Time and Determinism

H. Kopetz

TU Wien

September 2007

Outline

- Introduction
- Determinism
- State and Time
- TMR Systems
- Sparse Time Base
- Conclusion

Simplicity

The report on *Software for Dependable Systems: Sufficient Evidence?* from the National Academies of July 2007 contains as one of its central recommendations

. . One key to achieving dependability at reasonable cost is a serious and sustained commitment to simplicity, including simplicity of critical functions and simplicity in system interactions. This commitment is often the mark of true expertise.

The Major Design Challenge

It is impossible to reason about the behavior of an SoC (System on Chip) consisting of a billion transistors, switching with a frequency of 1 GHz at the level of activity of every transistor: We need a model, i.e., *a deliberate simplification of reality with the objective of explaining a chosen property of reality that is relevant for a particular purpose*.

The major challenge of design is the building of a software/hardware artifact (an embedded computer system) that provides the specified services under given constraints and where relevant properties of this artifact can be modeled at different levels of abstraction by simple models.

Models in the *Natural Sciences*

Deterministic models of the natural phenomena are the basis of our technical civilization. The identification of abstraction levels and the development of corresponding deterministic models, where the indeterminism of the world at the lower levels does have only a negligible effect, are at the root of scientific discovery and engineering practice.

Models in *Computer Science*

Whereas the natural scientist must uncover the regularities of a given reality and find a suitable level of abstraction in order to formulate appropriate models and theories that explain the observed phenomena, the computer scientist is—*at least theoretically*—in a much better position: The computer scientist has the freedom to design the system—an artifact—which is the subject of his modeling.

The requirement to build artifacts the properties of which can be analyzed by *simple models* should thus be an explicit design driver.

In many areas of computer science, this principle of building artifacts that can be modeled by *simple models* is violated. For example, the temporal behavior of a modern pipelined microprocessor cannot be captured in a *simple model*.

(First) Definition of *Determinism*

A model behaves deterministically if and only if, given a full set of initial conditions (the initial state) at time t_o, and a sequence of future inputs, the outputs at any future instant t are entailed.

Requirements:

- *A defined and stable initial state*
- *An appropriate model of time*
- *An algorithm that computes the next state and the output, based on the previous state and the input*

Why Should we Aim for *Deterministic* Models?

Deterministic models have many advantages over non-deterministic models since they enable the confident prediction of future behavior.

- *Determinism is an innate property of rational thinking*, since it forms the basis for logical reasoning.
- It is easier to abstract from deterministic models than from non-deterministic models.
- Determinism in needed in TMR Systems
- It is difficult to validate a non-deterministic system, since repeated test cases do not have to produce identical results in a non-deterministic environment.

For these reasons we should try to build artifacts that can be modeled by deterministic models whenever possible.

Is *Determinism* a *Boolean* Property?

Does it make sense to talk about:

- The ***absolute determinism*** is necessary to the pertinence of the tests [David, Aussag *et al.* 2000]
- From these measurements, the interrupt service routine (ISR) latency times can be seen to be ***quite deterministic*** and predictable[Baril 1999]
- An important class of multi-chassis test systems requires high I/O throughput, low latency and a ***highly deterministic*** I/O response [Cleary 1996]
- The communication over the network has to be ***strongly deterministic*** [Hedenetz and Belschner 1998]
- ***Limited deterministic*** by FTDMA media access (Flexible Time Division Multiple Access) in dynamic segment [Schedl 2003]

Citations compiled by John Peres in *Determinism Concept for Embedded Systems, Ikerlan-TU Wien, 2007*

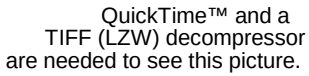
Ethernet *Determinism*

*The consensus appears to be that a network is **deterministic** if the chance of message delay is less than the chance for message loss as a result of noise, that is, one in ten million.*

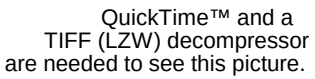
Ethernet behaves deterministically up to some 50 per cent loading for switched networks. This is quite easy to achieve for process control, because the messages are short, especially when using 100 Mbits/sec

From Jonas Berge, Fieldbus for Process *Control*, p. 64, Instrument Society of America 2004

AFDX Determinism

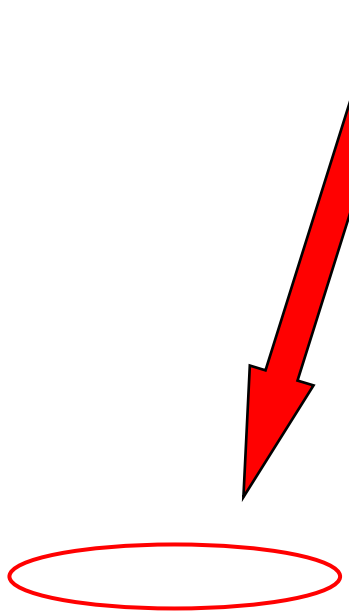


Ethernet vs. AFDX



AFDX Determinism

QuickTime™ and a
TIFF (LZW) decompressor
are needed to see this picture.



That is *Determinism . . .*

AFDX designers painstakingly compiled a log of how long message traffic takes to pass through the network under a wide variety of operational scenarios. AFDX builds gigantic database and they make sure not to exceed bandwidths that exceed their dependencies for determinism. Schuh says "There is nothing magic about it; you sit down with a spreadsheet to do the math for the worst-case scenarios to see if the worst possible delay is acceptable. ***That is determinism."*** (bold added).

From: Keller, J. (2007). "The coming revolution in commercial avionics data networking." Military & Aerospace Electronics (February).

URL: http://mae.pennnet.com/articles/article_display.cfm?article_id=284235

Richard Schuh is avionics product line manager at GE Fanuc embedded systems, Santa Barbara, California

Limits of Determinism

The success of deterministic models of natural phenomena is so spectacular that a number of philosophers believed that the *world* is totally deterministic.

- This belief was shaken at the beginning of the twentieth century with the publication of *Heisenberg's uncertainty principle* that states that is principally impossible to determine the *full set of initial conditions* of a system at the level of the micro world (quantum mechanics).
- The probability that this indeterminism of the micro-world will refute the determinism of the models that govern the macro-world behavior is so small, that it can be neglected in many situations.
- On the other end, the concept of *bifurcation* in Chaos Theory, which claims that in highly nonlinear systems arbitrary small causes can have unpredictable large effects in the future, is also incompatible with the view of a fully deterministic world.

State and *Time*

We follow the definition of Mesarovic, p.45 :

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.

The notion of *state* **presupposes** an appropriate *model of time.*

How can we establish a *consistent initial state* in a distributed system (which is the starting point of *deterministic behavior*)?

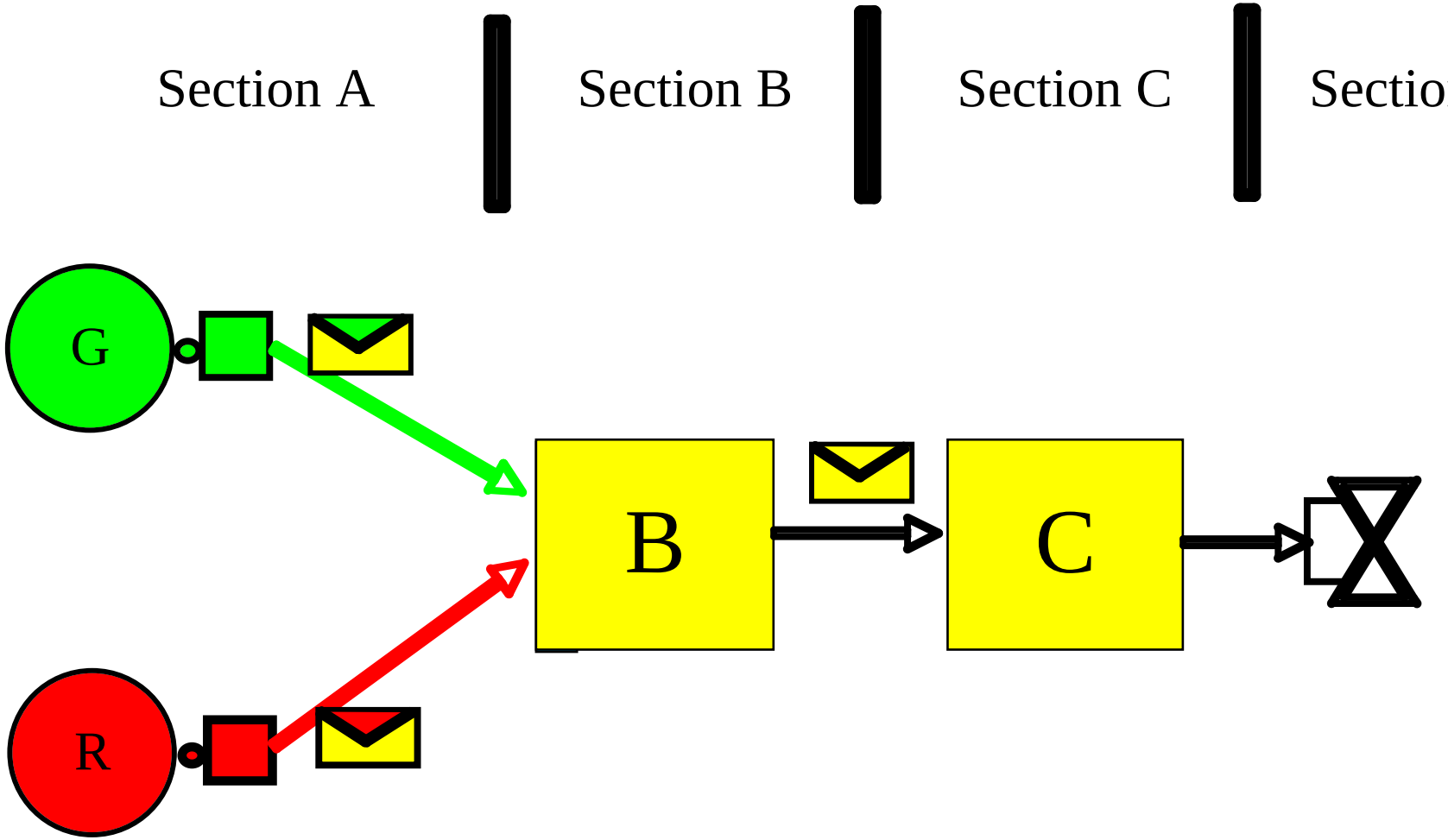
Simultaneity

The proper handling of *simultaneous* events poses a special challenge to the designer of a deterministic distributed system.

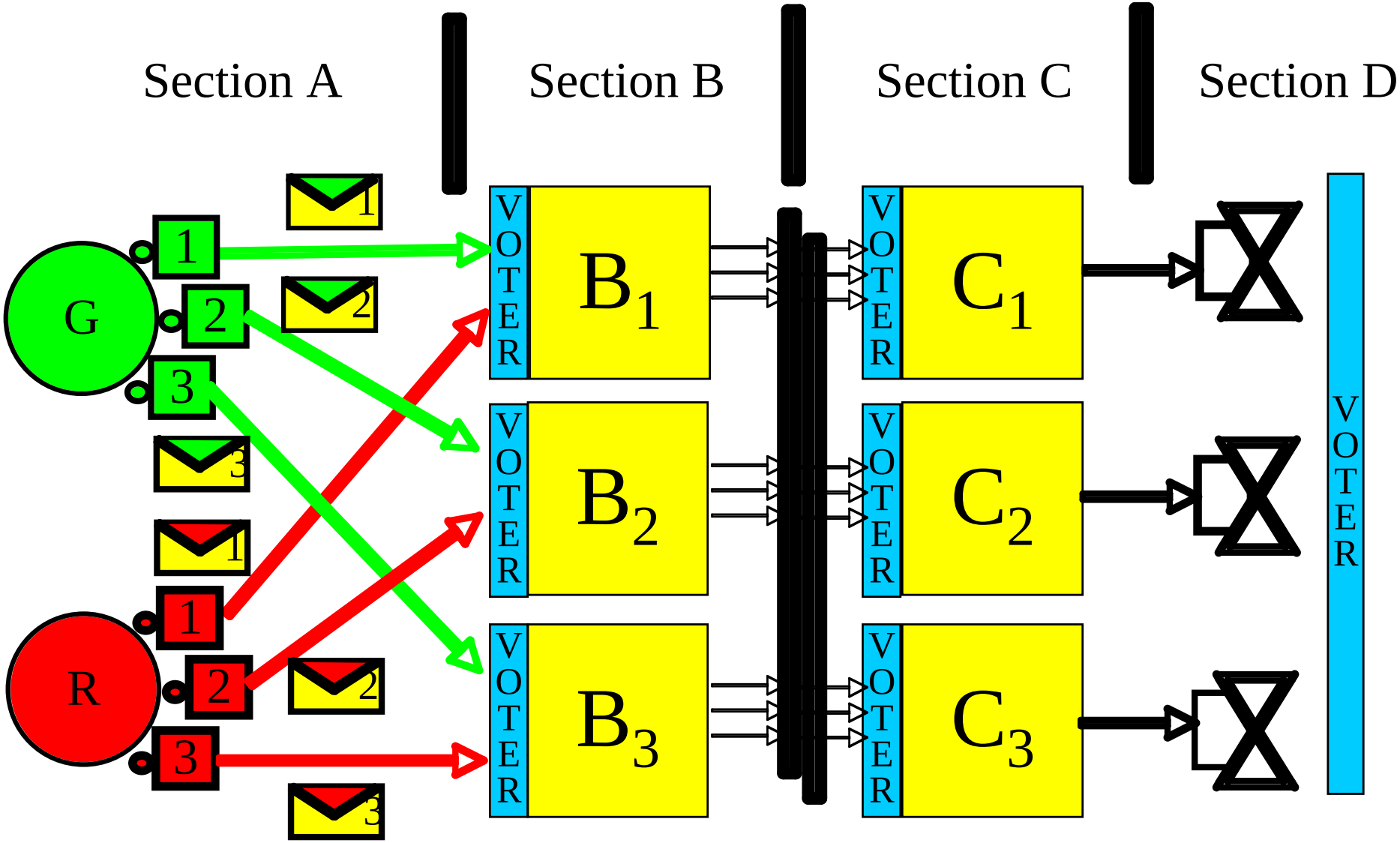
Simultaneity is at the root of a number of difficult problems in computer science:

- the *meta-stability problem* in the hardware,
- the *mutual exclusion problem* in operating systems, and
- the *consistent message ordering problem* in distributed systems.

Fault Masking by Triple Modular Redundancy

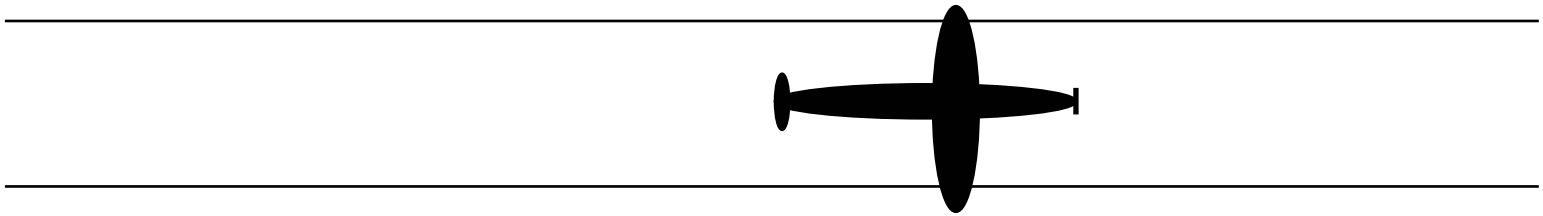


Fault Masking by Triple Modular Redundancy



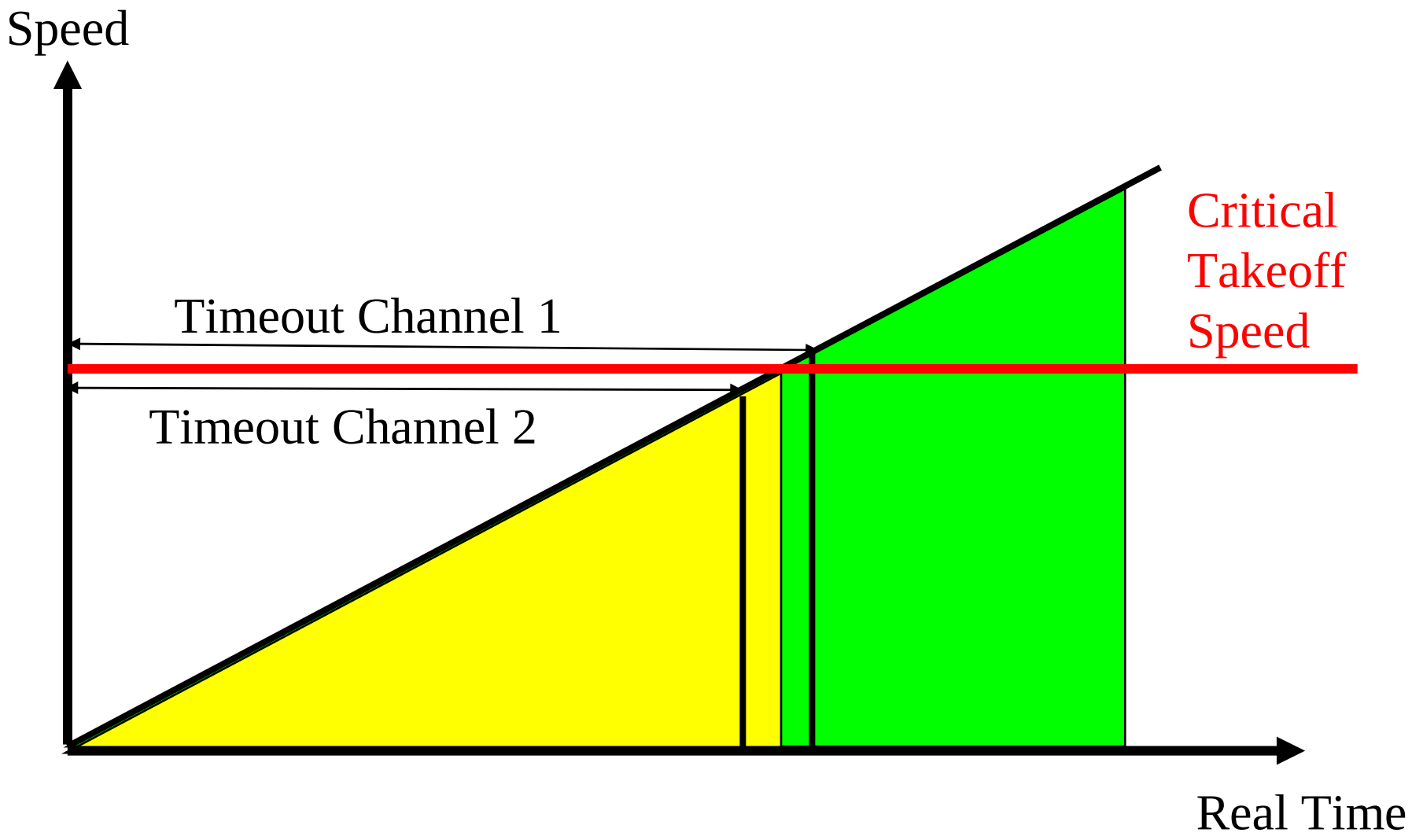
Example: Airplane on Takeoff

Consider an airplane with a flight control system consisting of ***three independent channels*** that is taking off from a runway. Consider the system at the *critical instant* before takeoff:



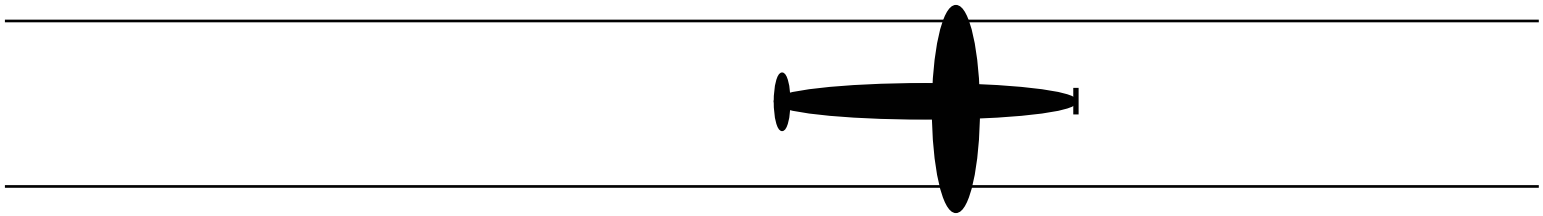
Channel 1	**Take off**	**Accelerate Engine**
Channel 2	**Abort**	**Stop Engine**

The Critical Role of Time



Example: Airplane on Takeoff

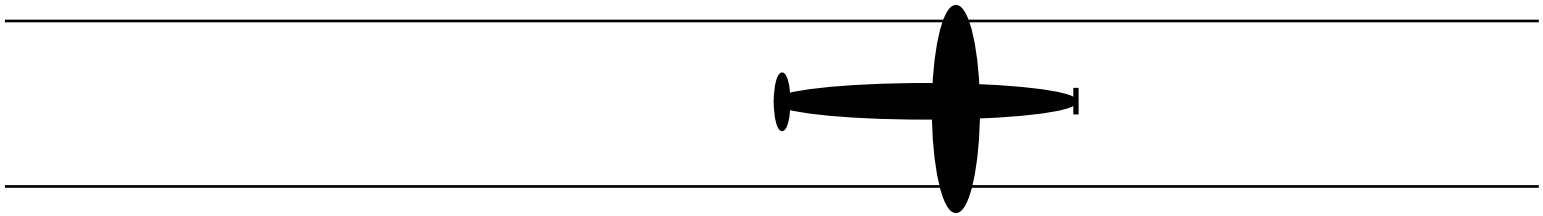
Consider an airplane with a flight control system consisting of ***three independent channels*** that is taking off from a runway. Consider the system at the *critical instant* before takeoff:



Channel 1	**Take off**	**Accelerate Engine**
Channel 2	**Abort**	**Stop Engine**
Channel 3	Take off	*Stop Engine (Fault)*

Example: Airplane on Takeoff

Consider an airplane with a flight control system consisting of ***three independent channels*** that is taking off from a runway. Consider the system at the *critical instant* before takeoff:



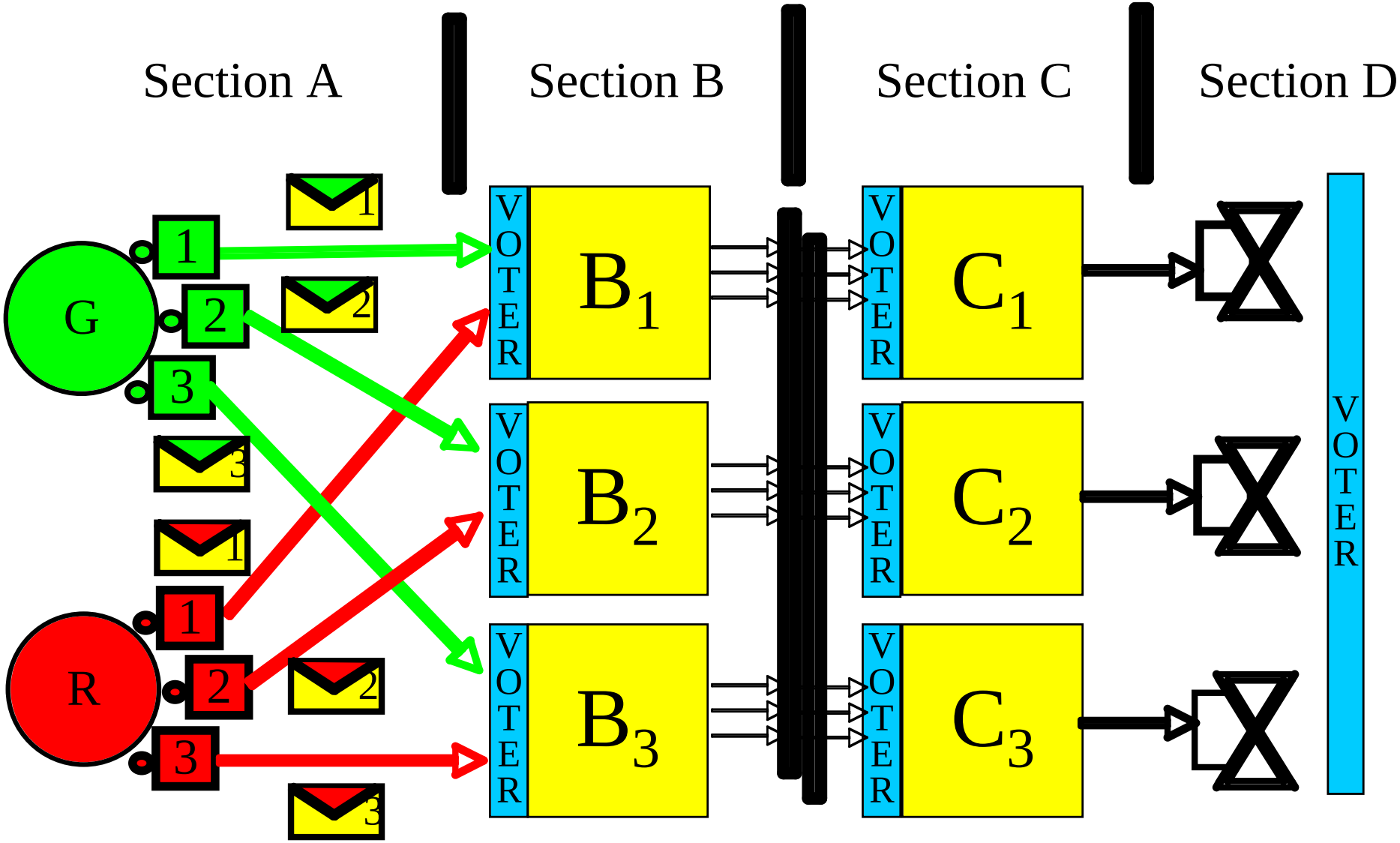
Channel 1	**Take off**	**Accelerate Engine**
Channel 2	**Abort**	**Stop Engine**
Channel 3	Take off	*Stop Engine (Fault)*
Majority	Take off	*Stop Engine (Fault)*

TMR Needs Determinism at all Levels

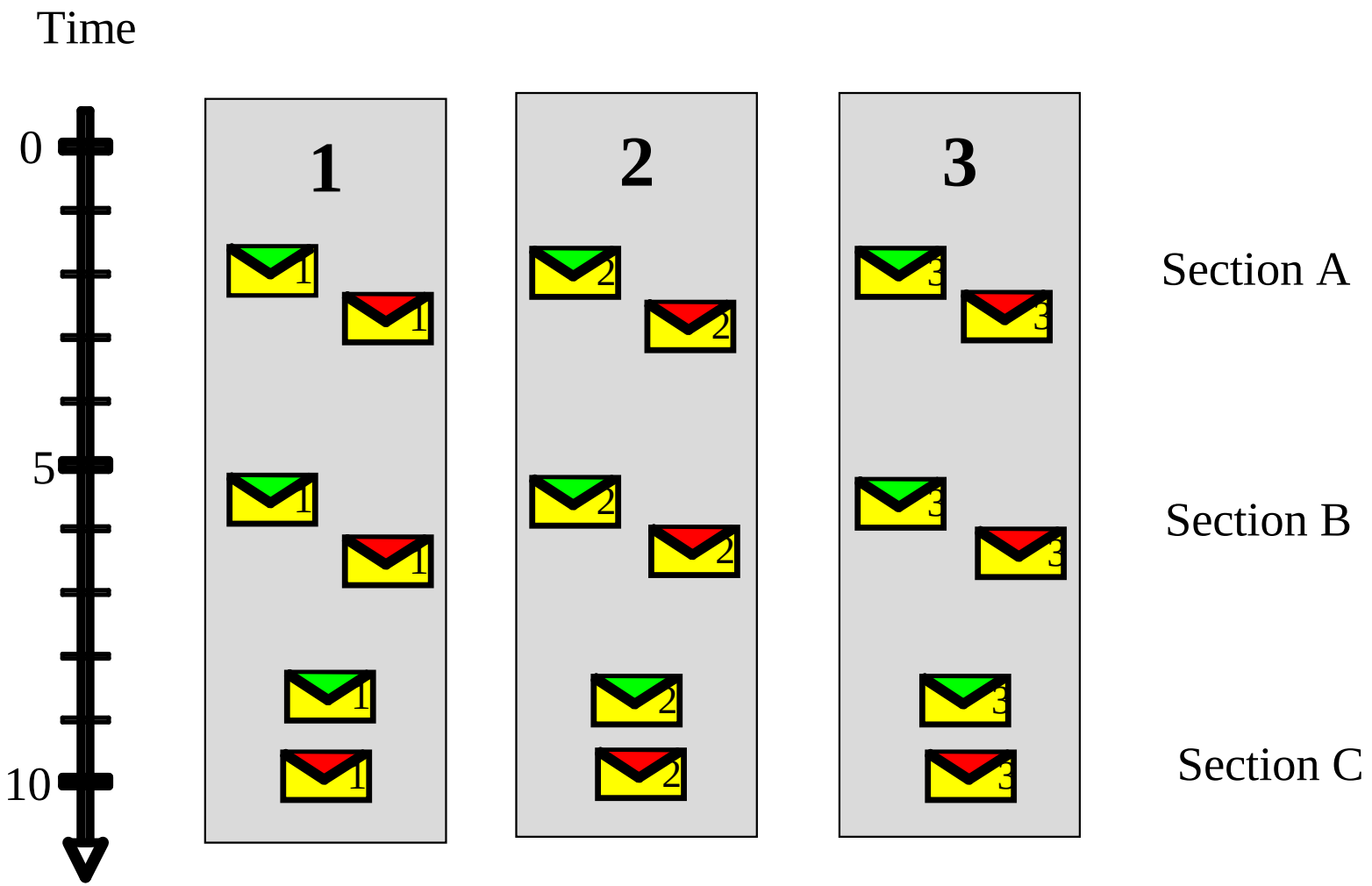
- Determinism of the Execution Environment
  - Hardware
  - Node Computer Operating system and Middleware
  - Communication System
- Determinism of the Application Software
- Consistent input (may need consensus protocols)
- Periodic masking of errors in the interface state
  - enough bandwidth for exchange of interface state
- Voting actuators

If determinism has *been lost*, it must be reestablished by agreement protocols (which requires additional software).

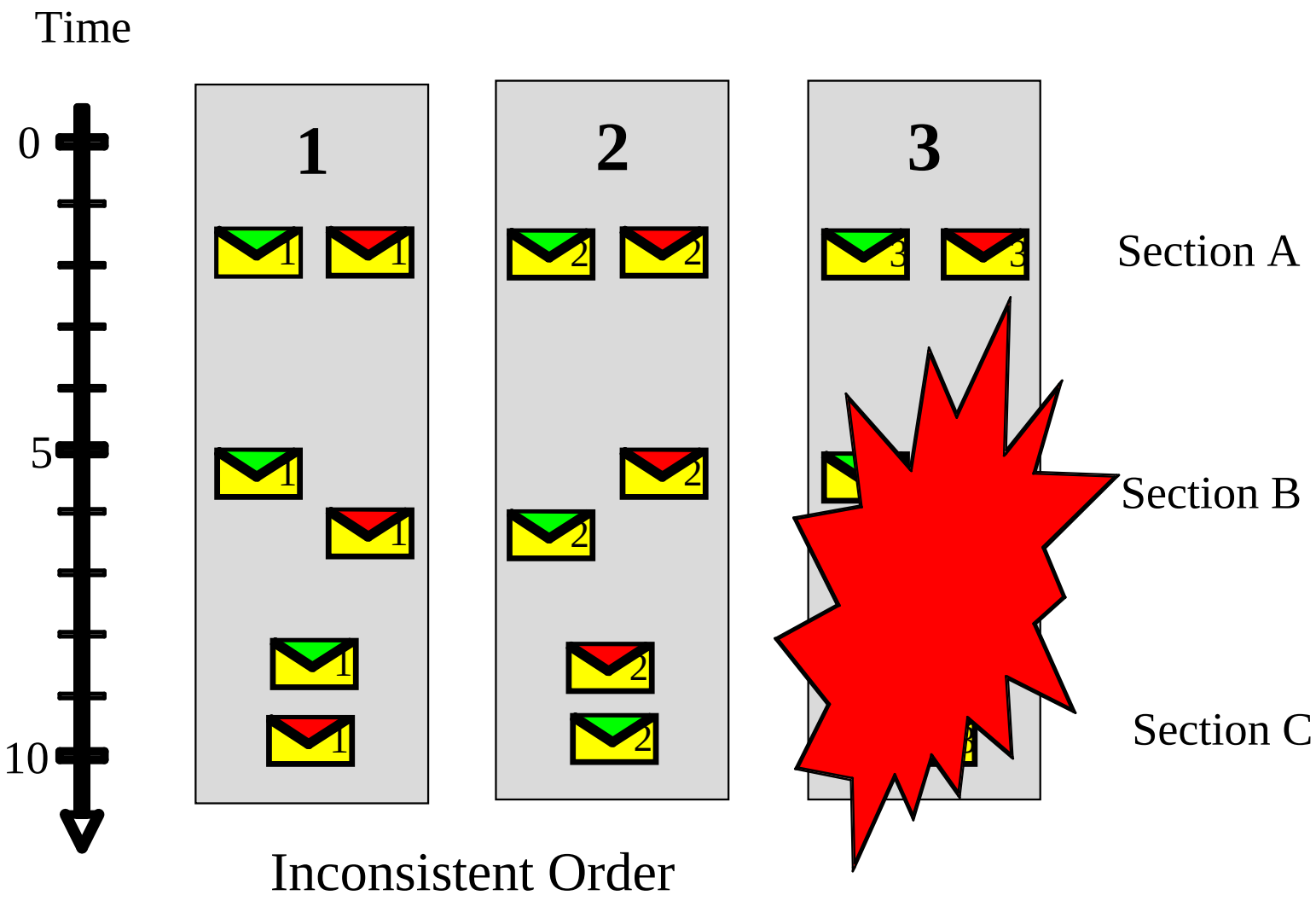
Fault Masking by Triple Modular Redundancy



Example: Three Channel System



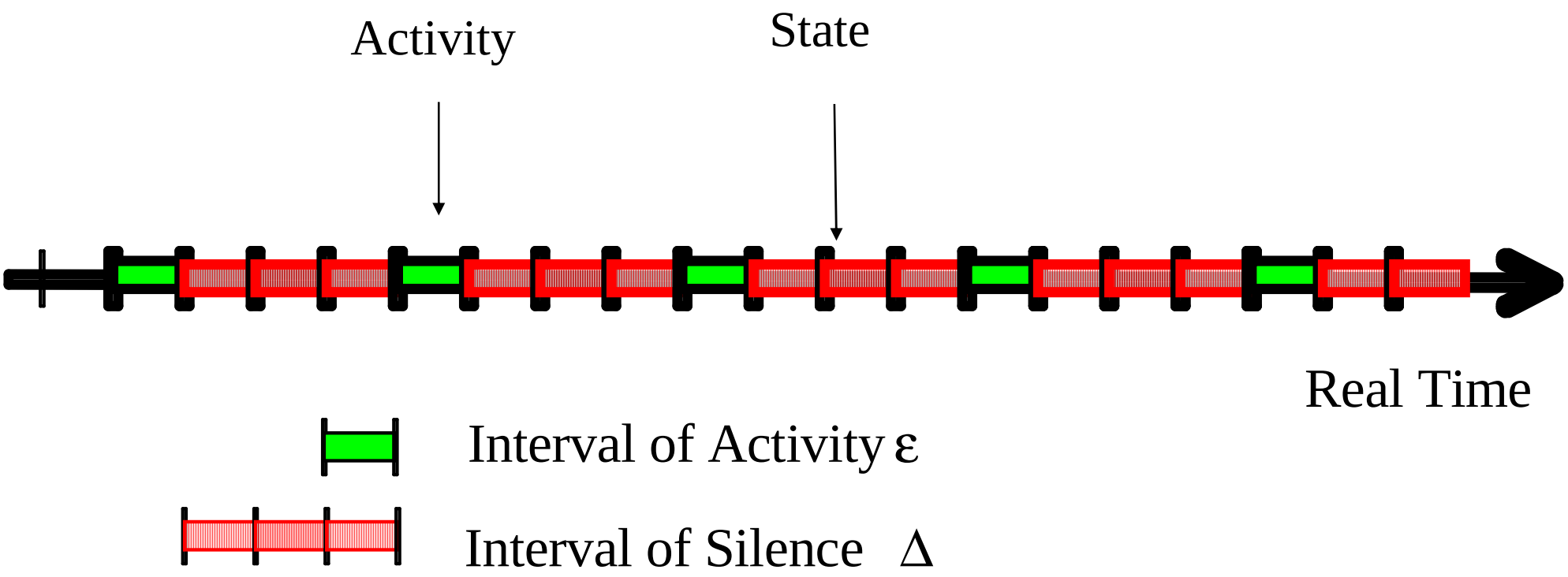
Example: Three Channel System Fails



Dense Time

It is principally *impossible* to arrive at a *consistent notion of simultaneity* in a physically distributed system if the events are allowed to happen on a *dense time-base*.

The Solution--A Sparse Time Base



Synchronous Languages are based on the sparse time model.

(Revised) Definition of *Determinism*:

Given the notions of sparse *time* and *state*, we are now in the position to define the *replica determinism* that is needed in Triple-Modular Redundant (TMR) systems more formally:

Let us assume a finite-state model with a state space Q, an input space Σ and an output space Δ then this model is said to behave R-deterministically iff, given a sequence of sparse real-time instants t_i, the state of the model q_0 $(t_0) \in Q$ at t_0 (now), and a sequence of future inputs a_i $(t_i) \in \Sigma$ then the sequence of future outputs $b_j(t_j) \in \Delta$ and the sequence of future states q_j $(t_j) \in Q$ is entailed.

Conclusion

- A lucid conceptual foundation forms a solid fundament for the construction of dependable systems.
- *Time*, *State*, and *Determinism* are three interlinked *basic-level concepts* that form an atomic triple.
- The sparse time model makes it possible to establish simultaneity consistently in a distributed system.
- Synchronous language are based on the sparse time model.