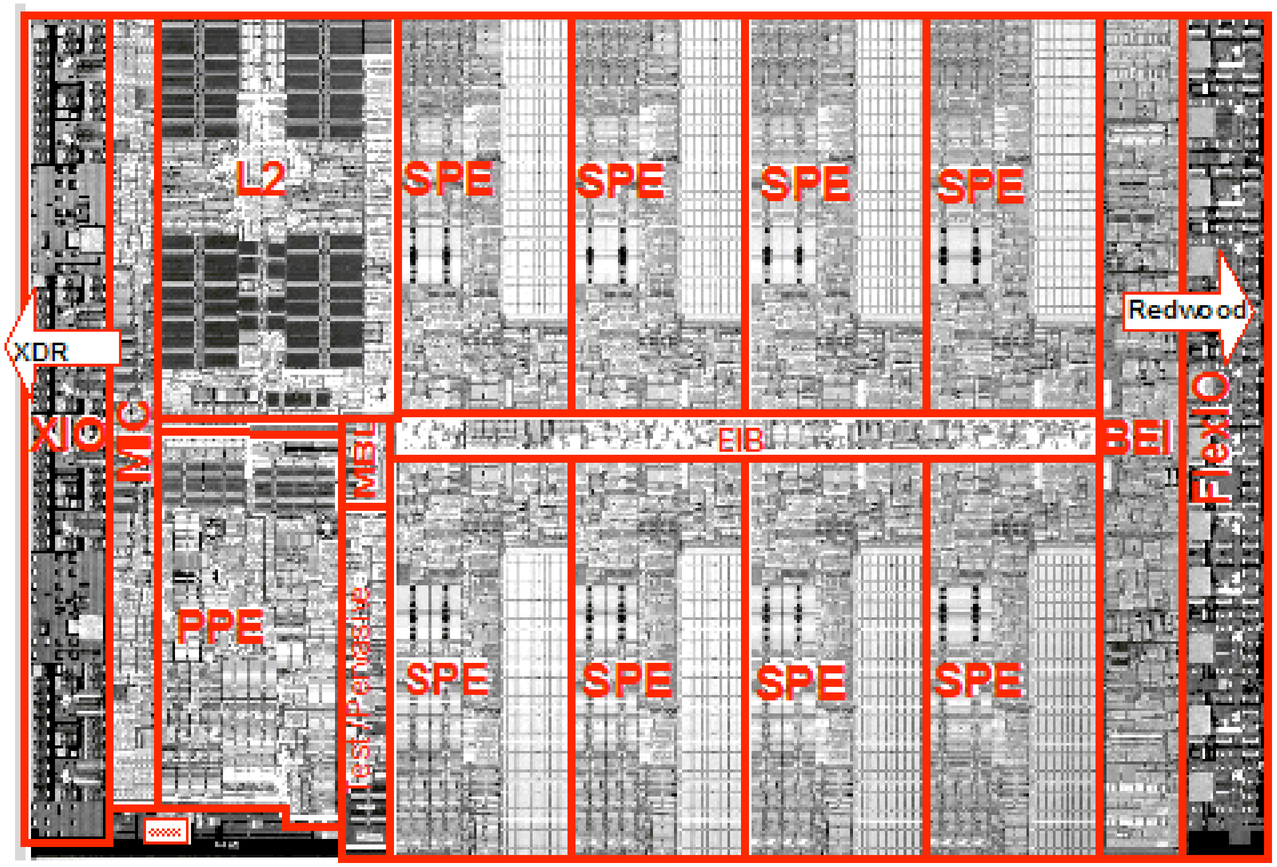
Federated vs. Integrated System

The ideal future avionics systems would combine the complexity management advantages of the federated approach, but would also realize the functional integration and hardware efficiency benefits of an integrated system.

Hammett R. *Flight Critical Electronics System Design*, IEEE AESS Systems Magazine, June 2003, p.32

Example: Cell Processor announced 2005

Joint development of IBM, Sony and Toshiba, 90 nm process 250 Mio Transistors, 221mm^2 , 4 Ghz, 250 GFLOPS



Proposed Solution: *Time triggered MPSoC*

A Multi-computer SoC with a time-triggered Network on Chip (TT-SoC) forms *the ideal execution environment for the integration of multiple DAS-Jobs into a single ECU*:

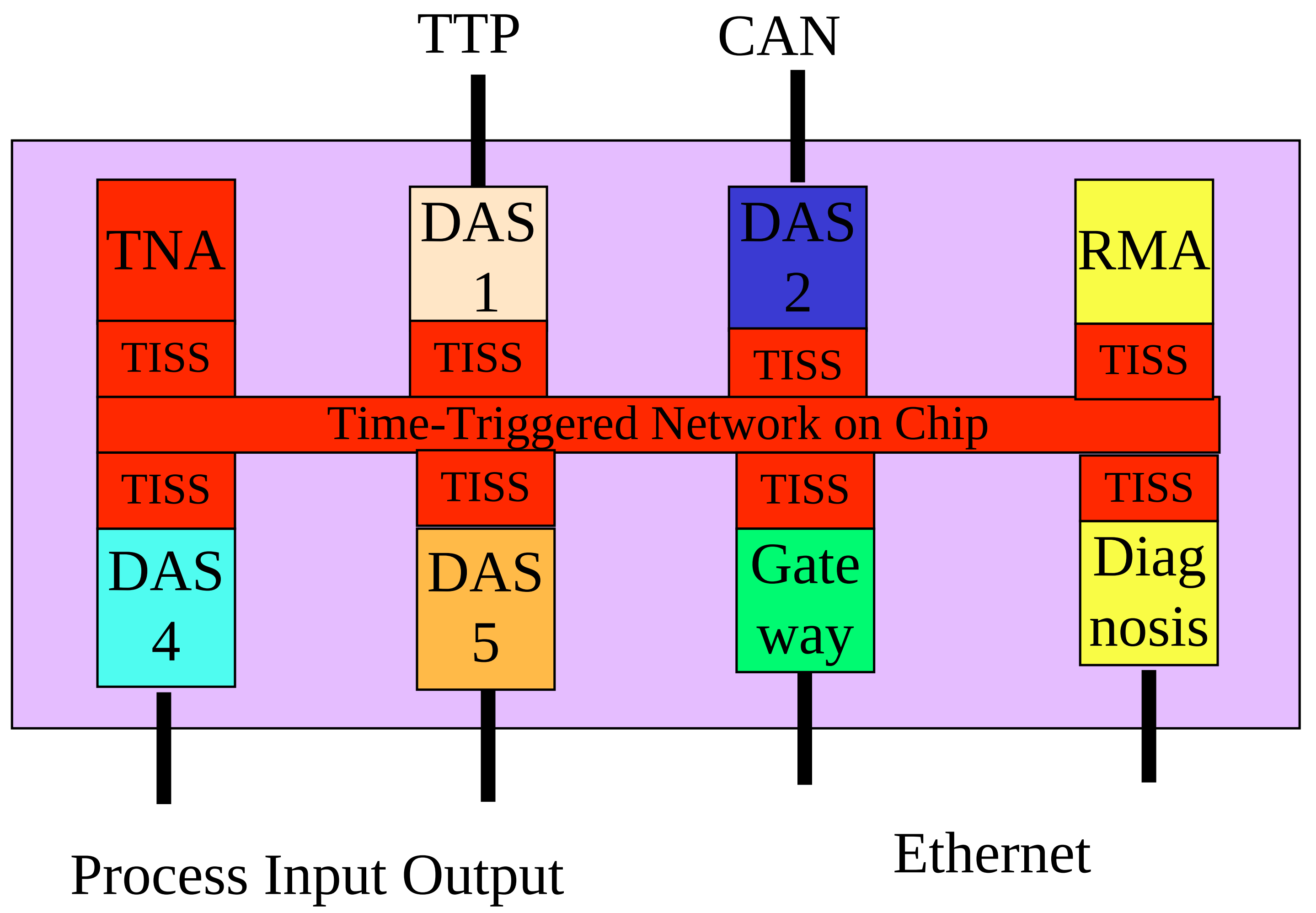
- ♦ Each core can host a *job* of a DAS in an execution environment that is encapsulated and protected by hardware.
- ♦ The time-triggered Network-on-Chip (TT-NoC) between the cores provides predictable message transfer and error containment.
- ♦ A trusted network authority (TNA) supports dynamic bandwidth allocation of the NoC.
- ♦ A diagnostic core collects and analyses diagnostic and security information and performs on-line analysis.

Time-Triggered Multi-core SoC

TISS: Trusted Interface Subsystem

TNA: Trusted Network Authority

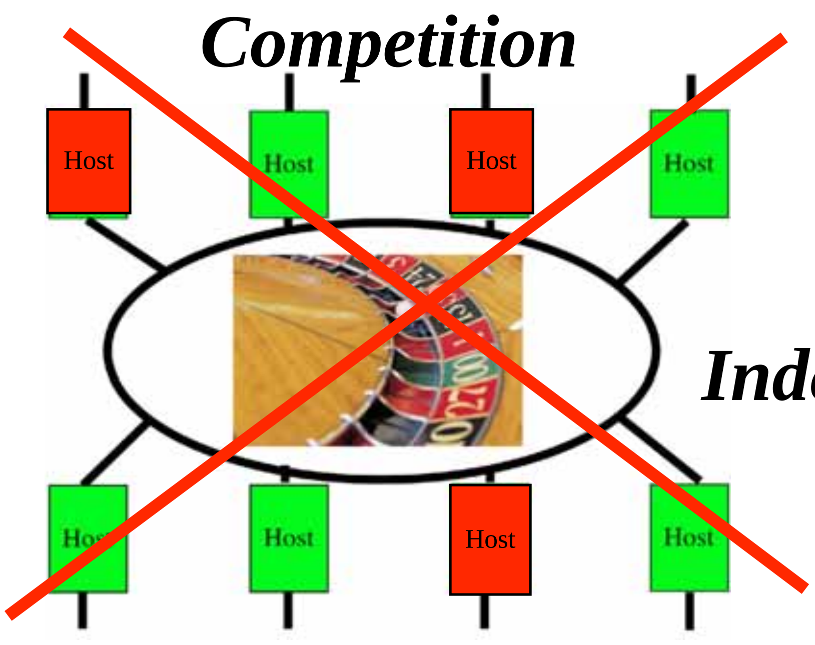
RMA: Resource Management Authority



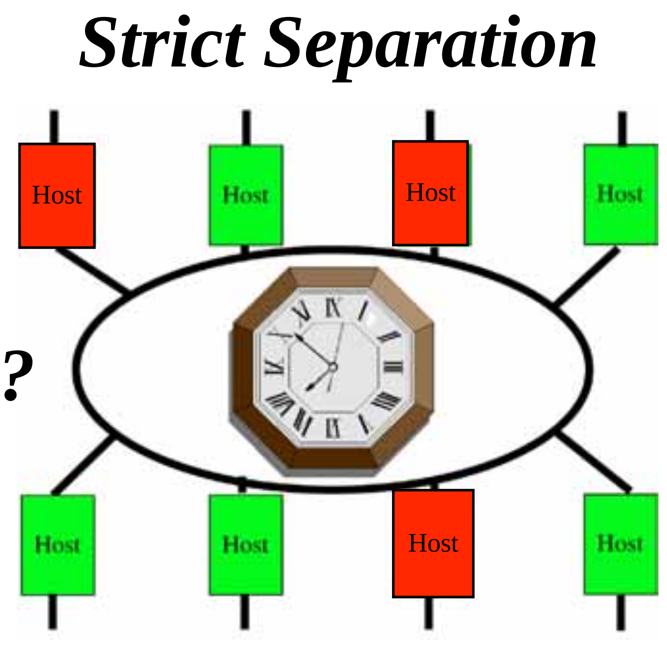
Difference between the Cell and our TTSoC

Cell Processor	**TT-SoC**
Designed to run a *single monolithic application*	Designed to provide an execution environment for *nearly independent jobs from different DASes*.
Decomposition of the application into parallel modules is the critical issue.	Fault Isolation and Error Containment between the jobs executing on different cores are the critical issues
Asynchronous on-chip communication	Time-triggered on-chip Communication
Single Distributed Operating System	Multiple heterogeneous Operating Systems (one in each core)

TT-NoC-Protocol Supports Composability



Independence?



Cell
Event-Triggered Protocol

TT-SoC
Time-Triggered Protocol

The TT-NoC protocol provides a basic predictable flow-controlled unidirectional message transport and a clock sync service.

Fault Hypothesis of the TTSoC

We distinguish between three types of faults

- ♦ Design Faults

  We assume that *any application core* of the TTSoC can have an *arbitrary* design fault, but the NoC and the TNA are *free of design faults*.

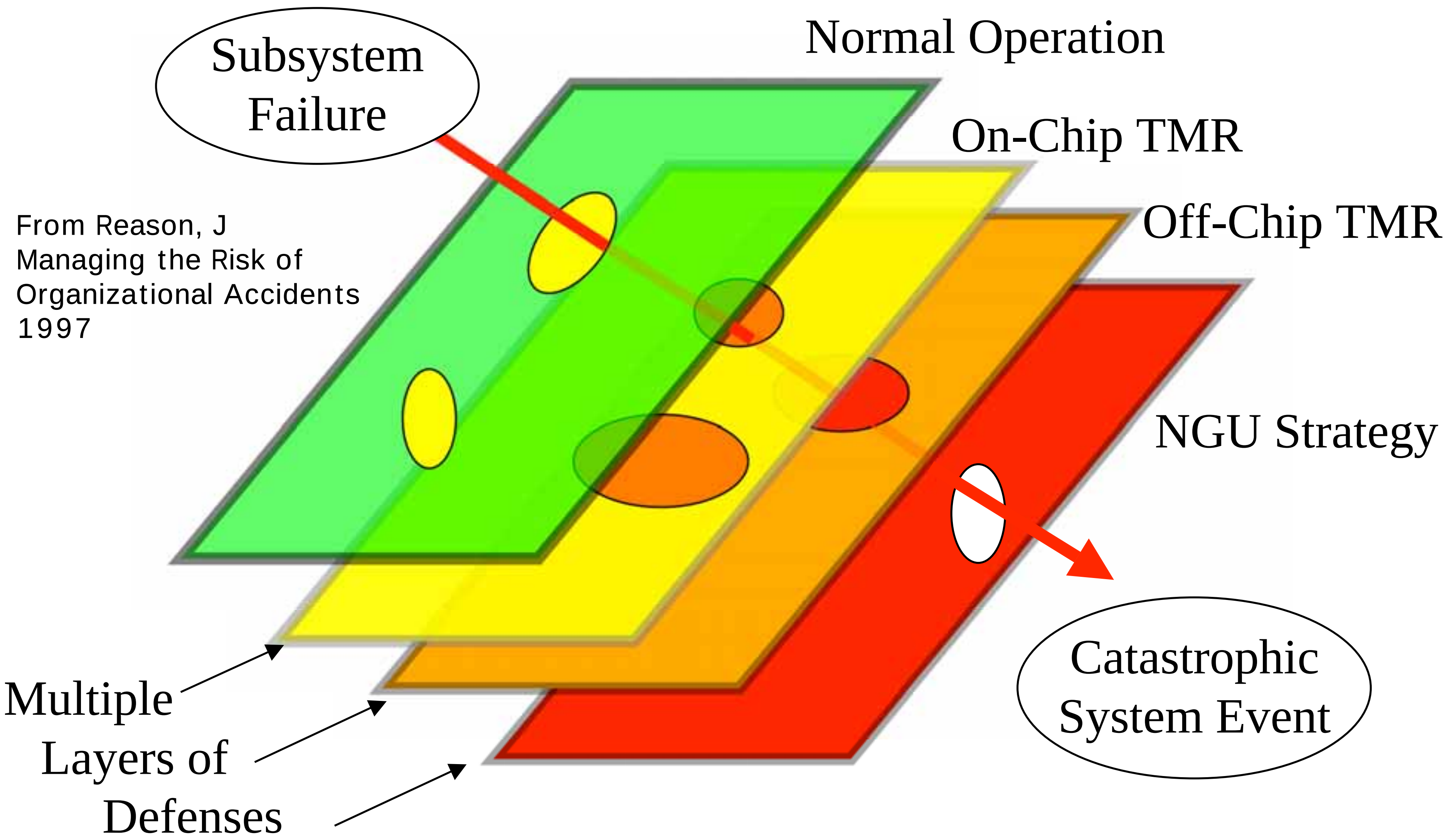
- ♦ Permanent Physical Faults

  Failure rate > 10 years

- ♦ Transient Physical Faults

  Orders of Magnitude higher than permanent failure rate (e.g., caused by ambient cosmic radiation).

Approach to Safety: The *Swiss-Cheese* Model



From Reason, J
Managing the Risk of
Organizational Accidents
1997

Integrity-Level of Application Domains

Application	System MTTF w.r.t. permanent failures (in years)	System MTTF w.r.t transient failures (in years)	Data-integrity requirement	Market volume	Examples
Low-Integrity	> 10	> 1	low	huge	Consumer Electronics
Moderate-Integrity	> 100	> 10	moderate	large	Present-day automotive
High-Integrity	> 1000	> 100	very high	moderate	Enterprise server
Safety-Critical	> 100 000	> 100 000	very high	small	Flight control

The Dilemma

- ♦ The consumer electronics (CE) domain has the size to support the large development costs needed to build powerful SoCs.
- ♦ Since in the near future there is no need to harden CE chips to mitigate the consequences of ambient cosmic radiation, the CE industry will not pay extra for hardening their chips.
- ♦ Architectural mitigation strategies have to be developed such that ***replicated mass-market chips*** can be used to build ***high-integrity embedded*** systems.

Mitigation Strategies w.r.t. Soft Errors

It is hardly possible to shield a device from the neutrons of ambient cosmic radiation, however the effects of this radiation can be mitigated on different levels:

- ♦ Material selection to reduce the *neutron absorption coefficient* (Example: SoI *Silicon in Insulator*)
- ♦ Layout of electronic devices: e.g., larger devices
- ♦ Radiation-hardened circuit design--more transistors
- ♦ Error detection and correction for SEU (e.g., 64 bit words requires 8 additional bits). Mitigation of SETs more difficult.
- ♦ **High-level architectural means: triple modular redundancy (TMR) in space and/or time.**

What is Needed to Implement TMR?

What architectural services are needed to implement Triple Modular Redundancy (TMR) at the architecture level?

- ♦ Provision of an Independent Fault-Containment Region for each one of the Replicas
- ♦ Synchronization Infrastructure
- ♦ Predictable Multicast Communication
- ♦ Replicated Communication Channels
- ♦ Support for Voting
- ♦ Deterministic (*which includes timely*) Operation

Fault Isolation (Independence)

We consider cosmic ambient radiation as the prime source of failures.

- ♦ Since the affected area of a *neutron hit* is in the order of a few μm^2 we make the assumption that only a single core is affected by an SEU--fault isolation is achieved by the geometry of the SoC.
- ♦ An SEU can cause an arbitrary failure of the affected unit.
- ♦ In a *non-protected NoC* an SEU hit is catastrophic if it affects the NoC.
- ♦ The effects of an SEU of a core is contained by error containment at the TISS.

Error Containment

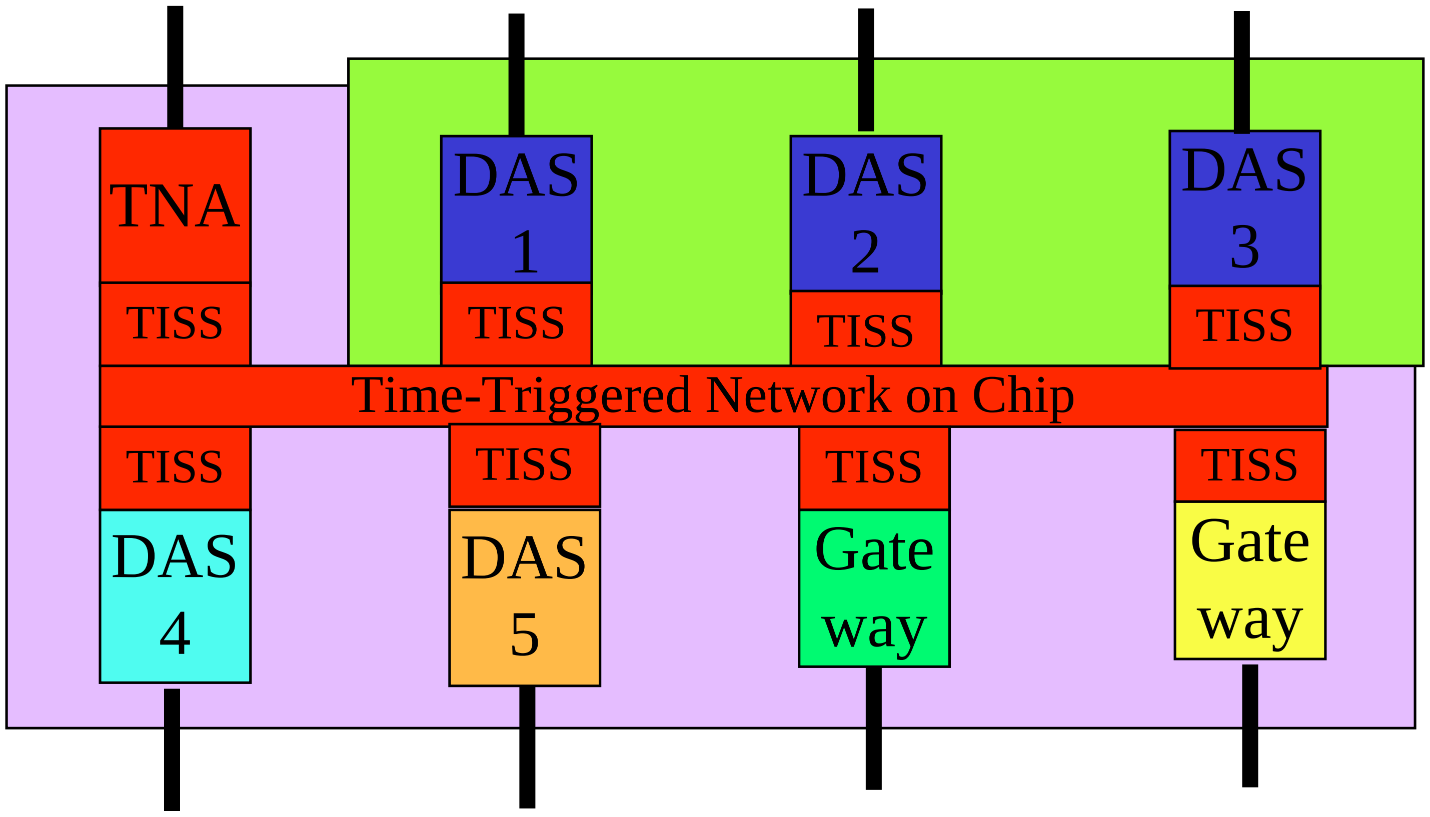
- ♦ The TISS (Trusted interface subsystem) is under the control of the TNA and cannot be modified by its core.
- ♦ Error containment of a core in the temporal domain is realized by the TISS.
- ♦ Error containment in the value domain is achieved by TMR that detects and masks value errors.
- ♦ The TISS protects the rest of the TTSoC of an arbitrary core failure.

Protection of the TT-NoC

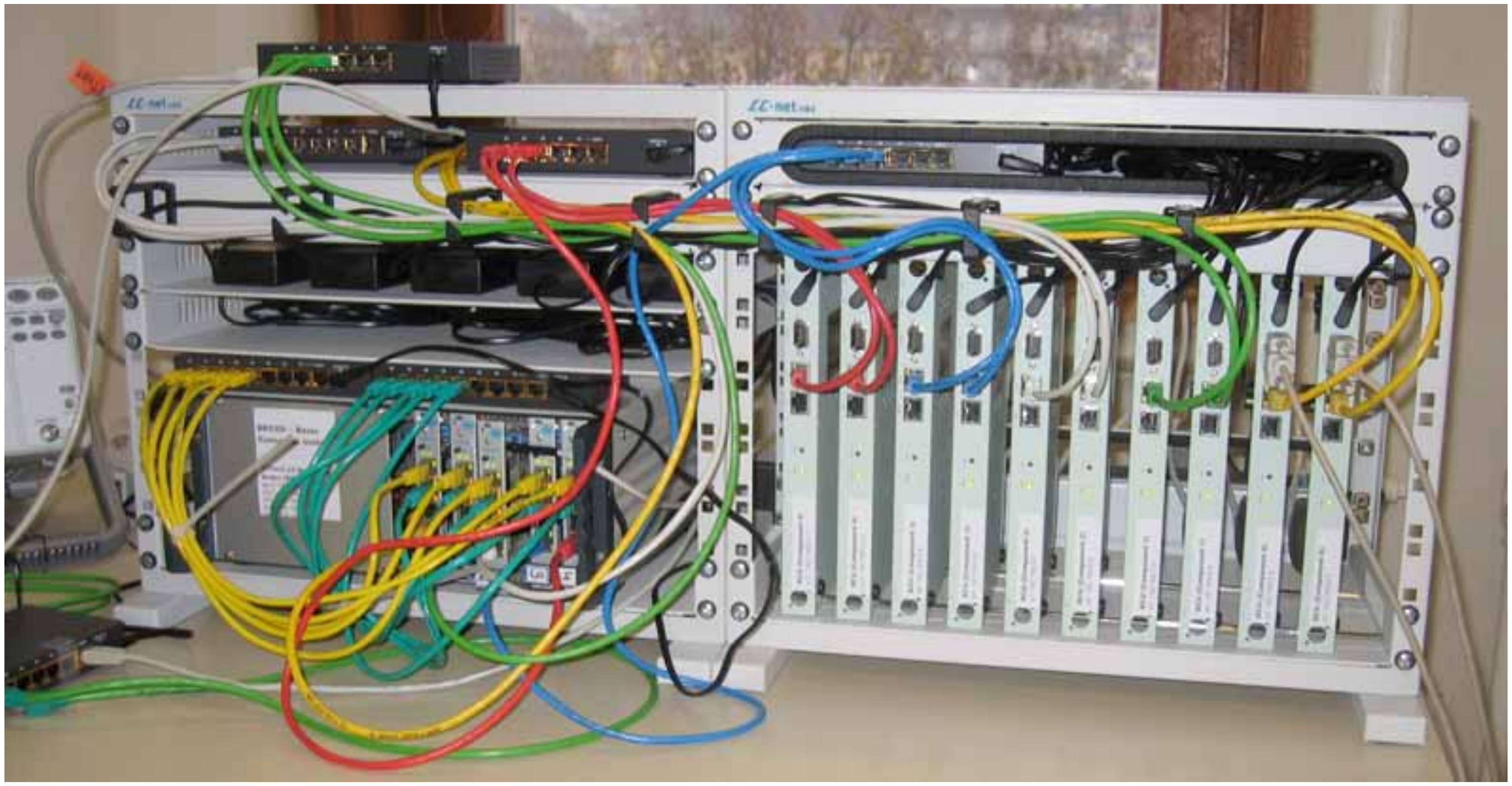
In order to make the TT-NoC resilient w.r.t. single bit upsets, we can provide error correction within the NoC and the TISSes:

- ♦ Extend the on-chip messages by an error correcting CRC
- ♦ Protect the state information in the NoC by error correcting codes.
- ♦ Transmit a message twice in order to mask a message error.

TMR within an SoC to mask Transient Faults



Prototype of a TT NoC at TU Vienna

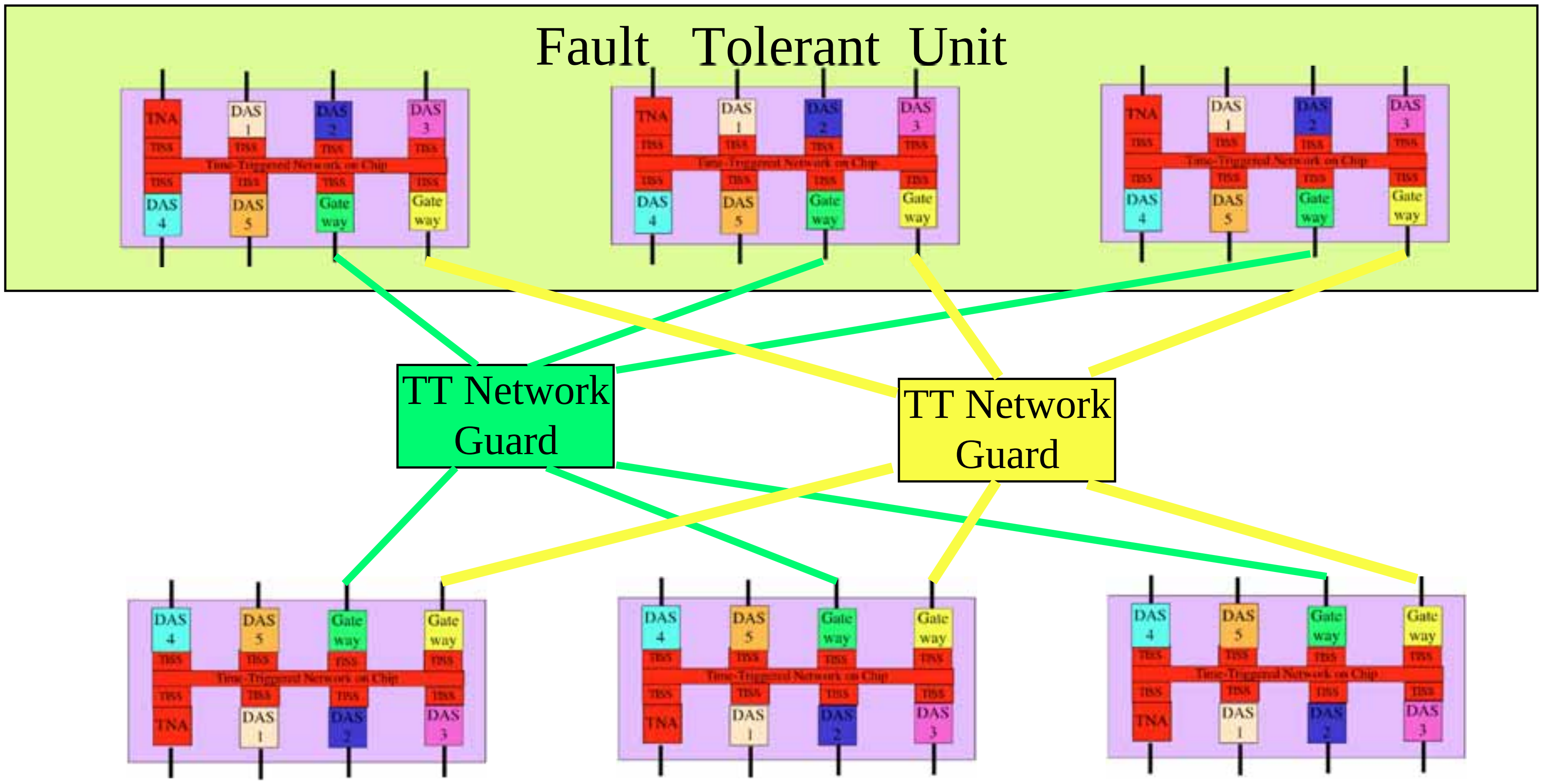


A set of application computers (micro-components) and connector units form a node of a cluster.

Experimental Evaluation

- A prototype of the proposed TTSoC was built consisting of two computers (one for the host, one for the TISS) at each core and TT Ethernet as the time-triggered communication system.
- Extensive fault-injection experiments were performed to test the fault hypothesis.
- Not much was learned, since the system behaved as expected.
- Work on a second prototype of the TTSoC, where the TTNoC is implemented in an FPGA is in progress.

Fault Tolerant Configuration



Conclusions

- ♦ Recent developments in computer architecture--the appearance of multi-core SoCs (MPSoC)-- support the move from a *federated* to an *integrated* architecture.
- ♦ A necessary precondition for the integration of multiple DAS jobs on a single MPSoC is comprehensive fault-isolation and error containment.
- ♦ Significant improvements in the dependability and substantial cost savings can be realized, if *properly designed multi-core chips* that support fault isolation and error containment are deployed in the transportation domain.