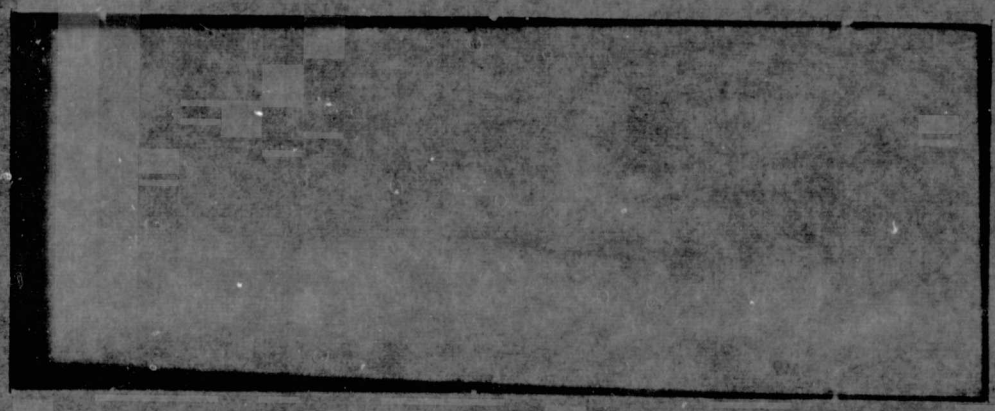


MCDONNELL DOUGLAS CORPORATION

Copy # 9

SIMULATION VERIFICATION TECHNIQUES STUDY

FINAL REPORT

MDC E 1246 31 March 1975

P. B. Schoonmaker

T. H. Wenglinski

Prepared for National Aeronautics & Space Admininstration

Johnson Space Center

Houston, Texas 77058

Submitted as DRL Line Item #4 of Contract NASA-13657

MCDONNELL DOUGLAS ASTRONAUTICS COMPANY - EAST

ABSTRACT

This report summarizes the results of the Simulation Verification Techniques Study performed for the Johnson Space Center of the National Aeronautics and Space Administration. This study consisted of two tasks. The objective of Task 1.0 was to develop techniques for simulator hardware checkout; the objective of Task 2.0, to develop techniques for simulation performance verification (validation).

The Hardware Verification Task, Task 1.0, involved definition of simulation hardware (hardware units and integrated simulator configurations), survey of current hardware self-test techniques, and definition of hardware and software techniques for checkout of simulator subsystems.

The Performance Verification Task, Task 2.0, included definition of simulation performance parameters (and "critical" performance parameters), definition of methods for establishing standards of performance (i. e., sources of reference data for validation), and definition of methods for validating performance.

Both major tasks included definition of verification software and assessment of verification data base impact.

An annotated bibliography of all documents generated during this study is provided in this report.

TABLE OF CONTENTS

LIST OF PAGES

Title Page

ii	to	iv
1-1	to	1-4
2-1	to	2-48
3-1	to	3-82
4-1	to	4-2
5-1	to	5-4

SECTION 1

INTRODUCTION

SIMULATION VERIFICATION

TECHNIQUES STUDY

(NAS9-13657)

The Simulation Verification Techniques Study was performed for the NASA Johnson Space Center by McDonnell Douglas Astronautics Company - East, Houston Operations, under contract NAS9-13657. K. L. Jordan, of the Simulation Development Branch of FSD, was NASA's Technical Monitor for the study. T. H. Wenglinski and P. B. Schoonmaker served successively as Principal Investigators for MDAC.

This report reviews the purpose of the study and our approaches to the technical tasks, and summarizes our results and conclusions.

This report consists of the vu-graphs prepared for the final presentation of the results of this study, supplemented by text which expands on the content or the conclusions derived from each vu-graph.

PROBLEM ADDRESSED; OBJECTIVE

TO USE A SPACECRAFT SIMULATOR for crew training and/or crew procedures verification, it is imperative that:

- the simulation function correctly, and
- its performance accurately represent the flight vehicle

THE OBJECTIVE OF THIS STUDY was to establish task guidelines and techniques for:

- checkout of simulation hardware
- validation of simulation performance

The basic rationale for initiating this study is shown above. There are actually two complementary problem areas addressed, both of which relate to the overall suitability of a spacecraft simulator for crew training and crew procedures development.

First, the simulation hardware must function correctly. Hardware malfunctions degrade simulator performance with respect to its training functions, which can produce "negative training". Second, fidelity of representation of the "real world" -- the in-flight operational environment -- is essential to ensure validity of training and appropriateness of developed procedures.

The objectives of the study, then, were to develop efficient means to checkout simulation hardware, thus ensuring proper operation, and to validate simulation performance, thus ensuring high fidelity. The net effect should be a substantial improvement in effectiveness of spacecraft simulators.

MAJOR STUDY TASKS

WBS 1.0: HARDWARE VERIFICATION

Develop hardware checkout techniques applicable to state-of-the-art spacecraft simulators.

WBS 2.0: PERFORMANCE VERIFICATION

Develop techniques to verify the performance (fidelity) of individual modules and the total simulation.

WBS 3.0: FINAL DOCUMENTATION

Final summary report, final oral presentation, new technology reports.

The definition of major study tasks parallels the preceding statement of problem areas and objectives for the study.

WBS 1.0, Hardware Verification, was aimed at the development of hardware checkout techniques applicable to the types of equipment to be expected on state-of-the-art spacecraft simulators. "Applicable" is perhaps the key word in this charter; in our techniques survey effort (WBS 1.2), to be described presently (see also TR-2a and TR-2 in the Bibliography), we found that very little research and development work in self-test techniques has been undertaken specifically for application to flight simulators.

WBS 2.0, Performance Verification, involved development of techniques to verify the performance (i. e., the fidelity of representation of the real world) of individual simulation modules, as well as total integrated simulations. (Later in the discussion, we shall provide a more or less formal definition of what we mean by a "module"; for the present, any intuitive notion of what a module is will suffice.)

The outputs of WBS 3.0 include this report, the final presentation on which this report is based, and the usual new technology reports.

SCHEDULE OF STUDY DELIVERABLES

REPORT	CALENDAR DATE: 1973 J A S O N D	1974 J F M A M J J A S O N D	1975 J F M
WBS 1.0			
TR-1: Simulation Hardware Definition TR-2: Self-Test Techniques Survey TR-3: Integrated System Self-Test DRL-2: Self-Test Hardware Design & Techniques Report	O	O O O Δ	
WBS 2.0			
TR-4: Module Perf. Param. & Standards TR-5: Subsystem Simulation Validation TR-6: Sim. Integration & Validation* DRL-3: Simulation Performance Validation Techniques Document		O	O O Δ
WBS 3.0			
DRL-4: Final Report DRL-5,6: New Technology Reports			Δ Δ

* Incorporated into DRL-3
Δ Denotes CEI report.
O Denotes added task reports.

The schedule of documents delivered under this contract is shown above. Individual documents are briefly described in the Bibliography, Section 5.

On this schedule, the triangles represent contracted end items, which are identified by their DRL (Data Requirements List) line item numbers and title. The circles represent additional delivered Task Reports (TR's), not required by the contract, which were generated to provide NASA with complete and current information on the results of individual study subtasks.

Task Report #2, the techniques survey report, was generated in two versions, denoted TR-2a and TR-2 in the Bibliography. TR-2a described company-funded research done before initiation of the contracted study, and was delivered at contract go-ahead; TR-2 covered further survey efforts undertaken during the contract, and was restricted to techniques which appeared directly applicable to simulation verification problems.

SECTION 2

HARDWARE VERIFICATION (TASK 1.0)

SIMULATION VERIFICATION TECHNIQUES STUDY

TASK 1.0

HARDWARE VERIFICATION

In this section, we discuss our approach and the results of the hardware verification task, which was the first task performed during the study.

PRECEDING PAGE BLANK NOT FILMED

HARDWARE VERIFICATION TASK 1.0

OBJECTIVES:

"... DEFINE SOFTWARE AND HARDWARE TECHNIQUES FOR CHECKING THE OPERATIONAL STATUS OF ALL SIMULATOR EQUIPMENT."

SCOPE:

IDENTIFY/DEFINE HARDWARE AND SOFTWARE REQUIRED FOR FOLLOWING:

- O VERIFY PROPER OPERATION OF ALL DATA PATHS, END TO END -- "READINESS TESTING"
- O ISOLATE FAILURES TO LINE REPLACEABLE UNIT (LRU) -- "FAULT ISOLATION"
- O ACCUMULATE DATA FOR IDENTIFICATION OF INCIPIENT FAILURES -- "INCIPIENT FAULT DETECTION"
- O PRIMARY EMPHASIS IS ON AUTOMATIC TEST TECHNIQUES
- O ACCEPTANCE TESTING IS NOT OF INTEREST

2.1 OBJECTIVES AND SCOPE

The primary objective of the Hardware Verification Task was to derive hardware and software techniques for implementing self test capabilities in an advanced manned spacecraft training simulator. This task was not concerned with the initial acceptance testing that a piece of equipment is first subjected to when delivered by the contractor. Rather, the testing of concern was that required on a daily basis for purposes of assuring the proper operation of the simulator hardware before beginning training activities.

The tests of interest on an operational, periodic basis may be further divided into several categories, readiness tests, fault isolation tests, and incipient fault detection tests. First, the proper operation, or "readiness", of the series arrangements of equipment terminating at the host computer can in many instances be tested for operational adequacy on an end-to-end basis. For example, a meter deflection of a certain amount can be commanded by software in the host computer. The proper deflection of the meter verifies that all of the hardware in the data path from the computer to the meter are functioning satisfactorily. Failure to function properly creates a need for an additional test, that is, a fault isolation test, to determine where in the string of hardware the equipment is defective. If there are so many meters in the simulator that meter failures become a regular occurence, then it may become desireable to collect meter performance data and to predict when an instrument is likely to fail in order that it may be serviced during a regular maintainance period, rather than allowing a failure to interupt training operations. These last test techniques, we have called incipient fault detection techniques.

HARDWARE VERIFICATION TASK 1.0

GROUND RULES:

- DESIGN NOT TO ENDANGER PERSONNEL OR EQUIPMENT
- MALFUNCTION OF SELF-CHECK HARDWARE SHALL NOT HINDER NORMAL OPERATION
- DESIGN SHALL MINIMIZE HUMAN INTERVENTION
- MAXIMUM USE OF AVAILABLE HARDWARE AND SOFTWARE
- MINIMIZE REQUIREMENTS ON COMPUTER RESOURCES
- MINIMIZE CHECKOUT TIME

Certain specific ground rules were specified by the statement of work which insured that the safety of personnel and equipment were not compromised for test purposes. Efficient utilization of existing equipment to implement the self test techniques was also required. However, the need to minimize the workload imposed on the simulator equipment by the test techniques was also considered and test approaches are recommended that minimize the impact of test operations.

HARDWARE VERIFICATION TASK I.0

SPECIFIED RESULTS:

- O SIMULATION SELF TEST HARDWARE DESIGNS AND TECHNIQUES REPORT

SPECIFIED CONTENT:

- O DEFINITION OF SIMULATION HARDWARE
- O DESCRIPTION OF CURRENT HARDWARE SELF TEST SYSTEMS
- O DESCRIPTION OF HARDWARE AND SOFTWARE SELF TEST TECHNIQUES INCLUDING THE FOLLOWING
  - OO DEFINITION OF A COMPLETE SET OF PARAMETERS WHICH CHARACTERIZE ALL SYSTEMS, SUBSYSTEMS AND HARDWARE UNITS
  - OO DRAWINGS, SCHEMATICS AND WRITTEN DESCRIPTIONS OF TECHNIQUES FOR ACQUIRING CHARACTERISTIC PARAMETER DATA FROM THE SIMULATION EQUIPMENT
  - OO DESCRIPTION OF THE SOFTWARE REQUIRED TO PROCESS AND EVALUATE DATA
  - OO DESCRIPTION OF THE DATA BASE REQUIREMENTS

The documentation requirements for this Task, as spelled out in the statement of work, were quite explicit. The documentation of the techniques recommended for each of the simulator subsystems includes schematics, circuit diagrams, software flow charts and accompanying text. We also formulated an integrated test software configuration and assessed the impact on a data base of program software and data requirements.

HARDWARE VERIFICATION SUBTASKS

1.1 DEFINITION OF SIMULATION HARDWARE

1.2 SURVEY OF CURRENT HARDWARE SELF-TEST TECHNIQUES

1.3 DEFINITION OF HARDWARE AND SOFTWARE TECHNIQUES FOR SIMULATOR CHECKOUT

The Hardware Verification Task was divided into three basic subtasks. The results of each of these subtasks were documented in interim reports. At the conclusion of the Task, the information in the three interim reports was revised and consolidated into one large report which addressed the documentation requirements previously noted.

HARDWARE VERIFICATION TASK 1.0

SIMULATOR HARDWARE (SUBTASK 1.1)	HARDWARE SELF TEST TECHNIQUES (SUBTASK 1.2)		
	READINESS TESTS	FAULT ISOLATION	INCIPIENT FAULT DETECTION
COMPUTERS DCE CONTROLS & DISPLAYS VISUAL SIMULATION o IMAGE GENERATION o IMAGE TRANSMISSION o DISPLAY EQUIP. MOTION BASE MISCELLANEOUS o AURAL CUES o ETC.	SUBTASK 1.3 DEFINITION OF HARDWARE AND SOFTWARE TECHNIQUES		

The relationship of the results of the three subtasks is indicated above. The hardware definition subtask (Subtask 1.1) identified the hardware devices expected in future manned spacecraft training simulators. The self test techniques survey task (Subtask 1.2) identified techniques and concepts applicable to testing of simulator hardware. The actual definition of hardware and software techniques, (Subtask 1.3) took the results of both of these subtasks and addressed, one by one, the problems of implementing self test techniques for each of the major simulator subsystems. This latter subtask constituted the bulk of the effort accomplished for this Task.

RESULTS OF

SUBTASK 1.1 - DEFINITION OF SIMULATION HARDWARE

- REFERENCE CONFIGURATION
- HARDWARE COMPONENT IDENTIFICATION/DATA

2.2 SUBTASK 1.1 - DEFINITION OF SIMULATION HARDWARE

The hardware definition activity accomplished several essential preliminary objectives. First, it identified the simulator system and subsystem configurations anticipated in future simulators and, therefore, the configurations with which we were to be concerned. Secondly, it identified the particular types of hardware anticipated for future NASA training simulators and enabled us to establish a base of information for future use. Examples of hardware types are the hydraulic, synergistic motion bases; the RGB color TV-model visual simulations; crew station equipment typical of the Shuttle orbiter.

SUBTASK 1.1

REFERENCE CONFIGURATION DEFINITION RESULTS

- DEFINED SIMULATOR REFERENCE CONFIGURATION
  - SHUTTLE RELATED
  - USE OF EXISTING STATE OF THE ART
  - BASED ON TRAINING REQUIREMENTS
- COMPILED SIMULATOR COMPONENT DATA
  - BASED ON SHUTTLE SIMULATOR DOCUMENTS
  - PREVIOUS SPACECRAFT TRAINING SIMULATORS
  - MILITARY AIRCRAFT SIMULATORS
  - COMMERCIAL AIRCRAFT SIMULATORS
  - ENGINEERING SIMULATORS

The reference simulator and simulator subsystem configurations were based on anticipated requirements for Shuttle training simulators. However, other simulator users and applications were surveyed in the process of identifying the specific types of equipment anticipated. This review of other simulator activities confirmed the fact that the manned spacecraft training simulators are at the state of the art in simulator design.

SUBTASK 1.2 – SURVEY OF CURRENT HARDWARE SELF–TEST TECHNIQUES

- CURRENT SIMULATOR SELF–TEST SYSTEMS
- BASIC SELF–TEST TECHNIQUES

2.3 SUBTASK 1.2 - SURVEY OF CURRENT HARDWARE SELF TEST TECHNIQUES

There were essentially two phases to the survey of current self test techniques. The first of these consisted of surveying simulator users around the country to establish their status with respect to implementation of self test techniques on simulators specifically. The second phase of the survey activity was concerned with identifying generic techniques, available from other sources but suitable for training simulator testing.

CURRENT SIMULATOR SELF-TEST SYSTEMS

NAME	USER	EQUIPMENT TESTED	TEST FUNCTIONS	TEST MODE
PSALT	JSC/ESD			
FLINT		DCE	READINESS TEST FAULT ISOLATION	AUTOMATIC INTERACTIVE
TAMS		ANALOG COMPUTER	READINESS TEST	AUTOMATIC
SWORD		STATUS WORD LINK	READINESS/FAULT ISOLATION	AUTOMATIC
CDT		PROGRAMABLE CLOCK	READINESS FAULT ISOLATION	AUTOMATIC
EQPCK		DISPLAY HARDWARE THRU DCE	READINESS	INTERACTIVE
ADC		A/D AND D/A CALIBRATION	READINESS	AUTOMATIC
SAFE	ARC	MOTION BASE	READINESS	SEMI-AUTOMATIC
	NTEC	VISUAL SIMULATION DYNAMIC RESPONSE	READINESS	SEMI-AUTOMATIC

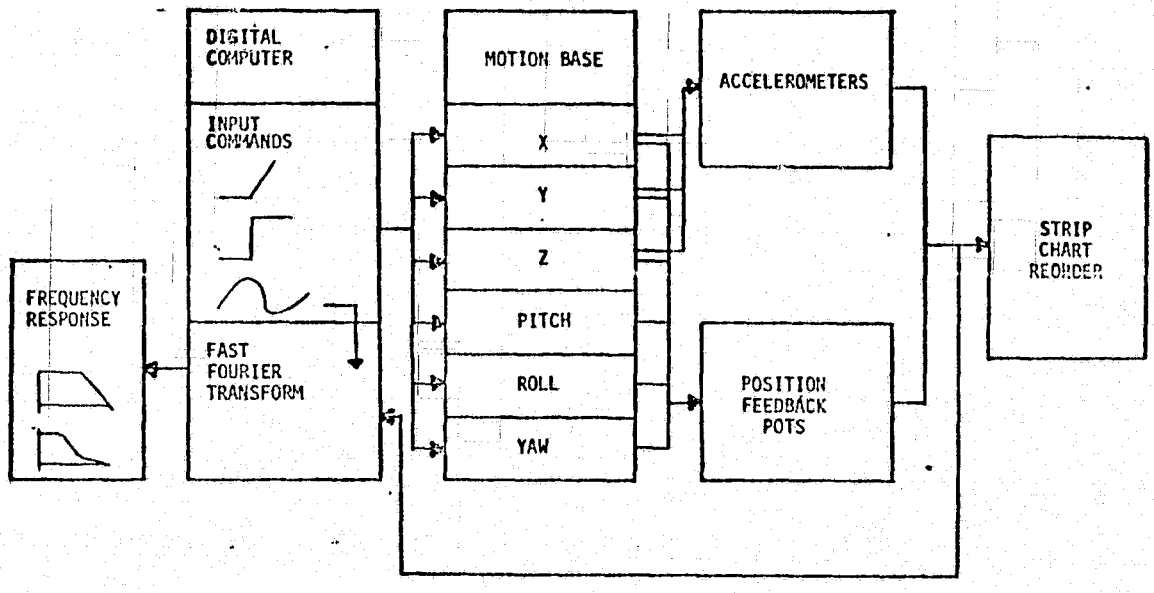
2.3.1 Current Simulator Self Test Systems

The largest amount of existing simulator self test techniques were found on the procedures simulators at the Johnson Space Center. These tests addressed the data conversion equipment, associated analog computers, the programmable clock and the displays. The tests for the displays are interactive rather than fully automatic as considered for this study.

In addition, the Ames Research Center has developed and applied a program called SAFE, Six Axis Frequency Evaluation, for the testing of their electro-mechanical motion base equipment. Langley has obtained a copy of this program and adopted it for use with hydraulic synergistic motion bases. Listings of the software for both programs as well as additional documentation have been supplied to the Technical Monitor.

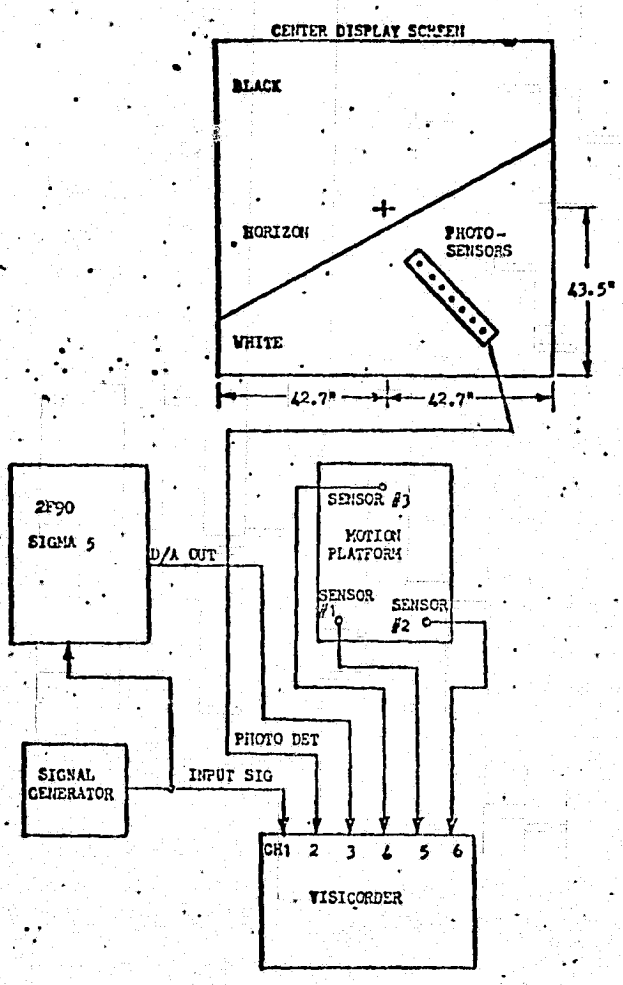
The Naval Training Equipment Center at Orlando, Florida has been doing development work toward checkout of the vehicle dynamic simulation as percieved from the visual simulation. This effort considers the recording and evaluation of the apparent horizon motion when a vehicle transient response is induced by a step control input. This effort does not represent a hardware test technique of the type we are concerned with for this study.

ARC'S SAFE PROGRAM
(SIX AXIS FREQUENCY EVALUATION)



Ames Research Centers' SAFE rpogram essentially generates test commands for various types of frequency response tests in the host computer. The response of the motion base is sensed either by accelerometers or by sampling the feedback voltage from the position feedback pots. The performance of the motion base is assessed by manual examination of the recordings. In terms of this study objectives, this is a semi-automatic technique.

SIMULATOR VERIFICATION BY FREQUENCY RESPONSE TESTING (NTEC)



The dynamic response test, as applied by the Naval Training Equipment Center, introduces an array of sensors (photo-transistors) specifically for test purposes. This enables automatic data acquisition, but the data is again recorded on strip charts and evaluated manually which falls short of this study objective to look at automatic techniques.

For the SVTS study, we specifically addressed techniques which not only generated necessary test signals, but also acquired and analysed the test data in order to give the operator the final answer automatically. Typical outputs to the operator might be as follows:

"METER NUMBER XXX IS OUT OF CALIBRATION"

OR

"MOTION BASE HYDRAULIC LINE FILTER NO. XXX
IS CAUSING EXCESSIVE PRESSURE DROP."

SUBTASK 1.2 - SURVEY OF CURRENT HARDWARE SELF-TEST TECHNIQUES

BASIC SELF-TEST TECHNIQUES

- O FAULT DETECTION
- O FAULT ISOLATION
- O INCIPIENT FAULT DETECTION

SUBSIDIARY TECHNIQUES

- O SIGNAL GENERATION
- O DATA ACQUISITION
- O DATA PROCESSING (REDUCTION)

2.3.2 Self Test Techniques

Self test techniques have been identified for implementing tests to accomplish the objectives previously noted, readiness testing, fault isolation testing, and incipient fault detection. In implementing tests which address these objectives, it is also necessary to accomplish certain fundamental or generic testing functions which include test signal generation, test data acquisition and test data processing. In this particular study, the data processing of interest is that processing which is required to achieve the objectives of the tests, starting from measurements of parameters that are available for sampling during the test.

The organization of the techniques information in the manner noted above has helped to minimize the repetion of certain material or ideas that are common to more than one-simulator subsystem.

SUBTASK I.2 - SURVEY OF CURRENT HARDWARE SELF TEST TECHNIQUES

BASIC SELF TEST TECHNIQUES

	FAULT DETECTION	FAULT ISOLATION
DIGITAL ELECTRONICS	o END-TO-END TESTING.	o FUNCTIONAL UNIT TESTING
ANALOG	o END-TO-END TESTING	o LRU LEVEL TESTING
	o EVALUATION OF CHARACTERISTIC PARAMETERS	o SWITCHING FOR SIGNAL INSERTION AND DATA ACQUISITION

For both digital and analog electronic equipment, the difference between readiness testing and fault isolation testing is basically the hardware unit level at which the test is conducted. Obviously, for a readiness test we are primarily concerned with verifying that all of the equipment in a series or string is performing properly. If the series of equipment is deficient in performance, then we need to isolate or determine which unit in the series is deficient. With electronic equipment, this is commonly accomplished by providing the necessary switching to make the appropriate selections.

For digital equipment, proper performance is verified by checking the bit state of each bit in a register, memory cell, etc. For analog equipment, more complex performance criteria are required and are referred to in our reports as "characteristic parameters." These are factors such as signal to noise level, linearity, gain, frequency response, etc.

SUBTASK I.2 - SURVEY OF CURRENT HARDWARE SELF TEST TECHNIQUES (CONTINUED)

BASIC SELF TEST TECHNIQUES

	FAULT DETECTION	FAULT ISOLATION
ELECTRO-MECHANICAL	o STATIC RESPONSE	o TRANSFER FUNCTION DEFINITION
	o DYNAMIC RESPONSE (IMPUL SE, STEP FREQUENCY)	o FREQUENCY RESPONSE CORRELATION

For electro-mechanical equipment, the characteristic parameters are those associated with basic control theory. These include static response, amplitude and phase response versus frequency, etc, However for these systems it is often not feasible to sample response at the unit level that we might wish for fault isolation. Consequently, the more sophisticated processing techniques are of more essential interest here; two of these have been noted above. Faults in electromechanical systems maybe isolated by performing a frequency response test and then performing an analysis of this response to define the current transfer function of the system. Changes in values of the frequencies at which there are inflection points in the linearized frequency response can be analyzed to determine what system components may have changed value. An alternate, more direct approach is to simply generate by simulation the frequency response characteristics associated with various types of failure and then correlate the test results with these patterns.

SUBTASK I.2 - SURVEY OF CURRENT HARDWARE SELF TEST TECHNIQUES

BASIC SELF TEST TECHNIQUES

INCIPIENT FAULT DETECTION TECHNIQUES

- O OVERSTRESS TESTING - EXCESSIVE POWER, FREQUENCY, MAGNITUDE, ETC.
  - REVEAL FAULTS APPARENT OUTSIDE NOMINAL PERFORMANCE LIMITS
- O MARGINAL PERFORMANCE TESTING - LOW POWER LEVELS OR SIGNAL LEVELS
  - REVEAL IRREGULARITIES SUCH AS FRICTION OR NOISE
- O GRAY AREA PERFORMANCE - COMPARE PERFORMANCE WITH DEGRADATION BOUNDARY
- O DEGRADATION RATE ANALYSIS - SAVE PERFORMANCE DATA FOR PERFORMANCE PREDICTION

Incipient fault detection techniques require substantially different procedures or analyses than the readiness and fault isolation tests. The first two of the techniques noted above, overstress testing and marginal performance testing, require unique tests for the incipient fault detection function. They also incur the risk of inducing failures during their execution. Consequently these tests are better suited for maintainance activites where they can still serve the purpose of incipient fault detection.

The last two techniques, "gray area" performance and degradation rate analysis, both provide an approach for detecting incipient faults using the data from either readiness or fault isolation tests. The gray area technique requires the definition of a marginal performance level for the units being tested. This marginal performance level is represented by an additional tolerance level on a characteristic parameter which must be stored in the data base. When the unit performance degrades to this level, a warning can be printed out by the computer doing the processing that, within a short period, the unit performance will become unacceptable.

In contrast, the degradation rate analysis technique requires no additional data input by the developer; rather it accumulates test results from day to day in order to perform regression analyses and consequently make a prediction of when unit performance is likely to exceed tolerable levels.

SUBTASK I.2 - SURVEY OF CURRENT HARDWARE SELF TEST TECHNIQUES

SUBSIDIARY TECHNIQUES

SIGNAL GENERATION:

o SIMPLE LOW FREQUENCIES	-	SOFTWARE TECHNIQUES
	-	PROGRAMABLE S/G
o COMPOUND LOW FREQUENCIES	-	SOFTWARE TECHNIQUES
	-	PROGRAMABLE S/G (?)
o SIMPLE HIGH FREQUENCIES	-	PROGRAMABLE S/G
o COMPOUND HIGH FREQUENCIES	-	PROGRAMABLE S/G WITH NOISE GENERATION CAPABILITY
o COMPLEX TEST SIGNALS (TV)	-	SPECIAL PURPOSE HARDWARE

Test signals must be generated for any tests to be performed for any of the simulator subsystems considered during this study. Binary test patterns for testing digital equipment can readily be generated by the various computer elements to be found in the simulator. However analog signals and some special test signals require other specific sources.

Low frequency analog signals for testing meters, the motion base, visual system servo drives and other low frequency equipment can be generated without much complication by the various digital computer devices available. These signals can be composite signals which are either sums of sinusoids or pseudorandom noise signals having frequency components up to the basic bandwidth limitations of the digital device or its output data channel.

Higher frequency analog signals with various wave forms such as sine, square or ramp shapes, as well as broad band noise signals can be readily generated by programmable signal generators. These generators are available for a very modest cost and their output signals can be commanded by digital commands.

In addition, testing of high performance video circuits in the visual simulation equipment requires the use of special purpose TV signal generators whose operation can be controlled remotely by provision of proper switching facilities; this will be discussed later (Section 2.4.4).

SUBTASK I.2 - SURVEY OF CURRENT HARDWARE SELF TEST TECHNIQUES

SUBSIDIARY TECHNIQUES

DATA ACQUISITION:

- o AVAILABLE SIGNALS - POSITION FEEDBACK
- o SENSORS - PHOTO TRANSISTORS
- o SWITCHING TECHNIQUES - SOLID STATE SWITCHES

A primary motivation for implementing self test techniques on a simulator is likely to be the need to ensure that the trainee is only exposed to proper operation of the controls and displays. Therefore, the test equipment must provide test sensors that can directly measure those effects that the trainee is expected to observe. The strength of this requirement establishes, for a self test system, the scope of the data acquisition problem.

The operation of electronic devices can be verified by sampling the input and output signals to these devices. Servo systems can be checked by sampling the position position feedback signals required for their operation. However, the performance of devices which the trainee observes visually can only be tested by provision of suitable light sensing devices for data acquisition. One device whose applications we explored at considerable length was the photo transistor.

TYPICAL DATA ON PHOTOTRANSISTORS

- o MECHANICAL
  - LENGTH: .1 - .25 INCHES
  - DIAMETER : .063 - .25 INCHES
  - ANGULAR RESPONSE : 3 - 45 DEGREES
- o ELECTRICAL
  - LIGHT CURRENT : .5 - 8.0 mA
  - DARK CURRENT : 25 - 100 nA
  - RISE TIME : 4 - 80 MICRO Sec.
- o THERMAL
  - OPERATING TEMPERATURE : -65 - +125 Deg. C.
  - STORAGE TEMPERATURE : -65 - + 125 Deg. C.

NOTE: ALSO AVAILABLE AS LIGHT SOURCE/SENSOR ASSEMBLIES

Typical properties of photo transistor devices are summarized above. These devices may be used to sense the light generated by external sources such as the cabin illumination or the light at the face of a CRT. They are, however, also available with their own light emitters as single emitter detector units. In the latter configuration, they may be used to sense the presence of a reflecting mark or device such as the back of a meter needle or a reflective mark on a servo driven device.

SUBTASK I.2 - SURVEY OF CURRENT HARDWARE SELF TEST TECHNIQUES

SUBSIDIARY TECHNIQUES

DATA PROCESSING (REDUCTION):

- o FREQUENCY RESPONSE FROM SAMPLED DYNAMIC TEST DATA
  - - FAST FOURIER TRANSFORM TECHNIQUES
- o FREQUENCY RESPONSE FROM LOGGED OPERATIONAL DATA
  - - TRANSFORM TECHNIQUES
  - - NOISE INJECTION TECHNIQUES
  - - CORRELATION PROCESSING

The fault isolation techniques that were identified for electromechanical systems involved the definition of the frequency response of the devices. Frequency response can be measured by simply driving the device with one frequency at a time and measuring the amplitude and phase response. However the overall test time can be reduced by using composite test signals which incorporate signals of the full spectrum of frequencies of interest. For a linear system, the response will of course be the sum of the responses to the component frequencies. This response may be disassembled into the components associated with each frequency by analysing the signal with fast Fourier Transform techniques. These techniques are in common use and copies of the required software have been supplied to the Technical Monitor.

In addition, it may be desireable to determine the frequency response of some devices by logging operational performance data and processing this data off line. A number of techniques have been identified for this purpose and flow charted. These techniques may be of particular interest for monitoring the performance of the motion base system during normal training operations and reduce the need for exercising the hardware additionally for test purposes.

SUBTASK I.3 DEFINITION OF HARDWARE AND SOFTWARE TECHNIQUES FOR SIMULATOR CHECKOUT

- o COMPUTERS
- o DCE
- o CONTROLS AND DISPLAYS
- o VISUAL SIMULATION
  - oo IMAGE GENERATION
  - oo IMAGE TRANSMISSION
  - oo DISPLAY EQUIPMENT
- o MOTION BASE
- o MISCELLANEOUS
  - oo AURAL CUES
  - oo POWER SUPPLIES

2.4 SUBTASK 1.3 - DEFINITION OF HARDWARE AND SOFTWARE TECHNIQUES FOR SIMULATOR CHECKOUT

The techniques discussed in the previous section were applied to development of self test techniques for typical components and configurations of each of the major simulator subsystems noted above.

Self test techniques for computers primarily addressed the interface computers, the flight computers and other digital computer equipment expected in fugure training simulators. The large computer systems usually incorporated as host computers were not addressed specifically, because of the amount of support available for these systems from their manufactures.

The overall results obtained for each of these typical simulator subsystems are reviewed in the following sections.

RESULTS OF HARDWARE VERIFICATION ANALYSES

- DEFINITION OF SIMULATOR SUBSYSTEM CONFIGURATION
- IDENTIFICATION OF LRU'S
- IDENTIFICATION OF CHARACTERISTIC PARAMETERS
- IDENTIFICATION OF FAILURE MODES
- DESCRIPTION OF FAILURE SYMPTOMS
- EVALUATION OF ALTERNATIVE TEST CONCEPTS
- DEFINITION OF HARDWARE AND SOFTWARE FOR MOST REASONABLE APPROACHES

The scope of the analyses completed for each of the simulator subsystems is indicated above. The subsystem configurations were analyzed in terms of the nature of the hardware and its structure and organization in order to identify the Least Replaceable Units (LRU's). The LRU's represent the level to which we addressed our fault isolation concepts. The characteristic parameters necessary for evaluating the performance of the simulator subsystems were identified. Failure modes and their associated symptoms were considered to assure the adequacy of the characteristic parameters and the data acquisition techniques for fault detection. Finally, the alternative approaches for implementing self test were evaluated and hardware and software configurations were selected and documented by means of hardware schematics and software flow charts.

SELF TEST TECHNIQUES - ANCILLARY COMPUTERS AND INTERFACE EQUIPMENT

SCOPE:

- O FLIGHT HARDWARE INTERFACE DEVICE
- O FLIGHT COMPUTER
- O OTHER MINICOMPUTERS

KEY PROBLEM AREAS:

- O FUNCTIONAL TESTING OF ALL BASIC OPERATIONS
- O SOFTWARE REQUIRED

RECOMMENDATION:

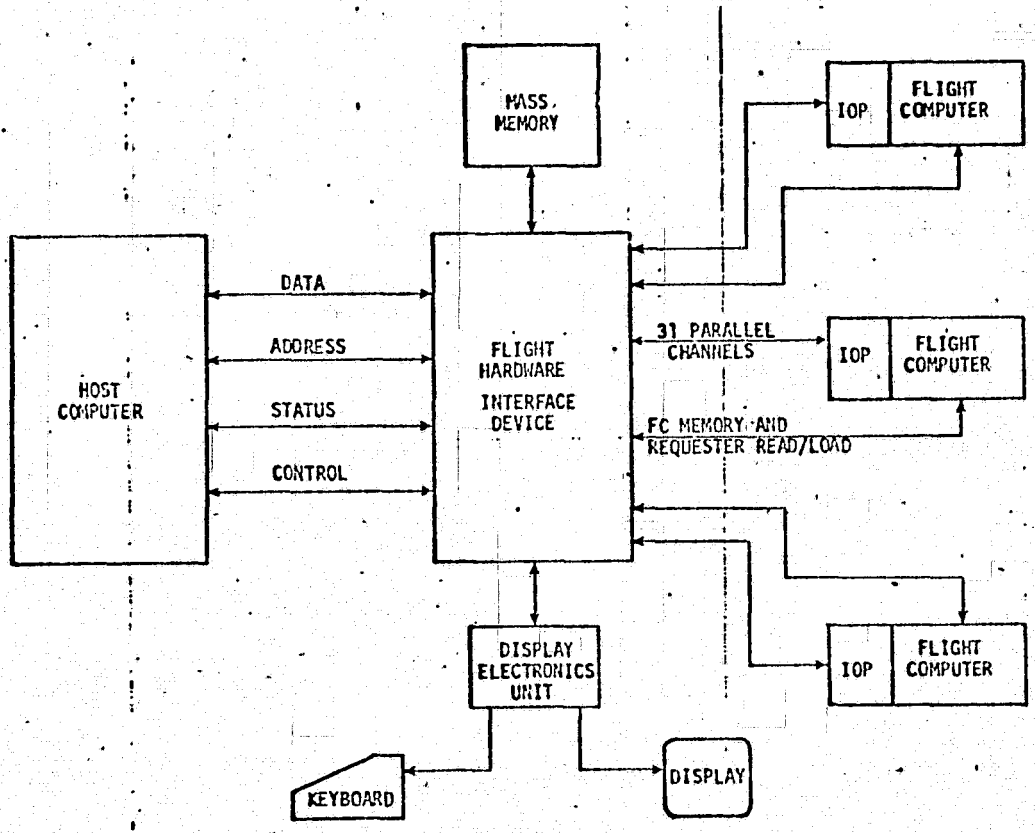
- O MAXIMIZE USE OF VENDOR-SUPPLIED DIAGNOSTIC SOFTWARE (INCLUDE DIAGNOSTIC REQUIREMENTS IN PROCUREMENT SPECIFICATIONS)

2.4.1 Computers

Large scale computers systems selected by the Johnson Space Center for host computers for manned spacecraft training simulators are well supported by their manufacturers with diagnostic software and diagnostic procedures as well as basic readiness tests. During this study we consequently concentrated on the smaller computer elements and the unique digital hardware that might be anticipated for future simulators. This equipment includes the flight computers that are interfaced to the host, the special interface equipment that is required for this purpose, and other small computers that may be introduced to service specific simulator subsystems such as the visuals or the motion base.

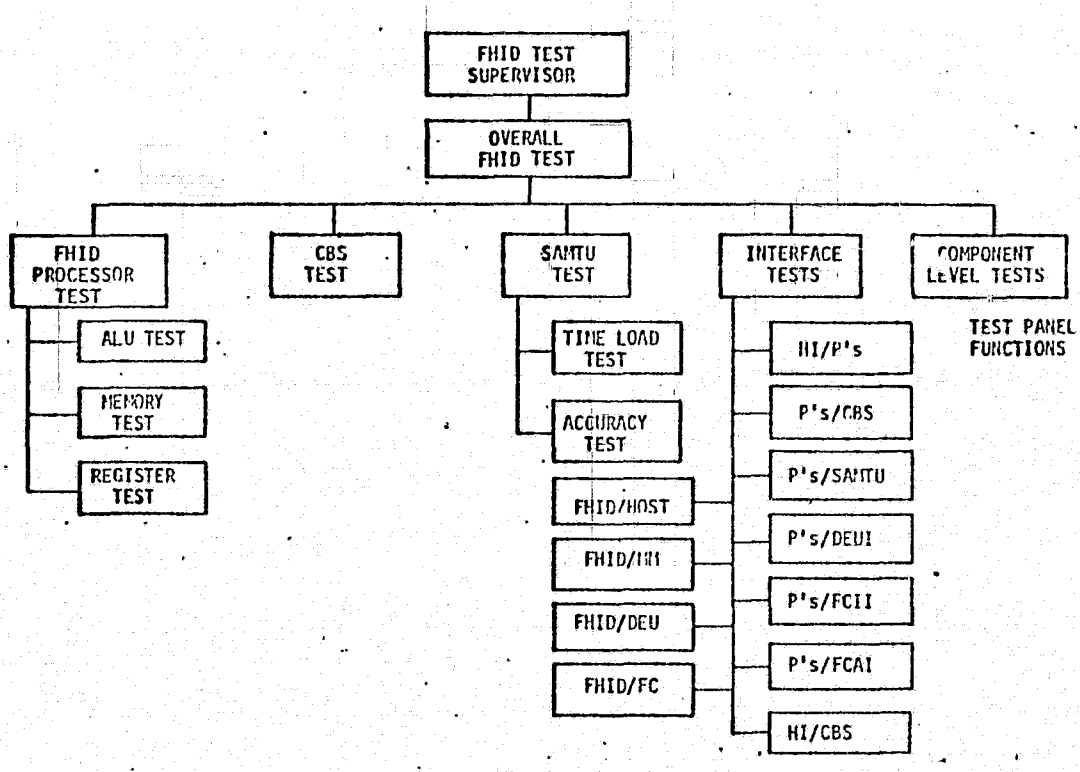
The key problem with respect to testing of this equipment is the identification of the basic functional operations to be tested and the description of the software required to implement the tests. This information then enables specification of adequate test support software as part of the procurement of this equipment.

FLIGHT HARDWARE INTERFACE DEVICE - EXTERNAL INTERFACES



For purposes of this summary, we will describe only the nature of the tests required and the associated software requirements for testing a flight hardware interface device. The configuration chosen as representative is shown above. (The number of flight computers shown is only typical and doesn't modify the functional test requirements.) The flight computers shown are typical of the Shuttle flight hardware, as is the mass memory and the display electronics. The interface device is itself a computer since it must necessarily contain buffer memory and processor devices.

FLIGHT HARDWARE INTERFACE DEVICE - HOST EXECUTED TEST SOFTWARE

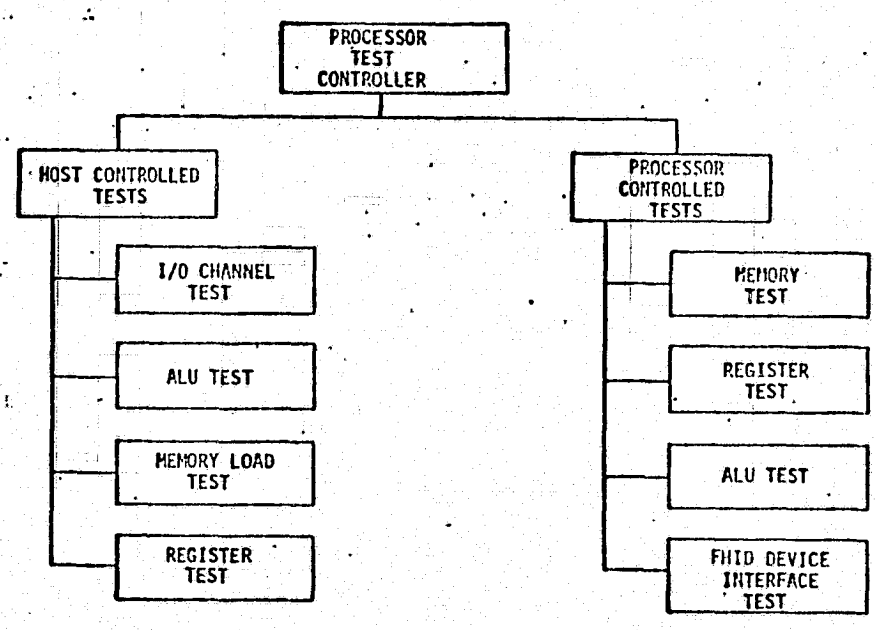


The test and associated software required for testing the interface device may be divided into test software that is executed by the host computer and the test software that is executed by the interface device processors. The structure above represents the host-executed test software.

These tests are distributed between the following major functions:

- o FHID Processor Tests
- o Central Buffer Storage (CBS) Tests
- o Simulated Avionics Master Timing Unit (SAMTU) Tests
- o Interface Tests
- o Component Level Tests (Test Panel Controlled) for Fault Isolation

FLIGHT HARDWARE INTERFACE DEVICE - INTERFACE EXECUTED TEST SOFTWARE



The software executed by the interface device under host computer control verifies the proper operation of functions internal to that device. The initial host controlled tests shown on the left verify proper operation of the I/O channel from the host to the interface, as well as basic operation of the arithmetic logic unit (ALU), registers, and the ability to load and read memory. This ensures that the processor is capable of accepting and executing the more comprehensive test software load shown on the right. The ALU tests verify proper and timely execution of the basic hardware instructions.

SELF TEST TECHNIQUES - DATA CONVERSION EQUIPMENT

KEY PROBLEM AREAS:

- CONFIGURING SELF-TEST FOR MAXIMUM USE OF EXISTING HARDWARE (MINIMUM ADDITIONAL HARDWARE)
- MINIMIZE IMPACT OF SELF TEST HARDWARE FAILURE ON NOMINAL OPERATIONS
- FAULT ISOLATION TO THE LRU LEVEL FOR BOTH DIGITAL AND ANALOG ELEMENTS
- SWITCHING TECHNOLOGY EVALUATION

2.4.2 Data Conversion Equipment

Testing of data conversion equipment is the area in which there is the most past experience at the Johnson Space Center on the procedures simulators. In addressing the key problems noted above, we were most concerned with evaluating the latest switching techniques available for implementing the DCE tests in a fully automatic manner, and the magnitude of the DCE testing problem as we anticipated it for the Shuttle training simulators.

DCE CONCLUSIONS/RECOMMENDATIONS

- DCE SELF-TEST ANALYSIS IS RELATIVELY STRAIGHT FORWARD BECAUSE INCREASED USE OF MICRO-ELECTRONICS INCREASES THE LEVEL AT WHICH LRU'S ARE DEFINED. (I.E. FAULT ISOLATION IS NOT REQUIRED TO A VERY LOW LEVEL.)

- SOLID STATE SWITCH TECHNOLOGY IS NOW COST COMPETITIVE AND OFFERS THE FOLLOWING ADVANTAGES:
  - VERY HIGH RELIABILITY
  - SMALL PACKAGING REQUIREMENTS
  - SMALL POWER
- TEST SOFTWARE REQUIREMENTS FOR BOTH SIGNAL GENERATION AND FAULT ISOLATION ARE VERY STRAIGHTFORWARD.

The increased use of micro electronics on DCE to be procured in the future impacts self test requirements by vastly reducing the number of components in the system to which faults may need to be isolated. The use of solid state switching and past software experience should enable implementation of self test capability to an appropriate level for a spedific simulator.

SELF TEST TECHNIQUES - DISPLAYS

- o ANALOG
  - METERS WITH GALVANOMETER MOVEMENTS
  - DC SERVO DRIVEN METERS
  - SYNCHRO/RESOLVER METERS
- o BILEVEL
  - LIGHTED INDICATORS
  - ELECTROMECHANICAL FLAGS
- o VIDEO
  - CRT GRAPHICS DISPLAYS

2.4.3 Controls and Displays

Testing of control devices requires primarily the design of a concealed actuation device since some electrical signal is nominally modulated or interrupted by the basic control operation and is therefore available for sampling. The actuation capability must be concealed to maintain the fidelity of the crew station hardware. We identified several techniques for actuating switches since the capability of automatically setting switches seemed to have synergistic additional benefits for initializing or resetting the simulator. Continuous controls require the design of a servo system and concealment is more peculiar to the particular control.

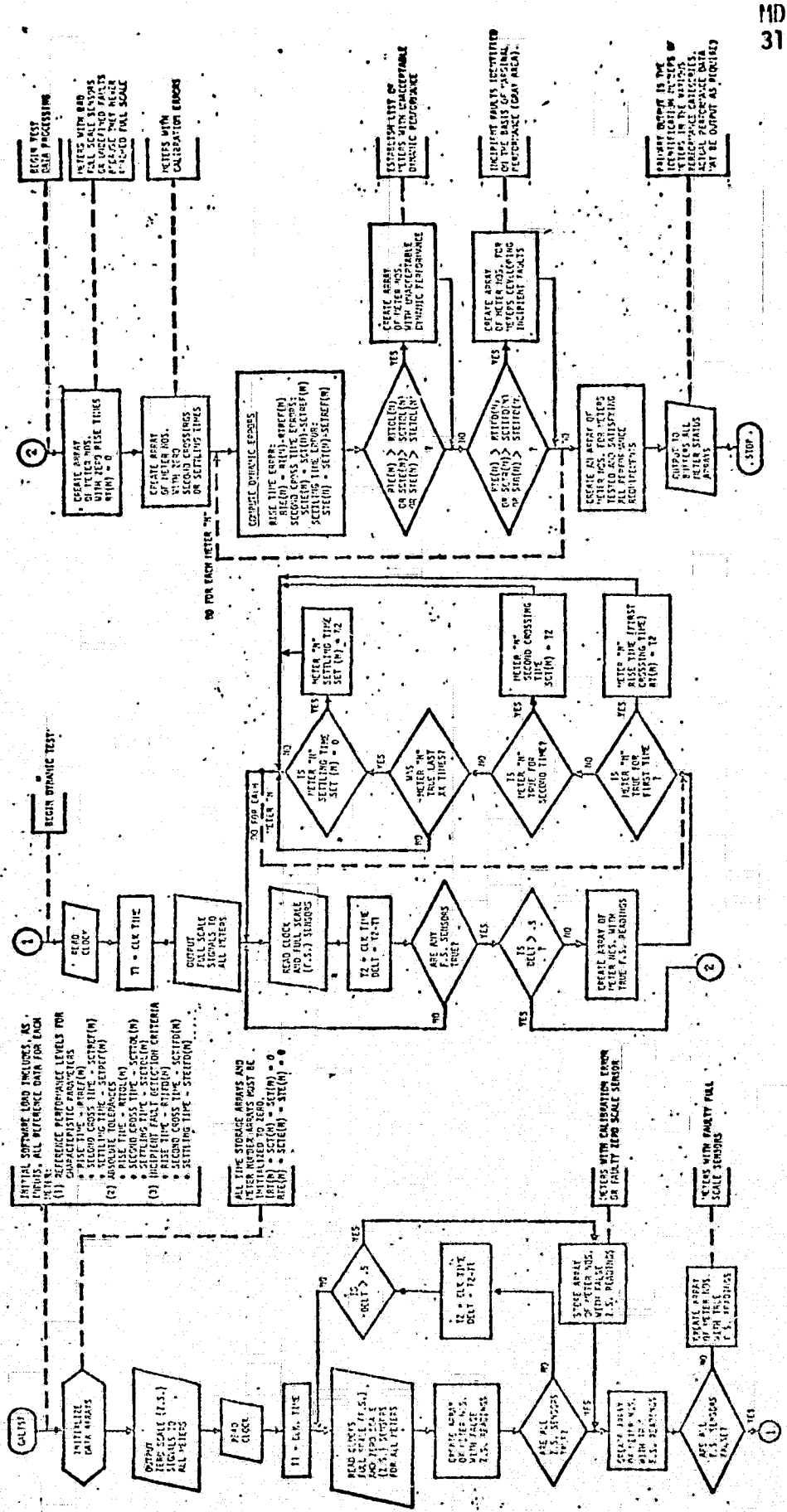
A larger variety of test problems exists for the various display devices. We considered the classes of displays noted above. Although we identified in considerable detail the functional requirements and scope of test software required for testing CRT graphics displays, these test facilities should be specified as part of the hardware procurement. For this summary we will review our recommended test techniques for the instruments and indicators noted above.

SELF TEST TECHNIQUES - DISPLAYS

DEVICES	DATA ACQUISITION	TEST REQUIREMENTS
GALVANOMETERS	PHOTOTRANSISTORS	STATIC/DYNAMIC RESPONSE
SERVO METERS	POSITION FEEDBACK	STATIC/DYNAMIC RESPONSE
SYNCHRO/RESOLVERS	POSITION FEEDBACK	STATIC/DYNAMIC RESPONSE
LIGHTED INDICATORS	CURRENT FLOW	ON/OFF CURRENT LEVELS
FLAGS	ELECTRICAL CONTINUITY	POSITION

The techniques we identified for testing the various display devices are summarized above. For these displays, sensing of the devices response to an input signal is the key test problem. For servo meters and synchro/resolver devices we are able to sample the position feedback signals to obtain continuous position data. For galvanometers, the use of phototransistors to sense descrete meter positions is recommended. Dynamic response can be obtained from discrete position information (two locations on the scale) if the nominal response time from one position to the other has been established. The software required for testing galvonometers with data from two meter position points available is shown on the following page.

We also defined techniques for checking either current flow or electrical continuity for purposes of testing mechanical and lighted indicators respectively.



VISUAL SIMULATION SUBSYSTEMS

- SCENE GENERATION EQUIPMENT
- IMAGE TRANSMISSION SYSTEM
- VISUAL DISPLAYS

2.4.4 Visual Simulation Equipment

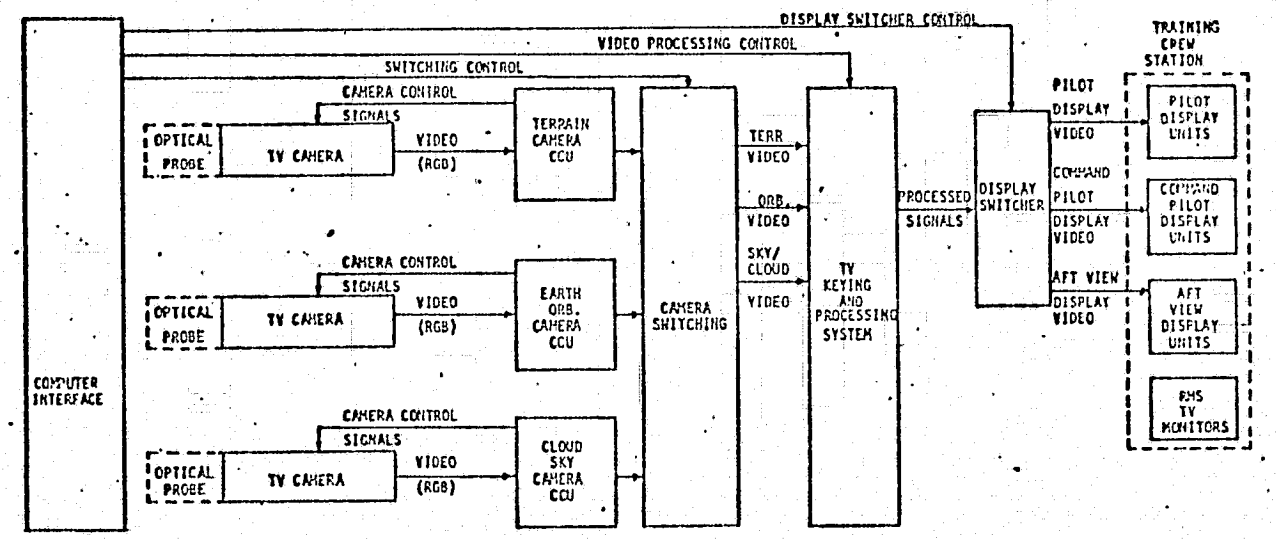
The visual simulation equipment for which self test techniques were derived was divided into three basic categories. The scene generation equipment, as we defined it, included primarily the models and associated servo drives including servo drives for camera positioning.

The image transmission system was defined to include all of the color TV hardware with the exception of the CRT's for the displays. The CRT's and their associated optics were grouped with the visual display equipment.

Although we defined hardware and software requirements for testing all three categories of visual equipment, we will primarily address the color TV Subsystems requirements for purposes of this summary report.

SELF TEST TECHNIQUES - VISUAL SIMULATION SUBSYSTEM

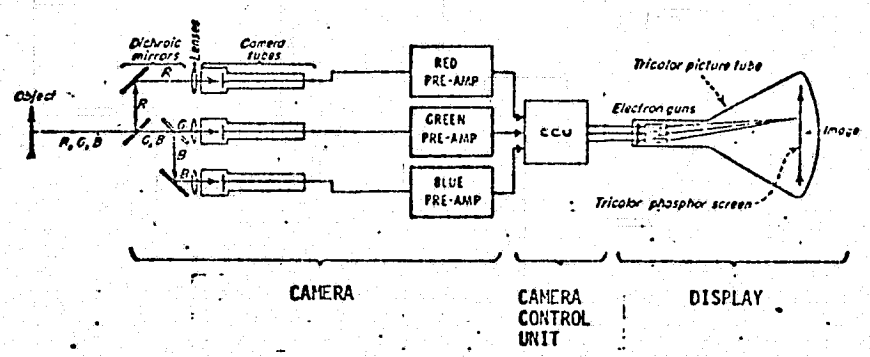
SIMULATOR TV SYSTEM SCHEMATIC



The general configuration of the TV system which we addressed is based in part on the results of the Space Shuttle Visual Simulation System Design Study recently conducted by McDonnell Douglas Electronics Company and is shown above. The camera switching for scene selection, the video processing and the switching of processed signals to appropriate displays are accomplished under computer control.

SELF TEST TECHNIQUES - VISUAL DISPLAY SUBSYSTEM

TYPICAL RGB SYSTEM SCHEMATIC

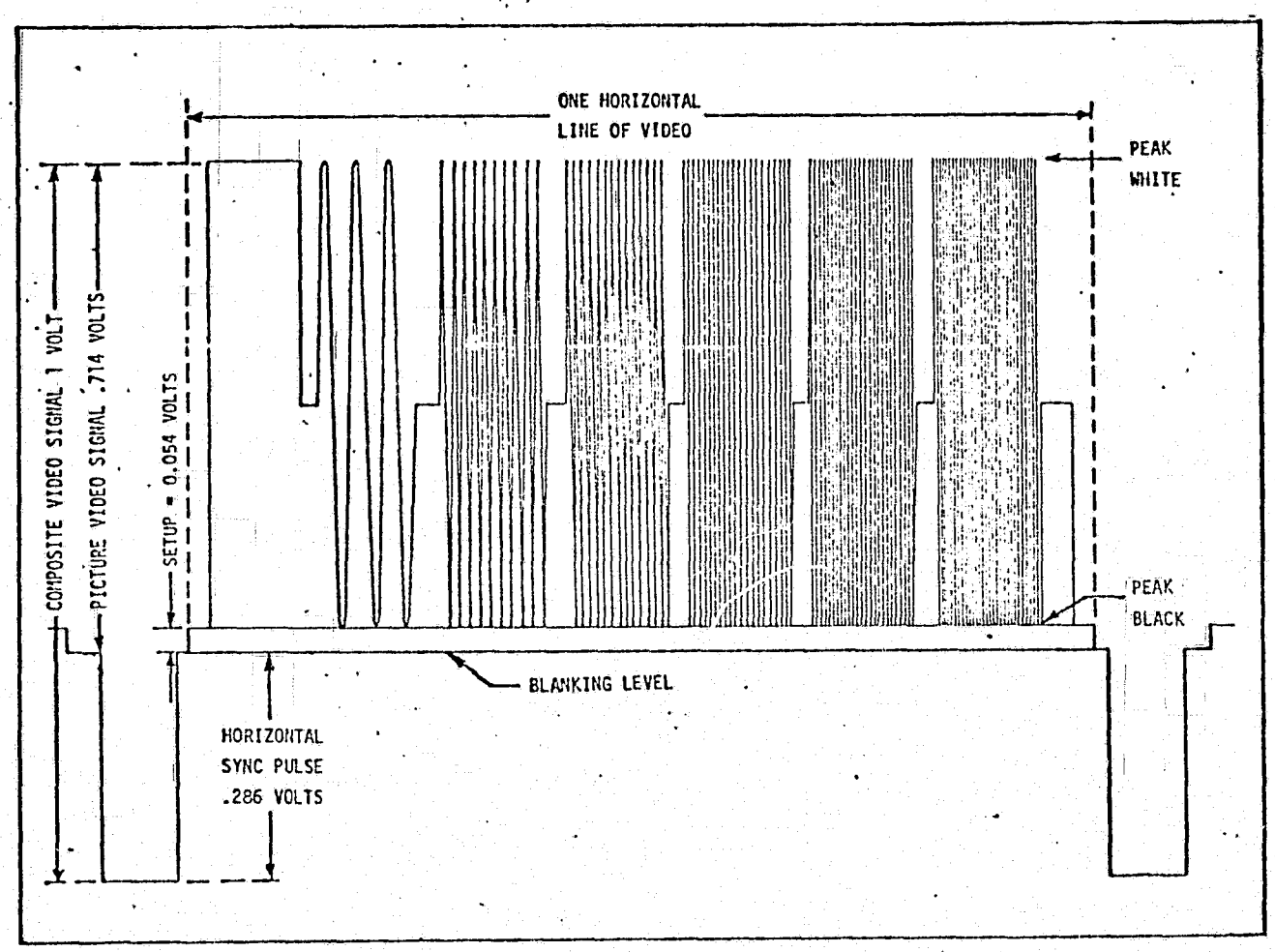


We assumed that the color images would be transmitted by a conventional RGB (Red/Green/Blue) color system. In this system, the three primary color images are separated by dichroic filters before they reach the camera tubes. Each color TV channes is effectively a separate black and white TV chain. No coding is introduced since the three channels do not have to be compressed into one channel for broadcast. The test techniques applied to each channel are identical and are the same as might be used for a black and white TV system since the video signals have no color qualities. The signal from each color channel is directed to the appropriate gun in the CRT and the images are recombined and color restored at the CRT phosphor screen.

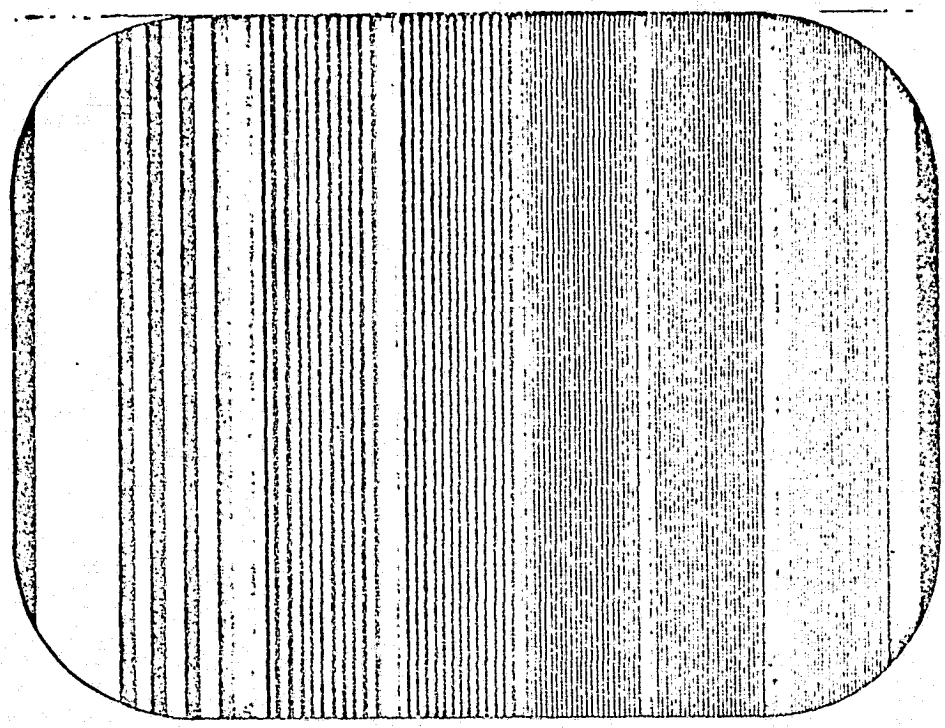
CHARACTERISTIC PARAMETERS FOR TV READINESS AND FAULT ISOLATION TESTS

	Characteristic Parameters / Equipment Tested	(1) Voltages (P/P) Composite Video	(1) Voltages (P/P) Video	(2) Setup	(3) Resolution	(4) Gamma	(5) Signal Noise Ratio	(6) Differential Gain	(7) Low Frequency Streaking	(8) High Frequency Ringing	(9) Linearity	(10) Convergence or Registration
Readiness Tests	Optical End to Electrical End	X	X	X	X	X	X	X	X	X	X	X
	Electrical End to Electrical End	X	X	X	X	X	X	X	X	X		
Fault Isolation Tests (LRU'S)	Camera (Tube Output)		X		X	X	X	X	X	X	X	X
	Camera (Pre AMP Input)	X	X	X	X	X	X	X	X	X		
	CCU Camera Control Unit	X	X	X	X	X	X	X	X	X		
	Keying	X	X	X	X	X	X	X	X	X		
	Display Monitor (at grid)	X	X	X	X	X	X	X	X	X		
	Display Monitor (optical)				X	X			X	X	X	X

In order to define self test techniques for testing the TV-system electronics we first identified the characteristic parameters which could be evaluated for the complete strings of TV electronics (readiness testing) and the characteristic parameters which would require measurement for specific units in the chain (fault isolation). The intent, in this case, is to accomplish fault isolation by looking at the performance of individual units. The correlation or applicability of characteristic parameters for the various units is shown above.



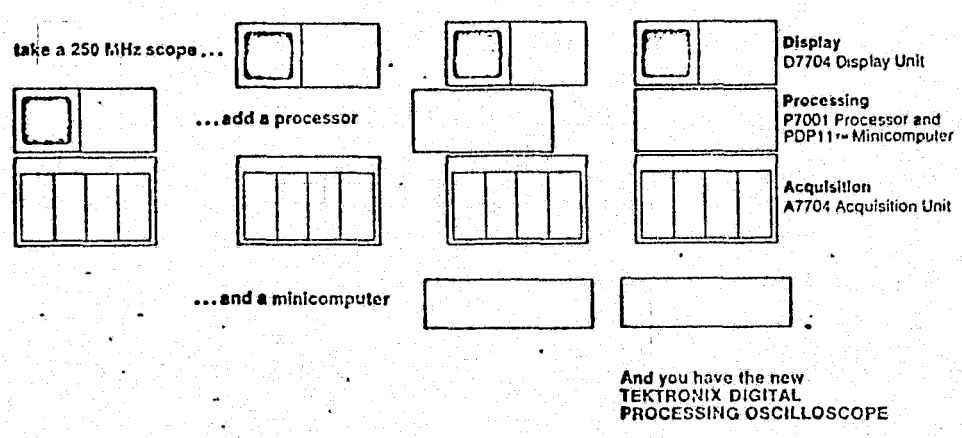
In order to implement the tests required to measure these characteristic parameters, it was necessary for us to identify the test signals from which each of these parameters might be defined. The signal shown above is the voltage time history for one horizontal line of the multi-burst TV test signal. This is the image that would be observed on a high frequency oscilloscope. This test signal enables measurement of the first four characteristic parameters; in the preceding table; the composite video signal peak-to-peak voltage, the picture video signal peak-to-peak voltage, the set-up voltage, and the resolution. The last parameter, the resolution, is determined by measuring the picture video signal peak-to-peak voltage for each of the frequency bursts. A reduction or loss of voltage amplitude at the higher frequencies corresponds to a loss of horizontal resolution.



The image shown above is a somewhat crude representation of the image that would appear on the TV screen while a multi-burst test signal was being transmitted. The narrower vertical lines correspond to the higher frequency burst on the previous page. We identified additional test signals or patterns for use in determining all of the characteristic parameters.

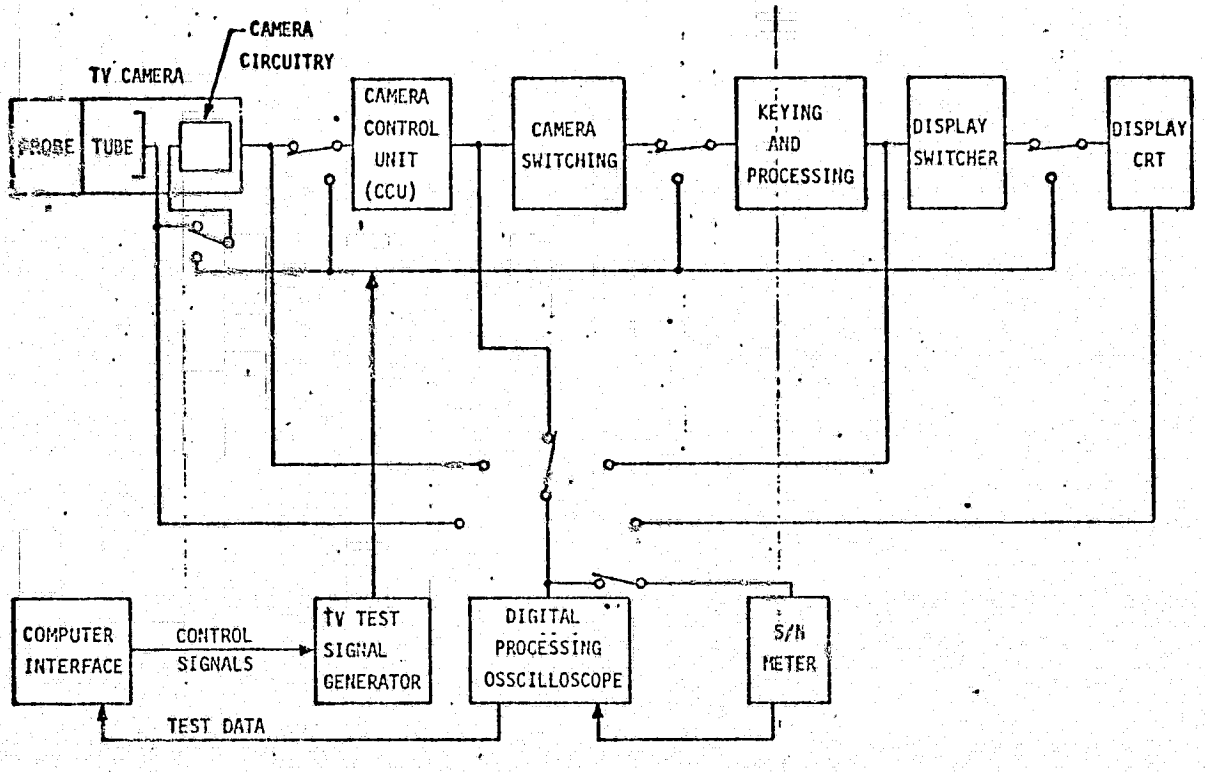
SELF TEST TECHNIQUES - VISUAL SIMULATION SUBSYSTEM

DIGITAL PROCESSING OSCILLOSCOPE



After we identified the characteristic parameters and the test signals that could be used for their measurement, we addressed the problem of finding a means to accomplish the necessary tests in an automatic manner. That is, the manual recording of voltages from an oscilloscope display was not an acceptable techniques; Fortunately, we found a new product on the market, the Tektronix Digital Processing Oscilloscope (DPO), shown above. The digital processing oscilloscope makes it possible, for the first time, digitize and store for further processing, any test signal that can be displayed on an oscilloscope. The figure above illustrates schematically how this is accomplished. The processor which is inserted between the oscilloscope display and the standard data acquisition modules, provides a buffer storage for sampled data and also provides an interface with a PDP minicomputer. Software in the minicomputer can control all oscilloscope operations as well as retrieve and process sampled data.

TV SELF TEST HARDWARE SCHEMATIC



The manner in which this oscilloscope would be introduced into a self test system is shown above for one chain of TV components. A fairly standard TV test signal generator can be used for signal generation by providing switching for turning on power and for selection of test signals. Additional switching can establish signal insertion points and test signal sampling points for end to end readiness testing or for single unit fault isolation testing. A standard Rhode and Schwart signal/noise meter can also be switched in and its output digitized by the oscilloscope. All of the data can be brought back from the oscilloscope to the computer interface for evaluation, storage, or processing. We have also defined a software flow for implementing all of the test required to define and assess the characteristic parameters that we previously noted.

VISUAL SIMULATION SUBSYSTEMS - CONCLUSION AND RECOMMENDATIONS

- DIGITAL PROCESSING OSCILLOSCOPE - SAMPLE AND DIGITIZE TEST RESPONSE
- RHODE AND SCHWARTZ SIGNAL/NOISE METER - STANDARDIZE THIS PERFORMANCE MEASUREMENT

The key problem area with respect to TV system self test was automating the measurement of the characteristic parameters which have been used for years for TV system testing. The recommended solution is the use of a digital processing oscilloscope, such as that available from Tektronix, although others are expected on the market shortly. The use of the Rhode and Schwartz Signal/Noise meter provides a standardized means for signal to noise ratio measurement.

Automated testing of TV systems in this manner is a new idea and this is also a new application for the digital recording oscilloscope which has only been available for a short period.

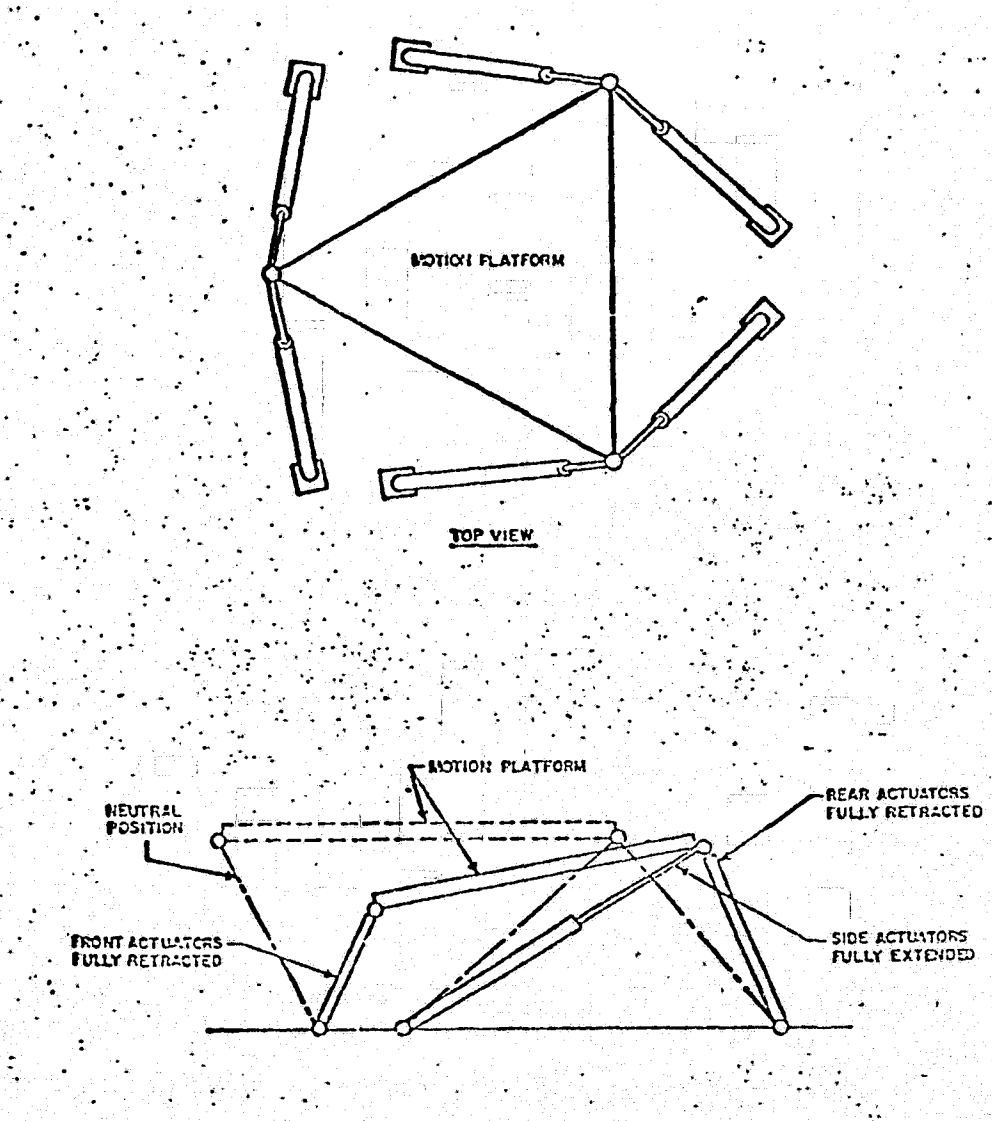
SELF TEST TECHNIQUES - MOTION BASE SYSTEM

- SYSTEM CONFIGURATION - SYNERGISTIC HYDRAULIC SYSTEM
- LRU'S - COMPONENTS OF ACTUATOR SYSTEMS AND HYDRAULICS
- CHARACTERISTIC PARAMETERS - STATIC/DYNAMIC RESPONSE
  - QUIESCENT CHARACTERISTICS
- RESULTS OF ANALYSIS
  - DEFINITION OF FAILURE MODES
  - DESCRIPTION OF FAILURE SYMPTOMS
  - DEFINITION OF ALTERNATIVE TEST CONCEPTS
  - DEFINITION OF HARDWARE AND SOFTWARE REQUIRED FOR SELF TEST

2.4.5 Motion Base System

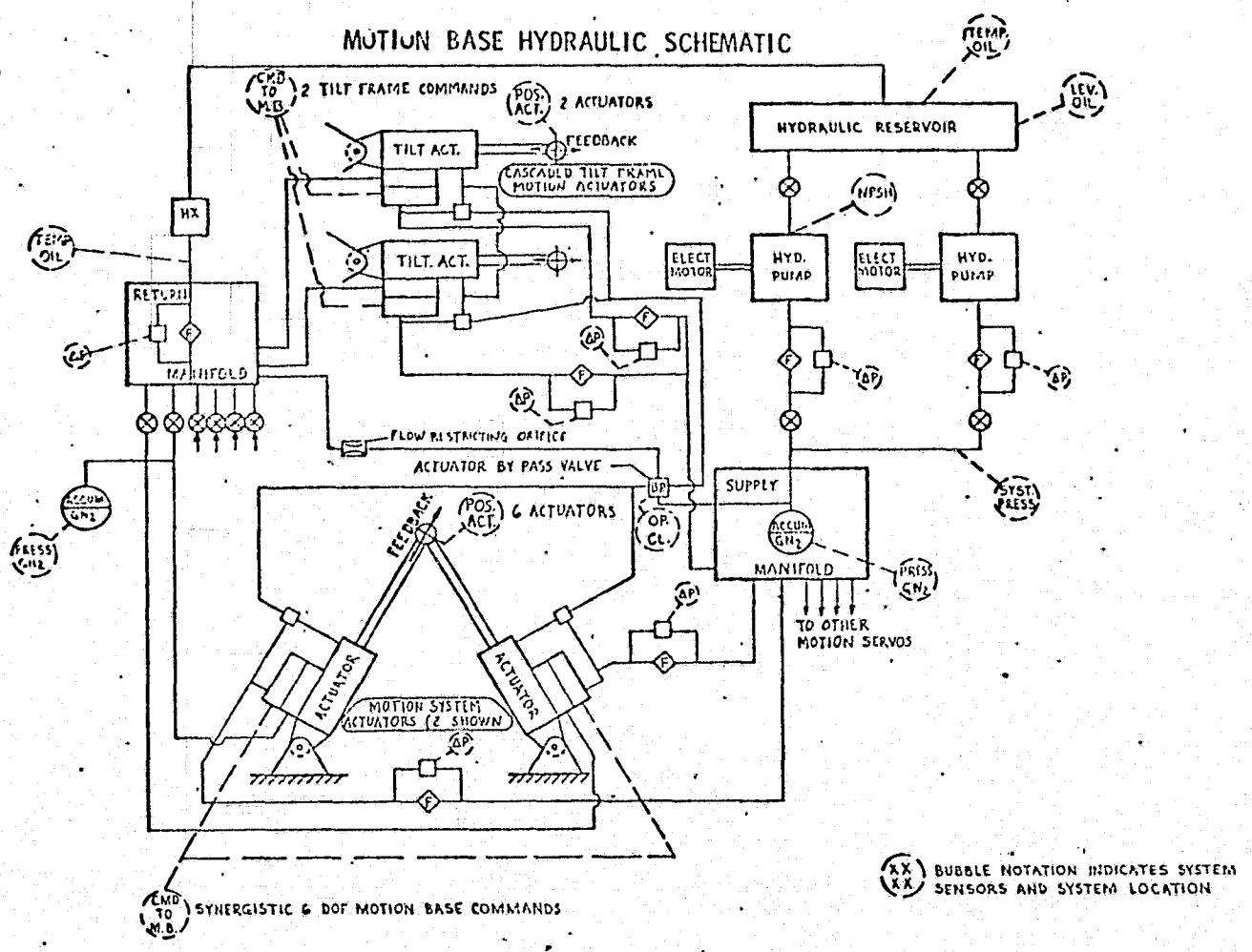
The motion base systems analyses addressed the self test requirements for synergistic, hydraulic motion base systems. The Least Replaceable Units for these systems are the various components of the hydraulic actuator systems including miscellaneous hardware such as filters, sensors, etc. The actuator systems are basically closed-loop servo systems, and their dynamic characteristic parameters are those associated with any control system. These are parameters such as characteristic time, frequency response, damping ratio, etc.

The results obtained from our analyses included a definition of failure modes, a description of failure symptoms, and the definition of various test procedures including static power off and power on tests as well as dynamic response tests. Description of available sensor hardware for motion base instrumentation and the software required for implementing the suggested tests were included in the final report for this Task.



MOTION BASE GEOMETRY

The basic kinematics of the synergistic, hydraulic motion base system are shown above. The motion of the platform is totally determined by the position commands transmitted to the three pairs of hydraulic actuators. There is no one-to-one relationship between the axes associated with aircraft/spacecraft (such as roll, yaw, or pitch) and any one degree of freedom of the motion base system.



The details of the hydraulics typically associated with each pair of actuators is shown schematically above. The dashed circles indicate parameters that are of interest for monitoring and the point in the system where they may be sensed or measured.

MOTION BASE MALFUNCTION MODES AND SYMPTOMS

MALFUNCTION MODE	SYMPTOMS	
	EXTERNAL	INTERNAL
ELECT. OPEN CIRCUIT	UNABLE TO ACTIVATE PART OR ALL OF THE COMPONENTS OF THE M.B.	NO CONTINUITY.
ELECT. SHORT CIRCUIT	AS ABOVE	LOW RESISTANCE.
COMPUTER AND OR DATA CONVERSION EQUIPMENT (DCE)	ERRATIC SYSTEM PERFORMANCE	INVALID COMMAND LOOPS AND/OR RESPONSE.
HYDRAULIC OIL RESERVOIR LEVEL LOW	LOW HYDRAULIC PRESSURE, PUMP NOISY	LOW RESERVOIR OIL LEVEL CAUSING HYDRAULIC PUMP TO CAVITATE.
HYDRAULIC PUMP PRESSURE OUTPUT LOW	LOW HYDRAULIC PRESSURE	PUMP INTERNAL LEAKAGE, PUMP DAMAGE, OR LOW RPM
ACTUATOR MOTION ERRATIC	MOTION BASE MOVEMENT JERKY OR LOCKS UP AND WILL NOT COMPLETE COMMANDED MOVEMENT.	ACTUATOR PISTON ROD BENT OR BEARING SEIZED.
ELECTRO HYDRAULIC SERVO VALVE	RESPONSE (OUTPUT) LAGS COMMAND (INPUT) IN AMPLITUDE IN VELOCITY IN FREQUENCY	MOVEMENT RESTRICTION (M.B. TABLE OR ACTUATORS). FLOW RESTRICTION OR INSUFFICIENT SUPPLY LINE ACCUMULATOR PRE-PRESS. MALFUNCTIONING SERVO VALVE
GAS IN HYDRAULIC SYSTEM	MUSHY SYSTEM	RESPONSE LAG
FILTER BLOCKED	HIGH FILTER DIFFERENTIAL PRESSURE	CONTAMINATED FILTER (DIRTY HYDRAULIC OIL)
FILTER OPEN	LOW FILTER DIFFERENTIAL PRESSURE	FILTER UNIT FLOW THROUGH (NOT FILTERING)
ACTUATOR POSITION FEEDBACK	DELAY IN RESPONSE TO COMMAND, FOR ACTUATOR POSITION.	EXCESSIVE ERROR (LAG) ACTUATOR POSITION TO COMMAND.
SUPPLY LINE ACCUM LOW PRE-CHARGE	FAILURE OF ACTUATORS TO REACH HIGH DEMAND COMMANDED AMPLITUDE.	INSUFFICIENT SUPPLY LINE FLUID
RETURN LINE ACCUM LOW PRE-CHARGE	ERRATIC (NOT SMOOTH) M.B. MOVEMENT	RETURN LINE OIL SURGES NOT DAMPED

The basic motion base malfunction modes and their associated symptoms are summarized above. The nature of its symptoms provides the basic indication of the type of test that must be implemented to check for that particular type of failure.

MODE MALFUNCTION	CHECKOUT TESTS					
	STATIC POWER OFF CHECKOUT	STATIC POWER ON CHECKOUT	MAXIMUM MISSION PROFILE (STEP INPUT)	TRAINING MISSION PROFILE +10% (STEP INPUT)	MOTION BASE LIMIT PROFILE (STEP INPUT)	FREQUENCY RESPONSE
ELECTRICAL OPEN CIRCUIT		✓				✓
ELECTRICAL SHORT CIRCUIT		✓				✓
COMPUTER AND/OR DATA CONVERSION EQUIPMENT(DCE)			✓	✓	✓	✓
HYDRAULIC OIL RESERVOIR LEVEL LOW	✓	✓				✓
HYDRAULIC PUMP PRESSURE OUPUT LOW		✓				✓
ACTUATOR MOTION ERRATIC			✓	✓	✓	✓
ELECTRO HYDRAULIC SERVO VALVE			✓	✓	✓	✓
GAS IN HYDRAULIC SYSTEM			✓	✓	✓	✓
FILTER BLOCKED		✓				✓
FILTER OPEN		✓				✓
ACTUATOR POSITION FEEDBACK			✓	✓	✓	✓
SUPPLY LINE ACCUMULATOR LOW PRE-CHARGE	✓					✓
RETURN LINE ACCUMULATOR LOW PRE-CHARGE	✓					✓

The various test techniques that may be applied to detect the failures noted on the previous page are summarized above. It is logical to assume that the static tests would most assuredly be implemented whether automatically or in a manual mode by the operator. The appropriate dynamic tests are then required to check the remaining failure modes. It should be noted that use of a frequency response test does not necessarily require that the motion base be exercised and subjected to sinusoidal motions of various frequencies. The techniques previously mentioned for determing frequency response by logging both the commands and response of dynamic systems may have their most profitable application to motion base systems. These techniques not only enable frequent checks during the course of each days training sessions, but they also minimize the wear and tear on motion base mounted equipment.

MOTION BASE SUBSYSIEM - CONCLUSIONS AND RECOMMENDATIONS

- MAXIMUM USE OF INSTRUMENTATION FOR IMPLEMENTING STATIC CHECKS AND MONITORING SYSTEM BEHAVIOUR
- APPLICATION OF OPERATIONAL DATA LOGGING AND ANALYSIS FOR DEFINITION OF SYSTEM FREQUENCY RESPONSE
- AS A CONSEQUENCE OF ABOVE, MINIMIZATION OF WEAR AND TEAR ON MOTION BASE MOUNTED SYSTEMS BY REDUCTION OF MOTION BASE OPERATION TIME
- THE SELF TEST HARDWARE AND SOFTWARE REQUIRED FOR IMPLEMENTING THE ABOVE RECOMMENDATIONS HAVE BEEN IDENTIFIED AND DOCUMENTED IN THE FINAL REPORT

The final report for the Hardware Verification Task, Task 1.0, presents in substantially more detail the descriptions of the hardware and software required for implementing the various tests mentioned. The conclusions recommended above are based on a broader, more comprehensive analysis of notion base test requirements than we have seen from any other source.

SELF TEST TECHNIQUES - MISCELLANEOUS EQUIPMENT

SCOPE:

- O AURAL CUES
- O POWER SUPPLY
- O EXTERNAL CLOCKS AND TIMING

KEY PROBLEM AREA:

- O MEASUREMENT AND EVALUATION OF AURAL CUE SIGNALS

RECOMMENDATIONS:

- O USE FAST FOURIER TRANSFORM TO DEFINE CUE SIGNATURE IN TERMS OF COMPONENT FREQUENCIES.

2.4.6 Miscellaneous Simulator Equipment

During the SVTS study we identified self test techniques for miscellaneous simulator equipment, including the simulator power supplies, the external clocks and timing functions, and the aural cues simulation.

Power supply checks are straight forward using simple switching techniques to sample necessary voltages and route them to the computer through the available data conversion equipment.

Testing of external clocks and timing equipment is performed regularly by the minicomputer manufacturers, for example, and has also been addressed here at the Johnson Space Center.

The Fourier transform techniques previously noted in the techniques survey section provide a unique but easily implemented tool for verifying the proper performance of the aural cues simulation. The basic sounds which are generated can be identified and evaluated for testing purposes in terms of their spectral signatures. These can be readily assessed by sampling the signal and using the Fast Fourier transform to establish its component frequencies and their phase and amplitude characteristics. This "signature" is readily compared with a reference signature the storage of which requires only limited memory.

HARDWARE VERIFICATION TASK I.0

SUMMARY OF RECOMMENDATIONS

- o DIGITAL COMPUTERS/INTERFACES - MAXIMUM USE OF VENDOR SOFTWARE
- o DCE - SOLID STATE SWITCHING FOR LOOP CLOSURES
- o CONTROLS AND DISPLAYS - PHOTO TRANSISTORS FOR POSITION SENSING
- o VISUAL SIMULATION - DIGITAL RECORDING OSCILLOSCOPE FOR DATA ACQUISITION/PROCESSING
- o MOTION SYSTEM - INSTRUMENTATION MONITORING AND USE OF LOGGED OPERATIONAL DATA
- o AURAL CUES - FAST FOURIER TRANSFORM FOR SIGNATURE DEFINITION

2.5 CONCLUSIONS AND RECOMMENDATIONS, TASK 1.0

In concluding the Hardware Verification Task review, it is appropriate to consider the summary of the recommendations which have been derived. Techniques have been identified and described in schematic diagrams, software flow charts, and text that enable the automatic testing of all of the basic simulator subsystems anticipated in future spacecraft simulators. In the area of digital computers and data conversion equipment, the techniques recommended are those established by the computer manufacturers or already used by simulator users here at the Johnson Space Center. In the areas of controls, displays, visual simulations, motion systems, and aural cues the self test techniques recommended represent new ideas or approaches and involve the application of both hardware and software techniques in a manner in which they have not previously been used for simulator testing.

SECTION 3
PERFORMANCE VERIFICATION (TASK 2.0)

WBS 2.0

PERFORMANCE VERIFICATION

P. B. SCHOONMAKER

In this section, we discuss our approach to the performance verification (validation) study task, and present the highlights of our study results.

WBS 2.0, PERFORMANCE VERIFICATION TASK: PURPOSE AND SCOPE

- PURPOSE: To define guidelines and techniques for verification of simulation fidelity relative to the real world (response and behavior indiscernible from those the crew will experience during actual flight).

- SCOPE: Must provide for validation of individual simulation modules, partially-integrated simulations, and all-up integrated simulations.

3.1 PURPOSE AND SCOPE

The purpose and scope of this study task, as defined by the contract statement of work, are shown above. While task 1.0 was concerned with verifying the performance of the simulator hardware with respect to its specifications, task 2.0 is concerned with verifying the performance of the total simulation (software plus hardware), with respect to the real world. This includes environment, trajectory dynamics, attitude dynamics, onboard subsystems, visual displays, and motion cues. The goal is to verify that while "flying" the simulator, the crews are presented with a task environment which is indiscernible from the actual operational spacecraft.

The task guidelines and techniques developed in this study are to be applicable to individual simulation modules, partially-integrated simulations (e. g., non-realtime simulation programs), and all-up integrated, man-in-loop, realtime simulators.

PERFORMANCE VERIFICATION SUBTASKS

WBS 2.1: DEFINE PERFORMANCE PARAMETERS

Define the parameters which best describe the performance of each spacecraft subsystem and simulation math model.

WBS 2.2: DEFINE METHODS FOR ESTABLISHING STANDARDS OF PERFORMANCE

Define methods to provide reference data for evaluation of simulation performance -- batch programs, lab test data, flight test data.

Determine data formats, access data base impact.

WBS 2.3: DEFINE PERFORMANCE VALIDATION METHODS

Define methods for realtime acquisition and formatting of simulation performance data.

Define methods and criteria for comparison with reference data, and evaluation of simulation performance.

The purpose of task 2.0 is satisfied by performance of the three subtasks defined above.

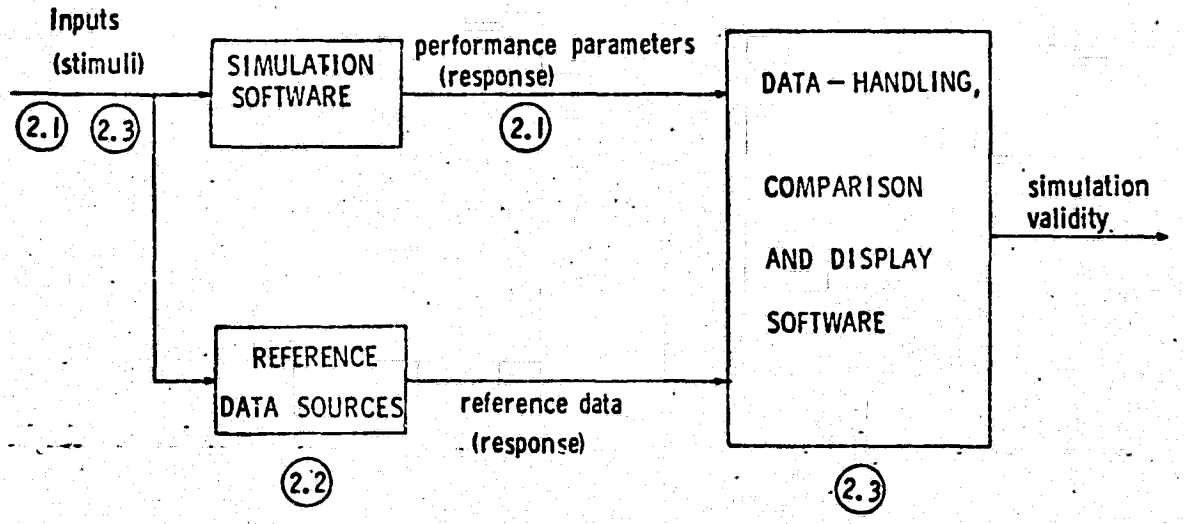
The performance parameters (and "critical" performance parameters), defined for each spacecraft subsystem/simulation module in Subtask 2.1, are the keystone of any validation effort. Performance parameters are the points of comparison between the simulation and the real world.

By "standards of performance" (Subtask 2.2), we mean sets of reference data which represent the real world. Comparison of simulation performance parameters with reference data representing comparable flight conditions enables analysis of simulation validity. Since the quantity and variety of data required for validation of a large spacecraft simulator will be so extensive, such subsidiary questions as data formats and data base requirements become important practical matters.

Finally, in Subtask 2.3, we define methods for acquiring, handling and processing simulation and reference data to evaluate simulation validity. This includes definition of check cases, support software, formatting methods for manual evaluation, and processing algorithms for automated evaluation.

WBS 2.0, Performance Verification Subtask

THE PERFORMANCE VERIFICATION PROCESS: USE OF SUBTASK RESULTS



The interrelationships of these three subtasks, in the context of the total performance verification process, are shown in the above figure; the circled numbers indicate areas where the outputs of each subtask are applied.

As in hardware checkout, the stimulus/response notion is useful in performance verification. In Subtask 2.1, we identify the inputs to and outputs (checkpoints) which should be supplied to each simulation module to properly exercise it. For the same inputs, reference data sources (Subtask 2.2) provide the "correct" performance parameter values for comparison (i.e., check cases). Finally, data-handling, comparison and display techniques developed in Subtask 2.3 enable assessment of simulation validity.

PERFORMANCE VERIFICATION TASK REQUIREMENTS

REQUIRED TASK OUTPUT:
Simulation Performance Validation Techniques Document (DRL-3)

REQUIRED REPORT CONTENTS (per DRD):

- A description of the simulation performance parameters.
- A description of batch programs to provide reference data for simulation validation:
  - -- program constants and variables
  - -- interrelationships of program modules
  - -- program input and output variables
  - -- math models and flow charts

  (for each module)
- Special test requirements for validation reference data.
- Formats for presentation of reference data.
- Realtime data acquisition methods.
- Realtime system impacts.
- Data comparison and evaluation methods, and associated software.
- Comparison criteria.

3.2 DOCUMENTATION REQUIREMENTS

The results of Task 2.0 were documented in the Simulation Performance Validation Techniques Document (DRL-3). The contents of DRL-3, as specified by its associated data requirements description (DRD), are listed above.

The module-oriented material in DRL-3 consists of the sections accented by the vertical line.

PERFORMANCE VERIFICATION REPORT/PRESENTATION OUTLINE

3. SIMULATION SOFTWARE HIERARCHY

4. PERFORMANCE PARAMETERS, STANDARDS OF PERFORMANCE, AND MODULE VALIDATION

- 4.1 Performance Parameter Guidelines
- 4.2 Alternate Reference Data Sources
- 4.3 Environment Modules
- 4.4 Crew Station Modules
- 4.5 Vehicle Configuration Modules
- 4.6 Vehicle Dynamics Modules
- 4.7 Vehicle Subsystems Modules
- 4.8 Module Integration
- 4.9 Special Test Requirements
- 4.10 Reference Data Formats
- 4.11 Data Base Impact

(4.3 – 4.7: module-oriented studies)

5. METHODS FOR VALIDATING PERFORMANCE

- 5.1 Validation Software Structure
- 5.2 Simulator Integration/Validation
- 5.3 Check Case Formulation
- 5.4 Realtime Data Acquisition and Formatting
- 5.5 Comparison Methods and Criteria

6. CONCLUSIONS AND RECOMMENDATIONS

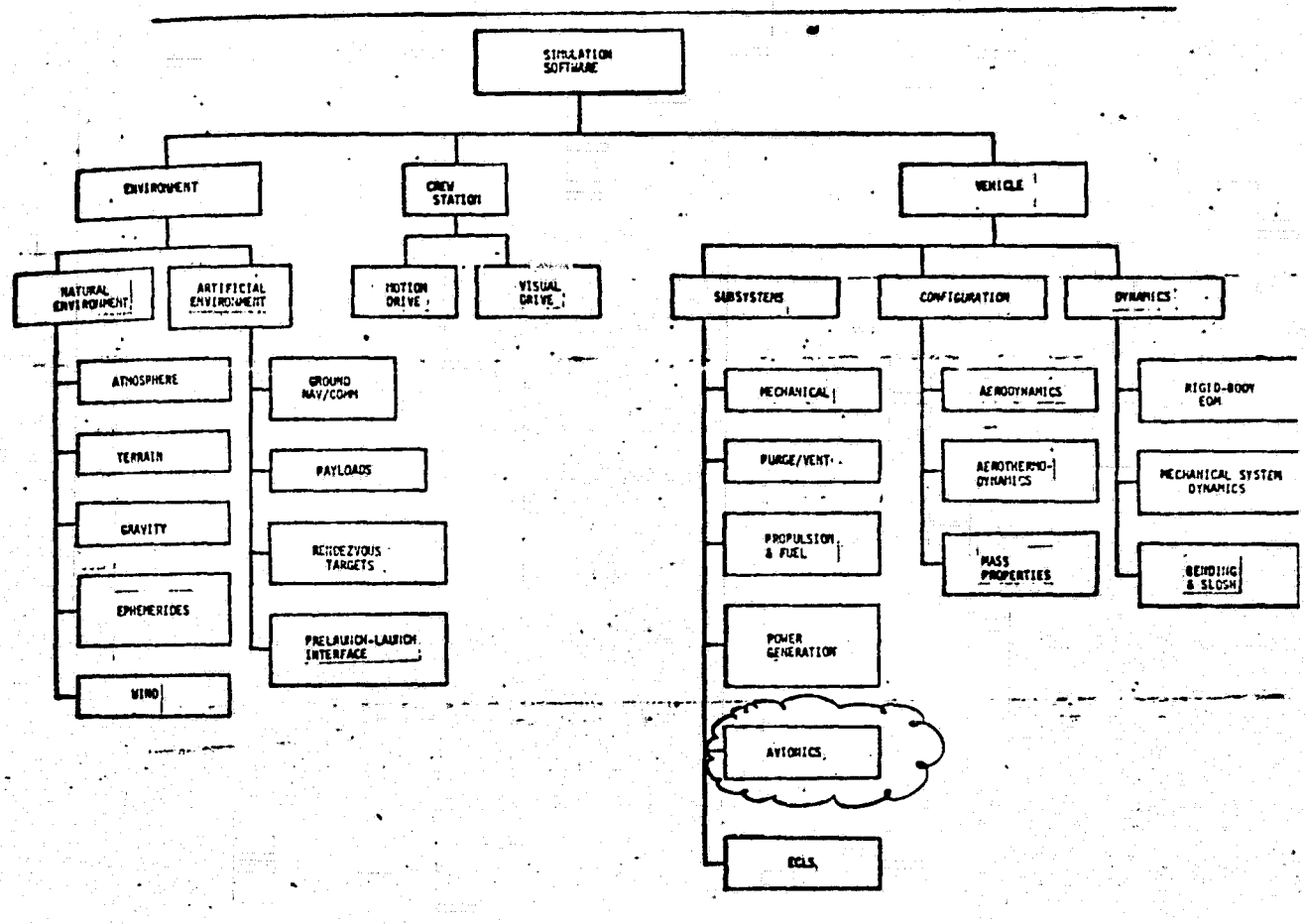
In final form, DRL-3 ran 760 pages, bound in two volumes, and represents a data base for use in future validation efforts. In this report we present only a very brief summary of the results of Task 2.0. It is our hope that after reviewing this final summary report, readers will be stimulated to go to DRL-3 to obtain more detailed information on topics of particular interest to them. For convenience, this section of the final report follows the outline of DRL-3 directly.

3.3 SIMULATION SOFTWARE HIERARCHY

The breadth of the module-oriented efforts is suggested by the uper figure on the following page, which identifies the major categories of simulation modules: Environment, Crew Station, Vehicle Subsystems, Vehicle Configuration, and Vehicle Dynamics.

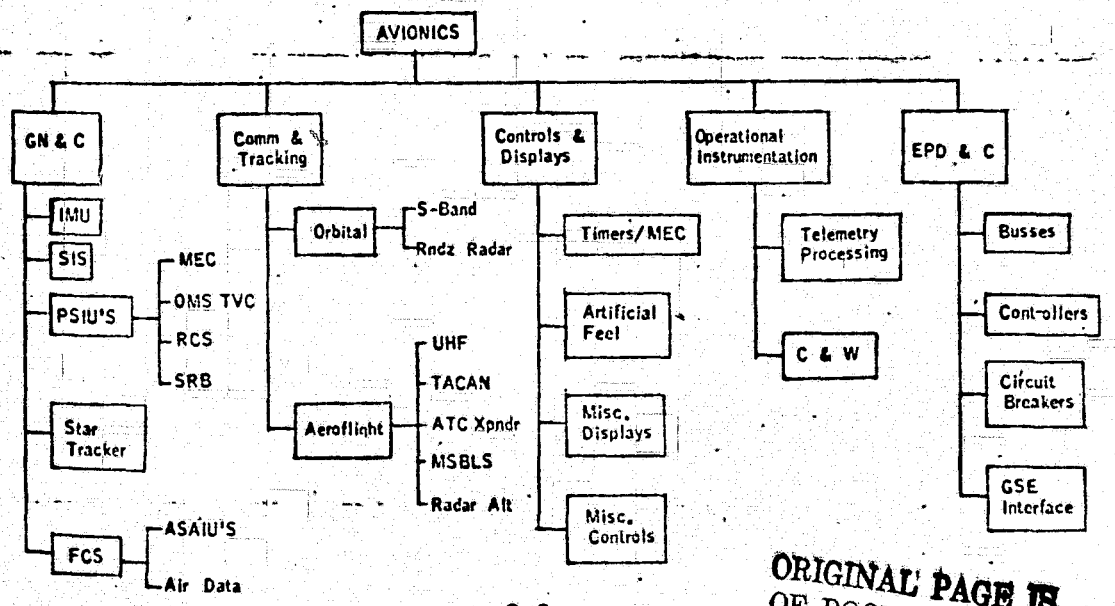
The depth of the module hierarchy cannot easily be shown on a single tree-type figure. In the lower figure, we take the avionics class of vehicle subsystems (which had the "deepest" hierarchy), and complete the expansion down to the individual module level. Every category of simulation software is similarly expanded in DRL-3, and every module thus identified is treated at an appropriate level of detail.

3. SIMULATION SOFTWARE HIERARCHY



Simulation Software Hierarchy:

EXPANSION TO INDIVIDUAL MODULE LEVEL



4. PERFORMANCE PARAMETERS, STANDARDS OF PERFORMANCE AND MODULE VALIDATION

Necessary Preliminaries:	4.1 Performance Parameter Guidelines
	4.2 Alternate Reference Data Sources
Module-Oriented Efforts:	4.3 Environment Modules
	4.4 Crew Station Modules
	4.5 Vehicle Configuration Modules
	4.6 Vehicle Dynamics Modules
	4.7 Vehicle Subsystems Modules
Unifying Efforts:	4.8 Module Integration
	4.9 Special Test Requirements
	4.10 Reference Data Formats
	4.11 Data Base Impact

3.4 PERFORMANCE PARAMETERS, STANDARDS OF PERFORMANCE, AND MODULE VALIDATION

Because of the specialized information needed to deal with each individual subsystem/module, we decided to treat the module-related aspects of all three subtasks -- performance parameters, standards of performance, and validation methods -- as unified study activities and document the results in module-oriented sections. Thus, the bulk of the documentation is organized on the basis of moudles rather than subtasks.

The module-oriented efforts were preceded by certain preliminary definition tasks, which were required to effectively perform the module-oriented studies, and followed by unified study of certain topics which apply to all modules, as well as to validation at various levels of integration.

4.1 PERFORMANCE PARAMETER GUIDELINES

- BASIS FOR "PERFORMANCE PARAMETERS"

  A set of parameters which completely describes each subsystem/module. Thus, complete module validation is achieved by verifying agreement of all performance parameters with their reference values, under appropriate conditions.

- RATIONALE FOR "CRITICAL" PERFORMANCE PARAMETERS

  Concentrate effort on a smaller body of data: the parameters which are most significant indicators of simulation performance.

  Particularly important for re-validation following simulation modifications.

3.4.1 Performance Parameter Guidelines

Our goal, as defined by the SOW, was to identify a set of parameters which would completely describe the performance of each subsystem/module (thus allowing complete initial validation of each module). In addition to this goal, we defined for ourselves the additional goal of selecting a subset of "critical" performance parameters.

Our rationale was that concentrating upon a smaller body of data would improve the efficiency of the validation process for all modules and at all levels of integration. The use of critical performance parameters will be particularly important for the inevitable revalidation exercises which will occur from time to time during the life of the simulation, following modifications and updates. In contrast to the complete initial validation, we contend that in most cases of revalidation, it will only be necessary to closely compare the critical performance parameters to their reference values. If a good match is secured for the critical performance parameters, comparison of other performance parameters can safely be omitted, or at most spot-checked.

On the next page, we list the guidelines for selection of performance parameters and critical performance parameters. These guidelines were uniformly applied by the study staff in analyzing the basic information describing vehicle subsystems, simulator requirements, etc.

PERFORMANCE-PARAMETER IDENTIFICATION GUIDELINES

- MUST BE REAL-WORLD VARIABLES (EITHER CONTINUOUS OR DISCRETE).
- MUST BE TIME-VARIABLE QUANTITIES, NOT CONSTANTS.
- ALL SYSTEM STATE VARIABLES ARE PERFORMANCE PARAMETERS.
- SOME MODULE OUTPUTS ARE NOT PERFORMANCE PARAMETERS (e.g., "INCIDENTAL OUTPUTS" -- POWER IN, HEAT OUT).
- SOME PERFORMANCE PARAMETERS ARE NOT MODULE OUTPUTS (ESSENTIAL, REAL-WORLD INTERNAL VARIABLES).
- EVERY VARIABLE AVAILABLE TO FLIGHT COMPUTER OR TELEMETRY MUST BE A PERFORMANCE PARAMETER OF SOME MODULE.
- A MODULE'S INPUTS ARE NEVER PERFORMANCE PARAMETERS FOR THAT MODULE (PREVENTS DOUBLE-COUNTING).

PERFORMANCE PARAMETERS: "CRITICAL" PERFORMANCE PARAMETER GUIDELINES

- PARTICULARLY SIGNIFICANT INDICATOR OF MODULE FIDELITY.
- HAS LONG-TERM OR CUMULATIVE IMPACT UPON SIMULATION VALIDITY.
- IS READILY AVAILABLE TO CREW; PLAYS A KEY ROLE IN OPERATIONAL PROCEDURES.
- IS SUPPLIED TO FLIGHT COMPUTERS; PLAYS A KEY ROLE IN COMPUTER CONTROL OF VEHICLE SYSTEMS

4.2 ALTERNATE REFERENCE DATA SOURCES

- SOURCES OF DIRECTLY USABLE REFERENCE DATA (check cases)
  - -- Closed-Form Solutions
  - -- Independent Math Models
  - -- Existing Analysis/Simulation Programs
  - -- Test Data
- SOURCES OF BASE INFORMATION (For development of math models or check cases)
  - -- Requirements documents
  - -- Specifications, drawings & schematics
  - -- Design data books, operational data books
  - -- Contractors' analyses, studies and simulations
  - -- Tests
- POTENTIAL ROLE OF PICRS/SIS

3.4.2 Alternate Reference Data Sources

For a given subsystem/module, there are a essentially two levels of data and information which will be desired by simulation development and validation personnel: directly-usable reference data (i. e., check cases for validation), and basic descriptive information, which can serve as a starting point for the development of math models and/or check cases. Naturally, our emphasis in the study of data sources was upon sources of directly-usable reference data.

Four basic classes of reference data were identified and compared in this part of the study. The comparison of alternative sources in general terms was documented in Section 4.2 of DRL-3; comparison of alternative data sources identified for a particular module appeared in the appropriate module-oriented section.

We anticipate that PICRS (Program Information Coordination and Review Service) and SIS (Shuttle Information Service) will play major roles in making both reference data and base information available to simulation development and validation personnel. A brief description of PICRS/SIS scope, operations, and retrieval aids is given in DRL-3.

ALTERNATE DATA SOURCES: PROS & CONS

DATA SOURCE	ADVANTAGES	DISADVANTAGES
Closed-Form Solutions	Simplicity Accuracy	Feasibility Scope
Independent Math Models	Scope Compatibility Control	Workload
Existing Analysis/Simulation Programs	Scope	Availability Documentation Incompatibility
Test Data	Fidelity	Timeliness Availability Scope Incompatibility Documentation

The above table summarizes our comparison of the four basic classes of reference-data source, listed in ascending order of realism.

Closed-form solutions (i. e., straight-through computations without iteration, numerical integration or table-lookup functions) are attractive from the viewpoints of simplicity and computational accuracy (not necessarily real-world fidelity). However, for many modules of interest we find that is is not feasible to formulate a single closed-form solution, or that we can formulate a closed-form solution only by sharply limiting the scope of the simulation.

At the other extreme, test data, we have the maximum fidelity, but enccunter a variety of practical problems in obtaining and making use of the data for validation.

3.3 - 4.7: SUBSYSTEM/MODULE-ORIENTED STUDY RESULTS

- DEFINITION: A simulation "module" is a set of software elements which is invoked as a unit, and performs a defined simulation function.
- COVERAGE: For each subsystem/module, we developed the following information:
  - -- System Description (the real-world system)
  - -- Module Description and Parameters (accounting for the fidelity requirements of various simulators)
  - -- Reference Data Sources and Data Formats
  - -- Module Validation Methods and Check Cases
  - -- Module Validation Data Base Impact

(Much "legwork", information search and compilation activity was involved in this phase of the study.)

3.4.3 to 3.4.7: Subsystem/Module-Oriented Study Results

With the necessary preliminaries accomplished, we were able to effectively tackle the subsystem/module-oriented part of the validation study. For the purposes of this study, a "module" is defined as a set of software elements which is invoked as a unit and performs a single simulation function. Modules may be large or small, simple or complex, and may even be further divided into submodules.

The module-oriented documentation follows the simulation software hierarchy previously shown. For each and every module identified, all of the above listed information was provided in the documentation, to an appropriate level of detail. Generation of this documentation required a great deal of "legwork" (i. e., acquisition and compilation of existing information), as well as considerable analysis and generation of additional original material (e.g., new math models).

In the following pages, we present a few examples from each category of information provided by these sections of DRL-3.

Subsystem / Module Results

SYSTEM DESCRIPTIONS

Understand the real-world system.

Pictorial and verbal exposition.

"Raw" data and analysis.

Brief exposition of:

- System purpose
- Functions
- Operational modes
- Hardware interfaces
- Flight crew interfaces
- Mission phases

Used previously-identified base information:

- Basically Shuttle-oriented
- "Design-insensitive" formulation

System Descriptions

An understanding of the real-world system is prerequisite to either simulation development or simulation validation for that system. Therefore, our report provides pictorial and verbal exposition of the purpose, functions, operational modes, interfaces, and other facets of each real-world system.

Some of the information presented in the system descriptions is taken directly from the base information in raw form. In other cases, considerable analysis of the base information is required to understand the system simulation requirements, identify performance parameters, and otherwise make effective use of the information.

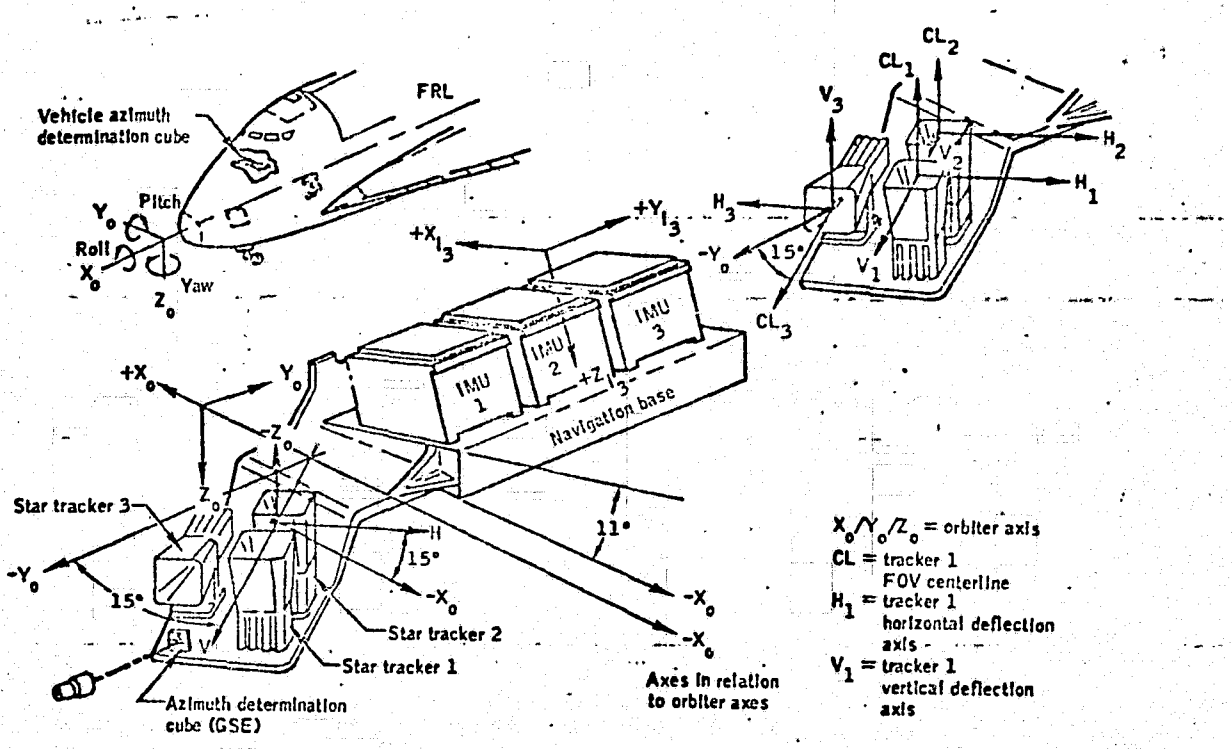
The system treatments are basically Shuttle-oriented, since the Shuttle is the spacecraft of greatest current interest. However, at attempt was made to formulate the system descriptions in a design-insensitive manner, so that they would remain valid and useful even if the Shuttle system designs were to change in detail aspects. Our experience with onboard systems and simulations from other aerospace programs (Gemini, Apollo, Skylab, DC-10, etc.) was useful in this effort.

Examples of descriptive information provided for the star tracker are shown on the following page. The top illustration shows the orientation geometry for the three star trackers, two of which are seen to produce overlapping coverage. This figure was taken directly from an existing report.

The bottom illustration shows our analysis of scan-pattern relationships, which was necessary to determine how certain star tracker performance parameters -- acquisition time, target brightness, and target coordinates -- might be generated in a high-fidelity star tracker simulation.

System Description

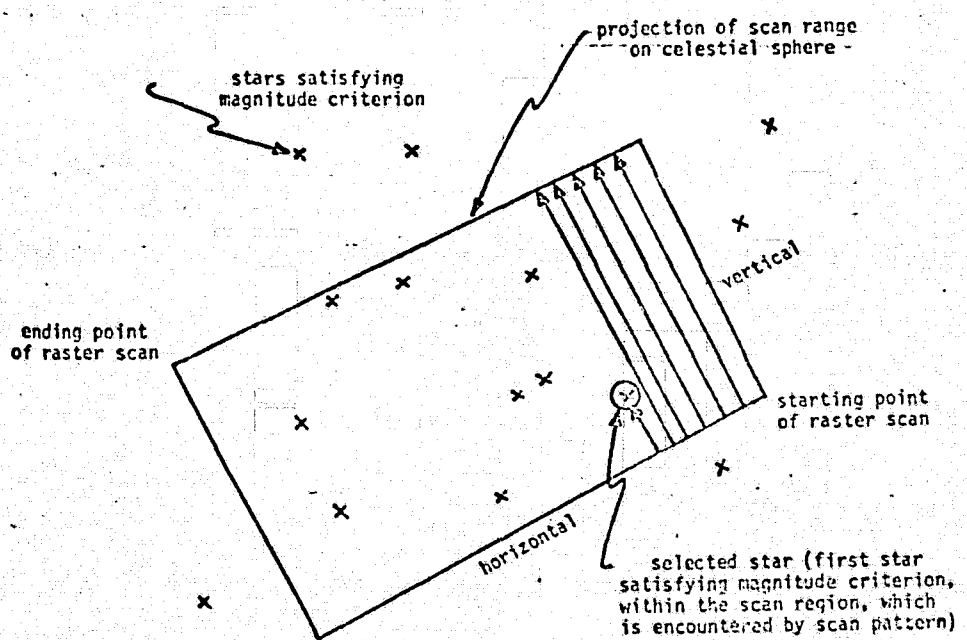
EXAMPLE: STAR TRACKER GEOMETRY



System Description

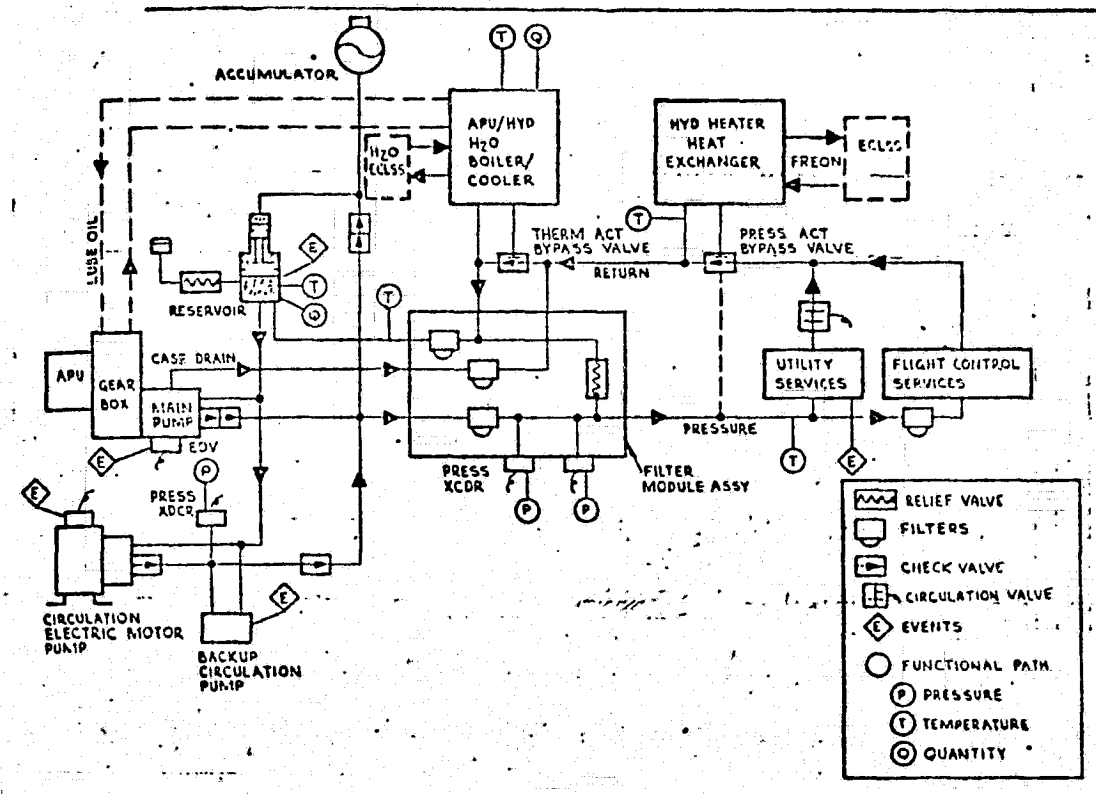
EXAMPLE: STAR TRACKER SCAN PATTERN RELATIONSHIPS

PERFORMANCE PARAMETERS: Acquisition time, target coordinates, target brightness.



System Description

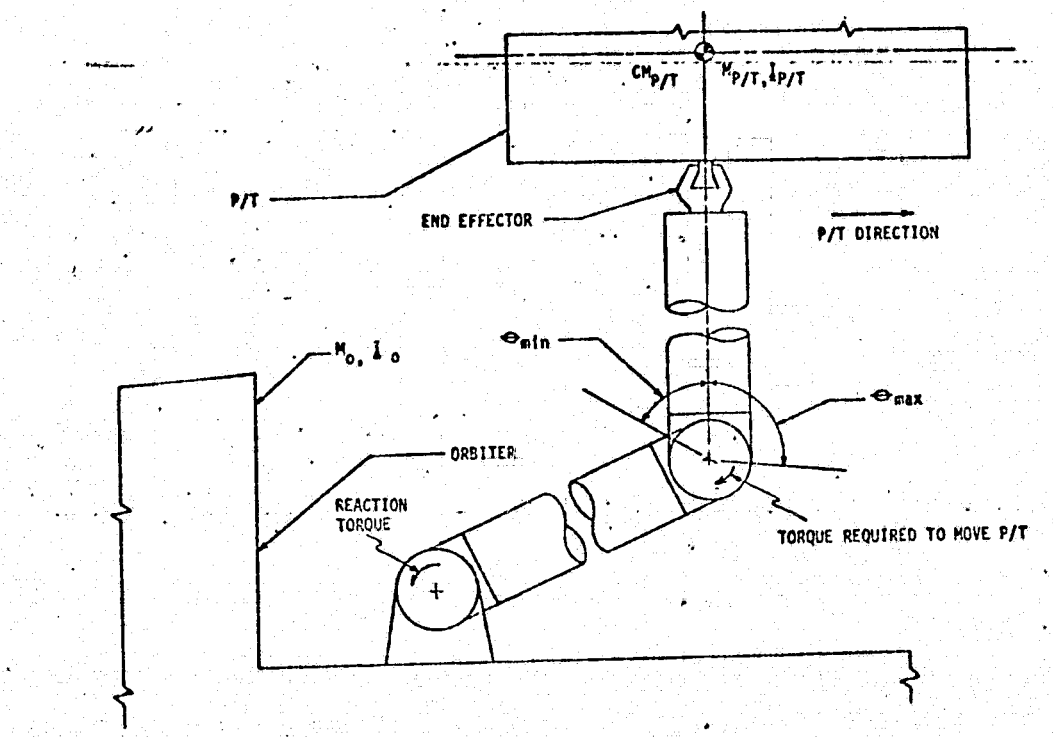
EXAMPLE: HPS SCHEMATIC, showing performance parameter location



The above schematic of the Shuttle hydraulic power system, although based upon existing information (e. g., specifications), was redrawn for greater clarity, and labelled with flags indicating points on the actual system at which values corresponding to the simulation module performance parameters could be monitored.

System Descriptions

EXAMPLE: PDRS (FUO) JOINT REACTIONS



The above figure schematically indicates certain performance parameters of the Payload Deployment and Retrieval System (PDRS) Follow-Up Output (FUO) subsystem; e.g., joint angles, actuator torques, and end effector open/closed position. It also indicates hardware constraints (max and min angles), and external variables which will enter into PDRS simulation, such as masses, inertias, and motions of the Orbiter and the payload/target.

Subsystem / Module Results

MODULE DESCRIPTIONS AND PARAMETERS

Subsystem representation and fidelity may differ for different simulators (SMS, SPS, OAS):

- Fidelity: high, medium, low, talkback, omitted.
- Use of flight hardware & software vs. functional simulation.
- Representation of redundancies.

Simulation modes and functions; mission phase.

Module description always includes:

- Module interface diagram.
- Parameter list.

Module Descriptions and Parameters

In addition to an understanding of the real-world system, an understanding of its simulation requirements is required to formulate the validation requirements for the associated simulation module. Of the three simulators of greatest interest to this study (SMS, SPS and OAS), the SMS generally requires the most detailed simulation. The variation in level of simulation detail among these simulators was taken into account in our analyses of simulation modules and their performance parameters. A good example of this variation in level of detail will be seen presently in our treatment of the main propulsion system.

Two things which were always included in our treatment of module descriptions were a module interface diagram and a parameter list; examples are shown on the following pages.

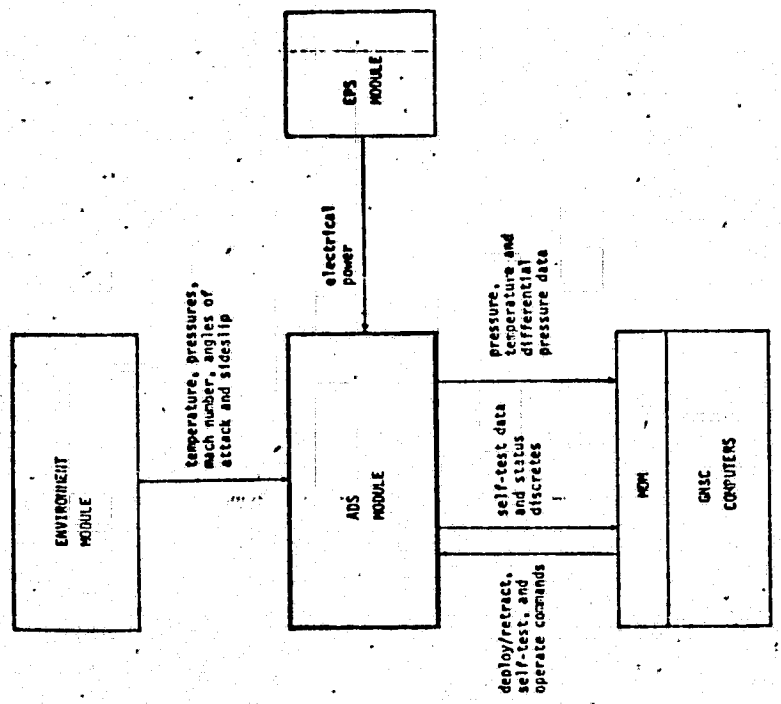
The module interface diagram and parameter list for the air data system module are shown on the following page.

The module interface diagram provides a convenient representation of the interactions of the module of interest with other parts of the simulation. It shows the inputs and outputs of the module of interest, where its inputs come from, and where its outputs go to. This is important when analyzing requirements for module "drivers", and formulating sets of check case data. Our analysis of module interactions also proved useful later on, in formulating the overall simulation integration/validation sequence.

Several different types of parameters are shown and defined on the module parameter list: inputs, incidental outputs, internal data base parameters, performance parameters, and critical performance parameters. Some modules, of course, may not have parameters of every possible type.

Module Description & Parameters

EXAMPLE: AIR DATA SYSTEM



Air Data System, continued

SYMBOL	DEFINITION	TYPE[a]
- -	Command for self-test mode or operation mode	I
T_o, P_o	Ambient air temperature and pressure	I
M	Mach number	I
α, β, V_a	Angle-of-attack, angle-of-sideslip and airspeed	I
- -	ADTA self-test values for P_{s1}, P_{t1}, T_{t1}, ΔP_1, and mode/status	DB
η	Temperature sensor recovery factor	DB
γ	Specific heat ratio for air	DB
P_{so}, P_{to}, T_{to}	Ideal probe values of static pressure, total pressure and total temperature	P
ΔP	Ideal probe differential pressure (function of vehicle aerodynamics)	P
ϵP_{so}, ϵP_{to}, ϵT_{to}	Changes in ideal probe values due to vehicle dynamics	P
ϵP_{s1}, ϵP_{t1}, ϵT_{t1}, $\epsilon \Delta P_1$	ADTA hardware errors	P
P_{s1}	Indicated static pressure (divided into most significant and least significant words)	CP
P_{t1}	Indicated total pressure	CP
T_{t1}	Indicated total temperature	CP
ΔP_1	Indicated pressure differential	CP
- -	ADTA Operational Mode and Status flag	O
- -	Power-on discrete from ADTA	O
- -	Probe heater status discrete	O
- -	Probe deploy/retract status discrete	O

[a]LEGEND: I = input
DB = data base input
O = output
P = performance parameter
CP = critical performance parameter

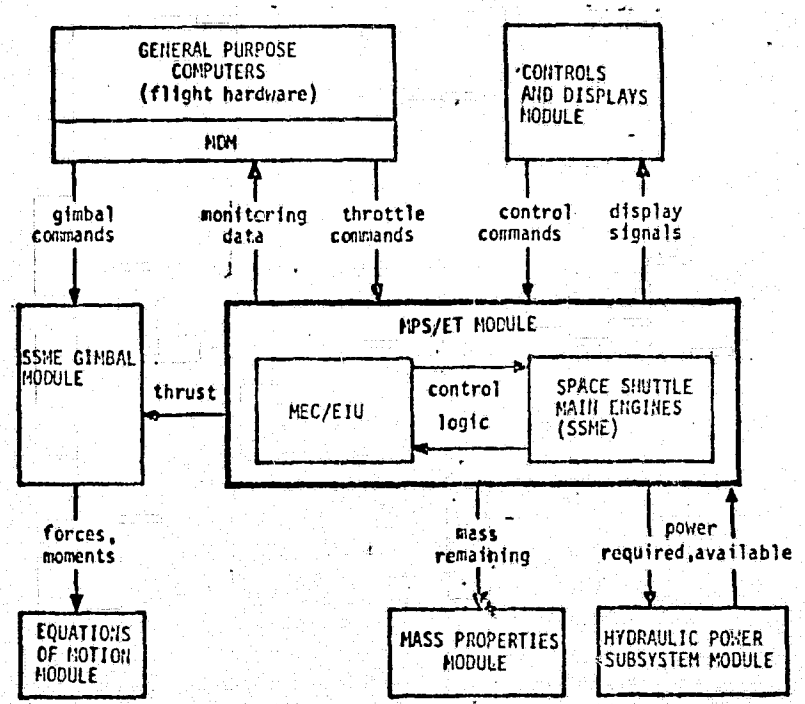
A portion of the complete parameter list for a high-fidelity (i. e., SMS) simulation of the Main Propulsion System (MPS) is shown on the next page. The complete parameter list ran several pages, because of the complex plumbing and valving associated with this system. The table was shortened somewhat by showing paired command/response discretes on a single line, as "... CMD/RESP... I/CP". In a functional simulation of the MPS, as might be used in the SPS, many of these parameters would not exist.

The module interface diagram for the MPS module (incorporating the main engine controller and engine interface unit) is also shown.

EXAMPLE: MAIN PROPULSION SYSTEM

PARAMETER NOMENCLATURE	DATA RANGE LOW	DATA RANGE HIGH	DATA RANGE UNIT	TYPE[a]
FIERG SHUTDOWN INHIBIT CMD/RESP[b]		ON	EVENT	I/CP[b]
AC POWER NO 1 ON CMD /RESP		ON	EVENT	I/CP
HEATER POWER ON CMD /RESP		ON	EVENT	I/P
CONTROLLED TEMP	-200	+400	DEG F	P
LH2 PREVALVE OPEN CMD /RESP		ON	EVENT	I/CP
LH2 PREVALVE CLOSE CMD /RESP		ON	EVENT	I/CP
LO2 PREVALVE OPEN CMD /RESP		ON	EVENT	I/CP
LO2 PREVALVE CLOSE CMD /RESP		ON	EVENT	I/CP
HELIUM BOTTLE TEMP	- 65	+500	DEG F	CP
HE ISLN VLV 1 OPEN CMD/RESP		ON	EVENT	I/P
HE ISLN VLV 2 OPEN CMD /RESP		ON	EVENT	I/P
MPS-LH2 INBD FILL VALVE OPEN CMD /RESP		ON	EVENT	I/CP
MPS-LH2 INBD FILL VALVE CLOSE CMD/RESP		ON	EVENT	I/CP
MPS-LH2 OUTBD FILL VALVE OPEN CMD/RESP		ON	EVENT	I/CP
MPS-LH2 OUTBD FILL VALVE CLOSE CMD/RESP		ON	EVENT	I/CP
MPS-LH2 RECIRC DISC VLV OPEN CMD/RESP	ON	OFF	EVENT	I/CP
MPS-LH2 RECIRC DISC VLV CLOSED CMD/RESP	ON	OFF	EVENT	I/CP
MPS-LH2 RECIRC DISC VLV OPEN CMD/RESP		ON	EVENT	I/CP
MPS-LH2 RECIRC DISC VLV CLOSE CMD/RESP		ON	EVENT	I/CP
[illegible] VLV OPEN CMD/RESP				
[illegible] CMD/[illegible]				

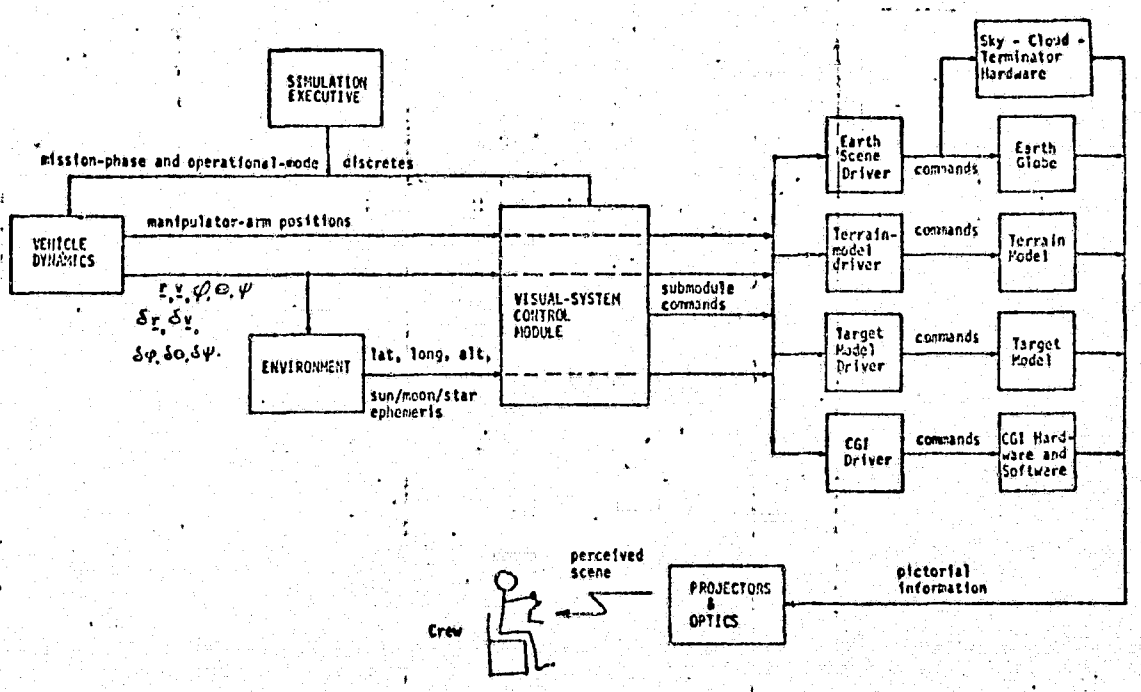
Main Propulsion System, continued



The visual system drive module is intimately related to the simulator hardware and the flight crew displays and controls. Therefore, both hardware and software interactions are indicated on the following module interface diagram and parameter table. The motion-base drive module (not shown) was treated similarly.

Module Description & Parameters

EXAMPLE: VISUAL-SYSTEM DRIVES



ORIGINAL PAGE IS
OF POOR QUALITY

Visual-System Drives, continued

SYMBOL	DESCRIPTION	TYPE[a]
$\underline{r}, \underline{v}$	Vehicle position & velocity vectors	I
φ, Θ, Ψ	Vehicle Euler angles	I
$\delta\underline{r}, \delta\underline{v}$	Multiple-body relative position & velocity vectors	I
$\delta\varphi, \delta\Theta, \delta\Psi$	Multiple-body relative Euler angles	I
--	Vehicle latitude, longitude, altitude	I
--	Sun/moon/star ephemerides	I
--	Mission-phase and operational-mode discretes	I
--	Submodule activation commands	P
--	Visual hardware scaling, lags, limits, etc.	I
--	CIG imagery data base	I
--	Earth globe and camera position commands	CP
--	Terrain model carriage & optical probe commands	CP
--	Target model & camera track and gimbal commands	CP
--	Visual-subsystem hardware discretes and position feedback	I
$\delta\underline{r}'$	Payload c.g. position relative to orbiter eyepoint(s)	CP
$\{\delta r_{PDRS}\}$	PDRS joint positions	I
$\{\delta r'_{PDRS}\}$	PDRS joint positions relative to orbiter eyepoint(s)	CP
$\{\Theta_{PDRS}\}$	PDRS joint angles	I
δr_c	PDRS camera eyepoint	CP
--	Camera and light activation discretes	I
--	Orbiter/payload/arm contact discretes	I

[a]Legend: I = input
P = performance parameter
CP = critical performance parameter

MODULE REFERENCE DATA SOURCES AND DATA FORMATS

Search for and study of existing sources:

- Big program systems - SVDS, SSFS, G189A (complete or individual subroutines)
- Math Model coordination catalog and other sources
- Test data

Development effort toward new reference-data generation programs:

- Closed-form solutions
- Independent math models (various levels of detail)

Study of data formats:

- I/O formats for existing programs
- Hard-copy formats

Module Reference Data Sources and Data Formats

Two basically different types of activity were carried on in our study of reference data sources: study of existing data sources, and development of new data sources.

We studied many existing computer programs, such as the Space Vehicle Dynamics Simulation (SVDS), the Space Shuttle Functional Simulation (SSFS), and the Generalized Environmental/Thermal Control and Life Support Systems program (G189A), to determine their potential utility for module validation.

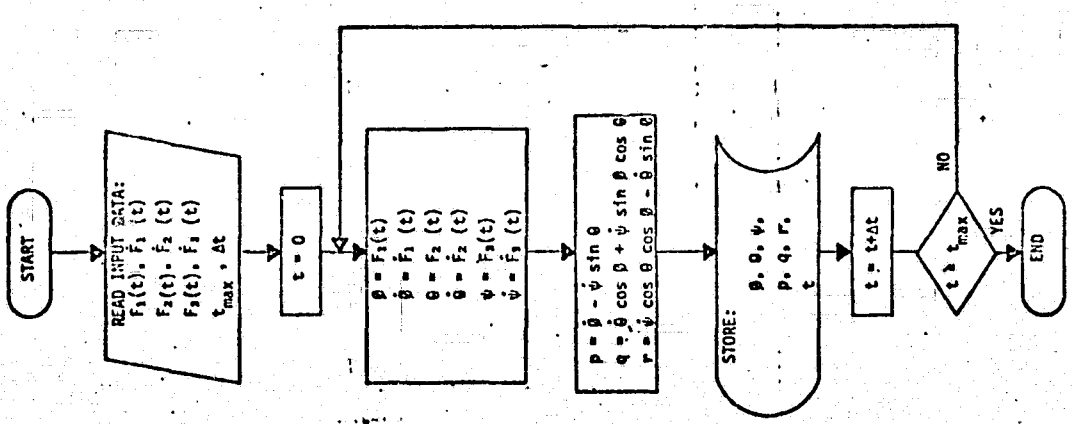
For certain modules, we were able to determine the types of test data which would become available, and to assess the advantages and potential problems of using such test data.

Where existing data sources seemed inadequate, we undertook to specify and perform the initial design of new software to serve as reference data generation program.

Our study of data formats included input/output formats for SVDS and other programs, standard trajectory tape and GEMASS formats, and hard-copy formats for test data and other reference data which would not normally be available in machine-readable form.

Reference Data & Format

EXAMPLE: IMU CLOSED-FORM SOLUTION

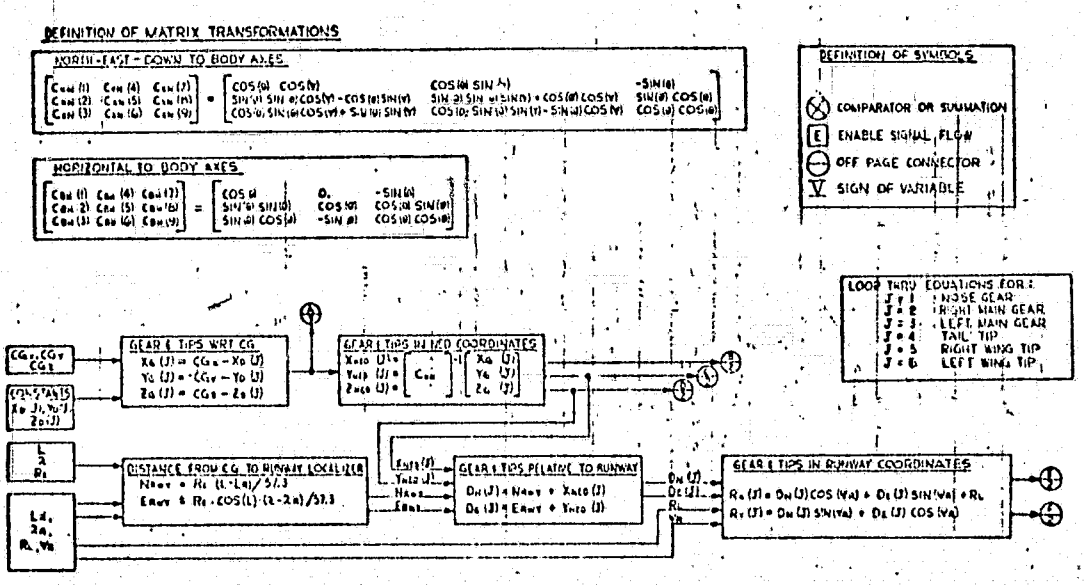


The math flow for a closed-form solution of inertial measurement unit kinematics is shown above. As previously stated in general for closed-form solutions, this solution is valid only under certain restrictions. First, the input gimbal angles must be differentiable functions of time. Second, the gimbal rates must be within the dynamical capability of the IMU hardware.

With these restrictions, the above algorithm "inverts" the IMU simulation function; i. e., given desired IMU outputs (gimbal angle and rate time histories), it computes required IMU inputs (body rate time histories). The manner in which this reference module is used for validation will be shown shortly.

Reference Data & Formats

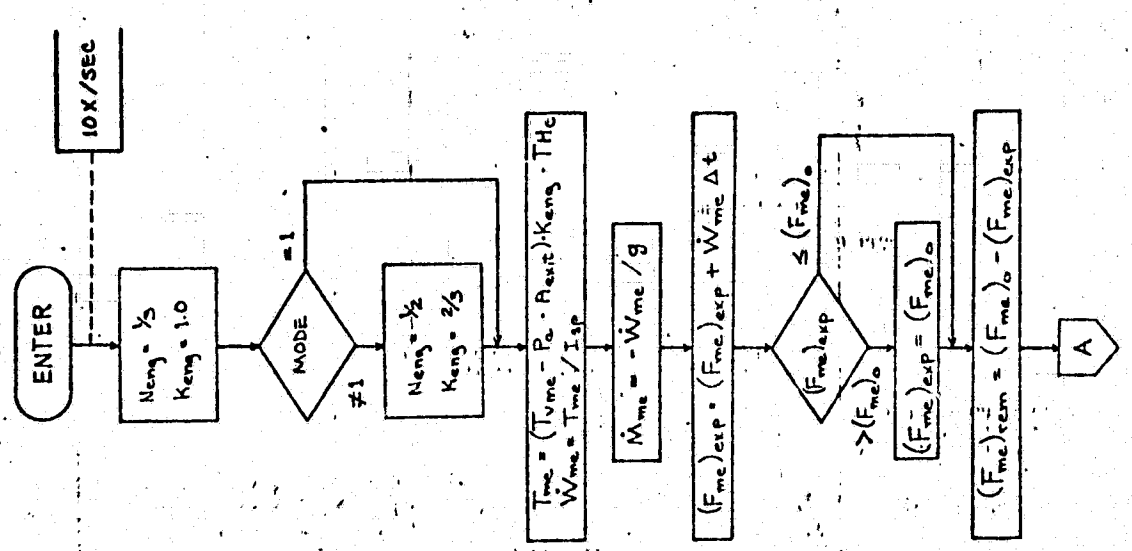
EXAMPLE: EXISTING LDS SIMULATION



A portion of the math flow for a rather detailed engineering simulation of the landing/deceleration system is shown above; the complete math flow runs several pages. This program has the capability to generate detailed check case data for validation of the LDS simulation module.

Reference Data & Formats

EXAMPLE: INDEPENDENT MPS MATH MODEL

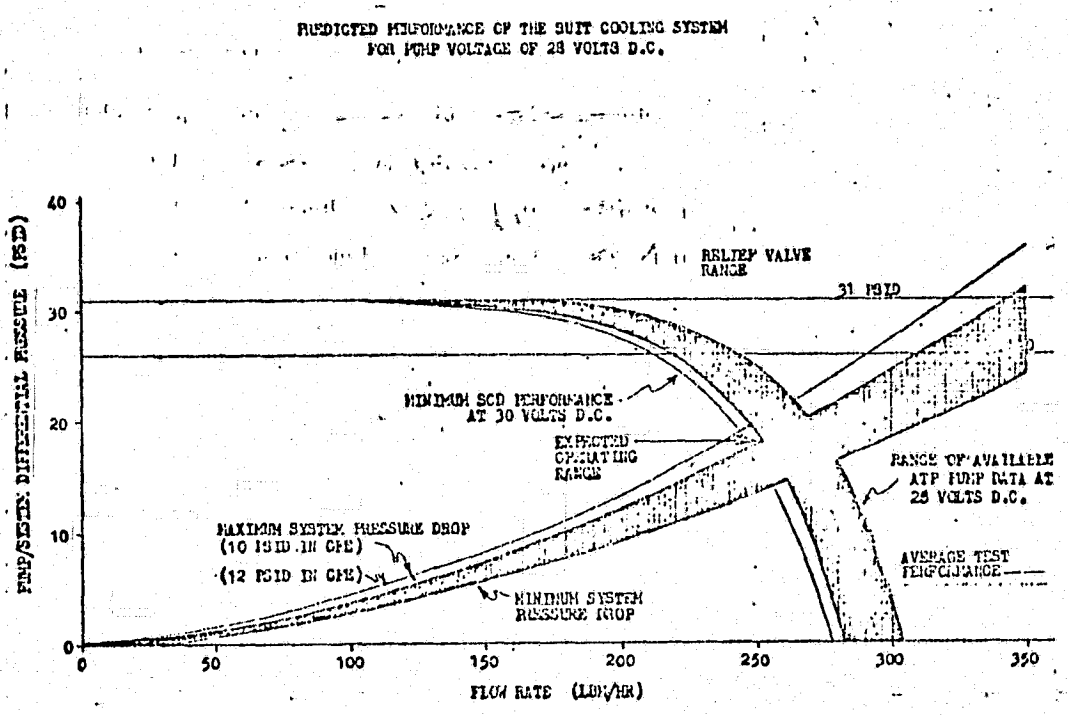


A portion of the math flow for an independently-developed functional simulation of the main propulsion system is shown above. This program, whose complete math flow runs several pages, has the capability to generate check case data including forces and moments, specific impulse, mass flow rates, and engine gimbal angles. Thus, it would be suitable for validation of the ascent-phase simulation for a simulator such as the SPS, or for initial validation of the corresponding portion of the SMS.

A highly detailed engineering simulation of the MPS is also described in our report. That program includes the dynamics of plumbing, valving, turbopumps, etc., capabilities which would be necessary for complete validation of the MPS simulation expected on the SMS.

Reference Data & Formats

EXAMPLE: PUMP TEST DATA (HARD COPY)



An example of test data in hard-copy form is shown above. These data represent the performance of a pump in the Skylab suit cooling system. The graph was generated by compiling test results from a number of different individual units, after considerable "legwork" to recover the original data. Multiple-unit data of this kind would be highly desirable, since it shows the "scatter" to be expected in the real-world system performance data. This in turn serves as a guide to the development of simulation fidelity requirements and the formulation of accuracy criteria for validation.

Subsystem / Module Results

MODULE VALIDATION METHODS AND CHECK CASES

"Driver" routines

Static & dynamic check cases to exercise module:

- Mission phases, operational modes
- Discrete inputs - singly and in combinations
- Continuous inputs - ranges and combinations

Accuracy considerations

Manual/auto techniques

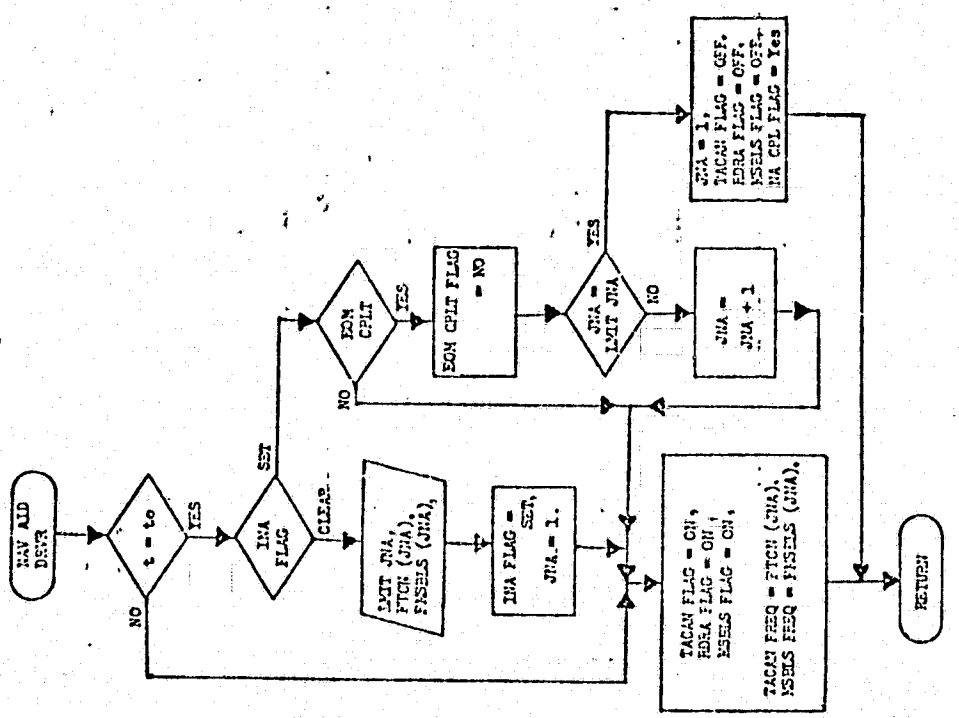
Module Validation Methods and Check Cases

At the module level, the treatment of validation methods consists primarily of a description of driver routines necessary to perform module interfacing and I/O, and a specification of static and dynamic check cases necessary to thoroughly exercise the module. Accuracy considerations are also discussed in cases where sufficient information presently exists to assess typical reference data accuracy.

Some discussion of manual and automatic techniques for validation is provided at the module level; however, the discussion of techniques is better handled in general than in the context of a particular module (see Sect. 3.5).

Module Validation Method

EXAMPLE: GROUND NAV AIDS CHECKPOINT DRIVER

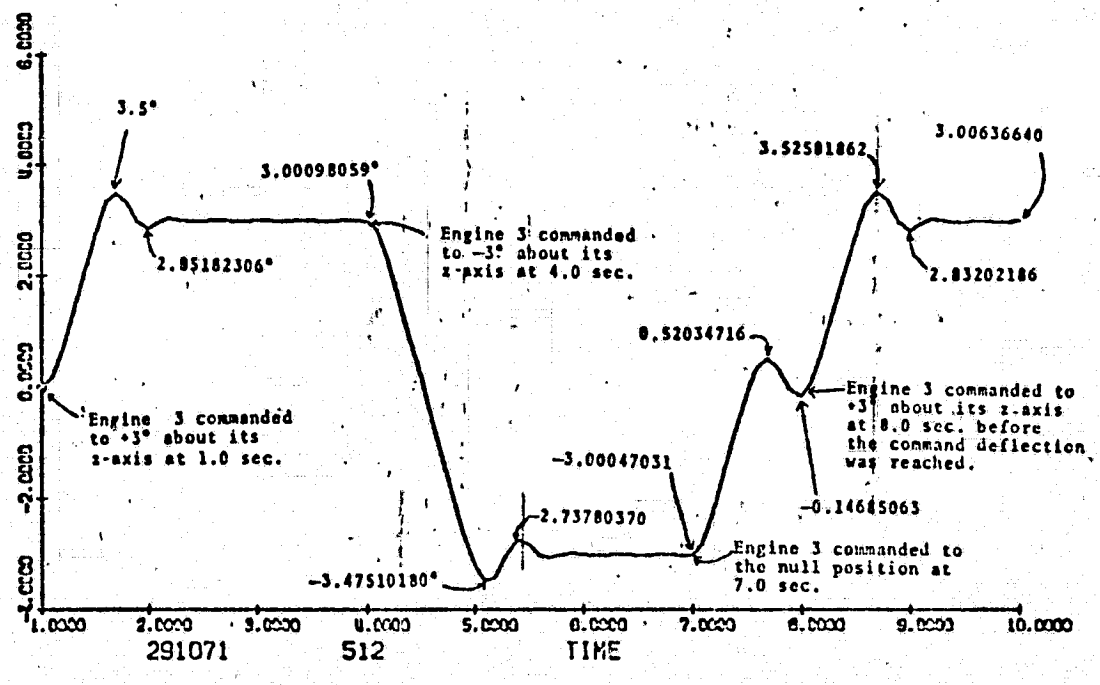


The above math flow represents an example of a "checkpoint driver" routine; i. e., a routine which provides the data and logic to automatically generate a set of check cases to exercise a particular module (in this case, the ground navaids module of the artificial environemnt).

As discussed in greater detail in Sect. 3.5.2, the driver used for validation of an isolated module must provide all the inputs which, in normal operation, will come from all the modules which interface with that module. For the ground navaids module, the primary interfacing modules are the EOM and onboard communications and tracking modules.

Module Validation Methods

EXAMPLE: OMS GIMBAL RESPONSE CHECK CASE

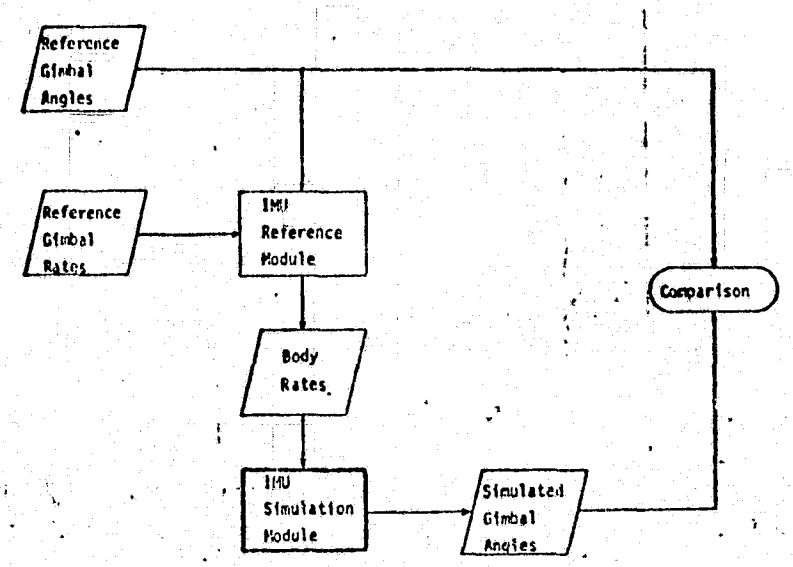


A typical rigorous check case for dynamical checkout of a mechanical-system simulation module (in this case the OMS gimbal actuator simulation) is shown above.

These command/response time-history data were generated by a detailed engineering simulation. Later in the Shuttle program, actual test data of simular form will become available.

Module Validation Methods

EXAMPLE: IMU VALIDATION USING CLOSED-FORM SOLUTIONS



The above figure shows the software organization appropriate to the closed-form IMU reference module previously described. The reference gimbal angle and rate time-histories are input to the reference module, which "inverts" the IMU, resulting in body rate time-histories. These body rate time-history data are fed into the IMU simulation module. The gimbal angle time-histories output by the IMU simulation module should then very closely match the reference gimbal angles.

A closed-form solution is very desirable for this application, because it provides a simple means of computing reference data with high accuracy.

Subsystem / Module Result

MODULE VALIDATION DATA BASE IMPACT

Enumerate data-base entries required to service validation of the individual module:

- Reference module (s)
- Driver (s)
- Checkpoint data files
- Service routines

Assess impact qualitatively (minor/moderate/major); include commonality of programs/data.

Module Validation Data Base Impact

Our overall analysis of validation data base impact is provided in Section 3.4.11. At the individual module level, the treatment of data base impact consisted simply of an enumeration of the programs and data files which would be required in the data base to service validation of the module of interest, and a qualitative assessment of the magnitude of the data base impact for validation of that module.

THIS CONCLUDES THE SUBSYSTEM/MODULE-ORIENTED TREATMENT.

We will now consider aspects of validation which apply to all modules and at various levels of integration.

4.8 MODULE INTEGRATION

- Simulation integration is a "clustering" process.
- Our Module Interface Diagrams (Sections 4.3-4.7) can be used to define the most natural clustering sequence.
- For maximum overall efficiency, use the natural integration sequence to schedule module development and validation.
- Hardware schedule constraints must also be considered.

3.4.8 Module Integration

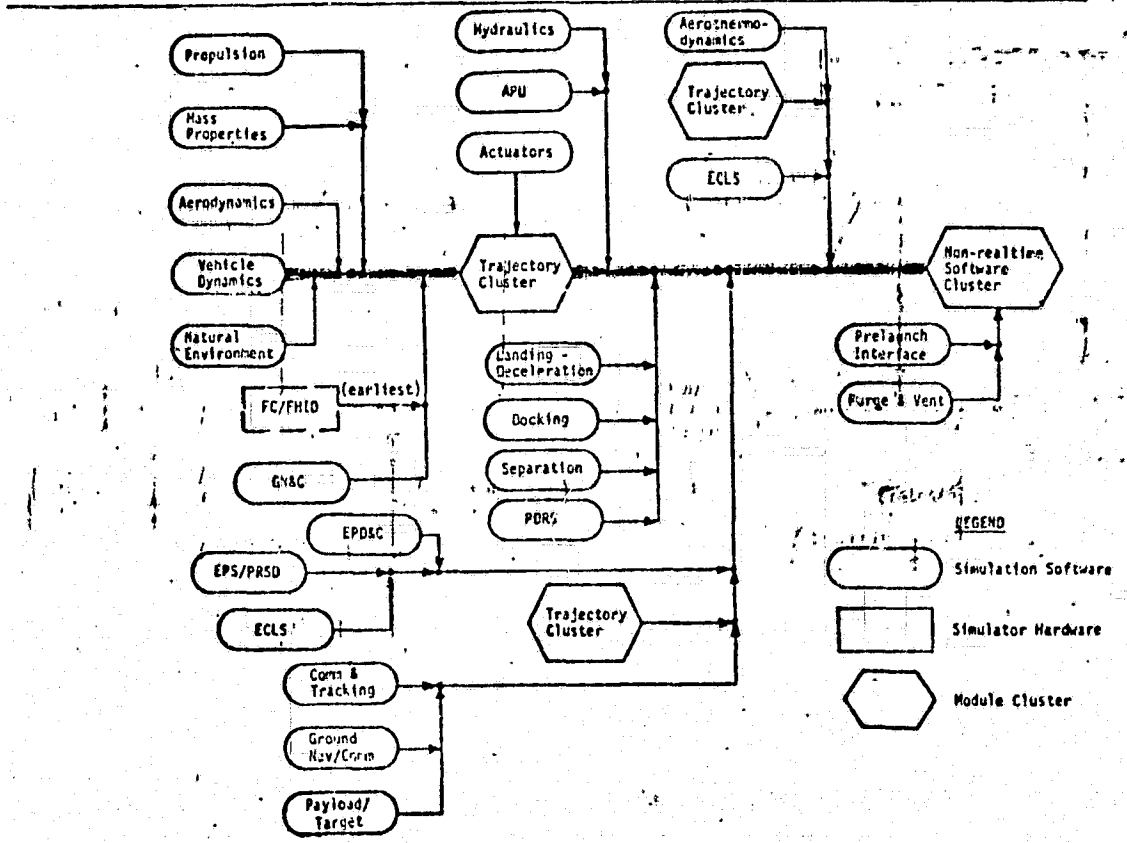
Considering the size and complexity of the simulators of interest in this study, we see simulator integration as a clustering process, rather than a pure "top-down" or "bottom up" sequence. That is, we expect that the simulation will be built up by integrating small clusters of naturally-interacting modules, then integrating these clusters with additional modules and with each other, until the complete simulation is assembled.

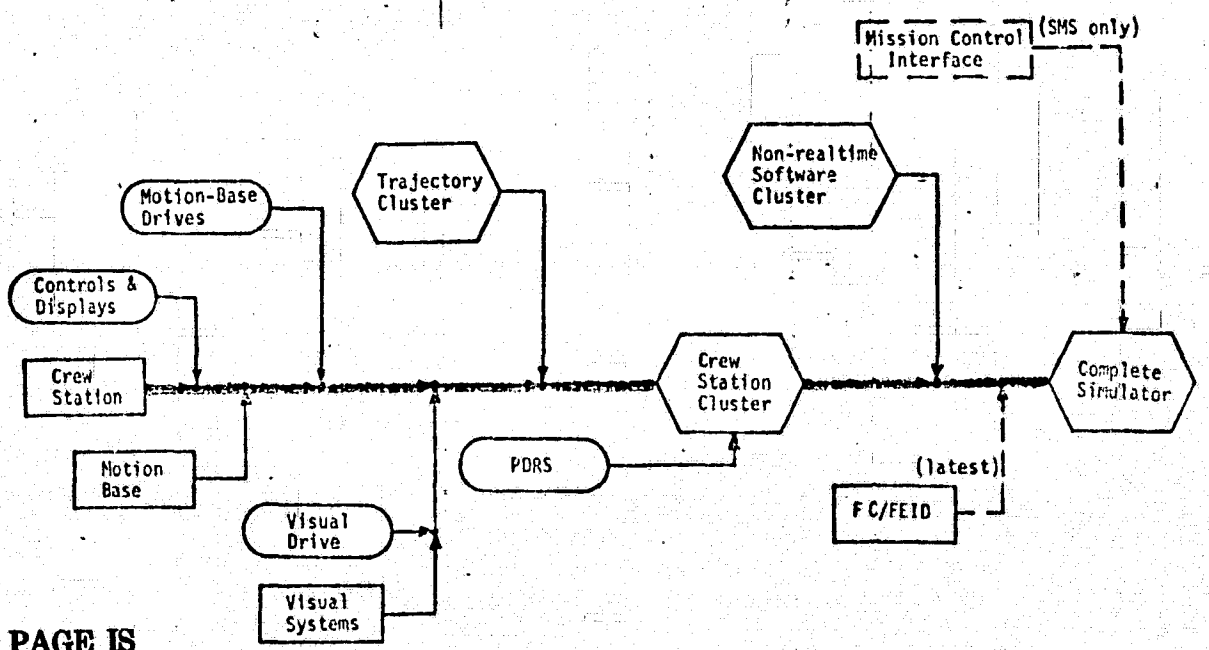
Having rather thoroughly studied module interactions in the course of our module-oriented efforts, we were able to derive a "natural" clustering sequence on the basis of the our module interface diagrams (samples of which were previously shown). The resulting integration/validation sequence is shown on the following page.

The top figure follows the software main line (the path emphasized by the heavy line), culminating in a non-realtime software cluster, which would be a complete batch-program spacecraft simulation. (If the Flight Computer/Flight Hardware Interface Device were integrated early, this software could actually be operated in a real-time mode.) The bottom figure follows the hardware main line, leading to a complete all-up simulator.

Definition of the natural integration/validation sequence should be of considerable aid in defining and controlling the simulator development/validation schedule for maximum efficiency.

A FIRST-CUT SIMULATION INTEGRATION SEQUENCE





ORIGINAL PAGE IS
OF POOR QUALITY

4.9 SPECIAL TEST REQUIREMENTS

Survey test operations as normally conducted:

- Component-level, subsystem/system-level, vehicle flight tests
- Consider purpose, time frame, type of data taken, typical documentation, potential problem areas

Identify test enhancements for validation purposes:

- Test operations
- Data-taking
- Data-handling and documentation

Consider management and liaison aspects of implementing desired enhancements.

BUT REMEMBER - the goal of testing is to prove out the system, not to support simulator development and validation.

3.4.9 Special Test Requirements

Earlier (Sect. 3.4.2), we indicated our assessment that test data was potentially a valuable source of standards of performance, despite certain practical problems to be expected in obtaining and handling the data. Later in the study we analyzed test operations and test data in greater depth, in order to identify methods to make more effective use of test data. We began with a survey of test operations as normally conducted and presently planned, based upon our experience in past aerospace programs and currently available test-related documents for the Shuttle program. We then defined general test program changes which would make test data more useful for simulation development and validation. Finally, we briefly discussed management-oriented approaches to the implementation of the desired changes.

Of course, test organizations have their own goals and problems; it is therefore unlikely that major test program changes will be made solely for the benefit of simulation development objectives. On the other hand, substantial simulation benefits may be realized with comparatively minor impact upon operations, and with the expenditure of minimal effort by simulation personnel. Clearly, test organizations cannot respond to the needs of simulation organizations unless those needs are made known. In summary, then, it appears than simulation projects have much to gain, and very little to lose, by making the effort to coordinate with test organizations.

Survey Normal Test Operations

COMPONENT-LEVEL TESTS

Expected to be most fruitful source of validation data:

- Best time-frame match
- Performance-oriented data

Potential problems in obtaining test data/documentation. Includes the following categories:

- Development and bench tests - prototype hardware.
- Qualification tests - go/no-go tests at spec limits.
- Acceptance tests - estimate "scatter" from data for multiple units.

3.4.9.1 Survey Normal Test Operations

In our survey of normal test operations, we again placed the most emphasis upon component-level tests. Data from these tests becomes available earlier, and tends to be more performance-oriented, than data from higher-level tests. As with all types of tests, availability of test documentation has historically been a problem.

A particular advantage of acceptance tests is the ability to obtain comparable data from a number of individual hardware units. This will provide an estimate of the inherent "scatter" in hardware performance, which serves as a guideline in the establishment of simulation fidelity criteria. An example of multiple-unit data (Skylab suit-cooling loop pumps) was previously shown.

Survey Normal Test Operations

SUBSYSTEM/SYSTEM-LEVEL TESTS

Late relative to initial simulation requirements; potentially useful for updates.

Complex setup; may be hard to duplicate on simulation.

Includes the following categories:

- Systems development tests - parametric data; systems may be incomplete.
- Integrated Systems Test - go/no-go data.
- Prelaunch checkout - go/no-go data; little access.

One problem with the use of subsystem-level and system-level test data is that it does not become available in time for initial simulation development and validation. Any performance-oriented parametric data available from such tests can of course be used for simulation updates. However, since many such tests only provide go/no-go (i. e., in-spec/out-of-spec) data, rather than actual performance parameter values, little use can be made of the results, no matter when they become available.

Vehicle flight test data is, of course, highly desirable, representing as it does the ultimate in realism. Since it becomes available so late in the program, it is usable for updates only.

One potential problem in making use of flight test data is the heavy data load, with hundreds or thousands of parameters recorded at high density over time spans of the order of minutes or hours. The flight-test organization will have to provide a high-capability data-handling system, to provide simulation organizations (and other users) with convenient access to the data of interest. Another problem in flight test data is the amount of uncontrolled and unknowable variation in the environment, hardware characteristics etc., generally preventing simulation data from matching flight data precisely.

The following table lists a number of Shuttle-related test documents which are currently available; many of these documents will be updated, augmented, or superseded as the program progresses.

Survey Normal Test Operations

VEHICLE FLIGHT TESTS

"Ultimate" in realism.

Time frame: updates only.

Complex; setup; may be difficult to duplicate on simulation.

Heavy data-load; good data-handling system required.

Survey Normal Test Operations

CURRENT SHUTTLE-RELATED TEST DOCUMENTS

MJ072-0004-3	Shuttle Master Verification Plan, Volume 3: Orbiter Verification Plan
MLO101-0001	Test Requirements: In-Process and Acceptance-Orbiter
SD72-SH-0009	Orbiter Quality Assurance Plan
SD72-SH-0112-6-II	RDD-Major Ground Test-Thermal Vacuum Test Program: CMS-RCS POD
SD72-SH-0112-12	RDD-Subsystem Ground Test-Docking Mechanism Dynamic Simulation
SD-72-SH-0112-13	RDD-Ground Subsystem Test-Orbiter/External Tank Separation Subsystem Test
SD72-SH-0112-18	RDD-Subsystem Ground Test-APU Integration Test
SD72-SH-0112-19	RDD-Subsystem Ground Test-ECLSS Test Article
SD72-SH-0112-21	RDD-Subsystem Ground Test-Escape System Test Article
SD73-SH-0062	Checkout Plan: Orbiter and Combined Elements Ground Operations
SD73-SH-0094	Manual, Technical and Non-Destructive Testing, Space Shuttle Specification for Preparation of
SD73-SH-0298	Avionics Development Laboratory General Test Plan
SD74-SH-0011 through SD74-SH-0049	Subsystem Certification Plans

Survey Normal Test Operations

VEHICLE FLIGHT TESTS

"Ultimate" in realism.

Time frame: updates only.

Complex; setup; may be difficult to duplicate on simulation.

Heavy data-load; good data-handling system required.

Survey Normal Test Operations

CURRENT SHUTTLE-RELATED TEST DOCUMENTS

MJ072-0004-3	Shuttle Master Verification Plan, Volume 3: Orbiter Verification Plan
ML0101-0001	Test Requirements: In-Process and Acceptance-Orbiter
SD72-SH-0009	Orbiter Quality Assurance Plan
SD72-SH-0112-6-II	RDD-Major Ground Test-Thermal Vacuum Test Program: CMS-RCS POD
SD72-SH-0112-12	RDD-Subsystem Ground Test-Docking Mechanism Dynamic Simulation
SD-72-SH-0112-13	RDD-Ground Subsystem Test-Orbiter/External Tank Separation Subsystem Test
SD72-SH-0112-18	RDD-Subsystem Ground Test-APU Integration Test
SD72-SH-0112-19	RDD-Subsystem Ground Test-ECLSS Test Article
SD72-SH-0112-21	RDD-Subsystem Ground Test-Escape System Test Article
SD73-SH-0062	Checkout Plan: Orbiter and Combined Elements Ground Operations
SD73-SH-0094	Manual, Technical and Non-Destructive Testing, Space Shuttle Specification for Preparation of
SD73-SH-0298	Avionics Development Laboratory General Test Plan
SD74-SH-0011 through SD74-SH-0049	Subsystem Certification Plans

Emphasis upon component-level tests.

- Inherently better source of validation data.
- Least complex and expensive.

Three-step approach to defining enhancements:

- Identify desired data - basically inputs and critical performance parameters.
- Develop an idealized test plan - optimal check cases; based upon prior experience/analysis/simulation.
- Define data recording and documentation desired - frequency, formats, accuracy.

3.4.9.2 Identify Test Enhancements

With an understanding of normal test operations, we can identify the types of changes which would be desirable to enhance the usefulness of test data for simulation development and validation. As before, our emphasis is upon conponent-level tests.

We suggest a three-step approach to defining an "idealized" test plan for any hardware component/subsystem of interest:

1. Identify desired data: In most cases, the data most desired for simulation--inputs and critical performance parameters -- is also the data most desired for hardware-evaluation purposes; thus there is a good chance of obtaining the basic data desired.

2. Develop an idealized test plan: This test plan will consist of a set of test conditions and operations which will generate check case data of the types desired to thoroughly exercise the simulation module or submodule corresponding to the hardware subsystem or component.

3. Define desired data recording and documentation: This will include specification of desired accuracy, data-recording frequency, presentation formats, and other factors involved in validation data-handling.

IMPLEMENTATION OF TEST ENHANCEMENTS

Early and continuing liaison with hardware development and test groups:

- Communicate needs for performance data.
- Identify desired data formats and documentation.
- Ensure receipt of available test data and documentation.
- Anticipate system changes resulting from outcome of tests.

Type of personnel desired - both simulation and test experience.

3.4.9.3 Implementation of Test Enhancements

Although simulation personnel will have no control over test organizations, and hence no assurance that desired changes will be implemented, they can vastly improve their chances for implementation by early and continuing liaison with hardware development and test organizations. At the very least, such liason will ensure that simulation groups will be kept up to date on test schedules, will know what test data are available at any given time, and will have access to test data which has been generated.

Ideally, the personnel involved in this liaison function should have both simulation and test experience. Since few engineers with this ideal background will be available, some cross-training will be required.

4.10 REFERENCE DATA FORMATS

Non-machine-readable reference data:

- DO NOT convert into machine-readable form.
- DO map the simulation data into a directly-comparable hard-copy format.

Machine-readable reference data - universal data format:

- Build into all new validation programs
- Conversion processors for existing programs and data files.

3.4.10 Reference Data Formats

Our study effort in the area of reference data formats included consideration of the data-handling aspects of both machine-readable data, such as card, tape, and disk files, and non-machine-readable hard-copy data, such as computer printouts, tables, graphs, and pictorial information.

We drew the following conclusions from our study of data-formatting problems:

1. In handling non-machine-readable data, one should not attempt complete conversion of the data into machine-readable form, in an attempt to automate the validation processing.

2. In generating and handling machine-readable data, maximum use should be made of standard or "universal" data formats.

3.4.10.1 Handling Non-Machine-Readable Reference Data

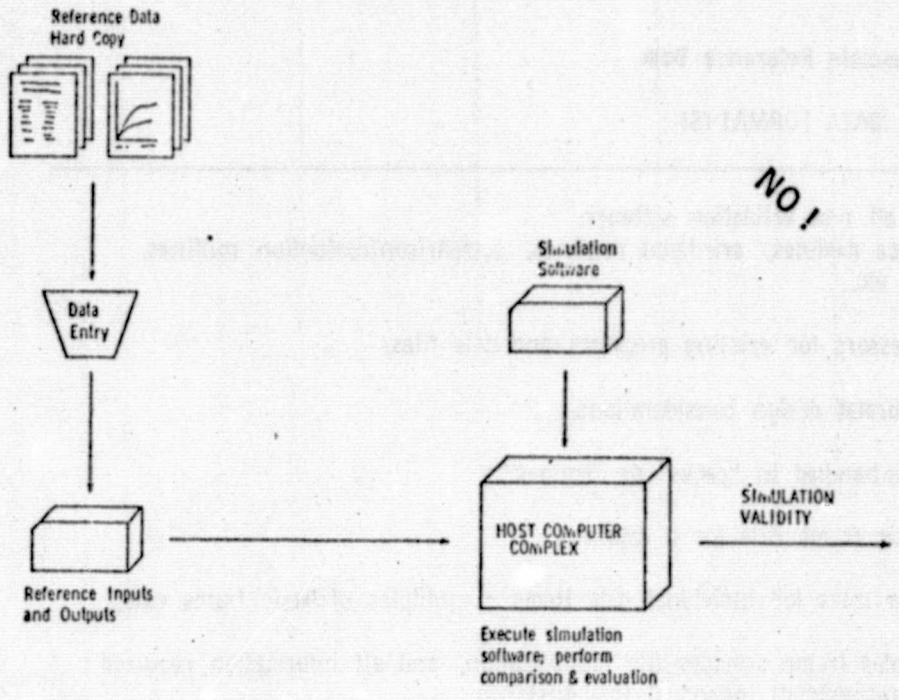
The following two figures schematically illustrate the recommended and non-recommended procedures for handling of non-machine readable reference data. In these figures, manual operations are denoted by trapezoids (ANSI standard flow-chart notation).

The upper figure shows manual conversion of the complete body of reference-data hard copy -- both inputs and outputs -- into machine-readable form; e. g., punched cards. Although this approach does allow automation of the data comparison and evaluation processes, we feel that the workload and error potential of the data entry process will more than offset any savings achieved by increased automation.

In the lower figure, the amount of manual data entry has been sharply reduced, since only the required input data is converted to machine-readable form. The simulation data is then output in tabular or graphical format corresponding exactly to the format of the original reference-data hard copy, thus allowing convenient manual comparison and evaluation with a minimum of error.

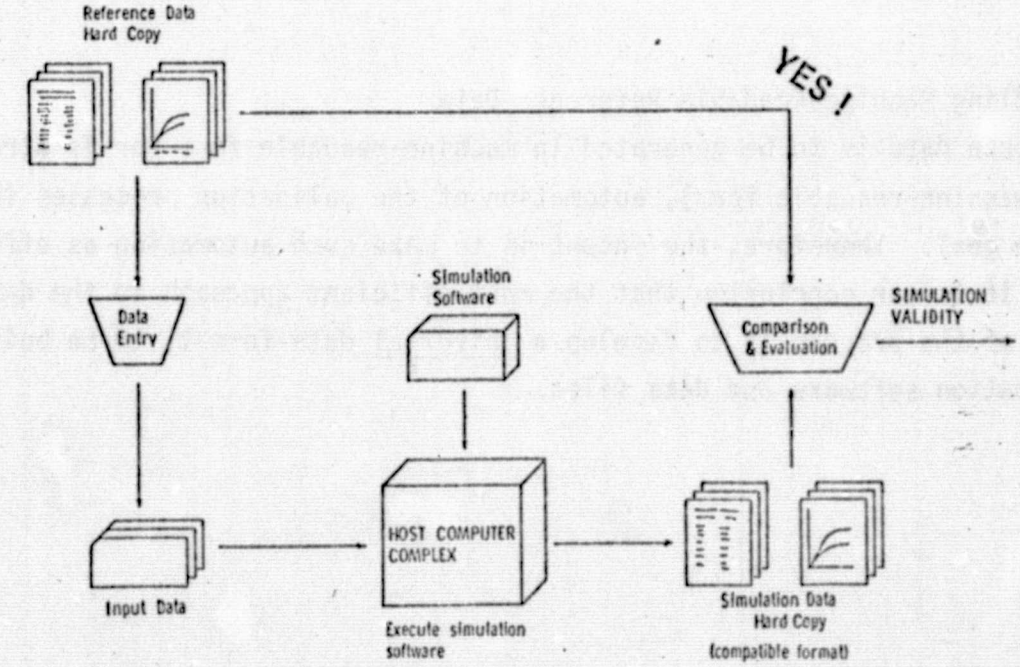
Non-Machine-Readable Reference Data

MANUAL DATA ENTRY FOR AUTOMATED COMPARISON



Non-Machine-Readable Reference Data

MANUAL COMPARISON OF COMPATIBLE HARD COPY



Machine-Readable Reference Data

UNIVERSAL DATA FORMAT(S)

Build into all new validation software:
reference modules, print/plot routines, comparison/evaluation routines, DBMS, etc.

Use reprocessors for existing programs and data files.

Universal format design considerations:

- Data handled in "pages" or "frames".
- Basic frame rate for a data file.
- Data rates for individual data items - multiples of basic frame rate.
- Header frame provides file identification, and all information required to reconstruct individual time-histories.

3.4.10.2 Handling Machine-Readable Reference Data

Where reference data is to be generated in machine-readable form (or is already available in machine-readable form), automation of the validation processes is an appropriate goal. Therefore, the intent is to make such automation as efficient as possible. It is our conclusion that the most efficient approach to the data-handling part of the process is to develop a universal data format, to be built into all validation software and data files.

The basic characteristics of this universal data format are illustrated on the following page.

The upper figure is a first-cut definition of the information which should be written on the header frame of each data file. This header frame completely identifies the data file, thus assisting in data-management operations. It also provides the data-handling routines with all information needed to strip out the desired time-histories from the file.

The lower figure shows how, using a "software commutator," variable data rates are achieved for individual parameters, while data frames are written out at a fixed rate. The frame time, of course, is the first parameter in every frame. Then, if a variable (e. g., x) appears in its assigned slot in every frame, its data rate is equal to the basic frame rate. If a variable appears in every second frame (e. g., y and z), or every third frame (e. g., p, q, and r), or every fourth frame (e. g., a, b, c, and d), or ..., then its data-rate will be one-half, one-third, one-fourth,... of the basic frame rate.

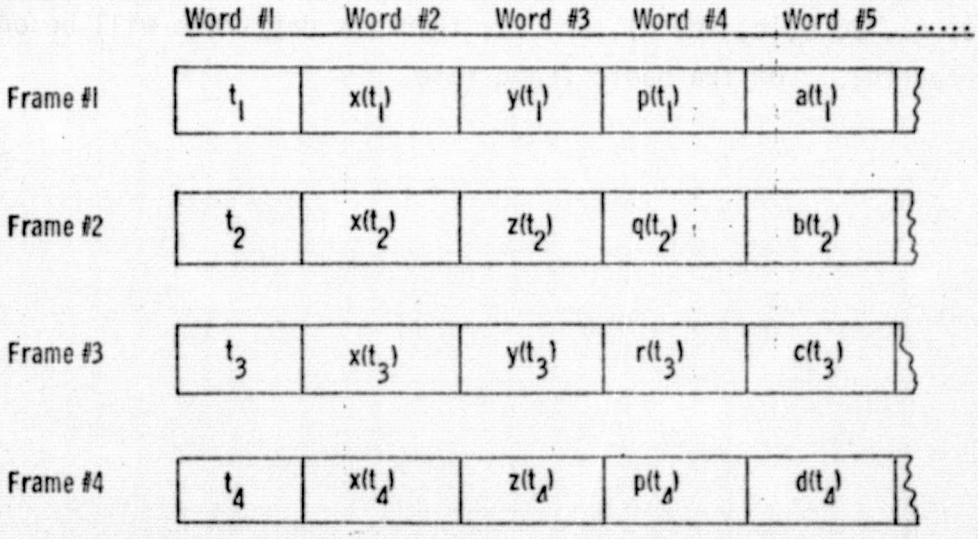
Universal Data Format(s)

HEADER FRAME INFORMATION

ITEM #	DESCRIPTION
1	Data file identification (fixed-length alphanumeric title)
2	Date file was generated
3	Type of data: reference, simulation, both
4-5	Identification of reference and simulation modules used to generate data
5	Data word length
6	K=Number of words per data frame
7	Nominal frame rate (frames per second)
8	M=Total number of frames (if known)
9	N=Total number of parameters in this file
10	Identification name or code for first parameter
11	Location of parameter #1 in each frame in which it appears
12	Word length for parameter #1 (several short parameters may be "packed" into a single word)
13	Frame frequency for parameter #1
14	(Same information for parameters 2 through N)
:	
4N + 9	

Universal Data Format(s)

FRAMING WITH VARIABLE DATA RATES



4.11 DATA BASE IMPACT

DATA BASE SCOPE AND STRUCTURE

- Machine-readable information and hard copy.
- Active and inactive materials.
- Accessed by mission, subsystem, date, time, etc.

DATA BASE MANAGEMENT SYSTEM (DBMS) REQUIREMENTS

- Capabilities for filing, retrieval, update, purge, physical-unit management, etc.
- Linkage with applications programs.
- Efficiency, reliability, stability, security.

DATA BASE IMPLEMENTATION CONSIDERATIONS

- "Make or buy" decision.
- Hardware, software, and personnel requirements.
- CODASYL standards.

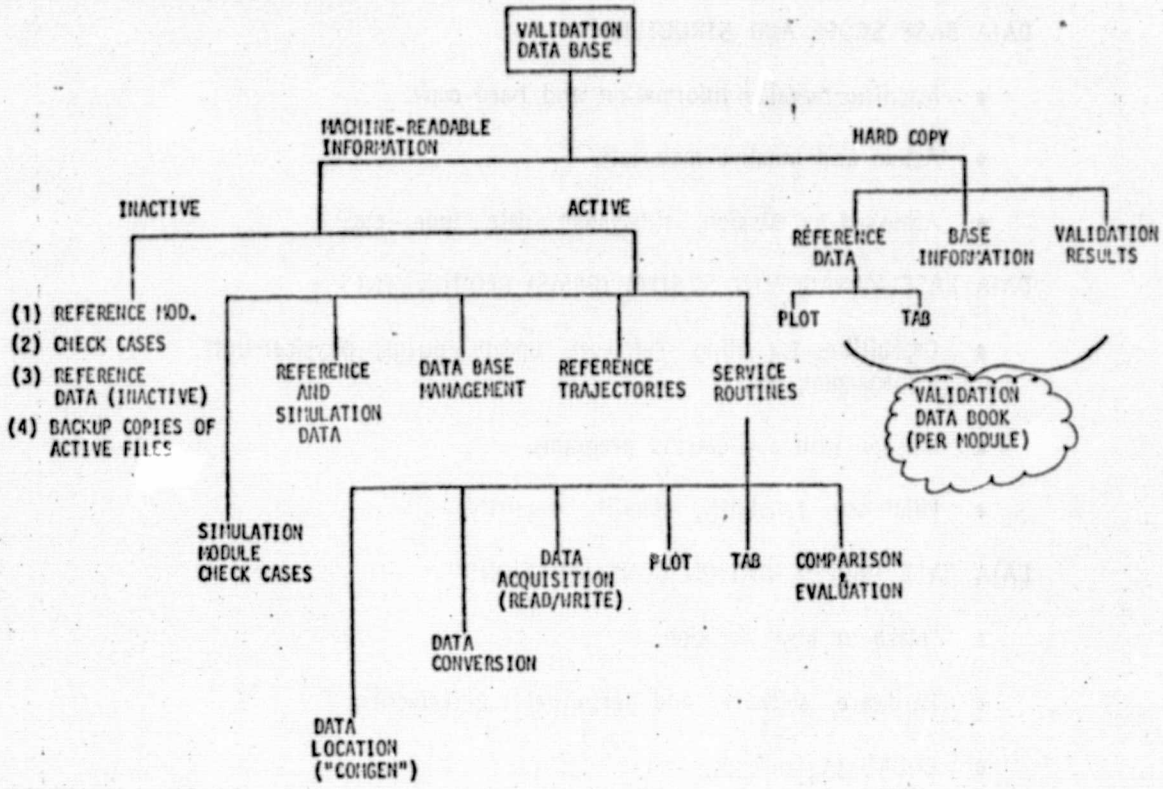
3.4.11 Data Base Impact

An extensive data base will be accumulated during the development and validation of any large simulation. (Indeed, the work performed in this study has already established an initial data base.) Although the data base impact assessment effort under WBS 2.0 was limited to validation-related data, it is well to remember that the simulation data base will also have to accomodate hardware-related data (see Section 2).

Our study of validation data base impact covered the basic scope and structure of the validation data base, requirements and design goals for the Data Base Management System (DBMS), and DBMS implementation considerations -- particularly the "make and/or buy" decision, which may have significant cost and schedule impact upon data base implementation.

4.11 Data Base Impact

DATA BASE SCOPE AND STRUCTURE



3.4.11.1 Validation Data Base Scope and Structure

The high-level structure of the validation data base is shown above in tree form. Here we define data base in the general sense, including both (a) machine-readable information which can be made directly accessible to the computer system and applications program, and (b) hard copy which is accessed by simulation personnel, using "pointers" generated by the DBMS in response to search queries.

Within the machine-readable category, the distinction is made between active and inactive files, which would be stored on media having different access time and cost parameters.

The core of the hard copy portion of the validation data base will be the Validation Data Book, which will be structured in terms of the vehicle subsystem/simulation module hierarchy shown in Section 3.3. Our Simulation Performance Validation Techniques Document (DRL-3) constitutes an initial version of this Validation Data Book, which will be updated and augmented as newer and more extensive data becomes available.

3.4.11.2 DBMS Design Requirements and Goals

The basic functions of a DBMS are to (a) store data and programs for later use, and (b) retrieve proper data and programs at the time they are to be used. Subsidiary functions, such as physical unit control and housekeeping capabilities, as well as operational considerations such as efficiency, reliability, stability and security, must also be considered before undertaking DBMS development and/or procurement.

To effectively perform its basic storage and retrieval functions, the DBMS must provide capabilities for the user to identify a data file by a variety of different parameters -- mission, subsystem, date, flight conditions, etc. -- at the time it is stored. It must then enable a user to search the data base in terms of any of these parameters, or combinations of these parameters, at the time he wants to retrieve the data. Once retrieved, the data should be made directly accessible to applications programs with minimal manual intervention.

Subsidiary functions include capabilities to update or purge obsolete files, move files between active and inactive storage, rearrange active files on physical media for greater efficiency, update the data dictionary, check file activity, generate notices to users, etc.

Operational efficiency, in terms of storage requirements and query processing time, will not be design requirements of overriding importance, since the DBMS itself will consume only a small fraction of the simulation project's computing resources. Reliability (freedom from errors) and stability (freedom from "crashes") will be more important, since the DBMS will in time become the user's major means of interfacing with the host computer for all types of simulation and validation activity.

Security, in the sense of prevention of unauthorized access to programs and data, may be important for simulators used to support DOD missions. Security in the sense of prevention of unauthorized destruction or modification of programs and data (essentially configuration control) can be provided by a fairly simple password system, and further ensured by maintenance of backup copies of essential files.

4.11 Data Base Impact

DATA BASE IMPLEMENTATION

"MAKE AND/OR BUY" DECISION

- DBMS development is a big task in itself.
- DBMS capability will be needed early in the development/verification phase - to support both hardware and software.
- DBMS development is much different from simulation development - concepts, machine requirements, language requirements, personnel requirements.
- Many proprietary packages are available - $5000 to $200,000+. (Check ICP, Data Pro, Auerbach, etc.)

CODASYL STANDARDS

- CODASYL has been studying the DBMS problem since 1970 or so.
- Applicable to either make or buy software.
- Enhances portability, speeds development and verification of DBMS.

3.4.11.3 Data Base Implementation

Probably the most important step in DBMS implementation is the first step: the "make and/or buy" decision. (The "make and buy" approach would be to procure a basic system for interim use, while proceeding with in-house development of a system of expanded capability.) There is a potential danger in jumping into in-house development of DBMS, without considering procurement of an existing package: i.e., the implementation may prove more difficult than anticipated, especially if the development staff has prior experience only in simulation development projects. Since a working DBMS will be needed throughout the simulator development and verification phases, to support both hardware and software, slippage in its implementation can impair the efficiency of the entire project.

Many proprietary packages (e.g., Mark IV, ADABAS, System 2000) are available, providing a broad range of capailities, at prices from $5 000 to upwards of $200 000. Many of these packages are listed in the ICP Quarterly, and rated by non-vendors such as DataPro and Auerbach, as well as in trade publications such as Datamation and Computer Decisions.

Whether the DBMS is procured and/or developed, it will be advantageous to conform to the data base standards defined by the Committee on Data Systems Languages (CODASYL). The table below provides CODASYL-standardized definitions for a few key data base concepts.

SOME BASIC CODASYL DATA BASE DEFINITIONS

Data Item	The smallest data base unit referenced by an assigned name.
Record	A collection of one or more data items; contains a named description of data items and attributes.
Set	Establishes a named logical relationship between two or more record types; the basic data base building block which allows the data base designer to establish complex data structures.
Area	A named subdivision of logical address space in a data base; each record must reside in an area which contains one or more records.
Schema	A complete description of all data items, record types, set types, and areas which exist in a data base; the foundation of a data base dictionary system.
Subschema	A logical subset of the schema which names only those record types, set types, and areas that are accessed by one or more specific applications programs.
CALC	Refers to one common method of record placement and retrieval within a data base; provides an access point via a "hashing" algorhytm.

5. METHODS FOR VALIDATING PERFORMANCE

The validation process consists of:

- Exercising a simulation with properly-chosen inputs,
- Collecting its output response data,
- Comparing the simulation data with reference data to evaluate simulation fidelity.

Section 5 is concerned with guidelines, techniques, and support software for the validation process.

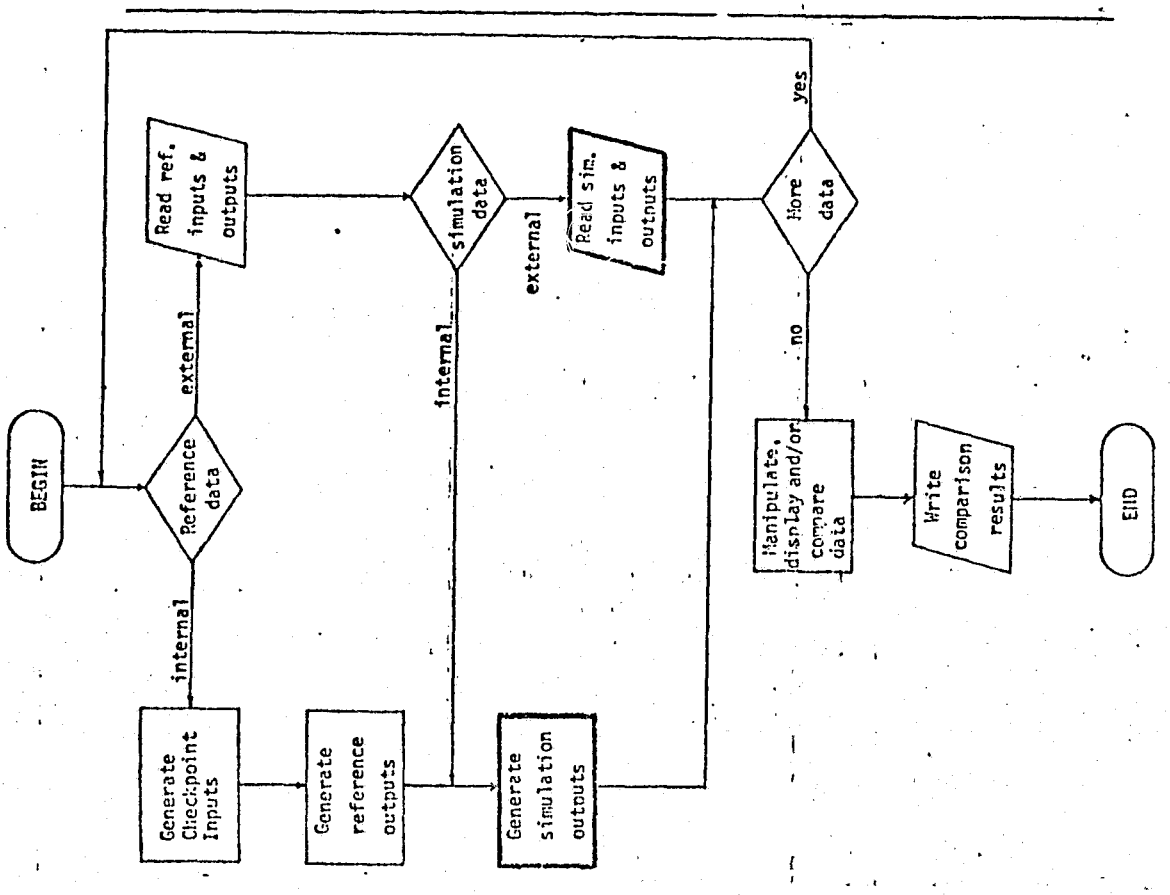
3.5 METHODS FOR VALIDATING PERFORMANCE

Performance validation methods have been touched upon in the preceding sections -- either in the context of a particular simulation module, or from the viewpoint of their influence upon data-handling methods. It is now appropriate to provide an in-depth treatment of validation techniques per se.

The total process of performance validation, as previously described in Sect. 3.1, consists of exercising a simulation (an individual module, a module cluster, or an integrated simulation) with appropriate inputs, collecting the outputs which it generates in response to those inputs, and performing comparison and evaluation operations to assess the simulation fidelity. This part of the study was concerned with definition of guidelines, techniques, and support software for the validation process.

5. Methods for Validating Performance

5.1 VALIDATION SOFTWARE STRUCTURE



3.5.1 Validation Software Structure

3.5.1.1 Validation Executive Overall Flow

The skeleton of an overall validation executive routine is indicated above. The emphasized blocks are the sources of simulation data: either the on-line exercise of a simulation, or access to a previously-generated file of simulation data. The other blocks indicate sources of reference data (again, either on-line or file access), as well as validation service routines required to efficiently perform validation processing.

Validation Software Structure

VALIDATION SERVICE ROUTINES

Checkpoint generation routines (cf. Section 5.3):

- Generate sets of input data (discrete/continuous).
- Systematic or random variation.
- Keep number of check cases reasonable.

Simulation software module drivers:

- Perform I/O and module linkage ("patchboard" analogy).
- Use COMGEN or equivalent support software to automate the module-linkage process.

External data-file handling routines:

- Reference and/or simulation data may be prerecorded on tape/disk.
- Strip desired parameter time-histories for driving and comparison.
- Use previously-discussed universal format.

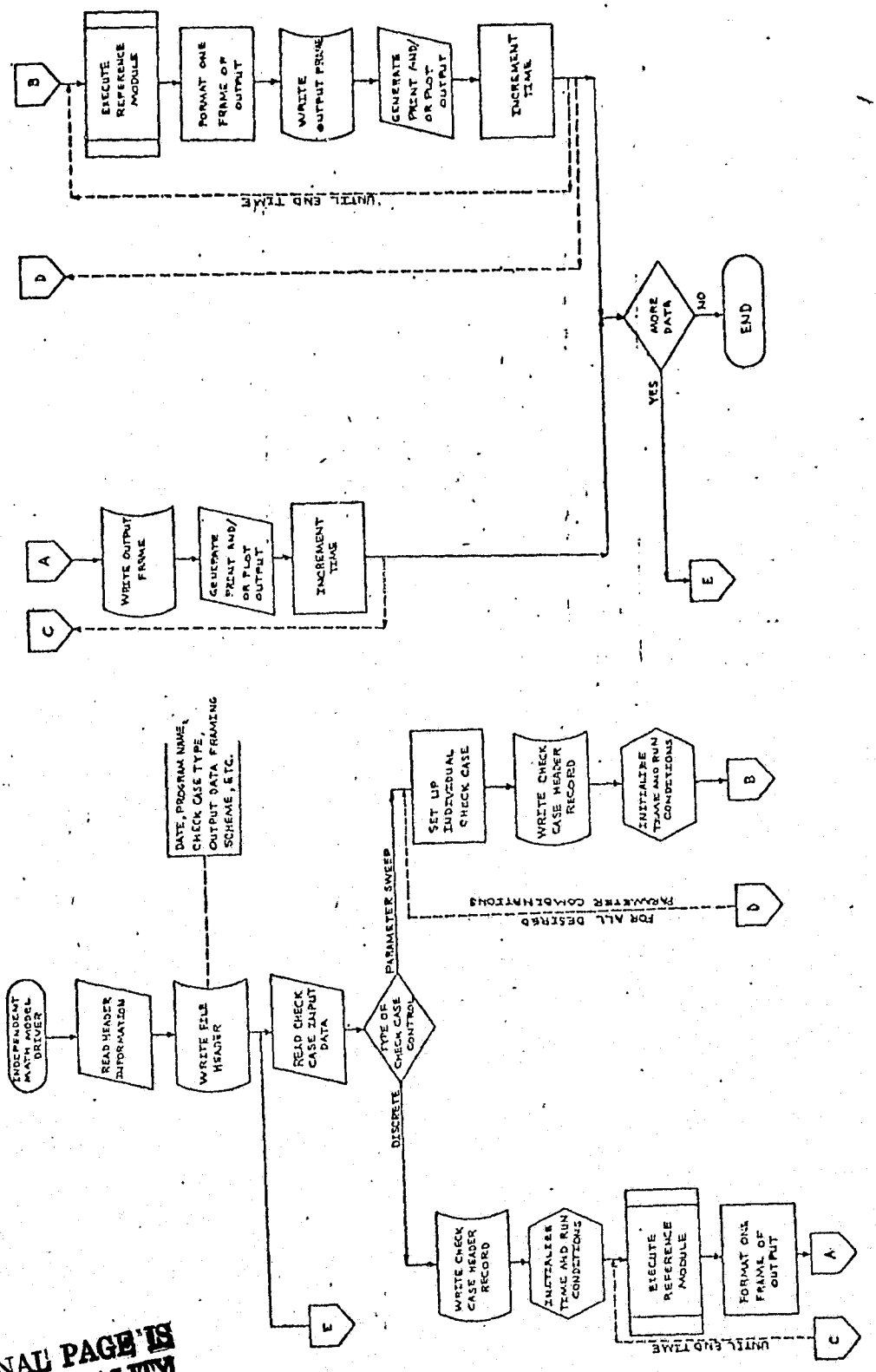
3.5.1.2 Validation Service Routines

Functions and properties of three classes of validation service routines (checkpoint generators, drivers, and file-handlers) are briefly discussed above. Functions and properties of the DBMS were discussed in Sect. 3.4.11.

An important design requirement for all classes of service routine is generality. There will be so many simulation modules and data files involved in the development of a large-scale spacecraft simulation that the workload imposed by generation of "customized" service routines for each module and data file would be prohibitive. This, of course, was the primary rationale behind our recommendation to develop a universal format for all reference-data and simulation-data files.

Multiple functions may be combined in a single service routine, which will sometimes be desirable. The figure on the following page shows a generic math flow for a combined checkpoint generation/module driver routine. Such a composite routine would provide capabilities for input of discrete (manually-selected) checkpoints, generation of parameter-sweeping sets of checkpoints (see Sect. 3.5.2), module interfacing, and data-file formatting.

MATH FLOW FOR CHECKPOINT-GENERATION/DRIVER ROUTINE



5.2 SIMULATOR INTEGRATION/VALIDATION CONFIGURATIONS

Validation can be performed at the following levels of simulation integration:

- Isolated module plus driver - driver must provide all inputs to setup and execute.
- Module "cluster" plus driver.
- Modified all-up simulator:
  - -- "Probes" or test points for I/O of internal variables (e.g., "canned man")
  - -- "Blocks" to simplify module interaction and error propagation.
- Normal all-up simulator:
  - -- Specially-constructed check cases
  - -- Realistic check cases

AREA OF EMPHASIS

3.5.2 Simulator Integration/Validation Configurations

From the first, it has been evident that validation must be performed at all stages of simulator integration, from the isolated operation of the smallest simulation module up to the final validation of the complete all-up simulator. To intelligently plan the total validation program, it is necessary to further examine the various alternatives, and determine the area of greatest emphasis: i.e., the stages of integration at which validation effort should be concentrated to maximize the efficiency of the total process.

The four basic validation configurations -- isolated module, module cluster, modified all-up simulator, and normal all-up simulator -- are defined above. Pros and cons of performing validation in each of these configurations are tabulated on the following pages. In examining the pros and cons, we have given the most weight to the following three points:

- o How thoroughly can we exercise the simulation?
- o How readily can we verify module interactions?
- o How much auxiliary software (drivers, etc.) is required to support validation?

As a result, we have concluded that the major area of emphasis should be the intermediate stages of integration: module clusters and modified all-up simulators. A particular modification of high potential value is the use of a "canned man": i.e., insertion of pre-recorded inputs downstream of the manual controls, providing more controlled and repeatable exercise of the simulation than would be possible by actually operating it in a man-in-loop mode.

Validation Configurations: Pros and Cons

ISOLATED MODULE

PRIMARY OBJECTIVE:

To validate detailed simulation capabilities of each Module.

ADVANTAGES:

- Easiest to devise check cases for which correct answers are known exactly.
- Easiest to fault-isolate following check-case failure.
- Easiest to ensure thorough exercise of module.
- Can be executed offline (batch runs).

DISADVANTAGES:

- Generation of each driver represents extra coding and debugging effort. (Development of "general-purpose" drivers will reduce the cumulative effort, but some tailoring of the driver to each module under test will still be necessary.)
- For "trivially" simple modules, the validation benefits may not be commensurate with the effort of building the driver and setting up and executing the check cases.
- Does not explicitly verify module-to-module interfaces.

Validation Configurations: Pros and Cons

MODULE "CLUSTER"

PRIMARY OBJECTIVE:

To verify interfaces among highly-interactive modules.

ADVANTAGES:

- Driver can be simplified because some required data is supplied by modules in the cluster.
- Less cumulative coding and debugging effort devoted to generation of drivers; a single driver serves validation of multiple modules.
- All exercises are "non-trivial".
- Verifies some module-to-module linkage.
- Can be executed offline (batch runs).

DISADVANTAGES:

- May be difficult to thoroughly exercise and validate all modules in the cluster.
- May sometimes be difficult to devise test cases for which correct answers are known exactly.
- May sometimes be difficult to fault-isolate following check-case failure.

MODIFIED ALL-UP SIMULATOR

PRIMARY OBJECTIVE:

To simplify signal/error propagation for system-level validation.

ADVANTAGES:

- No coding and debugging of drivers.
- Allows extensive verification of module-to-module linkage.

DISADVANTAGES:

- May be a complex, laborious process to modify and restore simulation, and to set up for check-case execution.
- Potential for later difficulties if all modifications are not restored to normal configuration.
- Requires realtime operation of dedicated system.
- Difficult to know correct answers for all variables which will be exercised by each check case.
- Difficult to fault-isolate following check-case failure.
- Few individual simulation modules will be thoroughly exercised.
- Difficult to obtain repeatable results from man-in-loop operation.

Validation Configurations: Pros and Cons

NORMAL ALL-UP SIMULATOR

PRIMARY OBJECTIVE:

To validate dynamic adequacy of total simulator system.

ADVANTAGES:

- No coding and debugging of drivers.
- Allows complete verification of hardware and software interfaces.
- Succesful operation builds confidence in complete simulator system.
- Contributes to simulator acceptance.

DISADVANTAGES:

- May be a complex, laborious process to set up simulator for check-case execution.
- Requires realtime operation of dedicated system.
- Difficult to know correct answers for all variables exercised by each check case.
- Very difficult to fault-isolate following check-case failure.
- Few individual simulation modules will be thoroughly exercised.
- Difficult to obtain repeatable results from man-in-loop operation.

5.3 CHECK CASE FORMULATION

PRIME CONSIDERATIONS:

- THOROUGHNESS
  - -- Individual and combined variation of discrete and continuous variables.
  - -- Exercise all operational modes.
  - -- Sweep out entire range of operation: normal/abnormal/failures

  (end-state considerations)
- EFFICIENCY
  - -- Minimize resources expended for a given level of confidence
- ORDER OF EXECUTION
  - -- Minimize resources to reach most likely outcome at decision points

  (as a function of time)

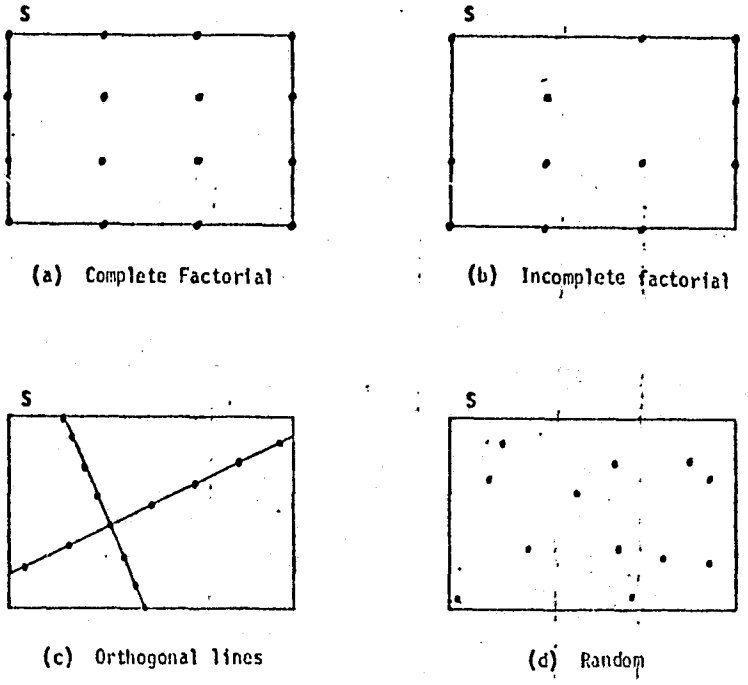
3.5.3 Check Case Formulation

Whether checkpoints are manually selected or automatically generated, the prime considerations involved are thoroughness of exercise of the simulation and efficiency. Normally, these are thought of as "end-state" considerations. That is, when the validation process has been completed, how much confidence do you have in the validity of the simulation, and what resources (manpower and computer time) have you expended to attain that level of confidence?

Another aspect of efficiency comes into play when we attempt to define the best order in which check cases should be run. Recognizing that each check case represents a decision point (i.e., the results of the check case will either be acceptable or unacceptable), we can consider, as a function of time, the resources we will have expended to reach the most likely outcome at each decision point. We shall see that this viewpoint leads to directly opposite check-case ordering strategies for initial validation and revalidation.

Check Case Formulation

BASIC CHECKPOINT-GENERATION METHODS
(Two-dimensional illustrations)



3.5.3.1 Basic Checkpoint Generation Methods

Four basic parameter-sweeping checkpoint-generation methods, for either manual or automatic application, are schematically depicted above for a two-dimensional "parameter space" -- i.e., for a hypothetical simulation module having only two inputs.

What these simple two-dimensional figures cannot adequately show, however, is the explosive increase in the number of checkpoints required for the conventional (complete factorial) approach, as the number of input parameters increases. For example, consider a simulation module having six continuous inputs and eight discretes (not at all unrealistic). Suppose we wanted to input a high, medium and low value for each continuous input, and an on and off (or zero and one) value for each discrete. It would then take 186,624 distinct checkpoints to run all combinations of inputs. This phenomenon, often called the "curse of dimensionality", makes it essential to use more efficient checkpoint-generation methods.

3.5.3.2 Order of Execution of Check Cases

The following two figures define general confidence relationships, and our resulting recommendations for check case ordering, for initial validation and revalidation of a module or integrated simulation.

Check case ordering for initial validation of a new simulation (upper figure) should be based upon the pessimistic assumption that the module will fail to perform acceptably for some or all conditions, thus temporarily halting the validation process while corrections are made. Therefore, check cases should be ordered on the basis of gradual expansion of the operational envelope, starting with verification of minimal operational capability and leading up to more rigorous exercise. This will achieve our stated objective of minimizing the resources expended up to the time of failure.

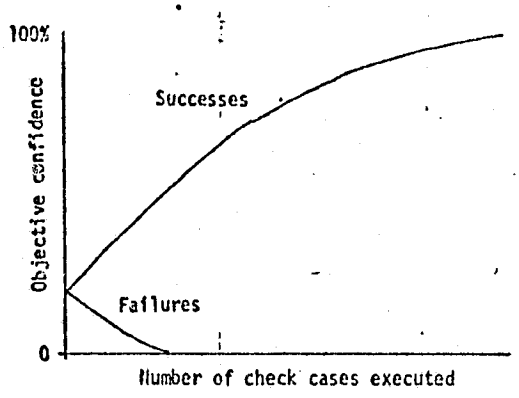
For revalidation of an existing, previously-validated simulation which has undergone some type of modification (lower figure), check case ordering should be based upon the optimistic assumption (based upon its prior "track record") that it will pass all its check cases. Therefore, the most rigorous check case(s) should be presented first. Validation will thus be completed in the shortest possible time if, as expected, the most rigorous check case(s) execute successfully. If this is not the case, a process of contraction of the envelope is followed, until the operational limit of the simulation is discovered, and the cause of unacceptable performance is determined and corrected.

Check Case Formulation

ORDER OF EXECUTION FOR INITIAL VALIDATION

Process of envelope expansion.

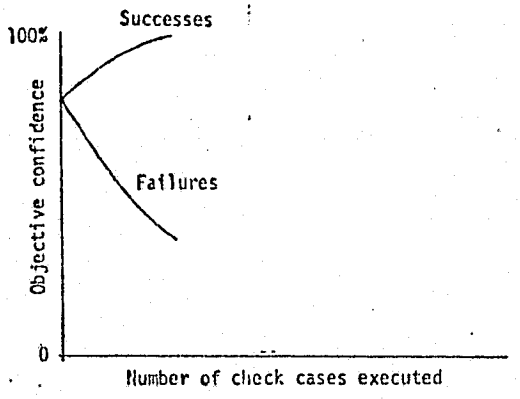
Based upon pessimistic assumption.



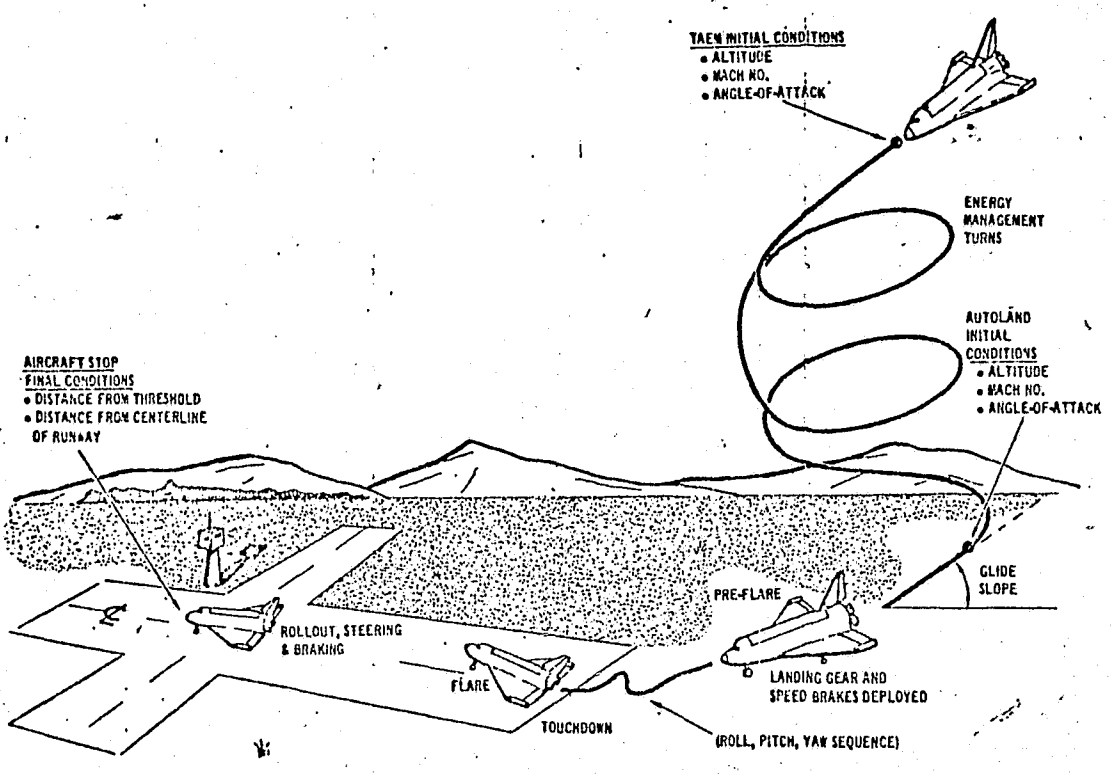
ORDER OF EXECUTION FOR REVALIDATION

Process of envelope contraction.

Based upon optimistic assumption.



EXAMPLE RIGOROUS CHECK CASE FOR REVALIDATION



An example of a rigorous check case -- a Shuttle mission segment consisting of energy-management glide, approach and landing -- is shown above. Operations to be performed, and critical variables to be monitored, are indicated on the figure. The high-rate roll-pitch-yaw sequence shown late in the approach is designed to verify the synchronization of visual and motion systems with the vehicle dynamics and crew-station displays.

5.4 REALTIME DATA ACQUISITION AND FORMATTING

Special software required to support validation of all-up simulator - acquire performance data for validation post-processor.

Must be integrated with the realtime executive, simulation common storage, computation cycle, and host-computer I/O system capabilities.

Must not interfere with simulator operation (basically a matter of priority).

(A similar system has been integrated with the SPS as part of our CPDT study.)

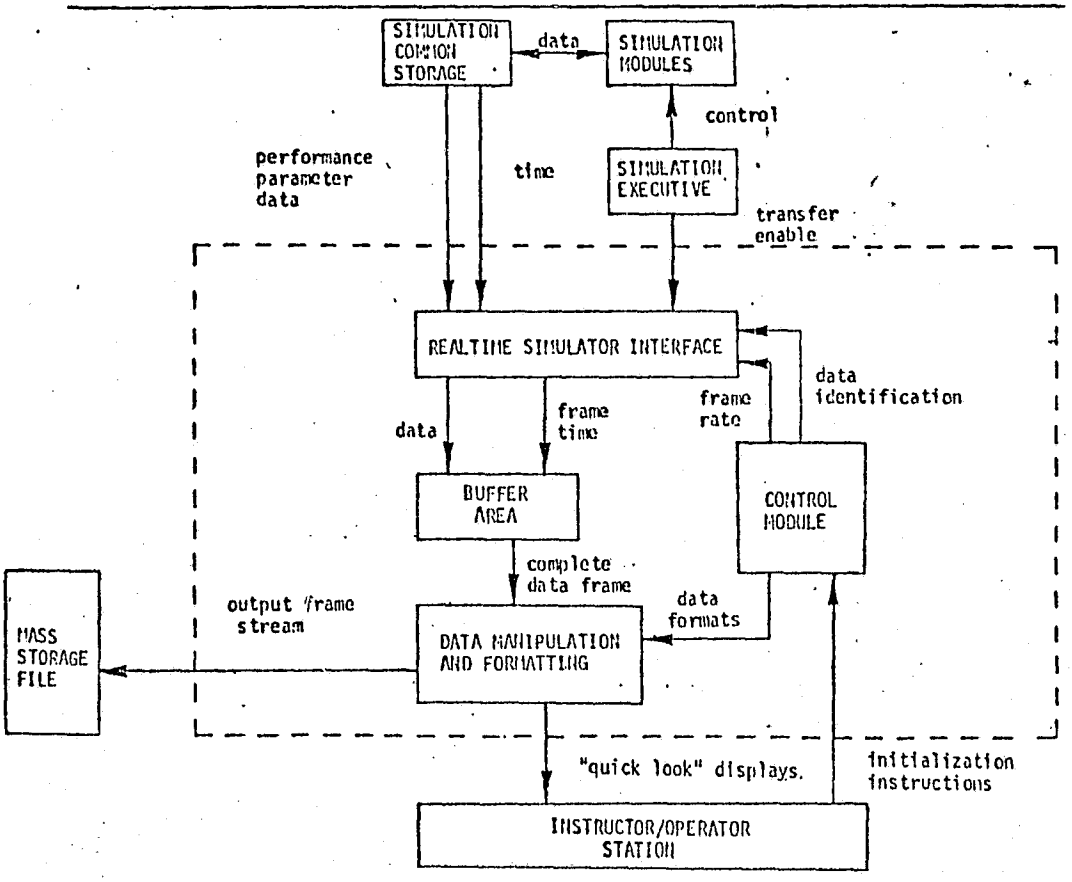
3.5.4 Realtime Data Acquisition and Formatting

In validating an individual module or a cluster of modules, the driver routine exercises control over the module or cluster, and performs data input and output. For validation of integrated simulations, the situation is reversed: the simulation is under control of its own executive and the computer operating system; the data-acquisition routines are in turn under control of the simulation executive, and data-acquisition operations are subordinate to simulation operations.

The data acquisition routines must be integrated with the simulation's realtime executive and common storage, and their operation must be constrained by the computation cycles of the realtime simulation and by the host-computer I/O capabilities.

The overriding design requirement is that data acquisition for validation purposes must not interfere with realtime simulation, neither causing the simulator to lose synch with realtime or preventing simulation modules or essential service modules from executing at their assigned rates. This is basically a matter of priority assignments. That is, the execution priority assigned to the validation data-acquisition module must be low enough that, if the simulator has difficulty keeping up with real time, the data acquisition operation will be the first thing sacrificed. (Provisions should be made to flag any resulting "drop-outs" on the validation data file, indicating that the validity of the data may have been compromised.)

REALTIME DATA ACQUISITION SYSTEM



The basic elements and interfaces of the realtime data acquisition system are shown above. Under control of the simulation executive, simulation data (times and performance parameter values) are passed to the data-acquisition system, and stored in a buffer area. When all data required for complete "frame" (see Section 3.4.10) are available in the buffer, the data frame is assembled and formatted as directed by the control module, based upon instructions previously input by the operator through the instructor/operator station. The formatted data frames may be transferred to the instructor/operator station for quick-look displays, and/or put out on a mass storage device for later post-processing.

Further discussion of data-acquisition system requirements and design characteristics is provided in DRL-3. That discussion is based in part upon information received from our Crew Procedures Development Techniques study staff, who designed and built a similar system for the SPS.

5. Methods for Validating Performance

5.5 COMPARISON METHODS AND CRITERIA

(Manual and automated methods)

DEVELOPMENT GOALS:

- For manual validation - develop display formats to minimize workload of manual validation, maximize reliability and consistency.

- For automated validation - find or devise comparison criteria which will give the same evaluation results as experienced simulation engineers or flight crews (i. e., the "right" answer).

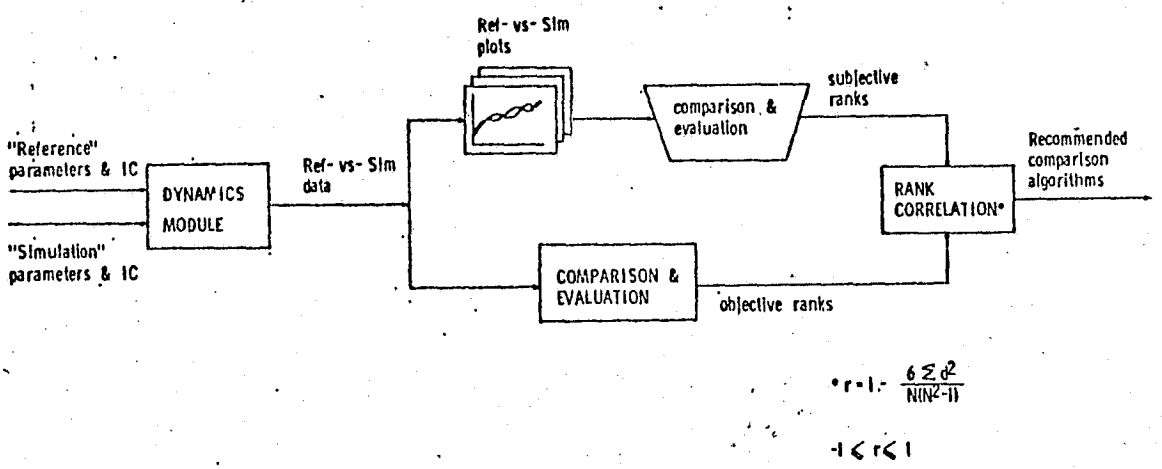
  (Our approach: conduct a simple experiment to obtain empirical data on this problem.)

3.5.5 Comparison Methods and Criteria

As discussed previously (e.g., Section 3.4.10), the comparison and evaluation operations involved in simulation validation may be implemented by either manual or automated methods. Since the ultimate test of a simulation's validity is acceptance by its end users (spacecraft engineers and/or flight crews), it is essential that any automated validation technique give results which are consistent with the subjective evaluation of the end users. As part of this study, we conducted a simple human-factors experiment to investigate the agreement between manual and automated evaluation results for a simple simulation-validation problem.

MANUAL/AUTO EVALUATION EXPERIMENT

(Using a set of ten cases of dynamical data.)



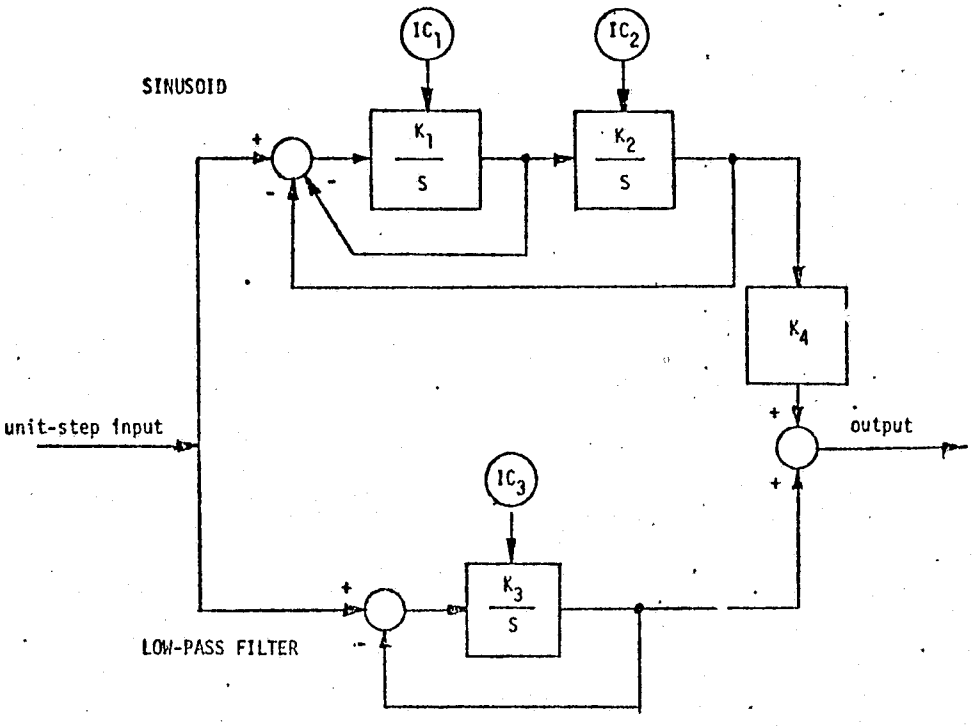
3.5.5.1 Experiment Description

The experimental process is depicted schematically in the above figure. A single "reference" time-history and ten "simulation" time-histories were generated by varying the parametric input to a simple dynamics module. The simulation validity (agreement between "reference" and "simulation" data) was then evaluated in two ways:

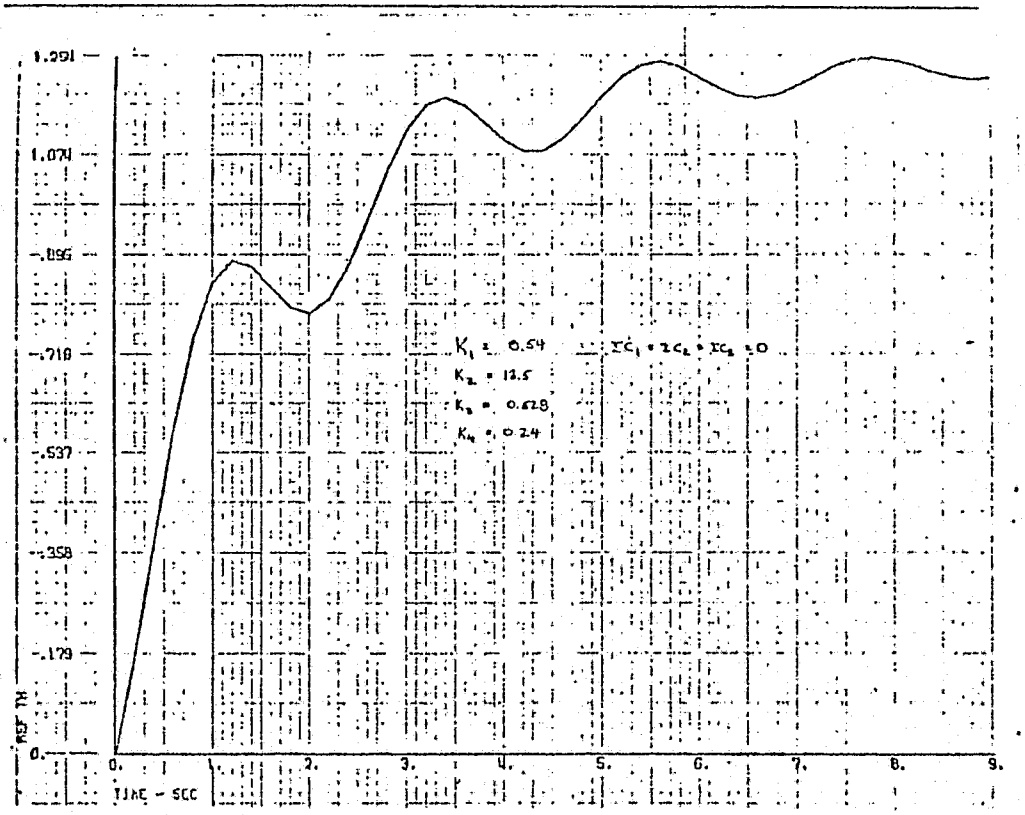
- o A set of time-history plots was generated and evaluated manually (trapezoid) by a panel of experimental subjects.
- o The same data was evaluated automatically (rectangle), using a variety of candidate comparison algorithms.

Finally, the agreement between manual and automatic comparison results was determined for each candidate algorithm, using a "rank correlation" process (for which the formula is shown above). The comparison algorithms which gave the highest positive correlation with our engineers' subjective evaluations thus became the recommended algorithms for automated validation.

DYNAMICS MODULE FOR EXPERIMENT



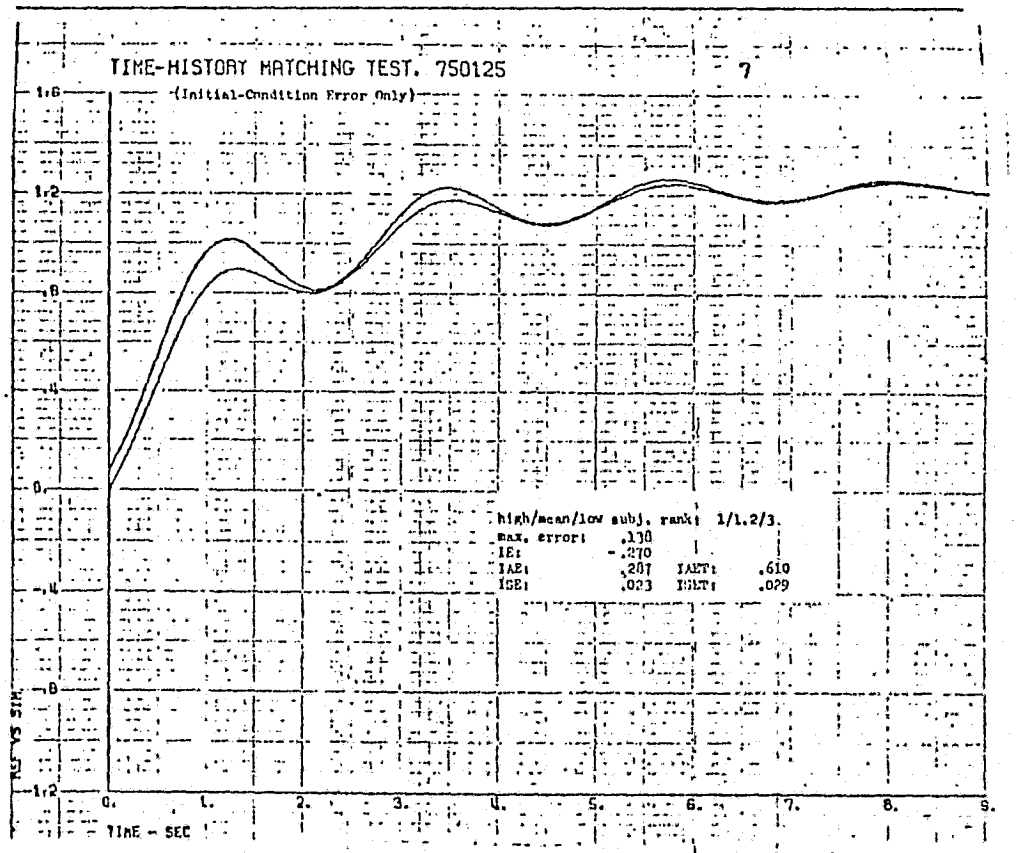
EXPERIMENT "REFERENCE" DATA



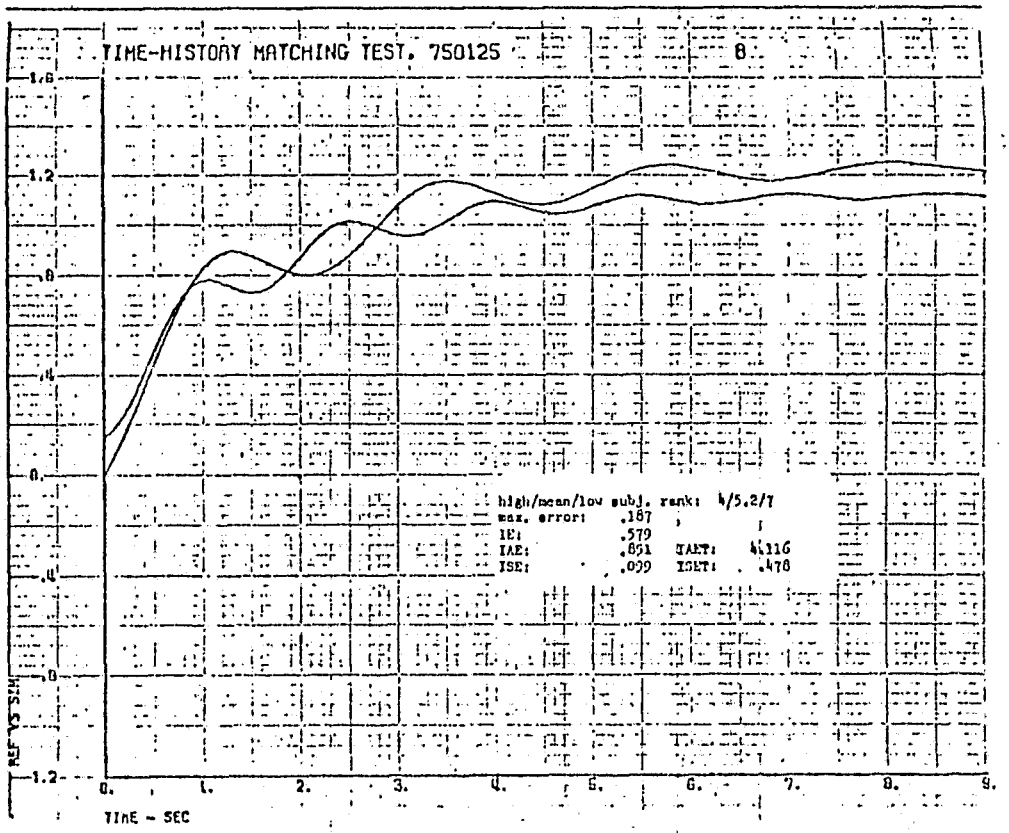
The dynamics module used to generate the experiment data (upper figure) consisted of a simple linear system: a decaying sinusoid added to a low-pass filter output. When forced with a step function, this system gave the "reference" output shown in the lower figure, for the choice of parameters and initial conditions shown on the figure.

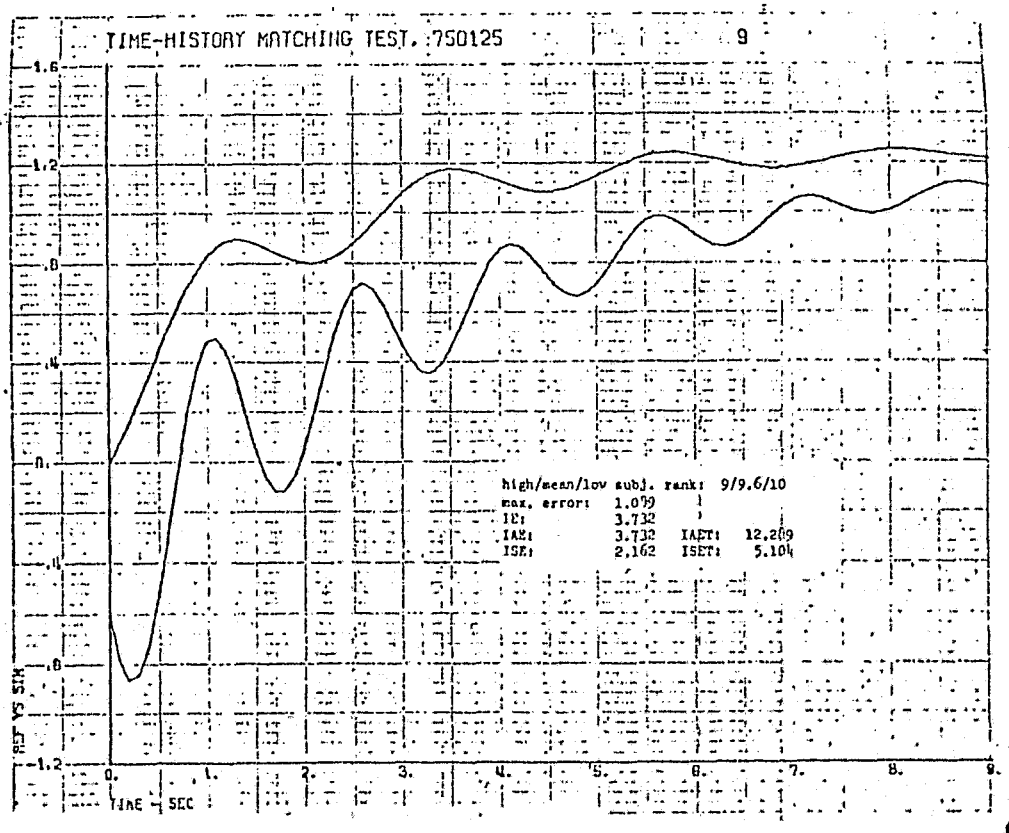
Ten sets of "simulation" data were generated by adding random perturbations to the parameters and/or initial conditions used to generate the "reference" data. Only three examples are shown here: the sample which was judged the best match by our panel of experimental subjects is shown below: a sample from the middle of the range, and the sample judged the worst match by the panel are shown on the next page.

EXPERIMENT "REFERENCE" VS "SIMULATION" DATA



Reference vs Simulation Data, continued





MANUAL COMPARISON: SUBJECTIVE RANKS

CASE	RANKING BY INDIVIDUAL SUBJECTS										RANK SPREAD		
	"High" Experience					"Low" Experience					High	Mean	Low
	1	2	3	4	5	6	7	8	9	10			
1	9	9	10	9	9	10	9	9	10	10	9	9.4	10
2	2	2	2	2	2	2	2	2	1	2	1	1.9	2
3	5	7	8	6	6	5	6	7	5	8	5	6.3	8
4	8	5	5	8	7	8	4	4	2	5	2	5.6	8
5	7	6	7	7	8	7	8	8	6	7	6	7.1	8
6	4	3	3	3	3	4	3	3	4	3	3	3.3	4
7	1	1	1	1	1	1	1	1	3	1	1	1.2	3
8	6	4	4	4	5	6	7	5	7	4	4	5.2	7
9	10	10	9	10	10	9	10	10	9	9	9	9.6	10
10	3	8	6	5	4	3	5	6	8	6	3	5.4	8

3.5.5.2 Subjective (Manual) Evaluation of Experiment Data

The ten experimental subjects, all engineers at our Houston facility, were classified in two groups, based upon their prior experience in simulation: the "high" experience group had from one to fourteen years' experience in simulation, the "low" experience group had zero to one year experience.

Each subject received the complete set of ten simulation performance plots, in a random order. Working entirely independently, and with no direction as to the criteria they should use, the subjects ranked the ten performance plots -- rank 1 for the best match, 2 for the next-best, and so on down to 10 for the worst match. The results are shown in the above table. As is typical of subjective experiments, there is considerable scatter in the data, the greatest unanymity being evident for the best and worst performance. A few systematic differences between the high and low experience groups were noted, and are discussed in DRL-3. The mean subjective rank (MSR), as well as the means of the two experience groups, were computed and used in the subsequent correlation computations.

AUTOMATED COMPARISON: SIMPLE CRITERIA

Maximum error: $E_{max} = \max \left\{ |r(t) - s(t)| : 0 \leq t \leq T \right\}$

Integral of error: $IE = \int_0^T [r(t) - s(t)] \, dt$

Integral of absolute error: $IAE = \int_0^T |r(t) - s(t)| \, dt$

Time-weighted integral of absolute error: $IAET = \int_0^T |r(t) - s(t)| \, t \, dt$

Integral of squared error: $ISE = \int_0^T [r(t) - s(t)]^2 \, dt$

Time-weighted integral of squared error: $ISET = \int_0^T [r(t) - s(t)]^2 \, t \, dt$

3.5.5.3 Objective (Automated) Evaluation of Experiment Data

A variety of simple evaluation algorithms were tested in this experiment. These algorithms, for which the formulas are shown above, all do some elementary mathematical processing of the reference and simulation time-history data, resulting in a single number whose magnitude is an indicator of the degree of mismatch between the two time-histories (zero for a perfect match).

Of these algorithms, the two involving squares of errors (ISE and ISET) give higher weight to large local errors: the two involving time-weighting (IAET and ISET) give higher weight to persistent errors than transient errors.

OBJECTIVE RANKING: SIMPLE CRITERIA

CASE	MSR	MISMATCH VALUE/RANK						RANK SPREAD		
		E_{max}	AIE	IAE	IAET	ISE	ISET	High	Mean	Low
1	9.4	0.879 9	0.422 4	2.948 9	11.203 9	1.411 9	4.286 9	4	8.17	9
2	1.9	.153 2	.618 6	.618 2	1.789 3	.065 2	.147 2	2	2.83	6
3	6.3	.753 8	1.781 9	1.792 8	5.931 7	.620 8	1.161 7	7	7.83	9
4	5.6	.510 6	.406 3	.883 5	1.903 4	.233 5	.293 4	3	4.50	6
5	7.1	.472 5	1.623 7	1.628 6	6.282 8	.432 6	1.328 8	5	6.67	8
6	3.3	.231 4	.206 1	.657 3	1.782 2	.094 3	.178 3	1	2.67	4
7	1.2	.138 1	.270 2	.287 1	.610 1	.023 1	.029 1	1	1.17	2
8	5.2	.187 3	.579 5	.851 4	4.116 5	.099 4	.478 5	3	4.33	5
9	9.6	1.089 10	3.732 10	3.732 10	12.289 10	2.162 10	5.104 10	10	10.00	10
10	5.4	.730 7	1.748 8	1.748 7	4.622 6	.571 7	.823 6	6	6.83	8

The above table shows the numerical values and resulting ranks determined by application of these simple algorithms to the experiment data. The MSR is also shown on the table for comparison.

Since they emphasize different properties of the response, the automated comparison results show nearly as much scatter as the manual results.

AUTOMATED COMPARISON: FEATURE EXTRACTION

- Would like a processor which would "simulate" the engineer's judgement.
- Processor would use filtering, peak detection, etc., to extract r sponse attributes - oscillation frequency, damping, phase, steady-state value, etc. - from the raw time-history data.
- Errors in the individual response attributes would then be summed, with appropriate weights, to generate a single criterion value:

$$F = a_1 |\Delta\omega| + a_2 |\Delta\zeta| + a_3 |\Delta\varphi| + a_4 |\Delta x_{ss}| + \cdots$$

- Our experience to date indicates that considerable further development will be required to effectively apply the feature-extraction concept.

We also did some work with "feature extraction" techniques in the course of this study. Our goal was to develop an automated comparison algorithm which would "simulate" human judgement by explicitly identifying the degree of mismatch in each of a number of response attributes -- e.g., frequency, damping, steady-state value.

A single number for the overall mismatch could then be computed by forming a weighted average of the mismatches in the individual response attributes. The weighting could be varied, depending upon the application of the module being validated. For example, initial response characteristics might be more important for validation of visual and motion system response, while presistent errors would be more important for variables lying upstream of integrators (e.g., engine thrust, aerodynamic drag).

Our initial results indicate that considerable further development effort will be required to effectively apply this concept.

5.5 Comparison Methods and Criteria

SUBJECTIVE/OBJECTIVE RANK CORRELATION: SIMPLE CRITERIA

SUBJECT GROUP	RANK CORRELATION VS. CRITERION						
	E_{max}	AIE	IAE	IAET	ISE	ISET	MOR
High experience	0.919	0.536	0.940	0.931	0.940	0.945	0.931
All subjects	.905	.558	.946	.954	.946	.971	.943
Low experience	.872	.562	.933	.959	.933	.979	.935

RECOMMENDATIONS:

- Don't use IE, AIE.
- Use ISE when initial response is important (e.g., visual/motion system response).
- Use ISET when steady-state value is important (e.g., variables upstream of integrators).

3.5.5.4 Comparability of Manual and Automated Evaluation Results

Subjective/objective rank correlation results are shown in the above table for the various experience groups, for each individual algorithm as well as the mean objective rank (MOR). Based upon these results and additional study of the characteristics of the various criteria, we recommend using either ISE or ISET, depending upon the response characteristics of greatest importance in each validation application.

SUBJECTIVE/OBJECTIVE RANK CORRELATION: FEATURE EXTRACTION

SUBJECT GROUP	RANK CORRELATION VS. ATTRIBUTE						
	Initial Value	Initial Slope	Final Value	Frequency	Damping	First Peak	First-peak Time
High experience	0.451	0.594	0.422	0.749	0.585	0.596	0.400
All subjects	.528	.547	.524	.750	.617	.644	.377
Low experience	.586	.482	.608	.732	.630	.673	.336

RECOMMENDATIONS:

- Pursue development of feature-extraction algorithms.
- Use only in weighted-average form.
- Conduct additional subjective-evaluation experiments to optimize weights for different applications.

The rank correlations shown above for the individual response attributes are all too low to be of any practical value in validation. This implies that feature-extraction methods can only be useful if multiple attributes are combined via a weighted-average formulation. In any event, additional development effort will be required to implement and apply feature-extraction techniques.

3.6 CONCLUSIONS AND RECOMMENDATIONS, TASK 2.0

This concludes our discussion of the performance verification task. The more significant conclusions and recommendations resulting from the study effort of this task are listed above.

WBS 2.0 Performance Verification Task

6. CONCLUSIONS AND RECOMMENDATIONS

- Perform simulation validation at the individual module level, at intermediate stages of integration, and in the final all-up configuration.
- Concentrate upon "critical" performance parameters.
- Use module interaction as the basis of the module development/integration sequence.
- Establish working interfaces with hardware test groups early in the development cycle.
- Define a universal data format for all validation service routines.
- Do not attempt to convert non-machine-readable reference data into machine-readable form.
- Perform a "make or buy" analysis for the DBMS to support simulator development and validation.
- For initial validation, execute check cases in an envelope-expansion sequence; for revalidation, use an envelope-contraction sequence.
- In automating data comparison and fidelity evaluation, use criteria which correlate well with engineers' subjective judgements - ISE and ISET.
- Continue development of automated feature-extraction techniques for data comparison and fidelity evaluation.

SECTION 4
CONCLUDING REMARKS

SIMULATION VERIFICATION TECHNIQUES STUDY -- "PAYOFF"

- o TECHNIQUES: Powerful, efficient techniques identified for both hardware checkout and performance verification.
- o BASIC DATA: Hardware and software characteristics and parameters identified, required data identified and/or compiled - for every hardware subsystem/unit and every simulation module.
- o SUPPORT SOFTWARE: Support software requirements identified; high-level software design accomplished.
- o DATA BASE: Hardware/software data base content and structure identified; initial data base assembled; high-level DBMS requirements identified; DBMS implementation recommendations made.
- o BASIS FOR FUTURE WORK: Identified potential pitfalls to avoid, promising approaches for further development.

IN SUMMARY - The ground work has been laid for substantial improvements in the effectiveness of the next generation of spacecraft simulators.

Having completed our review of the objectives and the results of this study, it is appropriate to consider what has been achieved -- i.e., what contributions these study results will make to the efficient and economical development and operation of the next generation of spacecraft simulators.

The above chart lists the major accomplishments of this study as we see them. Many of the study outpurs will be immediately useful, such as the check-out and validation algorithms and the initial hardware/software data base. In addition, the study has built a solid base for future development of checkout and validation techniques.

WHAT NEEDS TO BE DONE NOW

- Verification/validation requirements factored into upcoming simulator procurement specifications and/or proposal evaluation criteria:
  - Hardware instrumentation
  - Diagnostic software requirements
  - Contractor's verification plans and support software
  - Contractor's management structure and visibility of verification and validation functions
- Establishment of liaison interfaces with spacecraft system development and test groups.
- Expansion of verification data base, using structure defined by this study.
- DBMS requirements definition, make and/or buy decision, and initiation of development/procurement.

The work begun with this study must be carried forward, if the expected benefits are to be realized in upcoming simulator projects. One near-term activity which should be pursued is the incorporation of verification and validation requirements into simulator procurement specifications, proposal evaluation criteria, and development plans.

The other items listed above -- spacecraft hardware liaison, data base expansion, and DBMS development/procurement -- are long-term activities which should be begun early in the simulator development cycle.

SECTION 5

ANNOTATED BIBLIOGRAPHY

Documents are listed here in the same order in which they appear on the Schedule of Study Deliverables in Section 1. Contracted end items are identified by their Data Requirements List (DRL) line item numbers, Task Reports by their TR numbers.

5.1 RESULTS OF TASK 1.0, HARDWARE VERIFICATION

(1) TR-1: R. W. Foster, C. E. Jones, G. Montoya, and T. H. Wenglinski, Simulation Hardware Definition Report, MDC E1006, 25 January 1974.

This report documents the results of Subtask 1.1, Definition of Simulation Hardware. The next generation of spacecraft simulators at JSC are described: the Shuttle Procedures Simulator (SPS), the Orbiter Aeroflight Simulator (OAS) (known at that time as the Horizontal Flight Simulator (HFS)), and the Shuttle Mission Simulator (SMS).

Based upon a composite view of these three simulators, as well as a review of state-of-the-art equipment for flight simulators, we defined a "reference" simulator configuration to serve as the basic vehicle for hardware checkout studies. The description of the reference simulator includes overall system configuration, major subsystem configurations, and individual hardware components. Component-count estimates are included to indicate the potential magnitude of the checkout and data-management problems.

A glossary of checkout and test terminology is also included.

(2) TR-2a: P. B. Schoonmaker, Simulator Verification Study: Final Report, MDC E0861, 30 July 1973.

This report describes company-funded research into verification technology, performed before initiation of the contracted study.

The two basic problem areas treated in this report are operational verification and design verification. A variety of guidelines and techniques, identified

primarily from a survey of the literature, are described in each problem area. An appendix treats the application of directed graph theory to fault isolation.

A comprehensively indexed bibliography is also included.

(3) TR-2: G. Montoya, P. B. Schoonmaker, and T. H. Wenglinski, Hardware Self-Test Techniques Survey Report, MDC E1033, 1 November 1974.

This report documents the results of Subtask 1.2, Survey of Current Hardware Self-Test Techniques. In this subtask, we built upon the data base established by the company-funded research described in TR-2a, while concentrating upon techniques of high potential applicability to simulator hardware checkout.

The information in this report was based upon a survey of NASA, McDonnell Douglas, military, and commercial airline simulation facilities, as well as a continuing search of the verification literature.

Techniques for fault detection, fault isolation, and incipient fault detection are described in some detail. Test design approaches, test hardware design considerations, and test data processing algorithms are also covered.

The report also includes an extensive glossary, an annotated bibliography of documents considered of major interest, and a larger, comprehensively-indexed bibliography.

(4) TR-3: G. Montoya and T. H. Wenglinski, Integrated Simulator Self-Test System Concepts, MDC E1149, 20 September 1974.

This report documents the "system-oriented" results of Subtask 1.3, Definition of Hardware and Software Techniques for Simulator Checkout. Tests for checkout of individual simulator subsystems -- DCE, motion base, visuals, etc. -- are briefly described, to provide a framework for discussion of overall test execution and sequencing. Test executive software and subsystem test software are described, as well as test hardware for sensing, signal generation, and signal processing/display.

System design-change impacts and cost impacts are also discussed.

(5) DRL-3: Simulation Self-Test Hardware Design and Techniques Report, MDC E1150, 1 November 1974.

This report constitutes the final and complete documentation of Task 1.0, including results previously documented in Task Reports 1, 2 and 3, as well as results not previously published in the Task Reports.

The reference simulator configuration description and the self-test techniques survey description correspond closely to material published in TR-1 and TR-2, respectively. The treatment of hardware and software techniques for simulator subsystem checkout is much more comprehensive than the treatment in TR-3. Detailed descriptions of individual subsystem checkout techniques include high-level test software designs and data base requirements. The discussions of integrated test system design, test system cost impacts, and simulator design change impacts follow TR-3.

The glossary and indexed bibliography are also included in this report.

5.2 RESULTS OF TASK 2.0, PERFORMANCE VERIFICATION

(1) TR-4: Simulation Module Performance Parameters and Performance Standards, MDC E1127, 1 August 1974.

This report presents module-oriented results for simulation modules in the Environment, Crew Station, Vehicle Configuration, and Vehicle Dynamics categories.

To establish the context for the module-oriented developments, brief discussions of introductory topics are provided: guidelines for definition of performance parameters and "critical" performance parameters, identification and characteristics of alternate reference data sources, and basic validation techniques and required support software.

This work was later revised and expanded for publication in DRL-3.

(2) TR-5: L. M. Duncan, J. P. Reddell and P. B. Schoonmaker, Subsystem Simulation Validation Techniques, MDC E1201, 30 December 1974.

Subsystem/module-oriented results in the Vehicle Subsystems category (the bulk of the module-oriented results) are presented in this report. To establish the context for these results, material previously presented in TR-4 is briefly summarized. In these summaries, the emphasis is on implications for subsystem simulation modules; e. g., interfaces between environment modules and subsystem modules.

This material was later incorporated into DRL-3.

(3) DRL-3: L. M. Duncan, J. P. Reddell, and P. B. Schoonmaker, Simulation Performance Validation Techniques Document, MDC E1136, 27 January 1975.

This report covers all work done under WBS 2.0. It includes all subsystem/module-oriented results published in TR-4 and TR-5, and unified treatments of validation techniques, data-handling considerations, and other topics not previously published in the TR's. (TR-6 was not published separately; instead, the material intended for publication in TR-6 was incorporated into DRL-3.)

5.3 RESULTS OF TASK 3.0, FINAL DOCUMENTATION

The results of the Final Documentation task include the present report (DRL-4), the New Technology (Technical Detail) report (DRL-5), and the Summary Report of New Technology Review Activities (DRL-6).