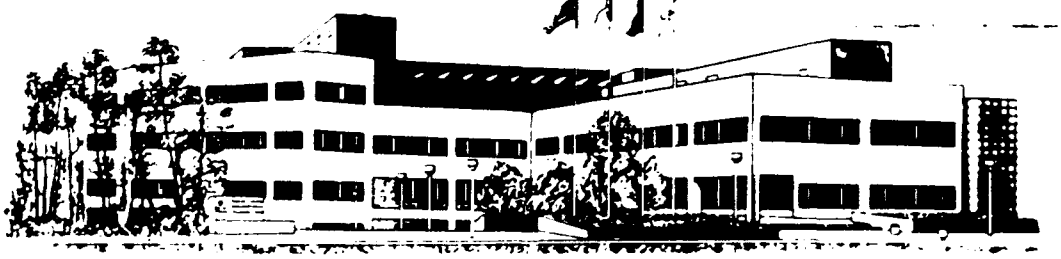
U.S Department of Transportation
Federal Aviation Administration

DOT/FAA/CT-88/10

DIGITAL SYSTEMS VALIDATION HANDBOOK-VOLUME II

FEBRUARY 1989

DTFA03-86-C-00042



FEDERAL AVIATION ADMINISTRATION
TECHNICAL CENTER

U.S. Department
of Transportation
**Federal Aviation
Administration**

Technical Center

Atlantic City Int'l Airport
New Jersey 08405

October 17, 1990

Dear Colleague,

Enclosed is additional material for the Digital Systems Validation Handbook - Volume II. These additions consist of a copy of Chapter 5 - Advanced Fault Insertion, and Chapter 15 - Electromechanical Actuator Systems. Please add these two chapters to your Handbook. In addition, there is a revised Table of Contents for the Handbook, as well as replacements for the existing Glossary and Acronyms lists. Please exchange these two sections.

If you have any questions or comments concerning this material, or if you would like to receive a copy of the Handbook, please feel free to contact me at the FAA Technical Center.

Sincerely,

Pete Saraceni

Pete Saraceni
Program Manager
FAA Technical Center
ACD-230, Bldg. 201
Atlantic City International Airport, NJ 08405

Phone: (FTS) 482-5577
(609) 484-5577

Enclosures: Chapter 5 - Advanced Fault Insertion
Chapter 15 - Electromechanical Actuator Systems
Table of Contents (Revised)
Glossary and Acronyms (Revised)

TABLE OF CONTENTS

DIGITAL SYSTEMS VALIDATION HANDBOOK - VOLUME II

TABLE OF CONTENTS

TABLE OF CONTENTS (Continued)

GLOSSARY
and
ACRONYMS

GLOSSARY

α-FAULT. (10) A fault activated by the baseline program (see ß-FAULT).

ß-FAULT. (10) A fault not activated by the baseline program (see α-FAULT).

A-SPECIFICATION. (9) The highest level specification typically produced by the contracting organization to define a system (see MIL-STD-1521).

ABSORPTION LOSS. (11) Attenuation or retention of electromagnetic energy passing through a material, a shield. Absorption loss and reflection loss contribute to total shielding effectiveness (SE).

ACTION INTEGRAL. (13) The action integral is a critical factor in the production of damage. It relates to the energy deposited or absorbed in a system. This energy cannot be defined without knowing the resistance of the system. The instantaneous power dissipated in a resistor is I^2R and is expressed in watts. For the total energy expended, the power must be integrated over time to get the total joules, watt-seconds. By specifying the integral of $i(t)^2$ over the time interval involved, a useful quantity is defined for application to any resistance value. In the case of lightning, this quantity is defined as the action integral and is specified as $i(t)^2dt$ over the time the current flows.

ACTIVE FAULT. (10) A fault that can produce an error (for some input) while executing the current program.

ACTUAL TRANSIENT LEVEL. (13) The actual transient level is the level of transients which actually appear at the system interfaces as a result of the external environment. This level may be less than or equal to the transient control level but should not be greater.

ADDRESSING CAPACITY. (6) The number of components addressable by the protocol used on a given data bus.

AIRCRAFT LIGHTNING INTERACTION. (13) An encounter with lightning that produces sufficient current within or voltages along an aircraft skin or structure to pose a threat to the aircraft electrical/electronic systems, as a result of a direct lightning attachment.

AMBIENT. (16) The substance which absorbs heat from the heat sink.

ANALYTICAL REDUNDANCY. (7) The use of software algorithms which use known mathematical relationships between different sensors for sensor failure detection and replace most of additional redundant sensor hardware.

ANALYTICAL ROOT SOLUTION. (4) Information obtained from the roots of the characteristic equations of the airplane model such as short-period or phugoid frequency response.

ANGLE OF ATTACK. (4) Angle between the longitudinal axis of an aircraft and the direction of movement.

ANODIZE. (11) A preparation by electrolytic process that deposits a protective oxide, insulating film on a metallic surface (aluminum). The oxide defeats electrical bonding. Alodine and iridite finishes on aluminum are conductive.

APERTURE. (11) An opening, such as a nonconductive panel joint, slot, or crack, allowing electromagnetic energy to pass through a shield.

ARTIFICIAL INTELLIGENCE. (16) The characteristics of a machine programmed to imitate human intelligence functions.

ASSURANCE ASSESSMENT. (4) Procedures whose purpose is to ensure that a proposed system functions according to design specifications.

ASYNCHRONOUS MESSAGES. (6) Electronic signals with transmission times that are not known a priori. These may include priority signals requiring immediate access to the bus.

ATTACHMENT POINT. (13) A point of contact of the lightning flash with the aircraft.

AUDIO FREQUENCY (AF). (11) The spectrum (20 to 20,000 Hz) of human hearing, often defined as extending from approximately 20 Hz to 50 kHz and sometimes to 150 kHz. Audio noise is nuisance hum, static, or tones from power line 400 Hz, switching regulator and digital clock harmonics, or HF, VHF transmitter frequencies.

AUTOFEATHER. (16) To automatically and swiftly feather the propeller when the engine fails to drive it.

AUXILIARY PROGRAMS. (10) Software executed occasionally.

AVALANCHING LATENT FAULTS. (10) The successive activation of latent faults.

BACKSHELL. (11) Metal shell connecting circuit shields or overbraid to an electrical connector.

BACKWARD RECOVERY. (9) Restoration of the system to some previous known correct state and restarting the computation from that point.

BALANCED CIRCUIT. (11) A signal, acting line-to-line, between two conductors having symmetrical voltages identical and equal in relation to other circuits and to ground. "Differential mode" is line-to-line; "common mode" is line to ground.

BANDWIDTH (BW). (11) Frequencies bounded by an upper and lower limit in a given band associated with electronic devices, filters, and receivers.

BASELINE PROGRAM. (10) A set of continuously executed software modules.

BENIGN FAULT. (10) A fault that cannot produce an error while executing the current program, regardless of input, but may produce an error for some other program.

BIT TIME. (6) The time it would take to transmit one bit. Usually this is "blank" time when nothing is being transmitted. One nth of the bus speed (i.e., on a 1 kHz bus, the bit time is 10^{-3} seconds).

BLOCK TRANSFER. (6) A data transfer mode allowing the transfer of variable length data blocks.

BOND, ELECTRICAL. (11) Electrical connection at two metallic surfaces securely joined to assure good conductivity often 2.5-mΩ maximum for electrical/electronic units and 1Ω for electrostatic dissipation or safety. A "faying surface" bond maintains contact between relatively large or long surfaces. Inherently bonded parts are permanently assembled and conductivity exists without special preparation: such as with welding, brazing.

BRAID, OVERBRAID. (11) Fine metallic conductors woven to form a flexible conduit or cableway and installed around insulated wires to provide protection against electric fields and radio frequencies. Best when peripherally connected to backshells. A grounding strap/jumper may be made of braid.

BROADBAND. (12) A frequency spectrum which is wide compared to the bandwidth of the device used to detect it.

BROADCAST CAPABILITY. (6) The capacity to transmit messages to all terminals simultaneously.

BROADCAST. (4) Transmission of messages to all terminals without reference to the identification of the receiving station or terminal.

BYZANTINE RESILIENCE. (5) A fault tolerant process which is tolerant of intermittent faults that can send good information part of the time.

CABLE OR HARNESS. (11) A bundle of separate, insulated, electrical circuits, shielded or unshielded, usually long and flexible and having breakouts, terminations, overbraid, and mounting provisions completely assembled.

CABLEWAY. (11) A solid metallic housing (liner, foil, coating) surrounding and shielding insulated electrical conductors. Also called conduit, tray, or raceway. Crosswise or transverse openings or breaks in the metallic cableway cause noise voltages to be transferred to internal wire circuits.

CANARD. (16) A tail-first aerodyne, usually with auxiliary horizontal surface at the front and a vertical surface at the back.

CAT IIIa LANDING. (6) One of several landing categories defined in FAR 91. CAT IIIa implies the need for an instrument landing approach.

CENTRAL CONTROL. (6) Control from one master, whether stationary or non-stationary.

HARGE TRANSFER. (13) The integral of the current over its entire duration, (t)dt, in coulombs.

HORD. (4) The straight line segment intersecting or touching an airfoil rofile at two points.

OMMAND/RESPONSE. (6) "Operation of a data bus system such that remote terinals receive and transmit data only when commanded to do so by the controller." MIL-STD-1553 Designer's Guide, 1983, p. II-3.)

OMMON MODE (CM) IMPEDANCE. (11) Impedance or resistance shared by two or more ircuits so that noise voltages/currents generated by one are impressed on the thers.

OMMON MODE REJECTION. (11) The ability of wiring or an electronic device to eject common mode (line-to-ground) signals and maintain fidelity of differential ode (line-to-line) signals.

OMMON MODE SIGNAL. (11) Identical and equal signals on input conductors or t the terminals of a device relative to ground.

OMPONENT DAMAGE. (13) Condition arising when the electrical characteristics f a circuit component are permanently altered beyond its specifications.

ONDUCTED EMISSION (CE) OR INTERFERENCE. (11) Voltage/current noise signals ntering or leaving a unit on interface conductors. Emission is the general erm, interference is undesired noise.

ONTENT ADDRESSING. (6) The system of identifying message recipients based on nformation embedded in the message. This is in contrast to destination terminal ddresses.

ONTROL LAW. (7) The physical relationship between various sensors and control urfaces.

ORONA. (13) A luminous discharge that occurs as a result of an electrical otential difference between the aircraft and the surrounding atmosphere.

OUPLING. (11) The transfer of energy between wires or components of a circuit lectrostatically, electromagnetically, or directly.

OVERAGE. (5) The conditional probability of the system successfully recovering rom a component fault and continuing to perform the intended functions orrectly, given the presence of the fault. Coverage is the measure of ffectiveness of a system's utilization of redundant hardware. Coverage can be ualified and applied to many different components of a system and phases of ecovery process. Examples include, fault detection coverage, fault isolation overage, latent fault coverage, sensor failure coverage, and memory failure overage.

COVERAGE. (7) The percent confidence level of a given analytical redundancy fault detection and isolation algorithm for all types of faults.

COVERAGE. (9) The probability that when a fault occurs, it will be detected and recovery from the fault will be successful.

CRITICAL. (13) Functions whose failure would contribute to or cause a failure condition which would prevent the continued safe flight and landing of the aircraft.

CROSS COUPLING (CROSSTALK). (11) Transfer of signals from one channel, circuit, or conductor to another as an undesired or nuisance signal or the resulting noise.

DAMAGE. (11) The irreversible failure of a component.

DATA BUS. (6) A system for transferring data between discrete pieces of equipment in the same complex.

DATA LATENCY. (6) The delay from the time when a piece of information becomes available at a source terminal to the time it is received at the destination.

DATA LINK ASSURANCE OF RECEIPT. (6) The guarantee of good data through the data link level.

dBμV. (12) Decibels referred to one microvolt. Zero db represents one microvolt.

DECIBEL (dB). (11,12) Decibel expresses the ratio between two amounts of power, P1 and P2, at two separate points in a circuit. By definition, the number of dB = 10 log to the base 10 of (P1/P2). For special cases, when a standard power level P2 = 1 mW or 1 W or 1 kW, then the ratio is defined as "dBm," "dBw," or "dBKW." Because $P = V^2/R$ and also I^2R, decibels express voltage and current ratios. Ideally, the voltages and currents are measured at two points having identical impedances. By definition, dB = 20 log V1/V2 and dB = log I1/I2. For convenience, V2 or I2 are often chosen as 1 μV or 1 μA and the ratio is defined as dB above a μV or dB above a μA when graphing emission or susceptibility limits.

DECOUPLED MANEUVERS. (4) Changes in an aircraft's direction and attitude in one axis without affecting direction or attitude in other axes.

DESIGN ERROR. (4) A functional flaw resulting from a misinterpretation of the specifications of the system.

DESIGN MARGIN. (13) The difference between the equipment transient design levels and the transient control level.

DETERMINISTIC. (6) A system where all parameters are known, as opposed to a statistical system where the outcome is subject to the laws of probability.

DIAGNOSTIC FILTER. (7) An analytical algorithm which processes data from N functionally related sensors. The data are used to estimate some sensor outputs and assess the correct functioning of the sensors.

DIELECTRIC STRENGTH. (11) Voltage withstand capability that an insulating material sustains before destructive arcing and current flow, usually expressed in volts per mil thickness. Dielectric withstand voltage is the voltage level at which insulation breakdown occurs.

DIFFERENTIAL MODE (DM) SIGNAL. (11) The signal in a two-wire circuit measured from line-to-line.

DIRECT EFFECTS. (13) Any physical damage to the aircraft or onboard systems due to the direct attachment of the lightning channel. This includes tearing, bending, burning, vaporization, or blasting of aircraft surfaces or structures, and damage to electrical/electronic systems.

DISTRIBUTED CONTROL. (6) Concurrent control from multiple points in the data bus system.

DOUBLE FAIL-OPERATIONAL SYSTEM. (4) A quadruplex (or higher) redundant flight-control system which is designed to incur failures in two redundant lanes (or channels) before it fails.

DUAL FAIL-OPERATIONAL. (7) A reliability requirement placed on a system which requires the system to be operational after two failures have occurred.

DUAL GROUND. (11) Equipment case ground/return through two independent circuit paths to structure implemented in flammable zones and water leakage areas-each path meeting electrical conductivity (resistance) requirements.

DUAL-DUAL ARCHITECTURE. (4) Two parallel dual computers with a voting plane at the output of each dual computing lane.

ELECTRIC FIELD. (11) High-impedance, radiated voltage field positive or negative, from a voltage source as contrasted to a low-impedance magnetic field from a current source.

ELECTROMAGNETIC COMPATIBILITY (EMC). (11) Operation within performance specification in the intended electromagnetic interference environment.

ELECTROMAGNETIC INTERFERENCE (EMI). (11) Conducted and radiated voltage/current noise signals, broadband (BB) or narrow band (NB), that degrade the specified performance of equipment.

ELECTROMIGRATION. (5) Drifting of metal atoms toward the cathode of a cathode ray tube.

ELECTROSTATIC CHARGE. (11) Electric potential energy with a surrounding electric field, uniform or nonuniform, moving or at rest, on a material.

EMISSION. (11) Voltage/current noise on a wire or in space. Broadband emission has uniform spectral energy over a wide frequency range and can be identified by the response of a measuring receiver not varying when tuned over several receiver "bandwidths." Or, energy present over a bandwidth greater than the resolution bandwidth where individual spectral components cannot be resolved. Broadband (BB) may be of two types: (1) impulse and coherent varies 20 dB per decade of bandwidth and (2) random or statistical, varies 10 dB per decade. A narrow band (NB) emission or signal, sometimes called continuous wave, occurs at a discrete frequency and does not vary with bandwidth.

ENVELOPE LIMITING. (4) General or additional limits imposed on the structural, "g" limits, speed, attitude, etc. of the aircraft. In some cases, envelope limiting imposes additional constraints on the envelope that cannot be exceeded regardless of pilot inputs.

EQUIPMENT TRANSIENT DESIGN LEVEL. (13) The level of transients which the equipment is qualified to withstand.

EQUIPMENT TRANSIENT SUSCEPTIBILITY LEVEL. (13) The transient level which will result in damage or upset to the system components. This level will be greater than the equipment transient design level.

ERROR. (4) A mistake in specification, design, production, maintenance, or operation of a system causing undesirable performance.

ERROR. (8) A state of the system which (in the absence of any corrective action by the system) could lead to a failure that would not be attributed to any event subsequent to the error. (More accurately known as an erroneous state.)

EVENT, EXTREMELY IMPROBABLE. (4) An event with a probability of occurrence on the order of 10^{-9} or less.

EVENT, IMPROBABLE. (4) An event with a probability of occurrence on the order of 10^{-5} or less.

EVENT, PROBABLE. (4) An event with a probability of occurrence on the order of 10^{-5} or greater.

EXTERNAL ENVIRONMENT. (13) Characterization of the natural lightning environment with idealized waveforms for engineering purposes.

FAIL-OPERATIONAL. (7) A reliability requirement placed on a system which requires the system to be operational after a single failure has occurred.

FAIL-SAFE. (7) A reliability requirement placed on a system which requires that safe flight not be hindered even after a failure.

FAILURE, HARD. (5) Repeated use of the same input and initial conditions results in the same incorrect response.

FAILURE, HIDDEN. (4) A failure that is not manifested at the time of its occurrence.

FAILURE MECHANISM. (5) Any situation that could produce an error condition. Examples of failure mechanisms include metal migration, voltage overstress, and lack of air-conditioning.

FAILURE, PERMANENT. (5) Repeated use of the same input and initial conditions results in the same incorrect response.

FAILURE, SOFT. (5) Repeated use of the same input and initial conditions does not result in the same incorrect response.

FAILURE, TEMPORARY. (5) Repeated use of the same input and initial conditions does not result in the same incorrect response.

FAILURE, TRANSIENT. (5) Repeated use of the same input and initial conditions does not result in the same incorrect response.

FAILURE. (4) The inability of a system, subsystem, unit, or part to perform within specified limits.

FAILURE. (5) The deviation of system behavior from specifications (arithmetic failure, storage failure, flight control function failure.)

FAILURE. (8) The situation when the external behavior of a system does not conform to that prescribed by the system specification.

FALL-TIME. (12) The time required for pulse amplitude to go from a predefined magnitude to a given level.

FALSE ALARM. (7) The declaration of a fault by a fault detection monitor or algorithm when there is no fault.

FAULT AVOIDANCE. (9) The attempt to prevent any software faults in the final delivered product through disciplined software development practices, testing, and IV&V.

FAULT CONTAINMENT. (6,9) The capacity of a system to prohibit errors and/or failures from propagating from the source throughout the system.

FAULT CURRENT. (11) The maximum current (magnitude and duration) flowing through a fault point. This current is equal to the supply voltage divided by the dc resistance of power line leads, circuit breakers, and the current return line wire or structure.

FAULT DETECTION. (6) The capacity of a system to determine the occurrence of erroneous operation.

FAULT DETECTION. (7) The determination that a sensor is faulted by using a software algorithm.

FAULT, HARD. (4) A defect in the hardware or software of a digital control system that permanently affects some functional performance of the system.

FAULT INSERTION. (4) A testing technique used to obtain information about data latency and built-in test coverage of a digital flight-control system.

FAULT ISOLATION. (6) The capacity of a system to isolate a failure to the required level so it can reconfigure.

FAULT ISOLATION. (7) The determination that a particular sensor is faulted by using a software algorithm.

FAULT, LATENT. (5) A fault which has not yet caused a failure. (For example, a fault in a memory chip that is not being used for the foreground program or in this particular mode of the system is a latent fault.)

FAULT, SOFT. (4) A transient defect in the software of a digital flight-control system that can be overcome by error-correctable code or by recycling of power to the computer system.

FAULT, STUCK-AT. (5) A logic signal which remains at zero (S-A-0) or one (S-A-1).

FAULT TOLERANCE. (6,9) The capability to endure errors and/or failures without causing total system failure.

FAULT TOLERANCE. (7) Accommodation of sensor hardware faults based on some type of comparator scheme.

FAULT TOLERANT SYSTEM. (5) A system that continues to function although certain components may have faults.

FAULT TOLERANT. (4,9) Software which continues to operate satisfactorily in the presence of faults.

FAULT TREE ANALYSIS. (4) A top-down deductive analysis that identifies the conditions and functional failures necessary to cause a defined failure condition. The fault tree can be used to establish the probability of the ultimate failure condition occurring as a function of the estimated probabilities of contributory events.

FAULT. (4) An error in the operation of a system.

FAULT. (5) The phenomenological reason for a failure (open wire, stuck-at fault, design fault, etc.). In general, any condition preventing a digital component from correctly changing state when directed to change by input parameters. For electrical components there is a one-to-one correspondence between faults and failures. The situation is not so simple with digital circuits. For if the circuit is S-A-1, any input causing a one output will be correctly processed; a little like the stopped clock that is correct twice per day. For a processor having a million or so logic gates, it is not possible to test for all the combinations of input and output states.

FAULT. (8) The adjusted cause of error.

FILTER. (11) Device or unit that passes or rejects a frequency band and is designed to block noise from entering or leaving a circuit or unit.

FLIGHT CODE. (4) The application software of the digital flight-control system.

FLIGHT-CRITICAL. (4,7) A description of functions whose failure would contribute to or cause a failure condition preventing the continued safe flight and landing of the aircraft.

FLIGHT-ESSENTIAL. (4) A description of functions whose failure would contribute to or cause a failure condition which would significantly affect the safety of the airplane or the ability of its crew to cope with adverse operating conditions.

FLIGHT-PHASE CRITICAL. (4) A description of functions which are critical only during certain phases of flight.

FLY-BY-GLASS. (16) Flight control system where fiber optics carry the signal.

FLY-BY-LIGHT. (4,16) Flight control system where fiber optics carry the signal.

FLY-BY-WIRE. (4,16) Flight control system with electric signaling.

FORWARD RECOVERY. (9) Restoration of the system to a consistent state by compensating for inconsistencies found in the current state so that the system may continue processing.

FOURIER TRANSFORM. (12) A mathematical method for deriving the frequency spectrum from a time dependent function.

GIGABIT. (16) One billion bits.

GLASS COCKPIT. (9) Advanced state-of-the-art electronic displays utilizing flat panel and/or cathode ray tube display technology for cockpit instrumentation.

GROUND EFFECT. (4) Increase in aircraft lift when operating near the ground.

GROUND. (11) A generic term having multiple meanings and indicating a circuit return path or a voltage reference: not "zero" voltage reference. Four hundred millivolts of noise voltage is common on "quiet" grounds. There are several types of returns and references.

HARD FAILURE. (12) A failure that requires a reset of the equipment.

HAZARD FUNCTION. (8) The conditional probability that a fault is exposed in the interval t to Δt given that the fault did not occur prior to time t.

IMMUNITY. (11) Capability of a circuit or unit to operate within performance specification in a specified electromagnetic interference environment.

INDIRECT EFFECTS. (13) Voltage and/or current transients induced by lightning in aircraft electrical wiring which can produce upset and/or damage to components within electrical/electronic systems.

INDUCED VOLTAGES. (13) A voltage produced around a closed path or circuit by changing magnetic or electric fields or structural IR voltages.

INITIALIZATION. (6) Setting the beginning parameters and values on system power-up. For redundant systems this includes setting the initial configuration of the system.

INTERNAL ENVIRONMENT. (13) The fields and structural IR potentials produced by the external environment, along with the voltages and currents induced by them.

ISOLATION. (11) Electrical separation and insulation of circuits from ground and other circuits or arrangement of parts to provide protection and prevention of uncontrolled electrical contact.

JOULE. (12) A unit of energy equal to one watt-second.

JUMPER/STRAP. (11) A short wire, strip, strap, or braid conductor installed to make a safety ground connection, to dissipate electrostatic charge, or establish continuity around a break in a circuit.

KILOBYTE. (16) One thousand bytes.

LABELED ADDRESSING. (6) The system of identifying message recipients based on labels. This is in contrast to destination terminal addresses.

LATENT FAULT. (10) A fault which has not yet produced a malfunction. (In the context of the single-fault model, benign and latent faults are equivalent.)

LIGHTNING FLASH. (13) The total lightning event in which charge is transferred from one charge center to another. It may occur within a cloud, between clouds, or between a cloud and the ground. It can consist of one or more strokes, plus intermediate or continuing currents.

LIGHTNING LEADER STROKE. (13) The leader forms an ionized path for charge to be channeled towards the opposite charge center. The stepped leader travels in a series of short, luminous steps prior to the first return stroke. The dart leader reionizes the return stroke path in one luminous step prior to each subsequent return stroke in the lightning strike.

LIGHTNING RETURN STROKE. (13) A lightning current surge that occurs when the lightning leader makes contact with the ground or an opposite charge center.

LIGHTNING STRIKE ZONES. (13) Locations on the aircraft where the lightning flash will attach or where substantial amounts of electrical current may be conducted between attachment points. The location of these zones on any aircraft is dependent on the aircraft's geometry and operational factors and often varies from one aircraft to another.

LIGHTNING STRIKE. (13) Any attachment of the lightning flash to the aircraft.

LIMITING, VOLTAGE/CURRENT. (11) Semiconductor components, diodes, Transorb, or filter designed to clip and shunt to ground an applied transient or steady-state voltage. Used to protect against noise frequencies, faults, lightning, and inductive switching transients.

LOW-PASS FILTER. (12) An electrical circuit which allows the passage of low frequencies and prevents the passage of high frequencies.

MAGNETIC FIELD. (11) A radiated, low-impedance field having lines of "flux" or magnetomotive force associated with an electrical current.

MALFUNCTION. (11) Failure or degradation in performance that compromises flight safety.

MEAN AERODYNAMIC CHORD (also mean chord). (4) The chord of an airfoil whose length is equal to the area of the airfoil section divided by the span.

MEAN FAILURE RATE. (10) A measure of survivability defined as the reciprocal of the mean time to system failure.

MESSAGE STRUCTURE. (6) The organization of both protocol and data information in a message.

MICRON. (16) One-millionth of a meter.

MISSED ALARM. (7) The failure of a fault detection monitor or algorithm to detect a fault when there is a sensor fault.

MONITORABILITY. (6) The capacity of the protocol to be viewed passively to allow observation of the dynamics of the protocol.

MULTIPLE BURST. (13) A randomly spaced series of bursts of short duration, low amplitude current pulses, with each pulse characterized by rapidly changing currents. These bursts may result from lightning leader progression or branching and may be accompanied by or superimposed on stroke or continuing currents. The multiple bursts appear to be most intense at the time of initial leader attachment to the aircraft.

MULTIPLE STRIKE. (13) Two or more lightning strikes during a single flight.

MULTIPLE STROKE. (13) Two or more return strokes occurring during a single lightning flash.

MULTIPLE TRIP MONITOR. (7) A fault detection algorithm which declares a fault after the sensor output has exceeded a predefined threshold N times.

NANOSECOND. (16) One-billionth of a second.

NEGATIVELY STABILIZED. (4) Aircraft design in which the point of effective lift is aft of the center of gravity.

NETWORK CONTROL STRATEGY. (6) The solution proposed by the designer in addressing his specific problem (design flexibility).

NOISE. (11) Conducted or radiated emission causing circuit upset, performance disorder, or undesired sound.

NUMERICAL APERTURE. (6) The angle of acceptance of light from a light source for a given fiber optic cable.

OBSERVER. (7) An algorithm which models physical relationships between sensor data and uses the data to provide fault detection for one or more sensors. This is also known as a Luenberger observer or a signal blender.

PARAMETERIZATION CAPABILITY. (6) A measure of how well the attributes of the protocol can be described by parameters.

PEAK RATE OF RISE. (13) The maximum instantaneous slope of the waveform as it rises to its maximum value. Mathematically, the peak rate of rise of a function, i(t), may be expressed as the maximum of d[i(t)]/dt.

PIN LEVEL TEST. (12) An EMC test in which voltage or current is applied directly to a conductor at a connector pin.

POINT-MASS SIMULATION. (4) Same as state variables airplane model (q.v.)

POSITIVELY STABILIZED AIRCRAFT. (4) Aircraft design in which the effective point of lift is forward of the center of gravity.

PRECIPITATION STATIC (P-static). (11) Electrostatic discharge, corona, arcing, and streamering, steady state or impulsive, causing circuit upset, receiver noise or component damage.

PREDICATE/TRANSITION NETWORK. (4) A bipartite graph (a type of linear graph) to model concurrency between redundant concurrent events. Basically a modified generalized petri net.

Q. (12) The quality factor of a resonant circuit which is the ratio of the energy stored to the power dissipated per cycle.

QUADRUPLEX ARCHITECTURE. (4) The use of four separate lanes (or channels) of computer redundancy. Each lane can fail separately providing a fail-operational capability for the digital flight-control system.

RADIATED EMISSION (RE). (11) Electromagnetic energy transmitted and propagated in space usually considered as audio frequency or radio frequency noise.

RADIO FREQUENCY (RF). (11) Frequencies in the electromagnetic spectrum used for radio communications extending from kilohertz to gigahertz.

RADIO FREQUENCY INTERFERENCE (RFI). (11) Electromagnetic interference in the radio frequency range.

RECONFIGURATION. (6) The capacity of a system to rearrange or reconnect the system elements or functions.

RECOVERY CACHE. (9) The location used to preserve input values until the outputs resulting from them have been accepted.

REDUNDANCY MANAGEMENT. (7) The computer processing which is needed to implement fault detection and isolation algorithms.

REFERENCE. (11) 1. Structure, for electronics, shields, power. 2. A grid of wires, solid sheet, or foil. 3. A wire from circuit to grounding block or case. 4. A wire from circuit to structure. 5. Shield tie. 6. Earth.

RELAXED STATIC STABILITY AIRCRAFT. (4) An aircraft whose center of gravity is behind the wing's point of effective lift.

RELIABILITY ANALYSIS. (4) A means of determining the probability of failure in a system. Military flight-critical systems typically are required to have reliability levels of 10^{-5} to 10^{-7}, whereas civil flight-critical systems have reliability levels of 10^{-9} or less.

RESONANCE. (12) Resonance occurs in an electrical circuit when the energy stored in the inductance is equal to the energy stored in the capacitance.

RETURN STROKE. (13) See lightning return stroke.

RETURN. (11) 1. Structure, for power, fault, and "discrete" circuits. 2. A grid of wires, solid sheet, or foil. 3. A wire from circuit load back to source or to case. 4. Circuit card "ground plane," also a reference and shield.

REVERSION MODE. (7) The high level of redundancy in a system having different redundancy requirements for some sensors. Critical sensors may have a high level of redundancy while other sensors have low levels.

RISE-TIME. (12) The time required for a voltage pulse to reach a predefined magnitude from a given level.

ROBUSTNESS. (9) The ability of the code to perform despite some violation of the assumptions in its specifications usually via substitution of an alternate value and continuation of execution if a software fault is detected.

ROLLBACK. (9) Retrying the calculation in the event that a failure is detected, under the assumption that some external condition may have changed thereby resolving the anomaly.

SEALANT. (11) An applied substance enclosing and protecting the integrity of a joint, fastener, or electrical bond from moisture, contaminants, oxidation, and acid or alkaline corrosion.

SENSOR. (7) An instrument which measures a particular physical parameter. The data output may be digital or analog and is utilized by the flight computer.

SEQUENTIAL LIKELIHOOD RATIO TEST. (7) A fault detection algorithm which is based on two hypothesized density functions of no fault or sensor fault.

SEQUENTIAL PROBABILITY RATIO TEST. (7) See sequential likelihood ratio test.

SHIELD EFFECTIVENESS (SE). (11) The ability of a shield to reject electromagnetic fields. A measure of attenuation in field strength at a point in space caused by the insertion of a shield between the source and the point.

SHIELD. (11) A conductive material, opaque to electromagnetic energy, for confining or repelling electromagnetic fields. A structure, skin panel, case, cover, liner, foil, coating, braid, or cable-way that reduces electric and magnetic fields into or out of circuits or prevents accidental contact with hazardous voltages.

SHIELDING. (12) Any metallic structure such as the aircraft fuselage or the woven braid on a cable that provides protection against electromagnetic fields.

SIGNAL RETURN. (11) A wire conductor between a load and the signal or driving source. Structure can be a signal and power return. Commonly, it is the low voltage side of the closed loop energy transfer circuit.

SINGLE-ENDED CIRCUIT. (11) A circuit with source and load ends grounded to case and structure and using structure as return.

SINUSOID. (12) A wave form that follows the mathematical values of a sine function.

SOFT FAILURE. (12) A failure which causes an alteration of data or missing data.

STATE-VARIABLE AIRPLANE MODEL (also point-mass model). (4) Fixed aerodynamic variables are used in the solution of the equations of motion of the model instead of using look-up tables in which each derivative varies with airspeed, altitude, etc. The model performance is only accurate at or near the point in the flight envelope for which the variables are chosen.

STATIC MARGIN. (4) The degree of instability in a relaxed statically stable airplane.

STRUCTURAL IR VOLTAGE. (13) The portion of the induced voltage resulting from the product of the distributed lightning current, I, flowing through the resistance, R, of the aircraft skin or structure.

STRUCTURE. (11) Basic members, supports, spars, stanchions, housing, skin panels, or coverings that may or may not provide conductive return paths and shields for electrical/electronic circuits.

SUPER-DIAGNOSTIC FILTER. (7) An algorithm which provides all the capabilities of a diagnostic filter. Additionally, it can isolate a specific faulted sensor. At the current time, this is the most complex technique used to implement analytical redundancy.

SUSCEPTIBILITY. (11) Upset behavior or characteristic response of an equipment when subjected to specified electromagnetic energy. Identified with the point, threshold, or onset of operation outside of performance limits. Conducted Susceptibility (CS) applies to energy on interface conductors; Radiated Susceptibility (RS) to radiated fields.

SWEPT STROKE. (13) A series of successive attachments due to sweeping of the flash across the surface of the airplane by the motion of the airplane.

SYNCHRONOUS MESSAGES. (6) Messages transmitted at a known a priori sequence and time or time interval.

SYSTEM EXPOSURE TIME. (4) The period during which a system may fail. This period extends from the last verified proper functioning to the completion of the next required performance.

SYSTEM FUNCTIONAL UPSET. (13) Impairment of system operation, whether permanent or momentary (e.g., a change of digital or analog state) which may or may not require manual reset.

SYSTEM INTEGRITY. (6) The degree to which a system is dependable.

SYSTEM RELIABILITY. (5) The probability of performing a given function from the some initial time, t=0, to time t.

TESTABILITY. (6) A measure of how well the protocol supports completeness of testing and the protocol's ability to produce repeatable or predictable results.

THRESHOLD, NOISE. (11) The lowest electromagnetic interference signal level that produces onset of susceptibility.

THROUGHPUT. (6) The productivity of a data processing system as expressed in computing work per minute or hour.

THYRISTORS. (16) Solid-state devices that convert alternating current to direct current.

TIME CONSTANT. (4) Time required to double the amplitude of the divergent real root in the pitch axis of the aircraft model.

TRANSIENT CONTROL LEVEL. (13) The maximum allowable level of transients appearing at the systems interfaces as a result of the defined external environment.

TRANSPARENT RECOVERY. (4) Correcting a soft fault without interrupting the system's intended performance.

TRIBOELECTRIC CHARGING. (13) Static electricity produced on a structure from the effects of friction.

UNACCEPTABLE RESPONSE. (11) Upset, degradation of performance, or failure, not designated a malfunction, but is detrimental or compromising to cost, schedule, comfort, or workload.

UNDESIRABLE RESPONSE. (11) Change of performance and output, not designated a malfunction or safety hazard, that is evaluated as acceptable as is because of minimum nuisance effects and excessive cost burdens to correct.

UPSET. (11) Temporary interruption of performance that is self-correcting or reversible by manual or automatic process.

UPSET. (12) A condition in which the state of a digital device is unintentionally altered, but may be restored by automatic means or by operator intervention.

UPSET. (13) See system functional upset.

VALIDATION. (4,11) Demonstration and authentication that a final product operates in all modes and performs consistently and successfully under all actual operational and environmental conditions founded upon conformance to the applicable specifications.

VERIFICATION. (4,11) Demonstration by similarity, previous in-service experience, analysis, measurement, or operation that the performance, characteristics, or parameters of equipment and parts demonstrate accuracy, show the quality of being repeatable, and meet or are acceptable under applicable specifications.

VOTING PROCEDURE. (8) An algorithm included in fault tolerant software which uses the consensus recovery block method. It compares outputs of the n independent versions and determines which outputs are correct by identifying agreements among two or more versions.

ACRONYMS AND ABBREVIATIONS

μm (6)	Micrometer
ϕM (6)	Phase Modulation
3-D (16)	Three-Dimensional
AAES (15)	Advanced Aircraft Electrical System
A/L (3)	Approach/Land
A/C (11)	Aircraft
AC (3,5,14)	Advisory Circular
ac (3,6,12,15)	Alternating Current
ACAP (13)	Advanced Composite Airframe Program
ACARS (11,12)	ARC Communications Addressing and Reporting System
ACES (13)	Applied Computational Electromagnetics Society
ACS (16)	Automatic Control System
ACT (11,12)	Active Controls Technology
ADC (11,12)	Air Data Computer
ADF (11,12)	Automatic Direction Finder
ADI (3)	Automatic Direction Indicator
AE (6)	Avionics Equipment
AE4L (5,13)	SAE Subcommittee (Lightning)
AEHP (13)	Atmospheric Electricity Hazards Protection
AERA (16)	Automated En Route Air Traffic Control System
AES-S (6)	Aerospace and Electronic Systems Society
AF (11,12)	Audio Frequency
AFBW (4)	Augmented Fly-By-Wire
AFCS (11,12)	Automatic Flight Control System
AFFDL (7,8,13)	Air Force Flight Dynamics Laboratory
AFM (16)	Advanced Fuel Management
AFWAL (6,13)	Air Force Wright Aeronautical Laboratory
AGARD (8)	Advisory Group for Aerospace Research and Development
AHRS (6)	Attitude Heading Reference System
AI (16)	Artificial Intelligence
AIAA (5,6,9,15)	American Institute of Aeronautics and Astronautics
AIRLAB (5)	Avionics Integration Research Laboratory
AK (7)	Altitude Kinematics
ALCM (13)	Air Launched Cruise Missile
ALU (3,5,10)	Arithmetic Logic Unit
AM (5)	Amplitude Modulated
AM (6,14)	Amplitude Modulation
AMSC (5)	Document number prefix used by the Department of Defense
ANSI (11,12)	American National Standards Institute
AOA (4)	Angle of Attack
APU (11,12,15)	Auxiliary Power Unit
AR (7)	Analytical Redundancy
ARC (11,12)	Aeronautical Radio, Incorporated
ARIES (3)	Automated Reliability Interactive Estimation System
ARINC (3,6)	Aeronautical Radio, Incorporated

ARTERI (15)	Analytical Redundancy Technology for Engine Reliability Improvement
ASCB (6)	Avionics Standard Communications Bus
ASDS (11,12)	Airport Surface Detection System
ASEE (5,15)	American Society of Electrical Engineers
ASME (5)	American Society of Mechanical Engineers
ATCRBS (11,12,14)	Air Traffic Control Radar Beacon System
ATF (16)	Advanced Tactical Fighter
ATI (16)	Access Time Interval
ATTR (5)	Attribute
AWACS (14)	Airborne Warning and Control System
B-dot (13)	Derivative of the magnetic field with respect to time
B-GLOSS (5)	Gate Logic Software Simulator developed by Bendix
BB (11,12)	Broadband
BCI (12)	Bulk Cable Injection
BGU (10)	Bus Guardian Unit
BIR (6)	Benchmark Information Rate
BIT (6,15)	Built-In Test
BITE (6,11,12)	Built-In Test Equipment
BIU (6)	Bus Interface Unit
bps (6)	Bits Per Second
BW (11,12)	Bandwidth
C/I (5)	Communicator Interstage
C (7)	Comparator
CAA (14)	Civil Aviation Authority
CAD (5)	Computer Aided Design
CAP (5)	Collins Application Processor
CAPS (3)	Computer Aided Production Simulator
CARE (3,5)	Computer Aided Reliability Evaluator
CARSRA (3,7)	Computer-Aided Redundant System Reliability Analysis
CAS (11,12)	Criticality Advisory System
CAST (3)	Complementary Analytic Simulative Technique
CBD (5)	Commerce Business Daily
CCITT (6)	Consultative Committee for International Telephone and Telegraph
CD (6)	Collision Detection
cdf (8)	Cumulative Density Function
CDU (11,12)	Control Display Unit
CE (11,12)	Conducted Emission
CM (11,12)	Common Mode
CMOS (5,12)	Complimentary Metal-Oxide Semiconductor
CONUS (14)	Contiguous United States
CPA (5)	Central Processor - A
CPU (5,10)	Central Processing Unit
CR (5)	Contractor Report
CR (6)	Command Response
CRC (6)	Cyclic Redundancy Check
CRMI (2)	Computer Resource Management, Incorporated
CRT (11,12,16)	Cathode Ray Tube
CS (11,12)	Conducted Susceptibility

CSC (9)	Computer Software Component
CSCI (9)	Computer Software Configuration Item
CSDL (5,10,15)	Charles Stark Draper Laboratories
CSMA/CD (6)	Carrier Sense Multiple Access/Collision Detection
CSMA (6,16)	Carrier Sensed Multiple Access
CT (2,6)	Technical Center (designation used in FAA report numbering scheme)
CTA (3)	CAPS Test Adapter
CTA (5)	Collins Test Adaptor
CW (13)	Continuous Wave
DADC (6)	Digital Air Data Computer
DARPA (16)	Defense Advanced Research Projects Agency
DATAC (6)	Digital Autonomous Terminal Access Communication
dB (6,12)	Decibel
dBi (14)	Decibels with respect to one milliampere
dBm (6)	Decibels per meter
dc (6,12,15)	Direct Current
DE (5)	Diagnostic Emulation
DEFN (5)	Definition
DEV (5)	Development
DF (7)	Diagnostic Filter
DFC (7)	Digital Flight Control
DFCS (3,4,7,16)	Digital Flight Control System
DFDAU (11,12)	Digital Flight Data Acquisition Unit
DFDR (11,12)	Digital Flight Data Recorder
DGAC (14)	Direction Generale Aviation Civile
DISAC (15)	Digital Integrated Servo Actuator Controller
DITS (6,11,12)	Digital Information Transfer System
DM (11)	Differential Mode
DM (6)	Delay Modulation
DMA (5,6)	Direct Memory Access
DME (11,12)	Distance Measuring Equipment
DNA (13)	Defense Nuclear Agency
DOD (5,8,12,14,16)	Department of Defense
DOE (13)	Department of Energy
DOT (2,3,6,7,8)	Department of Transportation
DRB (9)	Distributed Recovery Block
DSP (3)	Discrete Switch Panel
E-FIELD (11,12)	Electric Field
E/E (11,12)	Electrical/Electronic
E (11,12)	Electromagnetic Environmental Effects
E-dot (13)	Derivative of the electric field with respect to time
EADI (11,12)	Electronic Attitude Director Indicator
ECAC (11,12,14)	Electromagnetic Compatibility Analysis Center
ECM (14)	Electronic Counter Measures
ECS (11,12)	Environmental Control System
EEC (5,11,12)	Electronic Engine Control
EED (11,12)	Electro-Explosive Device
EFIS (11,12)	Electronic Flight Instrument System
EFMA (3)	Executive Failure My A

EFMB (3)	Executive Failure My B
EFOA (3)	Executive Failure Other A
EFOB (3)	Executive Failure Other B
EFW (3)	Executive Failure Word
EGT (11,12)	Exhaust Gas Temperature
EHSI (11,12)	Electronic Horizontal Situation Indicator
EICAS (11,12)	Engine Indication and Crew Alerting System
EIU (16)	Electronic Interface Unit
EM (5,11,12,13)	Electromagnetic
EMA (15)	Electromechanical Actuator
EMAS (2,15)	Electromechanical Actuator System
EMC (6,11,12,13,14)	Electromagnetic Compatibility
EMCad^{tm} (12)	Electromagnetic Computer aided design
EME (14)	Electromagnetic Environment
EME (11,12)	Electromagnetic Effects
EMI (5,6,11,12,13,15,16)	Electromagnetic Interference
EMIC (11,12)	Electromagnetic Interference/Compatibility
EMP (11,12,13)	Electromagnetic Pulse
EMR (5,14)	Electromagnetic Radiation
EMUX (6)	Electrical Multiplex
ENRZ (6)	Enhanced Non-return to Zero
EPR (11,12,15)	Engine Pressure Ratio
EPROM (16)	Erasable Programmable Read-Only Memory
ESD (11,12)	Electrostatic Discharge
ESE (11,12)	Electric (field) Shield Effectiveness
ESS (5,9)	Electronic Switching System
ETDL (13)	Equipment Transient Design Level
EUROCAE (14)	European Organization for Civil Aviation Electronics
EXCHNG (5)	Exchange
EXP (5)	Experiment
FAA (ALL)	Federal Aviation Administration
FADEC (6,15,16)	Full Authority Digital Engine Controller
FAFTEEC (16)	Full Authority Fault Tolerant Electronic Engine Control
FAR (3,4,6,16)	Federal Acquisition Regulation
FBL (15,16)	Fly-By-Light
FBW (4,7,16)	Fly-By-Wire
FCC (3,4,5,6,10,11,12,15)	Flight Control Computer
FCR (5)	Fault Containment Regions
FCS (4,5,10,16)	Flight Control System
FD (7)	Fault Detection
FDEP (11,12)	Flight Data Entry Panel
FDFM (15)	Fault Detection and Failure Management
FET (5,16)	Field Effect Transistor
FI (5)	Fault Insertion circuitry
FI (7)	Fault Isolation
FIAT (5)	Fault Injection Automated Testing
FICA (15)	Failure Indication and Corrective Action
FIIS (5,10)	Fault Insertion and Instrumentation System
FIM (5)	Fault Injection Manager
FIRE (5)	Fault Injection Receptor
FM (5)	Frequency Modulated

FM (6,14)	Frequency Modulation
FMC (11,12)	Flight Management Computer
FMEA (3,9)	Failure Mode and Effect Analysis
FMECA (5,15)	Failure Modes and Effects Criticality Analysis
FT (5)	Fault Tolerant
ft (14)	feet
FTC (5)	Fault Tree Compiler
FTMP (5,10)	Fault-Tolerant Multiprocessor
FTP (5)	Fault Tolerant Processor
G/E (13)	Graphite Epoxy
GaAs (6,16)	Gallium Arsenide
GAMA (6)	General Aviation Manufacturers' Association
GCR (6)	Group Code Recording
GE (5)	General Electric
GEMACS (13)	General Electromagnetic Model for the Analysis of Complex Systems
GEN (5)	Generation
GGLOSS (5)	Generalized Gate-Level Logic System Simulator
GLOSS (5)	Gate Logic Software Simulator
GNC (16)	Guidance, Navigation, and Control
GPC (5)	General Purpose Computer
GPS (11,12)	Global-Positioning-System
GPWS (11,12)	Ground Proximity Warning System
Gr/Ep (11,12)	Graphite/Epoxy
GS (10)	Glideslope
H-FIELD (11,12)	Magnetic Field
H1 (11,12)	Fan Speed
HARP (5)	Hybrid Automated Reliability Predictor
HDBK (5)	Handbook
HDLC (6)	High-Level Data Link Control
HERF (5,14,16)	High-Energy Radio Frequency
HF (11,12,13,14)	High-Frequency
HIRF (15)	High-Intensity Radiated Fields
HSI (3)	Horizontal Situation Indicator
HSRB (6)	High-Speed Ring Bus
HVDC (15)	High Voltage dc
Hz (3,15)	Hertz
I/O (5,10,15)	Input/Output
I (13)	Current
I-dot (13)	Derivative of the current with respect to time
IAA (3)	Integrated Assurance Assessment
IAAC (11,12)	Integrated Application of Active Controls Technology (to an Advanced Subsonic Transport Project)
IBM (5)	International Business Machines
IC (3)	Integrated Circuit
ICAO (14)	International Civil Aviation Organization
ICIS (5)	Intercomputer Interface Sequencer
ICS (5)	Intercomputer Sequencer
ID (5)	Identification

IDG (11,12)	Integrated Drive Generator
IEEE (5,6,9)	Institute for Electrical and Electronics Engineers, Incorporated
IFF (14)	Identification - Friend or Foe
IGGLOSS (5)	Gate Logic Software Simulator (improved version developed at NASA Langley)
ILS (3,11,12)	Instrument Landing System
INS (6,11,12)	Inertial Navigation System Institute
IOPA (5)	Input/Output Processor - Channel A
IOS (5)	Input/Output Subsystem
IRS (11,12)	Inertial Reference System
ITT (16)	(Consultative Committee for) International Telegraphy and Telephony
IV&V (9)	Independent Verification and Validation
JPL (5)	Jet Propulsion Laboratories
K (6)	Thousand
kA (13,16)	Kiloampere
kHz (5,6,12,14)	Kilohertz
km (6)	kilometer
LAN (5)	Local Area Network
LaRC (5)	Langley Research Center
LCC (11,12)	Life Cycle Cost
LCD (16)	Liquid Crystal Display
LED (6)	Light Emitting Diode
LF (13)	Low Frequency
LOC (11,12)	Localizer
LPN (13)	Lumped Parameter Network
LRC (6)	Longitudinal Redundancy Check
LRRA (11,12)	Low Range Radio Altimeter
LRU (5,6,11,12,13)	Line Replaceable Unit
LSB (6)	Least Significant Bit
LSI (5)	Large Scale Integration
LTPB (6)	Linear Token Passing Bus
LTRI (13)	Lightning and Transients Research Institute
LVDT (15)	Linear Variable Differential Transformers
LVDT (3)	Linear Voltage Differential Transducer
m (6)	meter
M (6)	Million
M (5)	Mutual (when used with RLC)
mA (3)	Milliampere
MAADS (6)	Multibus Avionic Architecture Design Study
MAC (4)	Mean Aerodynamic Chord
MAFT (16)	Multicomputer Architecture for Fault Tolerance
Mbps (6,16)	Million bytes per second
MCDP (11,12)	Maintenance Control and Display Panel
MCP (11,12)	Mode Control Panel
MDICU (3)	Modular Digital Interface Control Unit
MDICU (4,5)	Modular Digital Interface Conversion Unit

MDT (6)	Mean Down Time
Mflops (16)	Million floating-point operations per second
MFM (6)	Modified-Frequency Modulation
MFR (10)	Mean Failure Rate
MHz (5,6,12,14)	Megahertz
mil (11,12)	One thousandths of an inch (0.001)
MIL (5)	Military
MIL-STD (6)	Military Standard
MLE (8)	Maximum Likelihood Estimates
MLS (11,12)	Microwave Landing System
MOS (5)	Metal-Oxide Semiconductor
MPP (16)	Massively Parallel Processor
MPX (5)	Multiplex
ms (6,10)	Millisecond
MSB (6)	Most Significant Bit
MSE (11,12)	Magnetic (Field) Shielding Effectiveness
MSI (5)	Medium Scale Integration
MTBCF (6)	Mean Time Between Critical Failures
MTBF (9)	Mean Time Between Failures
MTTF (8,10)	Mean Time to Failure
MTTR (6)	Mean Time To Repair
Mux (5)	Multiplexed
MUX (4)	Multiplexer
N_1 (15)	Low Rotor Speed
N2 (11,12)	Core Engine Speed
N_2 (15)	High Rotor Speed
NA (6)	Numerical Acceptance
NA (3)	Normal Accelerometers
NADC (6,7)	Naval Air Development Center
NAECON (6)	National Aerospace & Electronics Conference
Naecon (5)	National Avionics and Electronics Conference
NAND (5)	Not AND
NASA (2,3,5,7,13,15)	National Aeronautics and Space Administration
NASC (13)	Naval Air Systems Command
NATO (14)	North Atlantic Treaty Organization
NB (11,12)	Narrow Band Signal
NEC (13)	Numerical Electromagnetics Code
NEMP (5,12)	Nuclear Electromagnetic Pulse
NHPP (8)	Non-Homogeneous Poisson Process
nmi (14)	nautical mile
NPRM (14)	Notice of Proposed Rulemaking
NRZ (6)	Non-return to Zero
NRZ-I (6)	Non-return to Zero Inverted
NRZ-L (6)	Non-return to Zero Dual Level
nsec (6)	Nanosecond
NSWC (13)	Naval Surface Weapons Center
NVS (8)	N-version Software
OMEGA (11,12)	Very Low Frequency Navigation
OMV (16)	Orbital Maneuvering Vehicle
OS (5)	Operating System

OTV (16)	Orbital Transfer Vehicle
P-Static (11,12)	Precipitation Static
P (10)	Processor
PAL (5)	Programmable Array Logic
PAS (6)	Pilot Assist System
PAWS (5)	Padé Approximation With Scaling
PBW (15)	Power-By-Wire
PC (5)	Personal Computer
PCS (16)	Primary Control System
PCU (11,12)	Power Control Unit
pdf (8)	Probability Density Function
PE (6)	Phase Encoding
pf (12)	picofarad
PLA (15)	Power Level Actuator
PLA (16)	Power Level Angle
PMS (5)	Physical Message Switch
PRF (11,12)	Pulse Repetition Frequency
PROC (5)	Processor
PROM (3,10)	Programmable Read-Only Memory
psi (15)	pounds per square inch
PVI (16)	Pilot/Vehicle Interface
PWM (11,12)	Pulse Width Modulation
PZ (15)	Piezoelectric
QUAD (5)	Quadruple
R (13)	Resistance
R-C (12)	Resistor-Capacitor
RADC (8)	Rome Air Development Center
RAE (13)	Royal Aircraft Establishment
RAM (3,10,16)	Random Access Memory
RAM (5)	Random Access Memory
RAT (6)	Ring Admittance Timer
RB (8)	Recovery Block
RBDCP (3)	Reliability Block Diagram Computer Program
RCA (16)	Radio Corporation of America
RCVR (5)	Receiver
RDFCS (4)	Reconfigurable Digital Flight Control System (facility)
RDFCS (3)	Redundant Digital Flight Control System
RDFCS (5)	Reconfigurable Digital Flight Control System
RDMI (11,12)	Radio Distance Magnetic Indicator
RE (11,12)	Radiated Emission
REG (5)	Register
REL (3)	Reliability
REL COMP (3)	Reliability Computers
RF (5,11,12,13,14,16)	Radio Frequency
RFI (11,12)	Radio Frequency Interference
RL (13,15)	Resistance/Inductance (Out of order in c.15)
RLC (5,13)	Resistance/Inductance/Capacitance
RLCM (13)	Resistance/Inductance/Capacitance/Mutual
RM (7)	Redundancy Management

RNRZ (6)	Randomized Non-return to Zero
ROM (5,10)	Read Only Memory
RPV (16)	Remotely Piloted Vehicle
RS (11,12)	Radiated Susceptibility
RSS (4,7)	Relaxed Static Stability
RT (5)	Remote Terminal
RTCA (2,3,11,12,14,16)	Radio Technical Commission for Aeronautics
RTI (5)	Remote Terminal Interface
RZ (6)	Return to Zero
S/A (11,12)	Spectrum Analyzer
S-a-0 (5,10)	Stuck at Zero
S-a-1 (5,10)	Stuck at One
S-GLOSS (5)	Gate Logic Software Simulator developed by Stevens
SAE (2,5,6,13,14,16)	Society of Automotive Engineers
SAS (4,7)	Stability Augmentation System
SDF (7)	Super-Diagnostic Filter
SE (11,12)	Shielding Effectiveness
SEU (5)	Single Event Upset
SHF (11,12)	Super High-Frequency
SHRD (5)	Shared
SIAM (15)	Society for Industrial and Applied Mathematics
SIF (14)	Selective Identification Facility
SLRT (7)	Sequential Likelihood Ratio Test
SMOTEC (14)	Special Missions Operation Test and Evaluation Center
SPRT (7)	Sequential Probability Ratio Test
SQL (5)	Software Query Language
SSI (5)	Small Scale Integration
SSP (3)	Servo Simulation Panel
SSPC (15)	Solid-State Power Controller
STANAG (14)	Standardization Agreement (NATO)
STC (2)	Supplemental Type Certification
STEM (5)	Scaled Taylor Expansion Matrix
STOL (16)	Short Takeoff and Landing
str (6)	string
SURE (5)	Semi-Markov Unreliability Range Evaluator
SYN (5)	Synch
T/R (6)	Transmitter/Receiver
TACAN (14)	Tactical Air Navigation
TASRA (3)	Tree Aided System Reliability Analysis
TC (2)	Type Certification
TCAP (13)	Threshold Circuit Analysis Program
TCAS (11,12,16)	Traffic Alert and Collision Avoidance System
TCL (13)	Transient Control Level
TDM (6)	Time Division Multiplex
THT (6)	Token-Holding Timer
TLA (11,12)	Thrust Lever Angle
TMC (11,12)	Thrust Management Computer
TMR (10)	Triple Modular Redundant
TRU (15)	Transformer Rectifier Unit
TTL (5,11,12,13,16)	Transistor-Transistor Logic

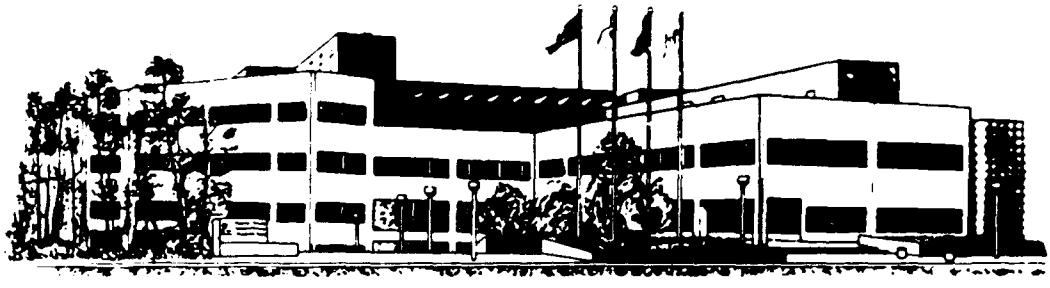
TV (14)	Television
TWTD (13)	Thin Wire Time Domain
TX (5)	Transmit
U.K. (13,14)	United Kingdom
U.S. (14)	United States
UART (15)	Universal Asynchronous Receiver Transmitter
UHF (11,12,13,14)	Ultra High-Frequency
UNIBUS (5)	Universal Bus
UPS (15)	Uninterruptible Power Supplies
USAF (16)	United States Air Force
USB (16)	Upper Surface Blowing
USEG (5)	Unsegmented
V/m (14)	Volt/meter
VHF (11,12,13,14)	Very High-Frequency
VHSIC (6,16)	Very-High-Speed Integrated Circuits
VLF (11,12)	Very Low-Frequency
VLSI (5,6,14)	Very Large Scale Integration
VLSIC (6,16)	Very Large Scale Integrated Circuits
VOR (11,12,14)	VHF Omnidirectional Range
VORTA/VHF (11,12)	Omnirange/Tactical Air Navigation
VRC (6)	Vertical Redundancy Check
VSI (11,12)	Vertical Speed Indicator
VSV (15)	Variable Stator Vane
VTOL (16)	Vertical Takeoff and Landing
W/P (15)	Fuel Flow to Burner Pressure
WAI (3)	Warning Annunciation Indicator
WFM (15)	Main fuel metering valve actuator sensor
WRU (11,12)	Weapons Replaceable Unit
XAB (5)	Transmit Compare A B
XMT (5)	Transmit
XMTR (3)	Transmitter
XOR (5)	Exclusive OR
ZM (6)	Zero Modulation

U.S. Department of Transportation
Federal Aviation Administration

DOT/FAA/CT-88/10

HANDBOOK-VOLUME II
DIGITAL SYSTEMS VALIDATION

CHAPTER 5
ADVANCED FAULT INSERTION AND SIMULATION METHODS



PREPARED BY:

COMPUTER RESOURCE MANAGEMENT, INC.
950 HERNDON PARKWAY, SUITE 360
HERNDON, VIRGINIA 22070

PREPARED FOR:

FEDERAL AVIATION ADMINISTRATION
TECHNICAL CENTER
ATLANTIC CITY INTERNATIONAL AIRPORT, NEW JERSEY 08405

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. FAULT TOLERANT SYSTEMS RELIABILITY PREDICTION BACKGROUND

1.1 Introduction and History

Application of microprocessor-based computers to flight critical systems is becoming more commonplace. The flight management systems on board the newest series of aircraft bring to aviation a new capability in management of the air traffic and additional capability for safe flight in marginal weather conditions. All-electronic engines have come into use on the latest generation aircraft as well. Stability augmentation puts the control of aircraft loads under computer control. The latest Airbus has brought military fly-by-wire concepts into commercial aviation for stability and control. This Airbus, operated by a new generation of on-board computers, offers greater fuel efficiency and smoother flight path control under varying aircraft configurations. Additional computer applications are under development to increase future aircraft system automation for stability and control, and for flight management. Considerable attention has been directed toward the technology and methods for validating the performance of these flight critical systems, (National Aeronautics and Space Administration [NASA] Conference Publication 2114, Working Group Meetings I and II).

Failures of these advanced electronic systems would place a severe load on the pilot and crew in operating the aircraft without automation. The reliability of electronic components is not good enough to ensure a sufficiently low probability of failure in flight critical systems. Computer processors, having millions of components on a single chip of silicon, are available. Testing of these devices under all possible conditions is not feasible. Single logic circuit failures within these high density Very Large Scale Integration (VLSI) components must not cause loss of function. Such loss of function could arise from either a random failure or a defect present within the device that escaped detection in the original design and fabrication of the device. If a microprocessor is part of a system, the user must assume that the device may have some inherent fault that eluded detection during manufacturing.

The addition of duplicate, redundant, computers is a promising approach to ensure that at least one good computer will be available in the event of failure during flight. Redundancy has been useful in many prior aircraft systems; the low cost, weight, and power requirements of microelectronic devices make this attractive. Pioneers in reliability modeling demonstrated mathematically that system reliability could be enhanced by redundant hardware. These results are based upon the assumptions that the devices were initially fault free, that failures occur independently, and that errors can be detected by comparing the results from redundant components. More recently it has been shown that increased reliability can be attained, even if the components have inherent faults and not all failures can be detected (Arnold 1973).

The critical factor in achieving increased reliability through redundant components is the determination of how the system detects and processes faults. Therefore, the study of faulted behavior of digital circuits has expanded for two reasons: (1) to determine whether the output is correct for a given input and the device is sufficiently complete to manufacture and distribute, and (2) to determine the effectiveness of the device or system to detect, isolate, and recover from faults and errors. Because of the complexity of the computer software and the system (i.e., the number of component circuits), the entire study and testing is viewed as a stochastic process.

This chapter deals with system reliability issues in depth, through discussion of the concepts and methods for system testing by inserting faults and measuring the recovery process. An extensive bibliography is included for further reading on this subject (references that are given in the text by the author's last name and date refer to entries in the bibliography). A majority of the discussion material results from efforts by NASA Langley over the last ten years. When NASA began these investigations, reliability modeling was almost entirely based upon the failure modeling of single components. Data for these models were obtained for military programs from accelerated life cycle testing and 100 percent acceptance tests on receipt. These methods are too costly and too restrictive for commercial programs. Furthermore, the concept of 100 percent testing of microcomputers has proved impractical to implement, even for MIL-SPEC parts, due to the size and complexity of the test effort. Some of the work on fault insertion and testing of fault tolerant systems is still in progress and all the answers are not available for this challenging topic.

1.2 Reliability Requirements

The reliability of a system is the probability of performing a given function from some initial time, t=0, to time t. Typically, it is more convenient to deal with the probability of failure, which is one minus the reliability. For mission critical applications, a typical requirement is that the probability of failure be no more than 10^{-5} per hour. For flight critical safety related applications, the acceptable probability of failure is 10^{-9} per hour.

1.3 Fault Tolerant Systems Performance Issues

1.3.1 Processor Redundancy

Redundancy uses the execution of the same code on two processors to detect and correct the effects of faults before they propagate errors into other tasks in a system. Reliability predictions are based upon independence of the faults in the individual processors; i.e., a fault affecting one processor must not be able to cause an error in another processor. Fault independence can be enforced by separating the computations into fault containment regions (FCRs). An FCR is a collection of components that operates correctly regardless of any arbitrary logical or electrical fault outside of the region. If the entire system is partitioned into FCRs, an arbitrary fault inside such a region of containment cannot cause any component outside that region to fail in any manner. Therefore, fault tolerance includes containment of faults and correct handling of errors.

In order to use hardware redundancy effectively against random component failures, it is necessary that faults in different FCRs be independent and uncorrelated. To avoid common mode failures, each FCR must have a power supply and clock independent and separate from those of any other region. Furthermore, it is necessary to provide electrical isolation at all inter-FCR communication points.

Duplex redundancy is sufficient to detect errors, however, triplex redundancy is needed to mask errors. Redundancy is very effective if all the computations are performed in lock step with identical inputs and intermediate steps. Exact match voting can then be used to detect and mask errors. Thus any value not bit-for-bit identical to the other two is considered to be in error and is discarded. The input values must be interchanged between the FCRs to ensure that the same values are used by all three processors. If the processors are not in lock step synchronization, exact match voting cannot be used, and fault detection and masking is compromised.

1.3.2 Byzantine Resilience

A fault tolerant process is termed "Byzantine Resilient" if it is tolerant of intermittent faults that can send good information part of the time. The term alludes to the problem faced by Byzantine generals trying to authenticate messages sent by traitorous messengers. Unfortunately computer components do not often fail in benign ways. Components involved in communication between processors have been known to fail in ways that send conflicting information to different parts of the system (Johnson 1989). If the redundancy is not sufficient to isolate these failures, then they are very difficult to isolate and correct. The difficulty arises because a processor correctly processing incorrect input will appear faulty to the rest of the system. In order for a system of parallel processors to be Byzantine resilient the network topology must be divided into four separate FCRs. Each computation must be performed in two separate regions; whenever data values pass from one containment region to another the communications must be via separate paths. If a containment region no longer has three separate paths for communications, then the system will no longer be Byzantine Resilient. The reconfiguration of such a system is very complex.

1.3.3 Synchronization

Synchronizing individual processors in a redundant computation makes it possible to schedule completion of individual process steps. This simplifies the determination of whether a processor has failed or simply is slow to complete a task. Failures of synchronized processors must not propagate beyond a containment region. Global clocking methods are not recommended because they permit too high a chance of a single point of failure. Much research has been done on synchronization methods. Various methods range from loosely coupled methods to lock step synchronization. The computation overhead for correctly dealing with processor synchronization is usually very large.

1.3.4 Scheduling

Task scheduling in the redundant processors requires a real-time executive that can operate in the presence of faults. This executive must be responsible for reconfiguration and fault masking on data between processors. The executive must also maintain data integrity in the presence of faults. This means that multiple copies of the executive must exist in the various FCRs.

1.3.5 Deadlock Protection

It is necessary to guard against the deadlock condition where every processor is waiting for a task to be completed as a subtask on another processor. Parallel processors require specific software measures to prevent deadlock. If the operating system provides no deadlock control during execution, then it must be shown, for each process, that no deadlock can occur in the worst-case combination of events. Implementation of a program to ensure that deadlock cannot occur is extremely difficult. The information necessary to determine deadlock is usually not available until operational testing is performed. Deadlock recovery schemes often use rollback to a prior processing step where the state was recorded. A combination of avoidance and recovery is usually sufficient to handle deadlock. If avoidance is used, it must be shown that the system will avoid deadlock under all possible circumstances or that deadlock is extremely unlikely (i.e., the probability acceptably low). If roll-back recovery is used, the probability that the system will miss a real-time deadline due to roll-back must be shown to be acceptably low.

1.4 Reliability Prediction Issues

Modern reliability prediction of fault tolerant systems is a mathematically intensive problem usually requiring computer modeling and analysis tools such as those developed at or in conjunction with NASA Langley (Hybrid Automated Reliability Predictor [HARP], Fault Tree Compiler [FTC], Computer Aided Reliability Evaluator [CARE III], Pade Approximation With Scaling [PAWS], Scaled Taylor Expansion Matrix [STEM], Semi-Markov Unreliability Range Evaluator [SURE], etc.). These computer programs differ from one another in their modeling complexity and solution techniques, but have been shown to be largely in agreement for predicting reliabilities of systems within each of their stated limitations.

The reliability models utilized are limited by their techniques for mathematically describing hardware-type events and solution algorithms. They are also limited by the availability of accurate data for input to the model.

Typically, the failure and recovery of a system of components is described by some type of fault tree diagram, such as those illustrated in figures 1.4-1 and 1.4-2. These modeling examples provide just a small glimpse of the complexity of modern reliability prediction problems.

One of the most direct reliability models is a fault tree. The fault tree shown in figure 1.5-1 is used to calculate the failure probability of the system for raising and lowering aircraft landing gear. This type of model was developed in 1961 by Bell Telephone Laboratories. The technique starts with a single

event and delineates all possible "basic event" paths to that single event. Each basic event must be independent, the characteristic distinction of a random/stochastic process. Each fundamental event must also have a known probability (perhaps calculated from another fault tree). This fault tree, in itself, will not allow for the reconfiguration in a modern fault recovery process, but it is a valuable tool for ascertaining the probabilities and failure rates of other parts of a system.

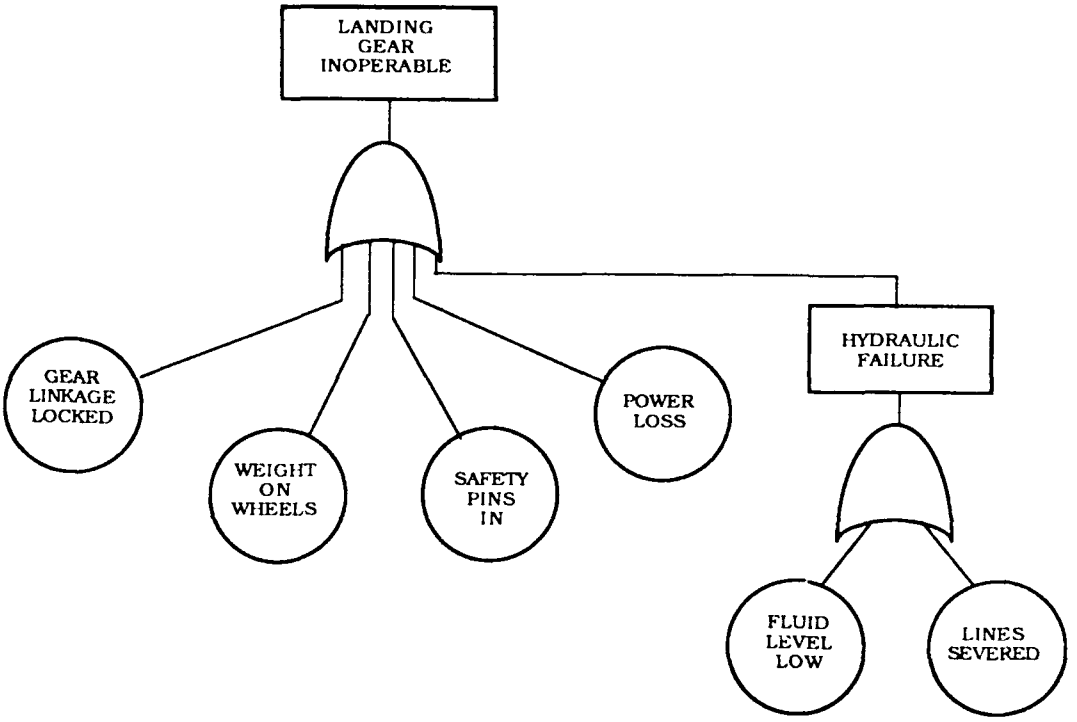
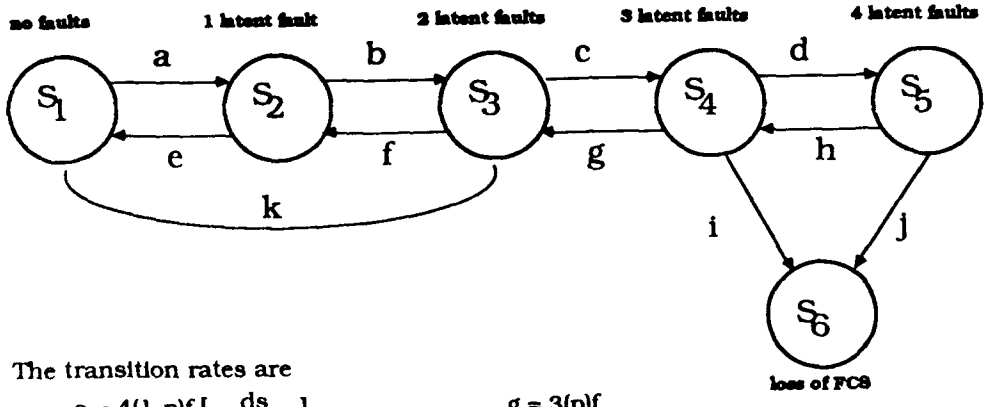


FIGURE 1.4-1. FAULT TREE RELIABILITY MODEL

A Markov model is required to incorporate the fault recovery reconfiguration process. Figure 1.4-2 illustrates a Markov model for a quadruplex Flight Control System (FCS) with latent faults (McGough 1988). The key difference from the fault tree of figure 1.4-1 is the incorporation of multiple end states, thus allowing for reconfiguration paths through activation of redundant components or degradation to simplex states (node 7 in figure 1.4-2). Transitions to each state are characterized by a time rate for failure and recovery. The Markov model becomes semi-Markov if the recovery rates, F(t), are not simply approximated by the common exponential distribution rates used for component failures. However, this adds significant complication to the mathematics required for solution of the model.

Computation of the reliability of a fault tolerant system is more complex than the examples shown in figures 1.4-1 and 1.4-2. The reliability of the overall system is a function of the reliability of the individual components of the system. It is also a function of the fault recovery process used in the fault tolerant system. Individual part reliabilities may be obtained effectively from life cycle testing and materials analysis. This data is fundamental to the failure prediction methods used by the most common component reliability prediction (MIL-HDBK-217E 1986). Including behavior such as fault recovery in the system during the presence of a fault is even more challenging. This data must be obtained from tests run on both hardware and software under controlled fault injection into real and simulated systems. Data obtained in this manner is ordinarily specific to the system under test.



The transition rates are

$a = 4(1-p)f\left[\frac{ds}{as+ds}\right]$

$b = 3(1-p)f$

$c = 2(1-p)f$

$d = (1-p)f$

$e = as + (p)f$

$f = 2(p)f$

$g = 3(p)f$

$h = 4(p)f$

$i = as$

$j = as$

$k = as$

Note that the backwards transitions (e, f, g, h, and k) are the result of repairs due to detection of faults.

FIGURE 1.4-2. MARKOV MODEL QUADRUPLEX WITH LATENT FAULTS

As has been indicated, the failure rate of components is an essential part of the data needed for reliability prediction. Studies have centered around the testing of components in manufacturer's plants and during incoming quality assurance inspections and testing at systems manufacturing facilities. The most comprehensive data is available for the large volume components and originates from the larger manufacturers. Testing is almost nonexistent at small and low volume commercial facilities. Device testing is much more stringent for MIL-SPEC parts.

Logic devices may be tested by generation of test pattern inputs to a device. Their outputs can then be compared to a known correct response. This approach

works well for combinatorial devices, but not so well for microprocessors. Testing of microprocessors has become a significant problem due to the incredible complexity of VLSI designs. Millions of logic gates, sequential and parallel data propagation, and very fast (to the tens of nanoseconds) timing relationships make it virtually impossible to expose the microprocessor to all possible combinations of inputs and outputs. For most applications the level of testing is a direct function of economics.

The inclusion of redundant fault tolerant hardware helps circumvent overall system failure due to the failure of a single component. Studies (Arnold 1973 and Johnson 1989) have shown that system reliability is enhanced through redundancy even with less than 100 percent fault detection; however, the fundamental operation of the fault tolerant system is fault detection and recovery. Therefore, fault recovery data is crucial to the accuracy of the reliability prediction model. One of the significant purposes of controlled fault injection into fault tolerant systems is the procurement of fault recovery data which may be used in the reliability prediction model. Complex and innovative system architectures make it difficult to predict fault recovery behavior based upon history and experiments on other types of systems. Experimental fault injection is usually required to ascertain a specific system's reliability.

1.5 Fault Modeling

Validation of fault toleration systems requires an understanding of faults. This section discusses the origin of various faults and models that have been defined for them.

1.5.1 Origin of Faults

The wide range of faults and where they occur in digital systems performance is illustrated in figure 1.5-1. These faults may be classified in three general areas: design errors, manufacturing errors leading to faults, and faults occurring in-service.

Design errors are the result of improperly translating an idea or concept into an operating system. Gathering information on design errors is difficult because each error occurs once per system development and the errors are usually complex. Design errors have been extensively studied in the software realm due to the increased emphasis on reliable software and the fact that software faults are entirely due to design and coding errors. (In general, coding errors will be identified and eliminated during testing.) Such errors often enter into the design of algorithms where partitioning into separate regions of validity is necessary; for example, with square roots, arc tangent, arc sine, and so on. If sufficient testing is not done between regions, the value at the region boundary can become undefined or very large or small. The effects of these errors are difficult to generalize due to their infrequency and diversity, hence few generalized models are available. These faults may produce little or no effect until just the exact combination of input conditions and time history present themselves.

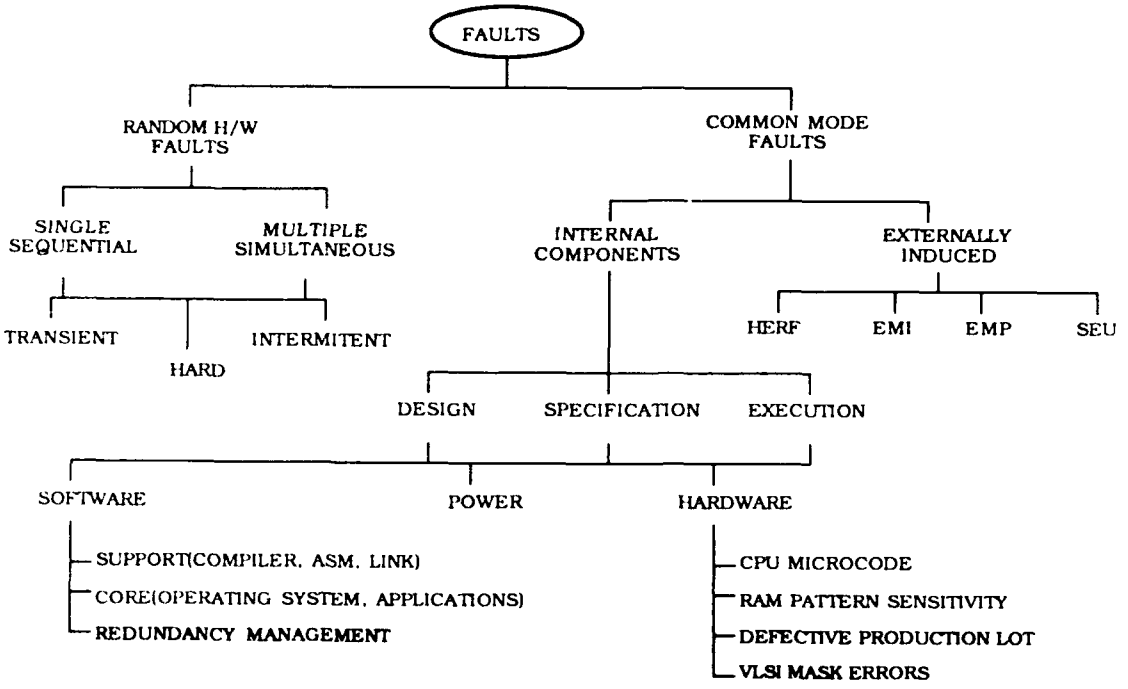


FIGURE 1 5-1. SIMPLIFIED BREAKDOWN OF FAULT TYPES

Manufacturing defects may be introduced by improper processing or by the use of defective materials in the fabrication of the system. Examples of manufacturing defects in semiconductor devices include cracked dies, over or under doping of the semiconductor material, and flaws in the mask. Effects of manufacturing defects on Complementary Metal-Oxide Semiconductor (CMOS) circuit behavior have been characterized. In a manufacturing research program, defective masks were utilized to fabricate several circuits and defective parts examined for faults (Ferguson 1987, Marchall 1985). Results of this study determined that 99 percent of all faults can be characterized by bridging or breaks, while only 50 percent of the faults can be modeled by stuck line faults.

In-service failures are caused by physical processes occurring through normal and abnormal use over the life of the system. Following are examples of CMOS device failures:

- Oxide breakdown due to high electric fields in the insulator, usually in the gate oxide.
- Electromigration, drifting of metal atoms toward the cathode, influenced by high current density in the semiconductor.
- "Hot electron" trapping in the gate oxide in the channel of a Metal-Oxide Semiconductor (MOS) transistor due to high temperature and high electric fields.

- Electron hole pairs or alterations to stored charge due to impacts by cosmic rays and alpha particles.

- Physical failures due to electrical overstress caused by overvoltages, electromagnetic (EM) environment (interference (EMI), radiation (EMR), lightning), and electrostatic discharge.

Other life cycle failures are caused by many other sources including design deficiencies, production techniques, mechanical stress, thermal stress, moisture, and corrosion.

Very little has been published on the distribution of in service faults due to manufacturing processes and the operating environment. Attempts to map physical failures into logical models are not satisfactory. Several failures resulted in stuck-at behavior, while others resulted in parametric faults. An interesting contrast is apparent between in-service and manufacturing operations. The results of one study (Ferguson 1987, reported in Czeck, Siewiorek, and Segall 1989), are shown in table 1.5-1.

TABLE 1.5-1. FAULT EFFECTS: MANUFACTURING VS. OPERATIONS

FAULT EFFECT:	MANUFACTURING STAGE	FIELD
Oxide	Less than 10%	25% - 75%
Metal	30% to 40%	4% - 17%

During in-service operations, the highest failure rate is for gate oxide (25 to 75 percent) and the next highest is for opens and shorts in metal runs (four to 17 percent). Oxide failures result in transistors being stuck off. Metal failures result in opened and shorted lines, and in breaks and bridges. During manufacturing, the most common failure (30 to 40 percent) is due to metalization, with a minor component (less than 10 percent) of faults causing transistors to have stuck-off faults.

1.5.2 Fault Models

Fault models are abstractions of failure mechanisms. They range from the physical device level to the gate, functional, and architectural level. A wide range of fault models have been developed to generate test programs. Fault models at various levels of abstraction, illustrated in table 1.5-2, cover the levels of system detail from network to switch.

TABLE 1.5-2. FAULT MODELS AT DIFFERENT LEVELS OF ABSTRACTION

LEVEL	FAULT MODELS	BASIS	LIMITATIONS
Network	Communication lost, delayed or unordered. Lost nodes.	Abstraction of behavior.	Failure modes are unknown and complex.
Physical Message Switch	Data Change. Message or process lost. Data inconsistent. Time outs.	Abstraction of behavior.	Failure modes are unknown and complex.
Remote Terminal	Data change. Wrong assertion, source, or destination.	Abstraction of behavior.	Based on RT models not implementation.
Functional	Complement or dual function Truth table modification.	Observation? Ad hoc.	Many realizations and fault modes.
Gate	Gate output stuck at 0 or 1, Single stuck line.	TTL and PC board behavior.	Technology outdated.
Switch	New and missing devices, Shorts and breaks (opens), Transistors stuck on/off.	Processing defects.	Simulation overhead, difficult to observe and fault insert.

Switch-level fault models are used for MOS devices where unidirectional logic gate models do not provide the bidirectional behavior of bridging, stuck open, and stuck closed transistors. Switch level models contain nodes connected by bidirectional transistor switches. Faults are modeled as nodes stuck high or low, transistors stuck open or closed, and extra or missing transistors. Low level simulations and models are required because the failure modes of the devices, especially CMOS, cannot be modeled at the gate or higher levels. Switch level simulations are a great aid in the design and generation of test programs, and assessment of test coverage, but the simulation is difficult to use for large systems due to the excessive run time for the simulations.

Gate-level fault models assume that inputs and outputs of gates are stuck at high or low logic levels, but the gates function correctly. These faults are based upon printed circuit boards, Transistor-Transistor Logic (TTL), and pre-TTL logic of the 1960s. The gate-level model is not applicable to MOS implementations since failure modes in CMOS can transform a combinational circuit into a sequential circuit. Furthermore, complex MOS circuit implementations do not map gate lines to circuit nodes.

Studies were conducted to determine the effects of processing mask defects on faults by mapping the defects to circuit fault models. These studies showed that only 50 percent of the faults are representable by gate-level models having a single line stuck-at fault. Further studies showed that the development of

fault models should be independent of the technology and updated as technology advances.

There is continuing research underway on higher levels of abstraction in fault models. Czeck, Siewiorek, and Segall 1989 provides a guide to this literature. This level of abstraction is driven by the VLSI in the electronic circuitry, and the complexity of dealing with testing based upon implementation faults. Some work has been completed laying the ground work for functional testing based upon the possible scenarios of failures occurring within the control section, data section, or data storage section of a generalized microprocessor. These models were independent of the implementation, hence they did not contain a fault library. The models were useful, however, in presenting a methodology for developing a set of test procedures for microprocessors.

1.6 Fault Tolerant Techniques

The degree to which a system can recover from faults depends critically upon the mechanisms for detecting and isolating faults. A careful distinction must be drawn between those techniques which are approximate and those which achieve near perfect fault detection and recovery. The following discussion is based upon a reference document from Charles Stark Draper Laboratory (CSDL) (1986).

Consider, for example, a dual channel system that relies upon bit-for-bit comparisons of the two outputs for fault coverage. This approach provides nearly perfect fault detection coverage. If a fault in one of the channels affects the correct operation of the channel output, then the output of that channel must differ from that of the other correctly operating channel. The latency between the occurrence of the fault and its appearance as an output error depends upon how frequently the malfunctioning hardware is exercised by the operating software (control routines and diagnostics). A fault will be readily detected when miscompare occurs, but it will not be obvious which channel is correct.

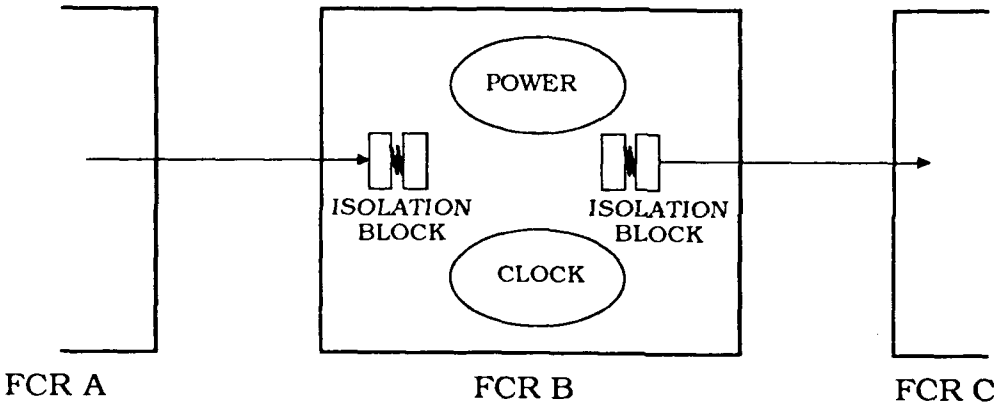
Diagnostic software routines may be run to isolate the channel which is faulty in a dual-channel system, but the coverage of the self-test software is often quite limited in its ability to detect faults. An example of a troublesome fault that is difficult to isolate is a pattern memory error in one bit position of the system memory. This fault may appear as a "dying one" (i.e., a one that is surrounded by all zeros occasionally turns into a zero). To catch such a fault requires a large number of runs with a great number of patterns on the memory. The likelihood that a "dying one" fault will be detected in a diagnostic test is not very large, and certainly not close to one hundred percent.

The addition of ad hoc solutions such as parity checks can help to isolate the memory pattern faults. Special circuitry to disassociate faults in the rest of a system adds considerable overhead in hardware and increased complexity. Furthermore, there is no technique for adding self-tests to logic circuits.

Other common techniques, which result in less than perfect coverage, include temporal checks, integral checks, and diagnostic checks. Watch dog timers have been used to catch certain hardware faults as well as programs stuck in infinite

loops. Integral checks are built into the basic number representation. Parity, Hamming codes, and Bose-Chaudadhary codes are all examples of programs written specifically to exercise the hardware. Examples of this type of check include Read-Only Memory (ROM) checksums, voter tests, and processor opcode tests. These methods add to the coverage of faults, but each one addresses only a limited number of possible faults, and unless all fault types are considered, the system will not achieve high fault coverage.

Current thinking on fault tolerant computer systems has identified four requirements that must be met to achieve a high degree of fault coverage. Any process which is critical to the operation of the system must be replicated. Critical processes must be correctly mapped into FCRs, such as defined in figure 1.6-1. There must be a sufficient number of FCRs and communication channels between them. And finally, there must be an "exact consensus" (bit-for-bit and synchronous, to be discussed later) among replicated processes. These requirements, described more fully below, are applicable for all processes, whether application program or operating system code.



FAULT CONTAINMENT REGIONS (FCR)
REQUIREMENTS:

1. INDEPENDENT POWER
2. INDEPENDENT CLOCKING
3. ELECTRICAL ISOLATION
4. PHYSICAL SEPARATION

FIGURE 1.6-1. FAULT CONTAINMENT REGION REQUIREMENTS

Most critical real-time control applications demand uninterrupted operation. Effects of faults must be masked by detection, isolation, and correction. These three processes must be mapped into three independent FCRs. In general, to mask the effects of f simultaneous faults, the nominal requirements call for (2f+1)

FCRs. However, this nominal requirement is not sufficient when the requirements for fault isolation are considered.

Fault isolation imposes additional requirements. The isolation of a faulty channel may be accurately determined for any output if the channels in the voting scheme are synchronized and have bit-for-bit agreement under normal, no fault, operation. Two processes which are initiated identically, receive identical inputs, have no failures, and operate on the inputs in the same way, are said to be congruent processes.

Congruence allows for a precise definition of the process of fault detection and isolation:

- Fault Detection - Two congruent processes not in bit-for-bit agreement indicate the presence of a fault.

- Fault Isolation - A process not in bit-for-bit agreement with the majority vote is the faulty process.

The concept of congruence has greatly simplified our thinking about fault detection and isolation. Since the processors are synchronized, the majority vote for a congruent process requires a simple truth table to determine the correct output and to isolate the faulty processor.

Incongruent processes need more complex fault detection and isolation algorithms and circuitry. These unsynchronized processes, having only approximately equal outputs, must rely on having "windows of agreement" or thresholds. In this case, fault detection coverage is a function of how precisely one can define these thresholds. For most dynamic processes these thresholds are a function of inputs and outputs. These parameters change with time and the operating mode of the systems. The use of application process driven thresholds for fault detection and isolation put a serious and uncalled for burden on the applications programs to ensure fault tolerance in the system.

Incongruent processes have been used successfully for tasks such as flight control, where computers were used as a one-for-one replacement of the analog counterparts. These implementations suffer from all the ills discussed earlier, including low fault coverage and excessive involvement of the applications programmer in redundancy management. Moreover, they also exhibit a high false alarm rate. These inexact consensus techniques will not work at all in more complex applications that require real-time fault isolation and reconfiguration of distributed system elements. Such systems communicate in abstract messages that have no semantic meaning (like actuator commands) if their redundant counterparts are only approximately equal.

Thus in the final analysis, fault detection, isolation, and masking need congruent processes. Achieving exact consensus in congruent processes having f faults, requires (3f+1) FCRs. This is more severe than stated earlier. In addition, processes must go through an algorithm for consensus that requires (f+1) communication rounds over links providing at least (2f+1) disjoint paths. For example, to tolerate a single fault there must be a least four FCRs, two communication rounds, and three disjoint paths.

This more precise mathematical basis provides a proof of fault tolerance from any arbitrary failure mode. This is driven from a need to account for more than Stuck-At-One (S-A-1) or Stuck-At-Zero (S-A-0) faults. A particularly difficult mode is one in which one receiver detects a one while the other receiver detects a zero. Such failures are termed Byzantine failures and can occur at sensor signal level, logic level, and even at program levels. For example, a redundancy management program in computer A can send the message to computer B that computer C is bad, while it tells computer C that computer B has failed. Although such modes may appear to be remote, they are in fact not unlikely. At least one in-flight failure of a triplex system has been traced to such a Byzantine fault and the lack of safeguards against such faults.

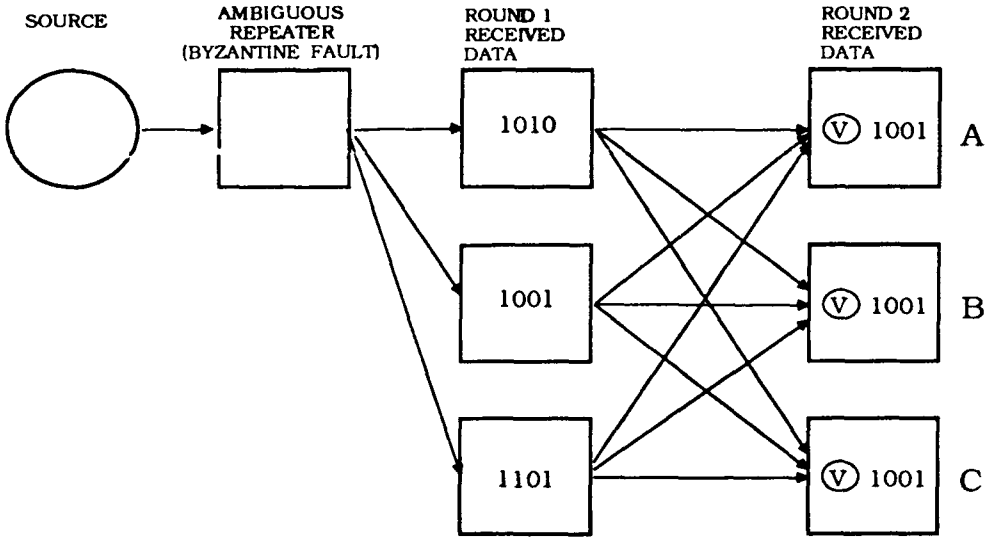


FIGURE 1.6-2. ARCHITECTURE FOR CONGRUENT TRANSFER FROM SIMPLEX SOURCE

An architecture that meets all the requirements for exact consensus in the presence of Byzantine faults is shown in figure 1.6-2. This diagram shows six FCRs that are interconnected by three disjoint paths and go through two-round communication protocol. Data is distributed, in each channel, to voters that produce majority voted outputs. The voters perform a two-out-of-three vote on a bit-for-bit basis. The least significant bits of the three values are voted to produce the least significant bit of the majority output. The next order bit in the three words is then voted to produce the next bit of the output. This process continues until the most significant bit is voted to produce an output word that is the bit-for-bit majority of the three inputs. In the diagram, a Byzantine fault in the form of an ambiguous repeater is postulated in the channel that is distributing the value of the sensor attached to that channel. After the two-round exchange and the bit-for-bit vote, it is evident that all correctly operating channels have identical or congruent values.

The CSDL Fault Tolerant Processor (FTP) architecture for implementing the requirements for fault containment and consensus voting illustrates these considerations. A picture of that fault tolerant implementation is shown in two views: a functional view shown in figure 1.6-3, and a physical layout shown in figure 1.6-4.

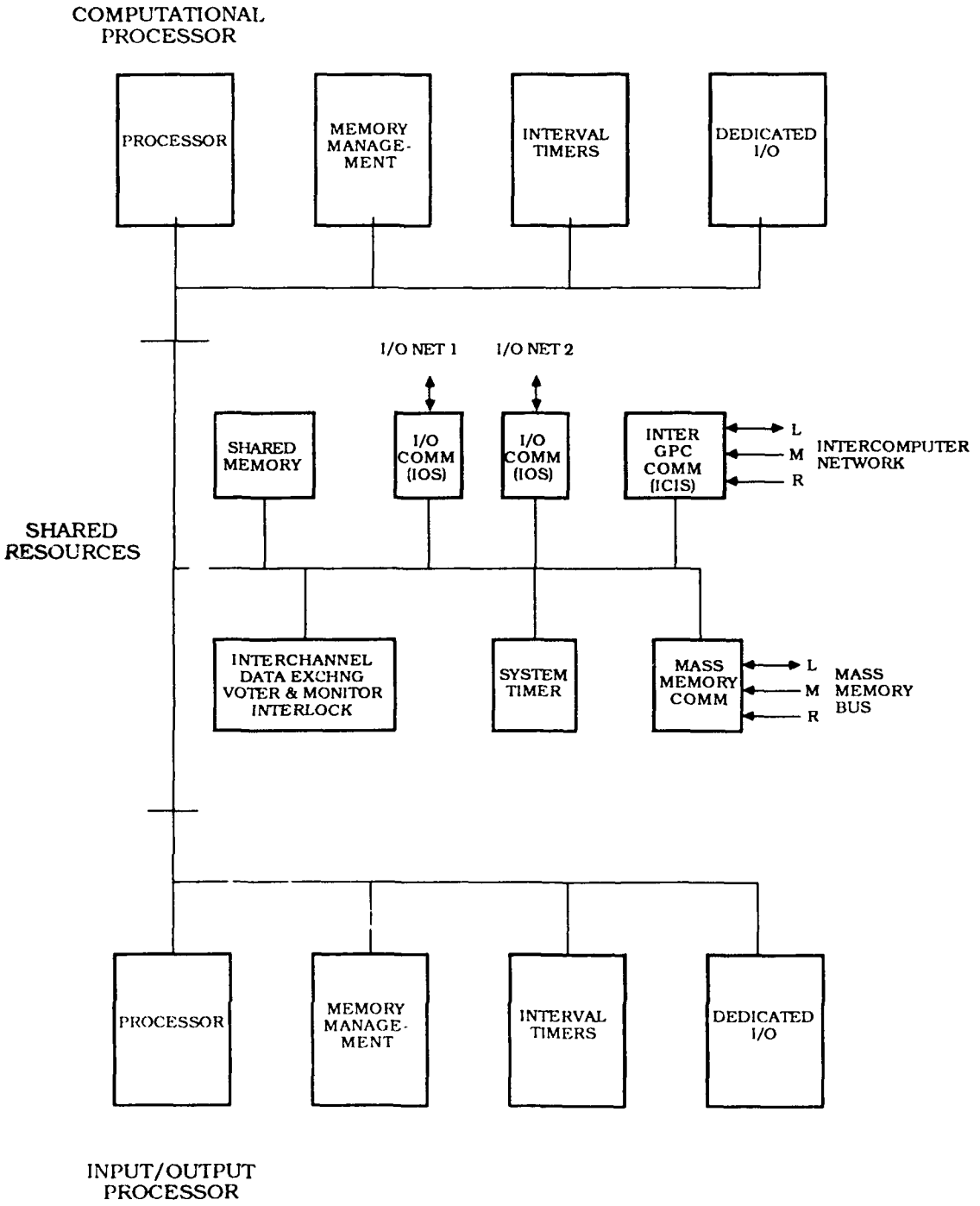


FIGURE 1.6-3. FAULT TOLERANT PROCESSOR - FUNCTIONS

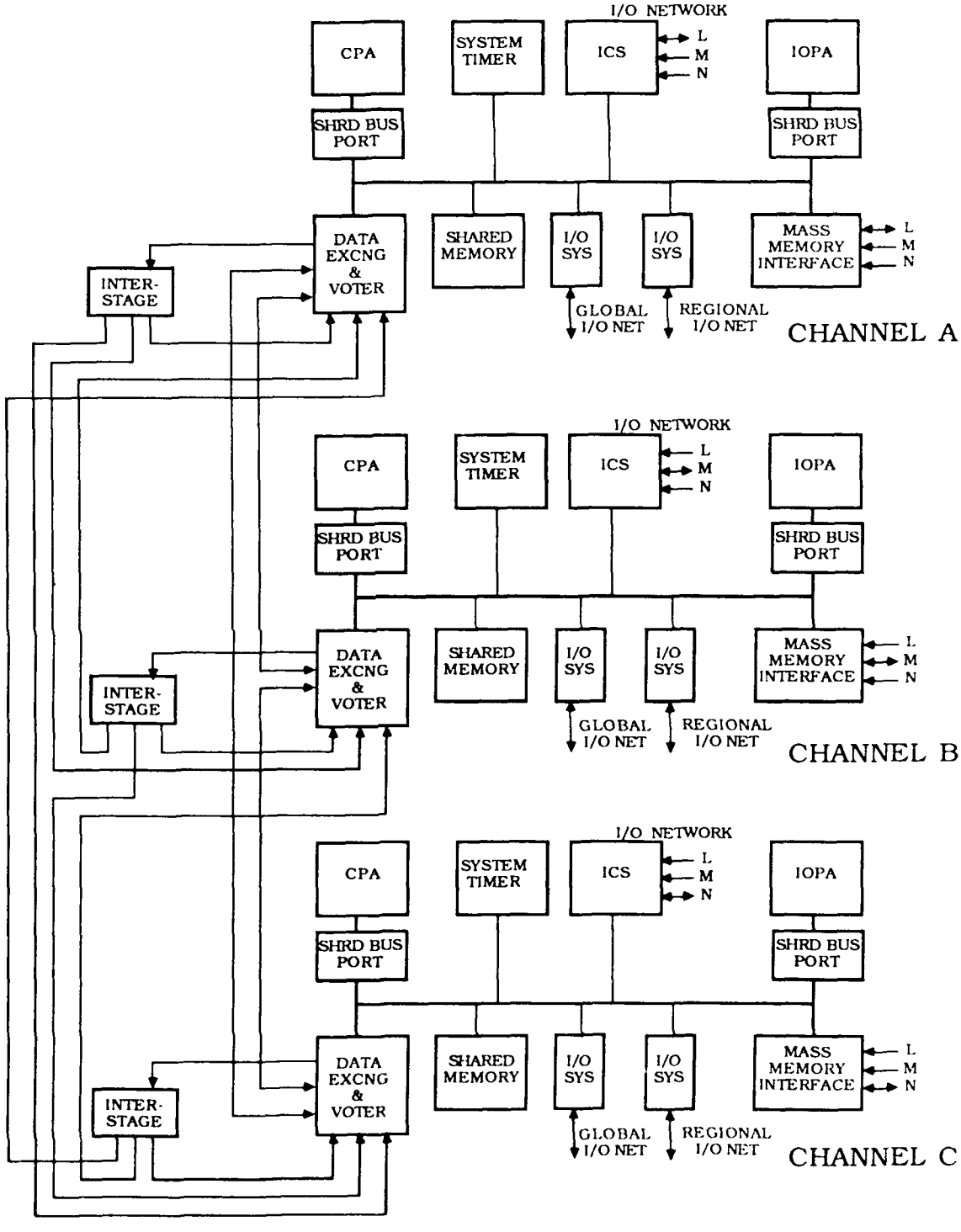


FIGURE 1.6-4. FAULT TOLERANT PROCESSOR - PHYSICAL VIEW

2. FAULT INJECTION SIMULATION AND TEST METHODS

Simulated faults are needed to demonstrate that an FT system response to faults is appropriate. Measurements are needed to demonstrate faulted behavior of the systems. These measurements also give added confidence that the predicted reliability, determined from coverage and component failure modeling, is correct. Furthermore, these tests validate that a system's performance under faulted conditions meets the need for safety and reliability of aircraft systems.

A typical reliability requirement for life critical operations is 10^{-9} per hour of operation. This translates into approximately 10 failures per million years per unit. This is about five orders of magnitude greater reliability than today's non-redundant systems. More importantly, for the purpose of demonstrating the reliability, life cycle testing will not be possible without simulation of the effects of faults.

Life cycle testing offers realism, but the rate of component failures is about one per 1,000 hours of operation per system. Fault insertion speeds up the rate at which synthetic faults occur. The use of inserted faults is needed to allow testing to cover all possible fault types and locations. For example, a small board consisting of 50 packages, each having 20 pins, has a possibility of 1,000 different pin-level (hardware) faults. This is without considering time dependence or sequence faults. Software faults must also be considered as the software becomes more complex. The number of potential software faults for a small operating system is also in the tens of thousands and time dependent as well.

Much work has been done on simulation and test methodology for insertion of faults in digital systems. (For details refer to McGough and Swern 1981; Benson, Mulcare, and Larsen 1987; Padilla 1988; Baker, Mangum, and Scheper 1988; and Migneault 1988.) A typical fault injection test setup involves the elements shown in figure 2.0-1. The faulted behavior is measured comparing faulted processor responses against those of an unfaulted processor. Measurement parameters include faults that are inserted and found, faults that are inserted and not found, and no-fault situations that are identified as faults.

A great deal of work is still needed in this area. The variation in test approaches ranges from entirely simulated software faults to injecting faults on real hardware operating under simulated flight conditions. The modeling accuracy, assumptions required, and interpretation of results also varies among the different methods. NASA, more than any agency, has developed the tools and techniques for fault insertion and testing to validate FT systems reliability.

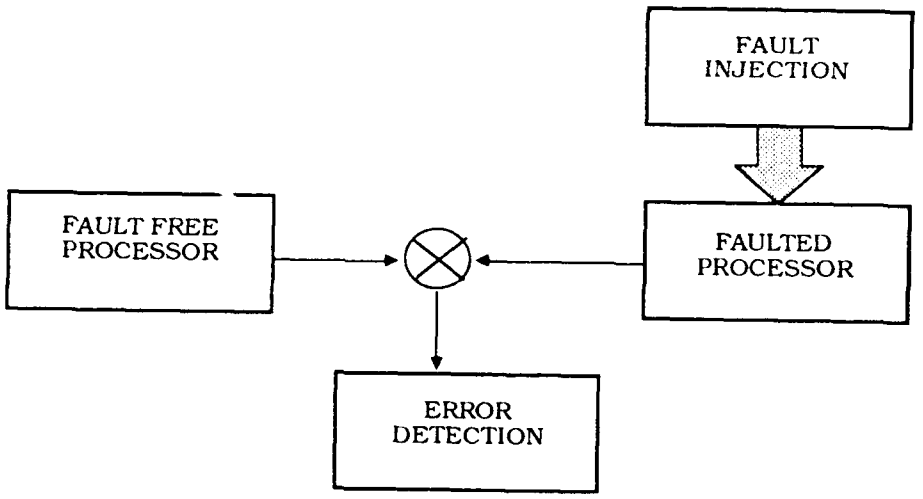


FIGURE 2.0-1. FAULTED PROCESSOR PERFORMANCE MEASUREMENT

2.1 NASA FT System Research Summary

Validation is the process of demonstrating that a system meets its specifications. For high-reliability systems, the specifications contain reliability requirements that necessitate operation under fault conditions. To demonstrate these extreme conditions, prediction methods must be used to determine nominal behavior of fault detection, isolation, and recovery schemes. Complementary to these methods are experimental methods, such as fault insertion, which are well suited to support modeling and analysis. Experimental methods are extremely useful when too many assumptions are needed in the analytical models or when analytical models have too much uncertainty.

Recent NASA research has emphasized fault injection methods for testing fault tolerant systems. The establishment of the Avionics Integration Research Laboratory (AIRLAB) at the NASA Langle Research Center provides a capability for government, industry, and research oriented institutions to investigate the faulted behavior of digital systems. NASA Ames is also involved in research in this area through efforts at the Reconfigurable Digital Flight Control System (RDFCS) Facility. Efforts recently completed at NASA Ames include development of the Fault Insertion and Instrumentation System (FIIS) to measure behavior of FCSs. (This work was sponsored by the Federal Aviation Administration (FAA) Technical Center.)

Several approaches have been developed for application throughout the system life cycle from design to deployment (Carter et al. 1989). Much work remains to be accomplished. Some of the methods are at different stages of development ranging from initial investigations to nearly final form (well understood and

ready for engineering application). A summary of methods and application areas is given in table 2.1-1.

TABLE 2.1-1. VALIDATION AREAS FOR THE DESIGN PROCESS

DEVELOPMENT LEVEL	ABSTRACT DESIGN PROOFS	ANALYSES	CONCRETE TESTS
Architecture	Prove Architecture Against Requirements	Reliability and Error Rate Markov Models	Design Review High Level Simulations
Implementation	Prove Implementation Against Architecture	Fault Tree Analysis	Simulation and Emulation of Hardware
Realization	Prove Realization Against Implementation	Failure Modes and Effects Analysis	Support Assumptions from Analysis. Fault Insertion

2.1.1 Architectural Design Methods

In the requirements and specifications phase, the tools needed include simulation and modeling for the hardware and software. These tools provide data for studies and analysis including conceptual design, hardware/software trade studies, architectural design options, and "what if" parameter bounds.

2.1.2 Detail Design Methods

In the design phase, little hardware is available for testing. During this time the architectures of redundant processors, operating system software, and some applications are partially designed. Many portions are in process. The tools needed for this phase include emulation and simulation of the software and processor architecture.

2.1.3 Developmental Phase Methods

In the development phase, the software is better defined and available for testing. Hardware is becoming available in prototypes. Everything may change as this phase continues. Tools needed for this phase include in-depth emulation and simulation for software, and prototype evaluation tools for hardware items such as error detection and communications.

2.1.4 Systems Implementation and Evaluation Methods

In the implementation and evaluation phase the first items are completed. The tools needed include methods for inserting faults directly into the system hardware, and methods to exercise the special capabilities built into the hardware.

2.2 Fault Insertion Methods

Four methods for insertion of faults are in use. The advantages and disadvantages of each are shown in table 2.2-1. These four methods (Czeck, Siewiorek and Segall 1989) are based upon:

- Software simulation using fault insertion by code modifications or special functions of the simulation engine.
- Hardware emulation using hardware representative of the system under test, such as engineering prototypes, for study.
- Fault emulation imitating fault behavior through software control of the hardware or special capabilities built into the hardware.
- Physical fault insertion inducing faults through special hardware directly into the actual system under test.

TABLE 2.2-1. FAULT INSERTION METHODS

	FAULT INSERTION METHODS			
	SOFTWARE SIMULATION	HARDWARE EMULATION (BREADBOARD)	FAULT EMULATION	PHYSICAL FAULT INSERTION
Advantages	Access to system at any level of detail. Fault types and control are unlimited.	Representative hardware with favorable access and monitoring.	True hardware and software in use.	True hardware and software in use.
Disadvantages	Simulation time explosion. Lack of tools limit ability.	Implementation and other parameters will change with deployed system.	Fault types are limited.	VLSI limits access and monitoring points. Task is difficult.

These methods have been developed and used extensively for at least ten years. They are used for measuring fault coverage, error detection coverage, and other fault tolerance performance parameters. Measurement of these parameters requires special methods such as fault insertion.

Fault insertion has been used extensively for a number of purposes including test coverage, generation of fault dictionaries, study of error detection, and system evaluation. Table 2.2-2 summarizes these studies.

TABLE 2.2-2. FAULT INSERTION EXAMPLES

SOURCE	TARGET	METHOD/LEVEL	GOAL
Avizienis and Rennels '72	JPL Star breadboard	Permanent and Transient/Pin (Gate) Level	Estimate of Detection, Recovery Procedures, and Coverage
Goetz '72	ESS Microstore	Simulation/Gate Level	Detection and Coverage Measure
Courtois '79	MC 6800	Simulation/Opcode (RT) Level	Detection Time
Decouty Michel and Wagner '80	GORDINI: Fault Tolerant micro	Physical/Pin (Gate and RT) Level	Tool and Methodology Development
Kurlack and Ch~bot '81	GE MCP-701 CPU	Physical Faults with FMECA analysis	Evaluation of Watchdog Timer
McGough and Swern '81	Bendix Simplex BDX-930	Simulation/Gate and Pin Level	Coverage Measurement
Lala '83	FTMP Engineering Model	Physical/Pin (Gate) Level	System and Coverage Evaluation
Yang, York, & Siewiorek '85	iAPX 432	Fault Emulation/ Memory Words (RT)	Coverage Management
Schuette et al. '86	MC 68000	Transient, Physical/ Bus (RT) Level	Evaluation of Error Detection Techniques
Czeck et al. '87	FTMP Engineering Model	Fault Emulation/ RT Level	Methodology Study
Finelli '87	FTMP Engineering Model	Permanent Physical/ Gate Level	Fault Recovery Distributions

By far the two most common methods involve computer simulation of hardware faults and physical fault insertion. Fault simulation has been used at all levels for the fault models discussed in section 1.5. Physical fault insertion has been used to determine fault coverage of test programs, fault latency, and fault detection efficiency. Experimental agreement among the different techniques provides reasonable confirmation of the validity of these test methods. However, there is considerable difference in fault detection coverage between tests with models at the gate level and pin level of detail. The selection of injection points and the number of points tested also affects the measurements.

Since system fault detection is a critical parameter, NASA initiated a pilot study to investigate the methodology for measuring the fault latency of digital computers. The methodology consisted of simulating a 1,000 gate equivalent computer in a host Cyber 173 computer. The simulated computer was a paper design and was referred to as the hypothetical machine. This hypothetical machine was simulated at the gate level. Actually two copies of the hypothetical machine executed identical copies of code in synchronism although one machine received a fault at the onset of the simulation. Detection or non-detection was determined after the non-faulted processor completed its execution. At that time, the computational results of the two machines were compared bit-for-bit; any difference constituted detection. If no detection occurred, the code input variables were randomly altered, and the programs were repeated for the same fault. This scheme was repeated for up to eight executions for the same fault. If a detection occurred in eight repetitions or fewer, or no detection occurred after eight repetitions, then a new trial began. This over-all process was repeated for up to 211 faults randomly selected throughout the circuitry of the computer. The 211 faults were selected as a function of the piece-part failure rates and distributed across the nodes of the gates. The latency time (i.e., time to detect) is expressed in terms of code executions or repetitions.

Results of the pilot study were both surprising and disturbing. Using six different programs ranging from simple fetch and store to a very complex linear convergence scheme, the pilot study indicated that only 50 percent of the induced faults were detected after eight repetitions for each of the six programs. Figure 2.2-1 provides a typical picture of these results.

Studies conducted by McGough and Swern (1983) on the Bendix 930 FTP showed quite significant differences between gate-level and pin-level fault injection. Preliminary results obtained from gate-level fault simulation, using a hypothetical simple computer, showed that only 50 percent of the induced faults propagate to an output port where detection is possible. This pessimistic and surprising result prompted the design and implementation of several follow-on studies using realistic avionic computers.

The same set of experiments were applied to the minicomputer with the addition of realistic flight control code. The results were once again surprising: the percentage of undetected faults was about the same for all the programs simulated. Only about 50 percent of the injected faults were detected when different instruction sets, executing in two different machines (one hypothetical and one real), were used. More complex code tends to alter the shape of the

histogram, so that almost all the faults are detected in the first execution of the code. The latency time decreases somewhat with increased code complexity, but not the percentage of faults detected.

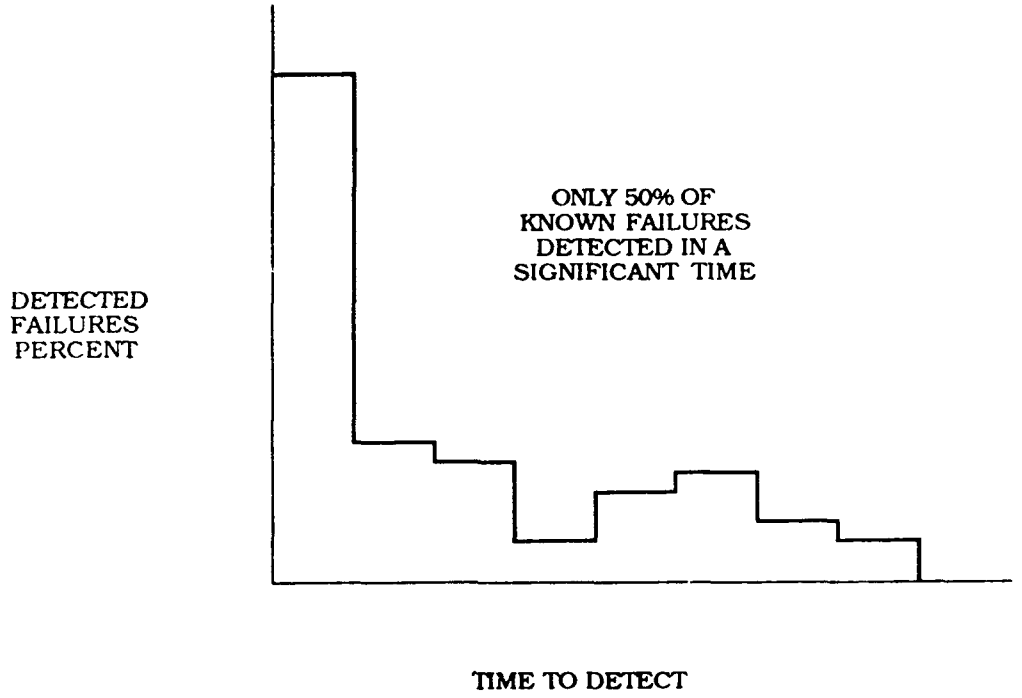


FIGURE 2.2-1. FAULT LATENCY DATA

More recently, experiments were conducted by the FAA Technical Center at NASA Ames (Benson, Mulcare, and Larsen 1987) on a Flight Control computer operating a simulated closed loop flight control. The computer configuration used was two CAP-6 processors in a dual configuration. This configuration is part of the dual-dual FCS used for flight control of the Lockheed L-1011 Tristar. An enhanced fault injection and data analysis system, based upon the CSDL Fault Tolerant Multiprocessor (FTMP) test setup, was used to control the experimental fault injection, and to collect and analyze the data. This computer was a much larger processor than that used for the Bendix study. The FCS program had 11,000 words, and the FCS computer had approximately 1,200 pins. Faults were injected on pins stuck at zero or one; approximately 2,500 faults were injected. The finding was that a much higher percentage of faults were detected; nearly 96.5 percent of all the injected faults were detected. These results were consistent with the earlier CSDL measurements on the FTMP. Additional discussion of these experiments is in section 2.6.3.

The following subsections discuss the four fault injection and test methods, and provide further examples and discussion of these issues.

2.3 Software Simulation Fault Insertion Methodology

2.3.1 Method

For this method of testing, the system digital logic circuits are simulated at the gate level-of-detail using high speed, specially designed, fault simulations. These simulations are like those for performance analysis, but stripped down to provide higher speed and specific outputs for fault detection and analysis. Logic levels are limited to zero or one. Little or no information is available for prediction of the system performance, such as speed and throughput.

Gate level-of-detail logic simulators were among the first tools for investigating the behavior of fault tolerant processors. Starting in 1977, NASA Langley initiated pilot studies on methodology for dealing with faults in digital systems. These efforts utilized a Cyber 173 to model a 1,000 gate equivalent processor. This was a paper design for a hypothetical machine. Following the pilot studies, a Gate Logic Software Simulator (GLOSS) was implemented to study the behavior of the Bendix 630 miniprocessor, a 5,000 gate machine. Subsequent to the Bendix study, it was realized that a more general capability could be useful for other processors. Several interim and improved versions of the GLOSS were developed: S-GLOSS by Stevens Institute of Technology; B-GLOSS by Bendix; and IGGLOSS, an improved version developed at Langley.

The current version being developed by Research Triangle Institute is called the Generalized Gate-Level Logic System Simulator (GGLOSS). In this version detailed gate-level simulations are run with faulted and unfaulted processors. The results are checked by comparison monitoring at critical outputs in the simulations.

Simulations may be run with faults injected at the gate level, pin level, or component level of detail. Since the entire processor is modeled at the gate level, faults may be injected at the opcode level. The gate faults are injected by holding the output node at zero or one. Pin-level faults are injected by holding the pin or terminal at zero or one.

2.3.2 Application

This method is most useful in early stages of development where no hardware is available, or for small enough computers so that a complete gate-level simulation may be processed.

2.3.3 Example Results

Researchers at NASA Langley began to study development of highly reliable prediction programs (B-GLOSS) for application to ultrareliable fault tolerant digital systems in the 1970s. One of the first efforts was modeling and analysis of the earlier Bendix 930 pilot study results. These new efforts were designed to test the validity of the pin-level injection result obtained from physical fault injection. It was desired to confirm that similar results could be obtained from a real processor executing practical software.

The pilot study results were tested in three phases using simulations of the Bendix 930 miniprocessor, a 5,000 gate equivalent computer. In the first phase, six application programs were coded using the primitive instruction set of a hypothetical machine: load, store, add, subtract, and branch. The next phase allowed the six programs to be coded using the rich instruction set of the Bendix 930. Finally, comparison monitoring detection was measured for an FCS code in lieu of the six programs.

The surprising outcome of these simulations is typified in figure 2.3-1. The results of all six programs follow the same trends. The percentage of undetected faults is about the same for all the programs, instruction sets, and two different machines: about 50 percent were undetected. For the more complex code, the shape of the histogram changes: most of the faults are detected in the first execution of the programs. The latency time decreases with increasing code complexity, but not the percentage detected.

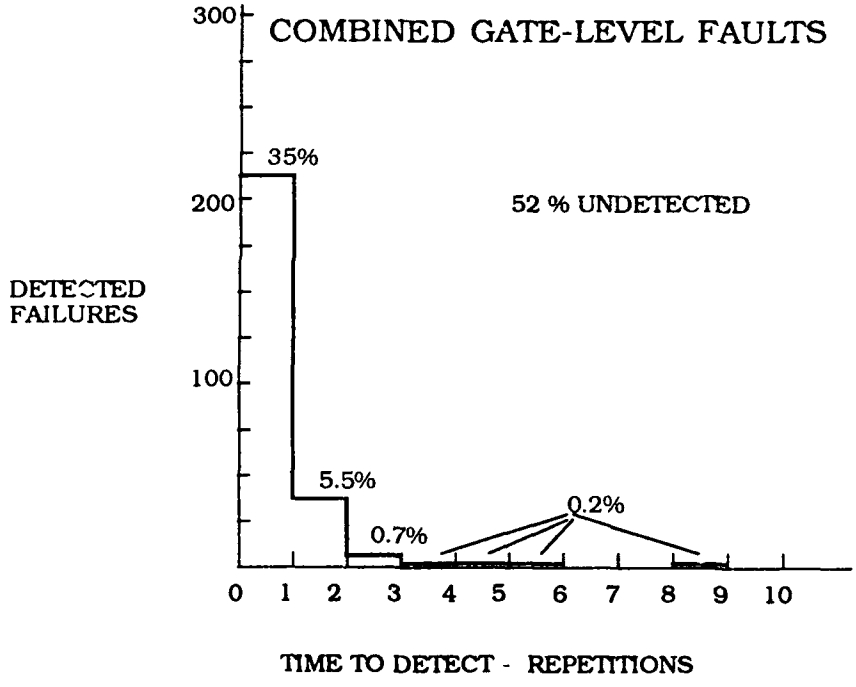


FIGURE 2.3-1. FAULT LATENCY DISTRIBUTION - GATE LEVEL FAULTS

The next set of experiments with pin-level fault injection produced more disturbing results. When the simulations were run with faults injected at pin level, but using the same programs and gate descriptions, the results shown in figure 2.3-2 were obtained. The notable difference is that the level of detection increases to about 62 percent of the injected faults.

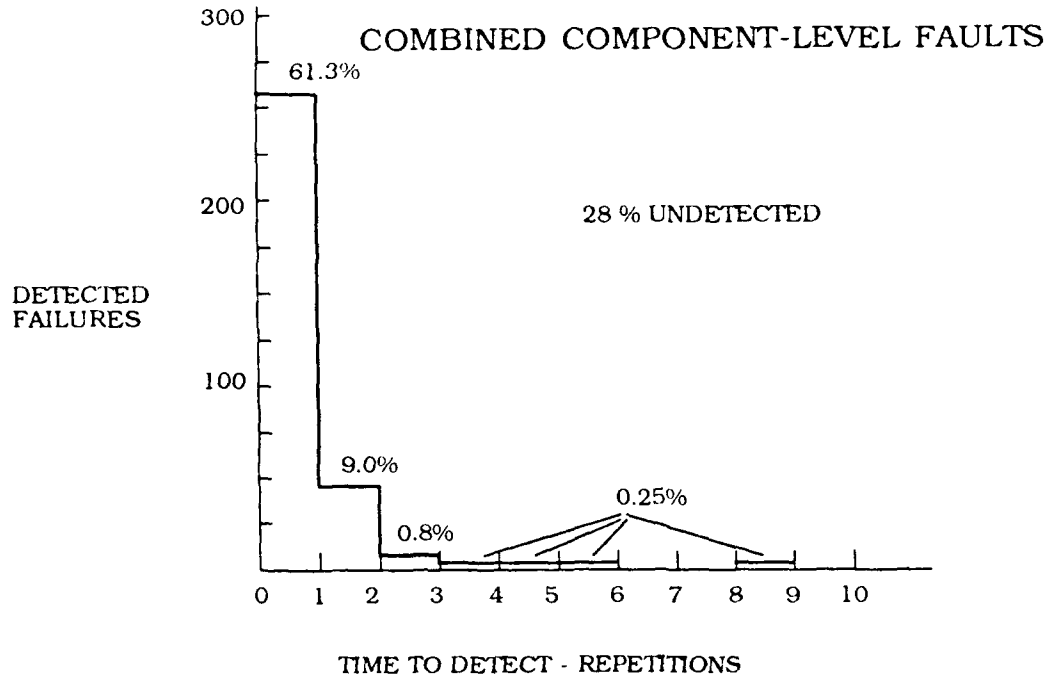
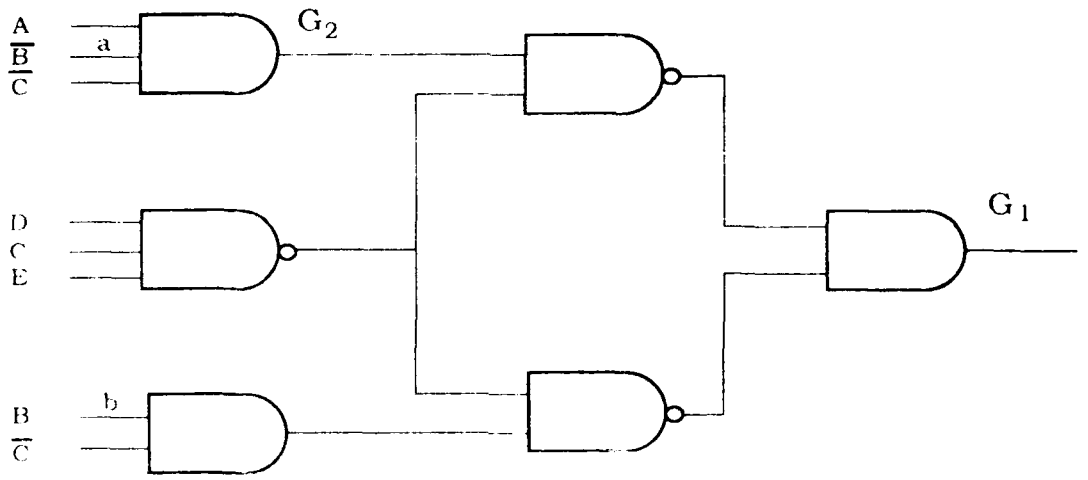


FIGURE 2.3-2. FAULT LATENCY DISTRIBUTION - COMPONENT PIN LEVEL FAULTS

Investigation of the difference between faults at the gate level and faults at the pin level determined that there is a class of faults that can never have an effect on the system and will never be detected. An example of these faults is a stuck-at fault located at a unused flip-flop output. The study found 16 percent of the faults were injected at such "don't care" locations.

An example of digital logic, shown in figure 2.3-3, may clarify the difference between pin and gate faults. The eight device terminal pins are A, $\bar{A}$, B, $\bar{B}$, C, $\bar{C}$, D, and E. Gate faults may be injected at any of the six gate output nodes, but only one is a pin. The "don't care" pin faults are line-a S-A-1. If line a is stuck at one, then line-b S-A-0 cannot be detected.

As an extension to the pilot project, the GLOSS simulation method was applied to a 2,000 instruction self-test program for the Bendix 930 processor. While the simulation exercised the program, faults were injected at the gate level and subsequently in another experiment at the pin level. The design goal of the self-test program was five percent coverage. For the experiments run with gate-level fault injection, the undetected faults were about eight percent of the detectable injected faults. With faults injected at the pin level, only two percent of the detectable injected faults were undetected. These data have been corrected for the 16 percent of "don't care" faults in this system.



THE SET OF TESTS $A\bar{B}CD$, $\overline{ABC}$, $ABCDE$, BCD, $AB\overline{CD}E$

CAN DETECT ANY DETECTABLE FAULT IN THE CIRCUIT G_1

NOTE THAT:
1. FAULTS ON PIN b ARE DETECTED BY $AB\bar{C}$
2. BECAUSE OF REDUNDANCY, THE FAULT PIN a: S-A-1 CANNOT BE DETECTED
3. IF THE FAULT PIN a: S-A-1 EXISTS, THEN FAULT PIN b: S-A-0 CANNOT BE DETECTED.

FIGURE 2.3-3. LOGIC DIAGRAM ILLUSTRATING DETECTABLE FAULTS

Work is still underway on the GGLOSS simulator and validation plans for it were recently developed (Bavuso and Miner 1989). GGLOSS has the capability to simulate S-A-1 or S-A-0 at any point in the circuit. Several innovative methods are used to eliminate the overhead in the usual event driven simulation. Sequential and combinatorial circuit elements can both be modeled. The combination of features gives GGLOSS the capability to simulate approximately 10^5 gate evaluations per MicroVAX II Central Processing Unit (CPU) second, while allowing users to monitor the effects of simulated faults.

2.3.4 Limitations

The types and numbers of faults are limited to logic levels in most implementations. Faults must be modeled as gate nodes or logic lines stuck at zero or stuck at one. Faults are inserted one at a time in an order determined by an experiment plan. Logic lines may be held for several cycles or inserted momentarily.

The selection of faulted lines or devices is often at random. This imposes a limitation on the measured coverage of possible faults. A great many "don't care" faults may be reported. Recent workers emphasize measurement of critical faults, such as detection and recovery times for reconfiguration processes (Padilla 1988, 1989).

2.4 Hardware Emulation Fault Insertion Methodology

2.4.1 Method

This approach uses special purpose digital hardware to emulate portions of the system under development. (Frequently the system under development is still on a breadboard.) The concept is similar to the use of emulators in the design and test of software using an emulation of a future microprocessor design. The instruction set of a new processor can be emulated in code using the instruction set of an existing microprocessor. In this way operating system software can be developed and tested while the new processor is still under development.

In the fault insertion emulation approach, algorithms for fault detection and recovery can be emulated by a special purpose computer consisting of hardware and software for emulating a wide variety of digital logical elements. The operation is based upon high-speed emulation of algorithms describing the digital logic at various levels of detail down to the gate level.

The technique and its concept, depicted in figure 2.4-1, are independent of any particular hardware or software implementation. Migneault (1988) contains a detailed discussion of this technique.

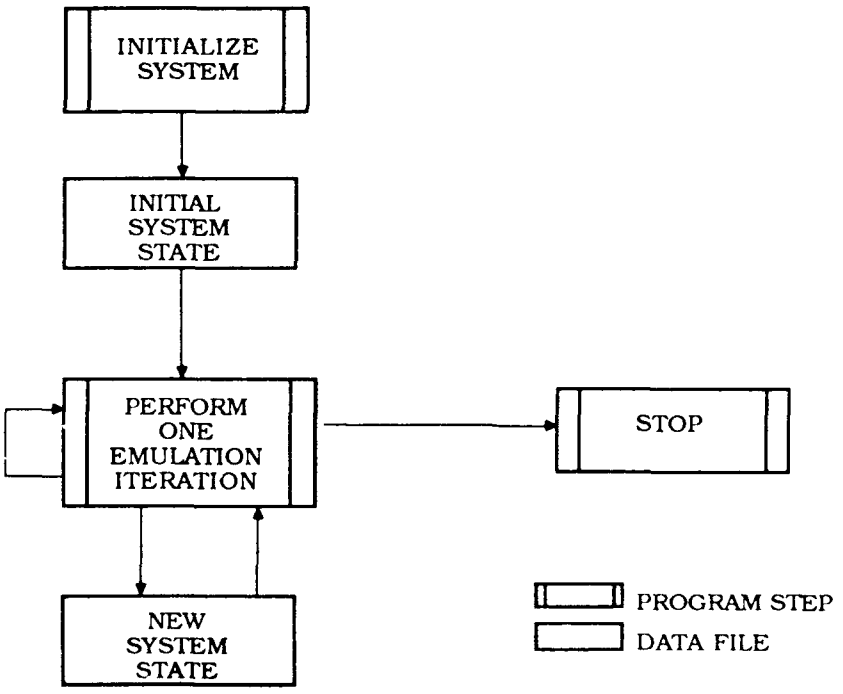


FIGURE 2.4-1. FAULT EMULATION COMPUTATION TECHNIQUE

2.4.2 Application

Fault tolerant digital systems are implemented by first setting reliability goals, and then designing the hardware and software systems as algorithms for fault detection and recovery. The tolerance to faults is usually in the form of redundancy components and algorithms which can detect faults and reconfigure the components in case of faults.

2.4.3 Example Results

This approach was developed at NASA Langley Research Center. In this approach (Bavuso and Miner 1989) an algorithm was developed to emulate any network of logic gates, flip-flops, and tristate devices. This algorithm is independent of any particular piece of hardware being emulated. A description of the target system is fed into a translator which converts the description to a form that the emulator can process. The processing of this representation of the target hardware by the software-implemented algorithm consists of the gate-level emulation of the target hardware. During emulation, faults can be injected and their effects studied.

Several unique features were incorporated in the Langley emulator system. Emulation speed is important because the target system must be run for many iterations of the fault insertion, detection, and recovery process. Memory space is necessary due to the large number of gates, flip-flops, and tristate devices in any modern digital system. The algorithm employs a general model for all types of gates (i.e., AND, OR, NAND, NOR, NOT, XOR), and a generalized model for all types of flip-flops. These general models allow for efficient use of computer memory. Time is conserved by processing, in a given cycle, those devices whose input has changed during the previous cycle.

This emulation system has a very general hybrid capability to model some parts of a system at the device level and other parts at the functional level. This capability allows the user to specify large portions of the system in a functional manner. At the same time, the user retains the capability to model other critical portions in detail, to the gate level, without the need to define every gate in the entire system.

In operation the emulator process is very straight forward. The state of the entire system at a given time consists of state descriptions of all logic devices in the network, state descriptions of all connections among devices, and a state description of the functional subsystem. The emulation must have the initial state of the entire system. Then for each time step T, the emulator calculates the output states for the next time T+1. The process continues to be driven by the series of events until the user-specified stop time is reached. The basic time step is the propagation time for input signals to propagate through a device to reach the output. For the initial emulator, the propagation time is taken as the same for all devices.

Fault insertion capabilities built into this implementation are as follows:

- The ability to insert or remove faults at user-specified times into the logic gates or ROMS.

- The ability to input to the digital logic at user-specified times from sources external to the emulation.

- The ability to output from emulated logic to external ports at times specified by the user or times driven by the system states.

As an initial experiment, a horizontally-microprogrammable computer, the Nanodata QM-1, was chosen as the host system. The emulation was coded at the microcode level to take advantage of the parallel capabilities of the host machine and to exploit the speed advantages of executing code at the most primitive level of the host computer. All processing of the hardware description and fault-injection data, as well as all post-processing of fault data, is performed on a Digital Equipment Corporation VAX 11 which is interfaced to the QM-1.

The emulation algorithm has been used to emulate a simplified model of the central processor of the Bendix BDX930, and the communications interface unit of the CSDL FTMP. Both of these system components have been the subject of other studies and comparative performance data is available to validate the emulation.

2.4.4 Limitations

The principal limitations of the hardware emulation technique are in speed and system modeling accuracy.

The speed of the emulation is dependent upon the size of the models and the speed of the emulation engine. The hybrid technique is important in reducing the size of the models. Speed is important both in shortening the run time and for emulation of real-time processes.

Modelling accuracy has some limitations due to the hybrid nature of the process. In this hybrid, system states are determined from the data structure and a subprogram module operating on that data structure and optionally on the gate-level network. This may not be serious, but care must be exercised in any applications to maintain consistency between the global and gate-level models.

The faults modeled are limited to pin faults, S-A-0 or S-A-1, or gate-level faults that can be defined by effects on the logic algorithms of the system under test. Conceptually this allows for a wide range of faults, but the effects of faults may be very difficult to model accurately and input to the emulation.

Propagation delay times are taken as the same for all devices. This is unrealistic and future versions of the emulator should address this parameter allowing the user to specify the delay time for each device in the gate model library

2.5 Fault Emulation Fault Insertion Methodology

2.5.1 Method

The fault emulation technique inserts software faults, or the effects of faults, into the real system software. A special purpose processor is used to control fault insertion and monitor the system under test. Faults, or the errors manifested by faults, are inserted by triggering the error detection mechanisms or by seeding errors into memory.

Faults are injected into software controlling the system hardware or into the special capabilities built into the hardware such as communications registers, comparison-monitoring detector, configuration status registers, and sensor output registers. Possible software faults include lost or delayed messages, corrupted messages, delayed tasks, abnormal task termination, timer corruption, and system clock corruption. Injection of faults directly into system software is accomplished by transfers over the data bus of the system under test. The process is controlled by timing and monitoring of data and status information received from the data bus. Instrumentation of the effects of testing is also accomplished using the data and status information on the system data bus.

Fault emulation fault insertion offers the following features:

- Allows insertion directly into the substantial software portion of a system. These software errors can be used to examine the software and the interactions of the hardware and software.
- Is often simpler, in terms of time and effort, than hardware fault insertion.
- Is complementary in terms of functionality to hardware fault insertion and does not exclude hardware fault insertion.
- Provides a direct means for testing software implemented fault tolerant detection and recovery strategies.

Data collection and analysis is also implemented by using information on the system or processor bus. Historical data records of normal events and functions may also be collected. Then, error reports can be generated indicating exceptions and abnormal events, allowing measurement of faulted performance. Data analysis supports developing data on runs, experiments, and multiple experiments.

2.5.2 Application

Software fault insertion, as described in this subsection, is most useful during the system development and validation. Faults may be inserted into developmental software modules such as the communications processor. The equipment utilized is essentially a general purpose test and evaluation instrument. This concept should have application during the operation and maintenance phase as well.

2.5.3 Example Results

As an example we will discuss the implementation of the Fault Injection Automated Testing (FIAT) environment, illustrated in figure 2.5-1. FIAT was developed by Carnegie Mellon University under a NASA Langley grant. This system is described in Czeck, Siewiorek, and Segall 1989. The FIAT hardware consists of four International Business Machines (IBM) personal computer (PC) remote terminals (RTs) connected via a 10 Megabit token ring. This hardware provides the operating environment for FIAT, without limiting the generality of the system. The software structure consists of two modules: (1) the Fault Injection Manager (FIM) for experiment control and data analysis, and (2) the Fault Injection Receptor (FIRE) for data collection and processing.

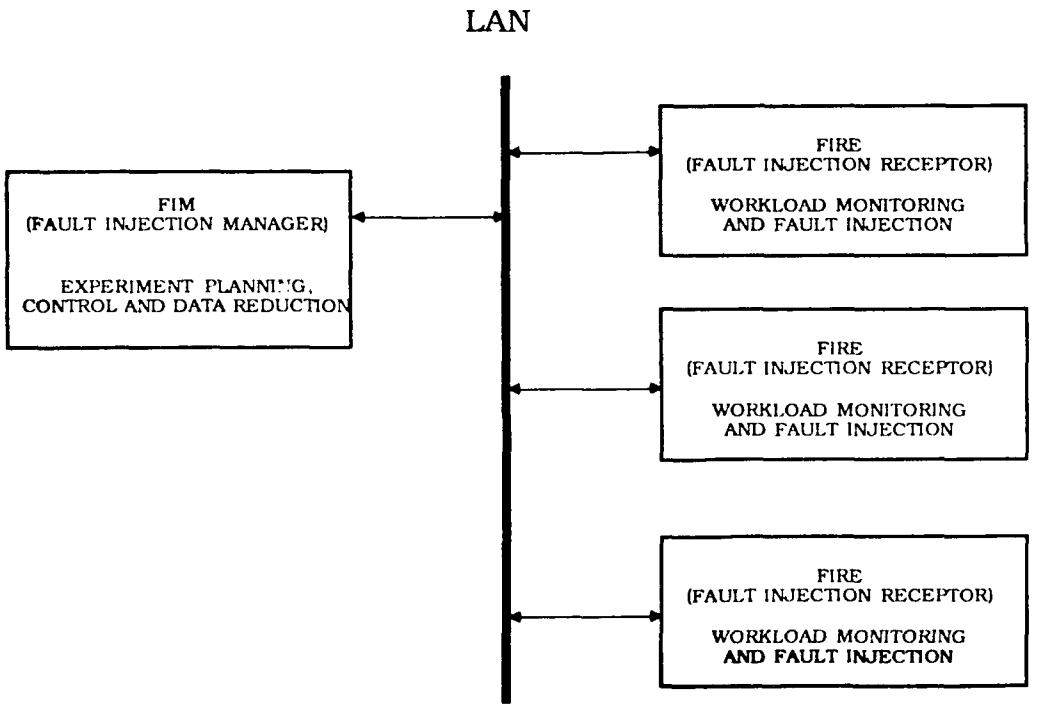


FIGURE 2.5-1. FIAT FAULT INSERTION AND EXPERIMENT MONITOR

The FIM software, shown in figure 2.5-2, processes users' programs and experiment definition to control the injection of faults and collection of data from the system under test. The experiment interface, illustrated in figure 2.5-3, has four modes of processing:

Library Preparation - Assists in preparation of task development, image generation, and attribute extraction. The fault class library groups domains and methods for selection of the fault insertion method during experiment definition.

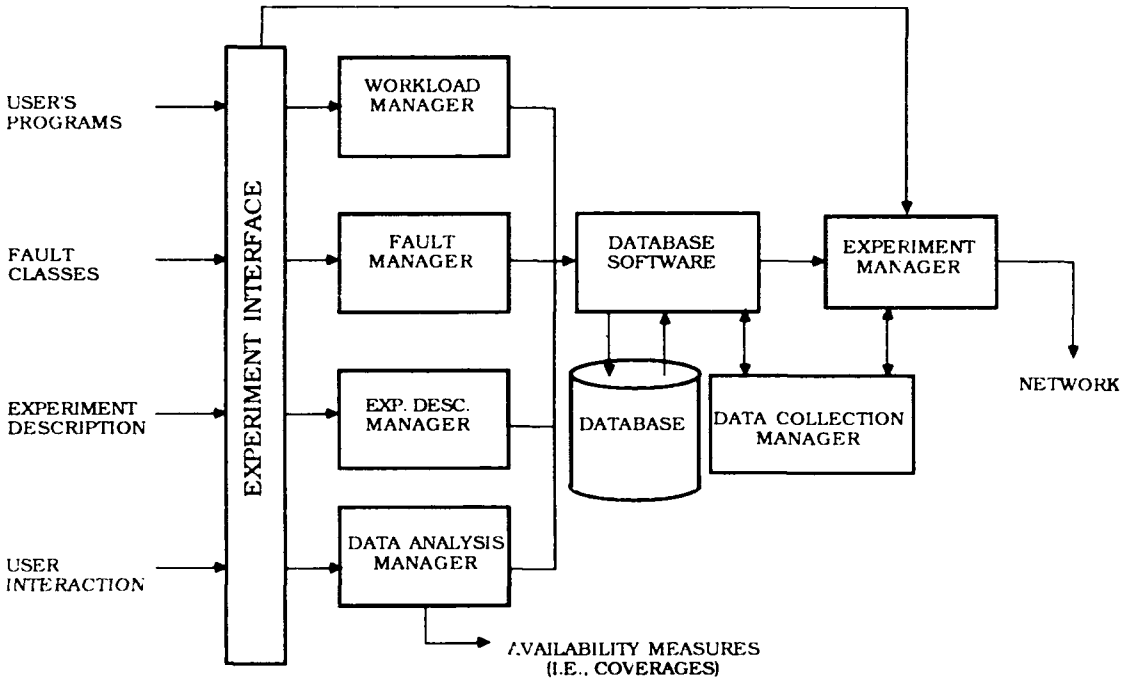


FIGURE 2.5-2. FAULT INSERTION MONITOR SOFTWARE

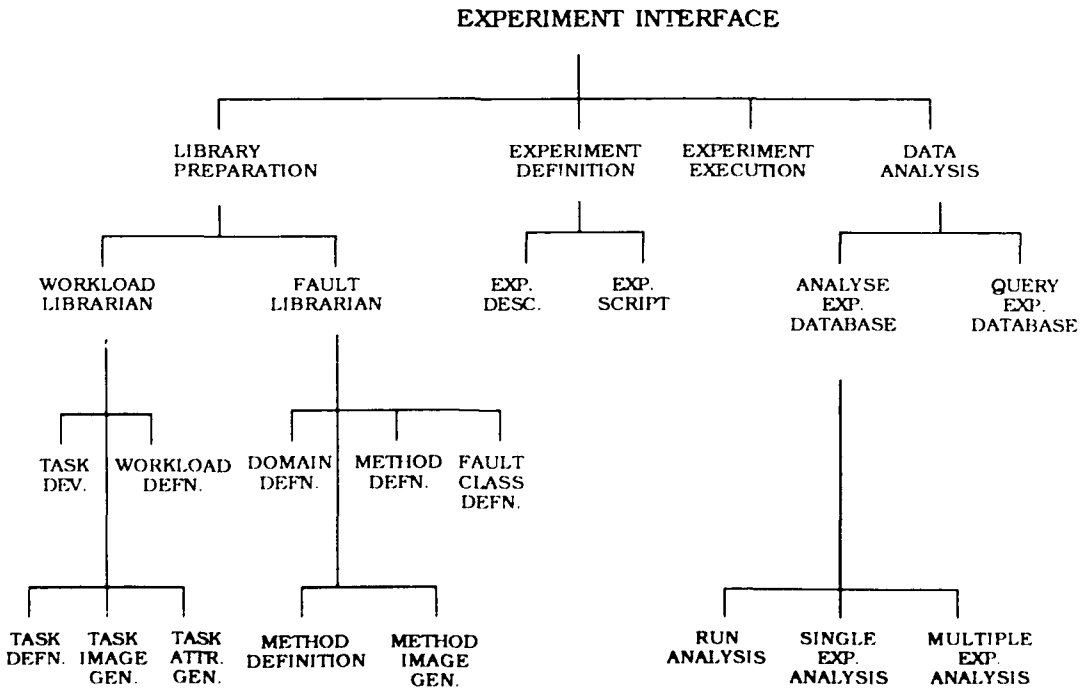


FIGURE 2.5-3. FIAT EXPERIMENT INTERFACE TREE

- Experiment Definition - Assists in developing an experiment description and in script generation.

- Experiment Execution - Utilizes the prepared scripts and executes the sequence of fault injections.

- Data Analysis - Utilizes the prepared experiment definition and provides two types of analysis: (1) user defined queries using Informex Software Query Language (SQL), and (2) generic FIAT data analysis routines.

The data collection and monitoring operate via the FIRE modules shown in figure 2.5-4. The function of each module is as follows:

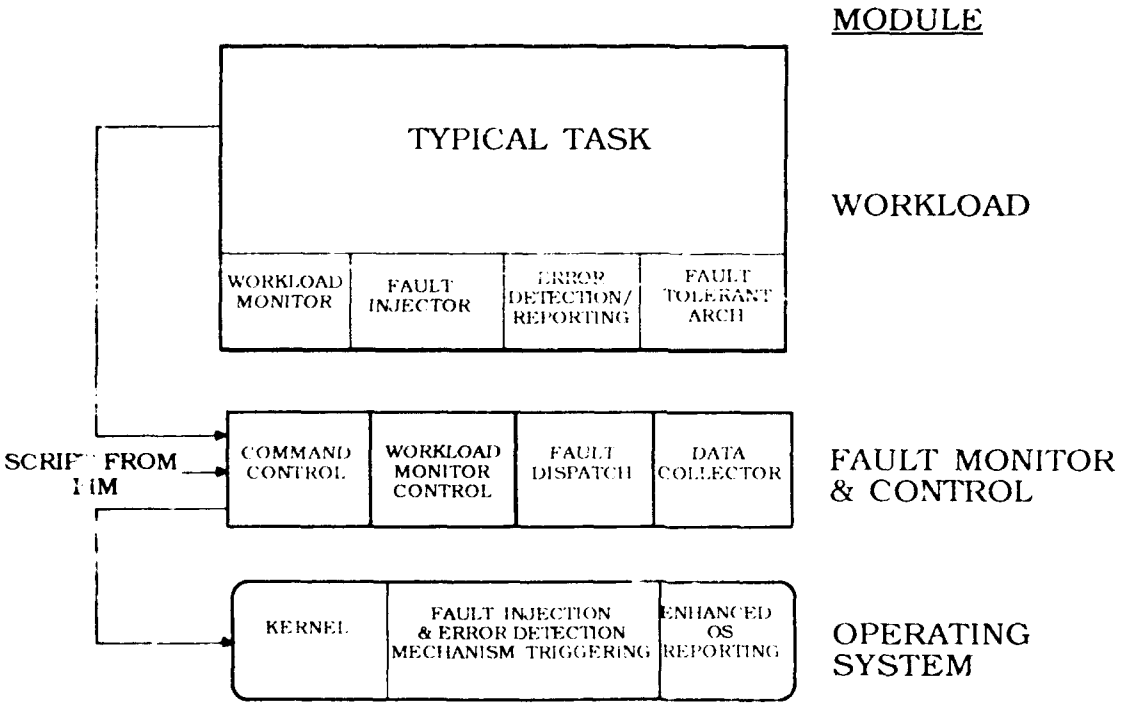


FIGURE 2.5-4. DETAILED VIEW OF FAULT INJECTION RECEPTOR SOFTWARE

- Command Controller - Stores and organizes commands from the experiment manager in a command queue and synchronizes the commands, such as task starting and fault insertion timing within the work load control flow.

- Workload Monitor Controller - Provides the functions of monitoring and control of the workload, task control, communications, and synchronization.

- Fault Dispatcher - Manage fault insertion of workload tasks, communications, and the operating system. Operating system faults include memory

and register faults and triggering of existing hardware or software error detection mechanisms.

- Data Collection - Transfers records generated by the monitor and error report scripted commands to a local experiment database storage.

As an example of the use of FIAT, a distributed checkpointing system was implemented using two FIRE machines. The block diagram, figure 2.5-5, consists of a two-computer system. The primary computer informs the secondary computer of task initiation and the time frame for the next task initiation. The primary machine executes the task and the secondary machine waits for the next interaction. If the time between interactions exceeds the time frame for completion (indicating primary failure), the secondary then initiates a recovery process and becomes the primary. If the primary detects that no secondary exists (indicating secondary failure), it creates a new secondary.

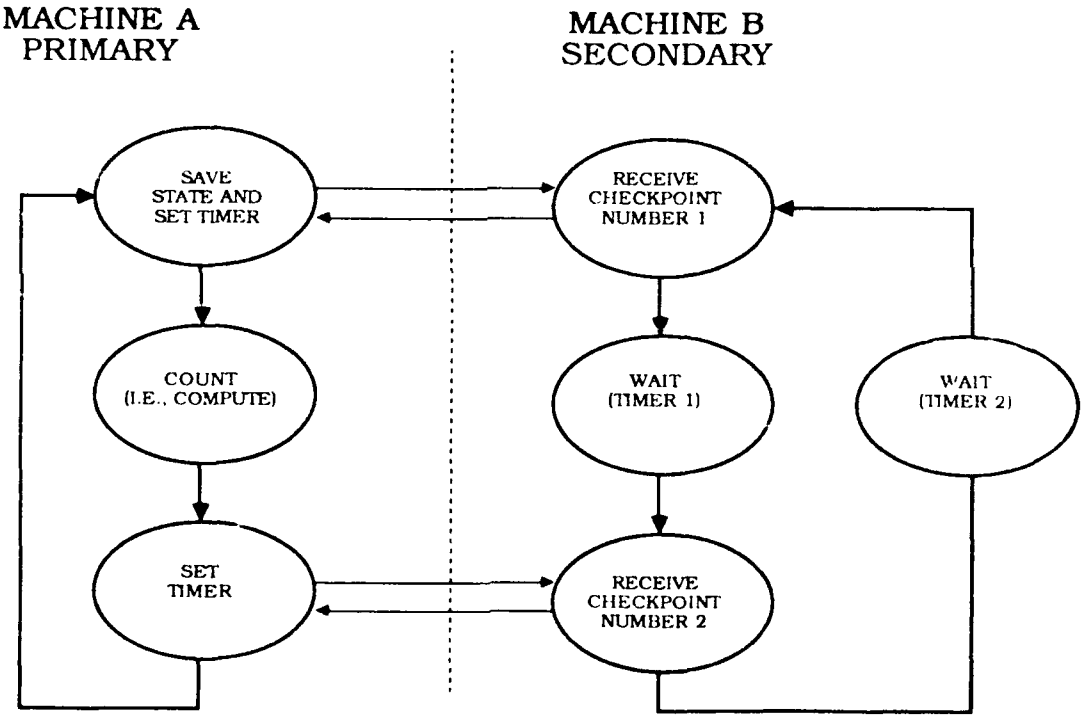


FIGURE 2.5-5. REAL-TIME CHECKPOINTING SYSTEM

Experimental results of inserting faults into the checkpointing system are shown in table 2.5-1. The faults considered were bus faults having double bit compensating errors. These double errors would not be detected by hardware parity and must be detected in software. Note the large latency time and that one-quarter of the inserted faults were not detected. To illustrate how fault tolerance strategies may be compared, the checkpoint system was improved by

TABLE 2.5-1. ERROR DETECTION STATISTICS - CHECKPOINTING ONLY

```
METHOD:  DETECTION STATISTICS

Experiment = 1a
Detection Statistics:

Number of Fault Injections ... 206
Number of Fault Detection  ... 153
Detection Coverage = 74.271845%

Average Detection Time = 4.219567 Seconds
Minimum Detection Time = 1.406250 Seconds
Maximum Detection Time = 6.218750 Seconds
```

TABLE 2.5-2. ERROR DETECTION STATISTICS - CHECKSUMS

```
METHOD:  DETECTION STATISTICS

Experiment = 1b
Detection Statistics:

Number of Fault Injections ... 91
Number of Fault Detection  ... 86
Detection Coverage = 94.505495%

Average Detection Time = 3.102834 Seconds
Minimum Detection Time = 1.203125 Seconds
Maximum Detection Time = 4.203125 Seconds

END OF METHOD:  Detection Statistics
```

adding double checksum comparison to the primary and secondary machines. A checksum is appended to code and data blocks during compilation. This code is regenerated when reading blocks from memory during run-time and is compared with the code determined at compilation to check for errors. Hence the detection methods are timeouts for the first experiment and checksums for the second. Improvements in the results may be seen from table 2.5-2. Note the increased fault detection probability and the decreased detection time.

2.5.4 Limitations

Fault emulation fault insertion has limitations primarily in its inability to force low level errors, such as gate output S-A-0 or S-A-1. These limitations can often be worked around by injecting faults or errors at another location. Since designers are interested in the behavior of the entire system this may not be a serious limitation.

2.6 Physical Fault Insertion Methodology

2.6.1 Method

In this technique faults are actually inserted into real hardware by computer controlled switches holding the voltage levels high or low on individual wires at connector pins of selected devices. Because only the device terminals are accessible, these faults are usually pin-level S-A-0 or S-A-1 faults. Some testing has been performed using inverted levels (i.e., change a zero level to one and conversely). The existence of faults is detected by the real hardware comparator monitoring systems. Measurements include the fault insertion time and detection time. Statistics are collected on the number of faults inserted, detected, detected in error, and not detected.

These experiments can be run at the gate, pin, or resistor or other electronic component. Since real system hardware is used for testing, the common method of fault injection is at the pin level of logic circuits including gates; flip-flops; memory clips; processors; and Small Scale Integration (SSI), Large Scale Integration (LSI), and VLSI components.

2.6.2 Application

Physical fault insertion is used for validation of completed hardware/software systems. This method is a validation tool. Modeling and simulations will have been used prior to final implementation. Those tools will have provided predictions and reliability estimates. A final level of testing using faults injected on real hardware provides the realism, and minimizes the assumptions and modeling errors that may exist in the other techniques.

This technique is most useful during full scale development. Some applications of this technique have been made using development hardware of portions of the system, such as the communications interfaces between multiprocessors.

2.6.3 Example Results

Physical fault injection has been used for several systems including the Bendix 930, the CSDL FTMP, and more recently on a dual-dual FCS for the Lockheed L-1011 Tristar. This system used four Rockwell/Collins CAPS-6 digital computers. Figure 2.6-1 illustrates the experimental setup for data acquisition experiments (Benson, Mulcare, and Larsen 1987). The additions from prior experiments include an external clock and an associated DR11C interrupt so that fault detection is quickly reported to the PDP-11 through the DR11C interrupt. Considerable software modifications were made to analyze, reduce, and present the data for reports. Prior efforts injected faults on the pins of the CSDL FTMP (references 4, 6, and 7, from Benson, Mulcare, and Larsen 1987) system which also uses the CAP-6 processors. Investigations into this methodology are in progress. Figure 2.6-2 shows the setup for data acquisition on this current effort on the CSDL FTMP at NASA Langley.

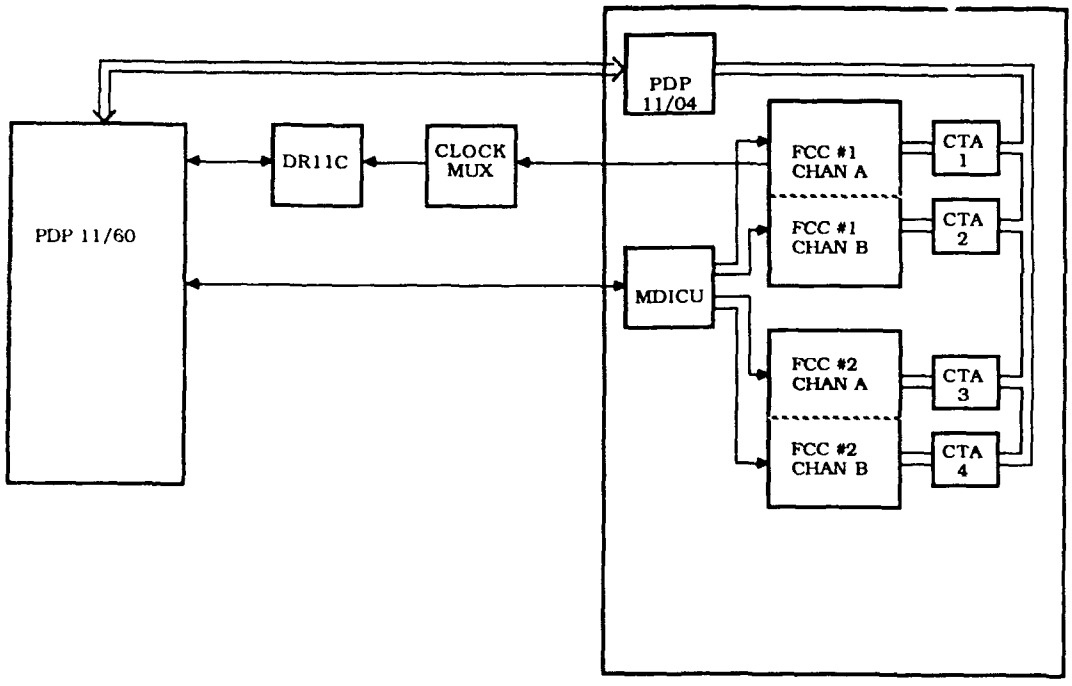


FIGURE 2.6-1. FAULT INSERTION AND INSTRUMENTATION SYSTEM (FIIS)

Both of these systems share common features in fault insertion circuitry (FI). The FI is connected to the UNIBUS in a manner similar to the Collins Test Adaptor (CTA) shown on figure 2.6-1.

The FI provides an interface between the UNIBUS and the system components under test. The fault injection is accomplished in the FI by signals sent to Field Effect Transistors (FETs) inserted between the socket and device, as illustrated in figure 2.6-3. These FETs also allow monitoring of the signals at device

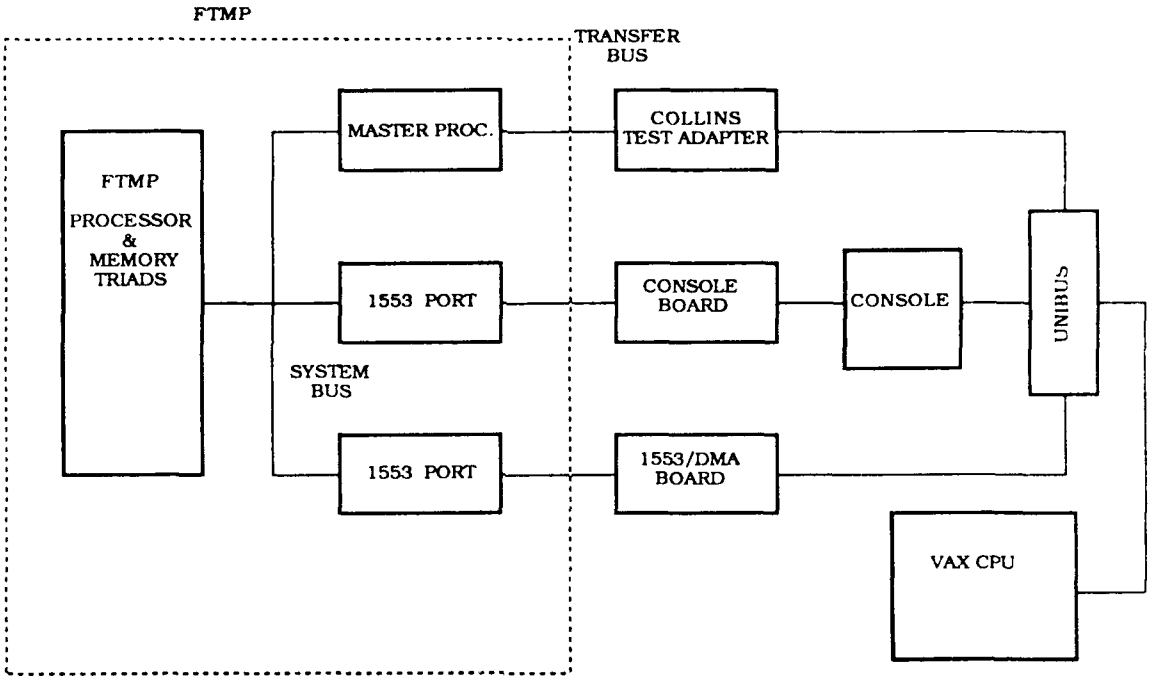


FIGURE 2.6-2. FTMP VAX DATA ACQUISITION ENVIRONMENT

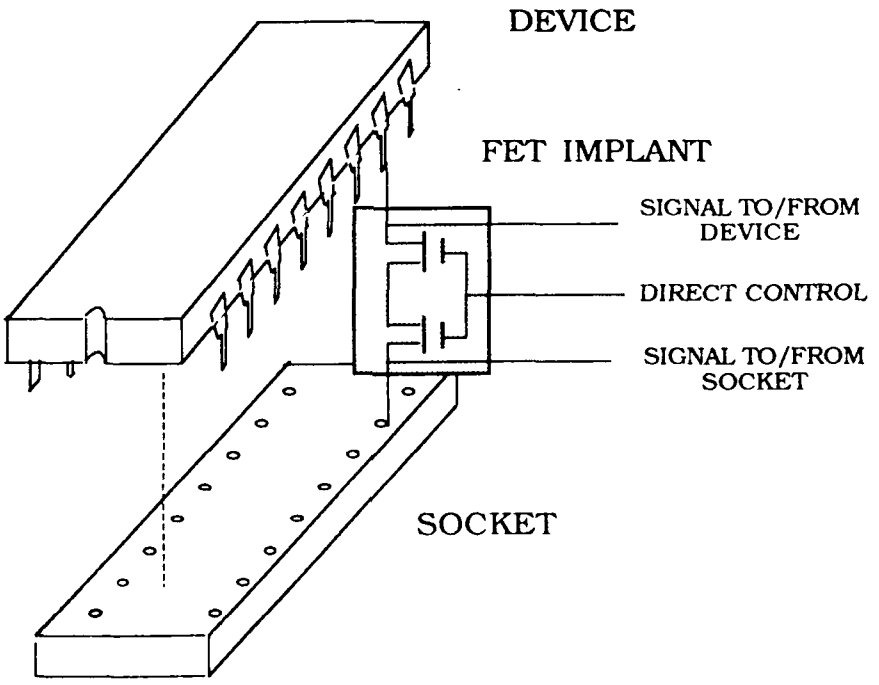


FIGURE 2.6-3. DEVICE PIN ACCESS FOR FAULT INJECTION AND MONITORING

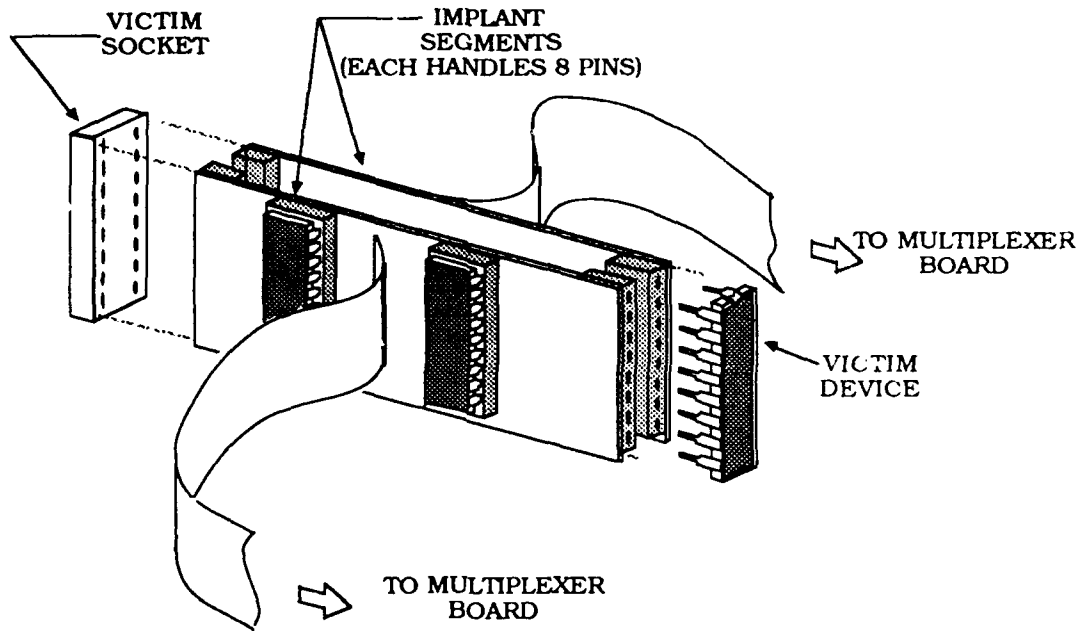
terminals. Turning on the FET establishes a direct connection between the device and the socket. This is the normal condition when no faults are injected. The connection can be severed by turning the FET off. In the off state, any desired signal can be applied to the device or the socket pin. Usually the pin selected is an input. It should be noted that an output pin is usually connected to an input pin of some other device. Added flexibility is provided by multiplexing signals to and from the package and socket under control by the VAX or PDP-11. A choice of eight possible injection signals is provided to the experiment via the multiplex system:

- Direct connection to the socket
- Multiplex A
- Multiplex B
- Multiplex C
- F(AB) Boolean combination of signals from Mux A and B
- F(A,B,C) Boolean function F(AB)and output of Mux C
- Ground, logic zero
- External signal

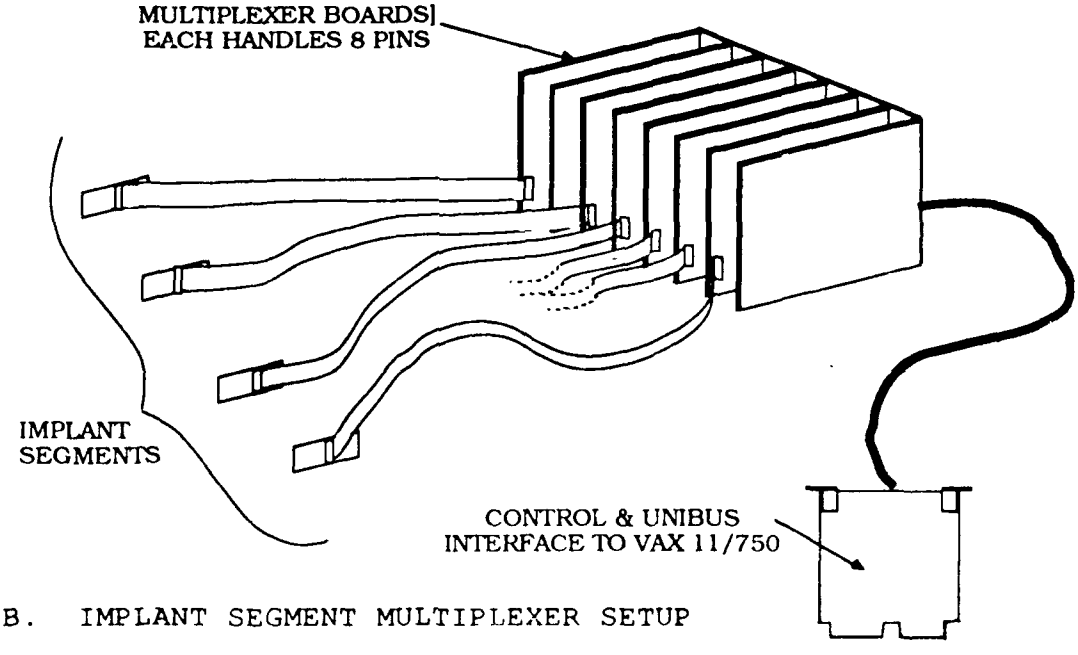
Each of these eight signals may be inverted before application to the victim pin, thus providing sixteen possible sources. The usual experimental run uses S-A-1, S-A-0, and "complemented signal" types of faults. This setup has powerful flexibility. For example, certain faults in integrated circuits can change a NAND gate into a NOR gate. It is possible with the fault injector to simulate such a fault by extending the pins of the target gate with the FETs and generating the Boolean functions. The main utility of this injector is to inject faults on tristate lines. The direction of the fault can be made a function of other signals on the device, signals that determine the state of the tristate control pin. Thus, it is possible to inject faults into the data pins of memory chips, bus drivers, and other tristate devices.

The injector hardware is packaged as hown in figure 2.6-4, with FET pairs on an extender segment. These extenders are used to provide space for the multiplex signals and interface to the sockets. Each multiplexer board accommodates eight pins. The six boards are connected to the instrumentation processor via a unibus card into the VAX or PDP-11.

The recent experiments using the FIIS at NASA Ames provided information on the behavior of a completed FCS operating in a simulated aircraft environment. The CAP-6 processors used in this system had approximately 1,200 pins. For this system a nearly complete coverage of single faults was injected on all possible pins. These results are an important addition to the information obtained from the earlier work on the Bendix 930 and on the FTMP. In these experiments over 2,700 simulated faults were applied to the chip pin level. Most testing was done open loop. The more persistent faults were subjected to closed loop



A. DEVICE TEST ADAPTER



B. IMPLANT SEGMENT MULTIPLEXER SETUP

FIGURE 2.6-4. FAULT INJECTOR HARDWARE

testing in order to involve the built-in fault detection mechanisms. The test results, summarized in table 2.6-1, are in reasonable accord with similar data, confirm well with the fault models, and indicate the value of this type of testing for practical system validation efforts.

The data shown in table 2.6-1 are the result of injecting 2549 "care" faults. While running the baseline FCS program under open loop control conditions, 2461 were detected and 88 were undetected. Open loop tests were conducted to speed up the process because the closed loop system provided a layer of error detection and recovery. Of the 88 undetected faults, 15 were invert faults. Of the invert faults, 1.71 percent were detected while 4.37 percent of stuck-at faults were undetected, indicating that invert faults are less latent than stuck-at faults. Each of the 88 undetected faults were injected while executing the self-test program. All these faults were detected indicating that the processors "care" about these cases. All the micromemory and processor faults injected were detected. The 100 percent coverage of the self-test program indicates the value of running self-test in the background, if possible, every few minutes even though no errors are detected by the system in its present mode.

TABLE 2.6-1. EXPERIMENTAL RESULTS SUMMARY FROM FIIS

DEVICE	# FAULTS INJECTED	# FAULTS DETECTED	# FAULTS UNDETECTED	# DON'T CARES	PERCENT DETECTED	PERCENT UNDETECTED
Data Path	1535	1502	33	83	97.85	2.15
Control	1014	959	55	83	94.58	5.42
S-a-0/S-a-1	1670***	1597	73	123	95.63	4.37
Invert	879	864	15	43	98.29	1.71
2901	423	423	0	25	100.00	0.00
Micromemory	231	231	0	9	100.00	0.00
Input Pins	1805	1741	64	87	96.45	3.55
Output Pins	744	720	24	79	96.77	3.23
Self-Test	1687	1687	0	0	100.00	0.00
Totals	2549	2461**	88*	166	96.55**	3.45**

NOTE: (1) Injected faults do not include "don't cares"
(2) Percent "don't cares" = 6.11
(3) * = Detected by self-test
(4) ** = Excludes self-test
(5) *** = 791 S-a-0, 806 S-a-1

Investigations are under way at NASA Langley, using the advanced data acquisition environment shown in figure 2.6-2, to investigate the FTMP error detection and faulty unit isolation functions (Padilla 1989). Earlier tests disclosed that the FTMP occasionally disabled the wrong units. This behavior continued

until the system crashed after 4,000 to 6,000 faults were injected. These investigations uncovered reasons for the failures due to the fault identification procedure incorrectly identifying that only one Line Replaceable Unit (LRU) detected the error. This event triggers a software component, designed to handle Byzantine faults, which keeps track of the accuser and disables it after a second event. In this case one fault causes two LRUs to be identified as in error, one bad unit and one good unit.

As a result of these investigations, a new methodology for selection of the pins for fault injection is proposed (Padilla 1990). To investigate the abnormal behavior of FT systems, tests should be based on two principles:

- Faults should be injected in signals having high fan-outs (board enables and other control signals), thus maximizing the amount of interaction (number of potential errors).

- Information to determine the system state should be observable at all times in real time.

These principles, although simple in concept, require a completely new experimental environment. The data acquisition environment overburdened the original injection and test system. The new data acquisition system, figure 2.6-2, has three orders of magnitude increase in data rates over the old system.

Some of the initial experiments using the new system disclosed that LRUs not being faulted were being disabled by the fault management software. Although the system classified the events as caused by real permanent faults that occurred during the experiments, no faults could be found at the conclusion. All the supposed faulty units were reactivated either manually or by rebooting the system.

2.6.4 Limitations

Faults used in testing are often limited in type and number. Since real hardware is used for the system under test, faults may only be injected at connector pins. Internal gate nodes may not be tested.

The hardware and software must be available or sufficiently well defined that prototypes may be made.

The methodology for selection of pins for fault injection does not have universal acceptance. Systems architecture and design assumptions may limit the faults tested to those locations where designs have addressed error detection and correction. On the contrary, pins may be selected at random, perhaps testing unused gates or overlooking important vulnerabilities.

Physical FI has been used extensively in system validation, with faults inserted at the pin level as S-A-1 and inverted faults. With systems implemented in an SSI or medium scale integration (MSI), pin-level stuck-at faults closely resemble failures which have been observed to occur in such devices. But with LSI and VLSI realizations, failures may be remote from the input/output pins. At these higher levels of integration, FI seldom claims to portray physical

faults accurately. However, the hope is that they prove a first approximation to the metrics under study. The data is not yet available to verify this assertion. Additional information is presented with detailed examples in section 3.

Although faults may be remote from the boundaries, promising results have been reported with pin-level FI for LSI and VLSI devices. Schuette et al. (1986) inserted transient faults of the data, address, and control lines of an MC68000 bus, representing faults within data and control sections of the processor. With this FI technique, two error detection schemes were evaluated. Czeck et al. (1987) inserted faults in an FTMP triad by causing one unit to execute a special code, causing a trigger in the error detection mechanism. This method was able to duplicate some of the results presented by Lala and Smith (1983). Even so, McGough and Swern 1983 illustrated a gap between gate and pin-level fault injection.

3. ADDITIONAL FAULT INSERTION AND TEST EXAMPLES

3.1 Draper Labs FTMP Tests Using Fault Emulation

This section describes the results of a simulation test using fault insertion. This experiment was conducted in the NASA Langley AIRLAB Diagnostic Emulation (DE) facilities, and was described fully in Baker, Mangum, and Scheper (1988).

The primary objective of this experiment was to determine the diagnostic self-test sequences used to uncover latent faults in a logic network providing the key fault tolerance features for a flight control computer. In this experiment, using the setup shown in figure 3.1-1, more than 1,600 faults were injected into a logic gate-level emulation of the Data Communicator Interstage (C/I), a key component of the CSDL FTMP. For each fault injected, diagnostic sequences consisting of over 300 test vectors were supplied to the C/I model as inputs. For each vector in a test sequence, outputs from the C/I were compared with the outputs of a fault free C/I. If the outputs differed, a fault was considered to be detectable for the test vector. These results were analyzed to determine the effectiveness of the various test sequences, to identify latent faults, and to identify ways for improving performance of the diagnostic sequences.

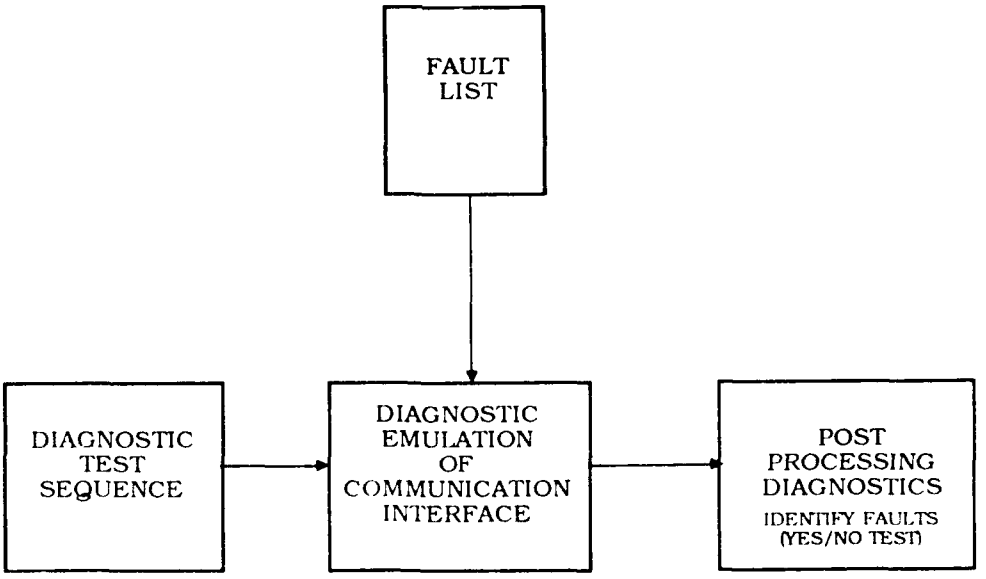


FIGURE 3.1-1. EXPERIMENT DIAGRAM

The C/I of the FTMP was selected because of the following reasons:

- The C/I network provides the critical features of input source congruency, error masking, error reporting, and system reconfiguration.
- The quadruplex C/I network used the full capacity of the AIRLAB DE.
- Documentation for the C/I design at the logic gate level was available.
- Results of these experiments could provide important information about the FTMP design.

The C/I provides the Fault Tolerant Processor (FTP) with certain key fault tolerance capabilities. These capabilities arise for circuitry to support correct replication of simplex source data in all good channels, to detect and mask errors in a single channel, and support reconfiguration by blocking out faulty channels.

The data exchange network associated with a quadruplex FTP is illustrated in figure 3.1-2. Each channel, A, B, C, or D, has transmit and receive registers which can be accessed by the input/output processor or the computation processor via a shared memory bus.

Simplex data (i.e., data available to one FTP channel) is distributed to all FTP channels by a "FROM x" exchange. This will cause the data word in the communicator transmit register of channel A to be sent to all other channels. Data transfers between communicators and interstages are byte serial with two bytes comprising a data word. Data sent to each communicator are distributed to associated interstages.

Each interstage replicates and passes a copy to each communicator. Communicators vote the data received and write the received data into the receive register. Receive registers are then read into the associated channel processor. In this manner simplex data from channel A is read into all good channels.

If each channel has the data required for fault masking, then a "FROM OWN" exchange can be performed. Errors detected during the voting process of an exchange are recorded in the Status Register of each communicator. Each status register can be read by its associated processor. These error reports are intended for use by the FTP in determining and isolating faulty channels.

A write-only Mask Register allows a channel to be excluded from the voting process.

The capability to execute exchanges which bypass the interstage and voter is provided for network testing.

The local data bus provides a large share of the communications. It is a 16-bit tristate bus communicating most of the signals required for redundancy and processor management.

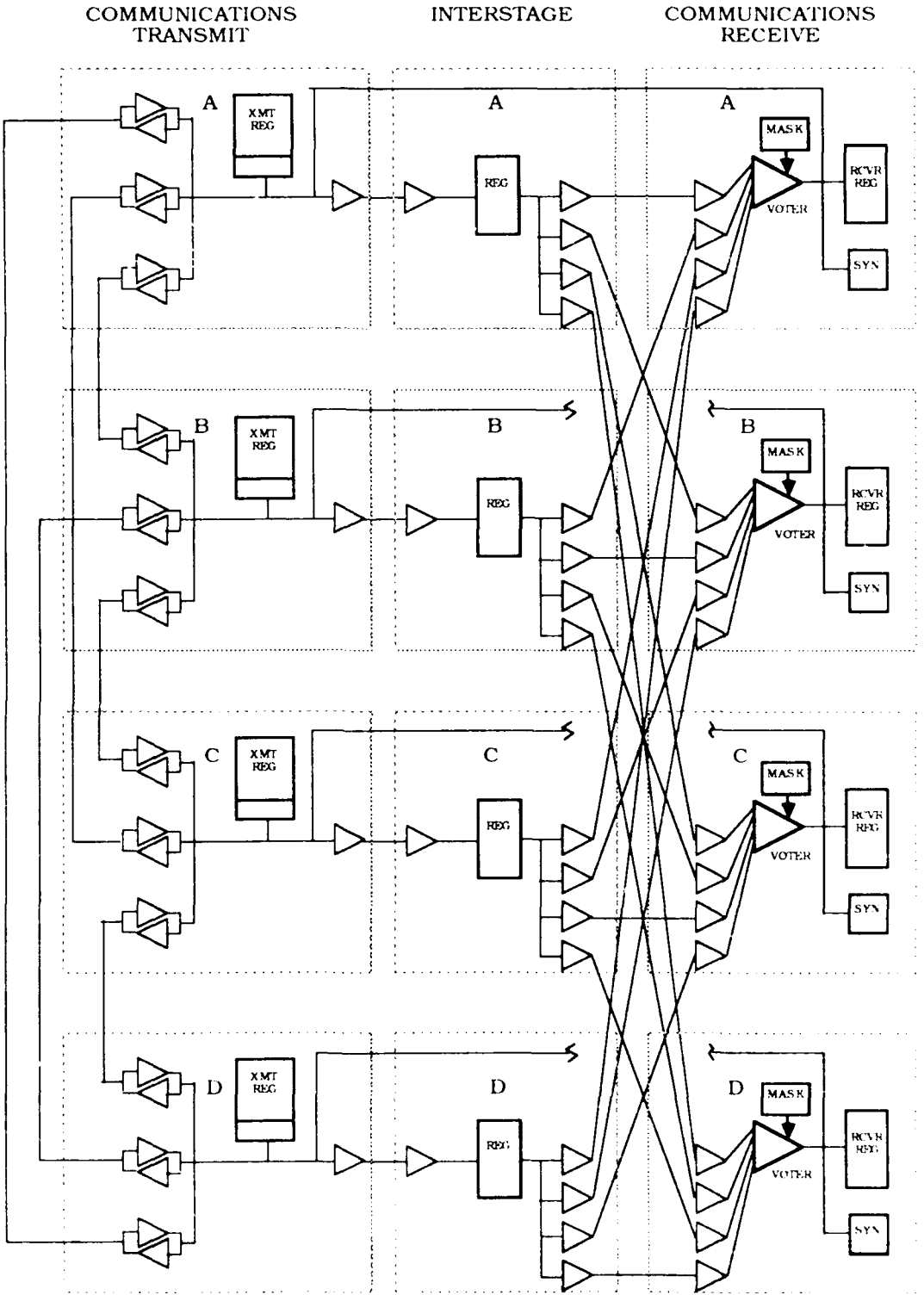


FIGURE 3.1-2. C/I EXCHANGE NETWORK

3.1.1 Gate-Level Model For Redundancy Communicator Interstage

During the analysis several assumptions and approximations were found necessary. Some of these proved limiting on the accuracy of the faulted performance prediction. Key features of the model and several limitations arising from these assumptions are discussed in the next paragraphs.

The local data bus was modeled using AND gates at each source enabled by the appropriate source enable line. The resulting model was functionally equivalent to a tristate bus. The OR function is a phantom gate in that there is no identifiable device in the network to which it can be associated. S-A-0 faults were inserted on this phantom gate to simulate the effects of certain bus faults.

The Transmit Register, Mask Register, and Receive Register are all implemented using D-type latches with tristate outputs. The storage portion of these circuits was modeled as D-flip-flops having tristate outputs. The source bus communicates data bytes from the Transmit Register to cross channel links and to the interstages. This bus was also modeled with AND gates and phantom OR gates similar to the local data bus.

The interstages consist of drivers, receivers, and D-type flip-flops. These devices were modeled using the diagnostic emulator gate and the same flip-flop primitives as above.

The cross channel links that distribute data from each communication source bus to all other communicator source buses consists of a bidirectional, tristate bus driver. The circuitry is diagrammed in figure 3.1-3. The model of this portion of the FTP required 384 connections and 96 phantom gates. A decision was made to simplify the circuit to 192 connections and zero phantom gates. This was based upon an incomplete analysis which indicated that the simpler circuit would adequately represent the network behavior. While data is handled by this model, the effects of faults are not properly treated. Consequently, certain gate faults were not found by the C/I diagnostics. In the discussion presented in Baker, Mangum, and Scheper (1988) these faults would have been discovered using the more complex model.

The Status register file is implemented with two 4x16 Random Access Memory (RAM) chips. Only 5x5 bits of the register in use were modeled. This required five five-bit D-flip-flops with address decoder and selection logic to direct the data to and from these registers.

The Mask transfer block is implemented with a 2048x8 bit programmable ROM. A feature of the diagnostic emulator was used to model this device. This feature allows storage of memory contents outside of the range of memory being modeled. In this function, bits are forced to the values contained in the ROM address. Two other programmable logic arrays are used in the FTP. These were modeled directly as logic equations. The adequacy of stuck-at faults for these devices is questionable since bridging faults cannot be represented with such a simple model. A more elaborate model was judged to be beyond the scope of the modeling effort.

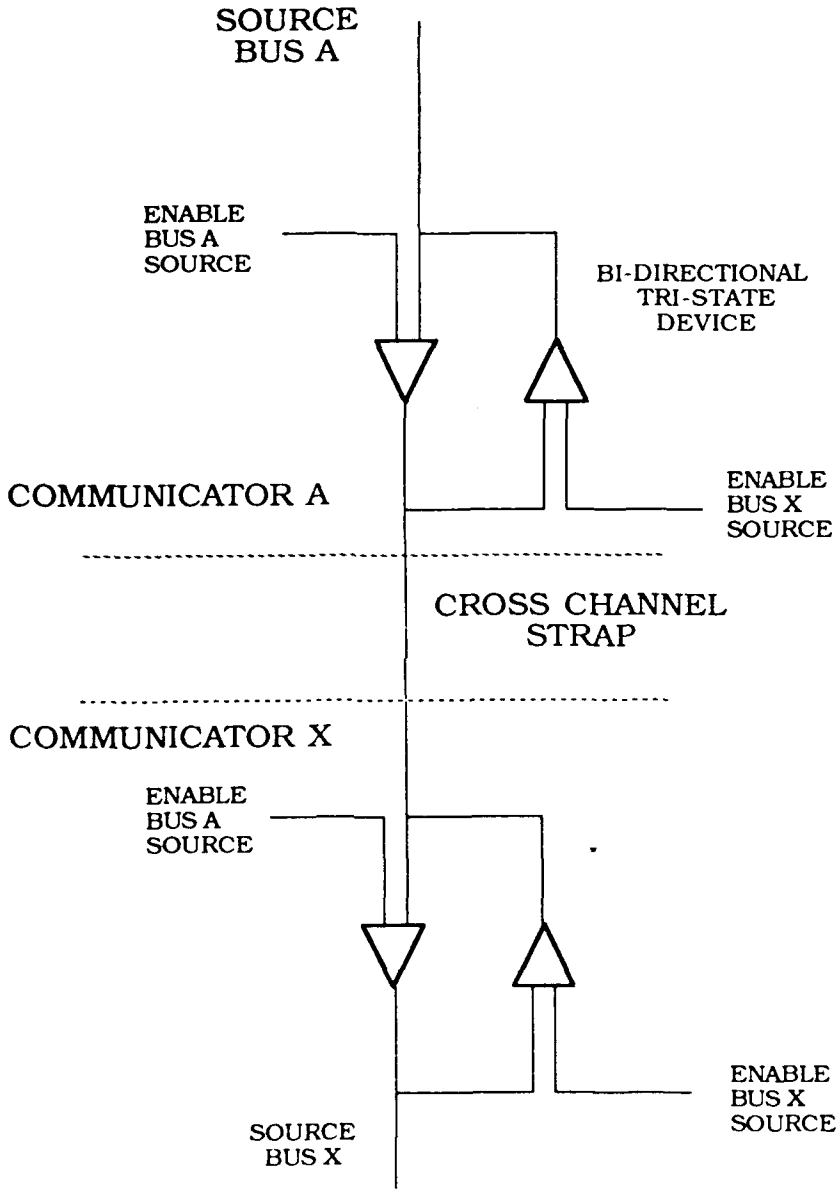


FIGURE 3.1-3. ACTUAL CIRCUITRY FOR CROSS CHANNEL LINKS

3.1.2 Fault Detection Effectiveness of Diagnostic Tests

Most of the effort was directed toward developing and validating the data C/I gate-level model and associated software. These areas of activity included:

- Development and testing of experiment support software.
- Activities associated with learning to use the diagnostic emulator.
- Development of a gate-level model of the C/I.
- Development of C/I sequences that would test the C/I in accordance with the CSDL C/I software description.
- Validation of the C/I model.

The system flow diagram for the diagnostic emulation experiment is shown in figure 3.1-4.

The C/I plays a critical role in FTP fault tolerance by ensuring that all good processors receive identical data values, and by masking errors and providing for reconfiguration after faults are detected. An important part of this operation is the use of diagnostic test sequences for the C/I. These sequences are used to uncover latent faults. The experiment was also designed to determine the effectiveness of proposed diagnostic test sequences.

A typical sequence of operation mixes the test sequences with voting and data transfer between processor channels. In operation, about nine transfers are required between the four processors; the test sequences using vectors consisting of self-test and two exchanges are intermixed with the normal transfers. A complete test requires approximately 2^{16} test vectors. Assuming that self-test is constrained to using only five percent of the processor through-put capacity, the self-test is constrained to use only five percent of the C/I throughput capacity. The self-test will require approximately two minutes to execute.

Estimates of the time needed to conduct a fault injection experiment using the diagnostic emulator are also of interest. Consider an experiment that requires the injection of 2,000 faults with the C/I self-test sequence being run for each of the faults. Assume that the NANO Fast QM-1 is used and that only self-test must be run (i.e., the simulation is 100 percent used for test sequences). The QM-1 will execute approximately 50,000 times slower than real time. The total simulation time required would be 19 years. If the VAX FORTRAN version of the DE were used, the simulation would require 3.8 centuries. These unacceptably long run times provided motivation for restricting the experiment. If the required simulation time could be reduced by a factor of about 100, then 200 hours of run time on the QM-1 would be needed. Such a simulation would be well within the feasible range.

A reduction in time was planned for and accomplished by noting that all of the transactions needed to share opinions in an actual FTP would not be needed in an experiment. The additional factor of 100 reduction was obtained by reducing

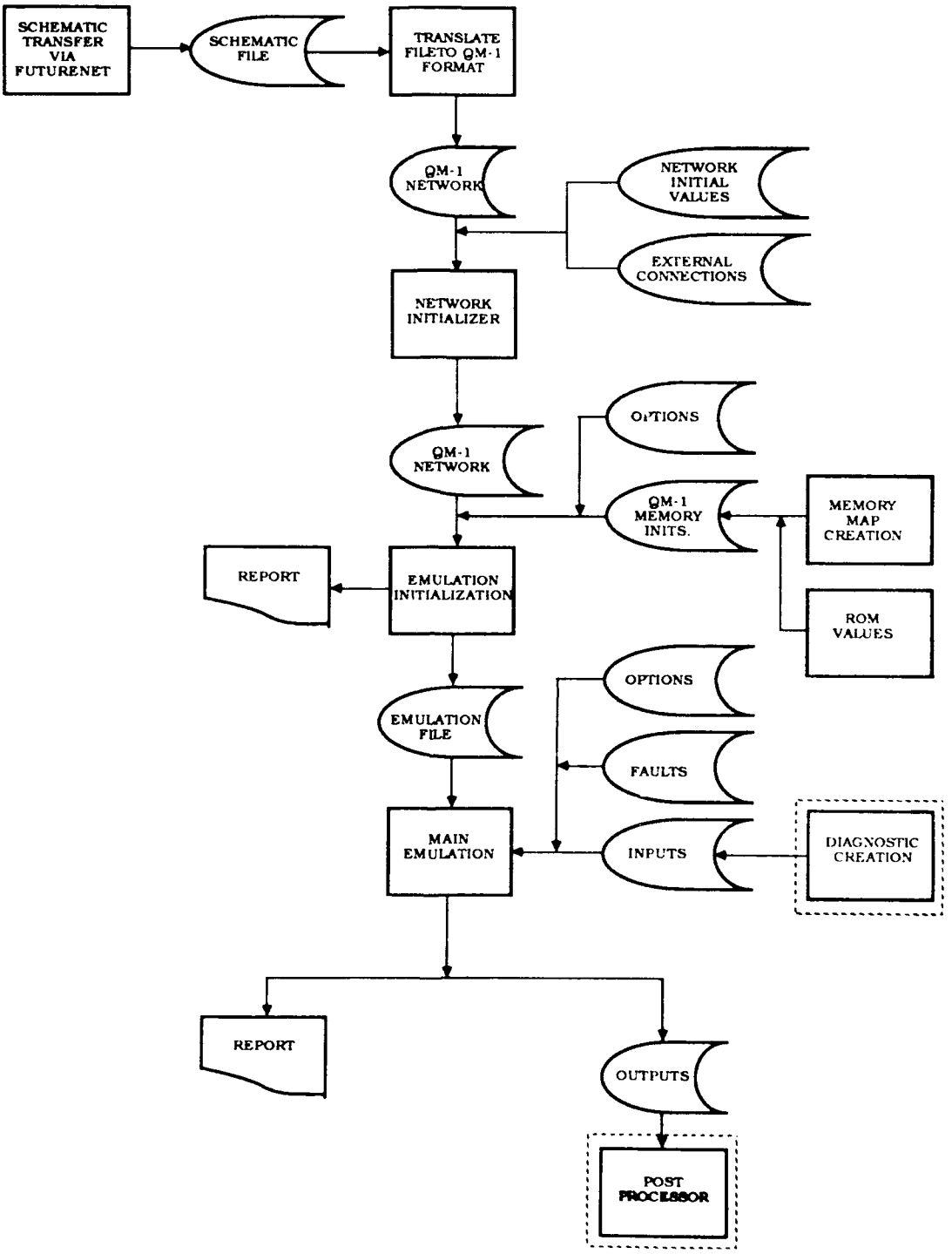


FIGURE 3.1-4. SYSTEM FLOW DIAGRAM

the number of test vectors. This was accomplished by eliminating the test vectors on the Programmable Array Logic (PAL) failure modes since these were modeled not to have errors. Further it was decided to test only the C/I vectors for the quadruplex and triplex configurations and only one for the PAL chips. The experimental tests used are shown in table 3.1-1. The C/I model is shown in figure 3.1-5.

3.1.3 Experimental Results on Hardware/Software

The experimental data presented are drawn from the test report by Baker, Mangum, and Scheper (1988). These data are not adjusted or modified to account for certain problems in the models. For each self-test, the percentage of faults detected is relative to the number of faults injected. It should be noted that any conclusions need to be drawn from the complete report and not this synopsis. Data shown is to illustrate the kinds of experimental results available and not the detailed conclusions.

TABLE 3.1-1. TESTS USED FOR EXPERIMENT

- VOTER TESTS

  Tested a single voter PAL chip (2 bits) in the quad and triplex configurations. Used both upper and lower bytes to give two test vectors per vote.

- MASK TRANSFORM TESTS

  Tested mask transform logic for quad and duplex configurations.

- CURRENT STATUS UPDATE TEST

  Tested the current status update PAL.

- PRESENCE TEST

  Equivalent to CSDL "Presence".

- BASIC TEST

  RTI test which tested a range of C/I functions beyond "Presence".

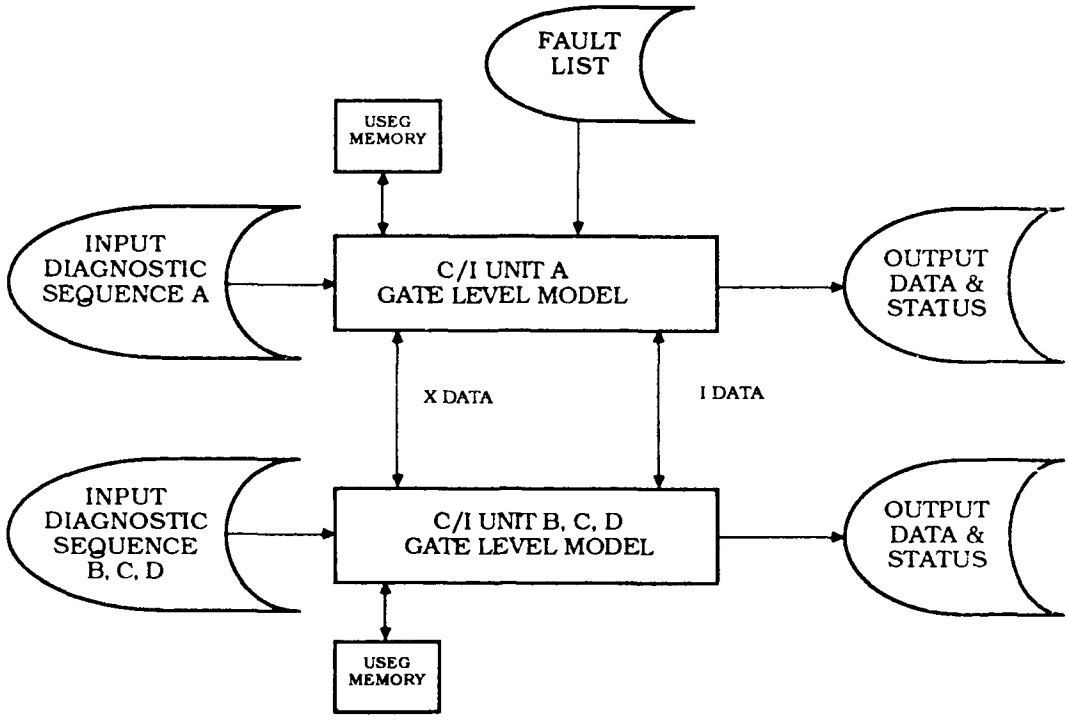


FIGURE 3.1-5. C/I MODEL FOR DIAGNOSTIC EMULATION

Figures 3.1-6 and 3.1-7 illustrate some of the detailed results in the report. The first shows the effectiveness of the diagnostic test run on the simulation. The second shows the performance of the non-voter versus the voter tests in detecting faults.

Table 3.1-2 summarizes the detected faults for the 2,000 injected faults. If the results were adjusted for assessments of the undetected faults, and if the identification (ID) logic faults are inconsequential, the adjusted performance of the C/I self-tests would exceed 98 percent and could approach 100 percent coverage.

Much of the effort in this experiment was directed toward recapturing the C/I design, and validating the model derived from the recaptured design. Well established Computer Aided Design (CAD) tools will be required to expedite the modeling process to deal effectively with VLSI designs.

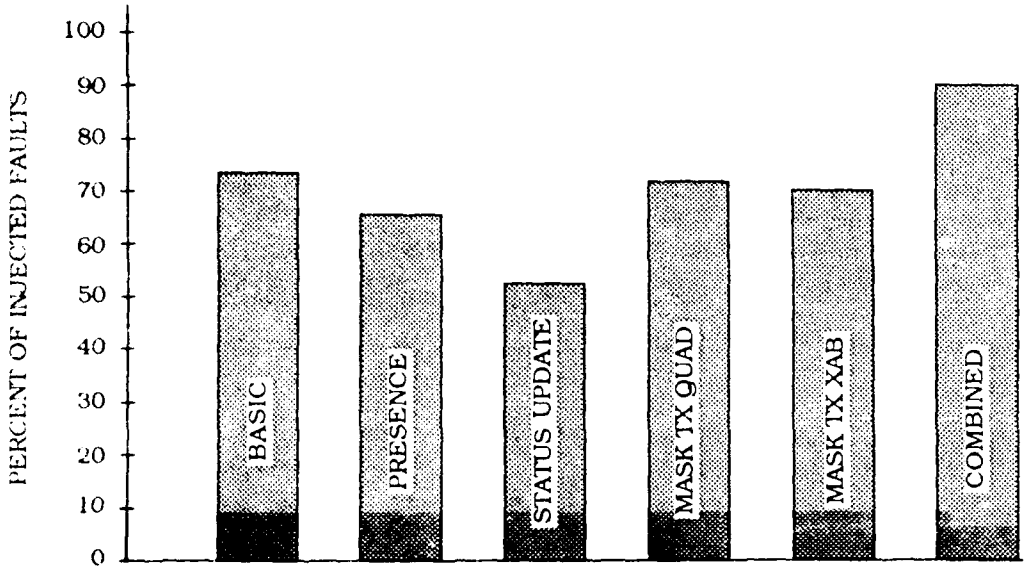


FIGURE 3.1-6. DIAGNOSTIC TEST PERFORMANCE

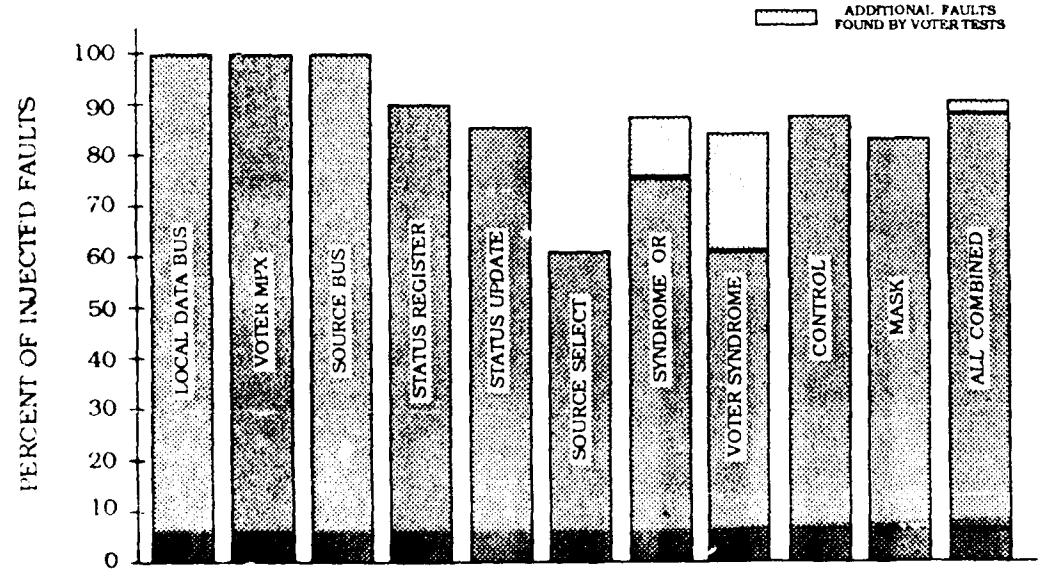


FIGURE 3.1-7. COMBINED PERFORMANCE OF ALL NON-VOTER TEST

TABLE 3.1-2. SUMMARY OF UNDETECTED FAULTS

FAULT SET	UNDETECTED FAULTS	SOURCE	GOAL	ESTIMATED UNDETECTABLE
Source Select	47	Cross channel model and ID logic		0
Control	26	ID logic and cross channel + product interface and reset		3
Voter Syndrome	19		Run Voter Quad Test	0
Status Update	16	Not known	Run Voter Quad Test	0-16
Syndrome Or.	14	Not known	Run Voter Quad Test	0-14
Status Reg.	15	Status reg. address decode	Modify Self-Test	0
Mask	4	ID logic		0
Local Data Bus	0			0
Voter Mux	0			0
Source Bus	0			0
Combined	141			3-33

3.2 Electromagnetic Induced Transient Injection Test Examples

3.2.1 Electromagnetic Transients and Fault Injection

It has been estimated that approximately 80 percent of sudden computer failures, or crashes, are caused by some form of electric transient (Iyer and Rosetti 1986). This statistic serves nicely to illustrate the crucial role that transient EM fields may play in flight system fault tolerance and reliability.

The avionic industry is experiencing a rapidly growing dependence on extensive digital control of complex and crucial flight systems. These systems necessarily demand high reliability and fault tolerance. Digital systems, however, are notoriously vulnerable to extraneous electric fields and transient pulses of voltage.

Modern aircraft are also utilizing more composite types of materials in their structural design. These materials offer high strength and lightweight structures which are very beneficial for increased fuel efficiency. However, these materials are ordinarily less efficient at reflecting and attenuating EM fields. Furthermore, a material joint between an all-metal part and a composite part can serve as an efficient penetration point for EM fields by disrupting the surface currents induced on the two materials. In short, modern aircraft electronic equipment will likely be subjected to significant EM induced transients in the wiring and equipment.

Guidelines are being developed for methods of determining upset susceptibility of redundant systems to EM transients. The studies of transient susceptibility and transient injection, discussed in this section, will lead to increased understanding of upset mechanisms and thereby improved tolerance strategies.

EM transient fields arise from a wide variety of sources. The most significant sources in terms of very high potential energy are lightning and high power Radio Frequency (RF) transmitters, such as for radio, microwaves, and radar.

Lightning pulses are similar to Nuclear Electromagnetic Pulses (NEMP), both having similar waveforms in their signature transient fields. Both may typically be characterized in the time domain by a double exponential waveform with a very rapid rise time, and a fall time from 10 to 100 times longer than the rise time. An example of this waveform is illustrated in figure 3.2-1.

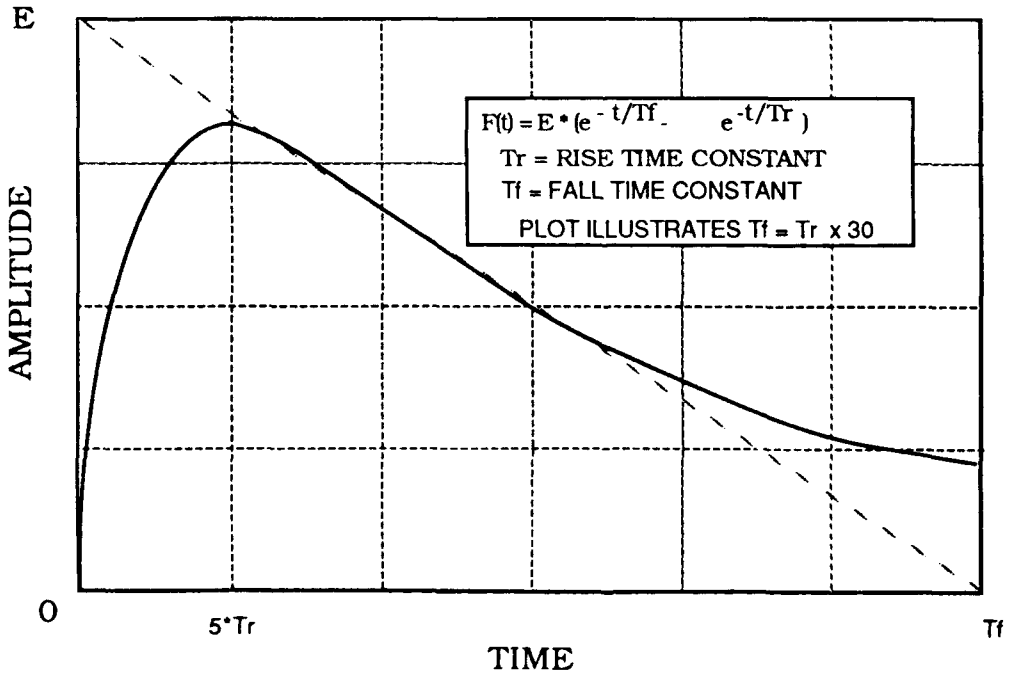


FIGURE 3.2-1. TIME-DOMAIN TRANSIENT WAVEFORM GENERATED BY LIGHTNING

These transients predominantly cover the frequency range from tens of kiloHertz (kHz) to hundreds of megaHertz (MHz). An adequate simulation waveform for laboratory and first order analysis of equipment response purposes is an exponentially damped sinusoid which reaches about 60 percent of maximum amplitude on the order of four cycles (S 3 AE4L 1988, Rasch 1987, Cooley and Shortess 1988).

Aircraft are regularly exposed to High Energy RF (HERF) fields from radio, microwave, and radar. Effects on aircraft systems from these power transmitters is a function of the aircraft materials and electronics technology, as well as the power levels. Important factors include the transmitter modulation characteristics (AM, FM, pulsed, continuous wave) and geometry, such as the orientation of the aircraft in high field regions and the proximity of the source antenna.

Ordinarily, the voltage and current waveforms on an antenna will be the same as the EM field waveform. However, an exponentially damped sinusoidal EM field will induce an exponentially damped sinusoidal vol--ge/current onto a line or within a component. The coupling mechanisms for transients in aircraft wiring are often much more complex events. The aircraft structure and wiring acts like a complex antenna for remote EM sources. Complex RLC and M parameters govern the coupling from currents on the aircraft structure to critical wire and circuits within. Induced transient waveforms inside an aircraft may be derivative signals of complex filter functions of the incident EM fields. The most dominant coupling mechanisms include current or voltage induced directly by the EM field onto a wire or component (essentially acting as part of an antenna). In addition, important interactions include inductive or capacitive coupling from some conductive medium or aperture near a wire or component, and other interactions, such as fluctuations in the ground potential as the transient passes.

Methods for simulating or electrically reproducing the EM induced transients has to date been accomplished through laboratory bench test techniques. It is currently unknown what the implications may be of introducing a transient through only one specific wire in a test experiment. One feature of EM transients is that they may affect all the components of a redundant system at the same time. This massive injection of faults could effectively circumvent the fault detection and recovery process. It would seem likely that the system would be affected in parts. Some parts of the system may be most vulnerable to perhaps an inductive coupling of a current to a wire, whereas other parts of the system will have difficulty with a ground potential fluctuation. Designing a test program to address all of these issues is very challenging. Of course, economics and time are always the practical limitation of any test capability.

Another consideration is the timing of the transient injection. It may be logical to inject the faults using the same clock used for the rest of the testing system. However, if this timer is also driving the test subject, the result could be a synchronous injection of the fault occurring at the same time as changing system states. This would likely influence the data with an unrealistic bias. A technique for random asynchronous fault injection would

more closely resemble actual environments. Detail studies could benefit from different time phase injections relative to system clock pulses.

Experimental investigations of EM transient induced faults may be further divided into two realms of testing depending upon the level of system definition in the test. As discussed in section 2, tests may be performed entirely as simulations on computers.

A real or conceptualized system may be modeled down to the functional block, VLSI chip, or the transistor gate level using available circuit and logic simulators. The simulated system is then stepped in time through a series of tasks as fault signals are injected into circuit nodes using ideal voltage or current sources. This technique has the advantage of being able to completely control where faults are injected, and at precisely what time and amplitude. Simulations can trace exactly what the injection produces throughout the entire system. The limitation of this method is in the expense (both time and money) of the computer time required to step the entire system through any substantial amount of real time, not to mention analyzing the enormous amount of data which is generated. Furthermore, a software simulation can never precisely parallel the actions of a real system.

At the other extreme, hardware testing physically injects EM transient currents or voltages into a real system.

An actual system may be brought into the laboratory, and the system altered to allow transient injection and monitoring of pertinent system data. Care must be taken not to perturb the true functionality by loading the system's hardware with the transient injection equipment or the data monitoring equipment. The obvious advantage of the real system method is that a real system is observed in an actual functional environment while subjected to controlled EM transients. A very broad range of system states may be investigated in this manner over long periods of time. The disadvantages to this method are also numerous. It may be very difficult to access some pertinent data points; fault injection levels are much less controllable (although more realistic), and tracking fault effects through the system may be difficult.

3.2.2 Upset and Failure Detection

Component disruptions in FT systems must be handled through conventional fault tolerant methods such as redundancy and reconfiguration. Digital systems can be particularly vulnerable to EM transients. System functions can be disrupted in two ways:

- The components themselves may be permanently damaged by a high potential or current. It is known that charge accumulation levels above nine picocoulombs injected into a node within a CMOS microcomputer VLSI chip can cause permanent damage (Nichols et al. 1985). The speed and high magnitudes of many EM transients can easily lead to such charge accumulations.

- Component functions may be disrupted without damage. This is termed an "upset". The normal functionality of a system or component may be altered

by the passing of a transient. The loss of function may persist after the transient has gone, although no actual physical damage occurs.

A simple example of a serious disruption would be a logic gate temporarily stuck in a single state. This particular upset could cause a microprocessor to access an erroneous address space, program execution to branch to a subroutine intended for a completely different function, or incorrect mathematics to occur in an internal register.

Testing components and systems for EM transient fault tolerance is also a new area of research. A few pioneering efforts have been made on general purpose microprocessors executing simple codes, but these systems are not necessarily representative of actual avionic systems. Two recent works (Belcastro 1989 and Carreno, Choi, and Iyer 1989) provide sources for references to earlier studies. To date there are very few studies which have been performed on actual avionic systems; section 3.2.3 describes two such studies on an avionic electronic engine controller. These studies are detailed in Belcastro (1989) and Carreno, Choi, and Iyer (1989).

The two studies covered in section 3.2.3 are particularly beneficial in that each study was performed in a different way using the same electronic engine controller. One study completely utilized the software simulation approach and the other study utilized actual hardware systems. Aside from these strategic differences, the two testing methods have much in common. It is likely that future EM transient testing strategies will not deviate too far from these fundamental practices.

To monitor a system for fault occurrences, the tester needs to know the "normal" system function. The known "normal" function detail must match the same level of detail as the desired testing level of the target system. If the testing is to be done down to the transistor level in a logic system, then the "normal" function of the system at the transistor level must be known.

This knowledge can be gained in several ways as discussed in section 2. One extreme (all in software) is to run a testing algorithm through the system (simulated or actual) without the injection of any faults. The system can then be rerun with precisely the same algorithm with fault injection. The second extreme (all in hardware - used primarily for actual hardware testing) is to run two or more systems in parallel with one system remaining non-faulted as the reference comparison for normal operation.

The multiple real hardware method is limited by such factors as system size, cost, availability, and manufacturing repeatability. Using real hardware is usually faster and more convenient for fault detection testing since detection may occur in "real time" through direct comparison of system states. The single system method is often more economical, slower, and may provide better comparison data since exactly the same algorithm is run through the same system.

The types of system data collected during EM transient testing will probably not differ much between the two testing strategies. This system data will normally include control lines, address buses, data buses, and microprocessor status registers. The system control lines establish the operational mode of the

system. These will often give the most obvious indication of incorrect operation. (The control lines are also probably one of the most significant sources for EM transient conduction to the sensitive digital components.)

The system data buses should be monitored for signs of incorrect command modes and erroneous data accessed (or induced). This data may provide the earliest indication of upset or failure. Similarly, the address buses provide indication of invalid memory access and program execution. This is obviously crucial data for detecting and tracking upset/failure response. Microprocessor status registers may be helpful in tracking a fault occurrence and response. The microprocessor status registers typically include operational information flags, such as parity, configuration, self-tests, watchdog timer, and arithmetic overflow.

3.2.3 Digital Engine Controller Test Examples

Two complementary studies, Belcastro (1989) and Carreno, Choi, and Iyer (1989), serve as excellent examples for describing the proposed processes of determining EM transient fault behavior in avionic systems. Belcastro (1989) actually documents the plan for a test which had not been completed at the time of its publication. However, the testing should be complete by the time of this publication. Carreno, Choi, and Iyer (1989) documents in detail an actual test completed in late 1989. These are two of the first such tests to be performed in the area of EM transient upset detection in actual avionic systems. Both of these examples offer tremendous insight into the current state of the art in EM transient fault modeling, detection, and injection. It must certainly be anticipated that this type of work will greatly increase in the coming years.

Both tests utilize the same Hamilton Standard electronic engine control (EEC) unit. This controller is a commercial unit for Pratt & Whitney engines. A block diagram of the EEC system is shown in figure 3.2-2. The fault tolerant strategy utilizes a dual-channel system with an extensive self-testing algorithm to determine which channel is in control of the engine.

3.2.3.1 Physical Transient Insertion

The testing strategy of Belcastro (1989) tests real hardware using a single system run in a non-faulted condition. This run establishes normal mode reference data and ranges. Another run is then made in which the hardware is subjected to EM transient waveforms coupled inductively onto wiring in the EEC. The EEC simulation strategy and a functional block diagram of the test system are shown in figures 3.2-3 and 3.2-4, respectively.

The EEC hardware is altered for test to allow access of the data bus, address bus, and the control lines of the microprocessors in each channel. Digital data is recorded through a DAS8200 digital analysis system using up to 240 input capture lines.

Upset detection is defined as the occurrence of one of three conditions:

- Selected parameter values for engine speed, throttle resolve angle, and inlet air temperature out of range for n cycles.

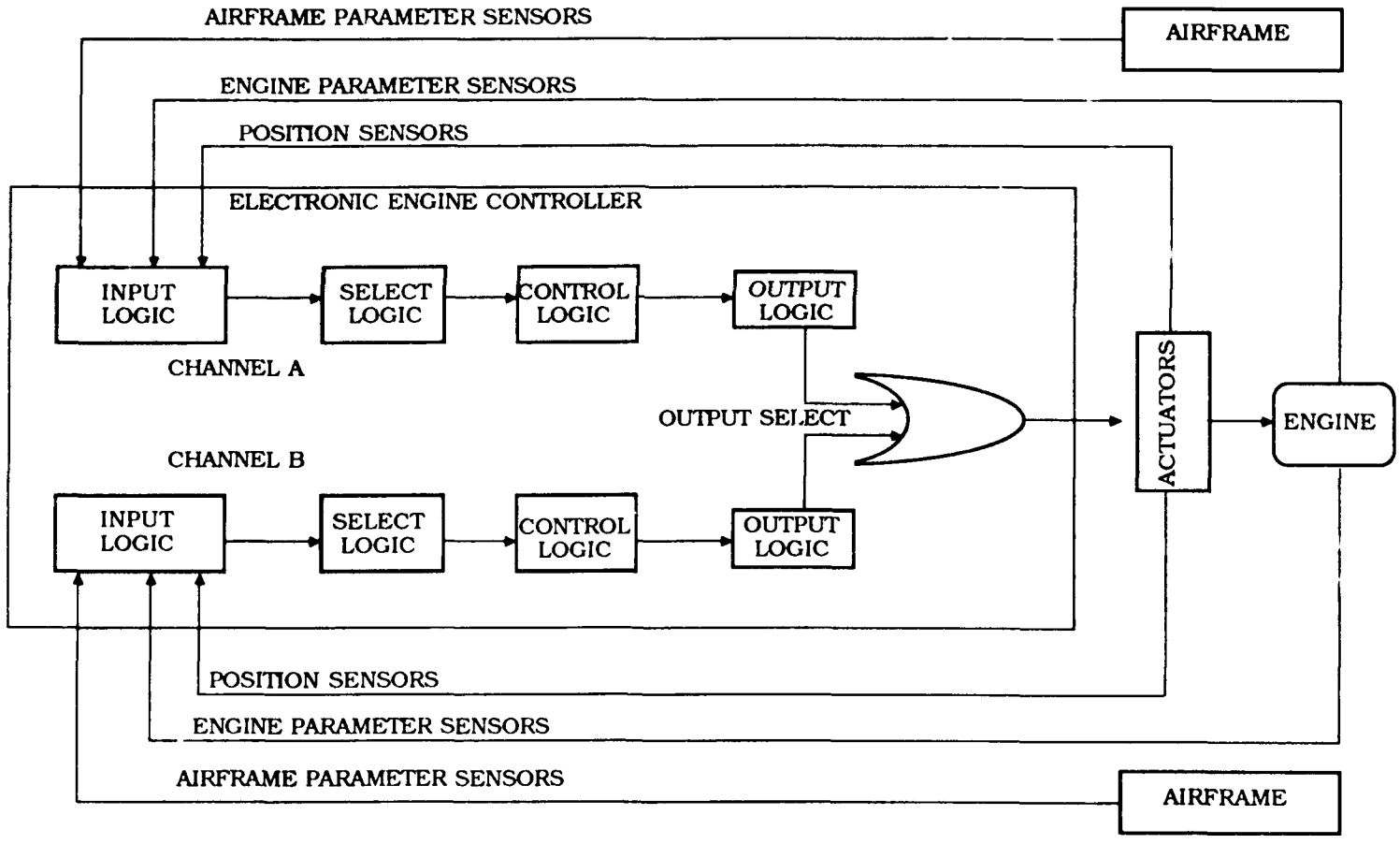


FIGURE 3.2-2. FUNCTIONAL BLOCK DIAGRAM OF THE HAMILTON STANDARD ELECTRONIC ENGINE CONTROLLER

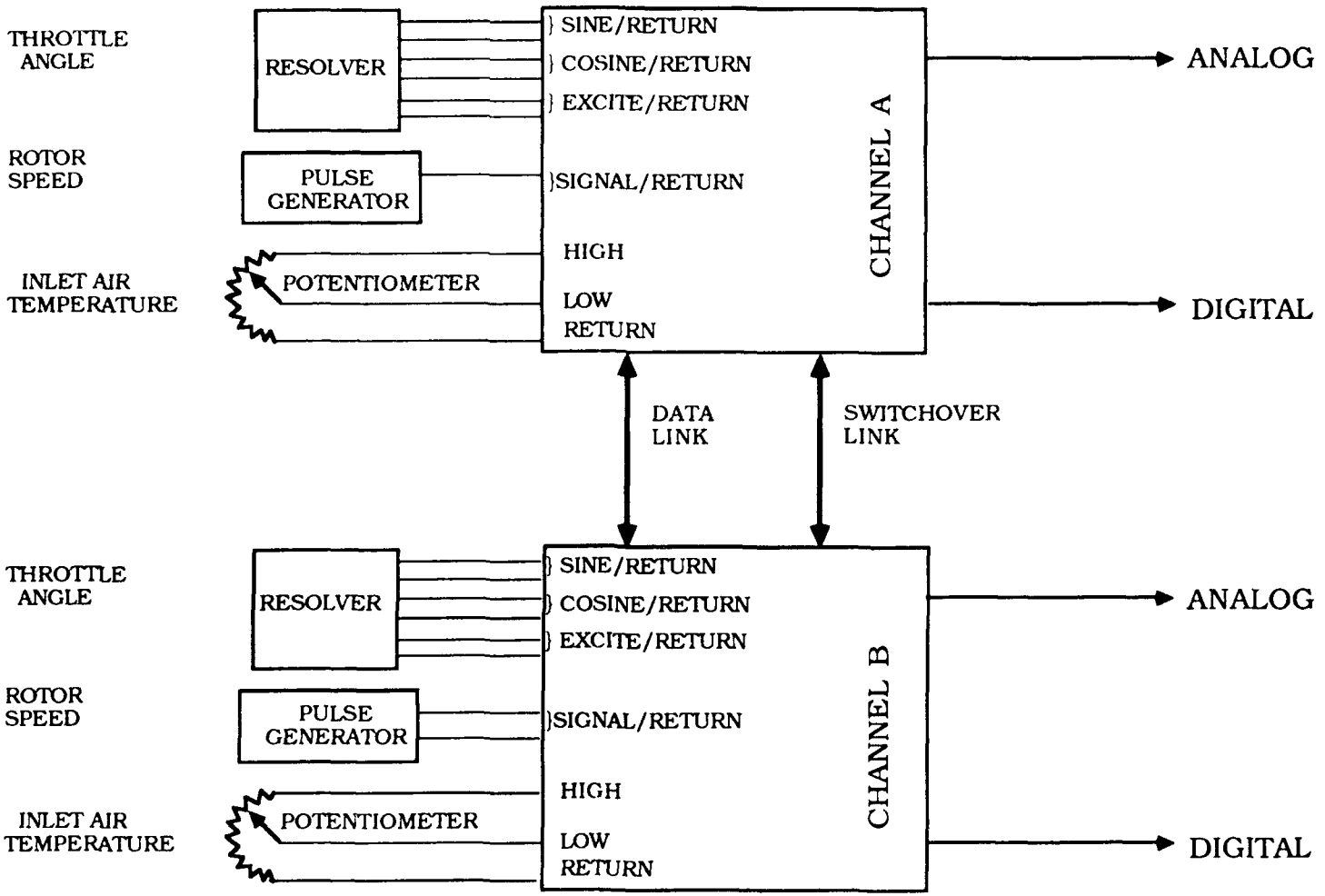


FIGURE 3.2-3. SIMULATION OF EEC INPUTS FOR EM TRANSIENT FAULT INJECTION

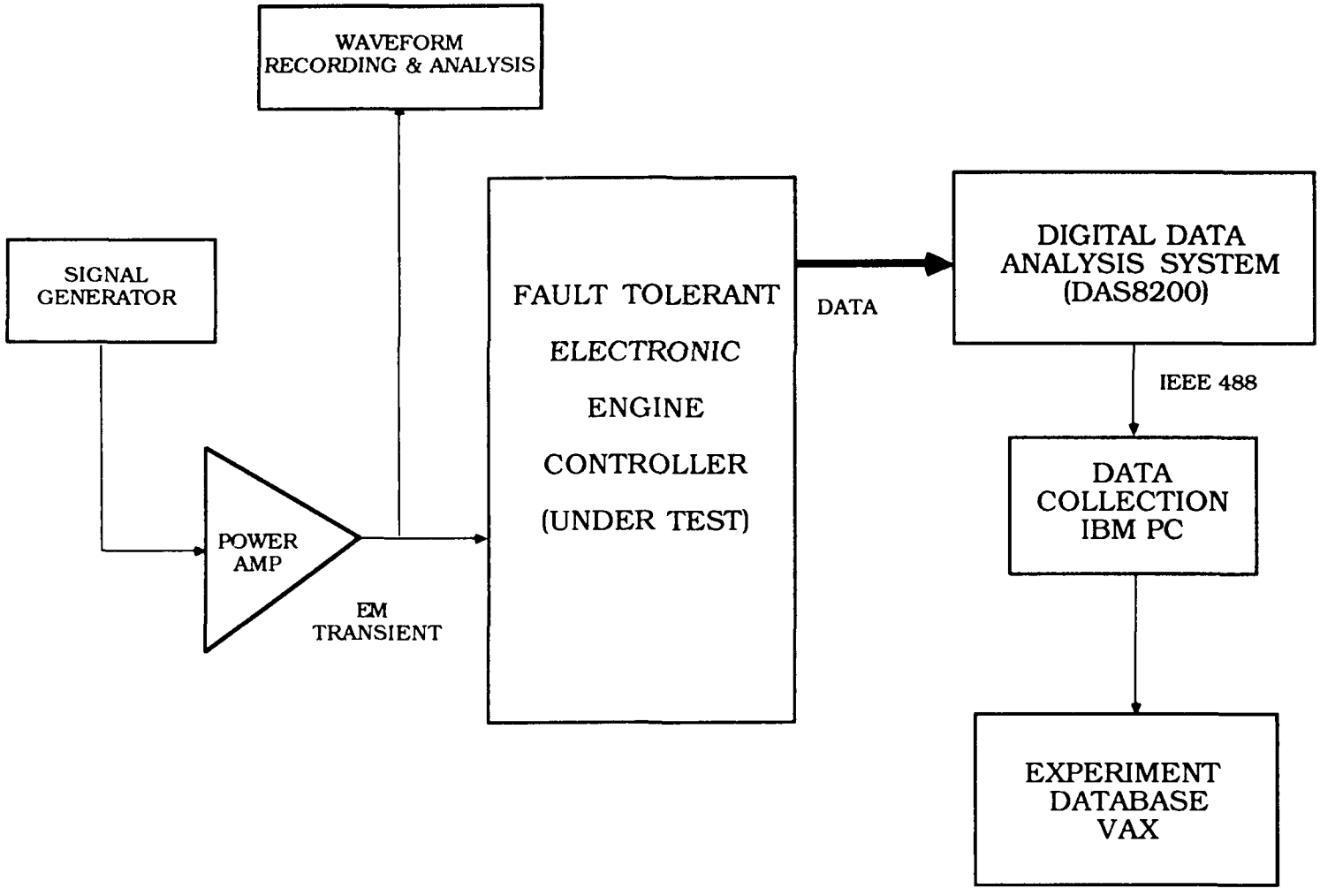


FIGURE 3.2-4 EEC INSTRUMENTATION FOR EM TRANSIENT FAULT INJECTION

- Control commands obtained from the data bus out of range for a given flight mode for n cycles.

- Invalid memory space accessed for n cycles.

Indication of any of these events in the monitored data will result in time-stamped data being recorded for that particular test run; that is, only data from upset/failure detection will be recorded.

EM transient injection is accomplished with an inductive coupler using an exponentially damped sinusoid transient. This test equipment is capable of delivering a 1,000 watt signal over a frequency range of 10 kHz to 220 MHz.

The EEC will be cycled through eight simulated flight conditions utilizing the simulation strategy depicted in figure 3.2-3 with nominal flight parameter values, including most actual engine parameters, stored in a ROM. The EEC will experience idle, acceleration, take-off, climb, cruise, deceleration, reverse, and partial power.

The objectives of initial testing are to help establish a methodology for upset detection strategies, to add to the EM transient fault tolerance database, and to further define characteristic induced-waveform threshold norms leading to upset conditions.

The results of the actual experiment outlined by Belcastro (1989) will complement the experiments reported by Carreno, Choi, and Iyer (1989).

3.2.3.2 Simulated Transient Pulse Injection

The study by Carreno, Choi, and Iyer (1989) approaches testing of the EEC from a software perspective. One channel of the EEC, excluding the interfaces, was simulated with the aid of an application program supplied by Hamilton Standard. Hamilton Standard also supplied a complete gate-level description of the entire 16-bit HS1602 microprocessor used in the EEC. The gate-level description was then further reduced to a CMOS transistor-level description of the gate-level modules. Capacitance loads in the circuit were calculated from metalization lengths used in the circuit layout.

The simulation was carried out using the SPLICE1 mixed-mode simulation program. This program is capable of performing both analog circuit analysis and logic simulation. SPLICE1 was modified to allow for specification of nodal EM transient injections (as opposed to reconfiguring the circuit model by adding a voltage or current source to the individual nodes one at a time). The SPLICE1 simulator was also modified to allow for manipulation of external data files in order to simulate access of memories (RAM, ROM). The program was expanded to output up to 80 nodes' data only when data changed, and to provide an extended trace facility which generates output for a node each time a gate driving that node is evaluated by SPLICE1. This latter feature is particularly helpful when tracing fault propagation through the nearly 4,000 nodes of the HS1602 microprocessor model.

The simulated EM transient fault was of the double exponential type illustrated in figure 3.2-1. A current source was used for the injection. This current source had a rise time constant of 0.05 nanoseconds and a fall time constant of 0.164 nanoseconds. The amplitude of the source varied so as to produce a well defined charge accumulation in the range of 0.5 to nine picocoulombs. This method was used because it was found that the total charge accumulation was the more significant factor in determining upset rather than amplitude alone.

The operation of the microprocessor was categorized on a six part functional modular level: watchdog self-test, multiplexer, decoder, control, countdown, and arithmetic logic unit (ALU). The microprocessor was stepped in one nanosecond increments (82 steps per 12.2 MHz clock cycle) through 74 instruction cycles which included a watchdog test, parity test, instruction set test, RAM test, ROM sum test, and test case parameter transfers to RAM.

Simulations were performed with "gate distances" of one, two, three, four, and five modeled to the transistor level. Gate distance is defined in figure 3.2-5. It was observed that for a combinatorial circuit a minimum of three gate distances are needed for accurate simulation, and at least four gate distances are needed if at least one latch (i.e. flip-flop) is within the three gate distance.

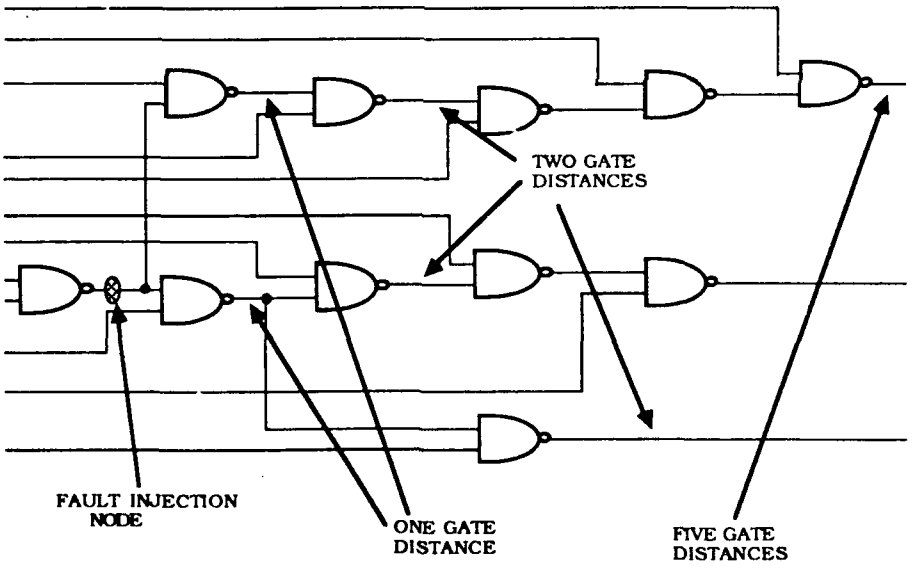


FIGURE 3.2-5. CLARIFICATION OF GATE DISTANCE DEFINITIONS

A non-faulted simulation run was used to generate the normal operation comparison data. This data was also compared to data obtained from an actual hardware unit including a comparison of such details as electrical delays, gate loading, etc. Correct digital data was observed in the simulation including

transitional times to within three nanoseconds of the actual system. These results suggest a high confidence in the modeling accuracy.

A total of 2,100 EM transient injections were performed by randomly selecting seven nodes from each of the six primary functional modules mentioned above. Charge accumulation levels of 0.5, one, two, three, four, five, six, seven, eight, and nine picocoulombs were injected at five different times during the program execution.

Errors were defined as first order if a logic error (difference from the comparison normal run) occurred one clock cycle after the transient injection. The error becomes second order if the logic difference persists two or more cycles after the transient fault injection. These definitions are also used to define pin errors if the device's pins were actually affected. A functional error was noted if the processor's operational function deviated from normal.

Table 3.2-1 summarizes the total error occurrence for the entire test as a function of the accumulated charge level induced by the simulated transient injection. Table 3.2-2 lists the occurrence of all first order errors (most sensitive detection) encountered within each of the six functional modules.

Both of these tables provide significant information about the sensitivity of the HS1602 EEC microprocessor. There is little sensitivity to transient charge accumulations of two or less picocoulombs. There are increasing errors with increasing charge accumulation between two and six picocoulombs. Charge levels above six picocoulombs do not increase upset errors. (Above nine picocoulombs permanent damage begins to occur.)

There is a very strong sensitivity of the microprocessor with respect to functionality. The watchdog self-test and the ALU are both very likely to be upset by this type of EM transient. The decoder, control, and countdown functions are about half as likely as the ALU and watchdog to suffer upset, whereas the multiplexer seems to be rather immune to the transient.

The results of this experiment offer a significant contribution to the growing database for EM transient injection experimentation in avionic systems. These results will be very interesting to compare with the results of the first test described in this section since the two experiments offer two very different perspectives on the same system.

It should be noted that significant computer resources are required. This experiment covers only a very small fraction of the whole system with respect to both components and function time. A simulation run of 12 two-input logic gates and two D-type flip-flops modeled at the transistor level running for 10 milliseconds requires about 130 hours of computer time on a MicroVAX II computer. This will obviously limit the simulation capability of complex avionic systems. However, the first test discussed will not likely produce the highly detailed results of this technique which provide definition of how the microprocessor upset occurs, and particularly in detail within the VLSI chip.

TABLE 3.2-1. ERRORS IN SIMULATED EM TRANSIENT ON EEC

INJECTION CHARGE LEVEL (Picocoulombs)	LOGIC ERRORS First Order	LOGIC ERRORS Second Order	PIN ERRORS First Order	FUNCTIONAL ERRORS
0.5	1	0	0	0
1.0	2	0	0	0
2.0	9	0	5	7
3.0	38	10	23	17
4.0	52	14	30	24
5.0	64	15	34	29
6.0	72	17	37	29
7.0	76	19	41	29
8.0	77	22	42	29
9.0	79	23	43	29

TABLE 3.2-2. ERRORS IN SIMULATED EM TRANSIENT ON MICROPROCESSOR MODULE

MICROPROCESSOR FUNCTIONAL MODULE	TOTAL FIRST ORDER ERROR INCIDENCE	ERROR INCIDENCE PERCENTAGE (Per Injection)
WATCHDOG	132	37.7%
MULTIPLEXER	0	0.0%
DECODER	68	19.4%
CONTROL	70	20.0%
COUNTDOWN	70	20.0%
ALU	130	37.1%

BIBLIOGRAPHY

Arnold, T. F., "The Concept of Coverage and Its Effect on the Reliability Model of a Repairable System", IEEE Transactions on Computers, Vol. C-22, pp. 251-254, March 1973.

Avizienis, A. and J. Laprie, "Dependable Computing: From Concepts to Design Diversity", Proceedings IEEE, 74(5):629-638, May 1986.

Avizienis, A. and D. Rennels, "Fault-Tolerance Experiments with the JPL Star Computer", COMPCON '72 Digest of Papers, pp. 321-324, September 1972

Baker, Robert, Scott Mangum, and Charlotte Scheper, A Fault Injection Experiment Using the Airlab Diagnostic Emulation Facility, NASA CR 178390, NASA Langley Research Center, March 1988.

Bavuso, Salvatore J. and Paul S. Miner, "Characterization of the Faulted Behavior of Digital Computers and Fault Tolerant Systems", Presented at the Hawaii International Conference on System Sciences, Honolulu, HI, January 1989.

Bavuso, Salvatore J., et al., Applications of the Hybrid Automated Reliability Predictor, NASA Technical Paper 2760, NASA Langley Research Center, December 1988.

Becher, Bernice, Diagnostic Emulation: Implementation and User's Guide, NASA CR 178391, NASA Langley Research Center, December 1987.

Belcastro, Celeste M., "Laboratory Test Methodology for Evaluating the Effects of Electromagnetic Disturbances on Fault-Tolerant Control Systems", NASA Technical Memo 101665, NASA Langley Research Center, November 1989.

Benson, J., D. Mulcare, and W. Larsen, Hardware Fault Insertion and Instrumentation System: Experimentation and Results, DOT/FAA/CT-86/34, FAA Technical Center, March 1987.

Butler, Ricky W. and Anna L. Martensen, The Fault-Tree Compiler (FTC), Program and Mathematics, NASA Technical Paper 2915, NASA Langley Research Center, July 1989.

Butler, Ricky W. and Allan L. White, SURE Reliability Analysis, Program and Mathematics, NASA Technical Paper 2764, NASA Langley Research Center, March 1988.

Carreno, Victor A., G. Choi, and R. K. Iyer, "Analog/Digital Simulation Study of Transient Induced Logic Errors and Upset Susceptibility of an Advanced Control System", NASA Technical Memo XXX, Draft, November 1989.

Carter, William C., et al., Design for Validation: An Approach to Systems Validation, NASA CR 181835, NASA Langley Research Center, May 1989.

Cohen, Gerald C., Charles W. Lee, and Daniel L. Palumbo, "Methods for Evaluating Integrated Airframe/Propulsion Control System Architectures", IEEE-Naecon, Dayton, OH, May 1987.

Cohen, Gerald C., et al., AIAA-89-2703, "Experiences With a Prevalidation Methodology for Designing Integrated/Propulsion Control System Architectures", AIAA/ASME/SAE/ASEE 25th Joint Propulsion Conference, Monterey, CA, July 1989.

Cooley, W. W. and D. Shortess, Lightning Simulation Test Technique Evaluation, DOT/FAA/CT-97/38, FAA Technical Center, October 1988.

Courtois, B., "Some Results About the Efficiency of Simple Mechanisms for the Detection of Microcomputer Malfunctions", 9th International Symposium of Fault Tolerant Computing, pp. 71-74, 1979.

CSDL, The Charles Stark Draper Laboratory, Inc., Document #43-334, CBD Announcement Ref SS017, Issue PSA-9214, November 1986.

Cullyer, W. J. and C. H. Pygot, "Application of Formal Methods to the VIPER Microprocessor", IEEE Proceedings, Vol 134, Pt. E, No. 3, May 1987.

Czeck, E. W., Z. Z. Segal, and D. P. Siewiorek, Software Implemented Fault Insertion an FTMP Example, Department of Computer Science, Carnegie-Mellon University, January 15, 1989.

Czeck, E., D. Sieworek, and Z. Segal, Software Implemented Fault Injection, an FTMP Example, NASA CR-178423, Carnegie Mellon University, October 1987.

Czeck, Edward W., Daniel P. Siewiorek, and Zary Z. Segal, Predeployment Validation of Fault-Tolerant Systems Through Software-Implemented Fault Insertion, NASA CR 4244, NASA Langley Research Center, July 1989.

Czeck, Edward W., et al., Software Implemented Fault Insertion: An FTMP Example, Department of Computer Science, Carnegie-Mellon University, January 1987.

Decouty, B., G. Michel, and C. Wagner, "An Evaluation Test of Fault Detection Mechanisms Efficiency", 10th International Symposium on Fault Tolerant Computing, pp. 225-227, 1980.

Dotson, Kelly J., "Examples of Nonconservatism in the CARE III Program", NASA Technical Memo 100524, NASA Langley Research Center, January 1988.

Dotson, Kelly J., Analysis and Testing of the SURE Program, NASA Technical Paper 2817, NASA Langley Research Center, August 1988.

Dzwonczyk, M. J., et al., Avionic Architecture Studies for the Entry Research Vehicle, NASA CR 181828, NASA Langley Research Center, May 1989.

Ferguson, F. J., Inductive Fault Analysis of VLSI Circuits, PhD Thesis, Carnegie-Mellon University, Electrical and Computer Engineering Department, October 1987.

Ferguson, F. J., J. P. Shen, and W. Maly, "Inductive Fault Analysis of MOS Integrated Circuits", IEEE Design and Test of Computers, 2(6):13-26, December 1985.

Finelli, G., "Characteristics of Fault Recovery Through Fault Injection on FTMP", IEEE Transactions on Reliability, R-36(2):164-170, June 1987.

Finelli, George B., AIAA-88-4436, "Results of Software Error-Data Experiments", AIAA/AHS/ASEE Aircraft Design, Systems and Operations Conference, NASA Langley Research Center, September 1988.

Goetz, F., "Design for Detection, an Attempt at Complete Fault Detection of State", Digest of Papers, COMPCON '78, pp. 325-328 1972.

Iyer, R. K. and D. J. Rossetti, "A Measurement Model for Workload Dependence of CPU Errors", IEEE Transactions Computers, vol. C-35, pp. 511-519, June 1986.

Johnson S. J., "Evaluation of Fault-Tolerant Parallel Processor Architectures Over Long Space Missions", NASA Memorandum 4123, NASA, 1989.

Johnson, Sally C., "Assist User's Manual", NASA Technical Memo 87735, NASA Langley Research Center, August 1986.

Kurlack, R. and J. Chobot, "CPU Coverage Evaluation Using Automatic Fault Injection", 4th AIAA/IEEE Digital Avionic Systems Conference, pp. 294-300, 1981.

Lala, Jaynarayan H. and Basil T. Smith, III, Development and Evaluation of a Fault-Tolerant Multiprocessor (FTMP) Computer, Vol. II, FTMP Software, NASA CR 166072, NASA Langley Research Center, May 1983.

Marchall, P., "Updating Functional Fault Models for Microprocessors Internal Buses", 15th International Symposium of Fault Tolerant Computing, pp. 58-64, 1985.

Martensen, Anna L. and Salvatore J. Bavuso, "Tutorial and Hands-On Demonstration of a Fluent Interpreter for CARE III", NASA Technical Memo 4011, NASA Langley Research Center, November 1987.

McGough, John G., Digital Systems Validation Handbook Volume II, "Latent Faults", DOT/FAA/CT-88/10, 1988.

McGough, John G. and Fred L. Swern, Measurement of Fault Latency in a Digital Avionic Mini Processor, NASA CR 3462, Bendix Corporation, October 1981.

McGough, John G. and Fred L. Swern, Measurement of Fault Latency in a Digital Avionic Mini Processor, Part II, NASA CR 3651, NASA Langley Research Center, January 1983.

McGough, J. G., Swern, F., and S. Bavuso, "New Results in Fault Latency Modeling", Proceedings of the IEEE EASTCON Conference, pp. 299-306, August 1983.

Meissner, C. W., Jr., J. R. Dunham, and G. Crim, "NASA-LaRC Flight-Critical Digital Systems Technology Workshop", NASA Conference Publication 10028, NASA Langley Research Center, April 1989.

Migneault, Gerard E., "On the Diagnostic Emulation Technique and Its Use in the Airlab", NASA Technical Memo 4027, NASA Langley Research Center, October 1988.

MIL-HDBK-217E, "Reliability Prediction of Electronic Equipment", Distribution Statement A, DOD/AMSC, October 1986.

Nichols, W., et al., "Trends in Parts Susceptibility to Single Event Upset", IEEE Transactions Nuclear Science, vol. NS-32, no. 6, December 1985.

Padilla, Peter A., "In-Circuit Fault Injector User's Guide", NASA Technical Memo 100478, NASA Langley Research Center, June 1987.

Padilla, Peter A., "FTMP Data Acquisition Environment", NASA Technical Memo 100636, NASA Langley Research Center, July 1988.

Padilla, Peter A., "Fault Recovery Characteristics of the Fault Tolerant Multi-Processor", NASA Memo L-16630, NASA Langley Research Center, to be published as a NASA report 1990.

Rasch, N., User's Manual for AC-20-53A Protection of Airplane Fuel Systems Against Fuel Vapor Ignition Due to Lightning, DOT/FAA/CT-88/3, FAA Technical Center, October 1984.

Rothman, Elizabeth, et al., HARP: The Hybrid Automated Reliability Predictor, Introduction and Guide for Users, Version 6, Department of Computer Science, Duke University, NASA Langley Research Center, April 1989.

SAE Committee AE4L, Recommended Draft Advisory Circular: Protection of Aircraft Electrical/Electronic Systems Against the Indirect Effects of Lightning-Induced Transients, Report AE4L-87-3, Revision A, October, 1988.

Schuette, M., et al., "Experimental Evaluation of Two Concurrent Error Detection Schemes", 16th International Symposium of Fault Tolerant Computing, pp. 138-143, 1986.

Smith, T. Basil, III, and Jaynarayan H. Lala, Development and Evaluation of a Fault-Tolerant Multiprocessor (FTMP) Computer, Volume I, FTMP Principles of Operation, NASA CR 166071, NASA Langley Research Center, May 1983.

Strickland, Michael J. and Daniel L. Palumbo, Fault Tolerant System Performance Modeling, AIAA, Boeing, 1988.

Trivedi, Kishor S. and Robert M. Geist, A Tutorial on the CARE III Approach to Reliability Modeling, NASA CR 3488, Duke University, Prepared for NASA Langley Research Center, 1981.

NASA Conference Publication 2114, Working Group Meeting I, "Validation Methods for Fault-Tolerant Avionics and Control Systems", NASA Langley Research Center, March 1979.

NASA Conference Publication 2114, Working Group Meeting II, "Validation Methods Research for Fault-Tolerant Avionics and Control Systems", NASA Langley Research Center, December 1979.

Yang, X., W. York, and D. Siewiorek, "Fault Recovery of Triplicated Software on the Intel iAPX 432", Distributed Computing Systems, pp. 438-443, 1985.

GLOSSARY OF TERMS

BYZANTINE RESILIENCE. A fault tolerant process which is tolerant of intermittent faults that can send good information part of the time.

COVERAGE. The conditional probability of the system successfully recovering from a component fault and continuing to perform the intended functions correctly, given the presence of the fault. Coverage is the measure of effectiveness of a system's utilization of redundant hardware. Coverage can be qualified and applied to many different components of a system and phases of recovery process. Examples include, fault detection coverage, fault isolation coverage, latent fault coverage, sensor failure coverage, and memory failure coverage.

ELECTROMIGRATION. *Drifting of metal atoms toward the cathode of a cathode ray* tube.

FAILURE. The deviation of system behavior from specifications (arithmetic failure, storage failure, flight control function failure.)

FAILURE, HARD. Repeated use of the same input and initial conditions results in the same incorrect response.

FAILURE MECHANISM. Any situation that could produce an error condition. Examples of failure mechanisms include metal migration, voltage overstress, and lack of air-conditioning.

FAILURE, PERMANENT. Repeated use of the same input and initial conditions results in the same incorrect response.

FAILURE, SOFT. Repeated use of the same input and initial conditions does not result in the same incorrect response.

FAILURE, TEMPORARY. Repeated use of the same input and initial conditions does not result in the same incorrect response.

FAILURE, TRANSIENT. Repeated use of the same input and initial conditions does not result in the same incorrect response.

FAULT. The phenomenological reason for a failure (open wire, stuck-at fault, design fault, etc.). In general, any condition preventing a digital component from correctly changing state when directed to change by input parameters. For electrical components there is a one-to-one correspondence between faults and failures. The situation is not so simple with digital circuits. For if the circuit is S-A-1, any input causing a one output will be correctly processed; a little like the stopped clock that is correct twice per day. For a processor

having a million or so logic gates, it is not possible to test for all the combinations of input and output states.

FAULT, LATENT. A fault which has not yet caused a failure. (For example, a fault in a memory chip that is not being used for the foreground program or in this particular mode of the system is a latent fault.)

FAULT, STUCK-AT. A logic signal which remains at zero (S-A-0) or one (S-A-1).

FAULT TOLERANT SYSTEM. A system that continues to function although certain components may have faults.

SYSTEM RELIABILITY. The probability of performing a given function from the some initial time, t=0, to time t.

ACRONYMS AND ABBREVIATIONS

AC	Advisory Circular
AE4L	SAE Subcommittee (Lightning)
AIAA	American Institute of Aeronautics and Astronautics
AIRLAB	Avionics Integration Research Laboratory
ALU	Arithmetic Logic Unit
AM	Amplitude Modulated
AMSC	Document number prefix used by the Department of Defense
ASEE	American Society of Electrical Engineers
ASME	American Society of Mechanical Engineers
ATTR	Attribute
B-GLOSS	Gate Logic Software Simulator developed by Bendix
C/I	Communicator Interstage
CAD	Computer Aided Design
CAP	Collins Application Processor
CARE III	Computer Aided Reliability Evaluator
CBD	Commerce Business Daily
CMOS	Complementary Metal-Oxide Semiconductor
CPA	Central Processor - A
CPU	Central Processing Unit
CR	Contractor Report
CSDL	Charles Stark Draper Laboratory
CTA	Collins Test Adaptor
DE	Diagnostic Emulation
DEFN	Definition
DEV	Development
DMA	Direct Memory Access
DOD	Department of Defence
EEC	Electronic Engine Control
EM	Electromagnetic
EMI	Electromagnetic Interference
EMR	Electromagnetic Radiation
ESS	Electronic Switching System
EXCHNG	Exchange
EXP	Experiment
FAA	Federal Aviation Administration
FCC	Flight Control Computer
FCR	Fault Containment Regions
FCS	Flight Control System
FET	Field Effect Transistor
FI	Fault Insertion circuitry
FIAT	Fault Injection Automated Testing
FIIS	Fault Insertion and Instrumentation System
FIM	Fault Injection Manager
FIRE	Fault Injection Receptor
FM	Frequency Modulated

FMECA	Failure Modes and Effects Criticality Analysis
FT	Fault Tolerant
FTC	Fault Tree Compiler
FTMP	Fault Tolerant Multiprocessor
FTP	Fault Tolerant Processor
GE	General Electric
GEN	Generation
GGLOSS	Generalized Gate-Level Logic System Simulator
GLOSS	Gate Logic Software Simulator
GPC	General Purpose Computer
HARP	Hybrid Automated Reliability Predictor
HDBK	Handbook
HERF	High Energy Radio Frequency
I/O	Input/Output
IBM	International Business Machines
ICIS	Intercomputer Interface Sequencer
ICS	Intercomputer Sequencer
ID	Identification
IEEE	Institute of Electrical and Electronics Engineers
IGGLOSS	Gate Logic Software Simulator (improved version developed at NASA Langley)
IOPA	Input/Output Processor - Channel A
IOS	Input/Output Subsystem
JPL	Jet Propulsion Laboratories
KHz	KiloHertz
LAN	Local Area Network
LaRC	Langley Research Center
LRU	Line Replaceable Unit
LSI	Large Scale Integration
M	Mutual (when used with RLC)
MDICU	Modular Digital Interface Conversion Unit
MHz	MegaHertz
MIL	Military
MOS	Metal-Oxide Semiconductor
MPX	Multiplex
MSI	Medium Scale Integration
Mux	Multiplexed
Naecon	National Avionics and Electronics Conference
NAND	Not AND
NASA	National Aeronautics and Space Administration
NEMP	Nuclear Electromagnetic Pulse
OS	Operating System
PAL	Programmable Array Logic
PAWS	Padé Approximation With Scaling
PC	Personal Computer
PMS	Physical Message Switch
PROC	Processor
QUAD	Quadruple
RAM	Random Access Memory
RCVR	Receiver
RDFCS	Reconfigurable Digital Flight Control System
REG	Register

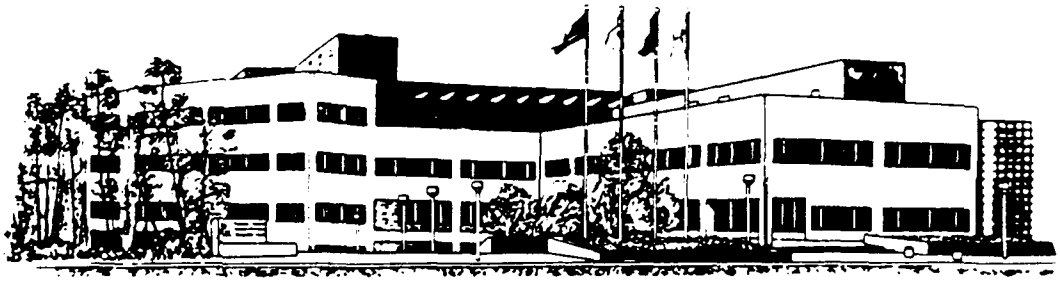
RF	Radio Frequency
RLC	Resistance/Inductance/Capacitance
ROM	Read Only Memory
RT	Remote Terminal
RTI	Remote Terminal Interface
S-A-0	Stuck-At-Zero
S-A-1	Stuck-At-One
S-GLOSS	Gate Logic Software Simulator developed by Stevens Institute
SAE	Society of Automotive Engineers
SEU	Single Event Upset
SHRD	Shared
SQL	Software Query Language
SSI	Small Scale Integration
STEM	Scaled Taylor Expansion Matrix
SURE	Semi-Markov Unreliability Range Evaluator
SYN	Synch
TTL	Transistor-Transistor Logic
TX	Transmit
UNIBUS	Universal Bus
USEG	Unsegmented
VLSI	Very Large Scale Integration
XAB	Transmit Compare A B
XMT	Transmit
XOR	Exclusive OR

Department of Transportation
Federal Aviation Administration

DOT/FAA/CT-88/10

HANDBOOK-VOLUME II
DIGITAL SYSTEMS VALIDATION

CHAPTER 15
ELECTROMECHANICAL ACTUATOR SYSTEMS
(ELECTRICAL SYSTEMS CERTIFICATION ISSUES)



PREPARED BY:

COMPUTER RESOURCE MANAGEMENT, INC.
950 HERNDON PARKWAY, SUITE 360
HERNDON, VIRGINIA 22070

PREPARED FOR:

FEDERAL AVIATION ADMINISTRATION
TECHNICAL CENTER
ATLANTIC CITY INTERNATIONAL AIRPORT, NEW JERSEY 08405

TABLE OF CONTENTS

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LISTS OF TABLES

1.0 Electromechanical Actuation Systems Technology

1 Introduction and Scope

This chapter covers the technological description of electromechanical actuation systems (EMAS) including technology trends in these devices and certification issues involving actuators, electric power, and digital engine controls. The scope of this chapter, illustrated in figure 1.1-1, covers actuators, engine controls, and electrical power. The present electromechanical actuation technology is reviewed in section 2; particularly actuator devices for use on advanced technology aircraft. Technology and issues related to digital engine controls are included in section 3. Electric power systems technology is discussed in section 4 with emphasis on the technology trends of future generation aircraft.

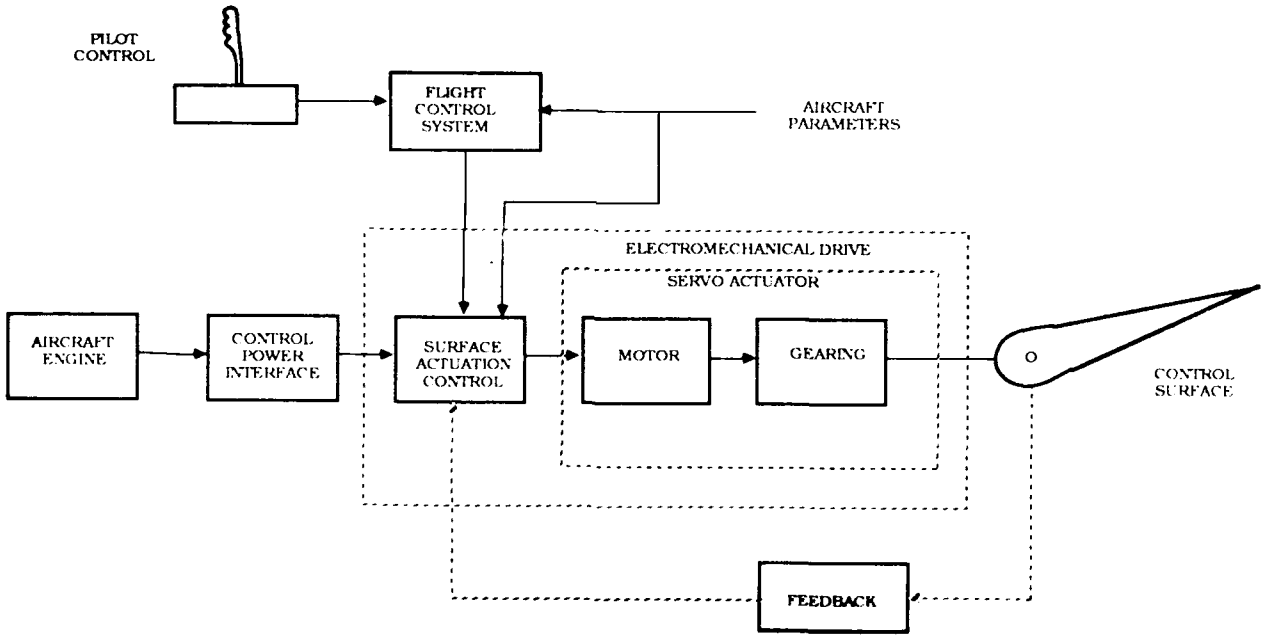


FIGURE 1.1-1. ELECTROMECHANICAL ACTUATION ELEMENTS

1.2 Emerging Technology

Areas of emerging advanced technology aircraft include the V-22 tilt rotor, all weather aircraft. This aircraft was originally intended for use by the military, but could find application in commercial operations, perhaps as a local interurban feeder to large airport hubs like the Los Angles, Dallas-Ft.

Worth, New York, Chicago, and Washington, D.C. areas. Many other potential application areas could be found within the U.S. and internationally.

Another important area of future application includes faster commercial transports, made either by the U.S. or international consortiums. There is renewed interest by Aerospatiale in developing the British/French Concorde into a larger faster model. The National Aeronautics and Space Administration (NASA) has initiated research on an aerospace plane that has captured the interest of U.S. industry and has achieved limited congressional funding.

Any of these applications will require systems having higher reliability than today's EMAS while operating in far more severe thermal, shock, and vibration environments.

1.2.1 Electromechanical Actuation Systems for V-22

One view is that the V-22 aircraft is a flying actuator (refer to figure 1.2-1 from McManus 1985). Actuation systems are required for thrust control, flight control and stability augmentation, and configuration change-over from vertical to horizontal flight.

This aircraft may tolerate failures for a very short time during configuration changes or even during normal flight for some systems. Requirements for reliability and redundancy for commercial service should be more severe than for more conventional configurations of transport aircraft, due to flights over populated areas and the unique variable configuration.

1.2.2 All-Electric Airplane

The use of electric power rather than hydraulic power for actuator systems has several advantages (Leonard 1983, AiResearch 1976). An industry trend toward high voltage (270 volt dc) distribution, proposed and investigated by the Naval Air Development Center and several major airframe companies, may make it possible to operate actuators and control systems without hydraulics (Perkins and Marek 1977). This trend is based on the advent of permanent magnetic motors providing high power in a small space. The maintenance and repair of such all-electric systems would be simplified. For example, no hydraulic system bleeds or pressurization would be needed when maintaining or repairing an actuator.

Since electric power is more easily distributed, redundancy could be implemented by having several actuators rather than a single one driving a torque tube connected to several surfaces.

Three types of all-electric actuators that have been developed and evaluated include linear, rotary, and rotary-hingeline. Although electromagnetic actuation systems could eventually surpass electrohydraulic reliability, the results to date indicate that electrical actuator reliability has not been much improved over hydraulic actuators.

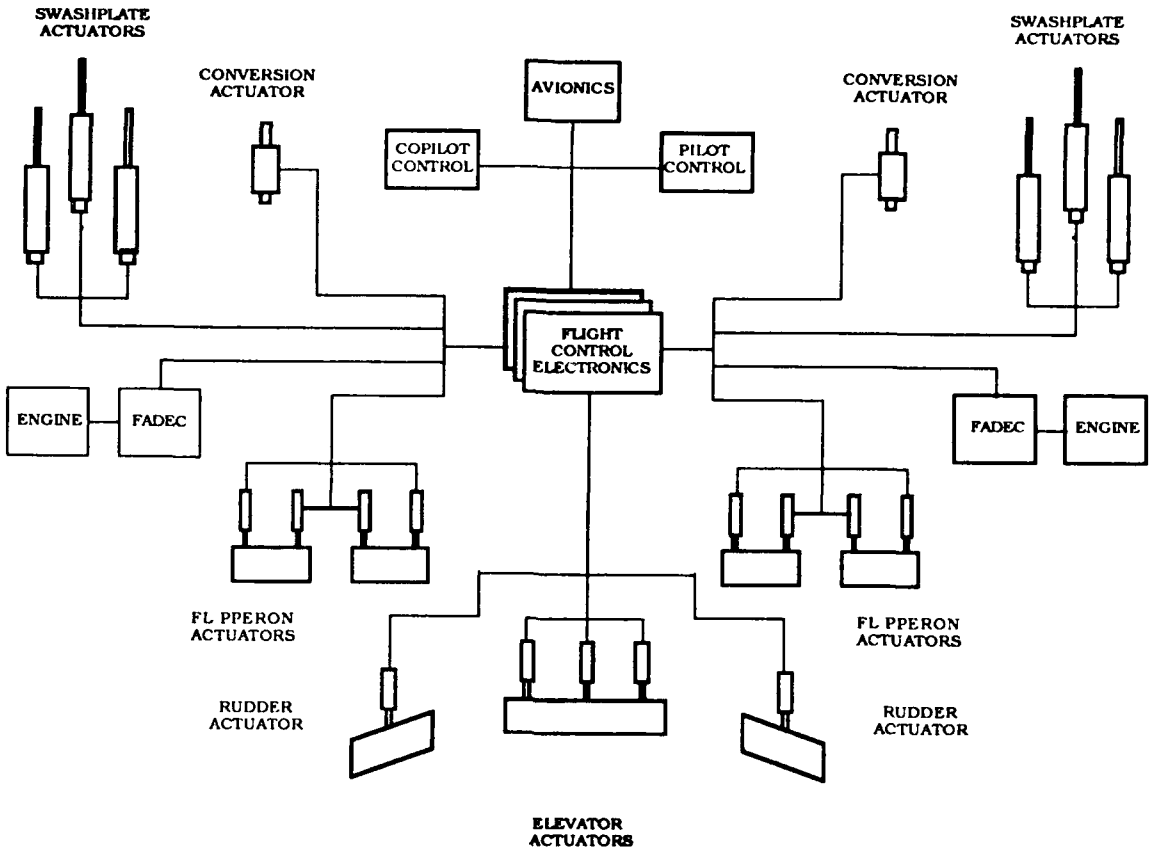


FIGURE 1.2-1. V-22 ACTUATION SYSTEMS

1.3 Technology Needs for an Aerospace Plane

An aerospace plane, described in Jorgensen 1983, is capable of flying intercontinental distances in commuter aircraft times. This imposes the most severe requirements for reliability and redundancy. Safe flight and immediate landing following a critical system failure are not possible from low earth orbit altitudes. In addition, a principal economic advantage of this aircraft lies in having flight operations with a round trip in a single crew day. This requires a low factor of unplanned maintenance. A two-hour turn-around is anticipated for the aircraft.

Flight operation at low earth orbit or suborbital altitudes requires flight critical systems having two orders of magnitude greater reliability than for present aircraft. Achieving these levels of reliability makes fault detection, isolation, and reconfiguration a necessity.

1.4 Electromechanical Actuation Systems Trends

In support of aircraft technology trends, such as those highlighted in this section, actuation system technology needs more reliable devices. One way of

accomplishing increased reliability is by using simpler devices. Current research and development is attempting to increase reliability as well as find ways for effectively applying computer technology. This is particularly true for the direct drive hydraulic actuators and electric actuators described in section 2. Military system applications have higher operating pressures for hydraulic systems and thus smaller and lighter actuator systems. This should also affect commercial actuation systems. The trend toward high voltage electric distribution is aided by a shift toward 270 volt dc power as an answer for providing an uninterruptable source of dc power for onboard computers. Newer engine control applications require additional computers on the engines. These additional computers provide automatic engine optimization (fuel versus power) during flight.

In addition to the development of the electromechanical actuation devices, research efforts by NASA Langley are also underway. NASA research will develop and demonstrate the means for controlling sensitive computer devices that must operate in severe aircraft electrical environments (i.e., electrical noise from onboard systems and from external sources of noise such as Electromagnetic Interference (EMI), lightning, and High-Intensity Radiated Fields (HIRF) environments). A multiyear advanced technology demonstration program is planned by NASA Langley to develop and demonstrate the technology for application of fiber optic devices as needed for the control of onboard computer devices. The program emphasizes Fly-By-Light and Power-By-Wire (FBL/PBW).

Fiber optic technology holds the promise for assuring that sensitive digital data is immune to electrical interference from external sources. The NASA program will include development and flight test of fiber optic devices to provide interconnects for digital control signals. This will be in the form of a fiber optic network data bus. Present computer devices may be interfaced to the network by bus interface devices that convert the electrical signals to optical signals. The electrical power to move surfaces and actuators will likely be provided by electrical means as for present systems.

2.0 Smart Servos and Actuators

A wide range of devices are currently available to operate the control surfaces on advanced technology aircraft. Improvements in hydraulic control devices and control configurations for fault detection and isolation greatly improve the reliability and performance of these devices. A realm long held by hydraulic actuators is being invaded by all-electric devices for control and actuation. This is due to the discovery of rare earth permanent magnetic materials. These materials allow operation of high power electric motors with small size and weight, and low standby power requirements.

2.1 Electrohydraulic Actuators

Hydraulic actuators operate via pressure applied to a power ram piston that converts the pressure to a linear or rotary motion. The positioning of a control surface is accomplished by directing the force into a mechanical linkage to the surface. The force is altered by directing pressure onto either side of a piston.

A simplex actuator, shown in figure 2.1-1, relies on only one hydraulic fluid and a single power ram piston. This configuration is widely used for ground based equipment and for some non-critical aircraft applications. It is not reliable enough for aircraft control in flight critical systems. While other parts of the actuator may be fault tolerant, any failure of the power ram or hydraulic system will disable the actuator. Therefore, for flight critical applications, power rams have two pistons in tandem, as shown in figure 2.1-2. Each piston is supplied from a separate hydraulic system. These actuators are designated as dual tandem actuators.

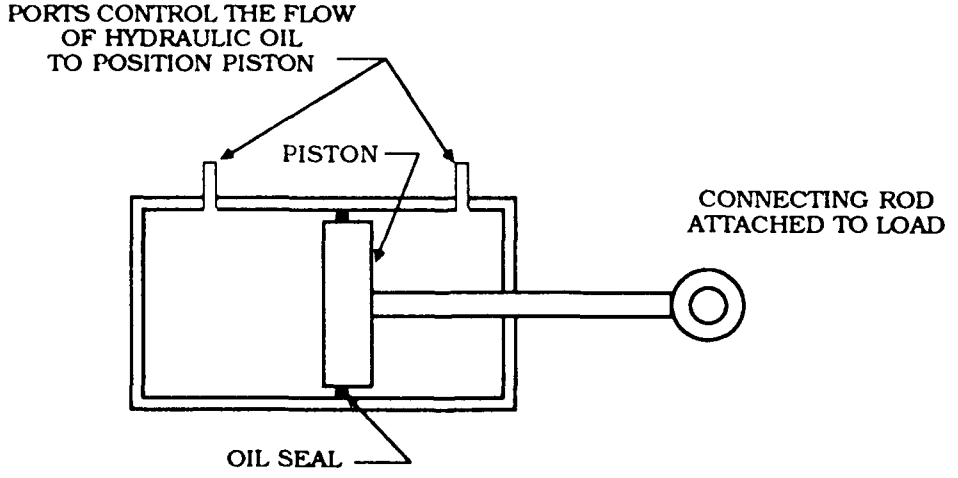


FIGURE 2.1-1. SINGLE POWER RAM HYDRAULIC ACTUATOR

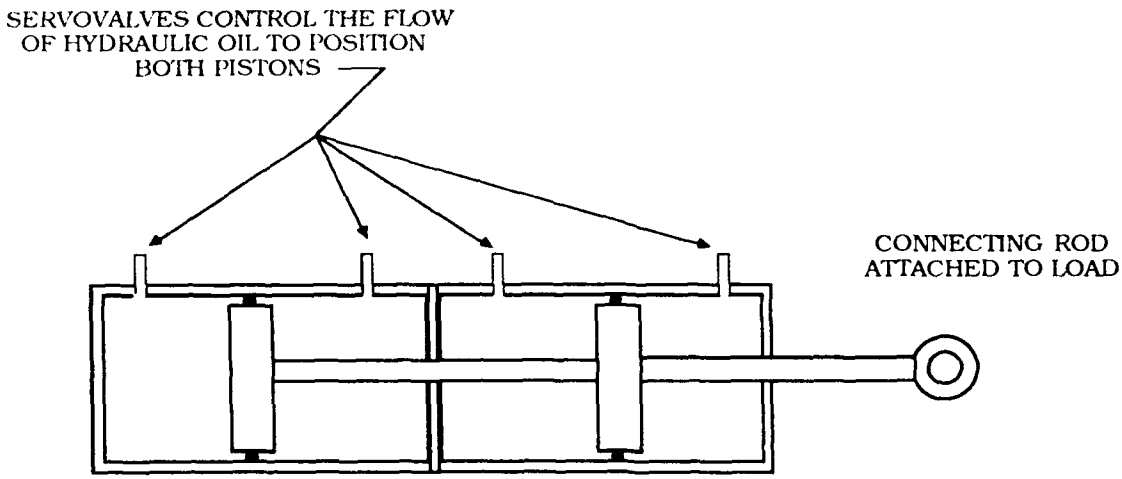


FIGURE 2.1-2. DUAL TANDEM POWER RAM ACTUATOR

A typical operational configuration for controlling a dual tandem actuator is illustrated in figure 2.1-3. These actuators use several stages to convert and amplify an electrical signal or mechanical input into useful motion. While current dual tandem actuators may differ from this diagram in some manner, the diagram conveys an understanding of the operation.

The first stage normally consists of three or four jet pipe or flapper valves that convert the input into a differential pressure for driving the second stage servovalve spool. The first two stages are often combined into a single unit called a two-stage electrohydraulic servovalve. With these devices, the spool position is controlled by feedback (normally mechanical) of the spool position to the first stage. The schematic and operation of a two-stage electrohydraulic control valve is shown in figure 2.1-4.

The second stage supplies hydraulic fluid to modulator pistons or servo ram pistons which in turn position the dual main control valve. The second stage may alternatively position the main control valve mechanically. Considerable force can be developed by these actuators (tens of thousands of pounds for large actuators). Because of limited power gain, the force to move the main control valve is also large. This two-stage control of the hydraulic flow to the main ram piston is the normal mode of operation. Closed loop control of the power ram position is often used. This control may be implemented by mechanical, analog, or digital processes. For mechanical feedback, the linear variable differential transformers (LVDTs) are replaced by mechanical linkages.

One experimental class of actuators consists of systems for eliminating the main control valve. These systems use more powerful 2-4 stage electrohydraulic control valves, or even electric motors, to operate the main control valve directly, as shown in figure 2.1-5.

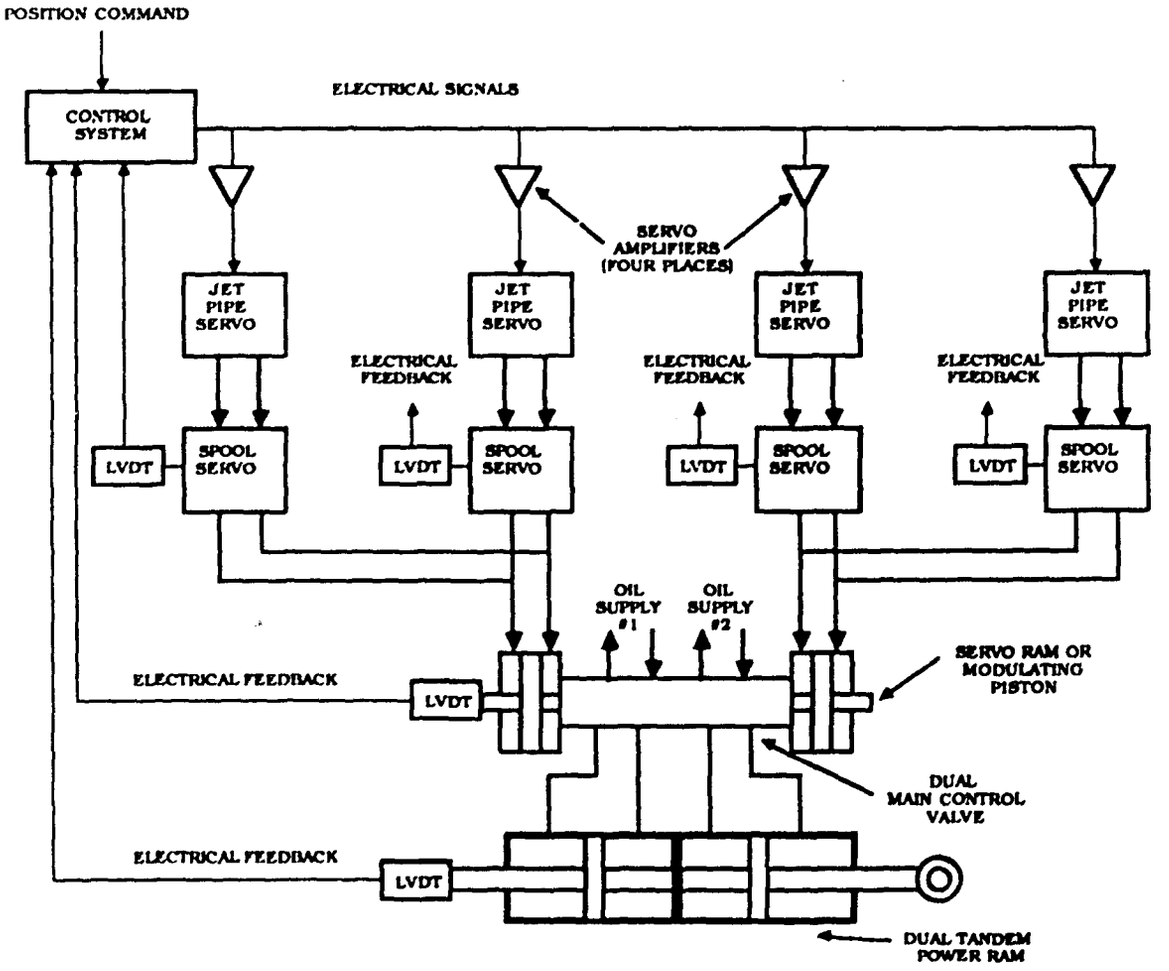
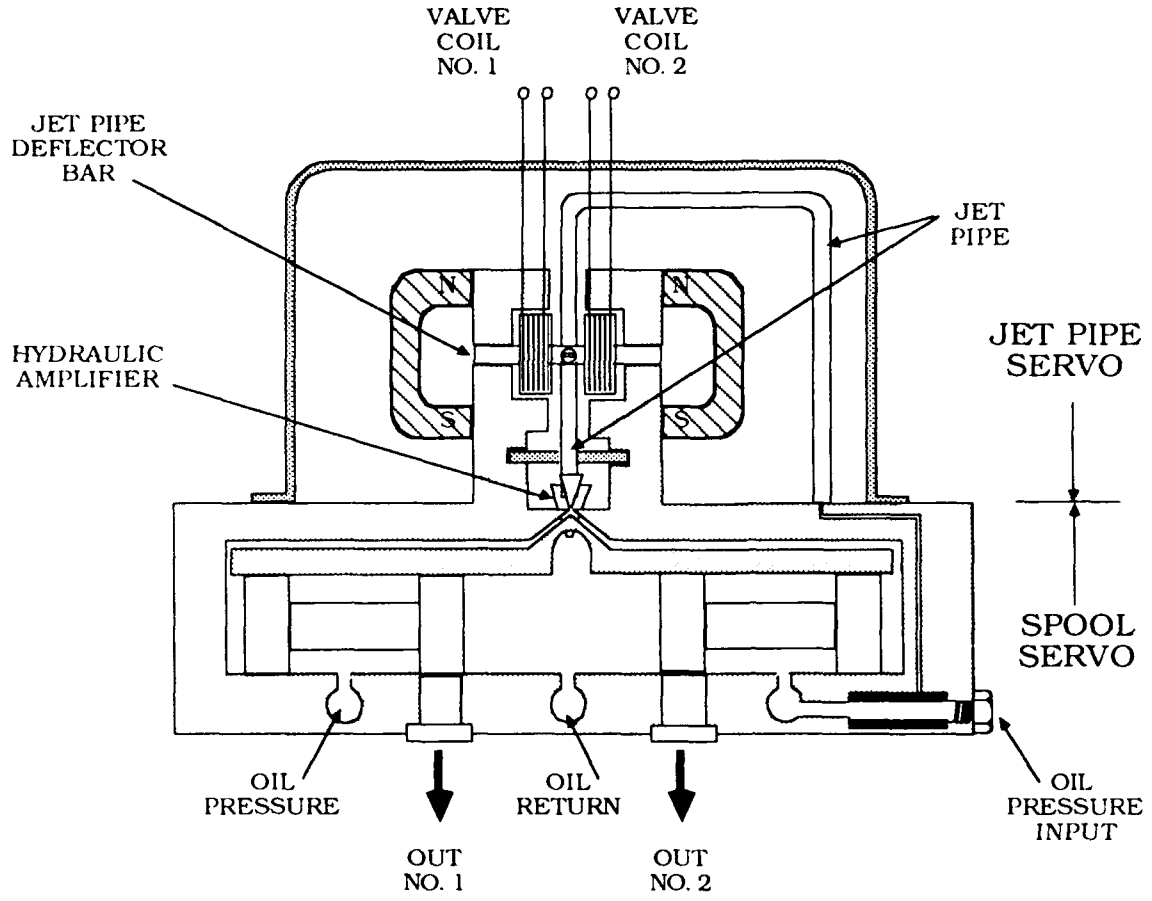


FIGURE 2.1-3. TYPICAL OPERATIONAL DUAL TANDEM ACTUATOR

2.1.1 Reliability and Redundancy of Electrohydraulic Actuators

Fault detection and isolation systems vary widely among actuator systems. These variations are the result of design decisions and design philosophy. Each example described in detail in the following sections represents a system at a different stage of technology. Current technology includes the in-place operational systems; research includes systems under development and laboratory prototypes. In these examples, the designers have the goal of increasing reliability of the actuation systems. Several fighter aircraft actuation systems are discussed. These systems are for application where extreme reliability is required because these systems must operate without failures in single thread fly-by-wire flight control systems.



The principal of operation is as follows:

Hydraulic fluid flows out of the jet pipe to the first stage of the hydraulic amplifier.

The first stage consists of the transmitter orifice on the end of the jet pipe and the two receiver oirifices below the transmitter orifices.

When the jet pipe is in position shown, the two receivers orifices have equal pressure and the piston will remain balanced between the two output ports.

By deflecting the jet to the right or left, the pressure will increase on the between the right and left orifices. This results in motion of the second stage spool and therefore a hydraulic flow to the actuator.

The jet piston is controlled by the force of generated by the electric currents flowing in the coils #1 and #2 and the magnets on the left and right and the mechanical feedback spring

FIGURE 2.1-4. OPERATION OF ELECTROHYDRAULIC SERVOVALVE

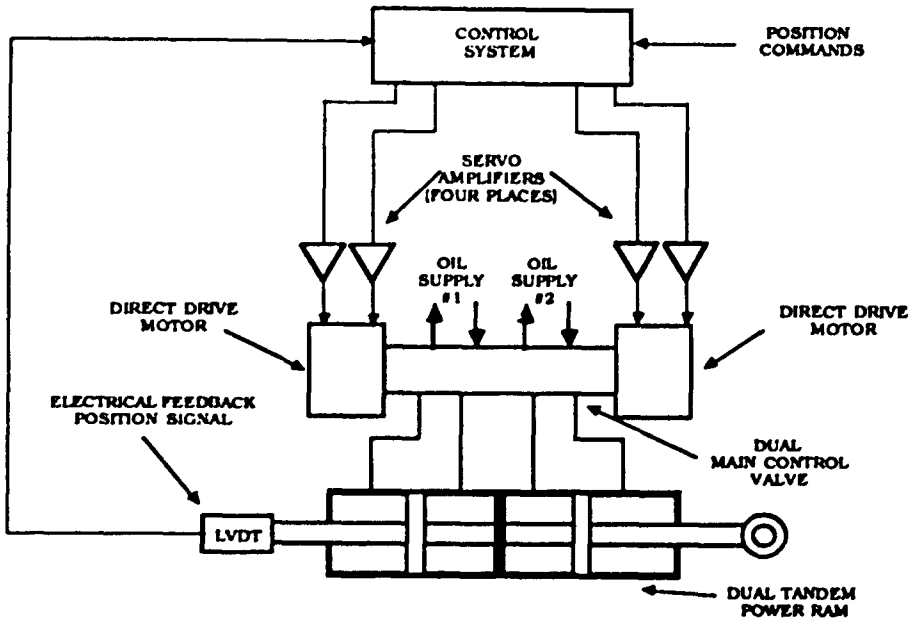


FIGURE 2.1-5. DIRECT DRIVE DUAL TANDEM ACTUATOR

The detection of component failure (and reconfiguration) must be considered in advanced fly-by-wire systems. One consequence of component failure may be the loss of the aircraft.

The most reliable hydroelectric actuators are built with the dual-tandem concept described in the previous section. This class of actuators uses more stages for amplification. Consequently, smaller first stage servovalves can be used, reducing the hydraulic power loss of the actuator. A major benefit of the main control valve is that it isolates the effect of load from the servovalves and produces a dynamic response having considerable stiffness. The loss of an electrohydraulic servovalve results in degraded performance (reduced response bandwidth) rather than total failure.

Current research is examining the removal of the main control valve in the dual tandem design. It is hoped that this modification will lead to a more reliable actuator having fewer parts. Although removal of the main control valve simplifies the design, more powerful two-stage servovalves are required. This design results in higher power loss and higher failure transients. In addition, less power is available for shearing metal chips that may jam the main ram piston.

A breakdown of failures for actuation systems was given in Kenmir 1972. This paper is relatively old, but indicates why the research trends are toward direct drive actuators. The following data indicates the percentage of operational failures due to actuator component parts:

- Tandem ram, control valves, bypass valves, and stabilization system 32%
- Servovalves and solenoid valves 45%
- Secondary Actuators 3%
- Transducers and connectors 20%

Increasing the system reliability by means of a dual channel will double the number of parts and thus double the total number of failures in a given time interval. The bulk of this increase is in servovalves and solenoid valves. Eliminating these parts would allow a significant increase in system reliability while having about the same number of failures requiring maintenance as for a single channel.

One approach to simplifying the design uses electric motors to control the main servovalve. Powerful dc motors, such as those presently used on direct drive experimental actuators, are needed to operate control valves with normal hydraulic pressures. The power loss of a standard two-stage design would be too large for a conventional servovalve to move the main control valve. Electric motors using rare-earth materials have the drive capability needed in a small size, weight, and power. At higher pressures, i.e., much greater than the normal 3000 pounds per square inch (psi), leakage around the piston becomes a significant problem. The dynamic performance is slower for this class of actuator compared to the direct drive or the conventional two-stage actuators.

The material in sections 2.1.2 and 2.1.3 is largely drawn from Baker and Bonnice 1988, and Jenney et al. 1989. These sections describe details of several selected actuators that include the following:

- F-16 Integrated Actuator
- F-18 Stabilator Actuator
- V-22 Swashplate Actuator
- Dynamic Controls F-16 Direct Drive Actuator
- Bell-4Valve actuator
- Boeing/Moog Digital Integrated Servo Actuator Controller (DISAC)
- Dynamic Controls All Digital Actuator
- Dynamic Controls F-15 Flutter Suppression Actuator

The first three, F-16, F-18, and V-22, are in use on recently developed military aircraft. They represent the best actuators currently in use. The rest of the listed actuators are experimental designs and indicate a variety of approaches for improving the performance or reliability of current designs. The Boeing Moog DISAC, Bell-4Valve, and F-16 direct drive actuators have eliminated the

main control valve. These designs are intended to improve reliability and performance. The Boeing Moog actuator is an experimental prototype developed to test microprocessor-based control, fault diagnosis, and failure management. The Bell-4Valve was developed for a flight test program. The F-16 direct drive actuator is a proof-of-concept prototype. The last two are unique conceptual prototypes built to demonstrate technology and evaluate new approaches for extended range of performance; as prototypes they have little built-in fault detection and isolation.

Table 2.1-1 shows a summary of the Fault Detection and Failure Management (FDFM) systems for the first six of these actuators (Baker and Bonnice 1988). For these systems the FDFM system is able to detect and reconfigure the actuators for servovalve, electrical, and sensor failures. No reconfiguration capability exists to address failures of the main control valve and power ram. The most likely common mode failure for the main control valve and power ram is jamming. The upper stages that drive these components are designed with sufficient shearing force to shear any metal chips that may jam the operation. An automatic reconfiguration capability exists for direct drive electrical motors. Since the most common failure modes are open or short circuits in the motor coils, the actuators are designed with several coils on a single solenoid and will continue operation with several coil failures. The shearing force and the dynamic response may be degraded due to a failure of one of the two motor coils.

Also not addressed in the systems examined are failures of the FDFM system components, including the fault detection and reconfiguration logic. This is especially true for logic circuits implemented hydraulically, or in analog circuitry where there is no redundancy. If the hydraulic or analog circuitry fails, the result could indicate a failure in error, or detect no failure at all. The latter case allows the actuator to continue operation but creates the situation where a subsequent failure may not be acted upon and a flight critical situation may arise without warning. In the case of false detection (isolation of a good component), the operation of the actuator may be shut off or degraded because of the failure in detection logic. Unless unnecessary loss of the actuator is critical to safe flight operations, false alarms are preferable upon failure detection. Digital implementations of the logic are more tolerant of faults since the logic is distributed to more than one processor.

In many cases, failures of fault reconfiguration components, such as bypass and solenoid valves and switches, are impossible to detect. This is because they are on-off devices and are not completely testable in flight. For example, if a valve or switch that is normally off fails in that position, the failure will not be detectable until the device is turned on. Failures that cannot be detected because they do not affect the present system operation are termed latent failures. Only when a component is commanded to turn on or off, and fails to act, will the failure be detectable. Since only one of the actuators investigated provided backup or redundant components for these devices (a backup coil for the DISAC actuator bypass valves), they are apparently reliable enough that fault detection is not required. Preflight testing of these devices would reduce the likelihood of flying with a failed reconfiguration device of this type.

TABLE 2.1-1. FAILURE DETECTION AND MANAGEMENT ON SIX SELECTED ACTUATORS

Component Failures	Actuators					
	F-16	F-18 stabilator	V-22 swashplate	Bell-4Valve	DISAC	Dynamic Controls direct drive
Servo amplifier and servovalve or direct drive motor coil	Direct comparison of servo amplifier current implemented in analog logic. Switches to standby amplifier and coil.	Servo amplifier current is compared to the output of a digital model. Eliminates failed electrical channel.	Servo amplifier current is compared to the output of a digital model. Hydraulic flow to the corresponding servovalve is shutoff.	Servo amplifier current compared to the output of a simple analog model. The corresponding EHSV is disengaged.	Not described in reference.	Cross-strapping design eliminates the need for explicit FDFM.
Servovalve or direct drive motor	Hydraulic comparison of output pressures (i.e., different pressure) from the first-stage servovalves. Switches to active standby servovalve if one of the primary servovalves has failed.	Hydraulic pressure output of paired first stage servovalves is compared. Hydraulic supply to the pair of servovalves with a discrepancy is shutdown.	Measured position of the second stage spool is compared to a digital model. Hydraulic pressure to the failed servovalve is shutoff.	Direct comparison of measured spool position implemented in analog logic. Control system accommodates small failures; hydraulic flow shutoff when failure detected.	Measured position of the second stage spool is compared to slow and fast models of the EHSV. Bypass valve actuated.	None
LVDT	None?	Position of the servo ram is compared to a model. Electric channel disabled.	Self-test. Hydraulic flow to the corresponding servovalve is shutoff.	Detected by servovalve detection test.	Self-test. Use redundant LVDT by switching microprocessor control?	Comparison of the two error signals to each motor. Command to the corresponding motor removed.
Other	Other failures detected by comparing the actual system output to the measured output. Solenoids are activated to mechanically center the power ram.	Loss of hydraulic power activates a bypass/damping valve.			Position sensor on the bypass valves allows preflight testing to detected bypass valve failure.	

Some failures are handled using the capability within some actuators and do not require an explicit FDFM system. For example, the Bell-4Valve and the F-18 actuators can accommodate hardover servovalve failures with closed loop control. For these actuators, the control system will cause the other three servovalves to move in opposition to the failed servovalve. The result is a force fight while maintaining an acceptable performance. Similarly, the cross strapped amplifier design on the Dynamic Controls direct drive actuator for the F-16 simply offsets the effect of one amplifier failure with an opposite current in the other amplifier. Force fighting is characteristic of many such passive failure reconfiguration schemes. The advantage of this approach is that no logic or reconfiguration devices are required. The disadvantage is that excess capacity, greater than for normal operation, is required demanding more power than necessary. In addition, force fighting will degrade performance, while increasing mechanical wear and fatigue, thus increasing the likelihood of subsequent failures. Unless the fault detection thresholds account for the effects of force fighting, false alarms will occur.

Fault detection and isolation logic is implemented in three basic ways: hydromechanically (F-16); in analog circuitry (F-16, Bell-4Valve, and Dynamic Controls direct drive); and in digital software (F-18, V-2, and DISAC). For digital implementations, the logic can be in the central Flight Control Computer (FCC) or a local microprocessor. The trend in technology is from hydromechanical to digital logic. The disadvantages of hydromechanical logic are additional cost, power, size, weight, and hydraulic complexity. More complex single thread systems are generally less reliable or at least require increased maintenance. In addition, hydromechanical logic can only be used for direct redundancy failure detection and must be simple in order to be more reliable than the devices monitored. Furthermore, this logic is inflexible because any modifications require a major effort.

Analog logic overcomes most of these objections. However, analog logic is primarily limited to direct redundancy comparisons and self-test; only a limited analytical redundancy capability is possible. Digital implementation of FDFM offers the best potential. Digital systems can use all of the detection approaches including self-test, direct redundancy, and analytical redundancy. In addition, digital implementations can fine tune the logic during the operations and maintenance phase, as may be needed. Benefits of a local microprocessor, rather than placing all FDFM in a central FCC, include reducing the overhead of the FCC, less cabling to the FCC, distributed processing, and better digital control response time. However, the local environment on the microprocessor may be a serious limitation for achieving high reliability.

While digital implementations offer the best performance, the F-18 and V-22 actuator systems had constraints imposed on them by the design of the fault tolerant computer system. Interfacing with the quadruplex FCC on the F-18 required separate electrical sensors and actuators for each of the four channels. Each of the four redundant components or sensors interfaced with one FCC. No crosstalking between systems was allowed. This approach eliminates using a direct comparison of sensor outputs for actuator fault detection. For this system, the response to channel faults is to eliminate the entire electrical channel if a single sensor or component fault is detected. The V-22 system is similar to the F-18 except a triplex FCC is used. The DISAC actuator also has

limited the communication between channels. These architectures with a "brick wall" between channels benefit from a simpler FDFM system at the expense of additional components and increased failure rate. Reliable communication between channels may be possible with increased complexity and increased risk in achieving reliability goals.

The performance of FDFM systems is determined by the performance of each individual detection test. Self-test, used to detect failures on the DISAC actuator and LVDT failures, is only able to detect certain specific failures. Direct and analytical redundancy approaches should result in better detection performance due to the ease and accuracy with which component responses can be modeled. In practice (Jenney 1977), the thresholds set for fly-by-wire systems are often 30 to 50 percent of the maximum level for hardover failure. As a result only failures near hardover conditions are detected. The remaining failures are compensated for by the control system. Setting lower levels results in an unacceptably high false alarm rate. False alarms are apparently more important to the designer than concern over missed failures (Biafore and Grieszmer 1983, and Weinstein 1987).

Following are three areas of potential improvements pointed out in Baker and Bonnice 1988:

- Reduced false alarm rate.

  The detection of failures in flight often cannot be duplicated on the ground. The two most common results during maintenance are "error condition could not be duplicated under test" and "component tests ok".

- More efficient FDFM design.

  Several practices tend to increase the need for maintenance and decrease reliability. The first practice is simply adding a single sensor to a component for fault diagnosis. Then a faulty component is indistinguishable from a sensor failure. The result is that good actuator components may be removed as failed. However, if analytical redundancy or some other method was used to detect sensor failure, the reliability could be increased.

  A second practice is including excess capacity to overcome failures by force fighting. While this approach may be the only one possible for a class of failures, several actuators use this approach simply to reduce the complexity of the FDFM system. In these cases, the result is increased size, weight, and power requirements. In addition, the increased wear from force fighting will ultimately increase the rate of failures.

  Actuation designs could also be more efficient if fewer sensors were used. In existing systems, the loss of a component results in the shut down of an entire channel. Reducing the number of sensors through analytical redundancy, for example, could reduce the number of channels from four to three. Of course the additional burden in computer resources and communications must be weighed against the overall benefit.

- Control system reconfiguration.

  None of the actuators reviewed alter the control system in any way following a failure. In addition to altering the control system to regain performance, reconfiguration could offer additional fault recovery options. Inner loop sensor failures might be compensated for by altering the feedback loop. Inner loop feedback improves the dynamic response of an actuator, but is often not absolutely necessary for continued emergency flight and landing. Effectiveness of control reconfiguration for failure management needs to be evaluated for each actuation system.

2.1.2 Detailed Examples of Current Technology

2.1.2.1 Standard F-16 Integrated Servo Actuator

Figure 2.1-6 is a schematic diagram for a standard F-16 actuator (Baker 1988). This actuator is used for controlling the horizontal tail, flaperons, and rudder of the F-16 aircraft. Because of the critical function of this actuator, it has considerable redundancy and fault detection and reconfiguration provisions. It is designed to accommodate dual electrical and hydromechanical failures. The actuator operates with a 3,000 psi supply pressure. The actuator has a force output of 36,000 pounds, a maximum slew rate of 5.5 inches per second, and a nominal frequency response having a 3 Hertz (Hz) bandwidth. This actuator is designed to accept two failures in the electrical system and one hydromechanical failure and continue to operate. Upon detection of a third failure the actuator will move to a centered position.

In the standard configuration (figure 2.1-6), the secondary actuator controls the position of the main control valve. The secondary actuator is controlled by three jet pipe servovalves. Failure detection and reconfiguration is accomplished hydromechanically using comparator spools to monitor the servovalve driving pressures and to drive current commands. A first failure of SV1 or SV2 will cause transfer of control to the standby servovalve SV3. A first failure of SV3 will lock the valve onto SV1 and SV2 control. Mechanical feedback loops are used for control of the servovalves, main control valve, and actuator ram. Solenoid valves are used to remove or apply pressure to the servovalves and the fail-center mode upon request from the actuator control electronics.

Note that the torque motors are not included inside the mechanical feedback loops. This results in the actuator position being commanded by current supplied to the servovalve coils.

The dual fail operation for electronic control is provided by each servovalve having two electrically separate windings. Either winding is capable of full servovalve control. These windings are driven by the FCC in active standby mode so that a first failure of a servo amplifier will be corrected within the flight control computer without activating any voting logic. The actuator control electronics incorporates analytical redundancy. This redundancy uses a mathematical model for the actuator position, including the dynamic rate plus position feedback signals. Comparison of the mathematically calculated position to the position measured using LVDT that is incorporated into the actuator

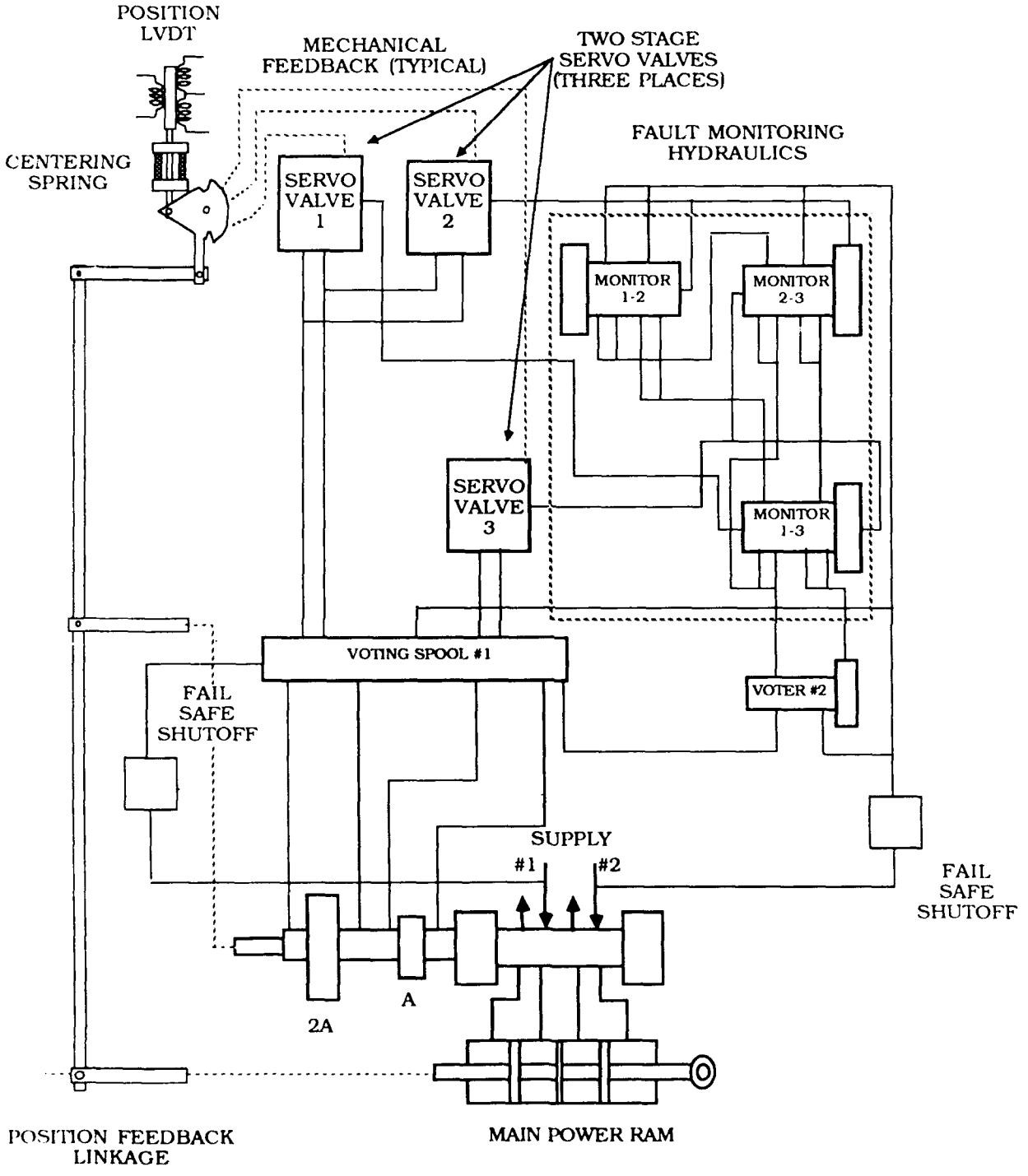


FIGURE 2.1-6. F-16 INTEGRATED SERVO ACTUATOR

determines an error signal. This error signal is used to determine if the actuator has failed. If the actuator has failed, move the piston to the centering mode of operation. In the centering mode, actuator position is determined by a centering spring attached to the position feedback linkage from the main actuator and the main control valve.

2.1.2.2 F-18 Actuator

The F-18 stabilator actuator (Harschburger 1983), illustrated in figure 2.1-7, has considerable redundancy and failure management. There are four single stage electrohydraulic servovalves which are paired to allow failure detection by comparison of their output pressures. This direct detection design uses four differential pressure transducers on each pair of servovalves. Failures of the servovalves, detected by these sensors, are contained by means of a solenoid valve which shuts off the hydraulic supply to that pair of servovalves.

Each of the servovalves is commanded electrically by four coils with the coil on one servovalve connected in series with a coil on each of the other servovalves. Each coil string is driven by a separate amplifier. Outputs from the four digital flight control computers are interfaced to a single amplifier supplying a string of coils. With this flux summing arrangement, the currents from the four separate amplifiers are effectively added.

Amplifier and coil circuit failures are detected by comparing the current from each amplifier to that calculated by a digital circuit model in the FCC. When a failure is detected, the amplifier for that coil channel will be disabled. This actuator design is able to meet performance specifications with two channel failures so that no failure management is needed. Detection of more than two channel failures results in the hydraulic flow to both pairs of servovalves being shut off, thus reverting to mechanical control.

Each pair of servovalves drives one piston of the dual tandem actuator. The two pistons are mechanically linked to a main control valve. This is motivated by the requirement for mechanical reversion in the event of a non-recoverable piston jam.

There are four LVDTs for sensing position for feedback on both the servo ram and the power ram. Each LVDT is connected to one of the four FCCs. Each FCC drives one of the servo amplifiers. Failure of an LVDT sensor or the command from the flight control computer is detected either by the servo amplifier current failure detection or by comparing the position of the servo ram with an analytical model. Sensor fault detection by direct comparison of sensor outputs is not possible because the digital quadruplex FCC architecture does not allow any cross-channel communication.

Common mode failures, resulting in loss of hydraulic power, are negated by a bypass valve that equalizes the pressure on either side of the ram piston, allowing the control surface to float.

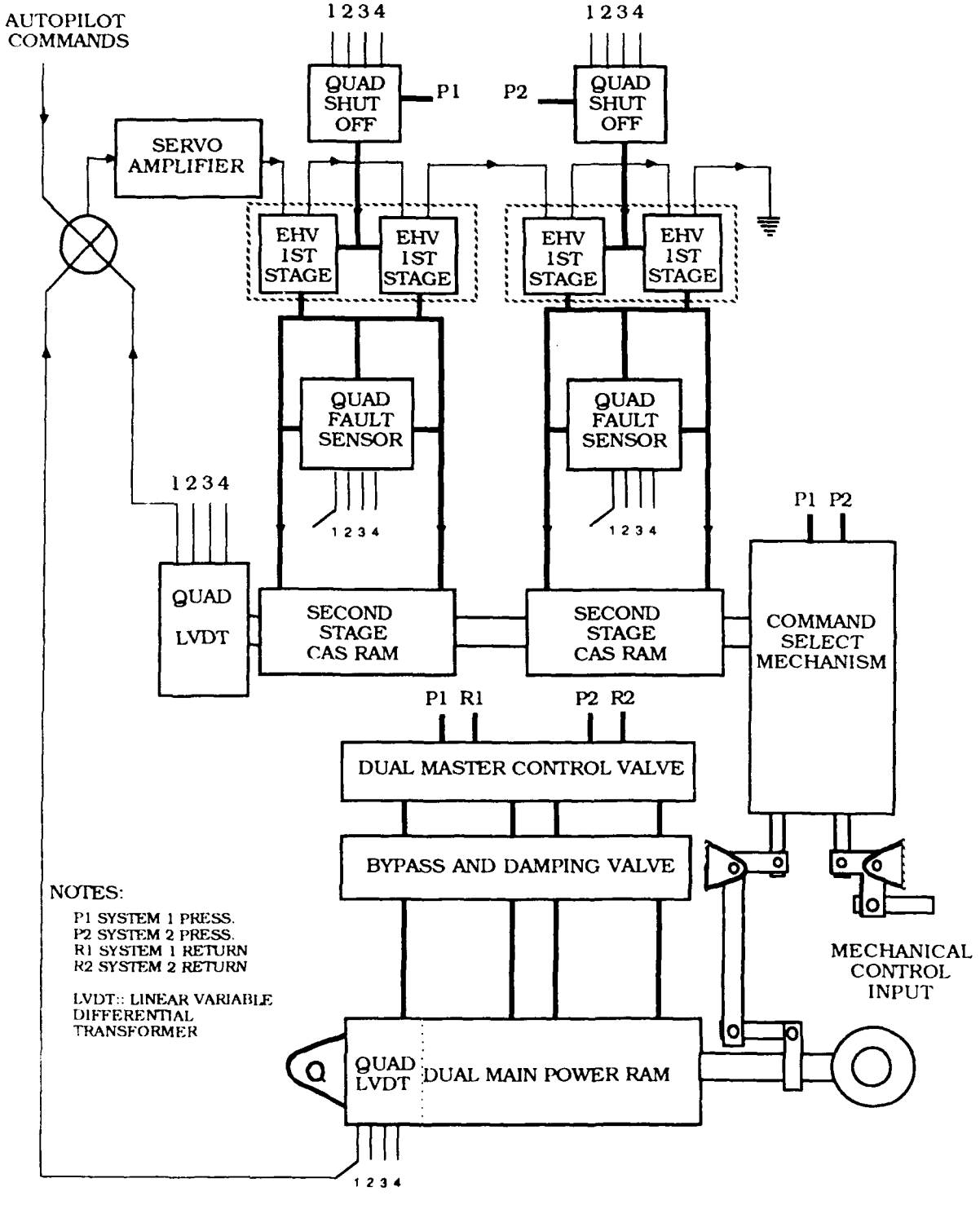


FIGURE 2.1-7. F-18 STABILATOR ACTUATOR

2.1.2.3 V-22 Actuator

The V-22 swashplate actuator (McManus 1985), illustrated in figure 2.1-8, is controlled by a triplex system. Three of these actuators are used on each main rotor. This system has considerable redundancy and failure management. The triplex uses self monitoring to detect failures within a local channel without depending on the other channels. However, the redundancy of the V-22 actuation system is not as great as for the F-16 quadruplex FCC. The triplex is configured as a duplex with failure reversion (a two-fail-operate system). There are two hydraulic supplies.

The actuator main control valve is driven with two unbalanced modulator pistons. Either of the two two-stage servovalves in system one are configured to drive one modulator piston with the third two-stage valve in system two driving the second smaller piston. Since the second piston has one-half the area of the first, this sets up a force fight. Since the drive currents are the same, the smaller piston loses. Each of the three servovalves are commanded separately by the three FCCs. In the normal no-failure condition, system one controls the main valve. Either FCC1 or FCC2 can perform the function. Any failure in FCC1 or FCC2 will be automatically detected by the comparisons between FCC1 and FCC2. Any failure in FCC3 will be unnoticed as long as either FCC1 or FCC2 is operating. Any two-failure situation will leave the system functioning with full performance.

Servovalve failures are detected using the position sensor on the second-stage spool of each servovalve. The current to each servovalve is measured and compared to the current calculated from a digital model in that FCC channel for detection of amplifier and servovalve failures. In the event of failure detection that channel will be shut down.

Each of the FCC channels contains a dual processor. The two FCC processors, A and B, in each channel monitor themselves. Either one can perform fault reversion. Some sensor inputs are compared across the channels to detect failures not covered by in-line monitoring. Single self-monitored sensors are used. These inputs are passed to the redundant processors (A and B) via a nonredundant sensor interface and single input/output (I/O) processor. Self checks by the two processors also check the I/O processor and sensors. Each processor calculates the output commands and expected response. Each processor can control engagement of the servo loops through discrete outputs. Commands from processor A are sent to the actuators by the I/O whose performance is checked by processor A and B using wraparounds. A watchdog timer is used to track each time frame; all actuators associated with an FCC channel will shut down upon failure detection.

The six LVDTs are used to monitor the position of the servovalves, the main control valve, and the power ram. Failures of LVDTs, including the triplex LVDTs on the main control valve and power ram, are detected using a self-test approach because no cross-channel communication is allowed for this FCC architecture. The response to failure of a servo amplifier, control coil, servovalve, or LVDT is the same: operate the shutoff valve or bypass valve to disable hydraulic flow to the servovalve in the failed channel.

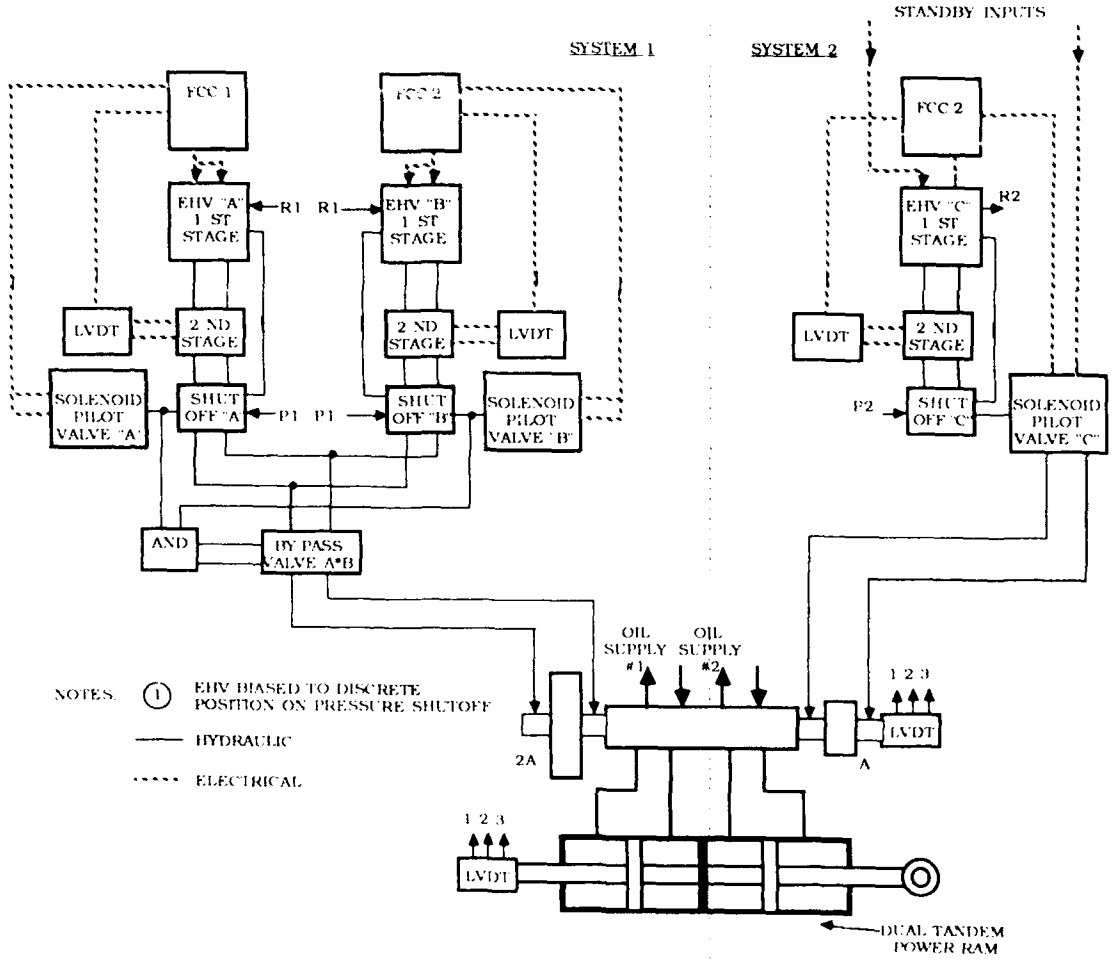


FIGURE 2.1-8. V-22 SWASHPLATE ACTUATOR

2.1.3 Detailed Examples of Advanced Technology and Research

2.1.3.1 Dynamic Controls' F-16 Direct Drive Actuator

The objective of this investigation, reported in Jenney 1989, was to implement and test a direct drive actuator for a current advanced technology fly-by-wire aircraft. The F-16 integrated actuator was chosen for the investigation because it was still in production and a sample flaperon/horizontal tail actuator was available for use.

A direct drive actuator directly modulates the flow of the main piston by hydraulic servovalves. This eliminates the requirement for a separate secondary actuator to position the main control valve.

Direct drive actuators operate the main servovalve using force motors having sufficient force. This technology was flight tested in 1981 on an F-4E at Edwards Air Force Base and has been subsequently refined in a laboratory investigation reported in Jenney 1989. The schematic diagram, shown in figure 2.1-9, illustrates a single-fail-operate direct drive actuator design. For the F-16 application a third channel of electronics was added to accommodate the dual-fail-operate requirement. Each channel was self-monitoring and independent of the others. Six feedback transducers were required to retain complete independence for redundancy purposes. The system designer placed the actuator inside a position feedback loop. This configuration is desirable because power is required only when the actuator is moving. Using the actuator in this manner also relaxes the requirement on linearity of the drive spool motion.

For this direct drive configuration, actuator performance degrades upon failure. Using electrical feedback for the direct drive actuator position also allowed inclusion of the force motors and control valves inside the position loop. This operation allowed simple reconfiguration following failure. Upon any single failure of the three self-monitoring control channels or both hydraulic supplies, transfer was made from the electrical-control mode to the self-centering mode. This transfer was accomplished with solenoid valves as for the normal F-16 actuator. With a multiple coil electrical control for each valve, the force applied is a direct function of the coil currents. When any channel fails, the position gain (and hence the flow gain) automatically changes. The actuator responds to position commands, with degraded performance, even if two of the three electrical channels fail.

Performance of this actuator could be maintained following failures by adding electronic elements. Using feedback of the spool position in an interloop would tend to keep the flow gain constant, with an increase of complexity. Gain changing after failure can be used to keep the valve performance constant following the loss of a channel. However, gain changing requires coupling between channels, which reduces control independence and could possibly introduce common mode failures. In the interest of high reliability, gain changing or interloop position feedback was not used for the prototype design.

The initial prototype design included analog failure monitoring. A second phase of the investigations (Jenney 1989) added microprocessor failure detection and correction. For either of these approaches, the failure monitoring would be checked pre-flight. These monitoring systems were designed so that failures in the monitoring system appear as single channel failures and could not cause hardover outputs.

The force motors were driven using pulse width modulation to reduce the heat sink requirements for the drive amplifiers. Because the coils are driven full-on or full-off, the power dissipated in the amplifiers is smaller than if operated in a linear mode. This minimizes the heat generation in the amplifiers and the size of the heat sinks. A potential disadvantage is the electrical noise generated by the current switching transients.

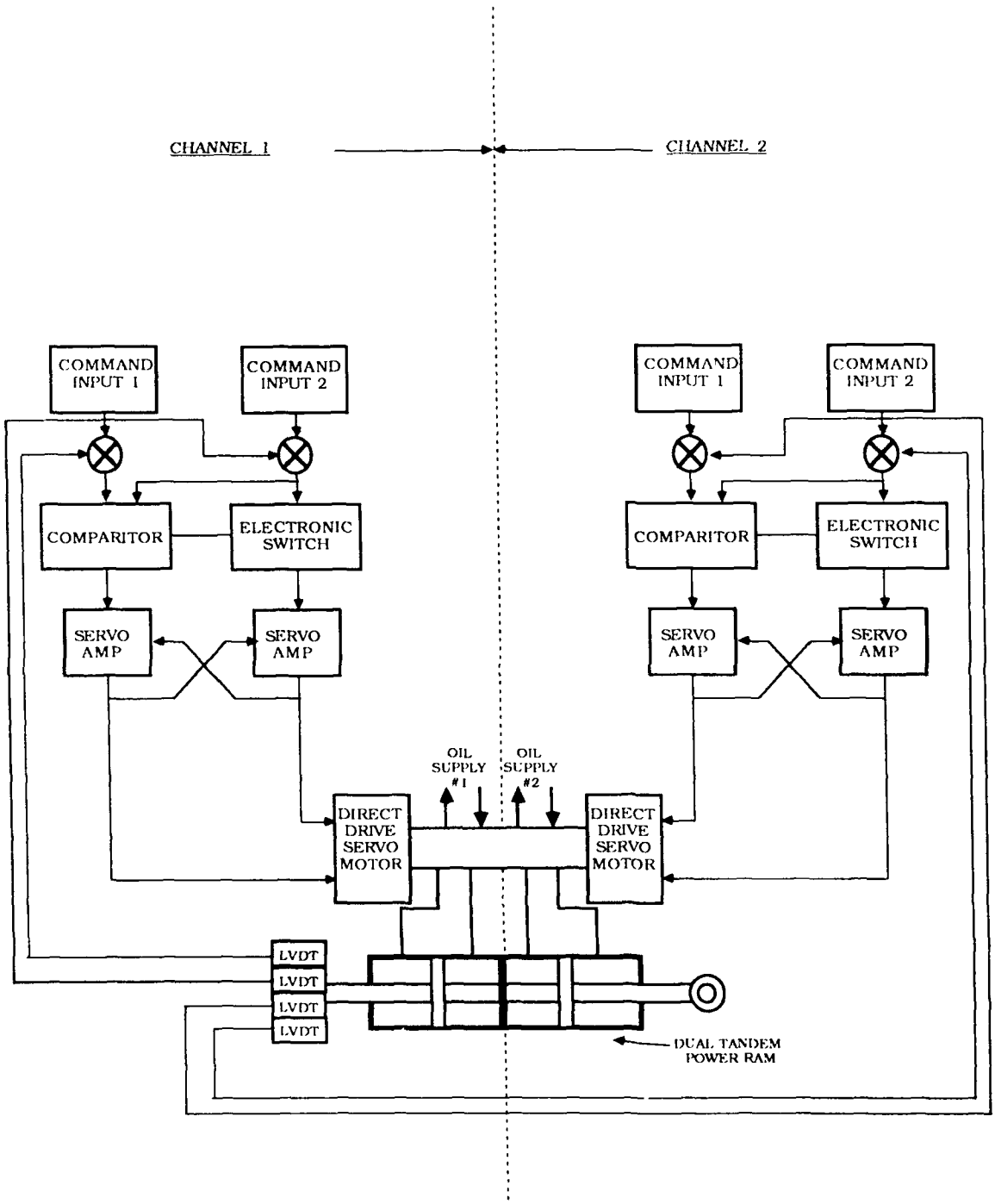


FIGURE 2.1-9. SINGLE FAIL OPERATE DIRECT DRIVE ACTUATOR

Figure 2.1-10 shows a block diagram of the two-fail-operate direct drive actuator designed for the F-16. All three channels operate normally, two for a single failure operation and one for the dual fail operate. Note that the three channels are self monitoring and no electrical mixing of the channels occurs. Since each channel uses two command inputs, the failure logic can detect input failures. These input signals are strapped together if failures are detected and corrected ahead of the control actuator system. Six feedback transducers allow each channel to be self monitoring. When a failure of a feedback transducer, summing amplifier, or command input is detected by the comparator, the inputs to the servo amplifier will be grounded. Two amplifiers are used for each channel. These two amplifiers are cross strapped together so when an amplifier fails, the other amplifier will be driven in the opposite direction, cancelling the output of the failed amplifier. If an amplifier fails resulting in a no-output condition, then the gain of the remaining amplifier will increase, tending to keep the channel force gain constant.

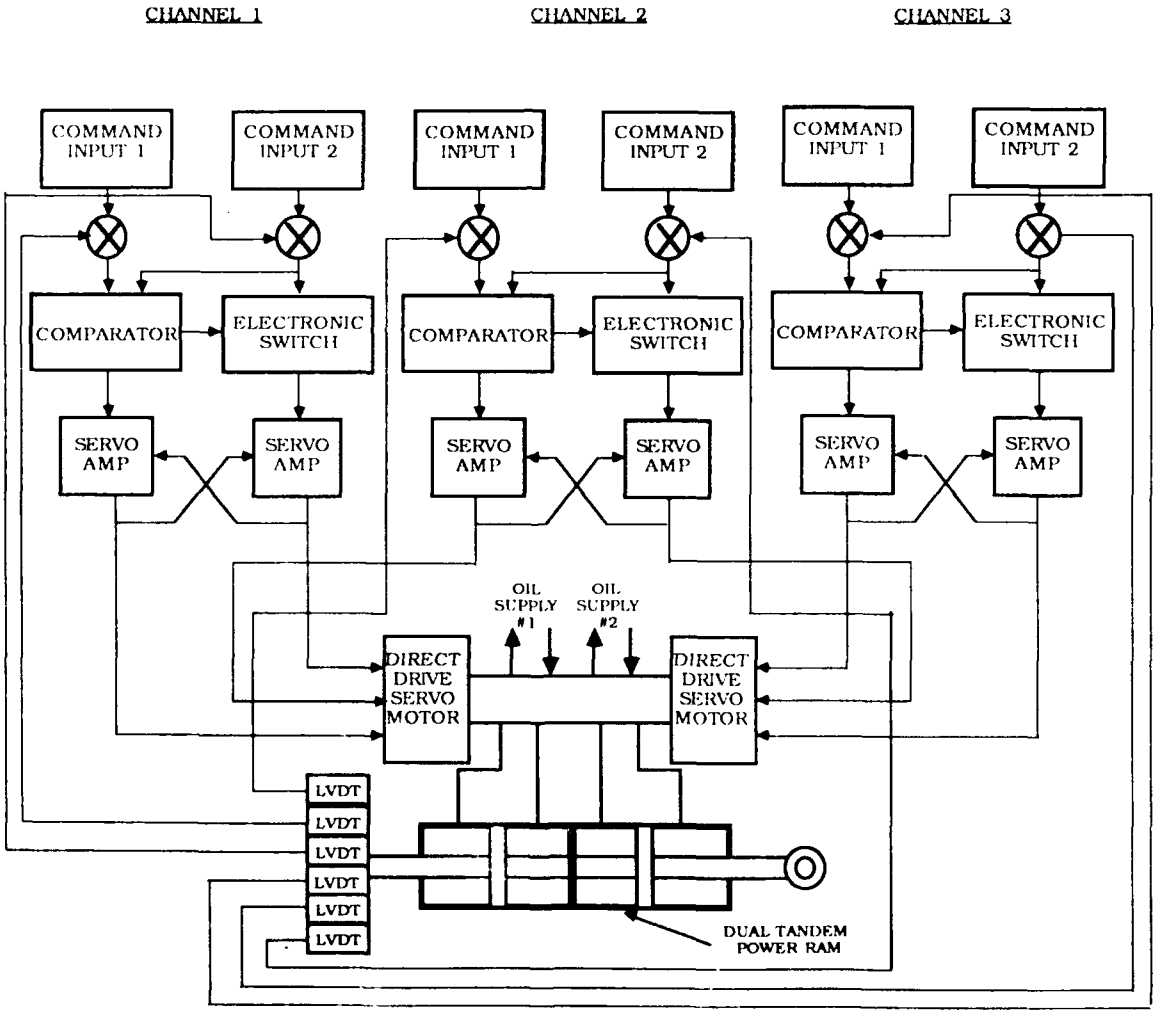


FIGURE 2.1-10. DYNAMIC CONTROLS' F-16 DIRECT DRIVE ACTUATOR

The direct drive control valve concept for laboratory investigations is based on a moving coil design. The coil assembly is suspended by a folded linkage incorporating both a centering spring and flexures for the pivot points. The coil assembly consists of three individual coils, electrically isolated from one another, with four layers of windings in each coil. The coils move within a magnetic circuit energized with a rare earth permanent magnetic material. Stainless steel 416 was used for the pole piece and outside cup. Tascore 21 was used for the permanent magnet material. The force output from the coil was 55 pounds with all coils energized at 1.2 amps. The resistance of each coil was nominally 4 ohms. Figure 2.1-11 shows the structure, coil, and housing. This is a wet coil design flooded with return oil. Use of a wet coil eliminates the need for a dynamic seal and provides cooling for the coil and damping for the coil motion. Since the oil is fed through a filter disk and only circulated locally, the problem of the magnetic structure acting like a magnetic oil filter does not occur.

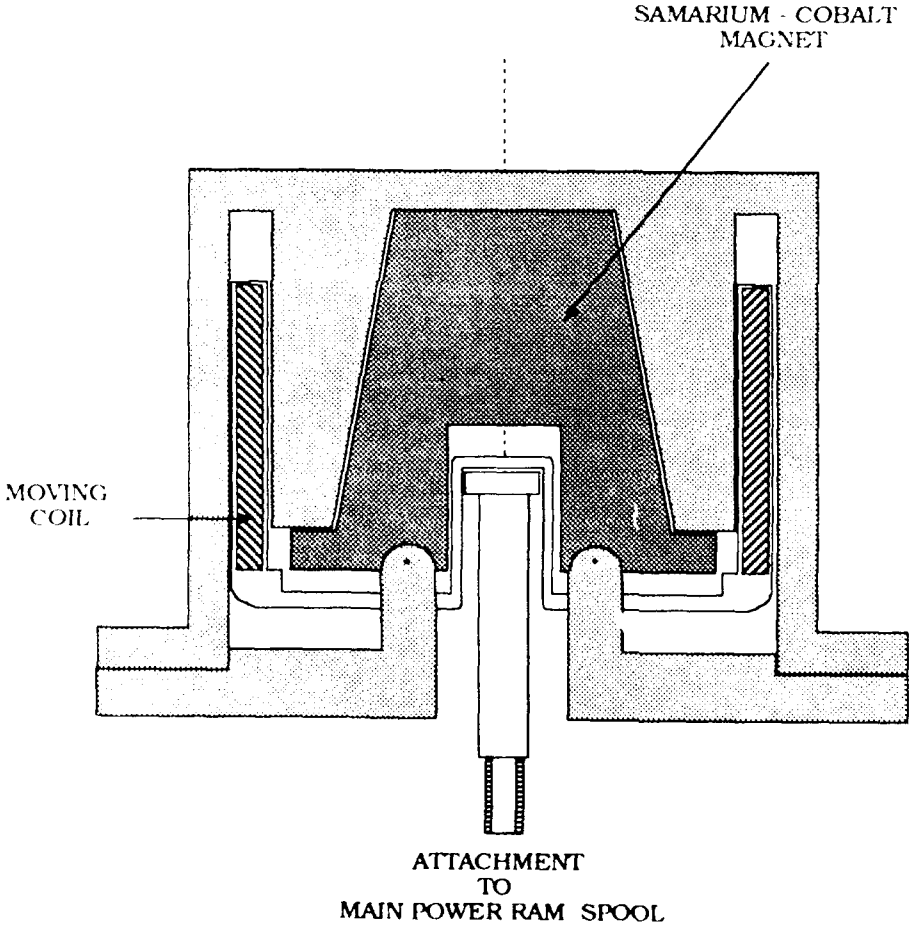


FIGURE 2.1-11. MOVING COIL TORQUE MOTOR

Two force motors were used for the F-16 actuator, one on each end of the main control valve. These motors were connected together in the valve package with a large flow passage so that the pressure could balance between the magnetic structures. The return pressure line of one hydraulic system supplied oil for both force motor coils.

The main control valve that both motors positioned was a three part design. The valve moved in a sleeve which was normally held stationary by a locking mechanism. Electrical feedback normally controlled the position. When mechanical feedback was used in the fail-operate mode, the sleeve was unlocked and driven by the mechanical feedback linkage. In this fail safe mode the driving electronics for the center spool are turned off and the control spool maintained in a center position by springs that center the force motor valves.

2.1.3.2 Bell-4Valve

The Bell-4Valve actuator (Abrams and Donley 1984, and Murphy and Haskins 1982 and 1984), illustrated in figure 2.1-12, uses four two-stage flapper valve servovalves to meter the hydraulic flow directly to the dual tandem power ram.

Any failures of the servovalve coil or drive wire are detected by comparing the drive current to that calculated from a simple analog model. If a failure is detected, then that particular valve is disengaged by operating a solenoid valve. Active servovalve failures are detected by directly comparing positions of the second stage spools of the four servovalves. For hardover failures, the control system causes a flow bypass around the piston of the failed servovalve.

Because the Bell-4Valve is able to accept a hardover servovalve failure, the detection thresholds are set so that only large failures are detected. The advantage of this approach is that false alarms are reduced. If both channels driving a piston fail, a bypass valve is activated for that piston. The FDFM system is implemented in analog logic.

2.1.3.3 Boeing/Moog DISAC

The DISAC prototype actuator (Waffner and Chenoweth 1984, and Chenoweth and Slaugh 1985), illustrated in figure 2.1-13, was developed to be controlled and managed by two microprocessors. It differs from the Bell-4Valve actuator in that only two servovalves are used.

In the primary mode, each microprocessor controls the servovalve and bypass valve for one channel. However, if one microprocessor fails a self-test, that processor drops control and the other processor takes control of both channels. [illegible] exists to keep both microprocessors from attempting to control the same component at the same time. Both processors access data from both channels by means of duplicate sensors. One sensor interfaces to each processor. In addition, both microprocessors are able to operate each servovalve and bypass valve using separate coils in each component.

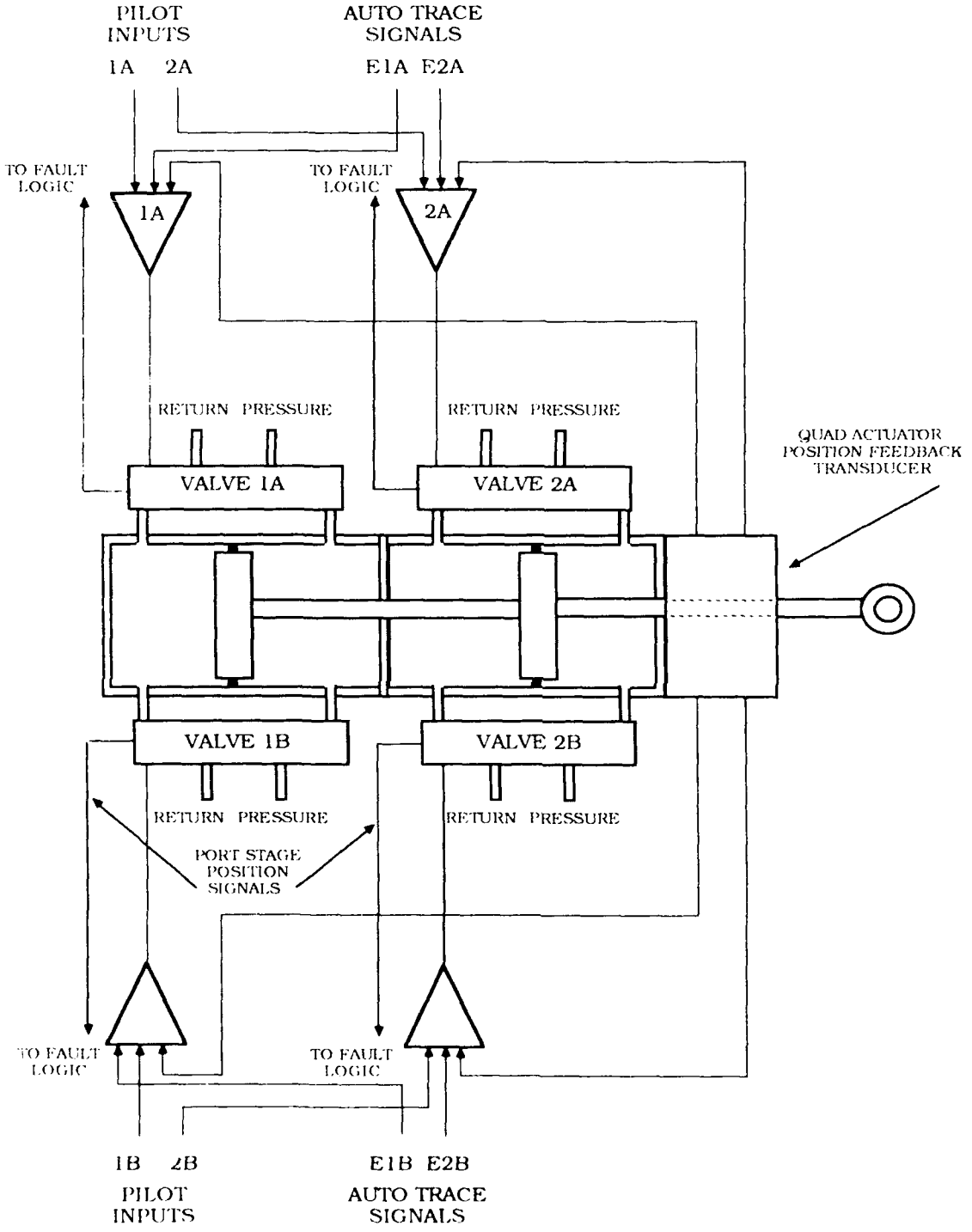


FIGURE 2.1-12 BELL-4VALVE ACTUATOR

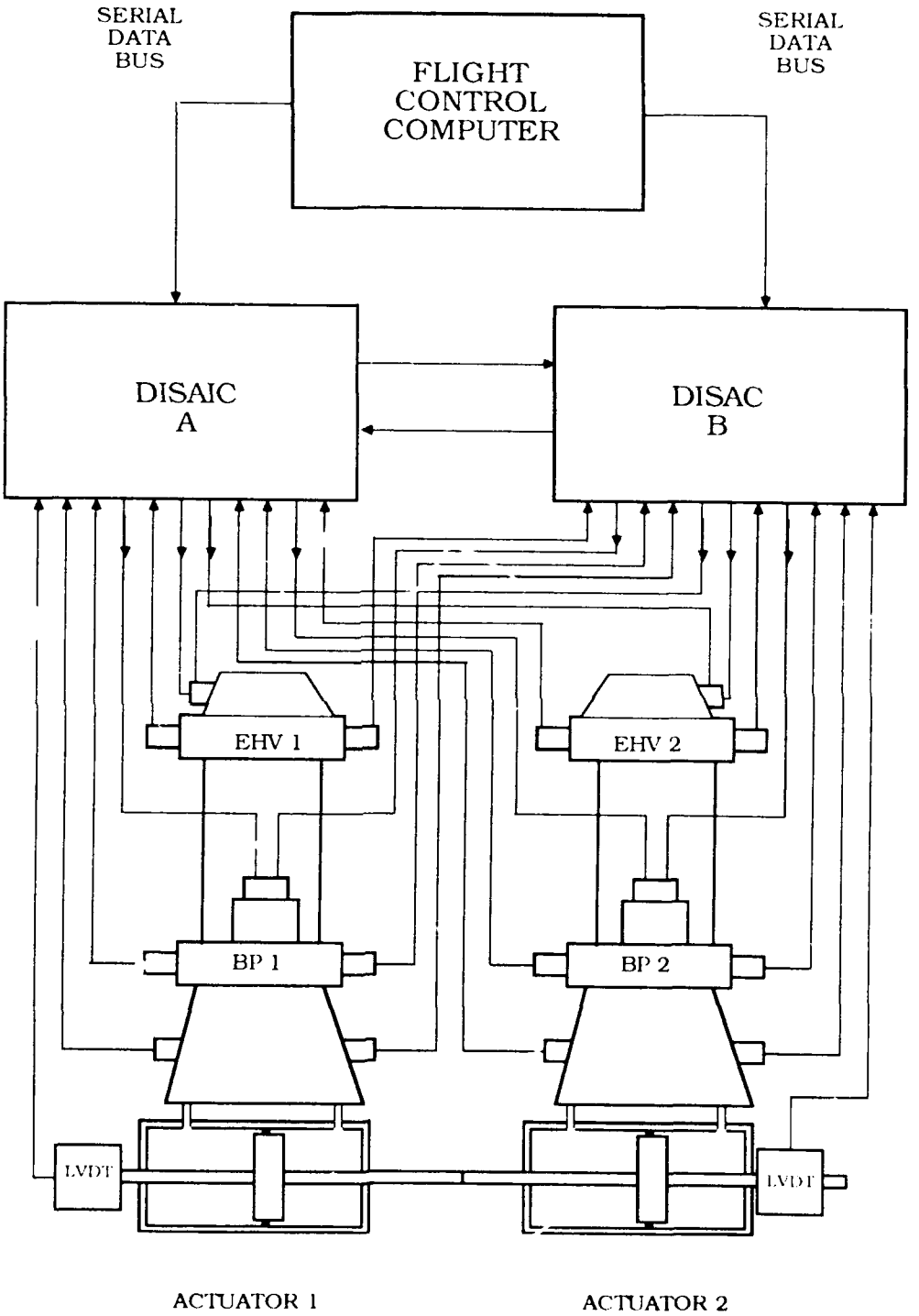


FIGURE 2.1-13. BOEING/MOOG DISAC ACTUATOR

Failure detection on this actuator consists of comparing the measured position of the second stage spools to fast and slow models of the servovalve to detect failures. LVDT failures are detected using self-test. The failure management response could be either to turn off the channel with the failed component or allow the other microprocessor to take over operation of the channel.

One unique feature of this actuator is the use of position switches on the bypass valve to verify its operation during preflight tests. Bypass valves failing open is a latent fault that cannot be observed during normal flight operations. Preflight testing at least verifies bypass valve operation prior to takeoff.

2.1.3.4 Dynamic Controls All Digital Actuator

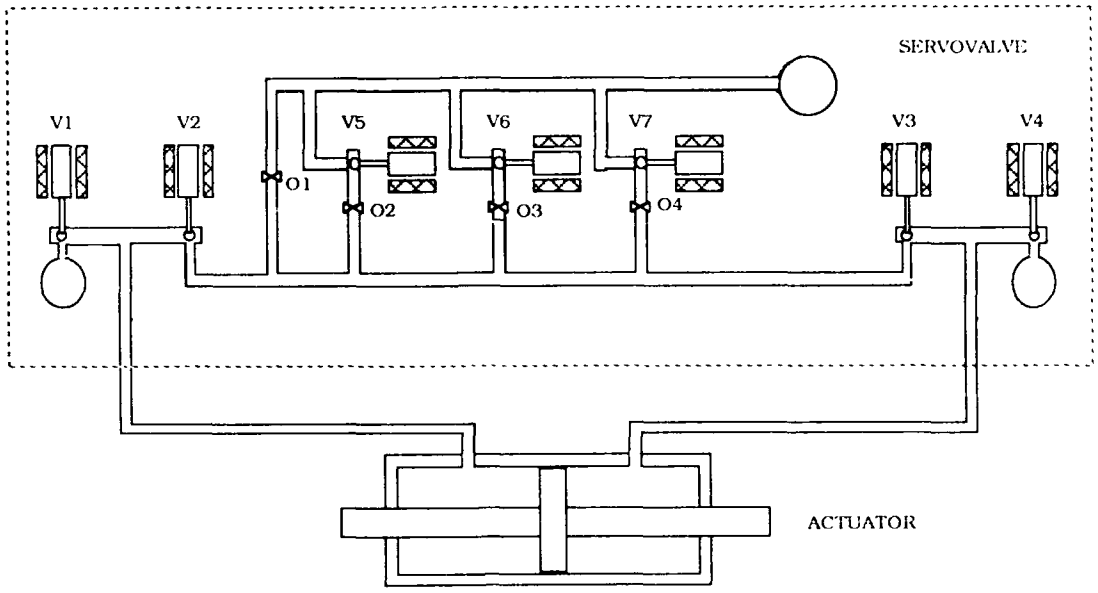
Recent research and development, reported in Jenney 1989, illustrates innovation in implementation of a direct drive valve. This all-digital hydraulic servovalve has digital signals applied directly to poppet valves controlling hydraulic flow to the main piston. In this implementation the poppet valves control flow in binary increments. Each valve is driven on or off by the value of a single bit in a binary word. Thus, the overall flow is proportional to the binary value of a digital command. The flow is controlled by means of a piezoelectric (PZ) driven poppet valve.

The poppet valves used in this configuration had a step response 30 times faster than the electromechanical valves normally used for control of the main piston. The concept has potential for very fast response of the main actuator. The position control loop is closed with a microprocessor and position transducer. Results from a laboratory demonstration achieved 0.5 millisecond operating speed, although the PZ poppet valves failed to meet their speed objective by a factor of two too slow.

A schematic of the operating elements of the digital control valve and actuator are shown in figure 2.1-14. This implementation used four orifices. Four poppet valves are used for directional control. Valves V1 through V4 are connected from a cylinder poppet of the valve to hydraulic pressure return (operating these valves controls the direction of the piston). Opening valves V2 and V4 causes the valve to move to the right (as shown in figure 2.1-14).

A gallery is connected to the combined output flow from the flow modulation poppet valves and orifices. As shown in figure 2.1-14, the four orifices, labeled O1 to O4, are used for flow modulation. Orifice O1 is connected to the hydraulic supply while O2, O3, and O4 are connected to valves V5, V6, and V7. These three valves, in combination with the non-switched orifice O1, provide metered flow to the directional valves.

The rate that the actuator moves is determined by the size of the orifices and the number that are opened by the poppet valves. The sizes of the orifices are scaled so that the flow increases in powers of two (i.e., O3 causes twice the flow of O2, and O2 causes twice the flow of O1). Thus binary coding of the digital word connected to the three orifices allows eight steps from off to full



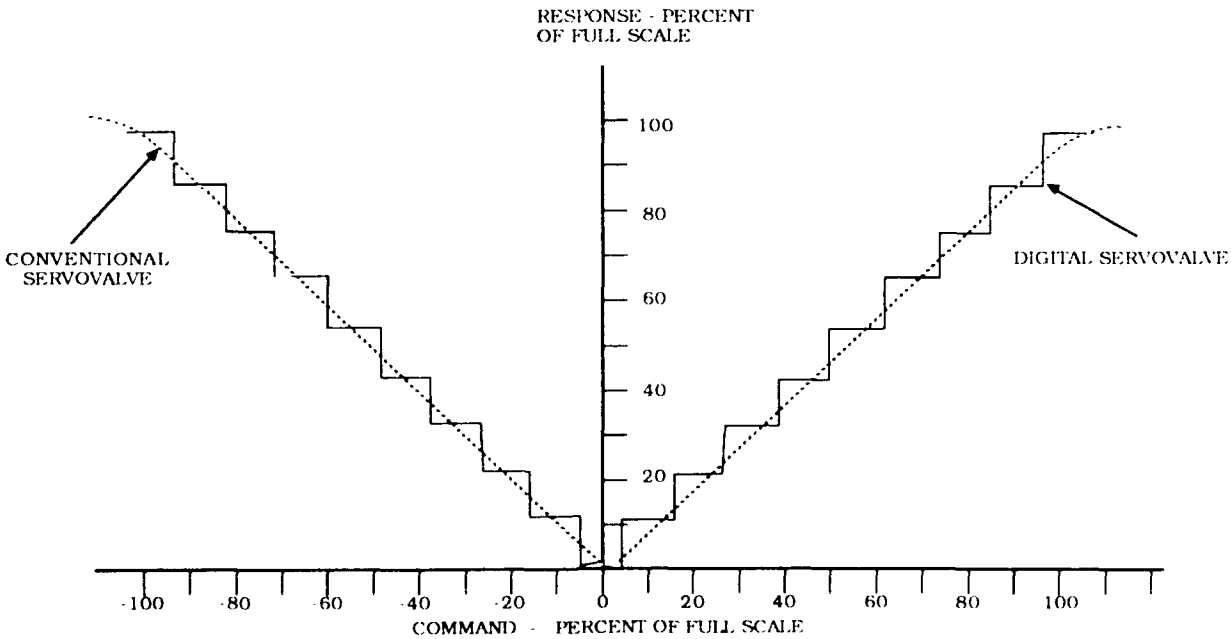


FIGURE 2.1-14. DYNAMIC CONTROLS' ALL DIGITAL SERVOVALVE ACTUATOR

on. With eight steps, flow linearity is maintained within ±6.25 percent of a linear output. A production article could be built having additional steps and bits in the digital control word. Eight bits would provide 256 flow step combinations (better than 0.5 percent resolution); this is in excess of a more normal requirement of two to three percent linearity for actuator position control. Addition of one more poppet valve would provide a flow linearity of ±3.1 percent, comparable to most analog servovalves.

From a reliability and redundancy viewpoint, this digital valve appears very promising. The valve motion is determined by fixed elements, the orifices. Control is afforded by digital devices that only operate in an off or on state. Further, if any one of the orifices or associated valves fails to operate, then the actuator functions at lower rates of motion. Degraded performance is usually more desirable than "hardovers" or "no response". The critical control element for establishing hydraulic flow is the poppet valve. Valve response times vary widely for electrical control of mechanical motion. This prototype demonstrates that great improvements in valve speed are possible.

A conventional magnetic solenoid valve is limited in response time by electrical and mechanical constraints. A survey of electromagnetic poppet valves, conducted as part of the research reported in Jenney 1989, provides some interesting results. The on-off response times for commercial poppet valves varied from 2.5 to 14 milliseconds and cost estimates were between $1,500 and $4,000 each. All of these valves failed to meet the required response of the control actuator. One reason for this can be found in the dynamics of the mechanical valve components. A typical solenoid valve consists of a spring, armature, and coil, as shown in figure 2.1-15. The motion of these elements requires the magnetic circuit to build up forces to accelerate the armature and open the valve in a

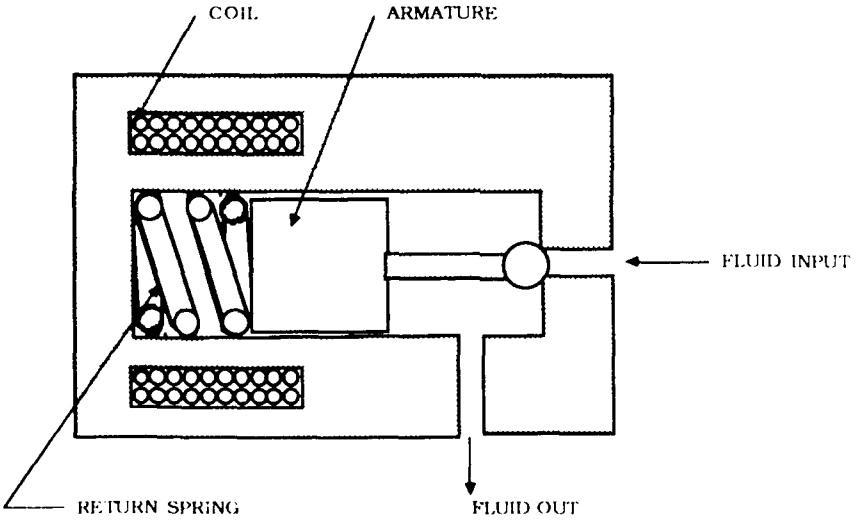


FIGURE 2.1-15. SOLENOID POPPET VALVE

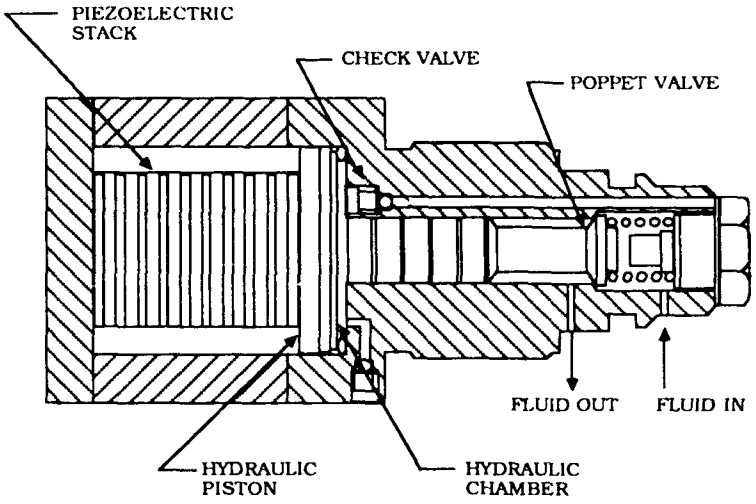
short time. The armature and coil must be sufficiently large to produce forces great enough to overcome the spring restraining force and inertia. For example, an armature weighing 0.022 pounds and travelling 0.02 inches in 0.001 seconds requires a driving force of 2.27 pounds. The spring preload to close a 0.30 inch diameter valve against a 3,000 psi pressure is 2.12 pounds. Thus the magnetic forces must exceed twice the spring force. An additional constraint is in the magnetic coil. A large number of turns on the coil will build up forces for the same current. However, the coil inductance increases with the square of the turns, requiring more time to build up the drive current. The RL time constant ultimately limits the current build up and speed of operation of the poppet valve.

A moving permanent magnet armature has also been tried for this application on other research projects and was found to have the same RL time constant limitation on rapid build up of control currents.

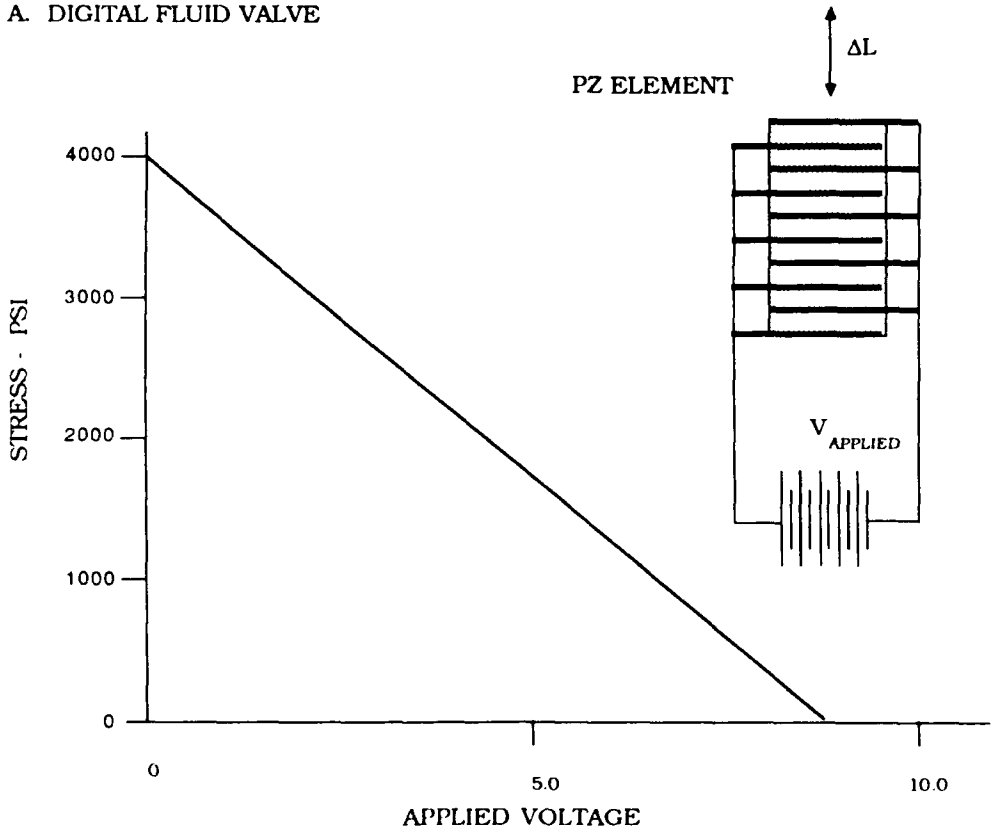
A review of PZ elements for application to poppet valves indicated feasibility of constructing a valve having a 1 millisecond operation time. A PZ material will distort when an electric field is applied to it. Conversely, an electric field will be generated when the material is distorted by a force. Figure 2.1-16 illustrates a poppet valve constructed from a multilayer PZ element.

The quantitative values for the relationship between the mechanical deformation and electric fields depends upon the material type, crystal axis orientation in the field, and physical configuration of the device. The output force capability of PZ materials is very high (on the order of 3,000 psi), although the strain is very small (on the order of 0.1 percent of the thickness). The electrical characteristic of PZ material is the same as a capacitor. Because of the high force to mass ratio for these materials, the strain can be achieved in a few microseconds. Typically, fields of 25,000 volts/inch are required to produce a strain of 0.001 inch. To reduce the driving voltage, PZ drivers are produced in thin layers, bonded together, and connected electrically in parallel. For a field of 25,000 volts/inch, a layer thickness of 0.020 inch only requires an applied voltage of 500 volts. When the material is preloaded, the strain will be determined by the modulus of elasticity and the stored energy. Since Young's modulus for a typical PZ material is about 25 percent that of steel, quite large preloads are no problem. The charge/discharge characteristic displays a large hysteresis: on the order of 20 to 30 percent for typical materials. This hysteresis limits the application of PZs to digital rather than analog control devices.

The laboratory investigation reported in Jenney 1989 achieved an operating digital controlled actuator having PZ poppet valves with 500 microsecond operation times in a hydraulic supply pressure of 3,000 psi. The microcomputer was an off-the-shelf Z80 having a real-time program processing time of 600 microseconds. This is very impressive compared to the fastest hydraulic poppet valve that has an operate time of greater than 5 milliseconds.



A. DIGITAL FLUID VALVE



B. PERFORMANCE OF PZ STACK

FIGURE 2.1-16. PIEZOELECTRIC SERVOVALVE

Since these devices operate very quickly and are easily driven from 270 volt dc with digital controllers, future aircraft should have numerous applications of PZ devices. The high speed and high force capability of the PZ actuators should be applicable to direct-drive force motors for controlling spool type servovalves.

2.1.3.5 Dynamic Controls' F-15 Flutter Suppression Actuator

There is a trend toward increasing the operating pressures of military aircraft hydraulic systems. Higher system pressure allows the use of smaller, lighter weight actuators with smaller drive areas. This conflicts with the normal design technique of using a larger drive piston area to increase actuator stiffness to avoid flutter of the control surfaces. An increased stiffness is usually accomplished by increasing the ram piston area (i.e., by using an actuator having a higher force output and flow demand than required for maneuvers). An Air Force project (Jenney 1989) developed a prototype actuator having electronic control signals for changing the impedance at the flutter frequency and thus effectively damping control surface flutter.

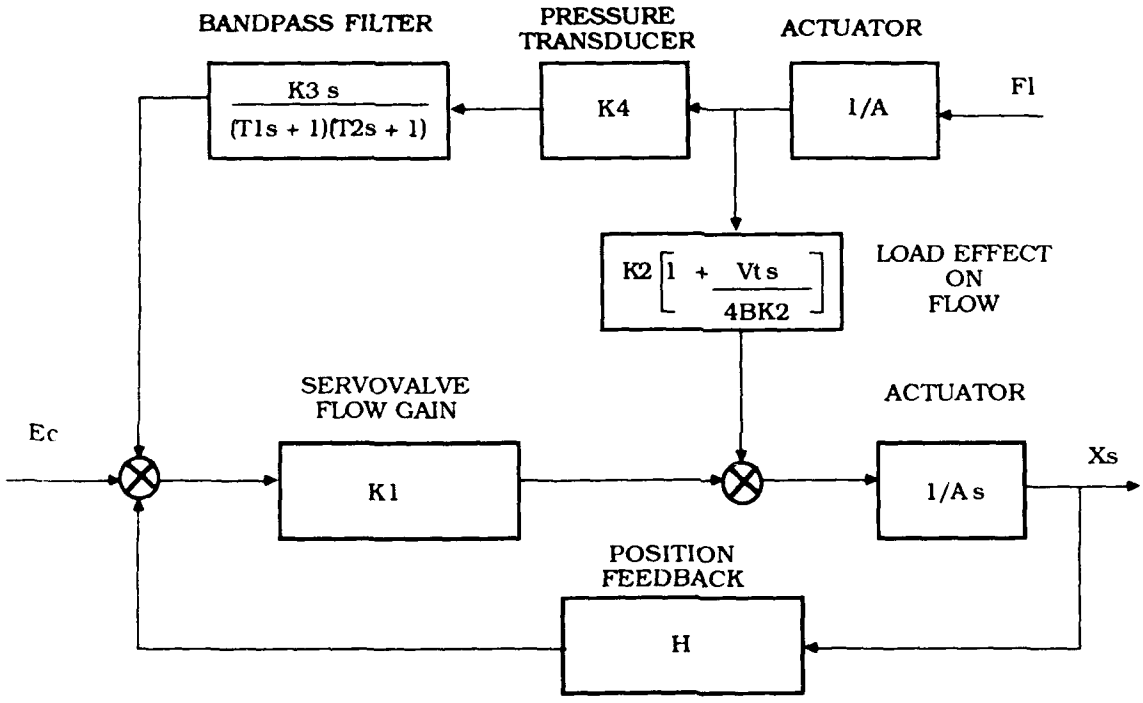
The control surface flutter phenomena are aeroelastic, self-excited vibrations in which airstream energy is absorbed by the control surface. The resulting surface motion consists of bending and rotation, each having a unique frequency and amplitude for a given airstream velocity. If either mode of oscillation is suppressed, classical flutter will not occur. The control surface rotational motion is primarily determined by the spring constant of the actuator and the rotational moment of inertia of the surface. Normally the rotational motion of a control surface is lightly damped. The normal technique for reducing flutter is to raise the resonant frequency by increasing the stiffness of the actuator.

An alternate technique for damping control surface flutter is to change the damping of the surface oscillation to be greater than critical damping, then the motion will be suppressed. This damping could be accomplished by load pressure feedback, thus changing the impedance of the actuator at the flutter resonant frequency. The technique of impedance modification has a good performance payoff by allowing actuators to be sized for maneuver requirements. As an example, the Tornado aircraft uses hydromechanical load-pressure feedback on the horizontal tail actuator for damping flutter.

Figure 2.1-17 shows the block diagram of the flutter suppression actuator. With no-load force, F_1, the control loop is a conventional electrohydraulic position control system. The effect of the load force is to reduce the flow to the integrating actuator from the servovalve. This reduction is both a steady state and dynamic effect. The steady state effect is a reduction of the flow to move the actuator because of leakage. This load pressure sensitivity is represented by the term K_2. A secondary effect of the load force feedback reduces the differential pressure across the servovalve (which causes a reduction of the valve output flow). This effect was ignored to linearize the model. This analysis assumption does not introduce large errors if the differential load pressures are much less than the differential supply pressure across the servovalve (within the constraint $P_1<(2/3)P_s$). The dynamic effect of the load pressure is due to the compliance of the fluid in the actuator and depends on the rate of change of load pressure. The effect is to reduce the dynamic flow

gain of the servovalve (particularly when the flow is a function of actuator motion).

The load pressure feedback, shown in figure 2.1-17, results in a positive feedback loop. The bandpass filter in series with the differential pressure signal across the actuator drive area provides a lead over a double lag filter function. This filter effectively blocks the low frequency effect of feeding back load pressure. Thus, the static stiffness and steady state position accuracy of the main control loop are not affected.



Notes:

Ec = Command input (volts)

T1, T2 = Bandpass filter time constants

K3 = Bandpass filter gain

K4 = Differential pressure

K1, K2 = Servovalve flow gain

Vt = Actuator fluid volume

B = Bulk modulus of fluid

A = Actuator drive area

s = Laplace operator

FIGURE 2.1-17. DYNAMIC CONTROLS FLUTTER DAMPING ACTUATOR

2.2 All-Electric Actuators

Electromechanical actuator (EMA) designs have evolved into a proven technology with a proven record. The material discussed in this section draws upon the discussion of Hair 1985 (refer to AiResearch 1976 for early work). The operation of these actuators is based on the use of high density rare-earth magnetic material and brushless commutated motors, coupled to a mechanical transmission. The motor converts an electric current into torque, which is transferred through the gear box into mechanical motion. The output gearing is connected to the flight control surface with a proper torque speed characteristic for control surface motion.

There are three types of actuator configurations: linear, rotary, and rotary hingeline. For a dual-drive motor configuration, the motor output can be velocity summed or torque summed. The linear actuator uses a ball screw mechanism to transmit mechanical force. Rotary actuators use a planetary gear arrangement for surface motion. A rotary hingeline actuator integrates the actuator into the hingeline and thus into an aerodynamic surface envelope, eliminating the need for any fairing.

2.2.1 Actuation Concepts

An actuator must be designed for a specific level of performance. This performance obviously depends upon the control surface application for a particular aircraft. Actuators for the trailing edge will have different performance than for a rudder. Efficient structural load transfer is also a benefit of the hingeline configuration, as illustrated in figure 2.2-1. Each actuator is unique in terms of performance: stall load, no-load speed, operating bandwidth, stiffness, and rotational envelope.

In terms of reliability, one of the first questions that needs answering is fault tolerance. Whether the actuator performance degrades gracefully with multiple faults (i.e., fail-op/fail-safe, fail-op/fail-op/fail-safe, etc.). These performance parameters depend upon the surface application and design limits. Once requirements are established, an electric actuator can be designed to a desired level of fault tolerance. Since faults in components occur more frequently than can be allowed for most applications, the level of fault tolerance determines the ultimate level of reliability for any given actuator. For a typical case, a triplex FCC system (with skewed sensors and reversion mode using analytical redundancy estimators for critical feedbacks) results in a two fault tolerant system. Obviously, redundancy must be applied to the actuation system.

Design to a level of performance is not enough. Performance, reliability, maintainability, cost, size, weight, and power parameters must be met for each application. All these requirements must be taken into the design process and evaluated for each change.

Electromechanical actuation affords considerable flexibility in redundancy designs. Multiple channels can be accommodated in an integrated package. The single motor design, shown in figure 2.2-2, provides actuation where redundant surfaces can be used for fault tolerance. The motor is mounted on one end and,

in the event of motor failure, surface motion is restrained by the brake mounted on the other end. Dual channel actuators, having a single output shaft driven by two motors, can tolerate single failures. A hingeline actuator tends to favor a dual motor for each channel, such as that shown in figure 2.2-3, due to the natural integration of the motors and gearbox. The motor/gearbox outputs may be summed as torques or speed. In the case of a torque summed actuator, both motors are connected along a common shaft. In the event of a failure, the failed motor is decoupled through a clutch and the remaining motor delivers one-half of the original torque. Similarly, a velocity summed dual-channel actuator allows operation at one-half speed, the failed motor being isolated with a brake.

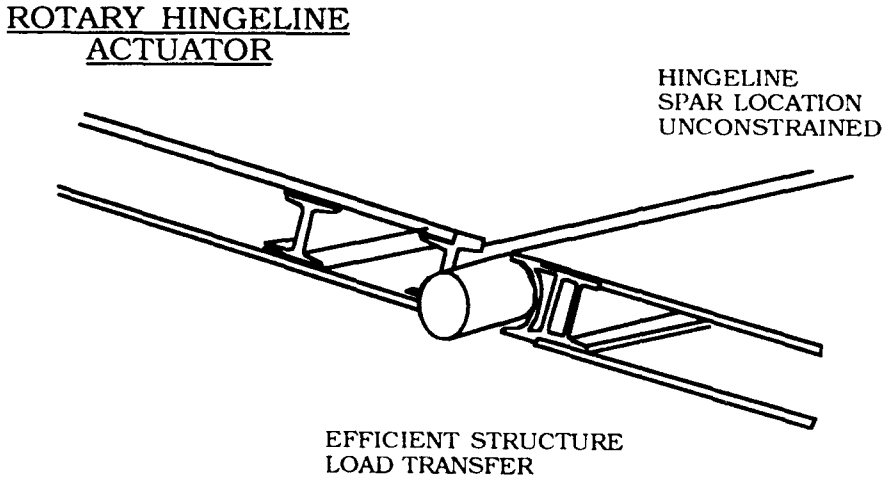


FIGURE 2.2-1. ELECTRIC ACTUATOR CONCEPTS

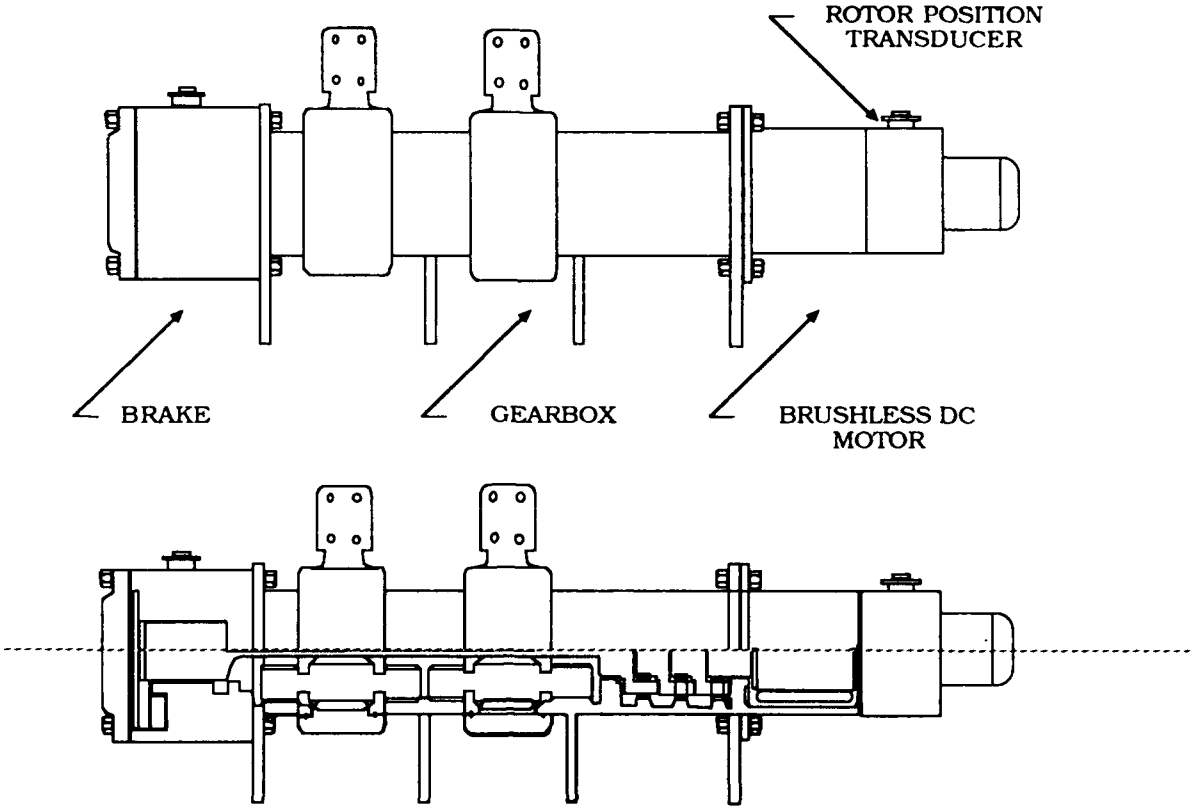


FIGURE 2.2-2. SINGLE MOTOR HINGELINE ACTUATOR

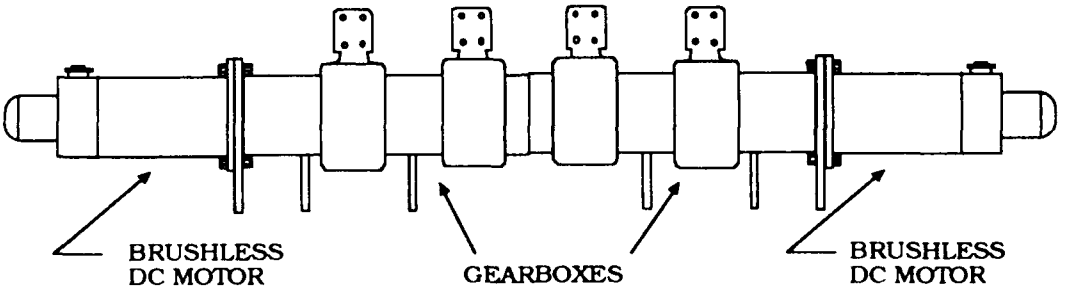


FIGURE 2.2-3. DUAL MOTOR HINGELINE ACTUATOR

Actuation performance under single faults is assured by sizing the actuator for 100 percent performance with loss of a single motor. This requires a 200 percent capability for a dual channel under no-fault conditions. Multiple channel configurations have lower size penalty. A four channel actuator, having four motors driving a common shaft, would be sized for three motors to provide the required torque. This would only require one-third increase in motor capability. If a single motor fails, then this quad configuration operates with full output.

2.2.2 Active Isolation

The ability to actively isolate motor and gearbox failures with brakes and clutches is a key advantage for EMAs over hydraulic systems. Although various computer logic and hydro-logic schemes can isolate failures in electrohydraulic actuators, jamming one piston of a dual actuator results in the complete failure of the actuator. The isolation, by brakes or clutches, of a failed motor or gearbox allows the EMA to continue with any single failure.

Thus active isolation results in tolerance for any single failure in a motor/channel. Typical of this case would be loss of power to one motor of a dual channel. A fault detection and isolation system would determine which motor was without power and, for a velocity summed actuator, engage the brake to isolate and contain the failure. If the brake were not engaged, then the motor would back drive the failed motor through the gearbox. A jammed motor could be isolated without the brake. A jammed motor is the critical failure for the torque summed actuator. When one motor jams, the clutch will be activated to release the jammed motor. This requires a signal from the fault detection system. However, the motor remains engaged in the event of a failure in the clutch electrical circuit. A subsequent second motor jam could not be acted upon with a failed clutch circuit.

2.2.3 Gearbox Technology

The gearbox is critical for operation of these EMAs. All the designs use some form of gears for mechanical transmission. Rotary actuator designs employ planetary gears in simple and compound arrangements to transmit the appropriate torque and speed. Linear actuators use ball and tube gears having similar criticality and constraints.

Planetary gearboxes have large gear reductions (on the order of 500 and 1,000 to one) and can be constructed in small volumes. The gearbox is the place where all loads and signals come together to form the final force output to the surface. This makes the gearbox the critical link. Without redundancy, an EMA relies on several gears to move the surface. This gear train should have redundancy; the cost of this redundancy is significant in terms of size, weight, and complexity.

2.2.4 Reliability

An Air Force sponsored study performed by Grumman (Hair 1985) developed some insight into the relative reliability of EMAS and electrohydraulic systems. For the study two actuators were compared for controlling an F-14's rudder surfaces. A prototype EMA was designed and tested as part of the effort so the results are

a valid point design. The selected electrohydraulic actuator was for the F-14 rudder actuation system. A failure modes and effects criticality analysis (FMECA) was conducted according to defined criticality classes I through IV (results are in table 2.2-1 [Hair 1985]). An initial observation is the increased number of failure modes for the EMA compared with the electrohydraulic unit. This increase is largely due to the increased number of parts, particularly in the gearbox. There is a need to reduce the number of parts in these actuators. Current research is on simpler actuators, development of direct drive valves, and reconfigurable flight control systems relying on simplex actuators.

TABLE 2.2-1. CANDIDATE ACTUATOR FAILURE MODES COMPARISON

ACTUATOR TYPE/ CRITICALITY CLASS	F-14 ELECTROHYDRAULIC	F-14 ELECTROMECHANICAL
CLASS I	3	11
CLASS II	10	12
CLASS III	3	0
CLASS IV	114	266
TOTAL MODES	130	289

The electrohydraulic actuator studied for the F-14 rudder system only exhibited three failure modes which resulted in Class I failures. These failures could result in the immediate forced landing of the aircraft and probable damage to the aircraft. These two modes are piston rod/rod end mounting structure failures and failures of the valve housing. Consequently, the probability of a failure is very low.

EMAs, on the other hand, exhibited 11 catastrophic failures. Two of these were structural mounting failures while the remaining ones were due to failures of nonredundant gears within the gearbox. Nonredundant gears are the major area for potential failures. The root of this problem is that the gears can fail in a number of ways, including wearing and shearing gear teeth, fracturing, and jamming. There are also multiple ways of failing the actuator controller. All of these failures combine to make the probability of failure larger than for electrohydraulic actuators. Since the structural failure modes have low probability of occurrence, the gearbox modes constitute the major contribution to increased failure probability of this EMA.

2.3 Actuation Reliability and Redundancy Technology

Fault diagnosis is the process of determining if a failure has occurred and, if so, what component or subsystem has failed. This information is sent to the

failure management system which determines an appropriate response to the failure. The tasks making up fault diagnosis and failure management are frequently referred to as failure detection, fault isolation, and system recovery and reconfiguration. The information flow for these tasks is illustrated in figure 2.3-1. In this section the contents of these tasks are discussed in general terms drawing from Baker 1988. This material provided a basis for further understanding of fault diagnosis and failure management.

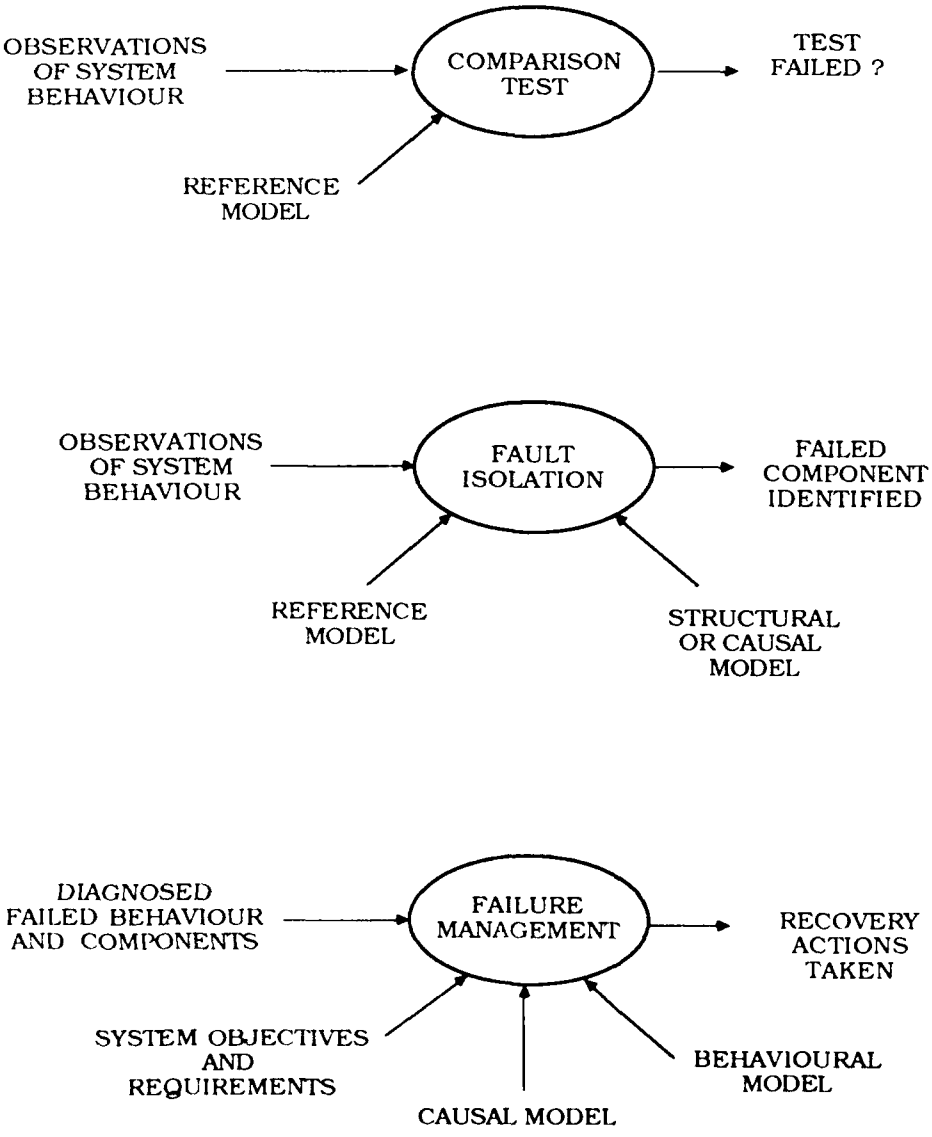


FIGURE 2.3-1. FAULT DIAGNOSIS AND FAILURE MANAGEMENT

2.3.1 Failure Detection

Failure detection is the operation of distinguishing between normal and abnormal behavior of a system. The detection process consists of a continuous cycle of monitoring, information processing, and comparison testing.

In general, a failure is detected by monitoring the behavior of a component, subsystem, or system of interest; converting the raw data into a useful form by filtering or numerical processing; and comparing the resulting measurement with the predicted normal values. A failure is detected when the comparison test inputs differ. The outcome of this comparison test is usually a binary value "ok" or "failed". This approach neglects simultaneous failures in the two channels and assumes the comparator did not fail. Common mode failures must be detected in other ways. For example, common mode failures can be detected by monitoring individual channel inputs and outputs and by monitoring the comparator in a calibration test.

Performance of the failure detection system depends upon the sensor measurements, reference models used to predict the measured values (or system state values derived from the sensors), and the comparison test. The sensors are an important part of the detection process because they are an additional source of failure that must also be managed.

Reference models can take on several forms. Use of hardware reference models is referred to as direct redundancy. Use of software models is referred to as analytical redundancy. In the case of software reference models, the most common models are analytical functional relationships. Using these models and information from the system other than direct measurements, a reference for the measured values is synthesized. Accuracy for these reference models can vary from approximate (to detect hardover failures) to highly accurate (to detect soft failures in performance optimization systems).

Either the normal behavior can be modeled or the failed behavior can be modeled. When modeling normal behavior, the reference model can be implemented in hardware or software. For hardware reference models, a duplicate set of hardware can be utilized to develop the normal behavior.

When modeling the failed behavior of a system, modeling all possible component failure modes is very challenging because the failures occur in many different ways and at different times. As a result the failure models are usually approximate.

One failure modeling method in use monitors each critical component or failure mode. The disadvantage of this scheme is that the failure modes must be defined in advance, before a system is fully developed. Nevertheless, there are acceptable systems that detect faults utilizing models of failed behavior.

Two failure models that are commonly used for fault detection are range checking and trend checking. Range checking declares a failure whenever an output exceeds a conservative estimate of the operating range for that variable. Trend checking declares failure when a variable has an abrupt change that is not physically possible.

Threshold detection is the most common fault detection method. For this method a threshold is set on the difference between the model and measured behavior. Errors occur from sensor noise so thresholds must be set high enough to account for these errors. Detection thresholds do not have to be constant. Some systems use scheduled thresholds.

Missed failures and false alarms plague all these fault detection systems. False alarms are of two kinds: removal of good equipment by faulty detection, and failure to remove faulty equipment.

System reliability depends upon the fault coverage of the detection scheme. Careful testing is necessary to verify that all possible faults can be detected by whatever detection method is used.

2.3.2 Fault Isolation

The process of isolating faults after a failure has occurred is called fault isolation. Faults may be isolated at a component, card, or channel. The results of this process are correct isolation, incorrect isolation, or no decision.

Approaches are local isolation, arbitration, and generate and test. Local isolation identifies failed components immediately upon test failure. Its implementation relies upon sensors monitoring specific components.

Arbitration uses comparisons of only two sources of information. Other information channels, such as from multiple sources, are tested two at a time and grouped by pairs. The failed components are then identified by using the test results from the pairs to isolate the failed component. The data for arbitration may be either direct data from redundant hardware or indirect data calculated from analytical models

The arbitration method is most efficient when used with sensors, as the outputs can be compared two at a time and isolated directly. When isolating other components or subsystems using arbitration, the sensed information used to isolate a failed component must be separately validated. For example, if individual sensors were used to isolate three redundant components, at least two sensors would be required on each component. With only one sensor per component, a sensor failure cannot be isolated from a component failure. A sensor failure on one of the three components can be isolated if the component being measured could be assumed not to fail at the same time, and if one of the sensors agrees with sensors on the other components.

Most other approaches to fault isolation include generation of a hypothesis of which component has failed and then testing the validity of the hypothesis. This requires identifying candidate failed components and testing to verify if the component failed. If the candidate has failed, isolate it; otherwise repeat the prior steps with another candidate.

Hypothesis testing requires a simulation of candidate component failures and generation of data for comparison with measurements on the system. This process is very challenging to implement for a large system.

There are several benefits and disadvantages of the generate and test method. This is a very powerful method, having fewer sensors and processors than other methods. Indirect methods have some ability to adapt to changes. However, this method has several weaknesses. There is uncertainty in the convergence of the process, the solution times are variable, and the candidate selection process is arbitrary. Computation requirements are greater for generate and test than for other methods of fault isolation.

2.3.3 Failure Management

Failure management logic evaluates failures and determines the proper response. Part of this logic must determine, in a very short time, where in the process to return control. Control should be returned at a point where a certain level of performance can be maintained.

The logic process and algorithms for fault management depend upon the process and the equipment still functioning. Few systems take fault isolation information and execute a predetermined sequence of actions.

Recovery usually means to shut down the processor and restart, bringing the system back to a known condition. Two processes are used: the first is reconfiguration to eliminate failed components, and the second is to recover control. Logic is also needed to restore control in a condition where the reconfigured system will operate properly.

The recovery process normally includes other actions taken to correct or minimize failure effects in addition to reconfiguration. These actions may include the following:

- Shutting down operation in a manner to provide a safe condition. Actuators may be locked holding control surfaces in a neutral position. Alternatively the actuators may allow control surfaces to float in the air stream.

- Opposing the effects of failure while the system is reconfiguring. For actuators, this may be a force fight with opposing valves or solenoid coils.

- Returning the system to a condition where control can be resumed. For actuators, this may happen automatically during reconfiguration.

Recovery is generally not needed for actuators as they reconfigure quickly; failures only cause small transients. The discussion in section 2.1 covers the reconfiguration methods for several actuators.

3.0 Digital Engine Control Reliability and Redundancy

3.1 Reliability Models for FADEC Systems

The utilization of full-authority digital engine controller (FADEC) systems for military and commercial aircraft is well under way (Newirth and Bosco 1980; and Barclay, Lenox, and Bosco 1978). The critical design requirement behind this development is achieving high reliability and availability. Current approaches use redundant sensors, processors, and actuators for meeting these reliability needs.

Design of redundant systems requires analytical tools for supporting the design process. The capability to quickly determine the effect of different design decisions on the reliability of the system is of great importance. Reliability models range from simplified parametric analysis of broad system level descriptions to detailed logic gate level models. There are numerous tools for building and analyzing reliability models including algebraic methods, computer simulation, hardware emulations, and logic analyzers. (Dotson 1988, January and August).

This section provides two examples of reliability models for electronic engine controllers, including a parametric analysis using system level models and a more detailed Markov model. An example is presented describing in detail the development of a Markov model for a dual redundant FADEC system. This model contains the detailed features of the design which impact the reliability. This detailed state model is simple to use and widely applicable. In the final subsection, additional examples are presented illustrating design tradeoffs that can be made using algebraic system level models. Although these models require parameters normally defined by more complex Markov models, they provide insight for system designers.

The operational state of a redundant system changes as the result of component failures, decisions made upon detection of faulty components, and reconfiguration resulting from control system actions to isolate and remove faulty components. Markov modeling techniques provide a natural method for determining the effects of random failures and assessing the effects on system performance. The classical fault tree (combinational) approach is less effective for dealing with changes to system configuration. Markov models easily accommodate the time varying probabilities and effects which occur in reconfigurable systems. Implementation of a Markov model is simplified because first order approximations can be used to define the state transition probabilities.

3.1.1 Detailed State Model For FADEC

The principal challenge in developing a Markov model is definition of the model states. As the number of components in a system increases linearly, the number of states in a model increases exponentially. The analyst's challenge is to construct a table of states that does not require too many entries. A complex

redundant system could be partitioned into less complex subsystems and modeled in segments. The results could then be combined logically to obtain results for the larger system (Smith et al. 1976). The approach demonstrated in this section is summarized from Gai, Harison, and Luppold 1983. This modeling approach uses prior knowledge about the system to construct a model having an acceptable minimum number of states in a single model. This is preferable to combining results from many models where any insight from the model on system behavior may be lost in the numerous subelements.

The organization of the components in a FADEC is shown in figure 3.1-1. Two processors form the central part of the system. Each processor, associated power supply, and set of dedicated inputs and outputs make up a single channel. Communication between channels is via a Universal Asynchronous Receiver Transmitter (UART) interface. This serial data channel operates at a low bandwith compared to the processor internal operations. Outputs of both processors are symmetric, allowing either channel independently to control the fuel flow for the engine and control of stator vane angles. Inputs to the system are asymmetric and include the following:

- Power level actuator (PLA) sensors
- Engine pressure (P2, Pb, P5, P_{amb}) sensors
- Fan and turbine speed (N1, N2) sensors
- Temperature (T1, T2.5) sensors
- Temperature (TD) diodes
- Feedback sensors from fuel flow actuators (WFB)
- Feedback sensors from stator vane actuators (SVA_{fb})
- Interfaces from independent air data systems (AIR)

During normal operation, the primary channel retains control of the engine until the first failure. When a failure of a sensor or actuator associated with the primary channel is detected, the primary channel substitutes the secondary sensor if available. Failure of a sensor which is not dual redundant results in a transition to a control mode not requiring that sensor, and control continues to reside in the primary channel. Transfer of control to the secondary channel occurs only in the event of failure of the primary processor or its power supply.

The reliability requirement of this engine controller is three shutdowns in a million hours. This is a probability of failure of 0.333×10^{-6}.

Redundancy management for the system is based on self-tests for failure detection and identification. The self-tests for processors and memories include watch dog timers, parity checks, and memory check sums. Built in self-tests are used for all sensors and interfacing electronics, and wraparound tests are performed

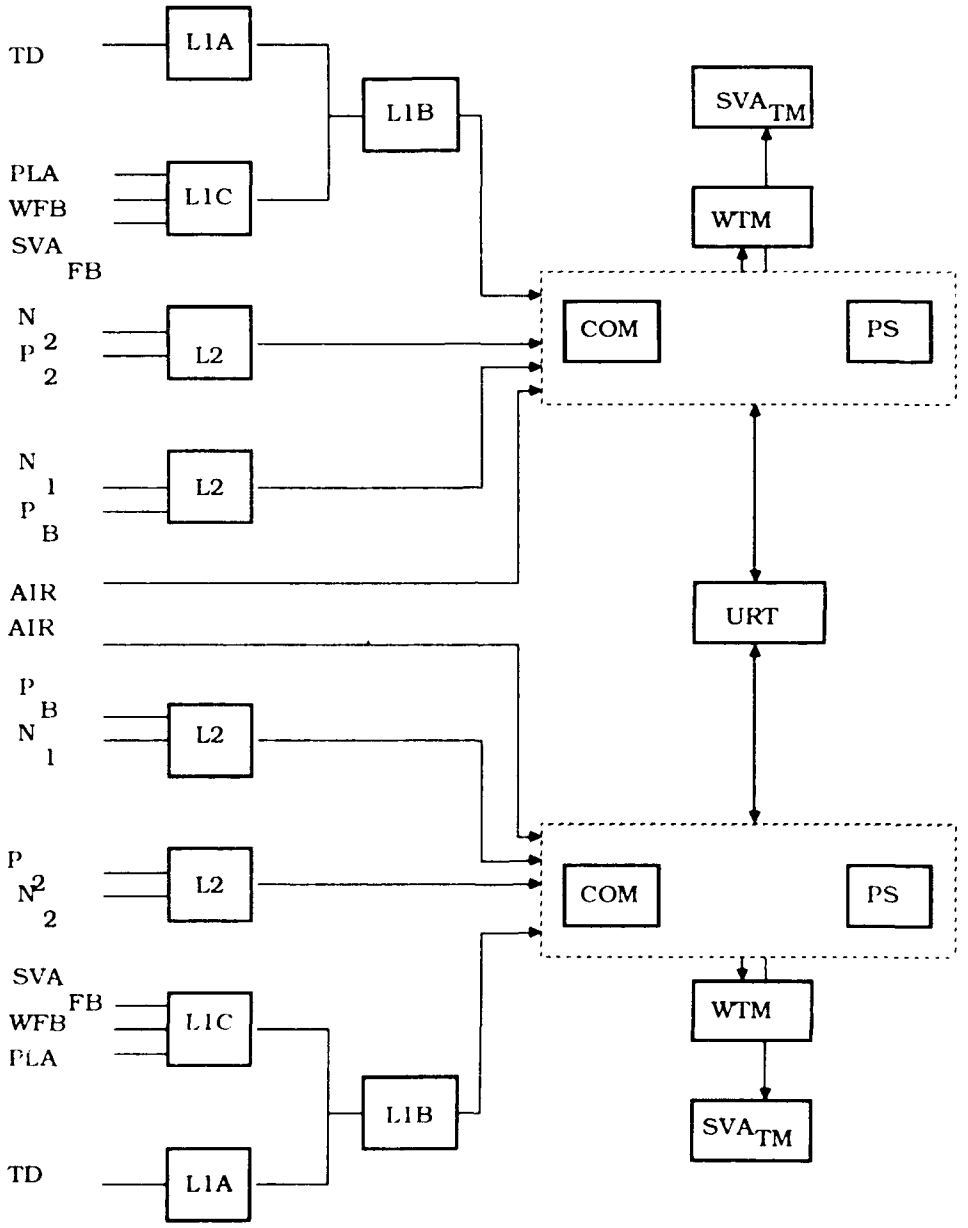


FIGURE 3.1-1. DUAL REDUNDANT ENGINE CONTROLLER

on output commands. In addition, each sensor input is tested for out of range conditions. Other redundancy management techniques include sensor comparisons and use of synthesized versions of sensor measurements as references. These analytical redundancy techniques are not considered in this discussion, but are normally implemented in operational controllers (refer to section 3.2).

The engine controller, shown in figure 3.1-1, operates in four distinct modes: Engine Pressure Ratio (EPR), Low Rotor Speed (N1), High Rotor Speed (N2), and Fuel Flow to Burner Pressure (W/P). The components that support these modes are listed in table 3.1-1. The preferred control method is EPR. For this mode the pressure sensor P5 is used for control. When P5 is not available, control is transferred to the N1 mode. When neither N1 or EPR are available, the N2 mode is used. Loss of N2 reverts to W/P mode.

TABLE 3.1-1. COMPONENTS REQUIRED FOR ENGINE CONTROL MODES

	SENSORS REQUIRED												
ENGINE CONTROL MODE	PLA	WTM	WFB	T2*	P2*	PB	P5	N2	N1	P^*_{amb}	SVA_{TM}	SVA_{FB}	T2.5
ENGINE PRESSURE (EPR)	X	X	X	X	X		X	X		X	X	X	X
FAN SPEED (N1)	X	X	X	X	X			X	X	X	X	X	X
TURBINE SPEED (N2)	X	X	X	X				X			X	X	X
FUEL FLOW (W/P)	X	X	X			X		X			X	X	X

* This information is derived from available air data.

3.1.2 Markov Model

The choice of system states for modeling is made in a systematic manner. Group 1 consists of one state representing the normal mode of operation in which there have been no failures or false alarms. Group 2 consists of those states having one failure or false alarm. Group 3 consists of those states where there have been more than one failure or false alarm.

Group 1 states, being the normal situation, follow from the normal definition of the system diagram. Group 3 states can be aggregated into the inflight shutdown state for any component of a channel or for the sensors. Group 2 states provide the main modeling challenge.

For modeling group 2, first consider the states resulting from fault detection of a failure (or false alarm) of a primary actuator or sensor. The effect of

any detected failure is the same: a sensor or actuator is designated as failed and removed from the system. Thus, for each of the sensors or actuators, one state can be used to model detected failures. Failures or false alarms of dual redundant interface modules are treated in the same way. One state can model effects of covered failures or false alarms of nonredundant components.

If the effect of removing a component from the primary channel is different from the effect caused by removal of its counterpart from the secondary channel, then failures or false alarms of primary and secondary components must be modeled by separate states. The processors, power supplies, and interface modules are in this category since removing them from the primary channel causes the transition from EPR to N1 mode. Removing them from the secondary channel leaves engine control in the EPR mode. Finally, an uncovered failure of any component in the primary control channel is assumed to result in an in-flight shut-down. There are a total of 30 states in group 2. Since any of the states from group 2 can transition to the shutdown state, the 30 states, shown in table 3.1-2, are all that is needed for the model of the dual engine controller.

3.1.3 Transition Probabilities

The second step in model development is to determine the probabilities of transition out of the system states for groups 1 and 2.

The transition probabilities for changing states are required to quantify each of the steps shown by the model. They are calculated as being single independent failure events at each time step. This assumption is completely general because the time step interval can be arbitrarily small. The single step transitions for the FADEC under consideration are shown in figure 3.1-2.

Four of the transitions out of state 1 correspond to a change in control mode. These are transitions from the EPR mode to the N1 mode (states 6, 15, 17, and 19). These transitions are due to detected and corrected failures of the P5 pressure sensor, its interfacing electronics, or the computer or power supply which make it available to the system. The transition from state 1 to state 30 accounts for the uncovered failures of components in the primary engine control channel. Finally p(1,1) represents the probability that no transition out of state 1 occurs in a single time step; this is computed as one minus the sum of the transition probabilities of all the other transitions.

As an example of the considerations used in calculating the probabilities, consider the transition out of state 2 (one PLA failure), as shown in table 3.1-3. This transition is due to failures in either of the PLA sensors or false alarms from the primary PLA. In order to calculate the probabilities, it is conservatively assumed that a covered failure of the primary channel caused the system to enter this state. As a result, the UART, processor, power supply, interfacing electronics, and PLA from the secondary channel must be included in the set of required components. Because of the detected failures and control reversion, the system is now vulnerable to failures of these additional components. This would not be the case if state 2 were the result of false alarms associated with a detected failure of the secondary PLA.

The transitions out of state 2 lead to state 26 (EPR mode with two failures), state 27 (N1 mode with two failures), or state 30 (inflight shutdown). The detailed investigation in Gai, Harison, and Luppold 1983 indicates 43 events are included in the transitions to state 26, five events lead to state 27, and 23 events lead to state 30. These computations are beyond the scope of this section and the reader is referred to Gai, Harison, and Luppold 1983.

TABLE 3.1-2. MARKOV MODEL STATE DEFINITIONS

STATE	DEFINITION
1	Normal mode, no failure or false alarms
2	One PLA failure or false alarm
3	One WFB failure or false alarm
4	One WTM failure or false alarm
5	One AIRINC failure or false alarm
6	P5 failure or false alarm
7	PB failure or false alarm
8	One N1 failure or false alarm
9	One N2 failure or false alarm
10	One TD failure or false alarm
11	One LIA failure or false alarm
12	One LIB failure or false alarm
13	One LIC failure or false alarm
14	One L2 failure or false alarm
15	L3 failure or false alarm on right side
16	L3 failure or false alarm on left side
17	COM failure or false alarm on right side
18	COM failure or false alarm on left side
19	PS failure or false alarm on right side
20	PS failure or false alarm on left side
21	UART covered failure or false alarm
22	UART uncovered failure
23	One SVA_{FB} failure or false alarm
24	One SVA_{TM} failure or false alarm
25	T2.5 or L4 failure or false alarm
26	EPR mode with two failures or false alarms
27	N1 mode with two failures or false alarms
28	N2 mode with two failures or false alarms
29	W/P mode with two failures or false alarms
30	Inflight shutdown condition

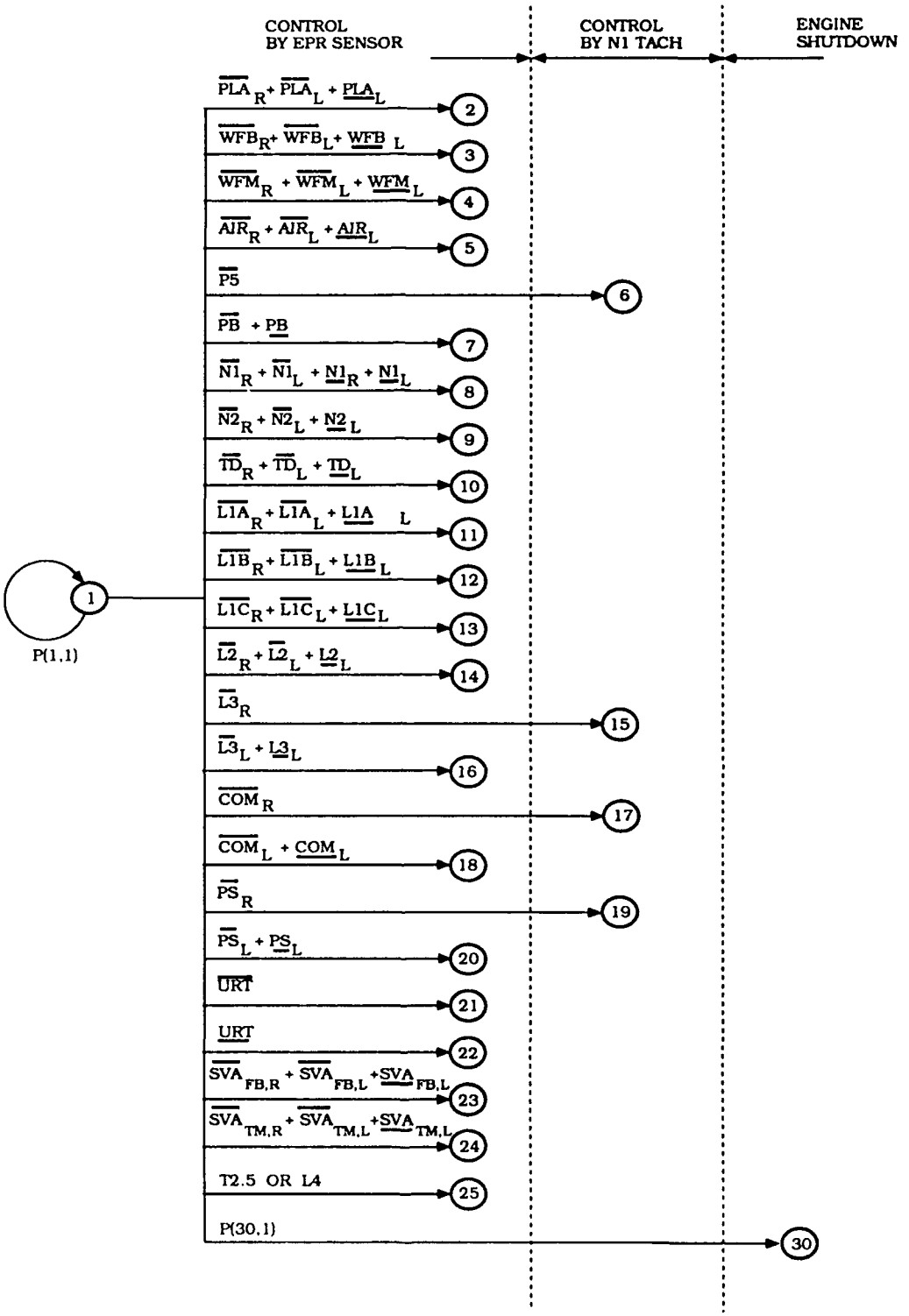


FIGURE 3.1-2. STATE TRANSITIONS FOR ENGINE CONTROLLER OUT OF STATE 1

TABLE 3.1-3. PARAMETERS FOR TRANSITIONS OUT OF STATE 2

Transitions to State 26	Transitions to State 27	Transitions to State 30
$\overline{WFB}_R + \overline{WFB}_L + \underline{WFB}_L$		$\underline{WFB}_R$
$\overline{WTM}_R + \overline{WTM}_L + \underline{WTM}_L$		$\underline{WTM}_R$
$\overline{AIR}_R + \overline{AIR}_L + \underline{AIR}_L$		$\underline{AIR}_R$
	$\overline{P5}$	$\underline{P5}$
$\overline{PB} + \underline{PB}$		
$\overline{N1}_R + \underline{N1}_R + \overline{N1}_L + \underline{N1}_L$		
$\overline{N2}_R + \overline{N2}_L + \underline{N2}_L$		$\underline{N2}_R$
$\overline{TD}_R + \overline{TD}_L + \underline{TD}_L$		$\underline{TD}_R$
$\overline{LIA}_R + \overline{LIA}_L + \underline{LIA}_L$		$\underline{LIA}_R$
$\overline{LIB}_R$		$\underline{LIB}_R + \overline{LIB}_L + \underline{LIB}_L$
$\overline{LIC}_R$		$\underline{LIC}_R + \overline{LIC}_L + \underline{LIC}_L$
$\overline{L2}_R + \overline{L2}_L + \underline{L2}_L$		$\underline{L2}_R$
$\overline{L3}_L + \underline{L3}_L$	$\overline{L3}_R$	$\underline{L3}_R$
	$\overline{COM}_R$	$\underline{COM}_R + \overline{COM}_L + \underline{COM}_L$
	$\overline{PS}_R$	$\underline{PS}_R + \overline{PS}_L + \underline{PS}_R$
		$\overline{PLA}_L + \underline{PLA}_L$
	$\overline{URT}$	$\underline{URT}$
$\overline{SVA}_{FB,R} + \overline{SVA}_{FB,L} + \underline{SVA}_{FB,L}$		$\underline{SVA}_{FB,R}$
$\overline{SVA}_{TM,R} + \overline{SVA}_{TM,L} + \underline{SVA}_{TM,L}$		$\underline{SVA}_{TM,R}$
$(\overline{T2.5 \text{ or } L4})$		$(\underline{T2.5 \text{ or } L4})$

3.1.4 Effect of Maintenance

Redundant fault tolerant systems can operate with one or more failed components and thus have higher reliability than simplex systems. This reliability enhancement only occurs if the redundant system is operated over short times relative to its mean time to failure (Clements 1980). This is because early in the operational life, not all the components are necessary for operation. As time passes and some components fail, more of the redundant components are added to the required set for system operation. As a result the failure probability of a redundant system is not constant, but increases with time. Therefore, the proper use of a redundant system is to operate it over a much shorter time than the mean time to failure. In addition, periodic maintenance is used to replace all the failed components, bringing the unit up to "like new" operational condition. Only in this way can a system be operated with low probability of system failure during the operational cycles.

The effect of the maintenance time interval on the system failure probability can be demonstrated with a simple model. Let N(t) be the number of failures per hour at time t, where $t \ll T_s$ the mean time to failure for the system. If, during operation, maintenance is performed every T_m hours, where $T_m \ll T_s$, then the expected number of failures in t hours is

$$N(t) = (t/T_m) * P_{isd}(T_m) \qquad [1]$$

The probability of failure of a dual redundant system with coverage C is given by

$$P_{isd} = (T_m/T_s) * [(1-C) + T_m/T_s] \qquad [2]$$

Equations 1 and 2 imply that the number of failures in a million (10^6) hours with maintenance is

$$N_m = (10^6/T_m) * [(1-C) + T_m/T_s] \qquad [3]$$

The parameter T_s, mean time for system failure, is determined from the Markov model solution. Since this is beyond the scope of this section, the reader may refer to Gai, Harison, and Luppold 1983 for details. The steps involve determining the model parameters for each of the state transitions in figure 3.1-2, and development of powers of a matrix, or alternatively, solutions of electrical circuit equations. The base line values for the component failure rates used for these calculations are shown in table 3.1-4.

The results of solving the Markov model for the probability of being in a given system state are as follows:

Operating in EPR Mode	0.995
Operating in N1 Mode	4.3×10^{-3}
Operating in N2 Mode	3.2×10^{-6}
Operating in W/P Mode	1.2×10^{-6}
Inflight Shutdown	4.0×10^{-4}

TABLE 3.1-4. BASE LINE COMPONENT FAILURE RATES

Component	Probability of Failure in 1 Hour
PLA	2.7×10^{-6}
WFB	2.7×10^{-6}
SVA_{FB}	1.0×10^{-6}
WTM	1.3×10^{-6}
SVA_{TM}	1.3×10^{-6}
AIR	2.5×10^{-6}
P5	4.6×10^{-6}
N2	1.0×10^{-7}
TD	1.6×10^{-8}
L1A	1.0×10^{-6}
L1B	2.1×10^{-6}
L1C	9.7×10^{-7}
L2	1.4×10^{-6}
L3	1.7×10^{-6}
COM	2.1×10^{-5}
PS	3.6×10^{-6}
URT	5.0×10^{-6}
N1	1.0×10^{-6}
PB	4.6×10^{-6}
T2.5 AND L4	2.7×10^{-6}

The probability of operating in the base line EPR Mode is very high (0.995); the probability of inflight shutdown is fairly low (4.0×10^{-4}). As a check, assume perfect coverage, C=1, and use equation 3 above to calculate a failure rate of 2.7 shutdowns in a million hours of operation.

Two other computational results are interesting. Figures 3.1-3 and 3.1-4 show the sensitivity of inflight shutdown to maintenance interval and fault coverage. These results indicate little sensitivity to maintenance interval for a 0.9 coverage. However, the maintenance interval is very important for lowering the probability of shutdown at higher levels of fault coverage. For a simplex system, inflight shutdowns are approximately eight in a million hours as shown by the curves in figure 3.1-3 for 0 coverage. Achievement of the 2.7 per million hours requires a fault detection and recovery system having fault coverage on the order of 0.95.

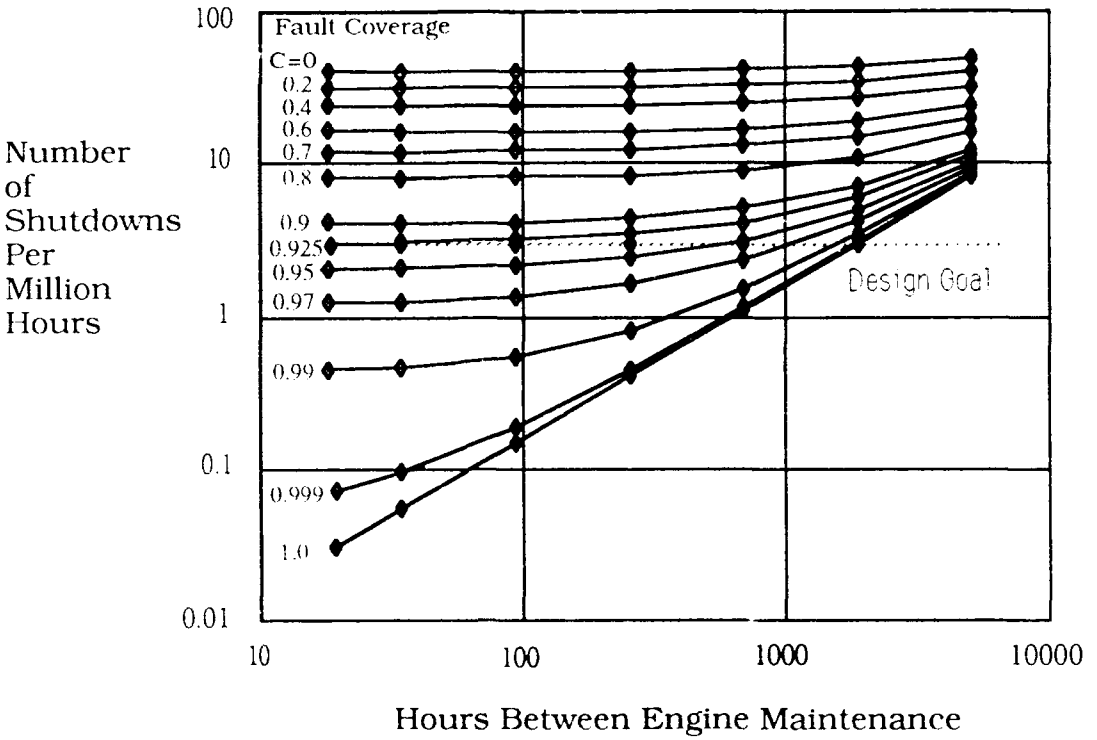


FIGURE 3.1-3. NUMBER OF SHUTDOWNS PER MILLION HOURS OPERATION - DUPLEX

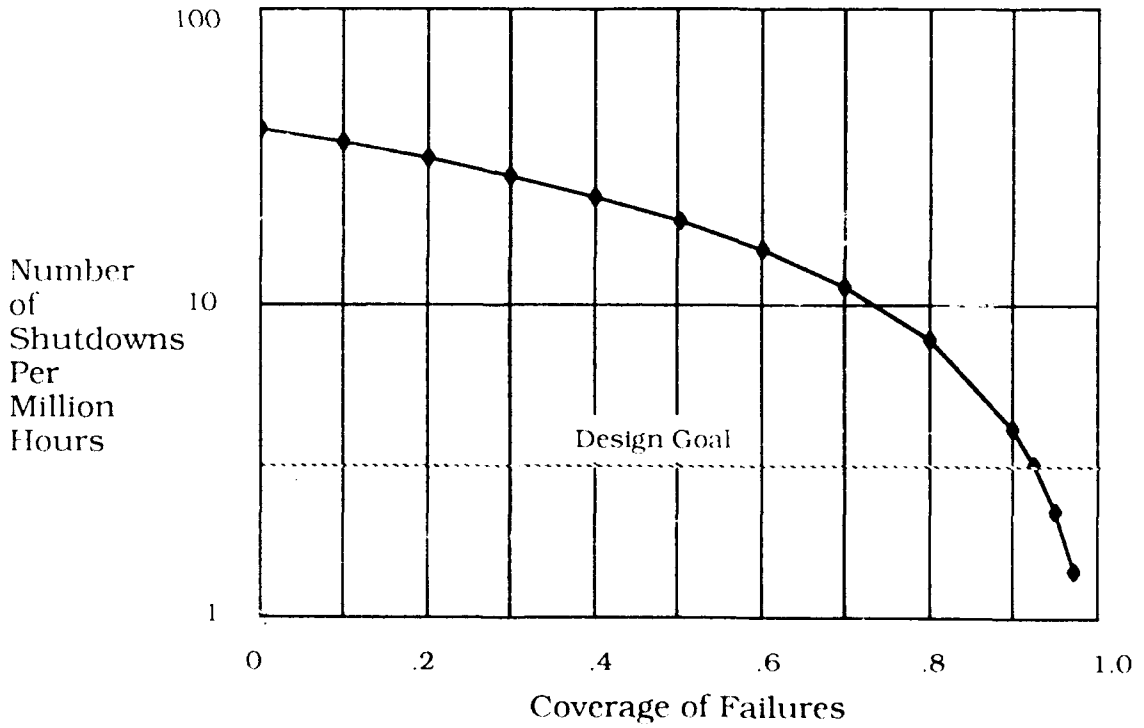


FIGURE 3.1-4. SENSITIVITY OF INFLIGHT SHUTDOWN TO COVERAGE

These results illustrate that attempting to improve system reliability by reducing the maintenance cycle limits the effectiveness of this approach unless very high levels of fault coverage can be achieved. For a controller having T_m of 150 hours and a mean failure time, T_s, of 15000 hours, coverage in excess of 0.99 is required to reduce the number of shutdowns by shortening the maintenance cycle.

3.1.5 Markov Models for System Safety and Availability

In the discussion of this subsection, the emphasis is on the two fundamental reliability objectives: system reliability and system availability. In general terms, flight safety is determined by having the minimum suite of equipment to insure no loss of life. Availability is the probability of being operationally ready at any point in time. In terms of safety, the probability of failure at a given time provides a measure of system safety. Reliability modeling is used to define and control the design for meeting the system safety and availability goals.

There is a fundamental conflict between high safety and high availability whenever redundant equipment is used for providing high reliability. For example, consider a system having a failure rate of 100 per 10^6 hours (MTBF is 10,000 hours) in operations having a two-hour flight duration. This system has a failure probability (P_f) of 2×10^{-4} and an availability (P_a) of 2×10^{-4}. The situation is quite different if a triplex (triple redundant) system is configured from these components. This new system has an improved failure probability of 8×10^{-12} (i.e., P_f^3), but a reduced availability of 6×10^{-4} (i.e., $3 \times P_a$) and MTBF=3333 hrs. Thus, while the triplex reliability is increased dramatically (nearly eight orders of magnitude) over the simplex system, the system availability has decreased by a factor of three.

This safety/availability conflict can be summarized simply. Addition of equipment in redundant or standby mode increases the likelihood that enough equipment will be working to provide safe flight. For adding redundant equipment, increasing the number of components increases the number of failures (not the system failure probability) in a given time. Since all the components are needed for dispatching the system, redundancy reduces system availability while increasing the useful operating time.

Since perfect reliability is not attainable, it is much more important to recover from failed components than to prevent them (Wulf 1975).

The probabilities can be calculated in broad terms using a system level Markov model having the following parameters:

L_t - Total Failure Rate	Rate of component failures
L_n - Nonfunctional failure rate	Failures that have no effect on the system operation
L_f - Functional failure rate	Failures that affect system performance

L_u - Uncovered failure rate	Functional failures that are not detectable or recoverable
L_c - Covered failure rate	Failures that may be detected, isolated, and recovered.

These terms are interrelated

$$L_t = L_n + L_f \quad [4]$$

$$L_f = L_u + L_c \quad [5]$$

For a single piece of equipment, coverage may be defined as

$$C = 1 - L_u/L_f \quad [6]$$

For a system or channel the coverage may be defined as

$$C = [\Sigma C_i * (L_c)_i]/[\Sigma(L_c)_i] \quad [7]$$

where C_i is the coverage for the i th component and $(L_c)_i$ is the corresponding failure rate.

Coverage is a term that requires careful attention to define properly. It is most accurately defined by testing the system by the fault injection techniques discussed in Chapter 5. In a multi-channel system, different coverage values are defined depending upon first or second failures. The first failure in a redundant system is more readily detected by the hardware and software implemented for this purpose (coverage often approaching one). Second failures in a duplex and third failures in a triplex being dependent upon self-test BIT and BITE mechanisms have a lower value of coverage (0.95 to 0.999).

The relationships defining system probability of failure in terms of component MTBF, component reliability (R), availability (Q), and coverage are as follows for duplex, triplex, and quadruplex systems (Hamilton Standard, August 1980):

- Duplex

$$P_{fl} = 2RQ(1-C_1) + Q^2C_1 \quad [8]$$

- Triplex

$$P_{fl} = 3R^2Q(1-C_1) + 3RQ^2C_1(1-C_2) + Q^3C_1C_2 \quad [9]$$

- Quadruplex

$$P_{fl} = 4R^3Q(1-C_1) + 6R^2Q^2C_1(1-C_2) + 4RQ^3C_1C_2(1-C_3) + Q^4C_1C_2C_3 \quad [10]$$

Where R = 1-Exp(-MTBF x hrs), Q = 1 - R, and C_1, C_2, and C_3 are the coverage of the first, second, and third failures respectively.

An important characteristic of coverage may now be illustrated. The likelihood of failure is extremely sensitive to the coverage of first and second failures. The first failure coverage in a triplex and the second failure coverage in a quadruplex is often considered to be unity. This is because detection techniques for first failures use cross-channel communications and direct comparisons. Changing this assumption on coverage slightly has a large effect. For example, changing the value of first failure coverage from 1.0 to 0.999 is summarized in the following figures for a system having a MTBF of 10,000 hours:

	P_f/hour (C_2 = 1.0)	P_f/hour (C_2 = 0.999)
Duplex	8×10^{-6}	8×10^{-6}
Triplex	1.2×10^{-9}	3×10^{-7}
Quadruplex	1.6×10^{-13}	4×10^{-7}

The duplex system is only shown for reference as it has no coverage for second failures. Such a change in coverage might result from ruling out cross-channel checking in a system design. The interesting point made by these figures is that the apparently small change in coverage for second failures has degraded the triplex and quadruplex to the extent that both these systems have approximately the same probability of failure.

Effects of periodic maintenance on the number of shutdowns can be developed parametrically using these algebraic models. The discussion in section 3.1.4 also used these system models for a dual system. The number of failures in a given number of hours is directly proportional to the time, provided that the probability of failure is small during the time interval. Following periodic maintenance, any system components that have failed are replaced and the system is in a "like new" state. As long as the maintenance time interval is small compared with the mean time between component failures, the number of failures can be determined from the probability of system failure.

For a triplex system an expression for the number of failures in a million hours, derived from equation 9 is

$$N_f = 10^6/T_m[3R^2Q(1-C_1) + 3Q^2C_1(1-C_2) + Q^3C_1C_2] \qquad [11]$$

where T_m is the hours between periodic maintenance. The other terms are defined in equation 10. Figure 3.1-5 shows the result of these calculations for first failure coverage of 0.999 and second failure coverage ranging from 0 to 1. Comparison of the data with that in figure 3.1-3 illustrates the benefit for a triplex, even for small values of coverage for the second failure.

3.2 Analytical Redundancy Design for Improved Engine Reliability

Control system reliability requirements for the next generation of aircraft propulsion systems are leading engine control systems toward dual or even triple redundancy. Sensor redundancy can be minimized by use of analytical models that provide signals for a failed sensor. These signals are derived from aircraft parameters and measurements using other sensors. This minimizes the need for physically replicated sensor sets. Analytical redundancy used together with a

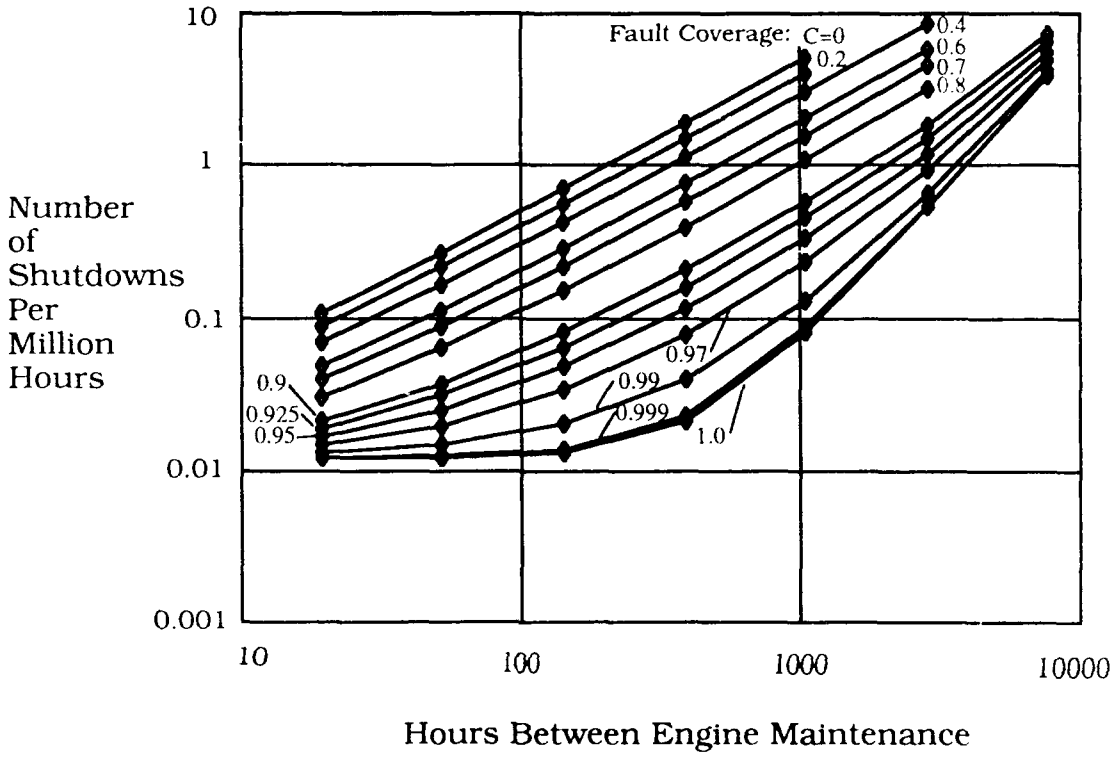


FIGURE 3.1-5. NUMBER OF SHUTDOWNS PER MILLION HOURS OPERATION - TRIPLEX

This section describes an analytical redundancy approach for increasing the reliability of aircraft engines reported in Swan and Vizzini 1988 and Brown and Vizzini 1986. (Also refer to Detroit Allison June, 1982 for a fighter aircraft study.) The original work was sponsored by the Naval Air Propulsion Center as the FADEC program (Vizzini and Toot 1980) and designated as a Failure Indication and Corrective Action (FICA) system. The initial work established the feasibility of this approach. The program represents an extension of the initial effort for a one-of-a-kind technology demonstrator to an operational product engine controller. Included in the new work is an analytical demonstration of operation over a full range of engine modes and parametric envelopes. Also included is a demonstration of fault isolation and recovery from failures in sensors and actuators. These investigations utilized a non-linear analytic model for a GE23A jet turbine engine to provide operational parameters and real-time modeling.

3.2.1 Analytical Redundancy Engine Control Concept

The control system layout, shown in figure 3.2-1, was used for the investigation reported in Brown and Vizzini 1986. It contains three main modules. The GE23A engine system predicts the outputs of engine actuators and sensors as a function of the sensed environmental conditions, actuator signals, and model updates. This engine model has been validated by prior work to represent the operation of a real engine, and is used to simplify the experiment and for providing known engine environments. The engine actuator inputs drive the non-linear engine model which provides sensor data for the control system. The engine module in the controller is an analytical model specifically designed to support the analytical redundancy system by providing model predictions for sensor outputs. The component tracking filter uses the sensor data and model values to compute differences between the sensed values and model values. This information is used to generate model updates and error information for failure detection and isolation logic. The failure-accommodation module logic computes the validated sensor information as a function of the sensor and actuator status, model values, and sensor data. In the event of actuator failure, this module provides the data for reconfiguring the engine control mode.

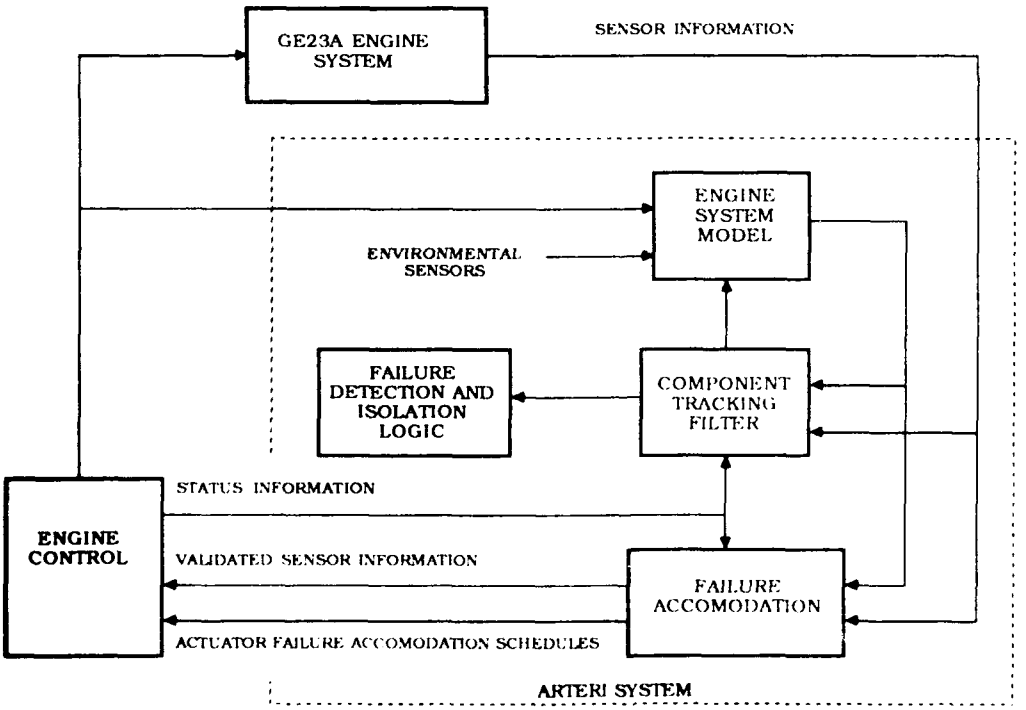


FIGURE 3.2-1. SCHEMATIC OF ANALYTICAL REDUNDANCY ENGINE CONTROLLER

3.2.2 Component Tracking Filter

The component tracking filter is the key element in this system. This adaptive filter updates engine module parameters as a function of differences between sensed and modeled values. This filter eliminates the need for manually tuning the engine model to an individual engine. Manual tuning was required for prior engine modules to accommodate differences between engines of the same type number.

There are two types of sensor failures: hard failures, where the sensor output violates physical limits on position or rate of change; and soft failures, where the sensor calibration gradually changes and slowly drifts, providing erroneous but believable values. Hard failures are the easiest to detect by parameter range checks or by range rate checks. Soft failures are much more difficult to detect.

The Analytical Redundancy Technology for Engine Reliability Improvement (ARTERI) system was designed to detect, isolate, and accommodate soft sensor failures while using the same sensor data to update the engine module parameters. The model accuracy needed for accommodating sensor failure is achieved by updating the engine module parameters to match the actual engine under control. This update process minimizes the errors between the sensed and measured values. Depending upon the rate of model update and the drift rate for failed sensors, the model may keep up with the failed sensor and corrupt the engine module. This makes fault detection and isolation difficult and may lead to high false alarms and removal of good equipment. All single-channel model-filter-based FDFM schemes have this same limitation. The ARTERI approach minimizes false alarms by integrating the functions of fault detection, isolation, and reconfiguration.

The approach to FDFM also uses the adaptive nature of the component tracking filter. Once the component tracking filter parameters were updated to match the actual engine parameters, the filter gain was reduced. Filter design requirements included the need to produce unique failure signatures, minimize corruption by soft failures, and provide a smooth transition to the model value. Smooth transitions were accomplished by decoupling the filter update process and by gain controls.

Sources of mismatch between the model and engine data are modeling inaccuracies and measurement uncertainties. Engine to engine differences occur from component quality and variation. Modeling approximations were necessary to meet the requirements of real-time operation. Measurement uncertainties are the random variation of a sensed parameter from its true value and the noise in electrical measurements. Compared with modeling inaccuracies, the measurement uncertainty is considered to be a small source of mismatch between modeled and measured data.

Two kinds of error result from model inaccuracy: steady state errors and dynamic errors. Steady state errors are the constant or near constant differences between measured and predicted values. These errors are due to parameter differences and biases in the measurements. Dynamic errors are the differences that occur during transient conditions such as during engine speed up and slow down. These errors are due to internal iteration loops, approximations for the component models, and high frequency sensor noise.

The component tracking filter, shown in figure 3.2-2, integrates four separate updates: state update, steady state error update, input update, and component update. Mismatch is minimized by a frequency decomposition of the sensed engine parameters and the model values. Actuator state updates are computed from position errors and a diagonal matrix of update parameters. This portion of the filter adjusts for steady state and dynamic errors. Engine model updates for steady state errors are computed as a function of time delayed differences between sensor measurements and predictions. Dynamic errors are treated as the difference between measured and predicted plus the sum of the steady state errors. Time constants for the filter lag are selected to be four to 10 times slower than the dominant engine response time. Engine state updates are computed as a function of the dynamic errors and an update matrix. The structure of this update is similar to a Kalman filter. However, only measured states are updated. The same structure is used for the component and updates loops. These updates drive the steady-state errors towards zero.

The structure of the component tracking filter allows the various filter updates to be computed at different sample rates. This reduces the throughput requirements of the filter. Since the state updates are related to the dynamics of the model, they are computed at the same 10 to 20 millisecond rate. The steady state error and input updates are low frequency loops computed at a rate four times slower. The lower rate is based on the assumption that the component performance levels change slowly over the life of an engine and component updates may be computed even more slowly.

For performance over the full engine operational envelope, the engine controller gains are scheduled as a function of power setting and flight condition. Initial gains are scheduled as a function of fan speed and sea-level operational conditions, thus accounting for various power settings. These first order curve fits are modified as a function of flight condition (altitude and mach number) based upon measured engine inlet environments. Inlet temperature and pressure measurements, divided by their sea level values, provide scale factors to correct the sea level model errors for speed, temperature, and pressure.

3.2.3 Failure Detection And Isolation

Errors computed from the component tracking filter form three separate error vectors. These vectors define an n-dimensional space, where n equals the number of vector components. When a failure occurs, the path that the vector follows is defined as that failure vector's signature.

The criteria established for computing the engine state update and input updates decouple the error vectors. When the error vectors are decoupled, the failure of one component will not affect the error signals for the others. The update criteria are designed as follows:

- Design the state update matrix so that the error produced by a failure in one sensor does not create errors for other sensors because of an update.

- Design the input update matrix so that the error produced by failure in one input does not create an error in other input channels because of the update.

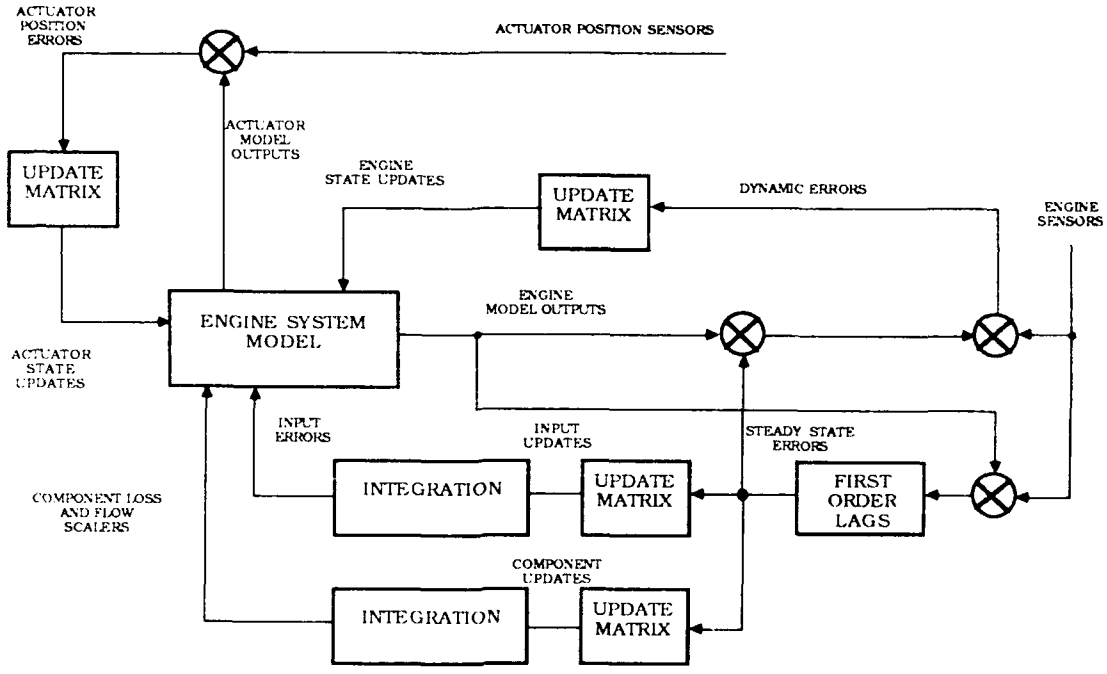


FIGURE 3.2-2. COMPONENT TRACKING FILTER

As a result of this decoupled design, engine state and input updates produce axial signatures. When failure occurs with this design, a change in error signal will only be observed in the failed channel. This design method is discussed at length in Brown and Swain 1987. The advantage of the axial error vector is that the error signatures only need to be tested; storage of the vector is not required. Out of bound indications will only appear in the error signal of the failed component. Bounds on the variation and rate of variation need to be established for each component. The individual error signals are compared with the established bounds to detect faults.

Examples of steady state and dynamic failure signatures for a soft failure in a fan speed sensor are shown in figures 3.2-3 and 3.2-4, respectively. Note that for these plots the threshold value for each channel is the full scale value (+FT and -FT) for each plotted parameter. Failure isolation can easily be accomplished with the data shown.

3.2.4 Failure Management

Failure management logic is based upon the following priorities: first, engine safety; and second, maintaining thrust and specific fuel consumption.

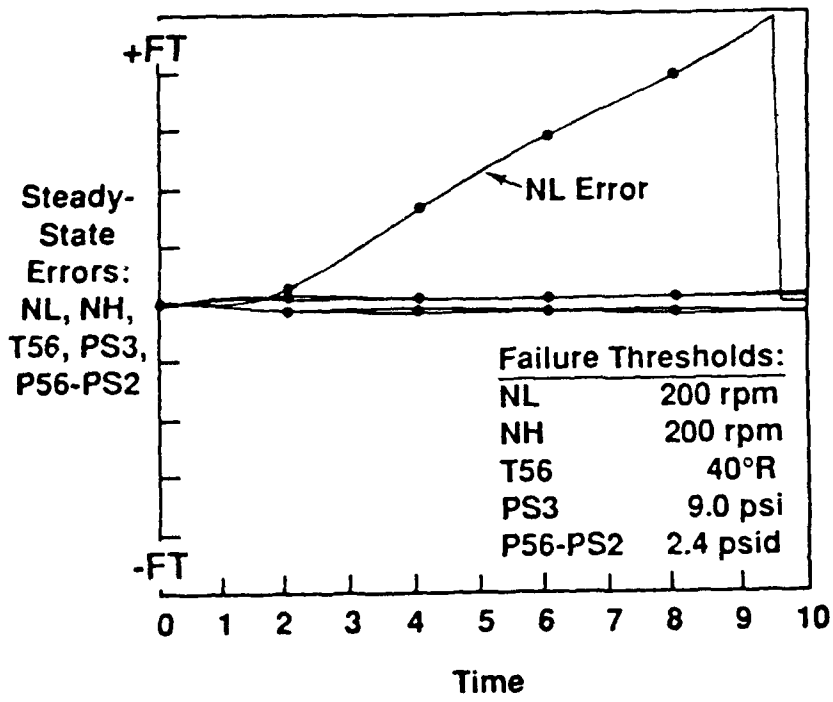


FIGURE 3.2-3. STEADY STATE ERROR DETECTION - FAN SPEED SENSOR

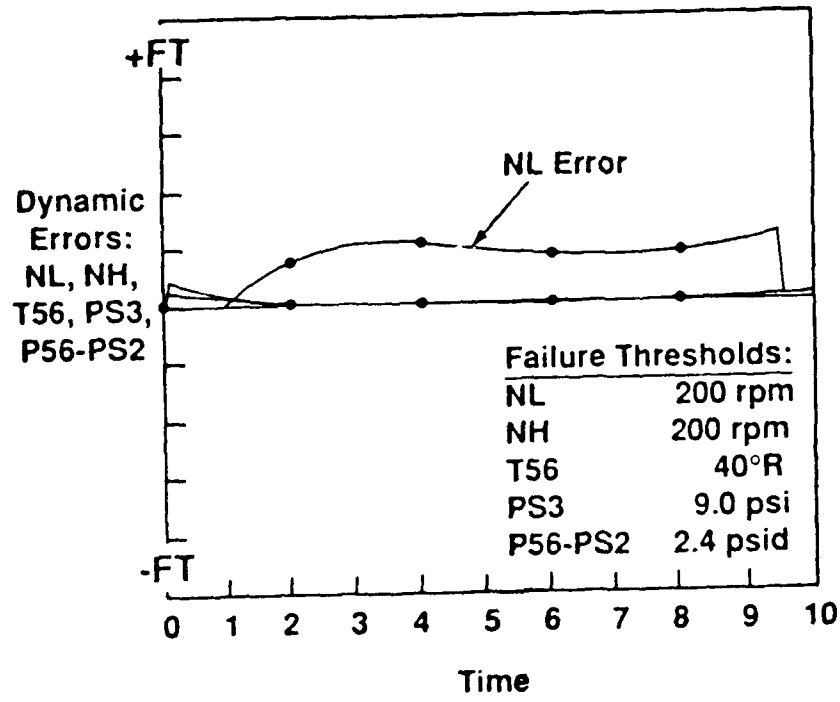


FIGURE 3.2-4. DYNAMIC ERROR DETECTION - FAN SPEED SENSOR

Failures of sensors and actuators must be detected and acted upon to isolate and correct engine control modes before any serious engine damage or loss in performance occurs. Corrective actions for sensor or actuator failures should not induce any large transients that could lead to loss of control. These corrective actions are accomplished by logic activated by error signals exceeding error threshold values.

Sensor failures are accommodated by analytical redundancy where the analytical sensor models replace the failed sensor. Smooth transitions are accomplished by computation of the valid sensor value from the sensor value, model value, and control gains. Actuator failures are accommodated by reconfiguring the control mode using fail-fix servovalves and failure accommodation schedules.

The primary failure usually observed for an actuator is a short or open circuit of a torque motor. When such a failure occurs, the fail-fix servovalve will cause the actuator to move to a preferred position. This position (open or closed) is selected to maintain safe engine operation and maintain performance as high as possible. The control schedules for the remaining actuators use precomputed schedules to minimize loss of engine performance. When safe operation cannot be maintained, the engine will be shut down. For other actuator failures, the corrective action is to shut off the torque motor allowing the fail-fix servo actuator to move to the preferred failure position.

Table 3.2-1 (Brown and Swain 1987) summarizes the results of the actuator failure accommodation studies. Note that for a compressor variable stator vane (VSV) failure, a fully open direction could result in blade flutter and possible engine damage, while a fully closed direction considerably reduces thrust.

3.2.5 Test Results

A number of tests were conducted to validate the design and implementation of the prototype engine controller. These tests included convergence tests to verify operation with different engines over the operational envelope; transient tests to verify the ability to track during large transient changes in engine operation including acceleration and deceleration; and fault detection, isolation, and accommodation tests to verify the controller ability to detect and correctly handle hard and soft failures.

Results for the soft failure tests, being the most difficult, are summarized in table 3.2-2. Figure 3.2-5 illustrates the resulting corrective action for a fan speed sensor failure. Note that the closed loop control for fan speed maintained constant speed prior to isolation of the failure. The controller tracks (maintains the demanded speed) from the failing sensor signals by increasing fuel flow. This is evident by noting the change in fan speed computed by the model. Also note that normal engine operation was resumed by the time the failure was detected. The results of accommodating a soft failure in the main fuel metering valve actuator sensor (WFM) is shown in figure 3.2-6.

The program (Brown and Swain 1987) summarized in this section developed an operational prototype engine controller having fault tolerance for both hard and soft failures. Operation of the prototype was validated by tests using a

non-linear simulation of a GE23A turbofan engine, although the design is adaptable to any engine.

TABLE 3.2-1. ACTUATOR FAILURE ACCOMMODATION TEST RESULTS

Actuator	Preferred Failure Direction	Effect on AB Operation	Effect on Dry Operation	Fault Accommodation Schedules
Afterburner Fuel Metering	Closed	Not Permitted	Normal Operation	N/A
AB. Exhaust Nozzle Area	Closed	Not Permitted	Maximum of 12% Thrust Reduction of 46% Reduction in Fan Stall Margin for Certain Altitude and Mach Number above 15,000 ft. and Mach 1.5 at IRP Increase in Flight Idle Thrust	Reduced Corrected Fan Speed Control Reference Schedules
VSV. Compressor Variable Stator Vane	Closed	Not Permitted	Shut Down Engine	N/A
WFM. Main Fuel Metering Valve	Closed	Not Permitted	Shut Down Engine	N/A

Tests on the prototype control of a simulated GE23A engine demonstrated the capability for achieving the following:

- Updating the control system engine model to match engine variations due to engine component quality and deterioration.
- Detecting and isolating hard and soft failures of environmental sensors, internal engine sensors, actuator position sensors, and engine actuation systems.
- Maintaining normal engine operation following single sensor failures.

- Maintaining safe engine operation following single failures of engine actuators.

Continued development and testing with a demonstrator engine is recommended.

Analytical redundancy appears to be viable for application to engine control and there should be commercial applications for this technology.

TABLE 3.2-2. SOFT FAILURE TEST SUMMARY

	Failures Simulated	Result
Engine Sensors	NL, Bias; Ramp ± 50 rpm/sec NH, Bias; Ramp ± 50 rpm/sec T56, Bias; Ramp ± 5° R/sec PS3, Bias; Ramp ± 3 psi/sec P56-PS2, Bias; Ramp ± 8 psid/sec	Detected, Isolated, and Accommodated
Actuated Sensors	VSV Bias; Ramp + 10%-st/sec WFM Bias; Ramp -5%-st/sec A8 Bias; Ramp +5%-st/sec	
Actuator	VSV Bias; Stuck Actuator, NL_{Rel} 105 - 95 WFM Bias; Stuck Actuator, NL_{Rel} 105 - 95 Uncommanded A8/Open, + 10 ma TMC	
Environ-mental Sensors	T2, Bias; Ramp +2.5° R/sec	
	PS2 Bias; Ramp -.25° R/sec	Detected, Turnoff Component Update

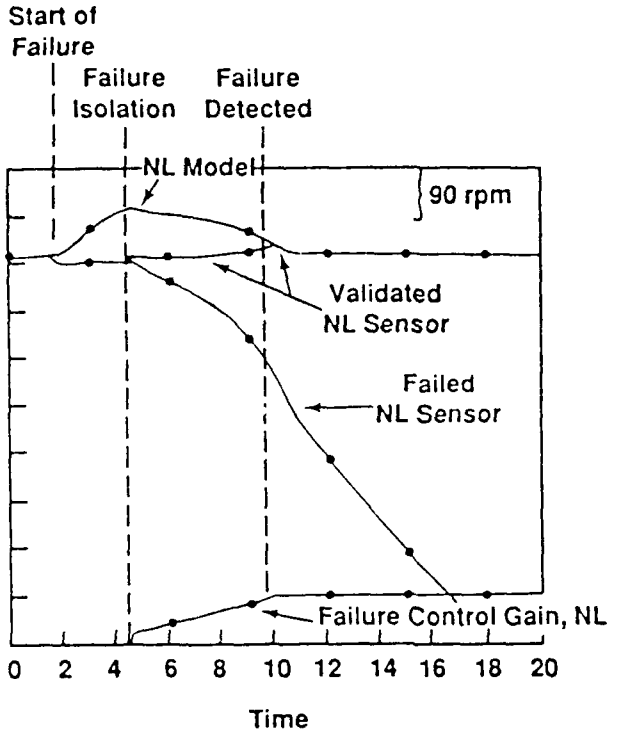


FIGURE 3.2-5. CORRECTIVE ACTION - FAN SPEED SENSOR

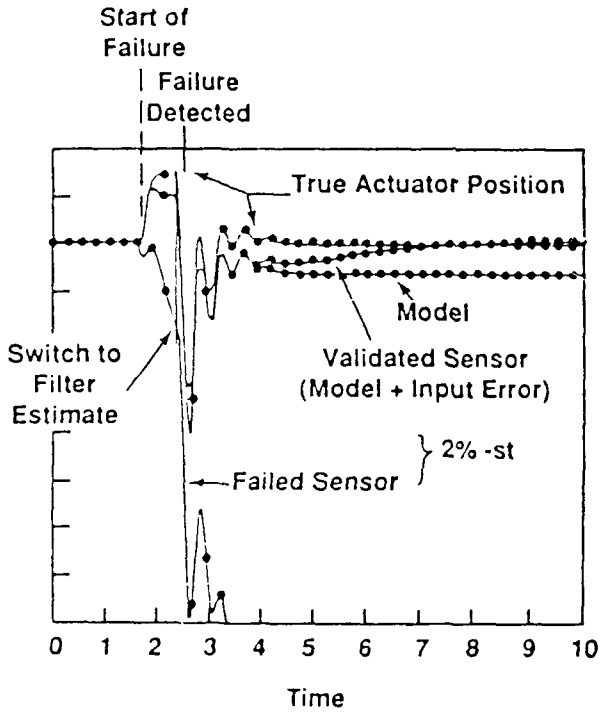


FIGURE 3.2-6. CORRECTIVE ACTION - FUEL FLOW ACTUATOR SENSOR

4.0 Primary and Secondary Electrical Power

4.1 Aircraft Power Systems

4.1.1 Power System Reliability Requirements

Electrical power system reliability directly affects the reliability of flight control and other flight critical avionic systems. The reliability of a single channel power system having a generator, generator controller, and load switches will not meet these reliability requirements. Hence, fault tolerance, including detection, isolation, and recovery, is an integral part of an aircraft power system. Material for this section is drawn from Mehdi and Leong 1985, Bret 1987, and Perkins and Marek 1977.

Requirements for reliability and redundancy, developed for the flight control and engine control systems, must be maintained for power generation and distribution. In a conventional power system, electrical generators driven by the aircraft engines provide the primary electrical power. A single generator on each engine can provide enough electrical power. A modern aircraft having computer equipment uses multiple generators and a backup auxiliary power unit (APU) to supply no-break power (at least no breaks longer than a few milliseconds). Flight critical systems operate from batteries charged from the generators.

Multiple generators may be needed on each engine to provide the reliability necessary for fly-by-wire flight controls, although a single generator could supply the electrical power. The F-16, for example, has a quadruplex redundant flight control system and a single jet engine. This aircraft also has multiple generators and batteries to provide an uninterruptible source of power.

Computer components are usually more susceptible to transients and noise on power lines than older technology. Added filtering is needed either in each component or in the power generation and distribution system.

Battery systems are needed on board the aircraft for operating critical systems in the event of an engine-out condition. Flight critical equipment must have a reliable source of electrical power to maintain safe flight and landing capability under extremely unlikely failure conditions.

Fast acting contactors and microprocessor generator controllers are needed for several reasons. Besides being quickly handled, faults in the electrical system must be handled so not to disturb power quality. Power transfer from one bus to another, such as from ground power to aircraft power, must not interrupt the supply. The normal contactors operate in a period of about 25 milliseconds. This delays the shut-down and turn-on time to about 50 milliseconds (DO-160 or MIL-STD-704). (Refer to Bret 1987 for a discussion of no-break power transfer.)

There is a trend in military power systems towards using 270 volt dc power for actuation systems. This places added requirements on power generation and distribution systems currently designed for 28 volt dc and 115 volt ac.

4.1.2 Electrical Power System Elements

Electrical power generation for a multiengine aircraft uses generators with constant speed drives mounted on the main engines. These generators can be isolated or paralleled for operation on the aircraft. Multiengine aircraft normally have parallel systems with three or more generators. Electrical systems for two-engine aircraft are usually operated in isolation. There usually are provisions, however, for tying the buses together if an engine or generator faults.

Failures can occur in the generator channel, distribution bus, avionic equipment, and wiring between these items. Protective devices and contactors segment the system for isolating faulted components. These contactors also supply power to the operating equipment.

Information measured from parts of the system provides fault identification and reconfiguration by control of these contactors. The electrical system is instrumented for measuring voltage and current at many locations. The instrument signals are fed to contactors in the power system control units including the generator control unit, bus control unit, and distribution center. Each group of contactors and instrumentation independently controls a portion of the system.

Because of independent control of the power system zones, faults in the system may allow an operating generator to reconnect or stay in parallel with a channel having an uncleared fault. This situation can be complicated if more than one failure happens or if a good generator is shutdown in error.

Fault tolerant power generation and distribution systems use fault detection and isolation at the local level. This tolerance may be accomplished by combining several techniques in the power system that include the following:

- Generator redundancy
- Power bus redundancy
- Functional load redundancy
- Power routing using bus interties
- Load shedding on prioritized basis

An aircraft power system may be configured in several ways to achieve high reliability. Power system redundancy may be accomplished by using several isolated channels or by reconfiguration using contactors.

An isolated channel approach, shown in figure 4.1-1, provides maximum independence between channels; it does not allow reconfiguration of power system elements for fault recovery. This approach provides isolation for power faults.

However, each single channel has only one power source, contactor, bus, and power control unit. The operation of the power controllers is discussed in section 4.2.1. Since the failure of any element leads to power failure, system reliability suffers. The reliability of a channel is the product of the element reliabilities. This isolated channel approach will meet flight control system reliability goals with the reliability of today's power systems.

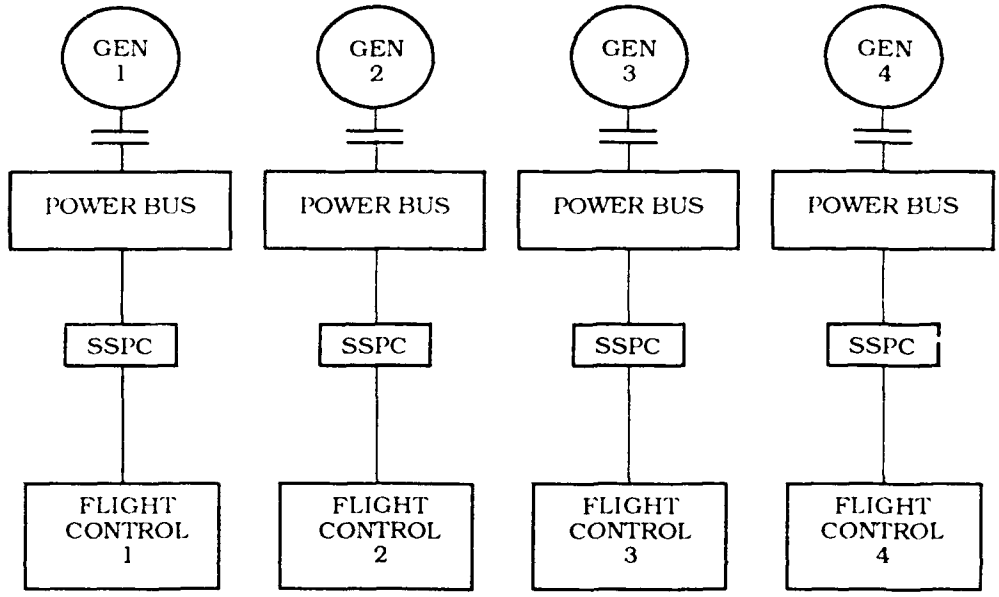


FIGURE 4.1-1. ISOLATED CHANNEL POWER SYSTEM

A reconfigurable approach, shown in figure 4.1-2, uses cross bus interties to provide power source redundancy. This approach does not have a "brick wall" isolation for flight control channels. This redundancy approach needs careful design and testing of the fault detection and fault isolation contactors. This example shows a three-generator power system supplying a four-channel flight control system. The use of two main power buses and contactors allows any generator to power any flight control system.

The reliability of the reconfigurable system is higher than the isolated channel system. Although in this system, correcting the effects of power system faults needs reconfiguration of working elements for maintaining power. The reliability of these contactors and their controls must be higher than those used for the system in figure 4.1-1. If the contactor controls fail, then the system will not operate properly. Present aircraft use manual contactors as a backup, if not the primary, for system configuration. Some systems use automatic load shedding (disconnecting non-critical loads) to keep critical equipment working.

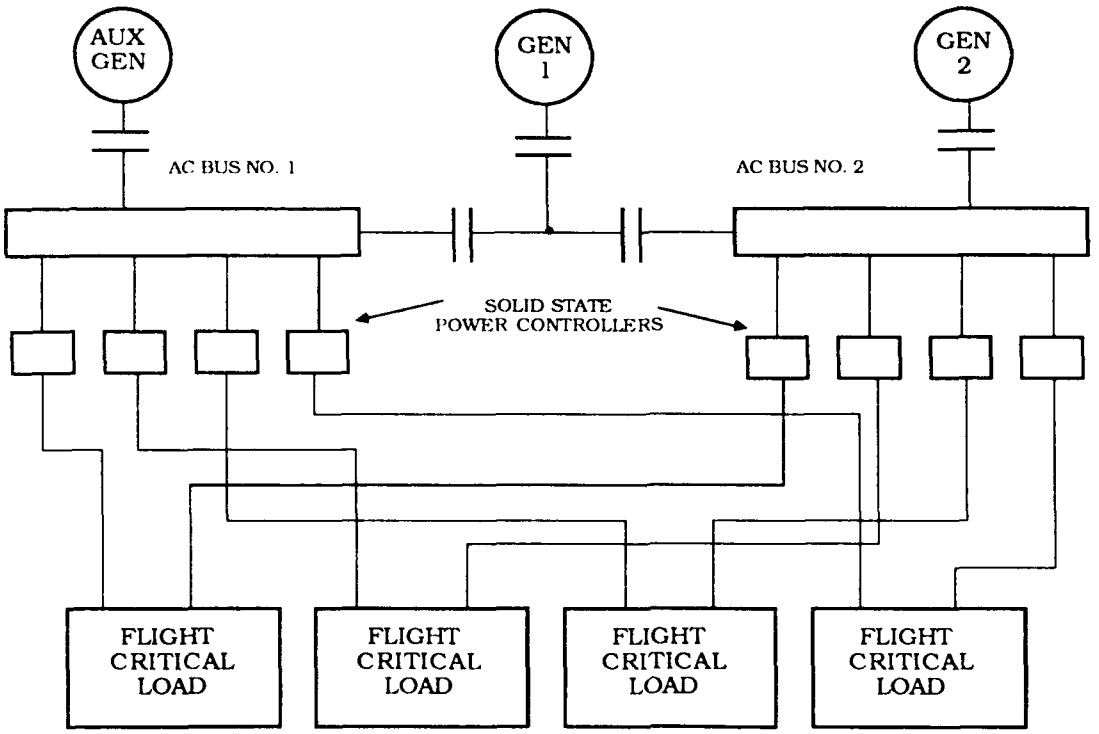


FIGURE 4.1-2. RECONFIGURABLE POWER SYSTEM

The most serious faults in any system are those affecting all of the channels at the same time. These common mode failures are often the hardest to protect against. The two-bus system, shown in figure 4.1-2, has two contactors between the main buses that give some common mode protection against a simultaneous bus fault and single contactor failure. Such a failure would prevent the isolation of the main generators. Careful system analysis and design can assure that such unlikely events have a lower probability of happening than any other event causing loss of flight control.

Power system design documents should provide fault trees and FMECA that include common mode effects and the normal component failure rate analysis.

4.1.3 Technology Trends

Advanced technology avionics have increased electric power demand and need power sources having improved reliability and power quality. The increased capability provided by application of electronic equipment to aircraft results in more equipment onboard, although the power demands of single systems have not increased. Large scale integrated semiconductor circuits are more susceptible to electrical transients and measures must be taken to protect them. A properly designed avionic system can operate in a very noisy environment with present aircraft power quality. The trade-off between protecting each unit of equipment

versus improving the power system source often comes out in favor of improving the source. This is particularly true of large commercial aircraft.

Uninterruptible power supplies (UPS) are needed for operating electronic equipment in the event of an interruption, or total loss, of electrical power generation. Flight critical systems need a back-up electrical power supply if all the engines or generators fail on a two-engine aircraft. Uninterruptible power can be supplied from batteries that fill in the gaps made by switching generators and by loss of generation.

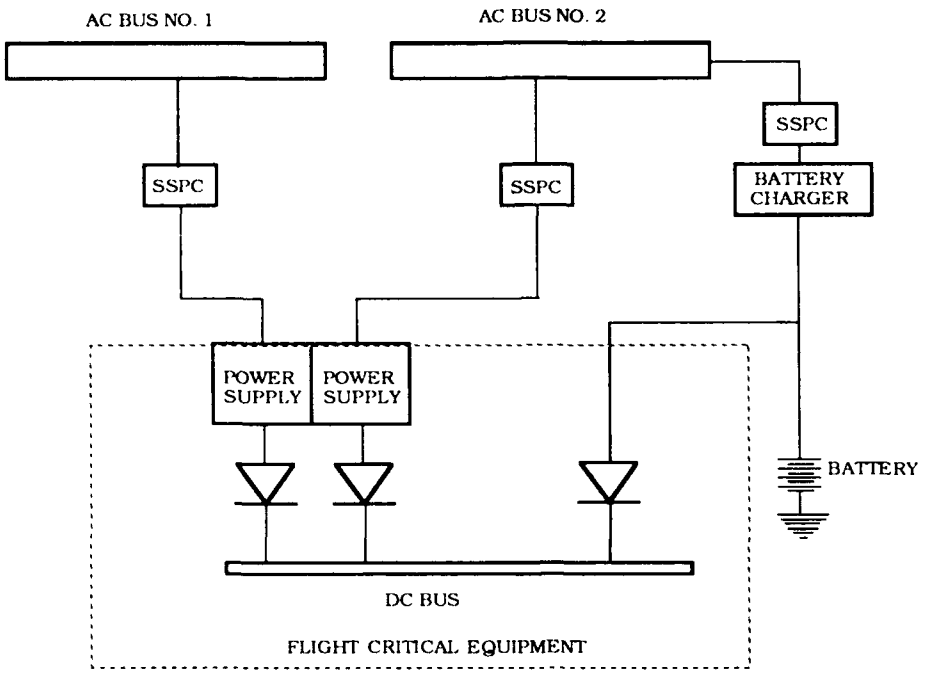
Two basic types of electrical power are available onboard an aircraft: ac and dc. Ac power is used for high energy electrical devices such as motors for landing gear and hydraulic pumps. Dc power could be used by electronic equipment, although present equipment is often configured for ac power. Some equipment operates from 28 volt dc and generates other dc levels internally. There is a trade-off between supplying power that the equipment most easily uses or putting power conversion equipment inside each component. For large aircraft the trade-off is on the side of supplying appropriate power to the equipment and minimizing power conversion in the equipment.

A current military aircraft system design trend to 270 volt dc power generation and distribution has several advantages over ac power. The major one being the elimination of the complex control and synchronization needed to parallel ac generators and to maintain frequency control with engine driven generators. An important consideration is supplying a UPS for computer equipment.

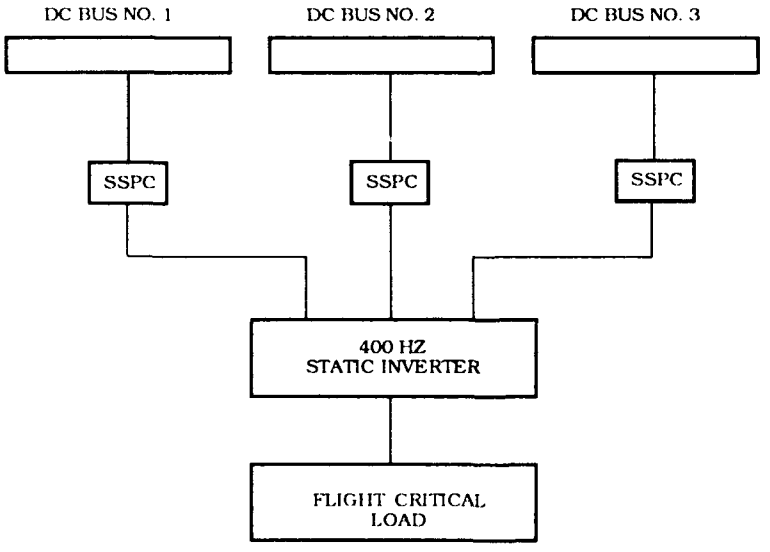
One method of providing uninterruptible power is by use of batteries supplied by rectifiers from ac generators. When a battery is used for primary power, other sources of power may easily be paralleled with diodes, as shown in figure 4.1-3. The battery is fully charged under normal conditions by a charger on the main power supply. Upon failure of the transformer rectifier unit (TRU), the battery will handle the load. This is called a battery float UPS. Batteries can handle the complete flight critical part of avionics and flight control for a period of thirty minutes to an hour.

Lead acid batteries are preferable to nickel cadmium because of the long time on float. Lead acid batteries may be fully charged with a 28-volt supply. But, nickel-cadmium batteries designed for 28 volts need a charger operating at about 36 volts. When discharging, the nickel-cadmium battery voltage drops quickly to 28 volts and stays there over the entire discharge time. Under normal float operation, the nickel-cadmium battery powered equipment must handle 36 volts. TRUs are typically unregulated and the higher voltage will pass to ac equipment as well.

Another UPS method is suppling dc directly from dc generators and using inverters to provide ac power where needed. Multiple sources of power can supply a flight critical bus. Figure 4.1-3 illustrates such a configuration for a TRU supplying a flight critical load.



A. BATTERY FLOAT UNINTERRUPTIBLE AC POWER



B. STATIC INVERTER VERSION OF UNINTERRUPTIBLE A C POWER

FIGURE 4.1-3. INTERRUPTIBLE POWER SUPPLY METHODS

4.2 Power Systems Detailed Examples

4.2.1 Fault Protection

The complexity of power generation and distribution systems for attaining high reliability needs automated fault protection. The following concepts provide isolation and protection against power system faults:

- Differential current protection
- Solid-state power controllers (SSPCs)
- Electromechanical power controllers
- Thermal circuit breakers

Generator feeders and main buses use differential current protection. This type of protection uses balanced transformers and voltage detection. The system provides fast and accurate protection levels. Operation is by a difference in current between two ends of a power feed, one being the load end and the other the bus end. Thus, any fault in the feed will be detected as a difference in the measured currents. Protection zones can be positioned to detect and isolate faults.

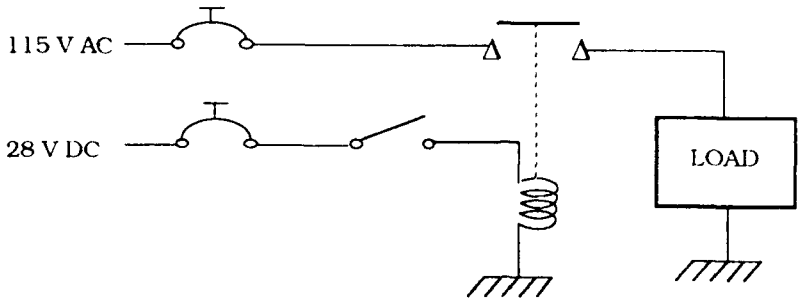
SSPCs replace the older thermal circuit breakers. SSPCs provide the same over current protection of the older units and provide a fast acting on-off switching control of electrical loads. Figure 4.2-1 shows an application where an SSPC replaces two circuit breakers and a relay for load switching: Fast fault clearing times can be achieved by using field effect transistors as the power switch. Test results from Mehdi and Leong 1985 indicate the fault isolation performance for five-amp 115 volt ac controllers with field effect transistors. When switched onto a 2,400 amp fault, the maximum current let through by these contactors was only 82 amps; the devices switched in about 10 microseconds.

Electromechanical power controllers or remote controlled circuit breakers perform the same function as SSPCs. Because they are mechanical devices, the switching speeds are slower and the devices are larger. These devices are suitable for high currents, as with larger motors.

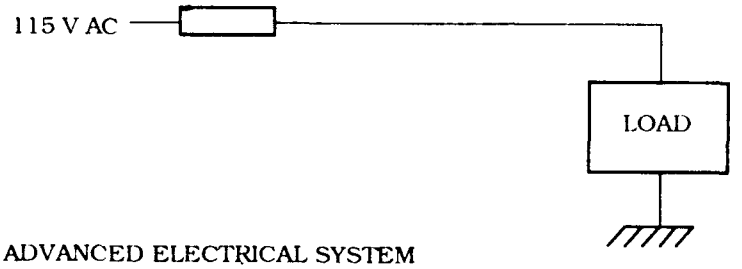
4.2.2 High Voltage dc Generation and Distribution

This section provides an overview of the technology for high voltage dc generation and distribution. This technology was developed under a Navy program (Perkins and Marek 1977) and defined an Advanced Aircraft Electrical System (AAES).

Key features of this technology are high voltage dc generation, microcomputer generator controls, and SSPCs. These components were designed and built to provide electrical power for an A-7E fighter aircraft. A prototype system, built and tested for a single channel, supplied the system parameters. Trade studies were conducted on the benefits of applying this technology to an entire aircraft.



A. CONVENTIONAL



B. ADVANCED ELECTRICAL SYSTEM

FIGURE 4.2-1. SOLID STATE POWER CONTROLLER SWITCHING

The AAES performed better than the conventional system in several areas. The areas of improvement included the following:

- Full system built-in test (BIT) showed failures down to the avionics unit and detected faults related to wire, connector pin, and terminations.
- The prototype included load management and power conditioning in the AAES although these functions are not feasible in the conventional system.
- The AAES design resulted in improved reliability by use of redundancy and distributed load management; neither is possible in the conventional system.
- Size, weight, and maintainability were all improved over the conventional system.

A trade study conducted to compare the High Voltage dc (HVDC) system with the conventional electrical system further illustrates the advantages of this system. An HVDC power system was designed for an A7-E aircraft using the prototype HVDC equipment parameters. The prototype AAES needed power conversion from 270 volt dc to standard 28 volt dc and 115 volt ac power. Since the penalty of distributing the three power levels biased the prototype parameters, the trade study included a "True HVDC" and "Projected HVDC" design. The AAES power system block diagram is shown in figure 4.2-2. Also shown are the military designations

for the hardware in the design from Perkins and Marek 1977. This prototype design uses samarium cobalt permanent magnet type generators for the 270 volt dc main and emergency generators.

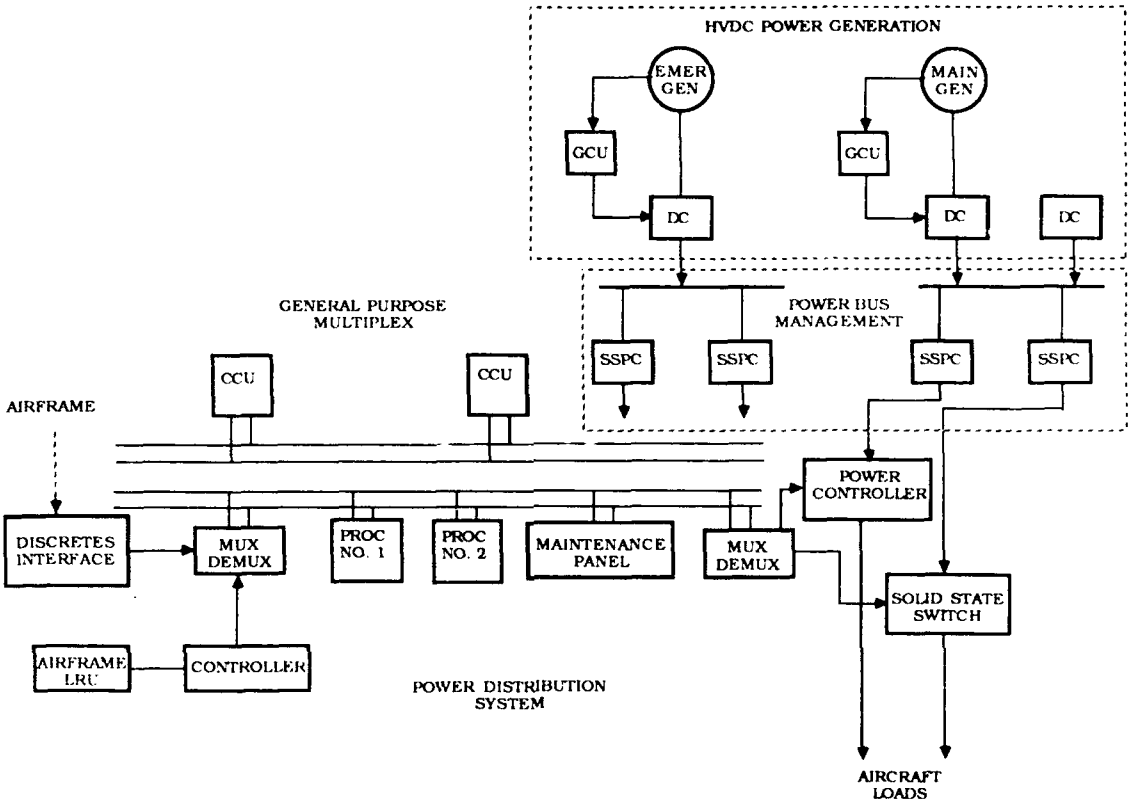


FIGURE 4.2-2. ADVANCED AIRCRAFT ELECTRICAL SYSTEM DIAGRAM

Comparative data from the trade study are presented in table 4.2-1; the "Projected" column shows the projection of the trade study to a new avionics suite that does not need 28 volt dc and 115 volt ac. If the avionics equipment were powered directly from 270 volt dc, then the 28 volt dc and 115 volt ac power conditioning and distribution could be minimized. Thus the "Projected" data is important for new aircraft.

Several benefits for an HVDC power system are given in Perkins and Marek 1977.

To the Equipment Manufacturer:

- Reduced voltage transients
- Uninterruptible power
- Smaller voltage variations
- Improved protection
- Smaller size and weight

TABLE 4.2-1. HVDC VS CONVENTIONAL POWER

PARAMETER	CONV	PROTOTYPE AAES	PROJECTED HVDC
Weight (Pounds)	190	76	64
Power Generation Power Bus Management SOSTEL Power Distribution	9	3	3
Equipment	177	223	163
Wiring	312	121	109
Total	688	423	339
Efficiency (%)	61	77	80
Reliability - MTBF (Hours)			
Essential Loads	63	509	625
All Loads	170	1442	1966
Maintainability (MMH/FH)	0.498	0.245	0.228
Cost			
Nonrecurring	Base	1.3B (-330)	0.5B (50)
Life Cycle	Base	0.3B (70)	0.2B (80)

NOTE: Numbers in parentheses show percentage of improvement over conventional power.

To the Airframe Manufacturer:
- Lower system weight
- Improved power quality
- Simplified system design
- Ease of parallel operation
- Higher overall efficiency
- Lower personnel hazard
- Easier change incorporation
- Elimination of constant speed drives

To the Engine Manufacturer:
- Lower overhung moment
- Lower weight

High- versus low-speed performance
Smaller size
Higher efficiency

To the Owner:
Lower life cycle cost
Improved reliability
Improved maintainability with BIT
Higher system efficiency
Lower personnel hazard
Lower weight

These benefits and economic factors in favor of HVDC power systems assure that further applications are expected in future aircraft.

BIBLIOGRAPHY

Abrams, C. R. and S. T. Donley, "Flight Test of a Helicopter Fly-By-Wire/Light Actuation Control System", Proceedings of the National Aerospace and Electronics Conference, Dayton, OH, May 1984.

AiResearch, Electromechanical Actuation Feasibility Study, AD-A031 146, US Department of Commerce, May 1976.

Baker, W. and W. Bonnice, Intelligent Fault Diagnosis and Failure Management of Flight Control Actuation System, NASA Contract NAS2-12414, NASA Contract Report #177481, May 1988.

Barclay, B. A., T. G. Lenox, and C. J. Bosco, "Full Authority Digital Electronic Control-Highlights of the Next Generation Propulsion Control Technology", Gas Turbine Conference and Products Show, London, April 9-13, 1978.

Biafore, L. P. and L. P. Grieszmer, "Navy Advanced Flight Control Actuation System - Analysis and Design for a Slimine Building Block Actuator Concept", Technical Report NADC-82039-60, Naval Air Development Center, Warminster, PA, November 1983.

Breit, J., "Aircraft No-Break Electrical Power Transfer", IEEE Transactions, May 1987.

Brown, H., and J. A. Swan, "Analytical Redundancy Design for Improved Engine Control Reliability", General Electric Co., Aircraft Engine Business Group, Cincinnati, Ohio, NAPC-PE-171C, October 1987.

Brown, H. and R. W. Vizzini, "Analytical Redundancy Technology for Engine Reliability Improvement", Aerospace Technology Conference and Exposition, Long Beach, CA, October 13-16, 1986.

Chandler, P. R. and D. P. Rubertus, "A System Approach to Flight Control Reliability and Maintainability", AIAA Paper 84-2463, AIAA/AHS/ASEE Aircraft Design Systems and Operations Meeting, San Diego, CA, October 31 - November 2, 1984.

Chenoweth, C. C. and D. B. Slaugh, "Microprocessor Controlled and Managed Fly-By-Wire Hydraulic Actuator", Proceedings of the National Aerospace and Electronics Conference, Dayton, OH, May 1985.

Clements, R. R., "Reliability of Multichannel Systems", SIAM Review, Vol. 22, No. 1, January 1980.

Detroit Allison, Full-Authority Fault-Tolerant Electronic Engine Control System for Variable Cycle Engines, AFWAL-TR-82-2037, Air Force Wright Aeronautical Laboratories, June 1982.

Dotson, K. J., "Examples of Nonconservatism in the CARE III Program", NASA Technical Memo 100524, NASA Langley Research Center, January 1988.

Dotson, K. J., Analysis and Testing of the SURE Program, NASA Technical Paper 2817, NASA Langley Research Center, August 1988.

Feo, Pio and K. C. Shih, "Requirement Analysis of an Intelligent, Redundant Actuation System", IEEE Transactions, August 1986.

Gai, E., J. V. Harison, and R. H. Luppold, "Reliability Analysis of a Dual Redundant Engine Controller", IEEE Transactions on Reliability, Vol. R-32, No. 1, April 1983.

Hair, K. A., "Electromechanical Actuation Reliability and Survivability", IEEE Transactions, August 1985.

Hamilton Standard, Reliability Advancement for Electronic Engine Controllers, Technical Report AFWAL-TR-80-2063, Air Force Wright Aeronautical Laboratory, August 1980.

Harschburger, H. E., "Development of Redundant Flight Control Actuation Systems for the F/A-18 Strike Fighter", Aerospace Fluid Power and Control Systems SP-554, Society of Automotive Engineers, Inc., Warrendale, PA, October 1983.

Jenney, G. D., "Research and Development of Aircraft Control Actuation Systems", Technical Report AFFDL-TR-77-91, Air Force Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, September 1977.

Jenney, G. D., et al., "Advanced Actuation Systems Development", Report WRDC-TR-89-3076 Vol. I, Dynamic Controls, Wright Research and Development Center, August 1989.

Jenney, G. D., et al., "Advanced Actuation Systems Development", Report WRDC-TR-89-3076 Vol. II, Dynamic Controls, Wright Research and Development Center, August 1989.

Jorgensen, D. E. and C. M. Hernandez, "Advanced Avionics System for an Aerospace Plane", IEEE Transactions, 1983.

Kenmir, C. V., et al., Non-Electronic Aspects of Avionic Systems Reliability, Bowty Boulton Paul Limited, England.

Kramer, M., "Development and Classification of Expert Systems for Chemical Process Fault Diagnosis", submitted to the International Conference on the Manufacturing Science and Technology of the Future, Cambridge, MA, June 1987.

Lala, J. H., et al., A Fault and Damage Tolerant Network For an Advanced Transport Aircraft, TP10-5:00.

Leonard, J.B., "A System Look at Electromechanical Actuation For Primary Flight Control", IEEE Transactions, August 1983.

McManus, B., "V-22 Tiltrotor Fly-By-Flight Control System", Eleventh European Rotorcraft Forum, City University, London, September 10-13, 1985.

Mehdi, I. S. and P. J. Leong, "Efficiently Meeting Electric Power Needs for Future Aircraft", Society of Automotive Engineers, Paper No. 859356, 1985

Murphy, M. R. "4-Valve Fly-By-Wire/Optical Control System", Technical Report NADC-85017-60, Naval Air Development Center, Warminster, PA, October 1984.

Murphy, M. R. and D. E. Haskins, "A Parallel Fly-By-Wire Helicopter Control System", Technical Report NADC-80132-60, Naval Air Development Center, Warminster, PA, March 1982.

Newirth, D. M., and C. J. Bosco, "Flight Test Reliability Demonstration of Electronic Engine Controllers", Aerospace Congress and Exposition, Los Angeles, Oct. 13-16, 1980.

Perkins, J. R. and A. J. Marek, Overview of the Advanced Aircraft Electrical System (AAES), A-7E Prototype Design, Naecon '77 Record.

Seeman, R., "An Electronic to Electrohydraulic Actuator Interface Concept for Redundant Flight Control System", Aerospace Fluid Power and Control Technologies, Society of Automotive Engineers, Las Vegas, NV, October 4-8, 1976.

Shannon, C. E., and E. F. Moore, "Reliable Circuits Using Less Reliable Relays", J. Franklin Inst., Vol. 262, pp 191-208 and 281-297, September/October 1956.

Smith, T. B., et al., "A Fault Tolerant Multiprocessor Architecture for Aircraft", CSDL Report R-1140, Vol. I, July 1976.

Swan, J. A. and R. W. Vizzini, Analytical Redundancy Design for Improved Engine Control Reliability, Final Review, AIAA-88-3176, Joint Propulsion Conference, July 11-13, 1988.

Tyron, J. G., "Quadded Logic", Redundancy Techniques for Computing Systems, Spartan Books, 1962.

Vizzini, R. W. and P. D. Toot, "Full Authority Digital Electronic Control Application to a Variable-Cycle Engine", SAE 1980 Aerospace Congress, Los Angeles, California, October 13-15, 1980.

Von Neumann, J., "Probabilistic Logics and the Synthesis of Reliable Organisms From Unreliable Components", Automata Studies, Annals of Mathematics, No. 34, pp 43-98, Princeton, 1956.

Waffner, W. D. and C. C. Chenoweth, "New Generation Flight Control Systems Hydraulic Actuation", presented at the 96th Meeting of the SAE A-6 Aerospace Fluid Power and Control Technologies Committee, Tampa, FL, May 1984.

Weinstein, W., et al., "Control Reconfigurable Combat Aircraft Development Phase I - R&D Design Evaluation", Technical Report AFWAL-TR-87-3011, Flight Dynamics Laboratory, Wright-Patterson Air Force Base, Ohio, May 1987.

Wulf, W. A., "Reliable Hardware/Software Architecture", IEEE Transactions on Software Engineering, June 1975.

ACRONYMS AND ABBREVIATIONS

AAES	Advanced Aircraft Electrical System
ac	Alternating Current
AIAA	American Institute of Aeronautics and Astronautics
APU	Auxiliary Power Unit
ARTERI	Analytical Redundancy Technology for Engine Reliability Improvement
ASEE	American Society of Electrical Engineers
BIT	Built-In Test
CSDL	Charles Stark Draper Laboratory
dc	Direct Current
DISAC	Digital Integrated Servo Actuator Controller
EMA	Electromechanical Actuator
EMAS	Electromechanical Actuator System
EMI	Electromagnetic Interference
EPR	Engine Pressure Ratio
FADEC	Full-Authority Digital Engine Controller
FBL	Fly-By-Light
FCC	Flight Control Computer
FDFM	Fault Detection and Failure Management
FICA	Failure Indication and Corrective Action
FMECA	Failure Modes and Effects Criticality Analysis
HIRF	High-Intensity Radiated Fields
HVDC	High Voltage dc
Hz	Hertz
I/O	Input/Output
LVDT	Linear Variable Differential Transformers
N_1	Low Rotor Speed
N_2	High Rotor Speed
NASA	National Aeronautics and Space Administration
PBW	Power-By-Wire
PLA	Power Level Actuator
psi	pounds per square inch
PZ	Piezoelectric
RL	Resistance/Inductance
SIAM	Society for Industrial and Applied Mathematics
SSPC	Solid-State Power Controller
TRU	Transformer Rectifier Unit
UART	Universal Asynchronous Receiver Transmitter
UPS	Uninterruptible Power Supplies
VSV	Variable Stator Vane
W/P	Fuel Flow to Burner Pressure
WFM	Main fuel metering valve actuator sensor

*U.S. GOVERNMENT PRINTING OFFICE: 1990-504-061/20155