

# Self-Test: Processors

# Processor Hardware Self-Test

Typically, a self-test program for checkout or restarting is a boot-strapping procedure which begins with the verification of the most elementary set of instructions, i.e., those which rely on only a fraction of the computer hardware in order to operate. These instructions are then used to construct a decision-making subroutine which verifies some primitive condition on a YES-NO basis. Once verified, this subroutine (or several similarly constructed) is used to check all other instructions and variations in sequence, beginning with the next least complex instruction and working up to the most complex instruction. After all instructions are verified, input/output (I/O) and memory self-test programs check the remaining hardware.

# Processor Hardware Self-Test

## Case Study: Gemini

Self-test routines are also important for detecting malfunctions during operation. In the Gemini project, for example, diagnostic subroutines were interleaved in the operational computer program. When they detected a fault, a discrete command was issued to light a malfunction indicator lamp on the control panel. The circuit had a manual reset capability to test whether it was set by a transient malfunction.

# Processor Hardware Self-Test

## Case Study: Gemini (cont'd)

Three self checks were performed during flight:

- A timing check, based on the noncoincidence of certain signals within the computer under proper timing conditions.
- A thorough diagnostic test which exercised all of the computer's arithmetic operations during each computer cycle in all modes.
- A looping-check, to verify that the computer was following a normal program loop. A counter in the output processor was designed to overflow every 2.75 sec. Each program was written to reset this counter every 2.7 sec; thus, any change in the program flow would cause an overflow and indicate a malfunction.

# Processor Hardware Self-Test

## Case Study: Apollo Guidance Computer

The Apollo guidance computer is equipped with a restart feature comprising alarms to detect malfunction and a standard initiation sequence which leads back into the programs in progress. The AGC has six malfunction detection devices that cause a restart, as follows:

- A parity test of each word read from memory. An odd-parity bit is added to each fixed- memory word at manufacture time and to each erasable word at write time.
- A looping check much like the one on Gemini. A specified register must be periodically tested by any correctly operating program. This register is "wired" and if it is not tested often enough will cause restart.

# Processor Hardware Self-Test

## Case Study: Apollo Guidance Computer

- A transfer control trap, which detects endless loops containing only control transfer instructions, such as a location L which contains the instruction "transfer control to location L."
- An oscillator fail check caused by stopping of the timing oscillator.
- Voltage fail circuits to monitor the 28-, 14-, and 4-V power levels which drive the computer.
- An interrupt check, which detects excessive time spent in the interrupt mode, or too much time spent between interrupts.

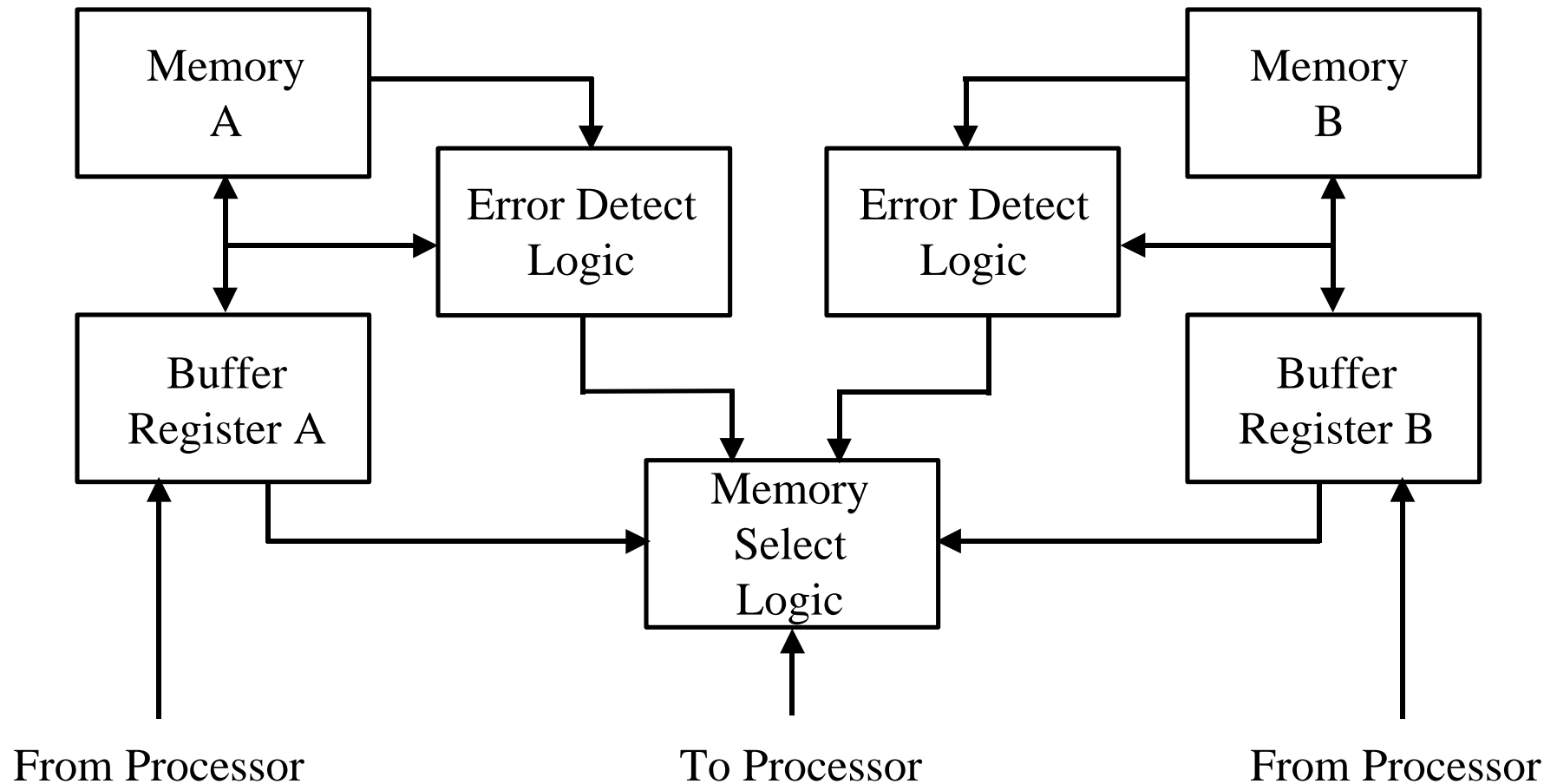
# Processor Hardware Self-Test

## Case Study: Saturn V Launch Vehicle

- Logic used TMR
  - Disagreement detector for faults
  - **Switch to simplex if fault? Verify.**
  - Memory was dual-redundant with parity
  - Both memories read in parallel
  - If fault, then backup memory read, correct data written to both memories (DRO core)
  - Switch prime and backup units

# Processor Hardware Self-Test

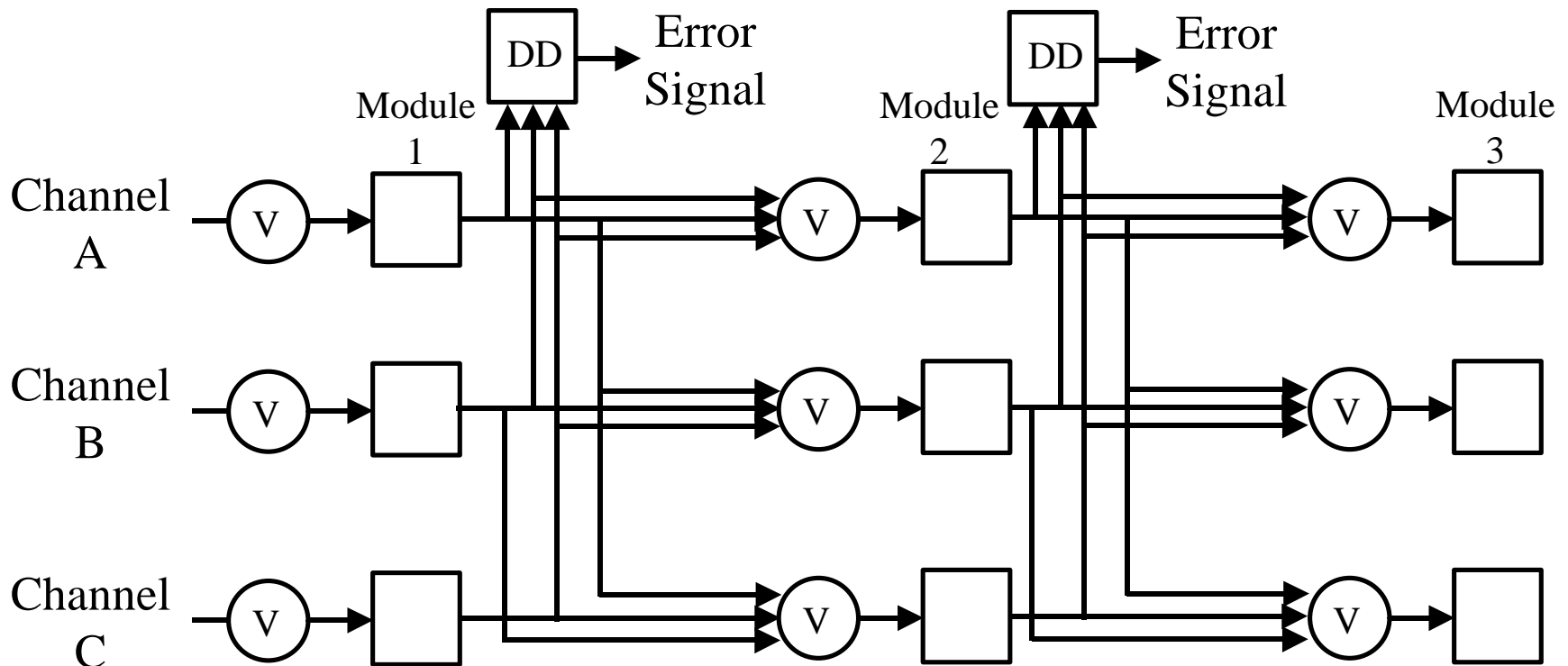
## Case Study: Saturn V Launch Vehicle



Saturn V LVDC Duplex Memory Diagram  
Self-Correcting Duplex Logic

# Processor Hardware Self-Test

## Case Study: Saturn V Launch Vehicle



V ::= Voter  
DD ::= Disagreement Detector

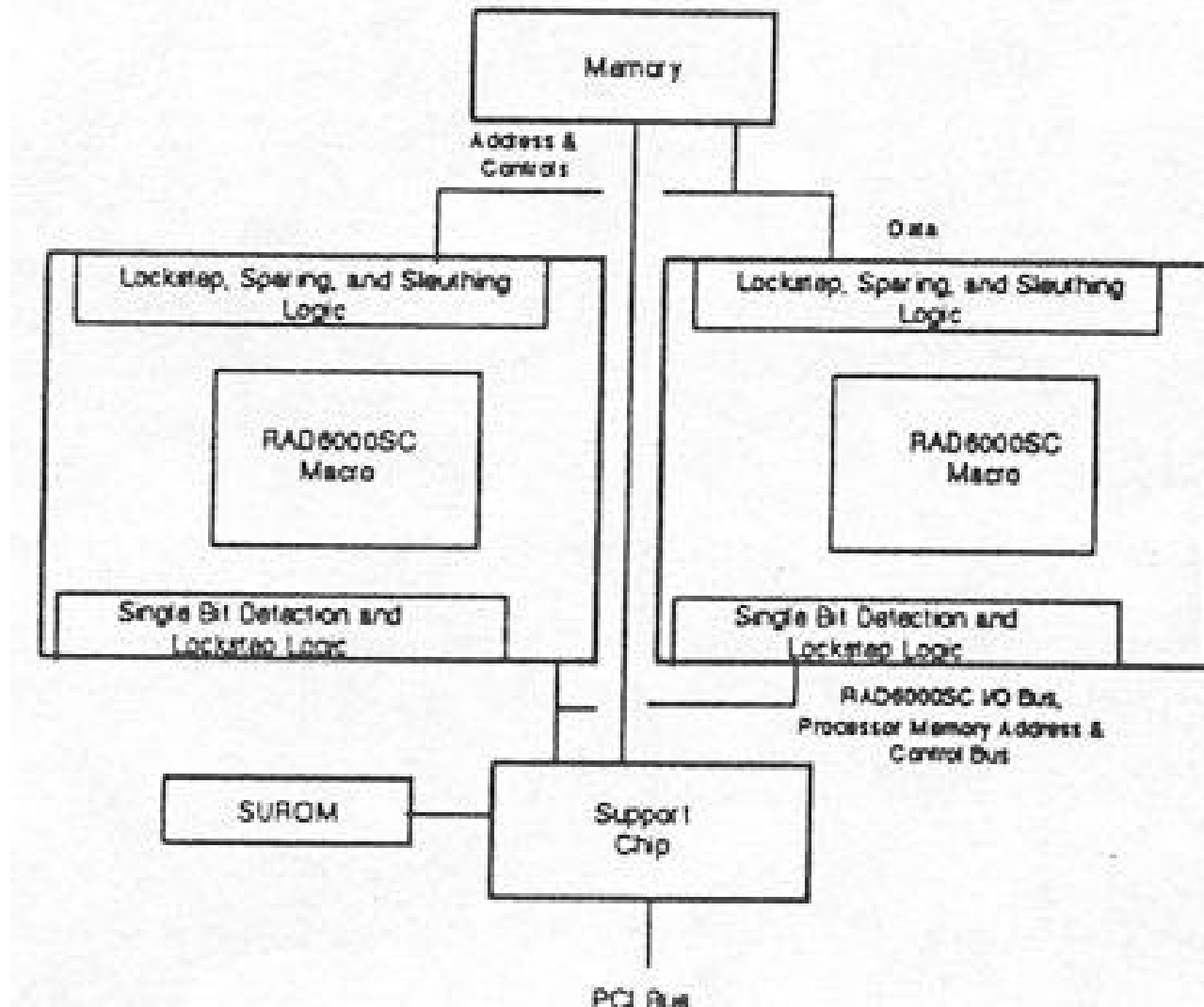
Saturn V LVDC TMR Logic

# Processor Hardware Self-Test

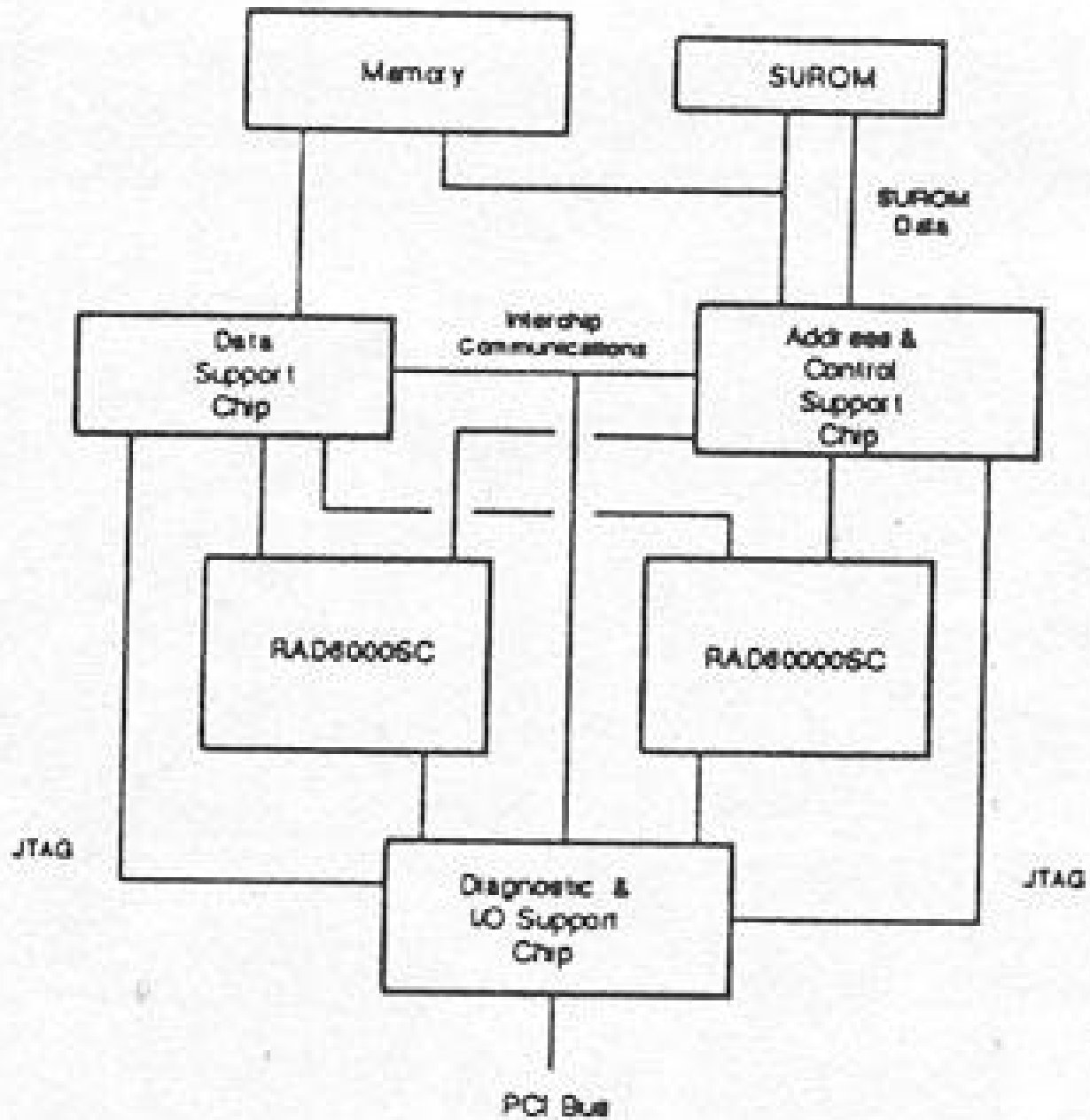
## Case Study: Space Shuttle

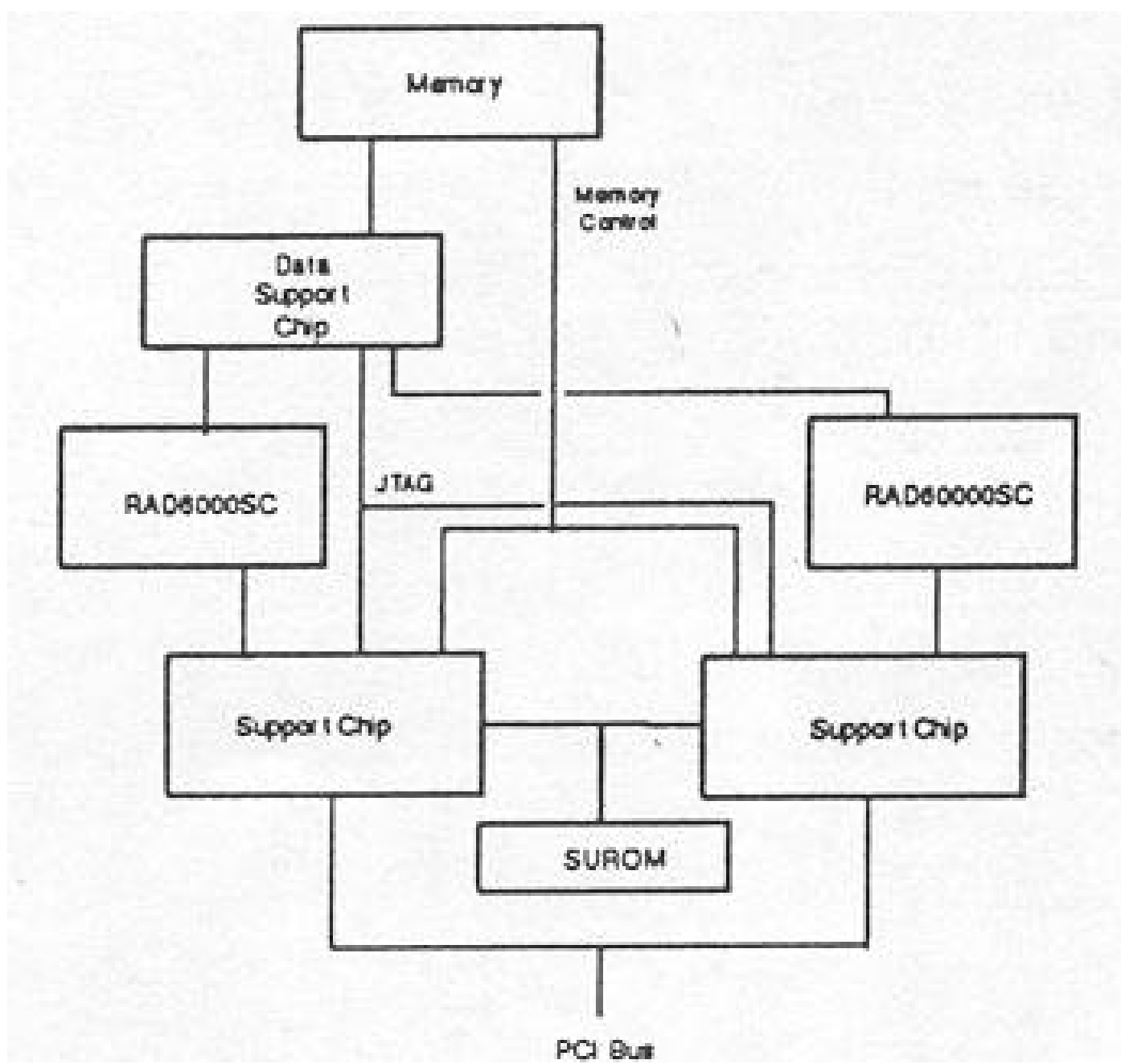
- 4 of the 5 identical computers operate in an NMR configuration
  - Computers synchronized and outputs between computers are compared on the I/O busses
- Voting at the actuator
  - hydraulic voting mechanism: force-fight voter
- After two failures, operates as a duplex system with comparison and self-test techniques

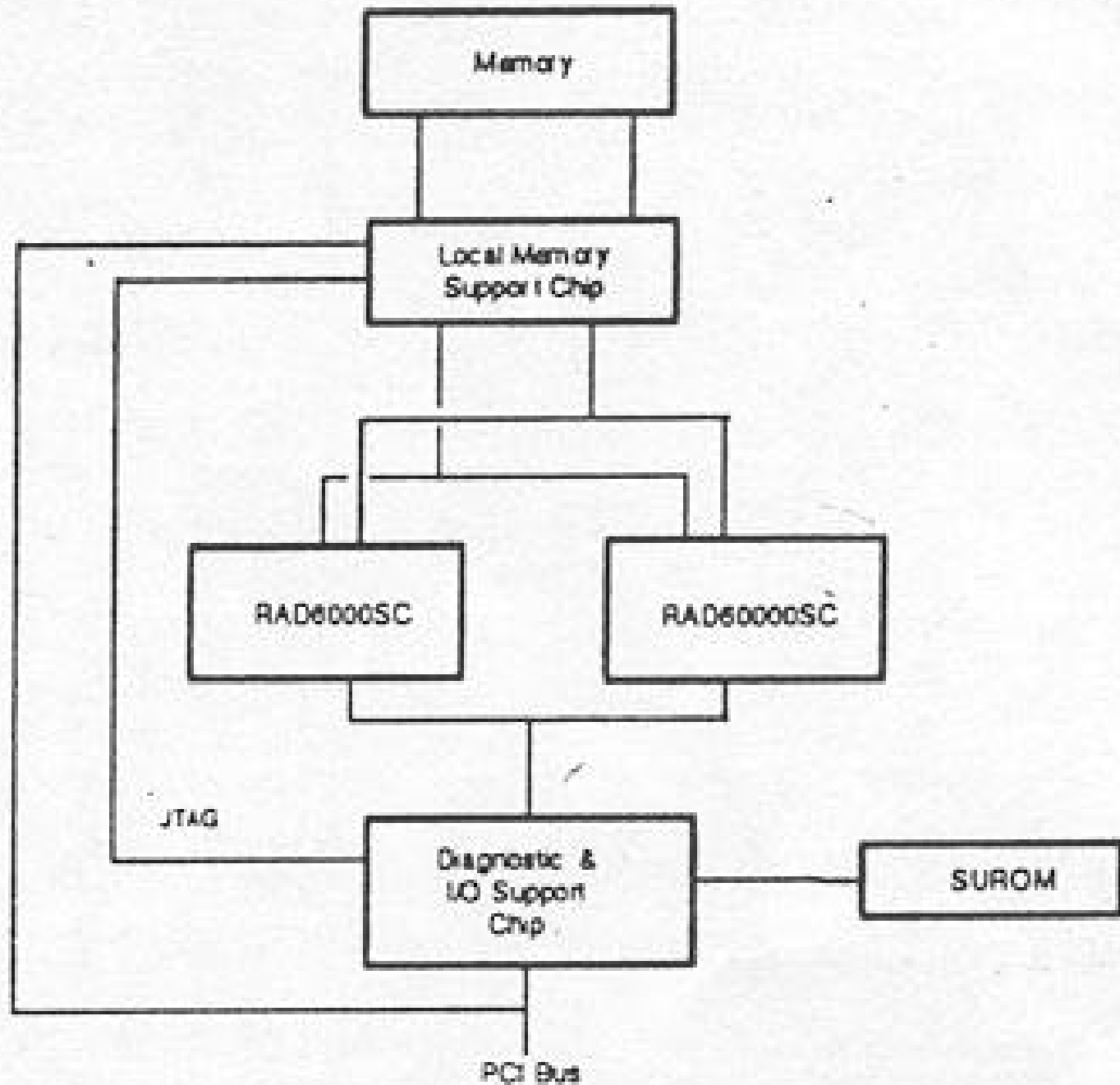
# Case Study: Lockstep Operation



“Embedding COTS Processors into Fault Tolerant Space Applications,” J. Marshall, Jr. and D. Langston, Paper AIAA-95-1032-CP, A Collection of Technical Papers, AIAA Computing in Aerospace 10, March 28-30, 1995, San Antonio, TX







# Processor Hardware Self-Test

## Case Study: MA31750/MIL-STD-1750A

- On-chip parity generation/checking
- Built-In test
  - Part of initialization
  - Manufacturer defined XIO Instruction
    - Code  $840D_{16}$
  - For Tracor RHEC and MAS281
    - BIT part of initialization
    - Called using Built-In Function (BIF) 4F

# Processor Hardware Self-Test

## Case Study: MA31750/MIL-STD-1750A

### Built-In Test (BIT) Coverage

- Temporary Registers (T0-T11)
- General Registers (R0-R15)
- Flags Block
- Sequencer Operation and ROM Checksum
- Divide Routine Quotient Shift Network
- Multiplier and ALU
- Barrel Shift Network
- Interrupts and Fault Handling and Detection
- Address Generator Block
- Instruction Pipeline

# Processor Hardware Self-Test

## Case Study: MAS281/MIL-STD-1750A

### Built-In Test (BIT) Coverage

- Microcode sequencer; IB Register Control; Barrel Shifter; Byte Operations and Flags
- Temporary Registers (T0-T7); Microcode Flags; Multiply; Divide
- Interrupt Unit - MK, PI, FT; Enable/Disable Interrupts
- Status Word Control; User Flags; General Registers (R0-R15)
- Timer A; Timer B

# Hardware Self-Check

## Case Study: IA-64

- L2 and L3 are ECC protected
  - L2 is on-chip, 96 kB unified, 6-way set associated, 64-byte line
  - L3 is on-cartridge, up to 4 MB, 4-way set associated, 64-byte line
- “The processor implements a machine check architecture (MCA) that provides the ability to continue, Recover, or Contain detected errors. All significant structures on the chip are protected by parity or ECC.”

“The First IA-64 Microprocessor,” S. Rusu and G. Singer, **IEEE Journal of Solid-state Circuits**, November, 2000.

# Hardware Self-Test

## Case Study: MIL-STD-1553B

- Mode Code 00011 - Initiate Self-test
- Terminal fail-safe. Hardware ensures that no transmission is greater than 800.0  $\mu\text{s}$  (4.4.1.3)
- Listening to the transmitted signal to ensure it matches what was sent.
  - (Look up to see if 1553 requirement or implementation)

# Galileo: Mission to Jupiter

Several options were studied and design although not implemented in the flight hardware.

- ATAC-16MS/Alien board plan to combat SEU sensitivity.
  - Duplicate electronics with comparison logic
  - Flight solution was to replace SEU-soft components with custom-designed hardened replacement components.
- TMR for the memory unit
  - RAM array
  - 54LSxx flip-flops