FAULT TOLERANCE TECHNIQUES

Fault Tolerance

“is the ability of a system to continue satisfactory operation in the presence of one or more non-simultaneously occurring hardware or software faults.”

DEFINITIONS

- Hardware fault
  - Some physical defect that can cause a component to malfunction. Eg a broken wire or a logic gate output that is toggling
- Software fault
  - Is a bug that can cause the program to fail for a given set of inputs
- Error
  - The manifestation of a fault.
- Fault latency
  - Duration between the onset of fault and its manifestation as an error
  - Impact the reliability of the overall system
- Error latency
  - Duration between when an error is produced and when it is either recognised as an error or cause the failure of the system

DEFINITIONS

- Error recovery
  - Process by which the system attempts to recover from the effects of an error.
  - Forward error recovery
    - Error is masked without any computations having to be redone
  - Backward error recovery
    - The system rolled back to a moment in time before the error is believed to have occurred.
    - Uses time redundancy, since it consumes additional time to mask the effects of failure

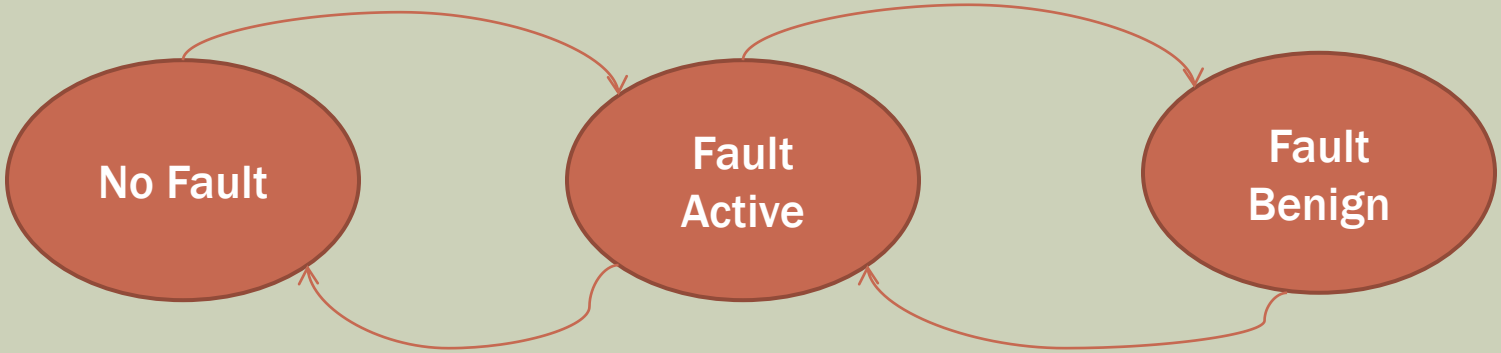
WHAT CAUSES FAILURE

- Errors in the specification or design
  - Mistakes in the specification and Design are very difficult to guard.
  - Many hardware failures and all software failures occur due to such mistakes.
  - It is difficult to ensure that the specification is completely right.
- Defects in the components
  - Hardware components can develop defects.
  - Wear and tear of components
- Environmental effects
  - Devices can be subjected to whole array of stresses, depending on the application.
  - High ambient temperatures can melt components or otherwise damage them.

FAULT TYPES

- Based on temporal behavior
  - Permanent Faults
    - Does not die away with time
    - Caused by broken wires
    - A>0, B=C=D=0
  - Intermittent Faults
    - The fault cycles between the fault active and benign states
    - Caused by loose wires
    - A>0, B>0, C=0 d>0
  - Transient Faults
    - Dies away after some time
    - Mainly due to environmental effects
    - A>0,B=0,C>0,D=0

FAULT TYPES



Fault Tolerance Systems

- To meet the demanding performance requirements,
  - FTS uses both type of fault tolerance – Hardware & software
  - Has the capability of automatic dynamic reconfiguration of the system.
- Depending upon the level of criticality and allowable POF
  - Dual, triple or quadruple are used
- Redundancy extends to all hardware elements, such as processors, sensors, actuators and data buses and to the software.

Fault Detection Scheme

- Central to all FTS principle is Fault Detection which identifies a fault.
- Approaches
  - **Replication** (Triple or higher)and **voting**
  - **Duplication** and **comparison**
  - **Self checking**

Electronic Flight Control System Architecture (most apt application of FTS)

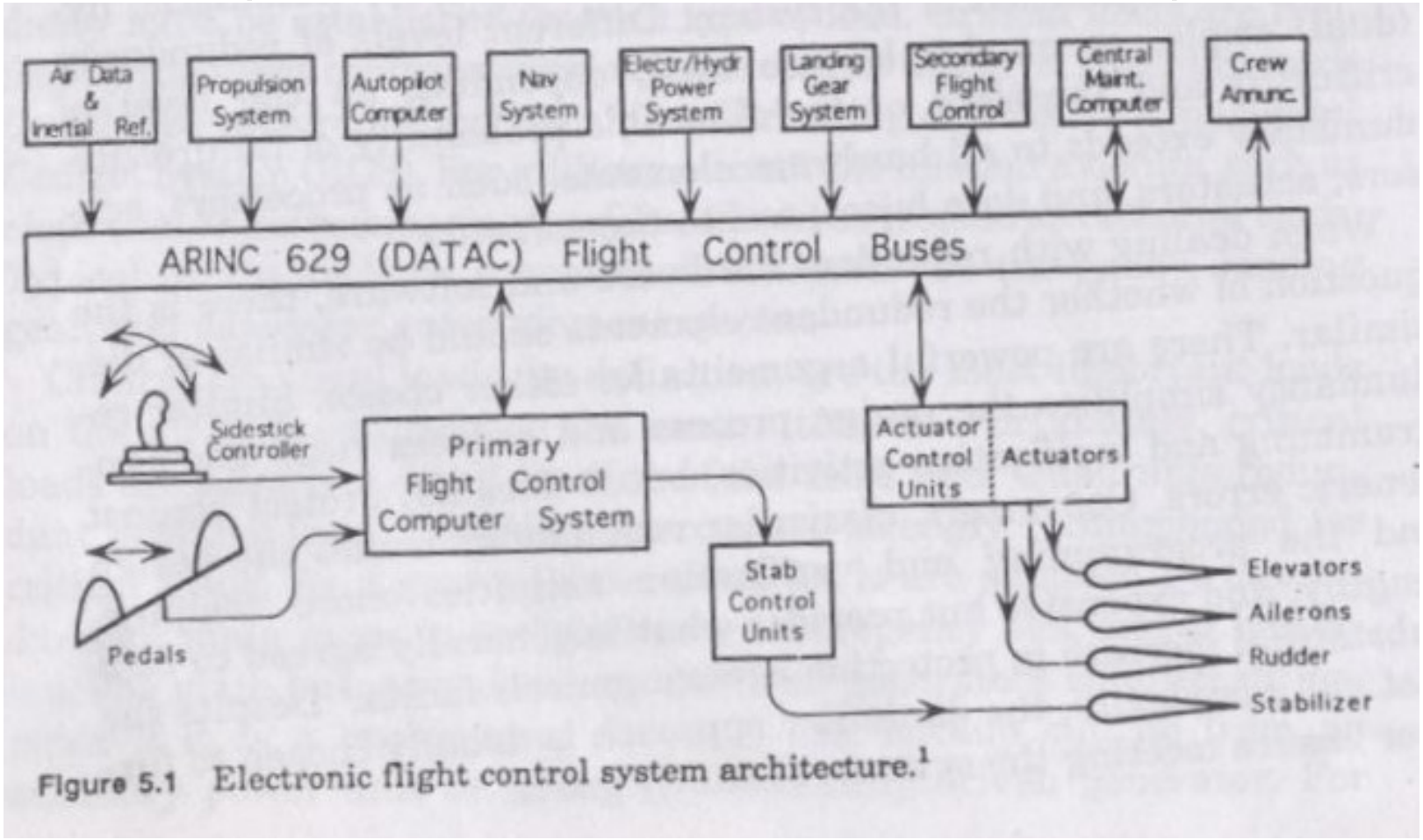


Figure 5.1 Electronic flight control system architecture.[1]

Multicomputer architecture for fault tolerance(MAFT)

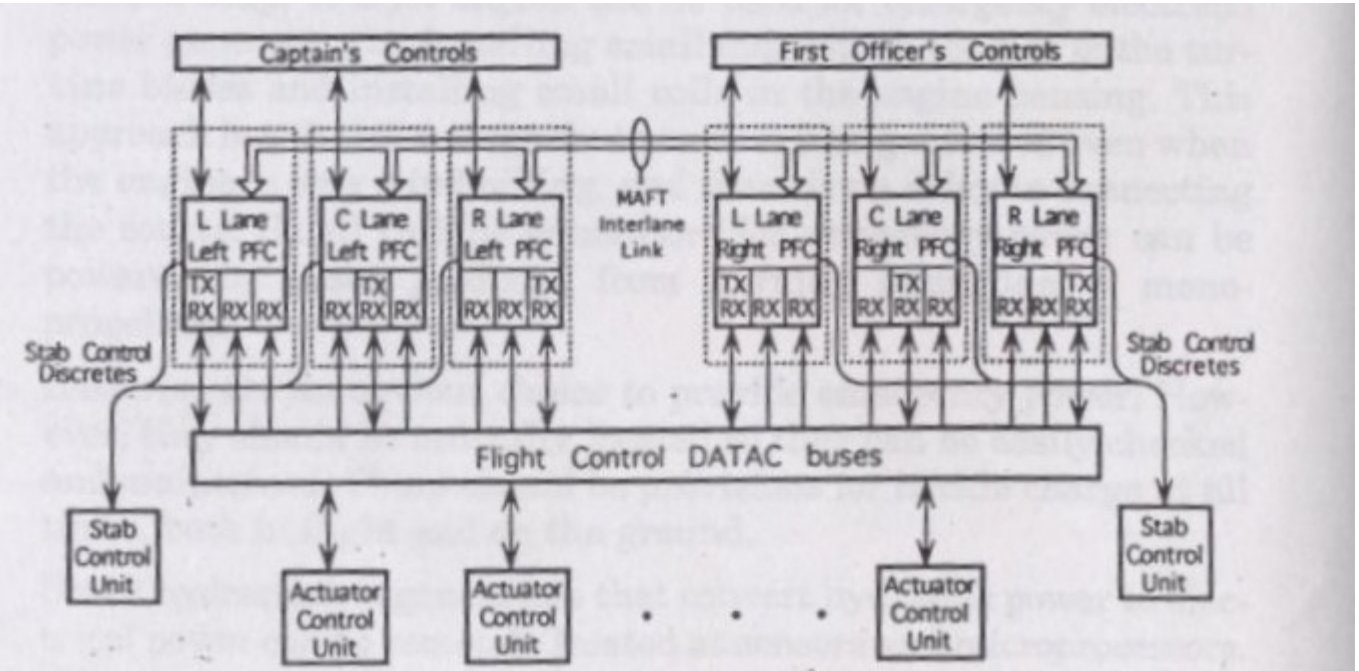


Figure 5.2 Multicomputer architecture for fault tolerance (MAFT).[1]

MAFT Lane Architecture

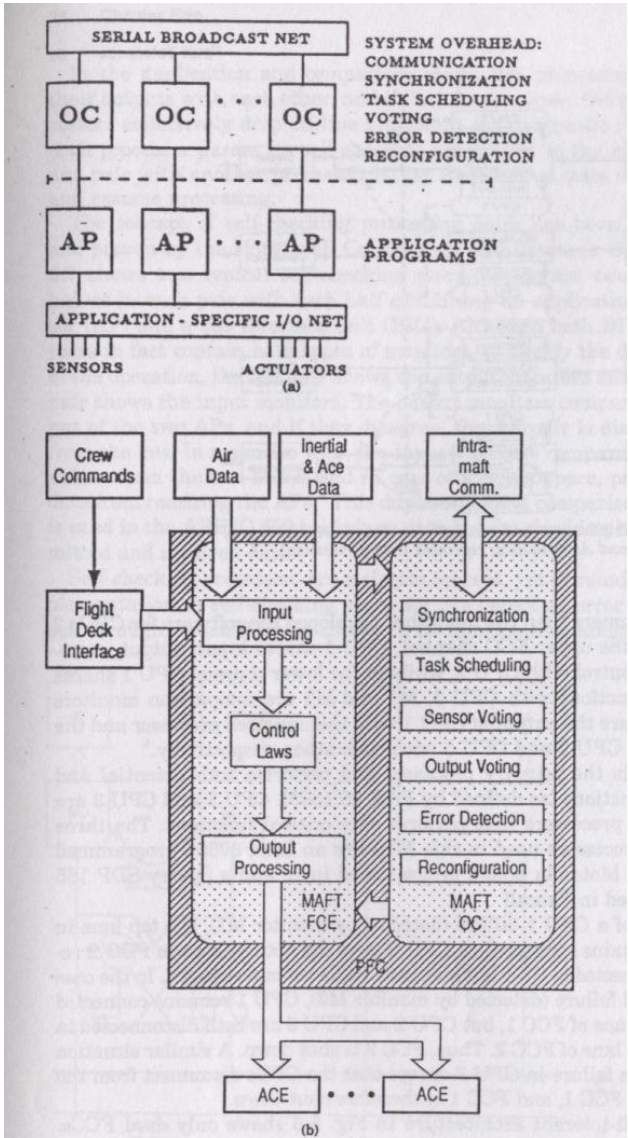


Figure 5.3 MAFT lane architecture (ACE=actuator control electronics).[1]

Primary Flight Control System configurations

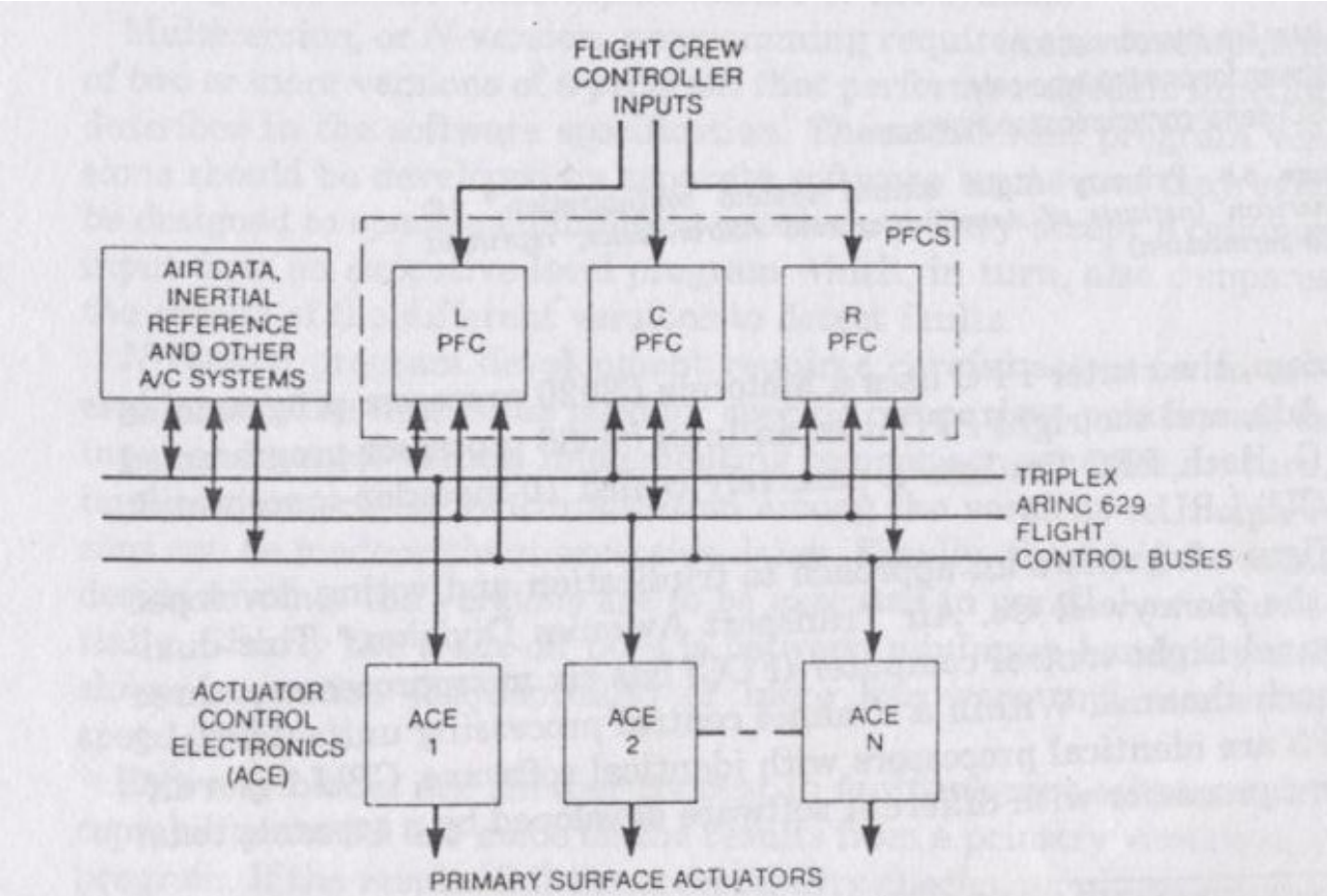
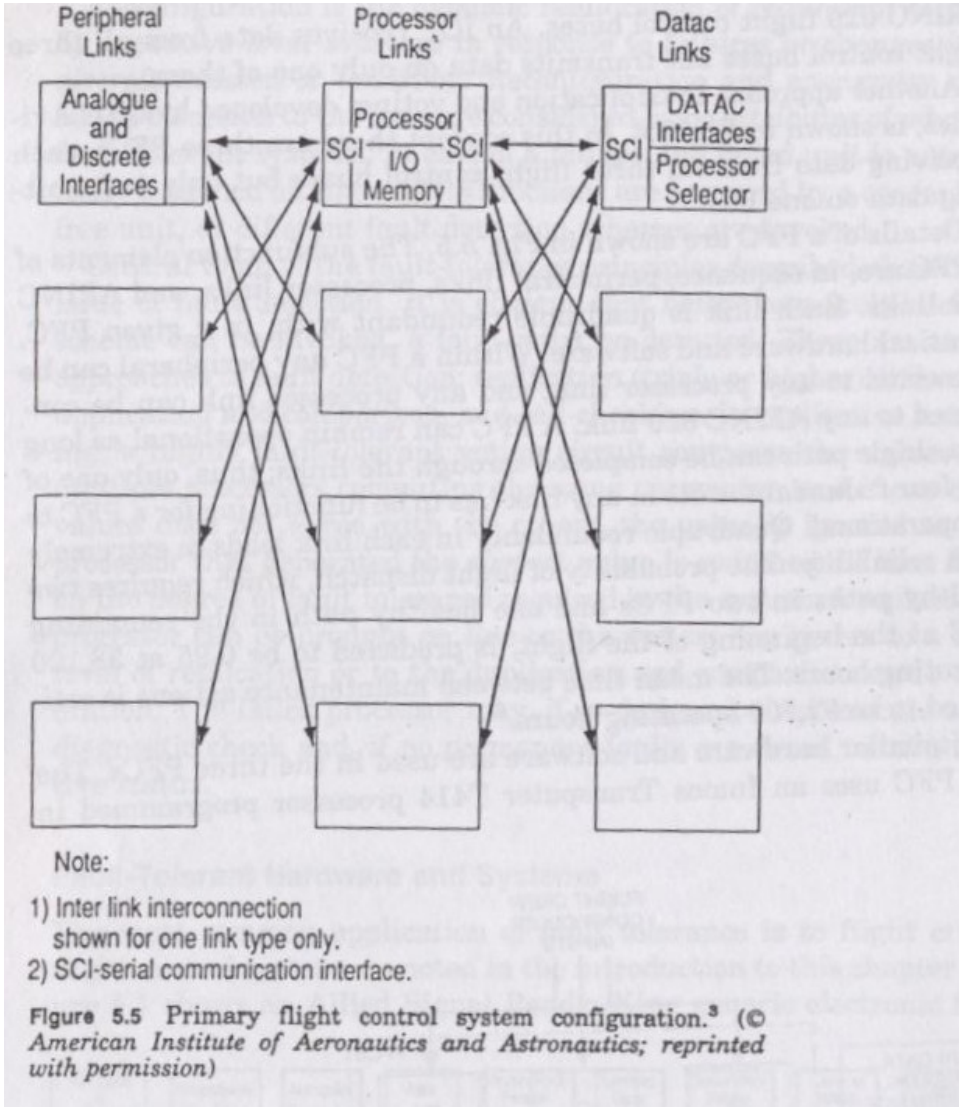


Figure 5.4 Primary flight control system configuration.[3] (© *American Institute of Aeronautics and Astronautics; reprinted with permission*)

Primary Flight Control System configurations



Note:

1) Inter link interconnection shown for one link type only.
2) SCI-serial communication interface.

Figure 5.5 Primary flight control system configuration.[3] (© *American Institute of Aeronautics and Astronautics; reprinted with permission*)

Triplication & voting

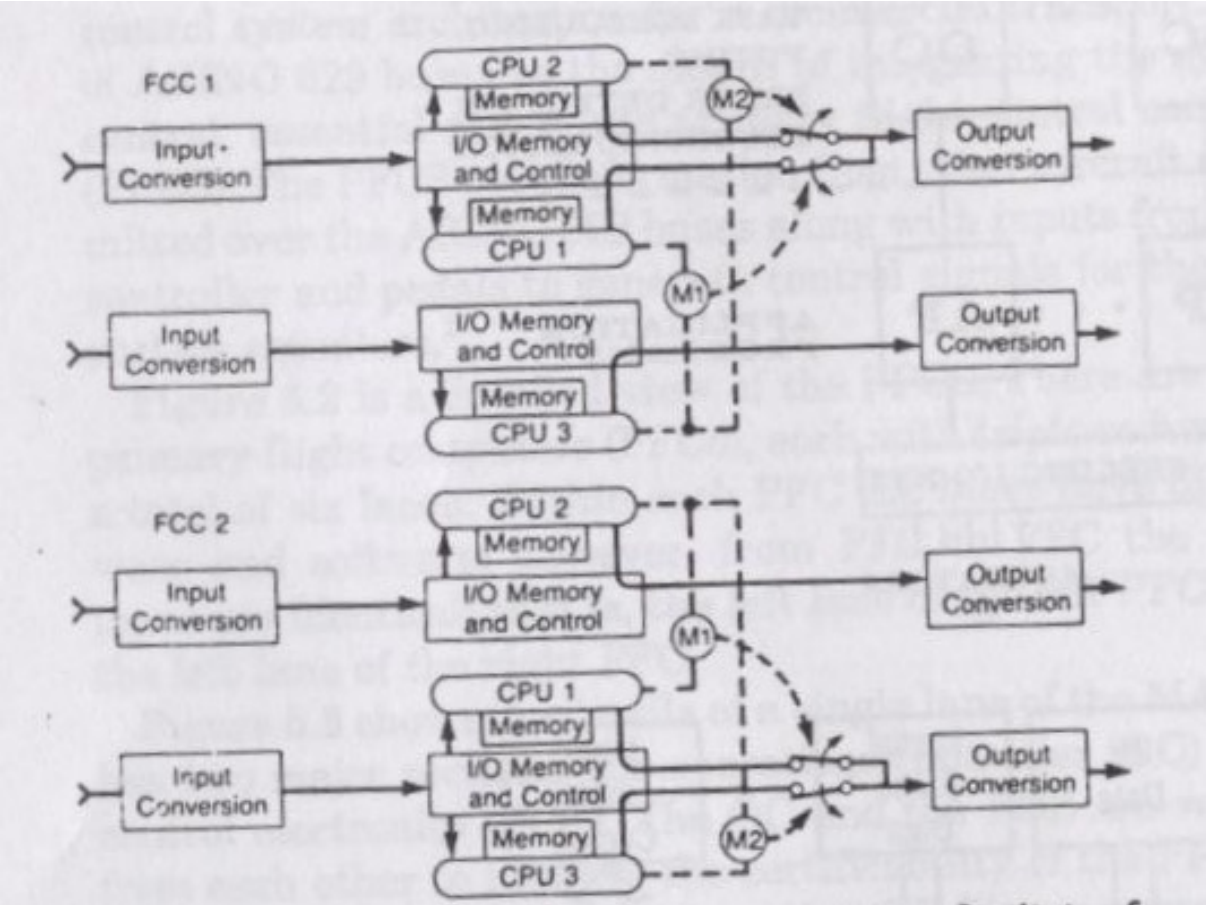


Figure 5.6 Triplication and voting.[4] (© *American Institute of Aeronautics and Astronautics; reprinted with permission*)

Replication and voting

- a highly fault-tolerant voting circuit compares the values from multiple processors computing the same parameter, and if one of the values does not agree with the others, the value is ignored and the processor that generated the suspect value is switched offline.
- Next
  - a replacement processor can be brought online
  - Or the system can revert to a lower level of replication
  - Or to the duplication and comparison mode of operation
- The failed processor may, if so designed, then execute a self diagnostic check and, if no permanent faults are found, return to active status.

Duplication & Comparison

- Two processors compare their outputs with each other, and if they do not agree, the pair of processors collectively drop off line and begin self-diagnostic routines
- If each processor passes its self check, it can return to the active state and pair with another processor, either its previous mate or another, and resume processing.

Self Checking pairs

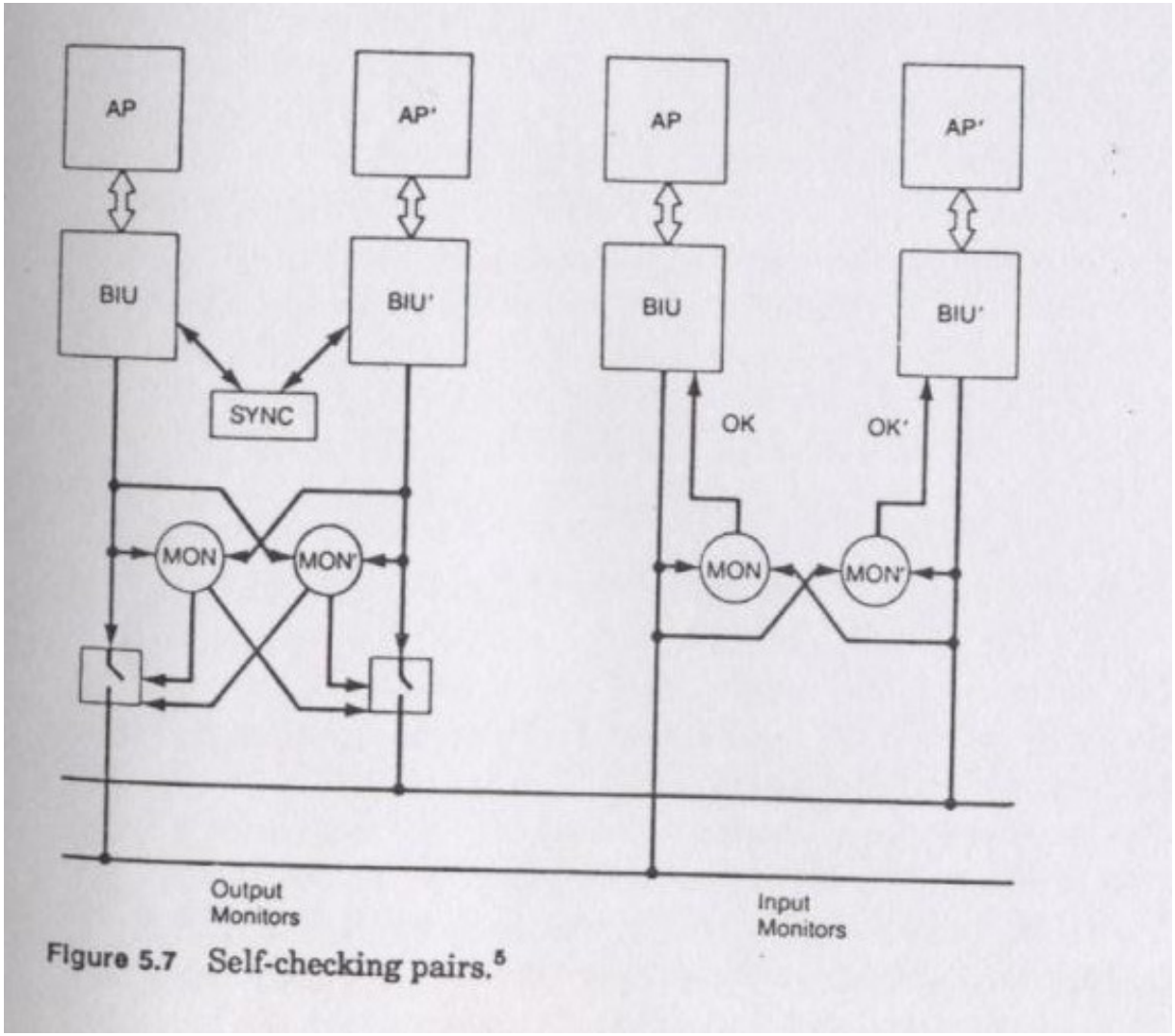


Figure 5.7 Self-checking pairs.[5]

Self checking pairs

- Simple concept but yet demanding in application
- Can detect an error within itself through reasonableness checks on its intermediate and/or final results without reference to other processors.
- In case of error, it will simply switch itself off and may if so programmed, automatically bring a spare processor on line as a replacement.

FAULT TYPES

- Based on Output behavior
  - Malicious faults
    - Also called as byzantine failures
    - Inconsistent output, harder to neutralize these errors
    - It behaves arbitrarily
  - Non malicious faults
    - Consistent output errors
    - Easier to neutralize these errors
  - Fail stop
    - Responds to up to a certain maximum number of failures by simply stopping, rather than putting out incorrect outputs
  - Fail safe
    - Its failure mode is biased so that the application process does not suffer catastrophe upon failure.

FAULT AND ERROR CONTAINMENT

- Fault containment zone (FCZ)
  - Failure in some part of the computer outside an FCZ cannot cause any element inside that FCZ to fail
  - Separate power supply units and separate clock inside and outside the FCZ.
- Error containment Zone (ECZ)
  - Prevent errors from propagating across zone boundaries
  - Hardware redundancy
    - Additional hardware
  - Software redundancy
    - Many versions of software
  - Time redundancy
    - Tasks can rerun if necessary
  - Information redundancy
    - Error detection and correction

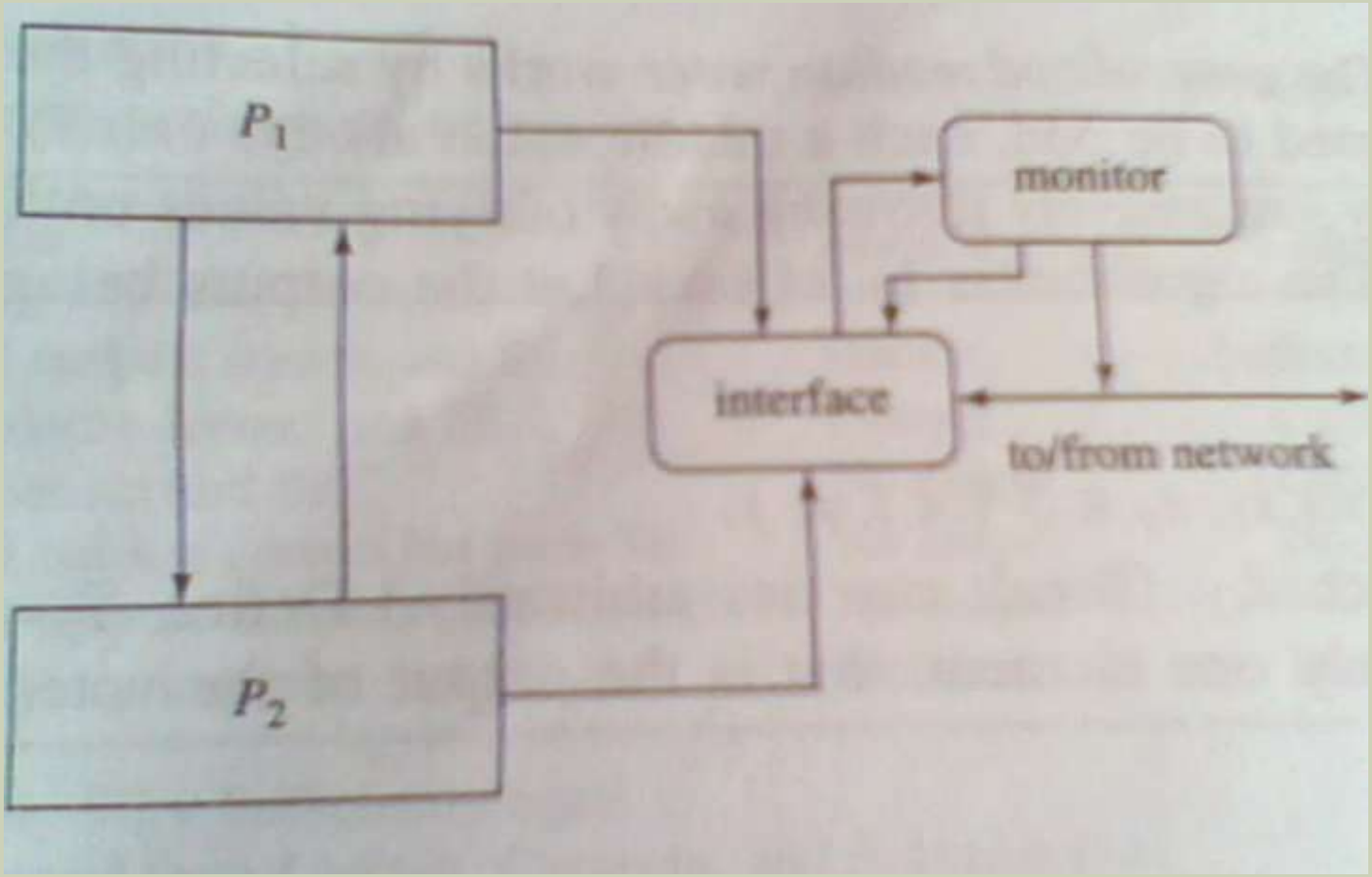
HARDWARE REDUNDANCY

- **Hardware Redundancy**
  - **Use of additional hardware to compensate for failures**
  - **This can be done in two ways**
  - Fault detection, correction and Masking. Multiple hardware units may be assigned to do the same task in parallel and their results compared. If one or more units are faulty, we can express this to show up as a disagreement in the results.
  - The second is to replace the malfunctioning units.
- Redundancy is expensive, duplicating or triplicating the hardware is justified only in most critical applications
- Two methods of hardware redundancy is given below are,
  - Static Pairing
  - N modular Redundancy (NMR)

HARDWARE REDUNDANCY

- **Static Pairing**
  - Hardwire processors in pairs and to discard the entire pair if one of the processors fails, this is very simple scheme
  - The Pairs runs identical software with identical inputs and should generate identical outputs. If the output is not identical, then the pair is non functional, so the entire pair is discarded
  - This approach is depicted in the following figure, and it will work only when the interface is working fine and both the processors do not fail identically and around the same time
  - So the interface is monitored by means of a monitor which monitors the interface. If the interface fails, the monitor takes care and if the monitor fails, the interface takes care. If both interface and monitor fails, then the system is down. The monitor block is added as a dotted box in the figure

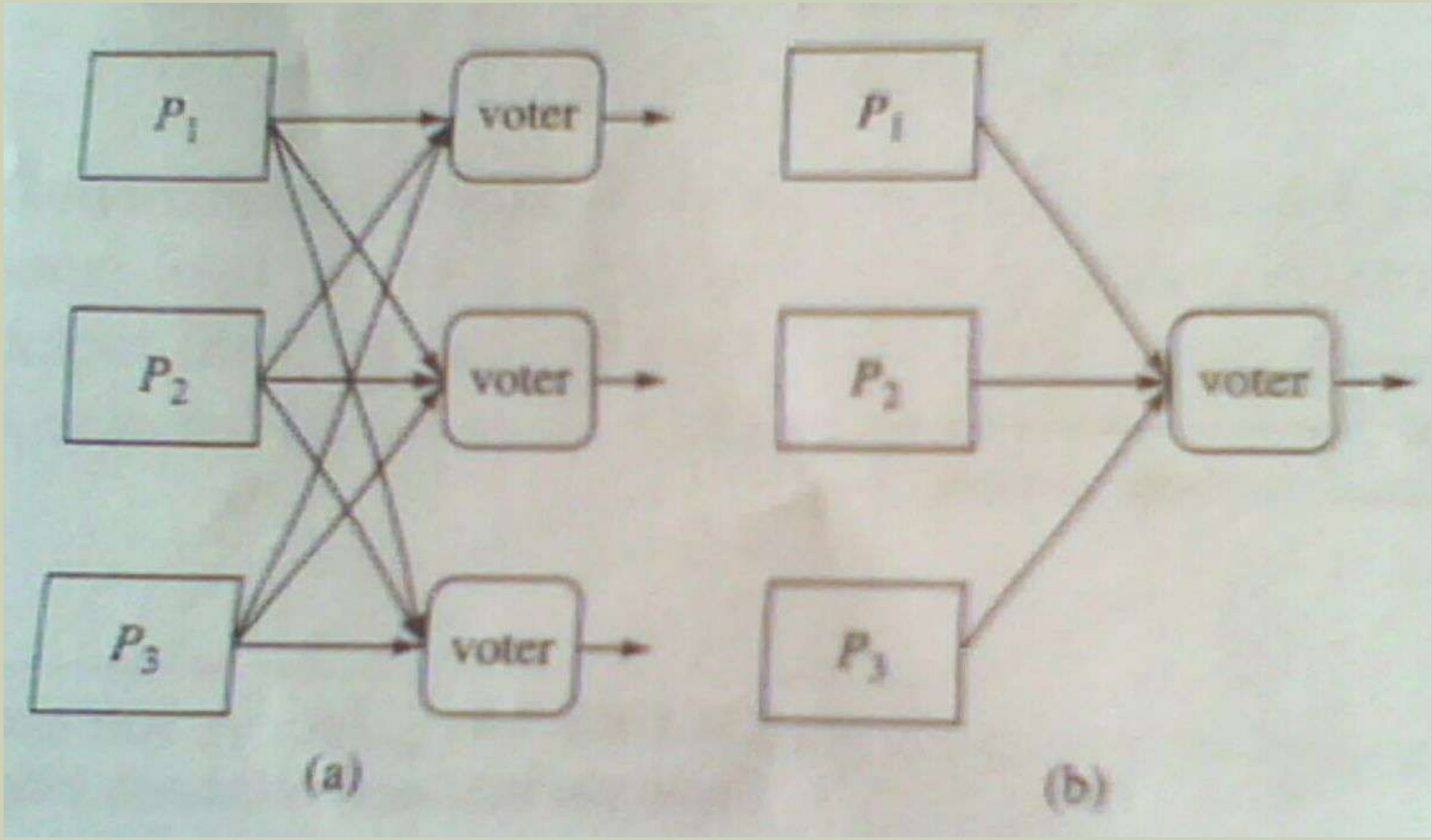
HARDWARE REDUNDANCY



HARDWARE REDUNDANCY

- **N Modular Redundancy**
  - It is a scheme for Forward Error Recovery.
  - It works with N processors instead of one and voting on their output and N is usually odd.
  - NMR can be illustrated by means of the following two ways
    - There are N voters and the entire cluster produces N outputs
    - There is just one voter
- NMR clusters are designed to allow the purging of malfunctioning units. That is, when a failure is detected, the failed unit is checked to see whether or not the failure is transient. If it is not, it must be electrically isolated from the rest of the cluster and a replacement unit is switched on. The faster the unit is replaced, the more reliable the cluster.

HARDWARE REDUNDANCY

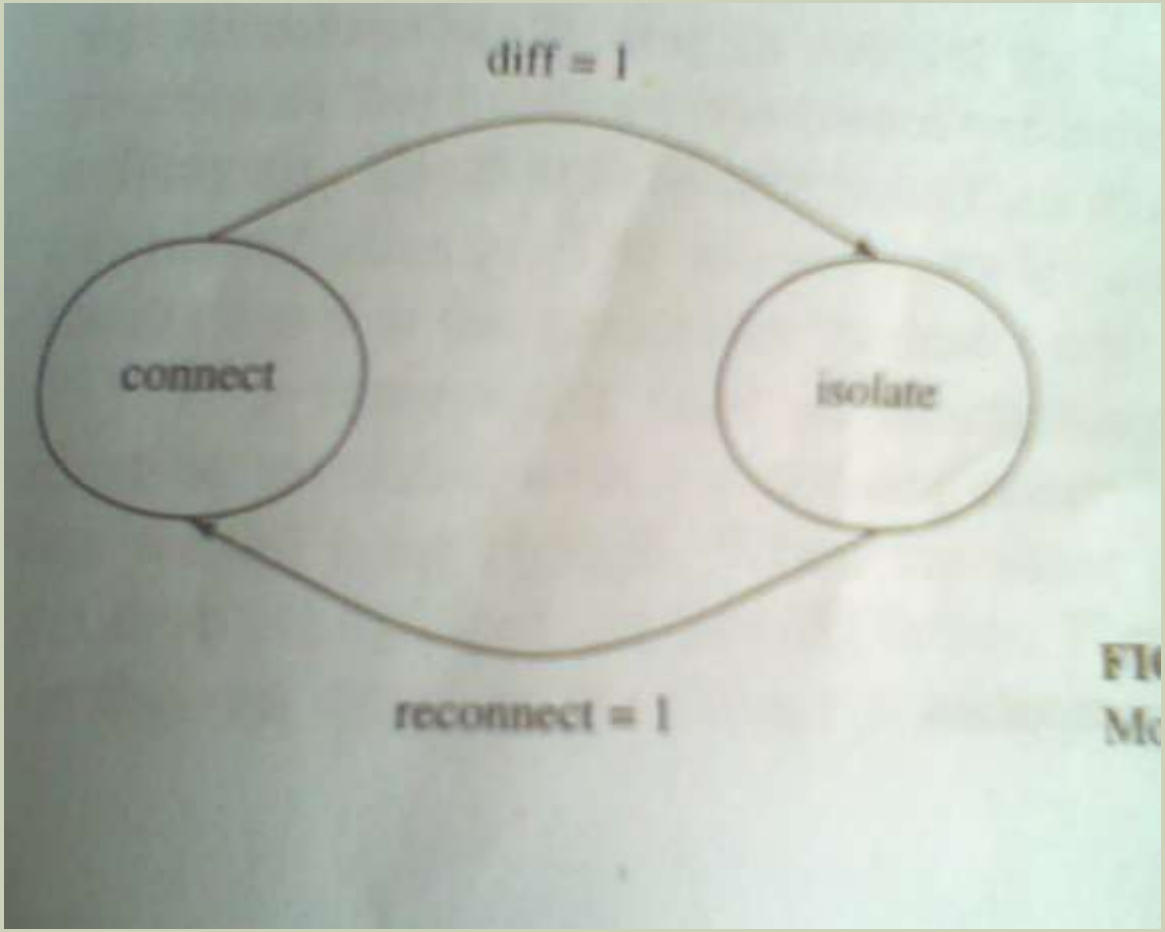


HARDWARE REDUNDANCY

- Purging can be done either by hardware or by the operating system.
- Self purging consists of a monitor at each unit comparing its output against the voted output. If there is a difference, the monitor disconnects the unit from the system.

- The monitor can be described as a finite state machine with two states connect and isolate. There are two signals, diff which is set to 1 whenever the module output disagrees with the voter output and reconnect, which is a command from the system to reconnect the module.

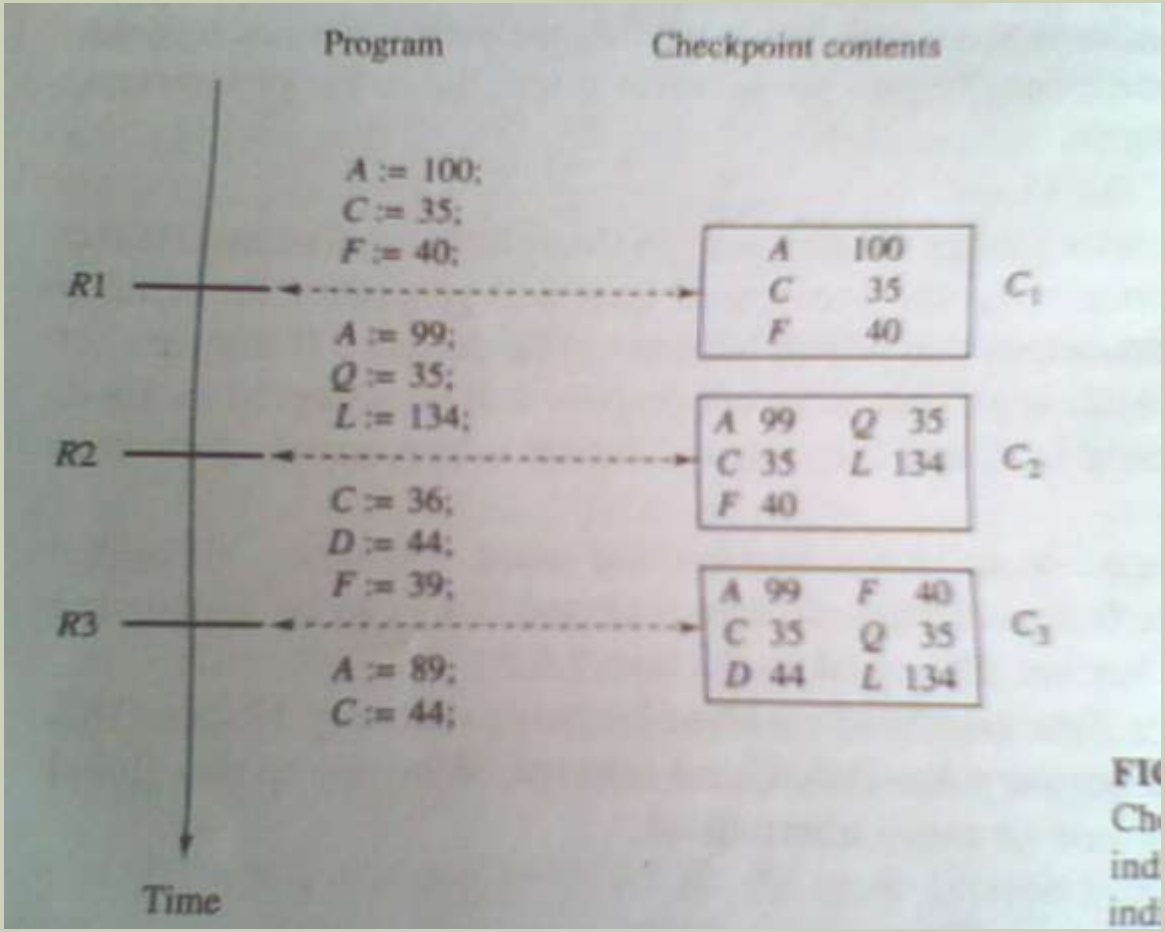
HARDWARE REDUNDANCY



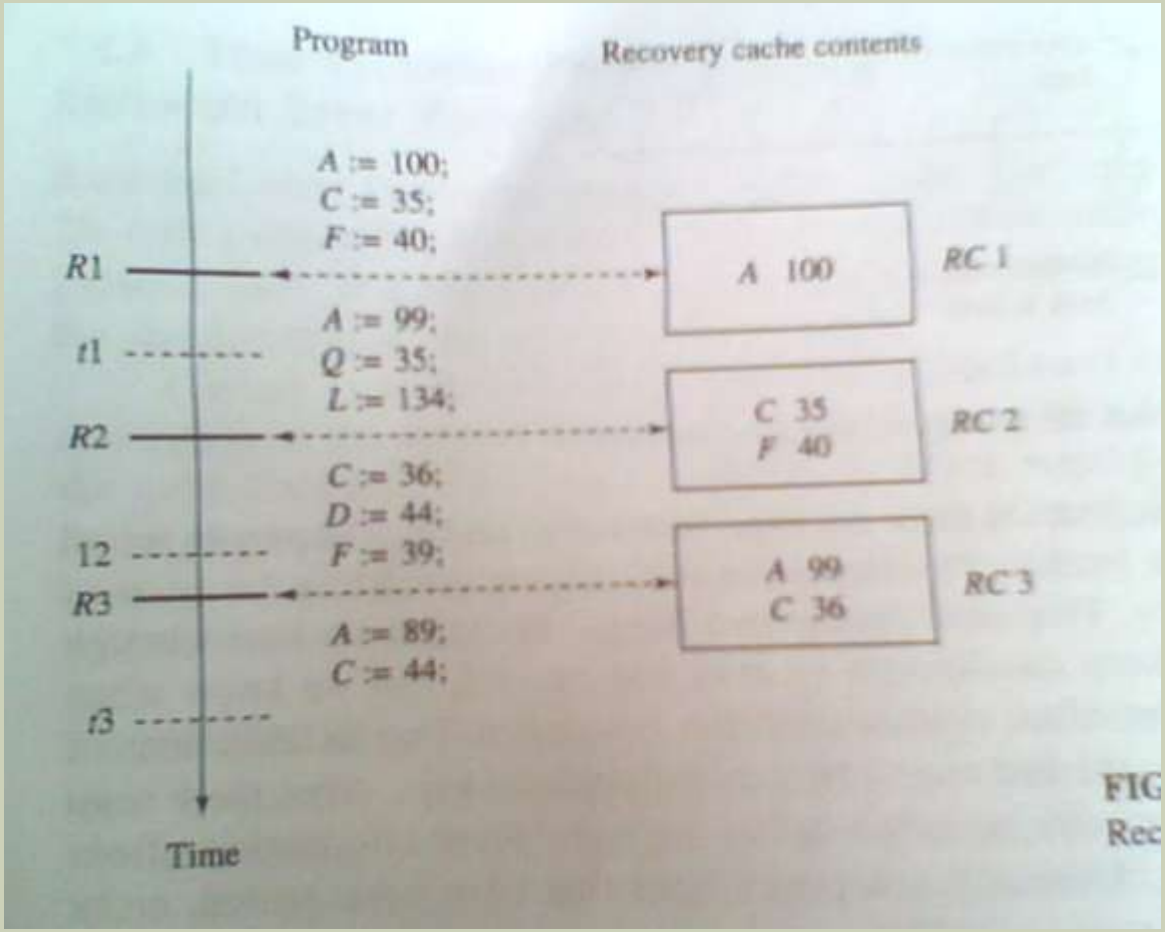
TIME REDUNDANCY

- Backward Error Recovery Scheme
  - Retry
  - Checkpoints
  - Recovery cache

TIME REDUNDANCY



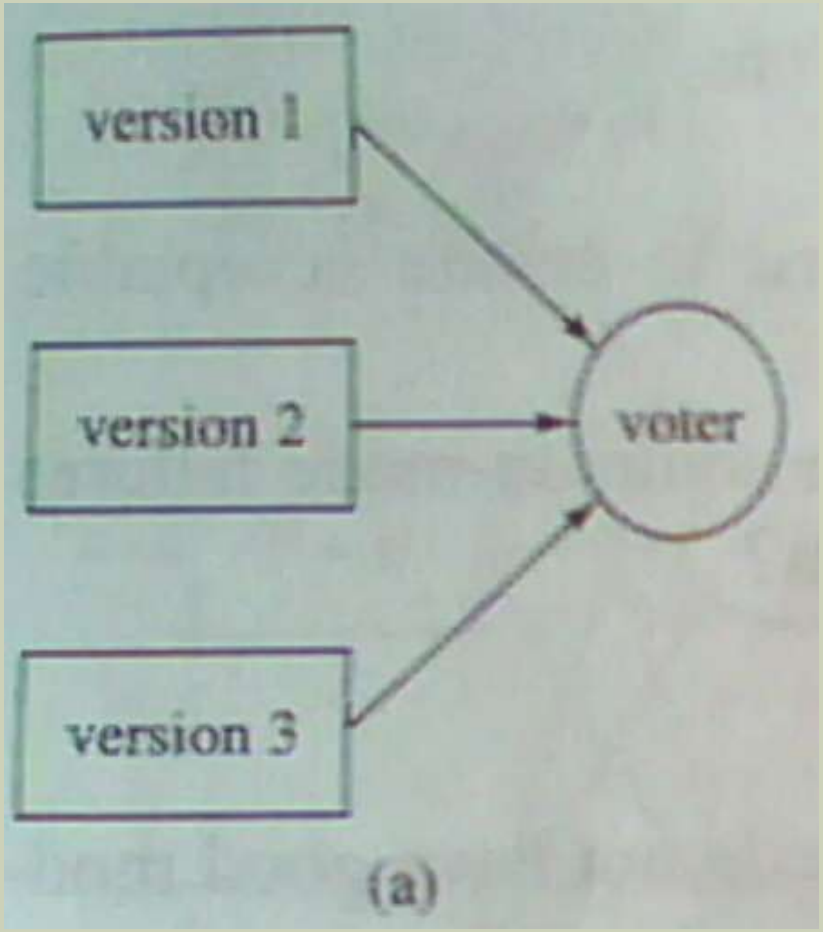
TIME REDUNDANCY



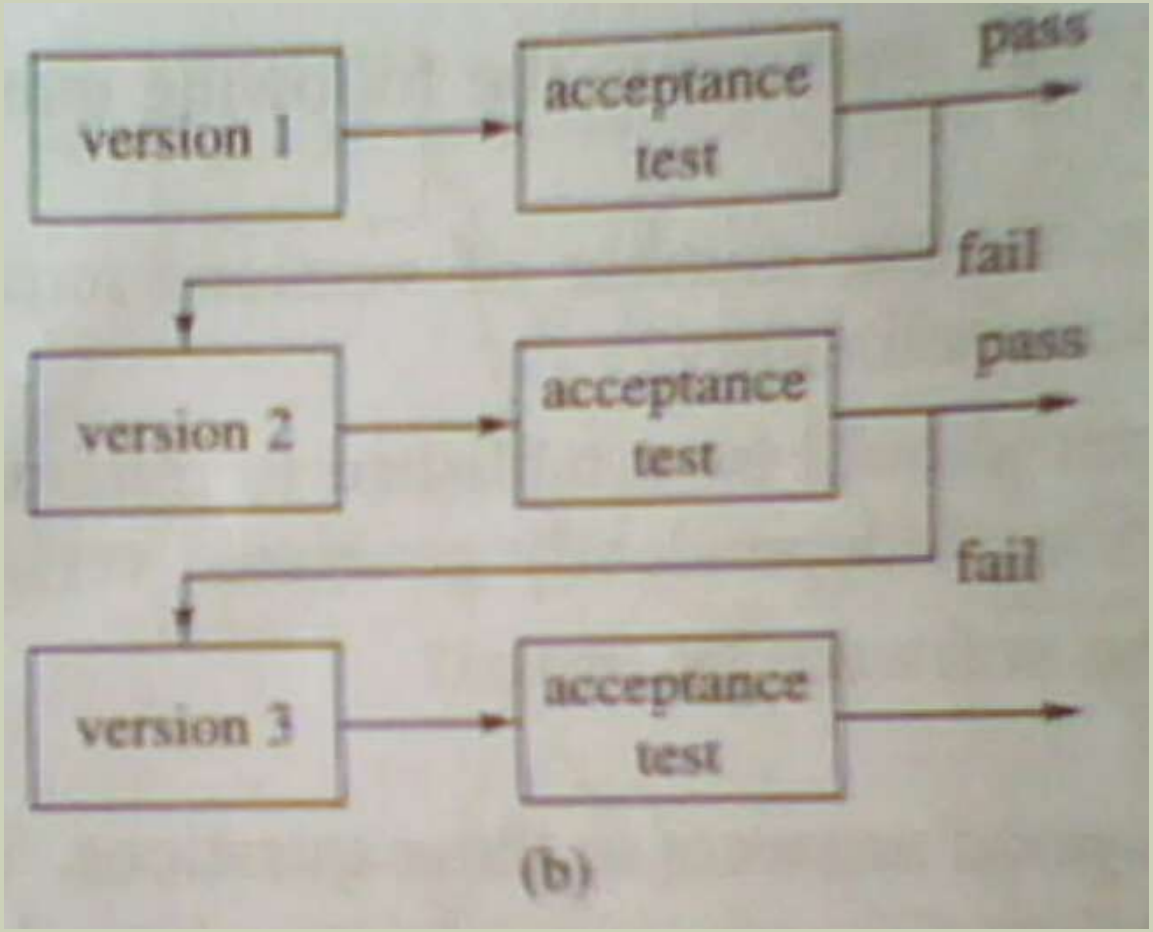
SOFTWARE REDUNDANCY

- Software faults are occurring mainly in the design
- Replicating the same software N times will not solve the purpose as the software fails for the same set of inputs
- But N versions of software can be running so that the probability of fault is less
- There are two approaches for that
  - N Version Programming
  - Recovery Block Approach

SOFTWARE REDUNDANCY



SOFTWARE REDUNDANCY



SOFTWARE REDUNDANCY

- Each version is being developed by a team of developers who never communicated with each other
- Common mode failures will be minimized
- To minimize the common mode failures
  - Write the specifications in very formal terms and subject them to a rigorous process of checking
  - Diversity of multiple software versions in different programming languages
  - Numerical algorithms used
  - Nature of tools that are being used.
  - Training and quality of the programmers

INFORMATION REDUNDANCY

- Parity codes
  - Ordinary parity bit
  - Interlaced parity bit
- Checksum codes
  - Single precision
  - Double precision
  - Honeywell
- Arithmetic codes
  - AN Code