QUICK REACTION GUIDANCE AND TARGETING STUDY

SECOND QUARTERLY SUMMARY REPORT

NOVEMBER 1966—JANUARY 1967

IBM REPORT NO.
67-M50-002

IBM

Federal Systems Division
Space Systems Center
Washington Avenue
Endicott, New York

952-5497-2

QRGT STUDY

SECOND QUARTERLY SUMMARY REPORT

IBM NO 67-M50-002

ORIGINATING GROUP QRG & TS Staff

CONTENT APPROVED BY

Howard M Robbins J H Slavin

CONTRACT NO AF 04(695)-1078

DATE February 7m 1967

IBM FEDERAL SYSTEMS DIVISION
Space Systems Center
Endicott, New York

QUICK REACTION GUIDANCE AND TARGETING STUDY
SECOND QUARTERLY SUMMARY REPORT
(NOVEMBER 1966 – JANUARY 1967)

CONTRACT NO: AF04(695)-1078

IBM REPORT NO: 67-M50-002

APPROVED BY:

Howard M Robbins

STUDY DIRECTOR

10 FEBRUARY 1967

ABSTRACT

This report covers the 3-month study period from October 22, 1966 through January 27, 1967. The major technical accomplishments achieved during that time were as follows:

- Preparation of a program plan for proposed follow-on effort in Phase II.
- Definition of a classification hierarchy of flight program modules in conjunction with development of an on-board multi-mission planner.
- Calculation of total impulse and time-to-rendezvous for a general two-burn rendezvous technique and the three-burn Variable Point Guidance technique. Both techniques are being considered for preliminary trajectory calculations as part of on-board mission planning.
- Development of an optimization algorithm for use in mission planning. The algorithm has been programmed for computer use and is being employed to determine the benefits of trajectory optimization.
- Development of explicit guidance equations for the exo-atmospheric phase of ascent guidance.
- Study of range safety and the restraints and requirements applicable to quick reaction guidance and targeting.
- Completion of functional flow diagrams and descriptions of major routines in the On-Board Executive System.
- Definition of major support software requirements of the Ground Operating System including Multi-Mission Library, flight program configurator, assembler, and simulation facilities.
- Estimation of memory size and speed for some of the functions to be performed by the on-board computer.
- Initiation of study of tasks to be performed by the man in the guidance and targeting loop.

TABLE OF CONTENTS

Section	Title	Page
1	INTRODUCTION	1
2	SUMMARY OF TECHNICAL ACCOMPLISHMENTS	3
	2.1 Program Management (Task 1)	3
	2.1.1 Introduction	3
	2.1.2 Project Planning Report	3
	2.1.3 Technical Direction	4
	2.2 Integrated System Definition (Task 2)	5
	2.2.1 Introduction	5
	2.2.2 Summary of Accomplishments	6
	2.2.3 Modularization of Missions	7
	2.2.4 On-Board Mission Planning Function	7
	2.2.5 Preliminary Trajectory Generation	9
	2.2.6 Development of Optimization Algorithms	12
	2.2.7 Trajectory Optimization	13
	2.2.8 Ascent Guidance	15
	2.2.9 Re-Entry Guidance	18
	2.2.10 Range Safety	21
	2.3 Computer Interface (Task 3)	25
	2.3.1 Introduction	25
	2.3.2 Summary	27
	2.3.3 The Ground Operating System	31
	2.3.4 On-Board Executive System	34
	2.3.5 Main Storage Supervision	53
	2.3.6 Computer Sizing	59
	2.4 Displays (Task 4)	61
	2.5 Error Analysis (Task 5)	65
3	APPENDICES	
	A. Modularization of Missions	A1 - A21
	B. Mission Planning Function	B1 - B 8
	C. Numerical Comparison of Alternative Orbit Transfers	C1 - C15
	D. Experimental Optimization Algorithms	D1 - D10
	E. Glossary	E1 - E13

SECTION 1

INTRODUCTION

SECTION 1

INTRODUCTION

This report presents a summary of technical accomplishments during the second quarter of the Quick Reaction Guidance and Targeting Study (QRGT) and covers the period from October 22, 1966 through January 27, 1967. The study is being conducted for Space Systems Division of USAF by Space Systems Center, Federal Systems Division, IBM, Endicott, New York under contract AF 04(695)-1078. Work was initiated on August 1, 1966; a summary of the first quarter's effort is contained in IBM Report No. 66-M21-003, "QRGT First Quarterly Summary Report", November 1966.

The over-all objective of the study is to develop techniques and software for mission planning, targeting, and guidance which significantly reduce the anticipated costs of preparing software and the anticipated reaction time for future military space missions.

Major reductions in costs and reaction time will be made possible by advanced general-purpose digital computers with increased computational capacity in the spacecraft of the future. The central study task, therefore, is to develop guidance equations and software programs which utilize the on-board computer in an efficient manner. These equations and programs should have minimum dependence on spacecraft configurations and mission objectives and should allow specific missions to be planned, targeted, validated, and performed without support from a large ground-based computer facility or a large number of highly skilled engineers. Equation development, software preparation, trajectory studies, flight profile planning, etc., will become, ideally, a one-shot process for a wide range of vehicle configurations and space missions.

Study efforts are organized under these five basic tasks:

- Program Management (Task 1)

  Technical guidance, program administration and control, and preparation of program plans and reports.

- Integrated System Definition (Task 2)

  Study of missions, guidance and navigation systems, booster and upper stage vehicles, and development of improved guidance equations and on-board mission planning, targeting, and validation techniques.

- Computer Interface (Task 3)

  Determination of computer speed, memory, and precision requirements and development of advanced software organization for space-borne and ground computer programs.

- Displays (Task 4)

  Determination of the function of man in the guidance and targeting loop and specification of the information required for displays and controls.

- Error Analysis (Task 5)

  Preparation of a guidance and targeting error model.

Detailed task descriptions are presented in IBM Report No. 66-512-003, "Quick Reaction Guidance and Targeting Study Program Plan (Revised) for Phase I", August 30, 1966 and IBM Report No. 66-M21-005, "Preliminary Phase II Program Plan", 7 December 1966.

Major accomplishments during the past 3 months are summarized in the subsequent sections, by study task. In addition, detailed supporting information is presented in the Appendices. A glossary of peculiar terms is provided in the final appendix for reference purposes.

To facilitate review of the major technical accomplishments to date, all classified material has been segregated and is contained in IBM Report No. 3-260-0012, which forms a supplement to this quarterly report.

SECTION 2

SUMMARY OF TECHNICAL ACCOMPLISHMENTS

SECTION 2

SUMMARY OF TECHNICAL ACCOMPLISHMENTS

2.1 PROGRAM MANAGEMENT (TASK 1)

2.1.1 INTRODUCTION

In addition to continuing efforts on technical guidance and program control, a preliminary plan for Phase II was generated during the past quarter. One technical direction meeting was held at SSD on December 21-22.

2.1.2 PROJECT PLANNING REPORT

A planning report entitled "Preliminary Phase II Program Plan", IBM Report No. 66-M21-005, dated December 7, 1966, was submitted. This document presents the proposed follow-on activity to the current Phase I contract. As defined in the planning report, the objectives of Phase I are:

- Develop an approach to the implementation of the QRGT concept.
- Identify problems
- Suggest and test solutions.
- Evaluate advantages of approach.
- Give evidence of feasibility.

The objectives of the proposed Phase II effort are:

- Demonstrate feasibility of the proposed concept.
- Provide an accurate estimate of the computer requirements.
- Define hardware requirements for the integrated guidance system.
- Define the requirements of on-board and ground software.
- Define an implementation plan for the generation of this software.

The proposed program for Phase II is closely integrated with Phase I within the framework of the five basic study tasks. The major characteristics of the technical approach to be followed in Phase I and Phase II is a two-pronged effort for reducing software preparation time and costs. Namely:

- Improved guidance algorithms which eliminate mission dependent pre-computations, reduce and simplify vehicle dependent pre-computations, treat constraints explicitly, and provide a high degree of autonomy.
- An advanced system of ground and spaceborne software and procedures which provide a highly automated method for generating, modifying and validating flight programs.

The planning document also included budgetary estimates of the manpower and cost of the proposed program together with suggested efforts to parallel the QRGT software contracts.

As a result of a review of the Phase I program status and the proposed Phase II plan at the technical direction meeting held December 21 and 22, the following specific efforts have been undertaken on Program Management:

- Task and subtask descriptions are being clarified, schedule for efforts detailed, and the interrelationships between subtasks identified. PERT-type charts are being prepared covering the Phase I activity and the proposed Phase II program.

- The requirements of Phase II are being studied in the light of an operational indoctrination by 1970. Application of the QRGT concept will be considered for existing Air Force programs as well as anticipated programs.

- The existing plans for Phase I and the proposed Phase II effort have been reviewed with personnel from IBM's Saturn launch operations.

A Project Planning Report incorporating the above items for Phase I and Phase II is in preparation.

2.1.3 TECHNICAL DIRECTION

A meeting was held at SSD with Air Force and Aerospace personnel on December 21 and 22, 1966. Major topics were:

- General review of program objectives.

- Current and projected status at end of Phase I.

- Review of first quarterly report.

- Discussion of preliminary Phase II program plan.

As a result of technical direction received, the following action has been taken:

- A study has been made of range safety operations and the requirements on mission planning and other aspects of QRGT (see Task 2).

- Effort on the organization and modularization of mission program elements has been increased in conjunction with development of the mission planner (see Task 2).

- A study of the role of man has been initiated to identify the man/machine interface and generate display information requirements (see Task 4).

2.2 INTEGRATED SYSTEM DEFINITION (TASK 2)

2.2.1 INTRODUCTION

An integrated approach is being taken to the design of the guidance and navigation system. The impact of quick reaction guidance and targeting is being established at all interfaces with the guidance system, and system requirements are being coordinated with the design of the on-board computer (Task 3) and displays (Task 4).

The efforts in Task 2 fall in the following categories and are being pursued in parallel:

- Analysis
  - Study of the spectrum of military missions.
  - Study of range safety restraints and requirements

- Systems Engineering
  - Booster and upper stage requirements.
  - Final stage vehicle requirements.
  - Configuration of multi-mission guidance and navigation system.
  - Specification of component development requirements
  - Ground support equipment requirements.

- Guidance Techniques
  - Trajectory generation for mission planning.
  - Explicit equations for ascent and orbital maneuvers.
  - Reentry guidance of lifting vehicles

- System Software Requirements
  - On-board multi-mission planning
  - On-board trajectory optimization.
  - On-board targeting
  - On-board validation.

Study efforts in the past quarter have concentrated on development of guidance techniques and system software requirements.

2.2.2 SUMMARY OF ACCOMPLISHMENTS

Significant accomplishments of Task 2 during the second quarter include:

- Definition of a classification and modularization scheme for flight program elements which will be of use in the development of on-board mission planner.
- Continued progress in the definition of the on-board mission planner.
- Development of techniques for generating preliminary trajectories for mission planning. Total impulse and time requirements for rendezuous were calculated for two techniques.
- Progress in the development and use of an optimization routine for the mission planner.
- A preliminary investigation of range safety requirements and generation of some tentative suggestions for satisfying these requirements without undue sacrifice of quick reaction capability.
- Progress in the development of an explicit, nearly optimal, algorithm for three-dimensional ascent guidance.
- Progress in the development of guidance equations for re-entry.

2.2.3 MODULARIZATION OF MISSIONS

A mission classification hierarchy has been defined to aid the development of both the on-board mission planner and the flight program organization. The levels in this hierarchy are: Set, Class, Mission, Phase, Function, Mode, and Element.

The presently defined QRGT missions have been broken down within this hierarchy to indicate the mission phases, the major functions performed in each phase, and the particular mode of operation required.

The classification concept and the mission breakdown appears in Appendix A.

2.2.4 ON-BOARD MISSION PLANNING FUNCTION

A continuing activity has been carried on to define the on-board mission planner. In line with the classification hierarchy discussed in Section 2.2.3, a Mission Planning Function has been defined and its requirements identified by categorizing its Modes. The Modes presently defined are Preliminary Trajectory Generation, Trajectory Optimization, Mission Safety, Trajectory Selection, Targeting, Flight Program Identification Validation, and Plan Update. Figure 2-1 represents a functional flow of the mission planner; the modes of this planner are described in Appendix B.

Work on the first two modes is discussed in Sections 2.2.5 and 2.2.7, respectively. Preliminary considerations of Range Safety requirements, which constitute the main role of Mission Safety during Pre-Launch and Ascent, are found in Section 2.2.10.

The remainder of the activity during the last quarter has been consideration of the related areas of Flight Program Identification, Validation and flight program organization. The last item is a Task III activity but it bears directly on the first two. The first step in this consideration was a modularization of all missions as described in Section 2.2.3. The next step, presently in progress with Task 3, involves a more detailed look at representative programs required to implement the various QRGT missions with the objective of defining Mode and Element programs. Presently, the latter program is considered to be the basic, on-board module for generating a flight program. But, at the same time, it is felt that the most efficient module size will be at least at the Mode level because:

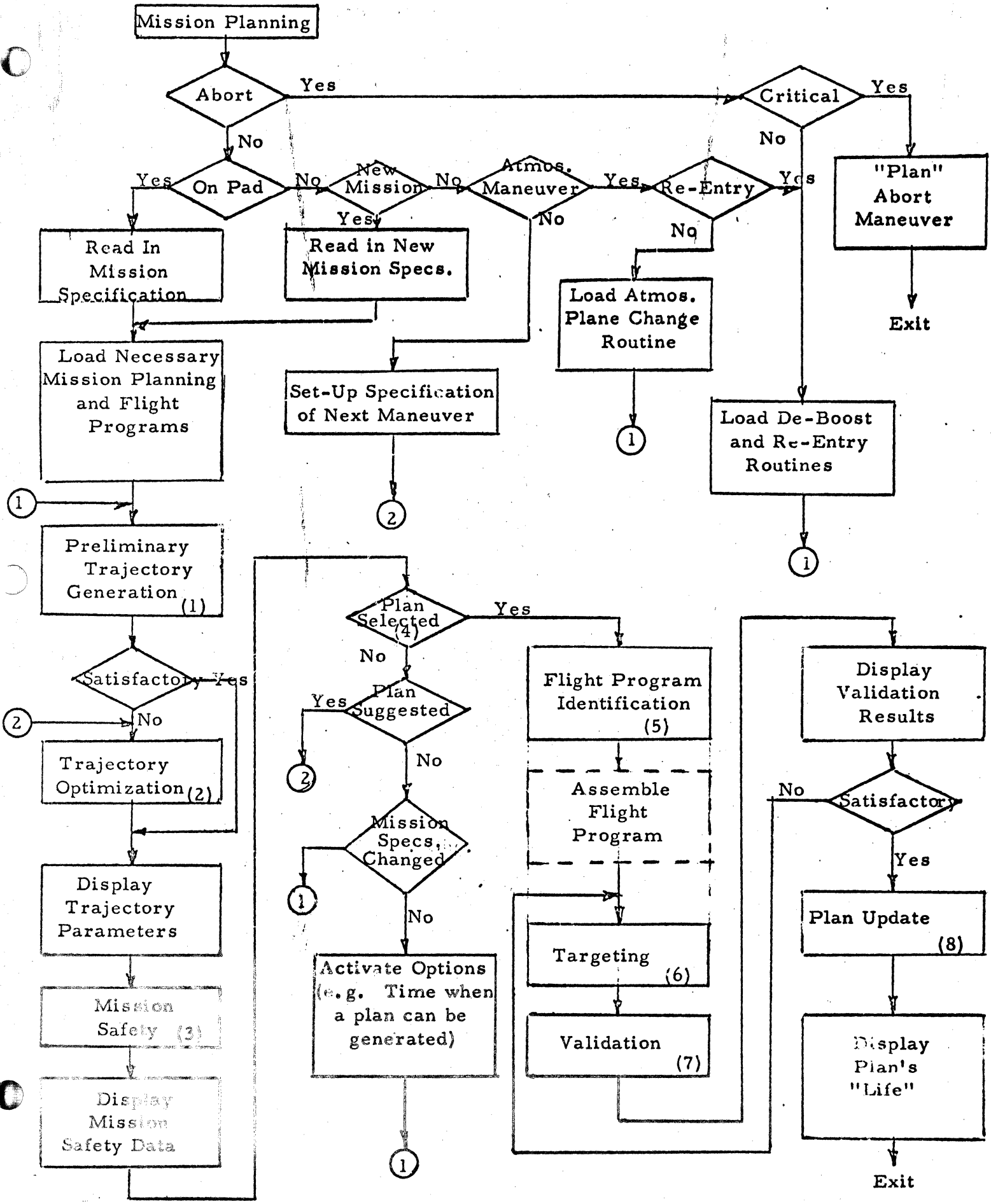


Figure 2-1 - On-Board Mission Planning Function

- The memory saved by commonality at the Element level will generally not be sufficient to justify the additional complexity of the programs needed for on-board assembly of a flight program from numerous small modules. Special cases can be made for any Elements which do justify commonality.

- Assembly of a flight program from numerous small modules creates a severe problem of on-board validation. Pre-assembly into larger modules, which are pre-validated as units, greatly reduce this problem.

If the on-board system has an Auxiliary Memory Unit (AMU), the "best" module size may be at the Phase level, with duplications resulting in just a longer tape, to further reduce the identification, assembly, and validation problems. In this case the Flight Program Identification Mode reduces to basically naming the Phases and their sequence, and the validation Mode involves checking targeting data and program sequencing (as the Phase module can be validated off-board).

2.2.5 PRELIMINARY TRAJECTORY GENERATION

The Mission Planning Function described in Appendix B initially generates several preliminary trajectories and then improves some or all of these trajectories by utilizing an on-board optimization routine. The optimizer seeks the "nearest" local minimum of the cost function. Thus, it is desirable to have several preliminary trajectories to increase the chances of obtaining an absolute minimum. In some applications the on-board computer may not have the capability of performing on-board optimization. In these applications the preliminary trajectories must include at least one which is feasible.

This section covers the progress in the development of preliminary trajectories for orbital rendezvous from a parking orbit. (Direct ascent from launch to rendezvous is discussed in the Ascent Guidance section). Trajectory prescriptions presently under consideration for rendezvous from a parking orbit can be classified in five groups:

- General two-burn rendezvous maneuvers.
- Optimal two-burn orbit transfer with impulse splitting for rendezvous.
- Three-burn rendezvous maneuvers.
- Variable Point Guidance (VPG).
- Modifications to VPG.

For two-burn rendezvous, IBM has developed a Rendezvous/Intercept Routine with Option A-4, called RIA4, which is discussed below and compared with the rendezvous transfers generated by VPG.

The Rendezvous/Intercept Routine

The fundamental idea in the rendezvous/intercept routine is that the slope, m_a, of the transfer trajectory at arrival is a pre-determined function of the range angle of transfer Θ_a, i.e.,

$$m_a = m_a (\Theta_a ; r_o)$$

The range angle is adjusted to insure proper phasing of the S/C and target. The time of the first burn is then varied to seek a local minimum of the payoff function which, in this case, is the total impulse required for rendezvous. There are several different functional relationships available for computing m_a in this routine. A modification of the equal change in slope criteria, Option A-4,[1] has been employed in the numerical results which follow.

Figure 2-2 shows a typical behavior of the total impulse as a function of the time of the first burn. In each subsequent revolution of the S/C in the parking orbit the graph has two local minima. The first branch of the curve corresponds to range angles of transfer greater than 180^o and the second for Θ_a less than 180^o.

The RIA4 and VPG techniques give many solutions of the rendezvous problem. Numerical results for the least impulse and the least time of rendezvous solutions are described in Appendix C. These results show that the two-burn technique RIA4 compares favorably, in time and fuel requirements, with the three-burn VPG technique. The RIA4 technique requires more computations but less storage than VPG since most of the RIA4 routine would also be used in active guidance. The optimization of the two-burn transfers should require a considerably smaller number of computations than that of the three-burn transfers. Further testing with other orbits is anticipated.

The results of optimization experiments will determine (a) the relative merits of RIA4 and VPG as generators of initial trajectories for optimization, and (b) the amount of improvement obtainable by optimization. In some cases, this improvement will be small, indicating the feasibility of using RIA4 or VPG without optimization in certain applications.

---

1 ORGN First Quarterly Summary Report, November 1966, p. D-15.

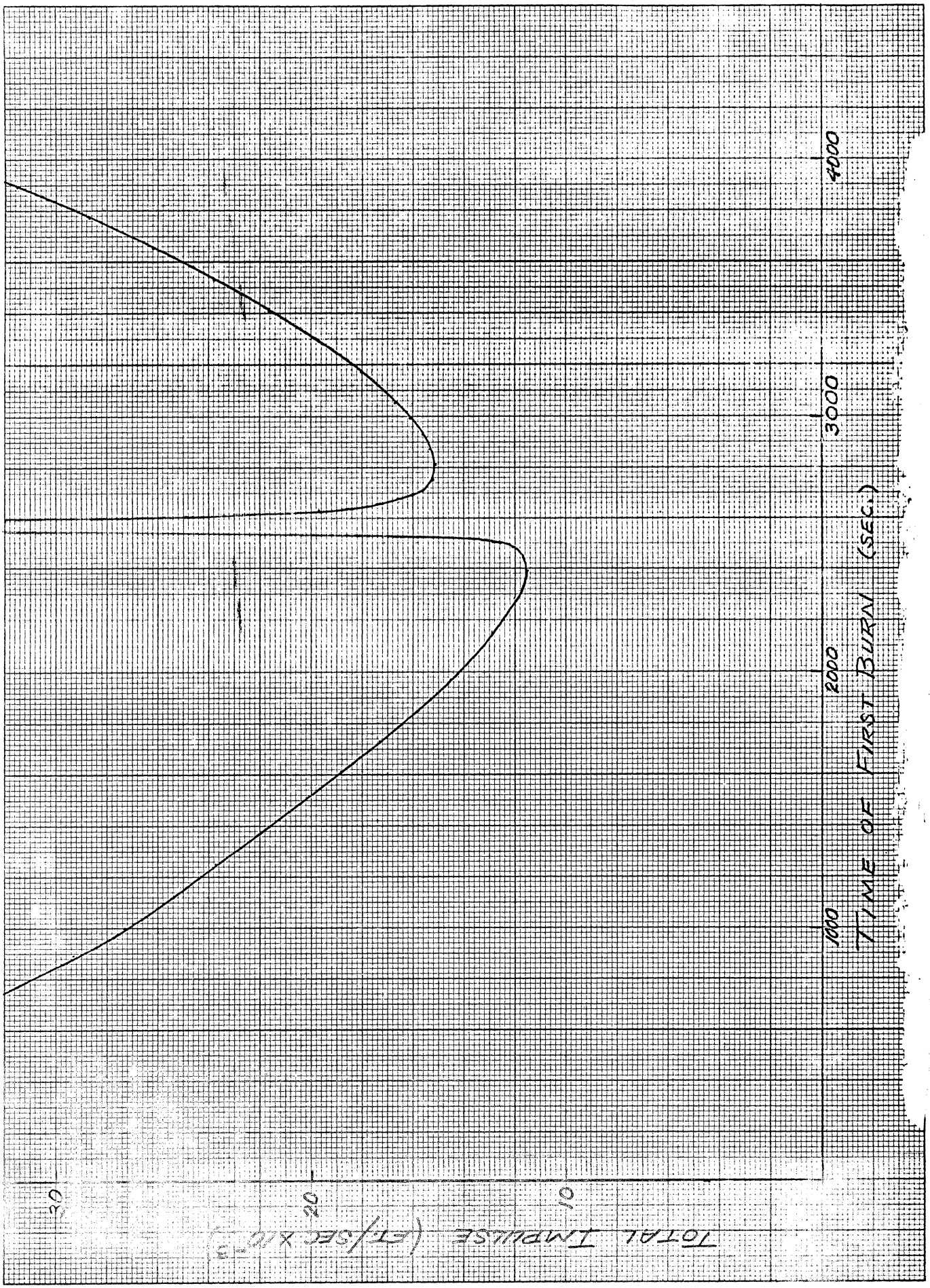


FIGURE 2-2. THE TOTAL IMPULSE AS A FUNCTION OF THE TIME OF THE FIRST BURN. TARGET ORBIT #1, TARGET TRUE ANOMALY AT EPOCH IS 0°

2.2.6 DEVELOPMENT OF OPTIMIZATION ALGORITHMS

For experimental study of optimization algorithms and their application to mission planning, IBM has developed a program called DSOP (Direct Search Optimization Program). This program is described in detail in Appendix D. DSOP consists of a number of subprograms which combine to form optimization algorithms. The principal subprograms of DSOP are PMS (Pattern Move Search), EMR (Explanatory Move Routine), and UNIVAR (Exploration by a sequence of one-variable searches). EMR and UNIVAR are alternative subroutines of PMS. Possible future developments include modifications of UNIVAR and/or EMR, development of new subroutines to replace UNIVAR and EMR in PMS, and development (or acquisition) of new algorithms to complement or replace PMS.

The algorithms were initially developed and tested for optimization without constraints, and subsequently modified to incorporate simple constraints (bounds on the independent variables). More general constraints are presently handled by a penalty-function technique. Future improvements under consideration include (a) techniques for accelerating convergence, and (b) techniques for handling general equality or inequality constraints.

The optimization algorithms of DSOP have been successfully applied to a variety of test problems. In particular, the PMS-UNIVAR combination has been successfully applied to the optimization of orbit transfer trajectories; these experiments are described in Section 2.2.5 above, and in Appendix C.

In addition to support of these experiments, and various modifications and tests of DSOP, effort during the last quarter has been devoted to a survey of the state-of-the-art for optimization algorithms, and to consideration of possible next steps in the development of PMS and its subroutines. Also, consideration has been given to the possibility that some existing optimization program may offer a useful alternative to, or complement to, the use of PMS in further trajectory-optimization experiments. A program listing for Powell's 1964 "conjugate gradient" direct search algorithm[1] has been obtained, converted to FORTRAN 4, debugged, and incorporated into DSOP for convenience. In preliminary experiments with simple test problems, Powell's method has shown performance considerably superior to PMS. This algorithm has no provisions for handling constraints of any type, and modifying it to handle constraints (in any way other than by penalty functions) is a nontrivial problem. However, the quadratic convergence of Powell's method may possibly make it an efficient constrained optimizer using penalty functions.

---

1 Powell, M. J. P., "An Efficient Method of Finding the Minimum of a Function of Several Variables without Computing Derivatives", Computer Journal 7, p. 115, 1964.

Also, experiments were performed with a modified version of UNIVAR incorporating some features similar to Powell's method. This modified UNIVAR showed a significant acceleration of convergence. It was less efficient than Powell's algorithm, but this was expected since it was an experimental combination not designed for efficiency.

In cooperation with the ascent guidance subtask (Section 2.2.8), a rapidly convergent method was developed, and successfully tested, for systematically varying five parameters to minimize an objective function (the total burning time) while accurately maintaining three equality constraints (required-velocity conditions). The method used a secant method for maintaining the constraints, and a simple version of Powell's conjugate gradient method for finding the minimum. The details of the method were specialized to the ascent problem, but the principles are general, and may be incorporated in a modification of DSOP at some later time.

During the next quarter, the effort devoted to algorithm development will diminish, since emphasis has now shifted to application of available algorithms to mission-optimization experiments. A demonstration of mission optimization with existing, possibly quite inefficient optimizers is an important contribution toward demonstration of QR feasibility, and is more urgent than improvement of efficiency. Major improvements in efficiency may be expected to result from the further development of optimization algorithms, which is being actively pursued by IBM and many other organizations.

2.2.7 TRAJECTORY OPTIMIZATION

A procedure has been defined and equations written and programmed for formulating N-burn or bit transfers with or without rendezvous, in a form suitable for trajectory optimization by direct search techniques. This program is being used with the optimization algorithm discussed in Section 2.2.6 to determine local time-optimal or full-optimal orbit transfers with or without rendezvous.

The complete program will represent the Trajectory Optimization Mode of the Mission Planning Function for all orbital maneuvers (the Ascent Phase will be added later) and will be combined with the Preliminary Trajectory Generation Mode which provides the initial trajectories to be optimized.

In its present form, the starting solution is specified by:

$\underline{R}_o, \underline{V}_o, t_o$ - Initial Orbit Ephemeris

$\underline{R}_T, \underline{V}_T, t_T$ - Target (or Final) Orbit Ephemeris

N - Number of burns (Presently $N \leq 5$)

M - Rendezvous Indicator (Yes or No)

R - Optimization Requirement (Min time or Min fuel)

t_1 - Time of 1st burn

t_f - Time of Nth burn

R_i - Radius magnitude at i^{th} burn

α_i - Plane Change angle at $(i-1)^{st}$ burn

$\Delta\Theta_i$ - Range angle between $(i-1)^{st}$ and i^{th} burn

ma_i - Slope (tangent of flight path angle) at arrival at i^{th} burn

} $i = 2 \ldots N-1$

ma_f - Slope at arrival at last burn (not necessary for rendezvous case)

and additional control parameters for the search routine.

The above trajectory parameters (t_1 to ma_f) are systematically varied by the search algorithm to minimize the pay off function (the time or fuel required for the maneuver). Inequality constraints on the trajectory parameters are handled by the search routine and functional inequalities are handled by penalty terms in the payoff function. The present program includes only a constraint on the total ΔV which is limited to ΔV_{max}, an input quantity.

The initial test problem, an unconstrained two-burn orbit transfer without rendezvous, was successfully run. The second test case, unconstrained two-burn rendezvous, encountered trouble because the time-of-flight routine was restricted to elliptical orbits, while during the search routine the transfer arc was sometimes hyperbolic. Herrick's universal time-of-flight equations have been substituted and additional changes incorporated so that 180° and 360° (phasing orbit) trajectories can be handled. For a 180° transfer between the last two burns (when $N \geq 3$), the transfer plane is not defined (the trajectory parameter α_i defines the plane for the other burns). This trouble was resolved by adding to the program a one-dimensional search for the plane that minimizes the sum of the last two impulses. In the case of a phasing ellipse (such as a VPG Full Orbit Phasing Option) the transfer angle between two burns in a multiple of 360° and the

Required Velocity routine would give a value of zero. The addition to the program treats this case as an impulse splitting problem by analytically calculating the two co-linear velocity impulses necessary to maintain the geometry and phasing.

With the above changes incorporated, the second test case was successfully run. This case corresponds to an optimization of the Rendezvous-Intercept Option A-4 example which appeared in the December Monthly Report (Table I, page 10, which is the same as Run #1, target true anomaly = 0° in Appendix C) and resulted in a total impulse of 11,493.963 feet/sec. and a rendezvous time of 13,879.676 sec. This compares with the Optimum 2-Burn Rendezvous results (11,494 and 13,870 respectively) appearing in the same report which was obtained with an IBM Grid Search Program.

The third test case exercised was the fuel optimization of a VPG result for the same problem as above. The input was a VPG Bielliptic Chase option with no high altitude pure plane change burn which requires 13,528 feet/sec. of ΔV and 19,920 sec. for rendezvous (Bielliptic b, Table I, p. 10 in December Monthly Report). This input involves a 180° transfer between the last two burns of a three burn rendezvous. The local optimum corresponding to this case was found to require a ΔV of 12,858 feet/sec. and rendezvous occurred at 19,713 seconds, which is a savings of 670 fps. in ΔV and 207 seconds in time.

The N-burn Orbit Transfer Optimization Program will be employed to investigate VPG and Rendezvous Intercept solutions for various target geometries and phasing. This will allow for the generation of trajectory data so that comparisons can be made between both the starting solutions (non-optimal) and the resulting optimized solutions. This information will be used to evaluate techniques to be used in the Preliminary Trajectory Generation Mode (see Section 2.2.5).

2.2.8 ASCENT GUIDANCE

One of the objectives of the QRGT study is to develop guidance equations which are completely explicit, in the sense of not requiring off-board precomputations, and which are suitable for use in on-board mission planning. For the exo-atmospheric phase of ascent guidance, this development was essentially completed during the second quarter. An oral presentation of the ascent guidance technique was given at the Aerospace Corporation on January 16, 1967. A written report is expected to be completed in February.

The ascent guidance has two principal modes: ascent for orbit injection, and ascent for intercept or rendezvous. For both modes of ascent guidance, a three-dimensional form of the linear-tangent steering policy has been implemented. The linear-tangent steering policy is a well-known optimal solution of the flat-Earth ascent problem with no time constraints, and is nearly optimal in the round-Earth case, for typical boundary conditions. A novel integration technique, combining analytic expressions for certain thrust integrals with a rudimental predictor-corrector method, is employed to rapidly and accurately predict trajectories for use in iteratively computing the steering coefficients and thrust cut-off time. (This integration technique is described in the First Quarterly Summary Report, which includes a complete description, with equations, of a two-dimensional version of the proposed ascent guidance scheme.) In the orbit injection mode, five equality constraints are simultaneously satisfied at thrust termination: three velocity components, and radial and out-of-plane position components. In the direct -ascent-to-intercept mode, an optimization technique systematically varies pitch rate and yaw rate to minimize burning time while maintaining three quality constraints (the conditions for intercept at a specified time). Direct-ascent-to-rendezvous is the same, except that the quantity minimized is total burning time, including the accommodation burn for rendezvous. This burn is estimated by the impulsive approximation.

Coplanar ascent for orbit injection was demonstrated in November. Coplanar direct-ascent-to-intercept was developed and demonstrated in early December. In late December the ascent-for-orbit-injection mode was expanded to three-dimensional form and demonstrated for ascents with dog-legs as large as 40°. In early January, the direct-ascent-to-intercept mode was expanded to a three-dimensional form, and direct intercepts with dog-legs as large as 40° were demonstrated. In both ascent guidance modes the ability to predict time of cut-off within 0.7 seconds, when the time remaining until cut-off is greater than 550 seconds, has been repeatedly demonstrated. Orbit injection errors have always been less than 0.01 feet/second in velocity and 10 feet in position; intercept errors have been less than 10 feet.

In the future, a minor revision of the direct-ascent-to-intercept implementation will be made which will replace the two-dimensional direct search for optimal pitch and yaw steering rates with an alternative procedure based on a transversality condition. Direct comparison with a theoretically optimum trajectory will be made to quantitatively determine the degree of suboptimality of both the explicit ascent guidance modes.

The ascent guidance equations are also usable for orbit transfer guidance. The two modes of ascent become two modes of orbit transfer; orbit transfer for injection into a new orbit, and orbit transfer for intercept or rendezvous.

During the next quarter, the study effort in the area of ascent guidance will be devoted to (a) completing and documenting the guidance equations for the exo-atmospheric phase of ascent, (b) developing equations for the atmospheric phase of ascent, and (c) studying the integration of ascent guidance with orbit transfer guidance and with mission planning.

2.2.9 RE-ENTRY GUIDANCE

The guidance policy selected by IBM for the re-entry phase (predictor guidance) uses the relative location of the destination within the instantaneous footprint to determine maneuver commands (bank angle and angle of attack) which will maximize future freedom of choice, by attempting to move the destination point into the interior of the footprint. These commands are overridden, when necessary, by modifications which ensure that thermal, g-load, and controllability constraints will not be violated. The footprint boundaries are computed by integrating simplified equations of motion forward to the terminal altitude. Some of the advantages of predictor guidance are:

- Predictor guidance is independent of preflight simulation. That is, predictor guidance is self-contained and independent of the trajectory or landing site.

- The footprint and the enclosed destination makes a very desirable display from which the pilot may monitor the flight or control the vehicle. The continuous footprint display allows instant determination of alternate landing sites.

- The presence of a faster than real time simulator, the predictor, on-board the vehicle allows the future flight conditions to be determined and isplayed to the pilot.

- In general, the footprint predictor is not dependent on a simplifying constraint, such as constant drag or equilibrium glide. Thus, the footprint-defining trajectories may be optimized (of particular importance is cross-range) if facilities are available to do so.

The two simplifications made in footprint generation are: (a) approximating the cosine of flight path angle by unity, and (b) an approximate treatment of earth rotation effects. Preliminary simulation results have shown that the errors resulting from these approximations are acceptably small: less than 10 nautical miles in the worst case, and usually less than 5 nautical miles.

The organization of the footprint computation is shown in Figure 2.3; Figure 2-4 shows the over-all organization of re-entry guidance.

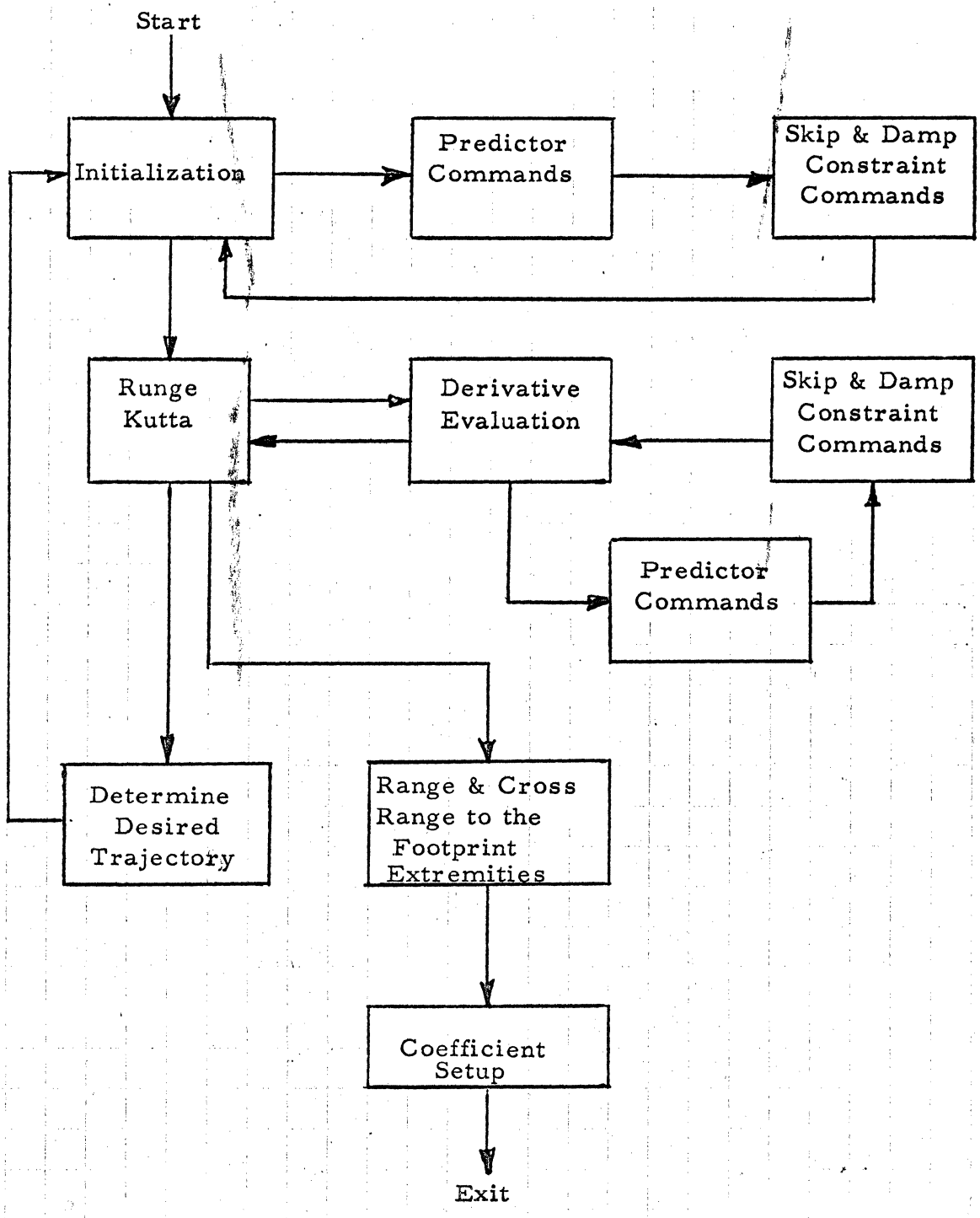


Figure 2-3. Footprint Predictor

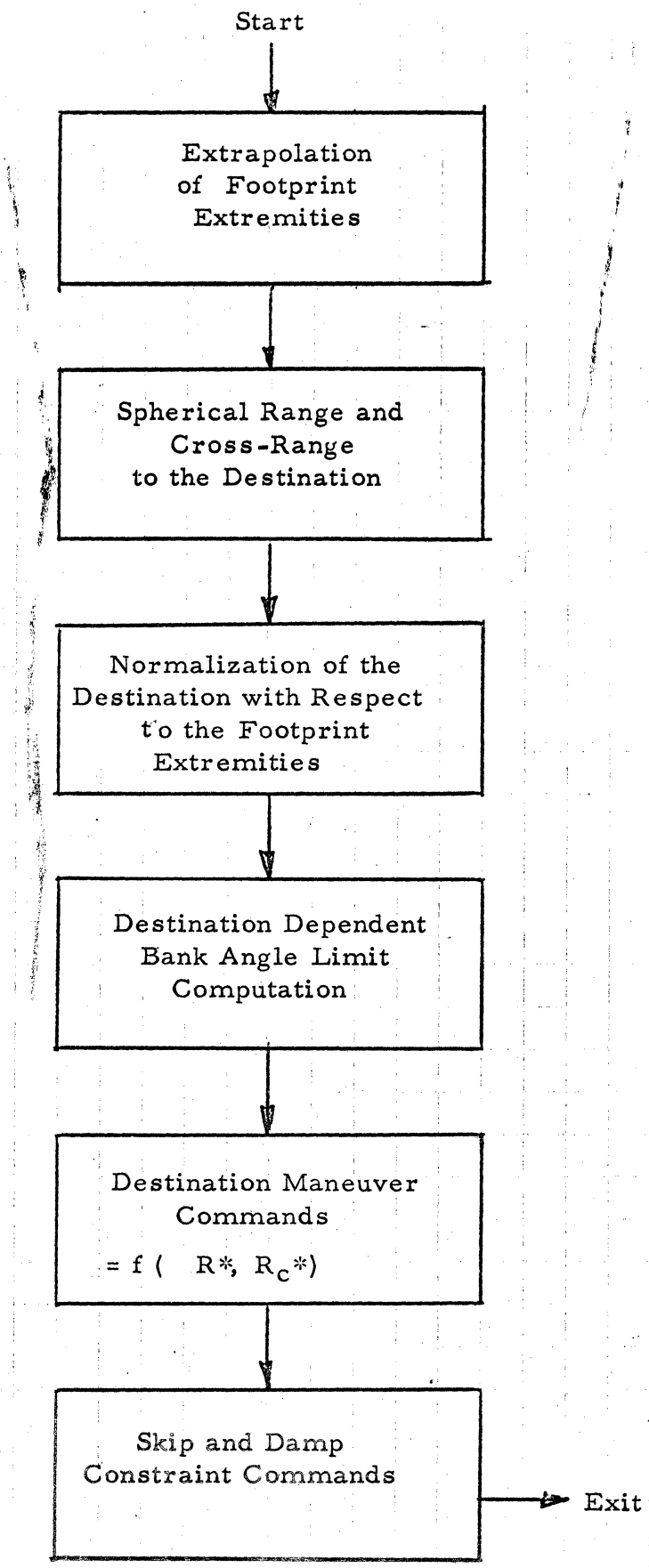


Figure 2-4.- Guidance Computations

2.2.10 RANGE SAFETY

2.2.10.1 Range Safety Requirements and their Relation to Quick Reaction

Range safety requirements must be met during the Pre-Launch and Ascent phases of all missions, except (possibly) for certain missions whose urgency is so great that a partial waiving of range safety requirements can be justified. This will not usually be the case.

Some requirements for range safety are not closely related to mission planning, guidance, or targeting, and will not present any serious problems relative to quick reaction, although they must be considered at various stages of the design and development of the vehicle and associated ground systems. Examples of range safety requirements of this type are:

Flight Termination System Approval
Missile Ground Safety Systems Approval
Non-Cooperative Launch Restrictions
On-Board Tracking Aid Requirements
Documentation Procedures.

Other range safety requirements present problems of varying degree which may possibly compromise the quick reaction concept. Some examples of these requirements are:

- Flight Plan Approval

  Data Requirements
  Lead Time

- Clearing the Range

  Clearing Designated Danger Zones
  Issuing Warnings to Aircraft and Ships in Approved Impact Areas

- Flight Termination Procedures

  Impact Limit Line Violation
  Land Impact Restrictions

The first two categories represent, basically, a timing problem (although Flight Plan Approval data requirements may impose a burden on the on-board system if this data is to be generated there) in that, according to present procedure, weeks are involved in gaining flight plan approval and warnings are issued days in advance. The last category can be viewed as mission constraints that must be considered in the planning of the mission, and requirements on the abort phase of manned missions.

For the purpose of assessing the effort of Range Safety requirements, the QRGT missions can be typed by the minimum desirable response time:

Type I Missions (Quick Reaction) Response time of approximately one day or more.

Type II Missions (Very Quick Reaction) Response time of the order of hours.

Type III Missions (Ultra Quick Reaction) Response time of the order of minutes.

The mission types can be defined by the point in the count-down at which the initial mission specification is received. For Type I the count-down has not begun, for Type II the count-down has progressed to a Hold point where the launch vehicle may be maintained in a ready state for an extended period of time (i.e., weeks), while Type III denotes the count-down has progressed to the last possible Hold before launch that can be maintained for some hours.

The Titan III presently has a hold capability of 30 days at t-195 minutes, 6 hours at t-45 minutes and 1 hour at t-1 minute (if repeated instrumentation and range safety checks are waived).[1] Thus a Type II Mission would correspond to the Hold at t-195 minutes and a Type III Mission to the Hold at t-45 minutes or possibly at t-1 minute if the Hold capability can be increased by, say, a factor of 5 to 10.

In the case of manned mission the major factor in response time may be the preparation and checkout of the crew.

For Type I Missions, attainment of quick reaction cannot be allowed to degrade present standards of range safety, although changes in the procedures are permissible and probably necessary. Missions of Types II and III may be of extreme urgency, justifying the waiving of some range safety requirements if necessary.

Another consideration is the desirability of reducing the amount of ground equipment and personnel required for support of launch operations. This is especially important for missions which may be launched from dispersed military sites. It is less important, though still desirable, for launches from the Eastern or Western Test Ranges, where elaborate facilities and organizations for range safety presently exist, and will probably continue to exist as long as R&D flights are made.

---

1 Demoret, R. B., "Titan III for Unmanned Space Exploration", Symposium on Unmanned Exploration of the Solar System, Denver, 1965, Preprint No. 65-37, p. 20.

2.2.10.2 Possible Approaches to the Range Safety Problem with Quick Reaction

The problem of attaining quick reaction while maintaining maximum range safety is complex and difficult, and hence requires extensive study. However, from a preliminary examination of the problem, it appears that there are a variety of possible approaches which may (in combination) provide an acceptable solution. Some of these approaches represent a natural evolution of the procedures presently used for variable-azimuth launches, and for sequences of similar missions.

The problem of providing adequate assurance of range safety prior to launch is analogous to the problem of validating flight programs, and suggests a similar approach. The Quick Reaction concept of flight program validation is based on off-line validation for a generic mission supplemented by a less extensive on-line validation when the mission becomes precisely defined. Similarly, information required for range safety purposes can be generated off-line for a generic mission in the form of sets of trajectories which represent all possible limiting cases, etc. and supplemented by additional information produced on-line before launch.

The problem of assuring range safety after launch requires a dual approach. The on-board computer can be required to consider range safety constraints in its mission planning and to enforce these constraints during flight, but range safety personnel must also have all information and facilities needed for detecting anamalous behavior, predicting possible consequences, and commanding flight termination if necessary.

A number of specific suggestions are listed below for changes to be made, or new capabilities to be developed, in the areas of Range Safety Operations, Launch Operations, Mission Planning, and Guidance. The items relating to Mission Planning and Guidance will be considered in detail in the QRGT study. Changes in operational procedures involve many complex issues which are outside the scope of the QRGT study; in these areas, it is proposed only to identify problem areas and suggest possible approaches.

- Changes in Range Safety Operations

  Present to the Range Safety Office, at the time the vehicle configuration is defined, generalized trajectory data which indicates vehicle/payload capabilities. This information can then be used to identify mission planning constraints (such as allowable launch azimuth sectors, dog-leg maneuvers, staging points, etc.) which will then become a part of any particular mission specification.

For Type I Missions, the Range Safety Office can be presented more specific data by planning the mission in advance (day or so) for a series of launch times. Range Safety can then select acceptable launch times within the planned launch window.

Automate the pre-generation of information required for Range Safety purposes, so it can be produced rapidly, and design its character and format so it can be rapidly understood and conveniently used. Automated production of display overlays should be considered or electronic substitutes for overlays.

- Changes in Launch Operations

Provide a Mission Planning display at the Range Safety Officer's console. Preliminary trajectory information for a mission being planned can be supplied in a form that indicates: ascent ground path including impact dispersions, normal impact areas and impact areas corresponding to the planned cutoff point of multiple burn stages.

Provide a ground computer which can duplicate the operations of the on-board computer, but has greater resources of speed and memory. Use this to generate fast-time simulations of the expected trajectory, both for nominal performance and for certain variations from nominal. Correct these simulations from tracking data, and use them for (a) predicting range safety villations before they occur, and (b) detecting deviations from expected behavior.

If there are aircraft or ships in the range which cannot be warned in time, or cannot obey a warning in time, consider requiring the on-board computer (in the pre-launch phase) to avoid such hazards by varying its plan, i. e., by varying launch time or launch azimuth or both.

- Mission Planning

Provide the capability of accepting Range Safety constraints in the on-board planning function. This involves not only allowable launch azimuths but also such factors as ascent ground track limitations, and predicted impact points for discarded stages or jettisoned components.

- Guidance

During the Ascent Phase, employ a Mission Safety mode of Mission Planning for impact prediction and abort mode determination. The Guidance should constrain the trajectory to obey range safety constraints if possible; otherwise, it should recommend (or initiate) an abort.

2.3. COMPUTER INTERFACE (TASK 3)

2.3.1 INTRODUCTION

The software system required to meet the multi-mission requirements of the period 1970-75 is a general and flexible multi-programming system capable of quick-response to requirements for any of a wide range of mission classes using a variety of booster-sensor-FSV configurations. It features a high degree of modularity in both support and applications software essential to a quick-response multi-system capability. The on-board executive system maintains centralized control of the interface between software modules and between the CPU, I/O devices, and vehicle systems. The ground operating system provides easy-to-use services for the preparation, debug, test, integration, and validation of software for on-board use as well as for ground mission analysis tasks.

The software system's purpose and design requires the availability of computer systems larger, faster, and of more advanced organization than those previously available. Such systems are now available and will undoubtedly be in widespread use during the period 1970-75 for ground-based (e.g., IBM System/360), airborne (e.g., Mark II avionics system and EA6B aircraft system), and spaceborne (e.g., MOL) applications. They feature increased CPU speed, larger and faster main storage units, more powerful instruction sets, standard I/O interface, and powerful priority interrupt systems. An on-board auxiliary memory unit (tape or drum) is desirable (a) to provide recovery capability in the event main storage data is destroyed or inadvertently altered, (b) to reduce main storage size, (c) to facilitate the extraction, integration, validation, and cataloging of subsets of the multi-mission software library to form specific vehicle libraries, and (d) to facilitate reconfiguration of software modules at pre-launch time or between mission phases in response to a respecified mission plan. No specific computer system need be presupposed but the features mentioned are considered essential. Compatibility between on-board and ground-based systems is highly desirable but not essential. The feasibility of the software recommended will not be compromised because of the specific equipment ultimately specified so long as the specified system possesses the essential characteristics mentioned. Computer system specification is a continuing effort in the QRGT study; progress to date in this area is discussed in Section 2.3.6.

The essential elements of the total solution to the problem of providing a quick-response, multi-mission capability are listed below. In some cases, these elements relate more directly to mission analysis than to computer hardware/software requirements, but virtually no part of the total solution can be divorced from the computer hardware/software study effort.

- A mathematical/logical formulation of mission functions which, to the extent possible, is generally applicable to any mission class or vehicle configuration.

- A clearly-defined logical hierarchal structure of all mission elements as they are represented and implemented by software modules catalogued on the multi-mission software library.

- Clearly defined modularity levels and standard structure of software modules in terms of inter-module linkage, competition for computer system resources, service class/ priority, and status monitoring.

- Ground capability for quickly and easily extracting software modules from the multi-mission library, including the integration and validation of modules so extracted to form a subset of all software required to support all capability of a specified booster-sensor-FSV configuration.

- On-board capability for mission planning at pre-launch time and replanning during the mission. The on-board executive system must provide capability for the integration and checkout of the modules required to meet the (new) specifications output from on-board mission planning routines.

- On-board, real time, orderly and centralized management of the control and data interface between software modules, I/O devices, and vehicle subsystems. This includes allocation of computer system resources and resolution of conflicting demands for system resources.

The above list is not exhaustive but includes those elements which differ considerably from the requirements of previous systems and includes those elements most closely related to the computer hardware/software study task. The capability required is achieved with larger and faster (than are in operation to date) airborne computer systems of advanced organization operating in conjunction with a general, flexible, and modular software system.

The QRGT software consists of two major subsystems: the Ground Operating System (GOS) and the On-Board Executive System (OBES). These are discussed separately in Sections 2.3.3 and 2.3.4 respectively.

The GOS operates only in the ground environment. Its purpose is to provide complete and easy-to-use services for the preparation, debug, test, cataloging, maintenance, integration, configuration, and validation of programs for on-board use or for mission analysis. Its overall organization is illustrated in Figure 2-5.

The OBES operates on-board during all mission phases. It is the control center for all on-board software and applies generally to applications software for all QRGT mission classes and vehicle configurations. OBES provides orderly and centralized management of control and data interfaces between software modules and between software, I/O devices, and vehicle subsystems. Its overall organization is illustrated in Figure 2-6.

Figure 2-7 presents the major components of each of the two subsystems and illustrates the inter-and intra-relationship between these components and subsystems.

2.3.2 SUMMARY

During the second quarter of the study, the effort has been concentrated on the over-all software organization and beginning of the computer sizing task. The over-all structure of the on-board executive system is presented in Figure 2-6. It is supported with functional flow diagrams and descriptions of the major routines which comprise the complete system. Included within the descriptions are estimates of the core storage and execution times required for these functions. In the area of the support software (GOS), definition of the major capabilities which must be added to a normal operating system has been completed. These include: (1) the Multi-Mission Library; (2) the Configurator which selects individual subprograms from the Multi-Mission Library and links them together into a function or mode program; (3) a description of the additions to the language translators, like an assembler, in order to produce object code which is required by the on-board executive system; and (4) the simulation capability required to validate flight programs. Where applicable, flow charts have been developed to show the functional operation of these facilities. The next effort on software definition will be devoted mainly to developing the detailed design of the Multi-Mission Library and the Configurator. Also there will be a continuing effort on the OBES to insure that it is cost effective in terms of storage and CPU relative to the QRGT missions.

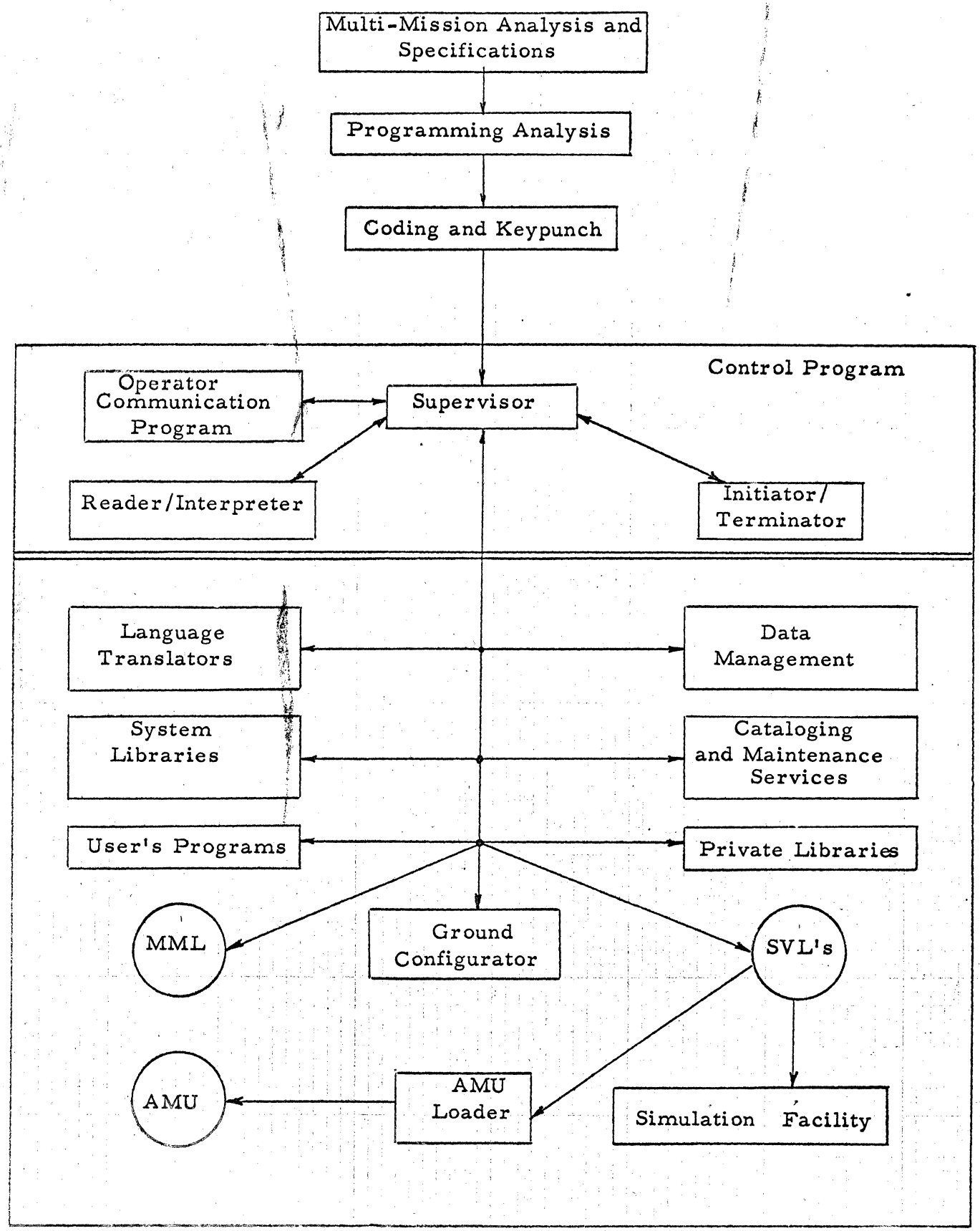


Figure 2-5
Overall Organization of the Ground Operating System

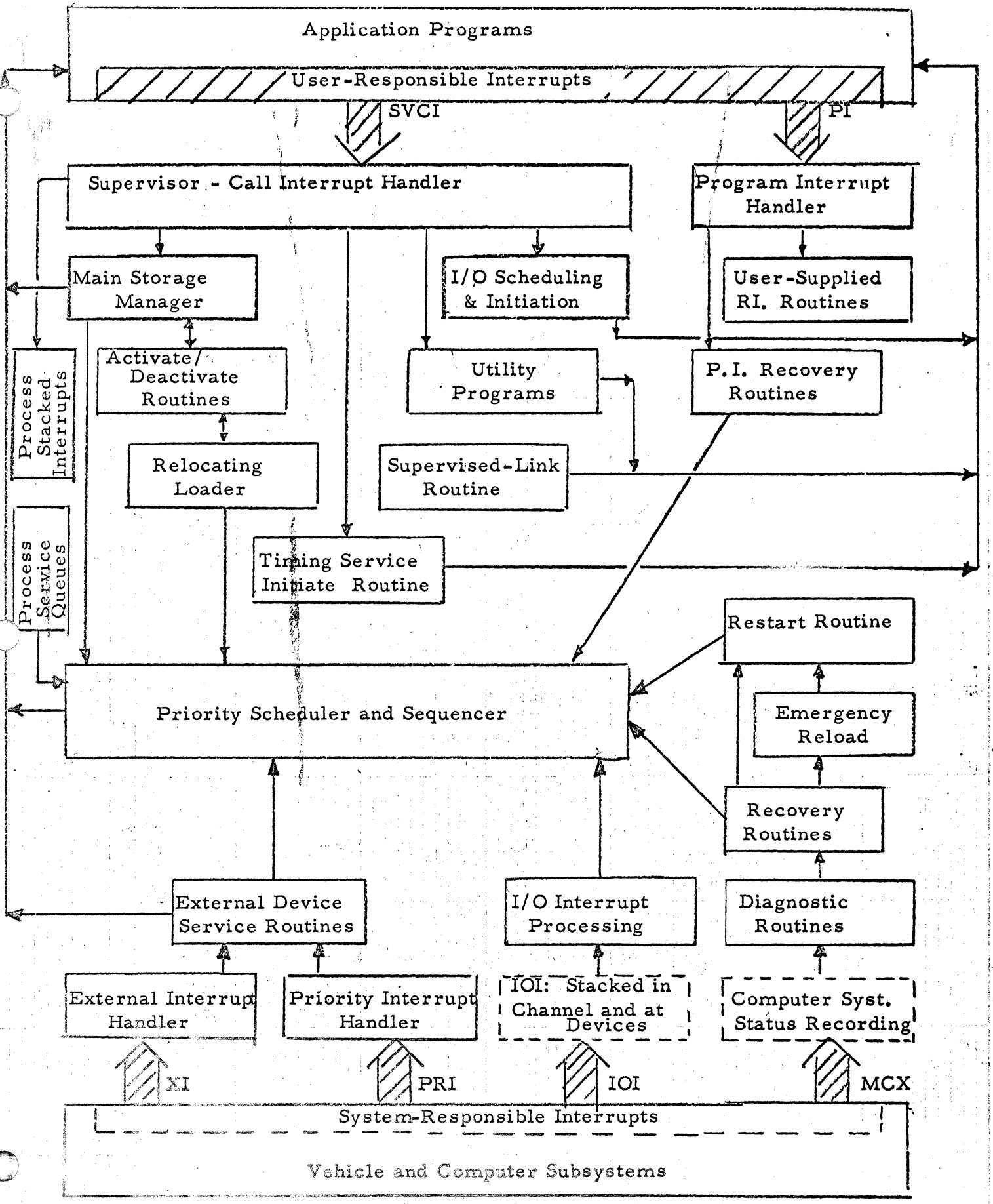


Figure 2-6
OBES Structure and Flow

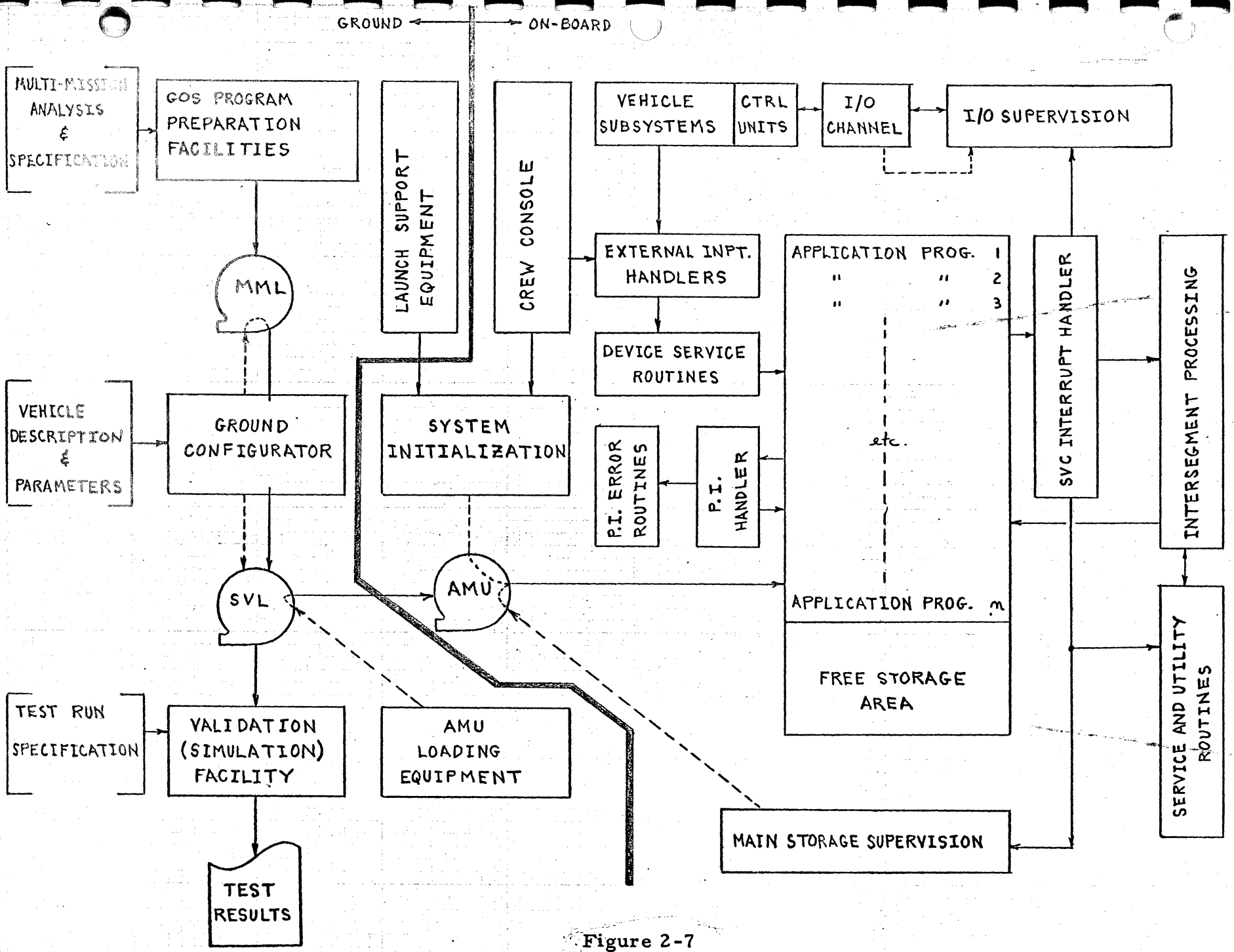


Figure 2-7
QRGT SOFTWARE - RELATIONSHIP OF MAJOR COMPONENTS

The computer sizing effort is now in progress. The best estimate of the computational requirements upon the computer has been identified down to phases of the mission. From this we are able to survey previous studies such as MOL and SSGS to obtain any available data relative to the computer sizing study such as flow diagrams or actual counts. Trial programming for two major areas is well under way (Gemini Re-Entry and Digital Autopilot) and estimates for other functions have been obtained from the MOL or SSGS studies. The results to date on this effort are presented later in this section. The actual size and speed estimates are classified and are reported in "QRGT Computer Sizing Estimates and Mission Descriptions" to be delivered under separate cover.

The following paragraphs describe the results of Task 3 efforts during the second quarter of the study.

2.3.3 THE GROUND OPERATING SYSTEM (GOS)

Standard "third-generation" operating systems available for commercial use provide much of the capability required for the QRGT ground operating system. They provide comprehensive support services to programmers with only minimal restriction on the manner in which a problem is organized and described. Since these systems are widely used and readily available, use of applicable components for the QRGT system provides the advantages of:

- Avoiding the considerable cost incurred in developing and implementing system components already available.
- Simplified coordination of effort and avoidance of redundant effort among contributors to analysis and operational software.

Existing operating systems do not, however, provide the full capability necessary to meet QRGT objectives. Changes to existing facilities and addition of new capability are necessary to adapt a standard system to new requirements unique to QRGT. The major areas of change are:

- Programming languages and translators for programming the on-board computer.
- Construction and maintenance of a Multi-Mission Software Library (MML).
- Extraction of MML modules to form a software subset for a specific vehicle configuration.

2.3.3.1 Programming Languages and Translators

All programming language facilities such as assemblers, macro facility, PL/1, FORTRAN, ALGOL, etc. could be retained at least for mission analysis programming. QRGT on-board programs impose additional requirements not now incorporated in a single programming language. Figure 2-8 shows features available in each of several commonly-used languages and features considered desirable for a QRGT higher-level language.

A common higher-level language for space applications is under study in separate efforts by SDC and IBM. Both studies are in the early stages of definition and specification. It is possible that one or both of these languages may be suitable for the QRGT system, but a significant implementation effort remains to be accomplished before either would be ready for use.

The extent of the effort involved in providing additional language translation facility in the QRGT ground-operating system depends heavily on the degree of compatibility between ground-based and spaceborne computer equipment. If a high degree of compatibility exists, each existing translator could be modified to operate in either a ground or on-board mode. In the on-board mode, the translator would produce object code (a) using only the instructions available in the O/B computer, or (b) flag unavailable instructions as coding errors requiring correction and reassembly or recompilation, or (c) substitute system macros to generate the on-board code required to perform the function of an unavailable instruction. If strong compatibility is not required, it would be necessary to provide new assemblers and/or compilers for the on-board computer specified.

The language(s) used for QRGT programming should eliminate, to the extent possible, all distinction between programs written during study and analysis and those written for operational on-board use. The extent to which this goal can be achieved also depends heavily on the degree of compatibility between ground-based and on-board computers.

2.3.3.2 The Multi-Mission Software Library (MML)

All software modules prepared for on-board use for all mission classes and vehicle configurations are stored and catalogued on an I/O device (tape or disc) accessible to the GOS. A ground configurator is provided to extract and integrate a subset of MML modules to produce a Specific Vehicle Library (SVL) containing all modules necessary to support all mission functions within a specific vehicle's capability.

Advanced Language Features Desirable for General QRGT Use		OS/360 Language Specs.		
		ALGOL	PL/1	Fortran
Data Types	Scaled Fixed Point		L	
	Integer	x	x	x
	Floating Point (Real, Complex, and Double)	x	x	x
	Bit String		x	
	Character String	x	x	L
	Mixed-Mode Expressions	x	x	
Data Aggregates	Arrays	x	x	x
	Structures		x	
	Arrays of Structures		x	
Operators	Arithmetic: + - * / **	x	x	x
	Comparison: =	x	x	x
	Bit String: (Complement) (And) (Or)		x	
	Character String: (Concatenate) (Extract Substring)	x	x	
	Matrix: (*) (/) (•) (x) (+) (-) (Transpose)			
Subroutines and Linkage	Link with other Languages		L	x
	Programmer-Specified Linkage Code			
	In-Line Assembly-Level Code			
	Standard Built-In Functions	x	x	x
	Standard Subroutine Library	x	x	x
	Generic Functions		x	
Input/Output	Remote I/O Lists		L	L
	Subscripted I/O Lists		x	x
	Default Formats		L	L
	Record-Oriented I/O	x	x	x
	Stream-Oriented I/O	x	x	
Macro Facility	Define a Function or Procedure		x	x
	Execute a Remote Statement			
	Programmer-Defined Operators			
Storage Allocation	Static	x	x	x
	Allocate on Entry		x	
	Free on Exit		x	
	Programmer-Controlled		x	
	Overlay		L	L
Multi-task Operation	Attach a Task		x	
	Terminate a Task		x	
	Wait for Event		x	
List	Pointer Definition & Manipulation		x	
	Path Vectors		x	

Figure 2-8

Advanced Language Features Desirable for General QRGT Use

x - Adequate

L - Limited

Figure 2-9 is a functional flow diagram of the ground configurator. Its inputs consist of data which identify and describe booster type, reference and control systems, sensors, and final stage vehicle. Using these inputs and a directory which appears as the first record on the MML, the ground configurator searches the MML, extracts the required modules, combines lower-level modules into programs, prepares an SVL directory for all programs formed, and writes the results on the SVL device. The SVL so formed is later used as input to validation programs and, after validation, is loaded on a vehicle AMU to be installed as part of the vehicle configuration.

2.3.4 ON-BOARD EXECUTIVE SYSTEM (OBES)

The on-board executive system is the control and service center for all on-board software. It maintains, in real time, a centralized and orderly control of the interfaces between application programs, vehicle devices, ground controllers, and crew members. Only the capability deemed necessary to meet QRGT objectives is provided. When fully implemented, the system will provide the required capability at the lowest possible cost in CPU time and storage space. Only minimal restrictions on program structure and programming conventions are imposed on applications programmers.

2.3.4.1 System Structure

The executive system is comprised of six categories of software:

- Routines for Handling Interrupts

  Their functions include machine status save/restore, recognition of the action or service commanded or requested by each interrupt, initiation of the required action or service, and return of control to the interrupted program.

- CPU Time Supervision Routines

  Their function is to determine when and in what sequence the various computational tasks (programs) are given CPU time. These routines provide capability for ordinary sequencing of programs, servicing of programs on a priority basis, and for scheduling programs with precisely-cyclic or immediate-response requirements. The sequencing and priority scheduling routines insure that repetition rates and duty cycles of all programs are satisfied.

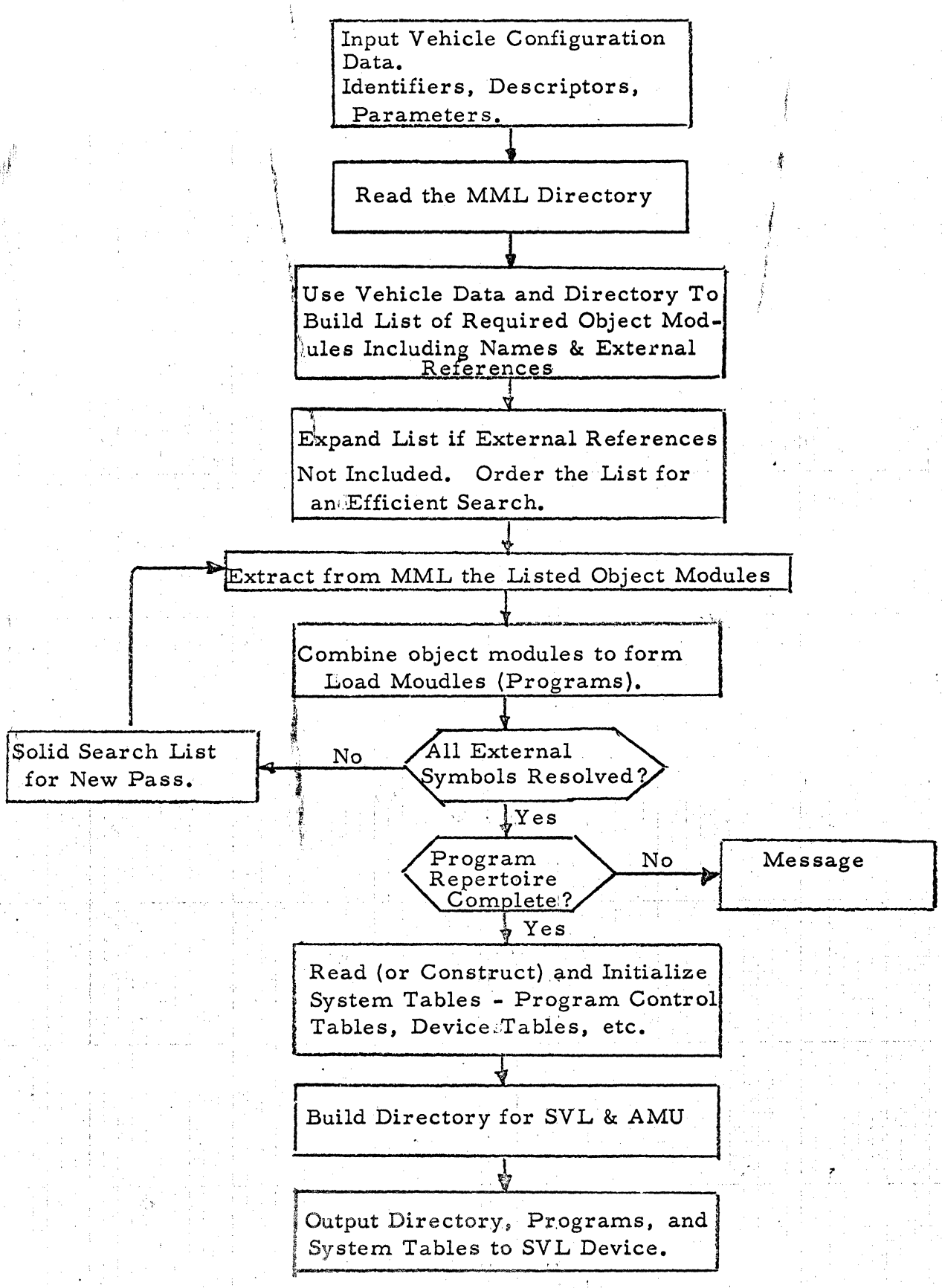


Figure 2-9
Ground Configurator Functional Flow

- I/O Supervision Routines

  They provide I/O services requested by application programs, and process and analyze all I/O interrupts resulting from either normal or abnormal termination of an I/O operation or from a signal originating in an I/O device.

- Main Storage Supervision Routines

  They maintain a record of the identify and extent of all programs in main storage, initiate the loading and relocation of programs, allocate and free main storage, and provide storage cleanup when it is both possible and necessary to make free areas contiguous.

- Other Interrupt Service Routines

  These fulfill requests or commands indicated by interrupts. These are mainly services provided in response to supervisor call interrupts originating in an application program, but they include also those services necessary to respond to interrupts from the timer, alarm and attention signals from vehicle subsystems, and interrupts indicating a computer system malfunction.

- Utility Services

  They include such subroutines as trigonometric functions, matrix operations, data conversion and formatting, etc. to the application programs.

In addition to the software categories listed above, OBES makes use of system tables to record the utilization and status of the CPU, the I/O channel, I/O devices, main storage, and software resources.

Before proceeding into the details of the OBES, a brief discussion of "overhead" costs (in terms of CPU time and storage requirements) incurred by using the centralized on-board executive system is in order. All of the capability provided is deemed necessary to meet study objectives; in this sense, none of the CPU time and main storage need be considered overhead. Previous on-board software systems were designed to meet a particular objective with all computer and vehicle equipment and software allocated to a specific and restricted mission class. Many functions, now considered supervisory, were previously dispersed (often redundantly) throughout the program; other functions not then required are essential in the QRGT system. The QRGT system must eliminate these circumstances

which contributed to the difficulty of changing programs and must at the same time provide more capability to support many mission classes and vehicle configurations.

The "overhead" incurred by use of the on-board executive system: (a) makes it possible for a single system to support a wide range of application programs for various mission classes and vehicle configurations, (b) provides many essential services which were previously considered a part of the applications programs, (c) permits last-minute specification or replanning of particular missions or phases and (d) relieves the application program of the intricate problems of core allocation, program scheduling, I/O handling, etc. In typical use, the on-board executive system requires about 3000 words of main storage and about 6% of available CPU time. A breakdown, by system function, of the current estimated costs is given in Figure 2-10.

Looking at the requirements of the QRGT system, it seems extremely likely that any attempt to perform on-board computational functions without a centralized executive system may fail to meet the major objectives of quick-response, generality, flexibility, and on-board (re)planning of missions or mission phases. One can only speculate on the outcome of such a course of action, but at best it would result in a somewhat haphazard evolution of a centralized executive system which would compromise established QRGT objectives. It is better to recognize early the need for the system and design, develop, and implement it in an orderly manner and in conjunction with the mathematical/logical formulation of mission functions.

2.3.4.2 System Operation

What comprises a flight program depends primarily on the application programmer's decision, which should be influenced by the fact that a program is a basic unit in terms of:

- Competition for CPU time.
- Loading from the AMU.
- Competition for main storage space.
- Relocation within main storage.
- Queuing of service requests.

Note: These estimates are preliminary and are provided prior to detailed OBES Design. They are based on judgment of application program requirements and IBM experience with MOL and OS/360.

OBES Service	Main Storage (Words)	CPU Time (MSEC/SEC)		
		Low	Typical	High
System Initiate/Initialize	300	--	--	--
Interrupt Handling (Excluding I/O & Segend)	200	3.0	6.0	9.0
Intersegment Supervision	150	10.0	12.0	14.0
Scheduling	220	20.0	25.0	30.0
Main Storage Supvn. Control	150	--	--	6.0
Program Loading	300	--	--	4.0
Program Relocation	50	--	--	120.0
I/O Request Processing	150	2.0	8.0	14.0
I/O Interrupt Analysis	150	2.0	7.0	12.0
Error Routines	100	--	--	5.0
Timer Services	40	--	2.0	5.0
Linkage Supervision	40	--	2.0	5.0
Keyboard/Display Support	200	--	--	26.0
Printer Support	100	--	--	12.0
System Tables	800	--	--	--
Total	2950	37.0	62.0	262.0
CPU% Utilization by OBES (Excludes I/O Interference)		3.7% Low	6.2% Typical	26.2% High

Figure 2-10

OBES Storage and CPU Requirements

Any variation desired by the programmer at a lower level than the basis described above is governed by the flow established within his program at coding time. Normally, but not necessarily, a program will be the implementation of a specific mode or mission function; for example, "navigation-prime mode", or "navigation-backup mode", or "attitude reference and control", or "display departure time vs. delta V required", etc. The choice of units existing as programs on-board does not prohibit the cataloging of lower-level modules (elements) on the MML. The GOS ground configurator combines such lower-level modules into programs in the process of extracting and integrating an MML subset to form an SVL. Such lower-level modules to be combined into programs might include, e.g.: "Basic gravity model", "gravity model with 3 higher earth-harmonic terms", "trapezoidal integration", "4th-order RK integration", "Kepler's Problem solution", or "Lambert's problem solution", etc.

To reduce the complexity and increase the computational efficiency of OBES routines, onboard programs are structured to consist of four parts: a program control table, program text, program common, and a relocation dictionary. This structure has been chosen to: (a) simplify the sequencing and priority scheduling procedures, (b) reduce overhead, (c) maintain flexibility in changing program mix, and (d) make it possible to satisfy real-time operational constraints imposed by each program.

- The program control table contains information required by OBES to service the program.
- The program text is the executable code required to implement the program's function.
- Program common is an area for the communication of data between segments of the program.
- The relocation dictionary contains pointers to location-dependent quantities within the program. These are required for relocation and initial loading.

The above refers to the physical structure of a program. There is also a logical structure which provides for text segments, only one of which is executed each time the program is scheduled. Segment break points are determined on the basis of execution time to insure that OBES regularly receives control within a specified maximum time interval. Segmentation insures that high rate sampling requests can be satisfied.

Basic flow of control from program to program is governed by the order in which the system sequencer and priority scheduler grants CPU time to each program. This is determined by information provided in each program's control table - primarily repetition rate, duty cycle, and priority. Unsolicited external interrupts, such as subsystems alarm or attention signals, may change system flow by requiring execution of a device service program. In some cases, these signals require immediate response; in other cases the required service can be scheduled as a priority or background task. The system grants CPU time to programs on an interleaved basis in a manner which satisfies the real-time constraints imposed by each program. Each time a program is granted CPU time it can retain control for only one segment of the program. High repetition rate requirements are met by granting CPU time as frequently as necessary. This might often require that a program be given several (or many) adjacent time slots.

Interrupts always transfer control to OBES; furthermore, they provide the only means by which the OBES can obtain control. The most common sources of interrupts during normal system operation are application program's issuance of supervisor call interrupts representing a request for OBES service. Application programmers request services by coding a macro-instruction such as READ, WRITE, ENDSEG, LINK, LOAD, etc. Each such macro is represented by machine code as a supervisor-call (SVC) instruction with which is associated a code field to indicate to the executive system which service is requested.

In addition to the SVC interrupts described above, there are other classes of interrupts which cause control to be transferred to OBES. Sources of these interrupts include the timer, the I/O channel, I/O devices, vehicle subsystems, malfunctioning computer system equipment, abnormal program operation (e.g. overflow), and crew member commands. All interrupts may be classified in two categories:

- User-Responsible Interrupts

  This category includes all user-issued SVC interrupts and all interrupts resulting from abnormal completion of a program operation. OBES response to SVC interrupts consists of the execution (including transfer to and return from) of system service routines designed to implement all macro-instructions available to application programmers. If earlier or higher priority requests are utilizing a service requested, the new request is queued and provided later

when the service becomes available. Most services which involve only the execution of a routine are implemented by re-enterable code and can thus be used "simultaneously" by several requestors. OBES response to abnormal program operations (such as overflow) depends on whether or not the possibility of the abnormal condition was anticipated by the application programmer. For such conditions anticipated, OBES would cause execution of a user-supplied analysis/recovery routine; if unanticipated, OBES would attempt recovery by use of system-supplied analysis/recovery routines.

- System Responsible Interrupts

  This category includes all interrupts originating outside the CPU. Depending on the interrupt class and priority level, response to system interrupts may be immediate, scheduled, or background. Most system-responsible interrupts can be held pending in an I/O channel or device with provision for later allowing them to occur and be processed - normally between application program segments. Figure 2-11 shows the general method of handling all interrupts.

Each program is divided into logical segments. When each application program returns to the OBES at the end of a segment the OBES performs intersegment processing, consisting of the following:

- Updating the execution (CPU) list.
- Unmasking and processing of stacked interrupts.
- Processing of queued requests for services.
- Selecting the next program to be given CPU time, including reinitialization of the scheduler if programs have been added to or removed from the multiplexing loop.

CPU time required for intersegment processing and for processing system-responsible interrupts is not considered a part of an application program segment time. CPU time required to respond to user-coded macros is considered a part of the segment time for the segment issuing the request. For I/O requests, the CPU time required to initiate the operation plus CPU interference caused by data transfer, is considered segment time. CPU interference due to data transfers across the I/O

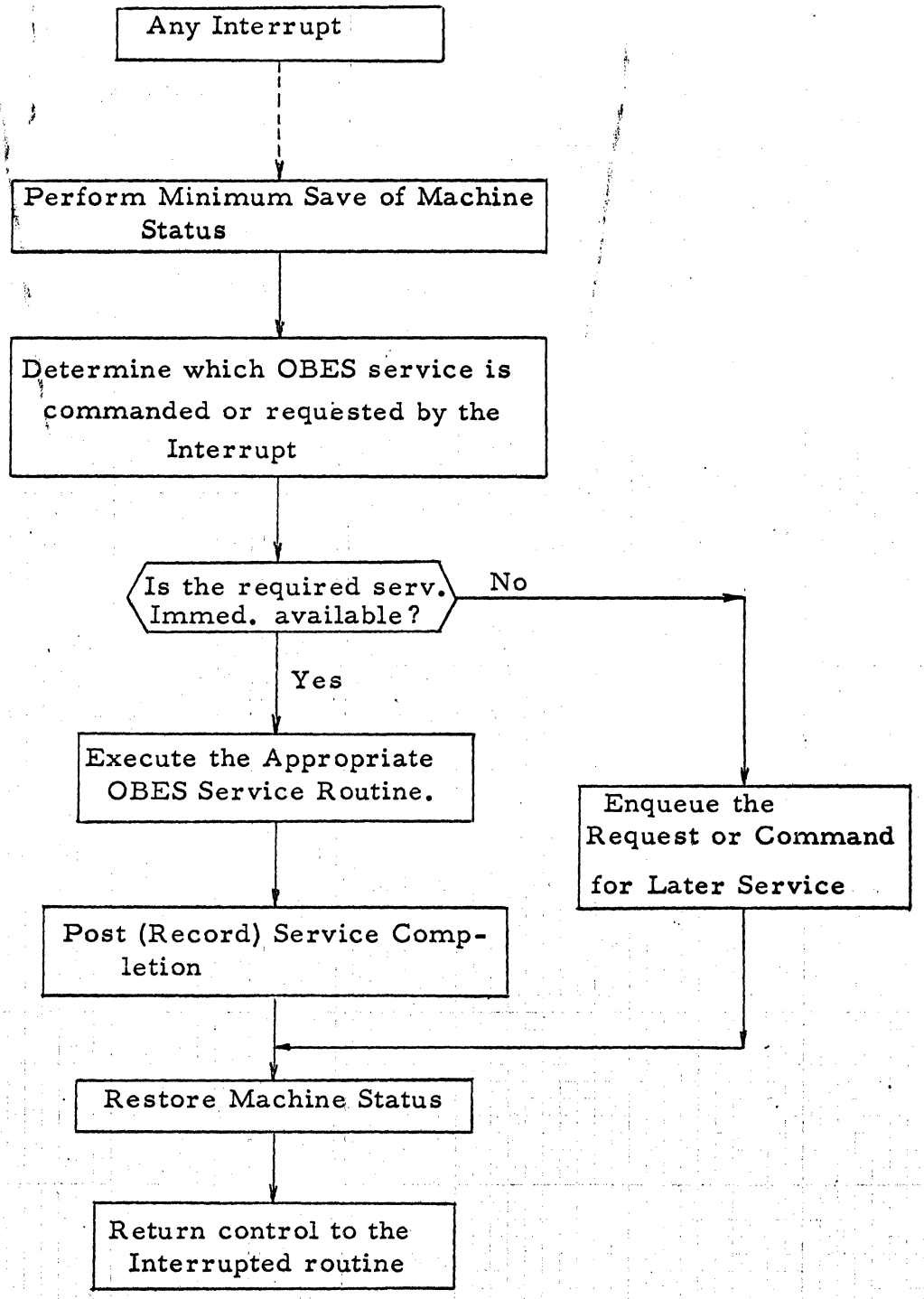


Figure 2-11

General Description of OBES Response to Interrupts

interface is not entirely predictable; data transfers occurring during one segment of a program may be in response to an I/O request from a different segment of the same program or from a different program. However, this presents no problem since a timer interrupt is provided to limit any segment to an established maximum segment time. If maximum segment time is used before an end-segment macro is issued, the timer interrupt insures return to OBES and the exit from the segment does not effect the program's execution rate. The use of timer interrupts for always establishing segment time is prohibited because of the excessive overhead it would impose on OBES. If this were permitted, OBES would be required to perform a complete status save and restore since the point of interruption would be unpredictable, thus making it impossible to save/restore only to the limited extent required. Use of segmentation, coded by application programmers using segment-end macros, permits a significant reduction in OBES overhead since it permits the application programmer to prescribe only the limited save/restore required at the point of issuance of the segment-end macro. Figure 2-12 is the functional flow diagram of the intersegment processing.

2.3.4.3 CPU Time Supervision (Scheduling)

Routines for CPU time supervision schedule the concurrent execution of most tasks by a fast multiplexing procedure. This procedure satisfies the requirements of any program mix consisting of nominally (but not precisely) cyclic programs with different assigned relative priorities. Programs not in the regular multiplexing loop are accommodated according to their special requirements by altering, only for so long as is necessary, the normal flow through the multiplexing procedure. Figure 2-13 describes OBES supervision of CPU time for tasks which are multiplexed, precisely-cyclic, immediate-response, unsolicited, or background. Figure 2-14 is a time-line sketch showing when and in what order CPU time is allocated to a sample mix of programs.

Each program in the multiplexing loop has associated with it the following descriptors:

- Number of segments n_i in program P_i.
- Repetition rate r_i, in program executions per major cycle (e.g. 1 second).
- Priority p_i of program P_i relative to other programs. (Low values of p_i indicate higher priority).

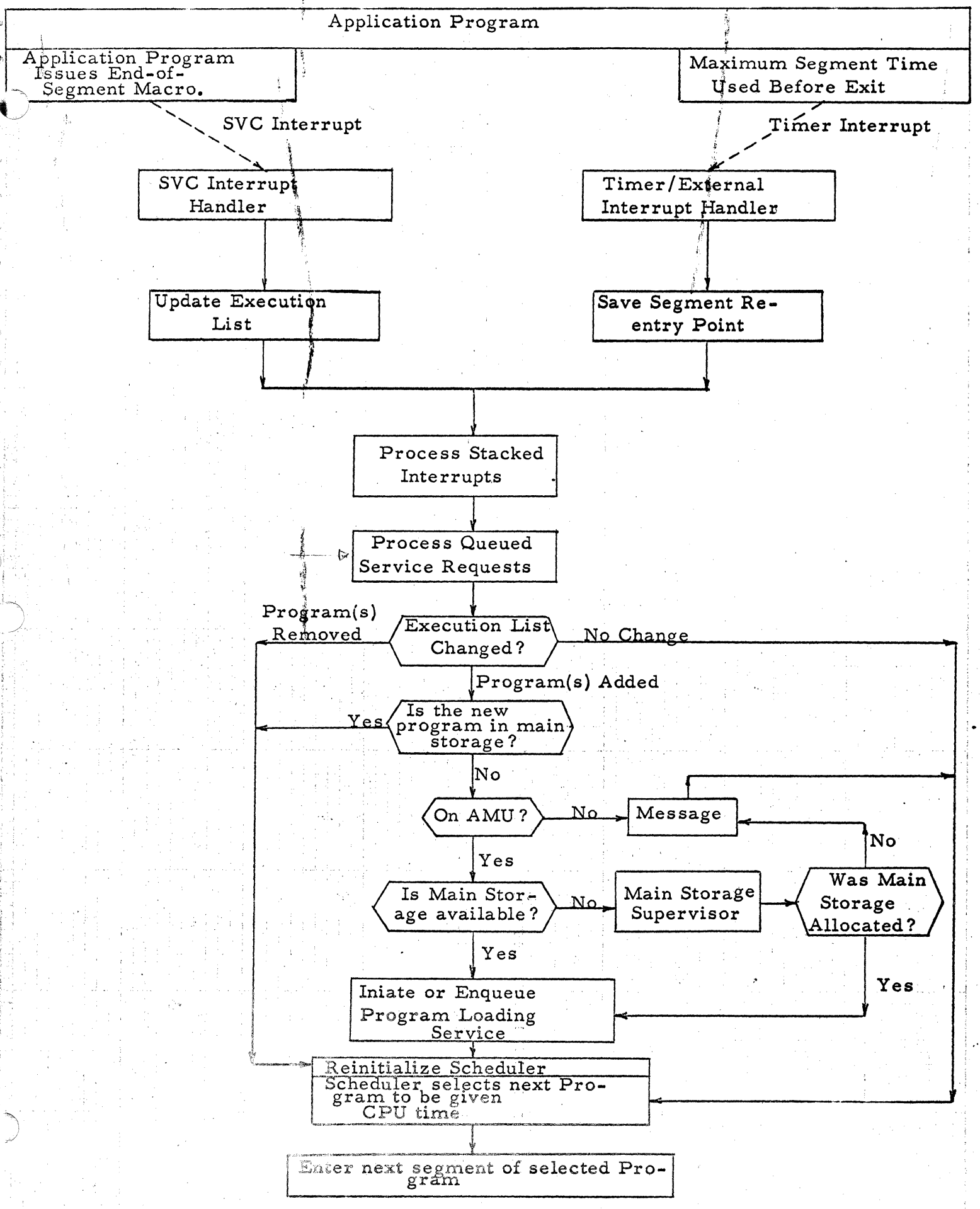


Figure 2-12- Intersegment Processing

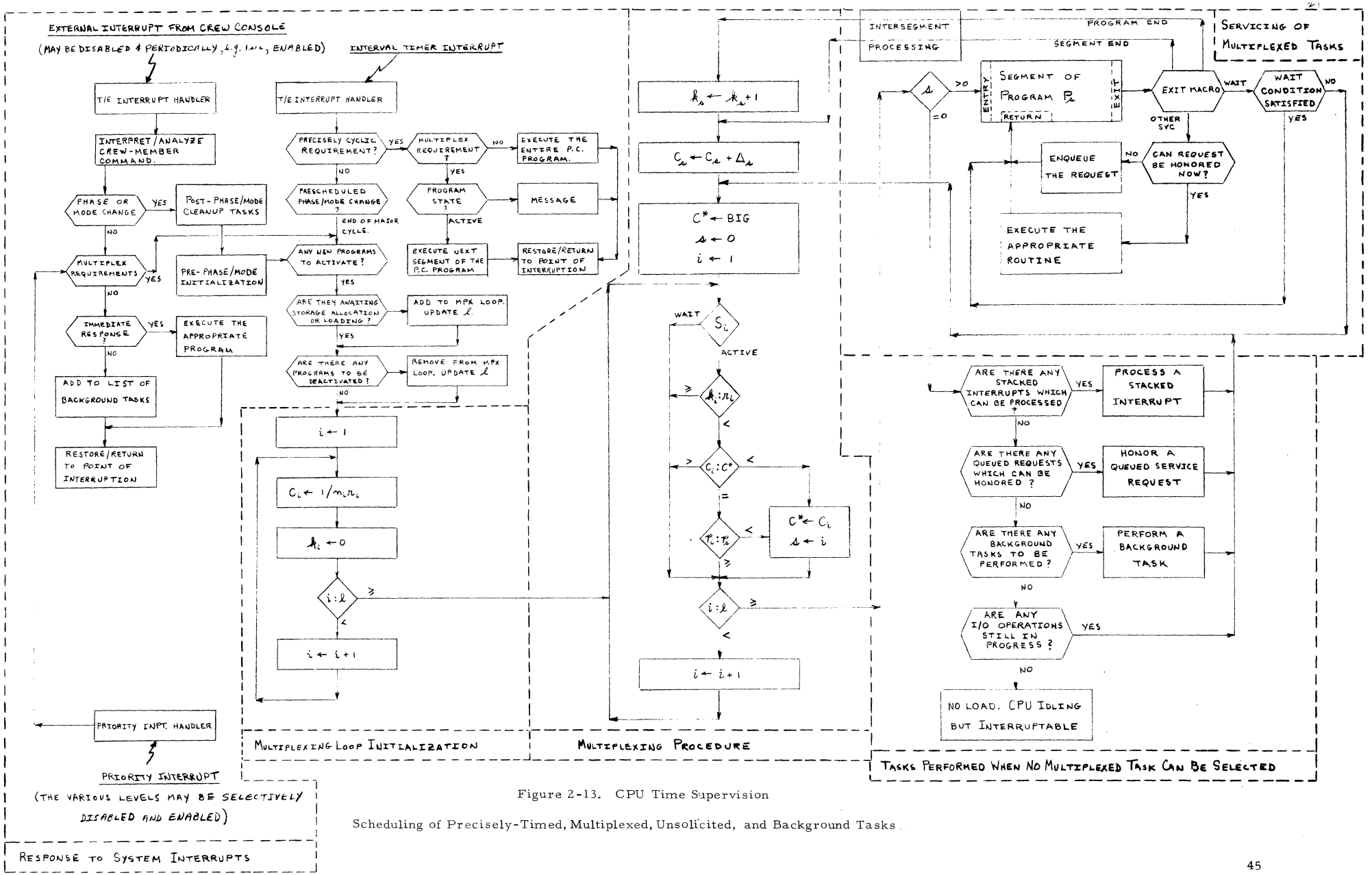


Figure 2-13. CPU Time Supervision

Scheduling of Precisely-Timed, Multiplexed, Unsolicited, and Background Tasks

PROGRAM	PRIORITY	REP. RATE	NO. OF SGMNTS	SEG. TIME (msec.)				Service Requirements
				1	2	3	4	
A	4	4/sec	4	5	10	5	5	Normal scheduling
B	2	2/sec	2	5	5	-	-	Normal scheduling
C	1	1/sec	2	7	5	-	-	Normal scheduling
D	3	2/sec	4	5	5	5	10	Normal scheduling
E	5	10/sec	2	10	10	-	-	Precisely cyclic scheduling every 50 msec.
F	*	*	2	5	7	-	-	Immediate and complete response on external signal (Assume t - 28 msec.)

Description of Sample Computational Load

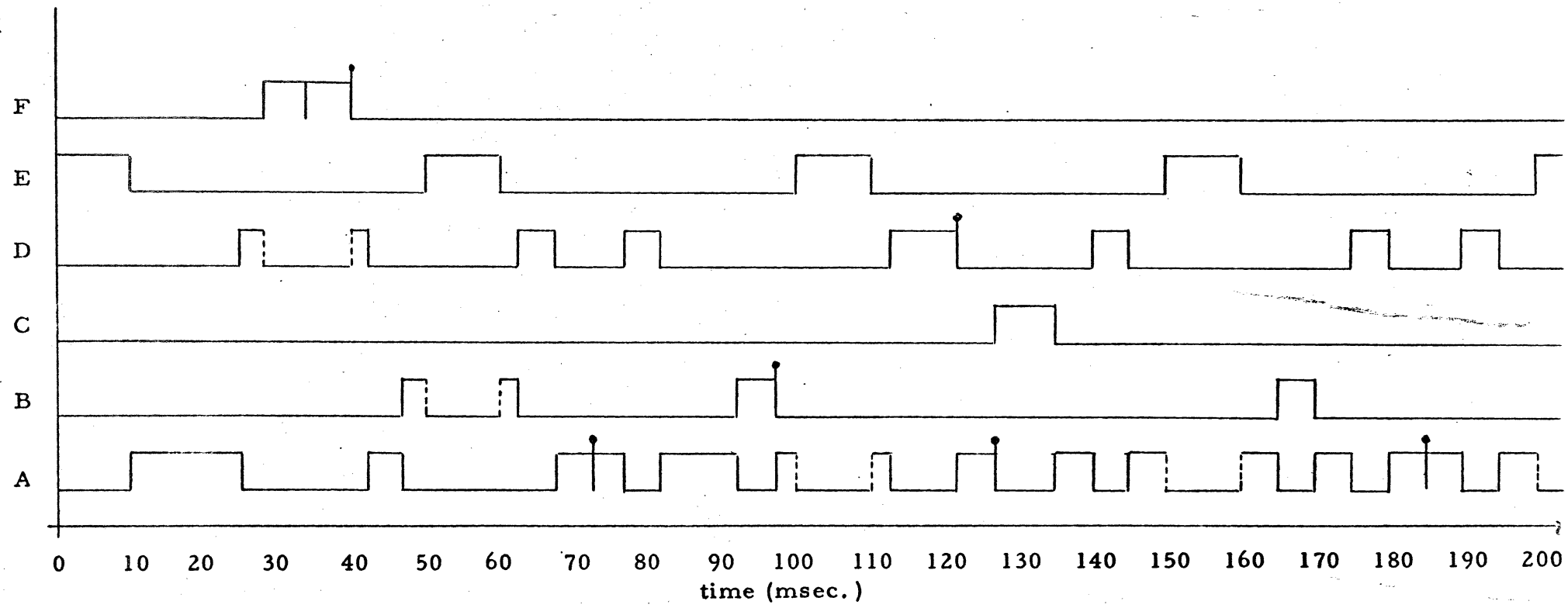


Figure 2-14 - OBES Handling Of Sample Computational Load

- A selection factor C_i is computed within the multiplexing loop. C_i is initialized for all programs as $1/n_i r_i$ and is incremented by $1/n_i r_i$ each time program P_i is selected.
- A selection count k_i of the number of times program P_i has been selected during the current major cycle.
- Program state S_i.
- Selection integer s denoting the program selected at any given time. If s = 0, no program in the loop can or need be selected.

There are ℓ programs P_i ($i = 1, \ell$) in the multiplexing loop at any given time. The method of allocating CPU time is to select, at each stage of the procedure, the program with minimum selection factor (C*); programs whose $k_i \geq r_i$ are not permitted to compete. The procedure is illustrated in Figure 2-13. Depending on what is meant by "nominally" cyclic, it may be necessary to utilize a minor cycle as well as a major cycle to insure that the required r_i executions do not "bunch-up" at the beginning of a major cycle. Also, it is not necessary to perform the initialization of the loop every major cycle. These two features and associated logic for implementation is somewhat dependent on characteristics of the overall application program repertoire and are not illustrated in Figure 2-13.

Programs not entered directly into the multiplexing procedure either: (a) disrupt temporarily the regular multiplexing to perform an urgent task, or (b) are serviced at the end of major cycles when no program in the loop can be scheduled, normally because $k_i = r_i$ for all $i = 1, \ell$. Such non-multiplexed programs may be classified as follows:

- Precisely-cyclic programs which must be entered at fixed and precise time intervals. Where possible, requirements for precisely-timed entry should be avoided in coding applications programs.
- Immediate-response programs which must be entered immediately; usually on occurrance of a signal from a device external to the CPU.
- Background tasks whose completions are not urgent and can be performed as time permits.

All programs in the regular multiplexing loop are logically divided into segments, each of which will be completed within maximum segment time (e.g. 10 msec) or will be interrupted at the end of that time. Programs not handled by the multiplexing procedure may or may not be segmented but may be interrupted, if their requirements permit, to maintain regularity in repeating the multiplexing procedure or to update the status of programs within the multiplexing loop. Precise or immediate entry or completion requirements will be satisfied for such programs whether or not they are segmented. A good general rule is to segment all programs when possible. Failure to do so will not result in failure to meet all program requirements, but may result in unnecessary OBES overhead to process stacked interrupts, process service queues, and perform more complete save/restore than would otherwise be necessary.

The requirements of programs may be described by one of the "service classes" listed below:

- Immediate-response.
- Precisely-cyclic.
- Nominally-cyclic.
- Priority.
- Background.

The status of any program in the system may be described by one of the "program states" listed below:

- Selected - now in control of the CPU.
- Active - actively competing for CPU time.
- No longer competing for CPU time during this major cycle because its $k_i = r_i$.
- Waiting - competing for CPU time but must first await the completion of an event (such as an I/O operation).
- Inactive - in main storage but deactivated and not presently requiring service.
- Activated but its servicing has not begun because it is awaiting main-storage allocation or loading service.

- Dormant - available on the AMU, from which it may be retrieved and serviced on demand (usually) by a signal to switch phase or mode.

As is the case throughout the description of OBES, such categorization as "service class" and "state" described above is mainly for explanatory purposes. One does not necessarily find one-for-one correspondence between categories and flow-charts or coding lists. Logic and computation is coded in a manner which meets system requirements, considers special contingencies, and allows for overlap of categories at the lowest possible cost in CPU time and main storage space.

2.3.4.4 I/O Supervision

I/O supervision routines handle all I/O requests issued by application programs and process all I/O interrupts. Assuming a channel type I/O, an I/O request is a request to execute a channel program; I/O interrupts result usually from the execution of channel programs. I/O supervision routines perform the following functions:

- I/O Request Processing

  Determine channel and device availability.

  Enqueue requests that cannot be immediately honored.

  Initiate I/O operations when requests can be honored.

- I/O Interrupt Analysis

  Maintain tables describing device status and usage.

  Resolve conflicting demands for device usage.

  Test channel, control unit, and device status, and, when a complete path to a device is available, select the queue entry to be processed next.

  Error analysis and recovery.

When the I/O request processor determines that a complete path from (to) the required device is available, it initiates the channel program. If an error occurs during the initiation of a channel program, control is given to the I/O interrupt analyzer to determine the reason for the error and initiate error recovery. If (in the case of control unit shared by more than one device) the control unit is busy, the channel program is not accepted and the request remains queued. Figure 2-15 is the functional flow diagram of the I/O Request Processor.

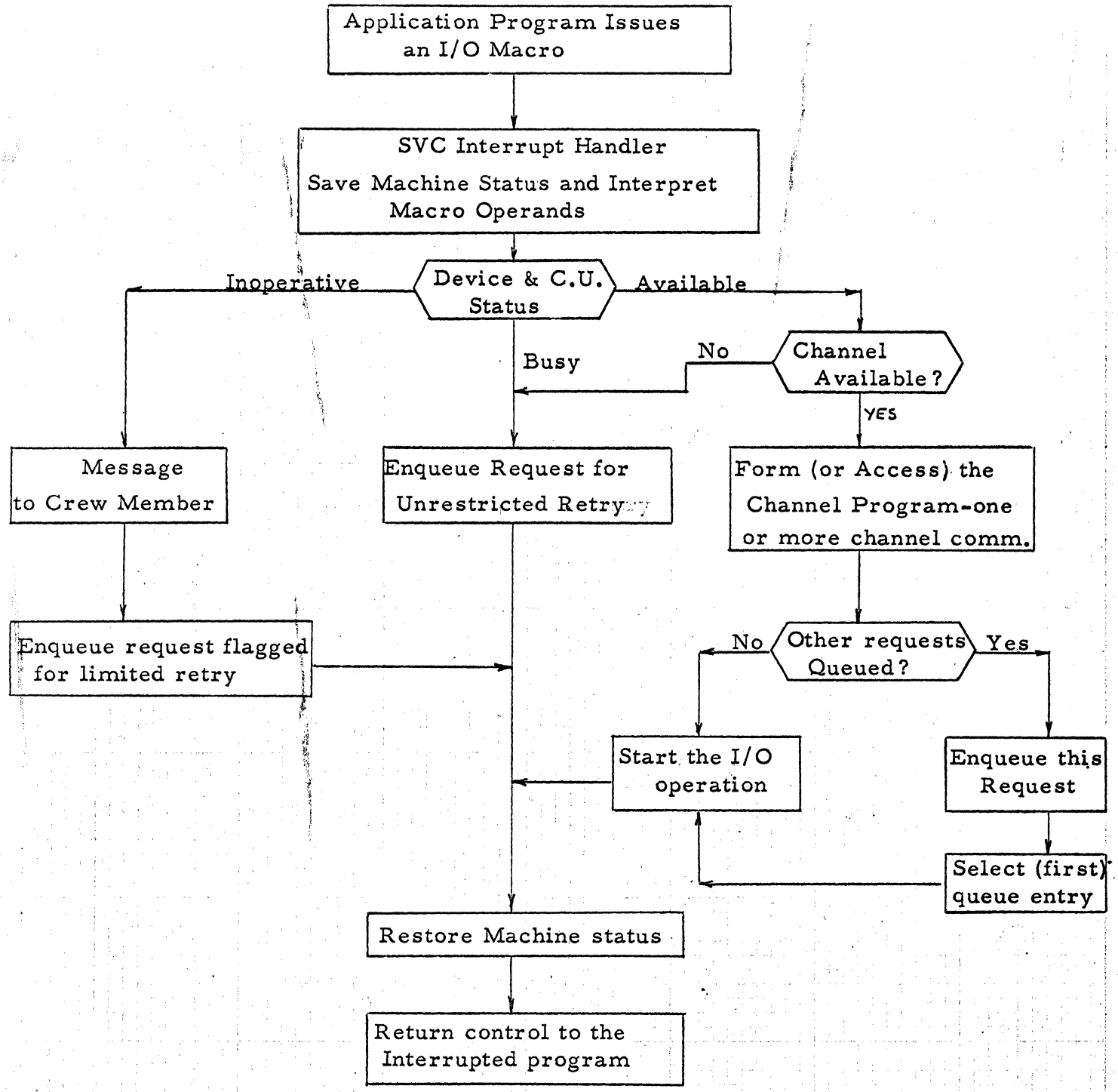


Figure 2-15

I/O Request Processing

If the I/O interrupt analyzer determines that an I/O interrupt was caused by a device attention signal, the appropriate attention routine is entered. If an interrupt is due to an attempt to start an I/O operation when the control unit or device is busy, the I/O request remains queued. If an I/O interrupt is due to normal completion of an I/O operation, normal processing is permitted to continue. Error conditions cause an error routine to be executed and, if the error is uncorrectable, it is so indicated to the requesting program.

Figure 2-16 describes the analysis of I/O interrupts. Supervision of I/O resources is heavily machine-dependent; I/O interface/control unit/device specifications are inextricably bound to I/O software control systems. The OBES I/O supervisor routines maintain tables describing the status and control requirements of each device, respond to user requests for I/O service, determine the availability of a complete device-to-control unit-to-channel path, select and initiate the operation, and monitor the status of the operation. Present-day I/O capability permits I/O processing to progress concurrently with CPU processing, concurrent I/O operations on the same channel, and capability for programmer-controlled interruption of channel activity. Such capability is invaluable in meeting the QRGT objectives of flexibility and generality over a wide range of mission requirements and vehicle devices. Hardware provision of status information for the channel, control units, and devices permit the OBES I/O supervisor to rapidly and efficiently allocate and control the system's I/O resources.

2.3.4.5 Main Storage Supervision

Main storage supervision routines are entered when it is necessary to activate a program not residing in main storage. Deactivated programs are tagged "inactive" but remain in storage until necessary to free space they occupy in order to reallocate it to a program required to be activated. The status of main storage utilization is described in a system table. Main storage cleanup is performed to make scattered free areas contiguous only when a sufficiently-large contiguous area necessary to meet an activation requirement is not available. Programs with varying (expanding or contracting) requirements for data storage can issue requests for the allocation or freeing of assignable data storage areas.

The program loader and program relocator are separate routines included in the category of main storage supervision. Program loading requires also program relocation since all programs on the AMU are referenced to location zero and relocatable. Relocation is also required as a separate function in cases where already-loaded programs must be moved to form a single contiguous area from several scattered free areas.

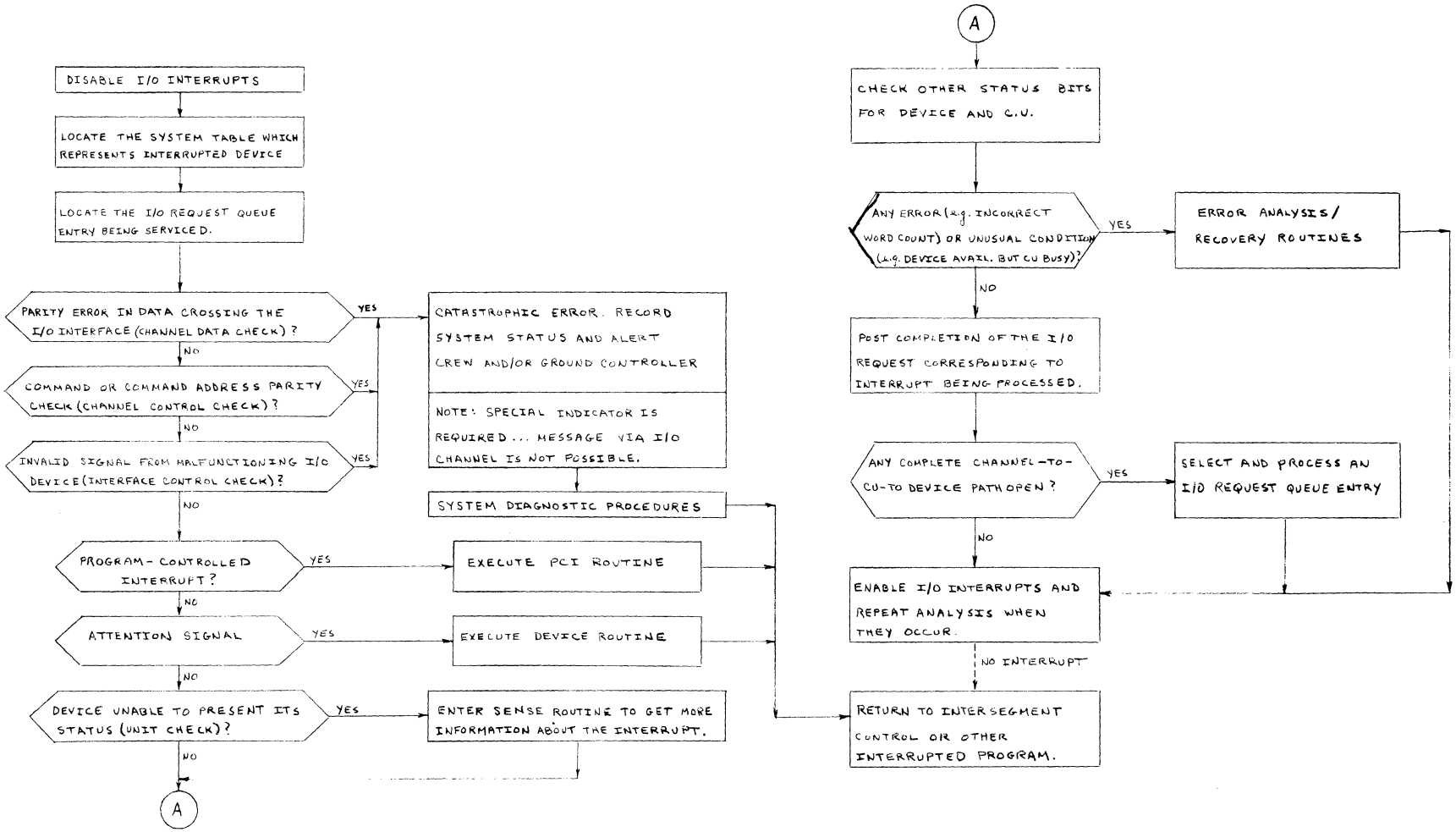


Figure 2-16
I/O INTERRUPT ANALYSIS

If an activation requirement or request is awaiting loading or storage allocation services, it is not entered into the multiplexing loop until these services have been provided. Deactivated programs are not removed from main storage unless there is no other means of accommodating a program to be activated. Requests or requirements for program activation or deactivation are normally considered only between major cycles (e.g. 1 sec) of the multiplexing procedure. This does not apply to programs which are required for immediate response to external signals - these programs and others with precise timing requirements, must be in main storage during all periods within which their invokation is possible. The functional flow diagram of the main storage supervisor is shown in Figure 2-17.

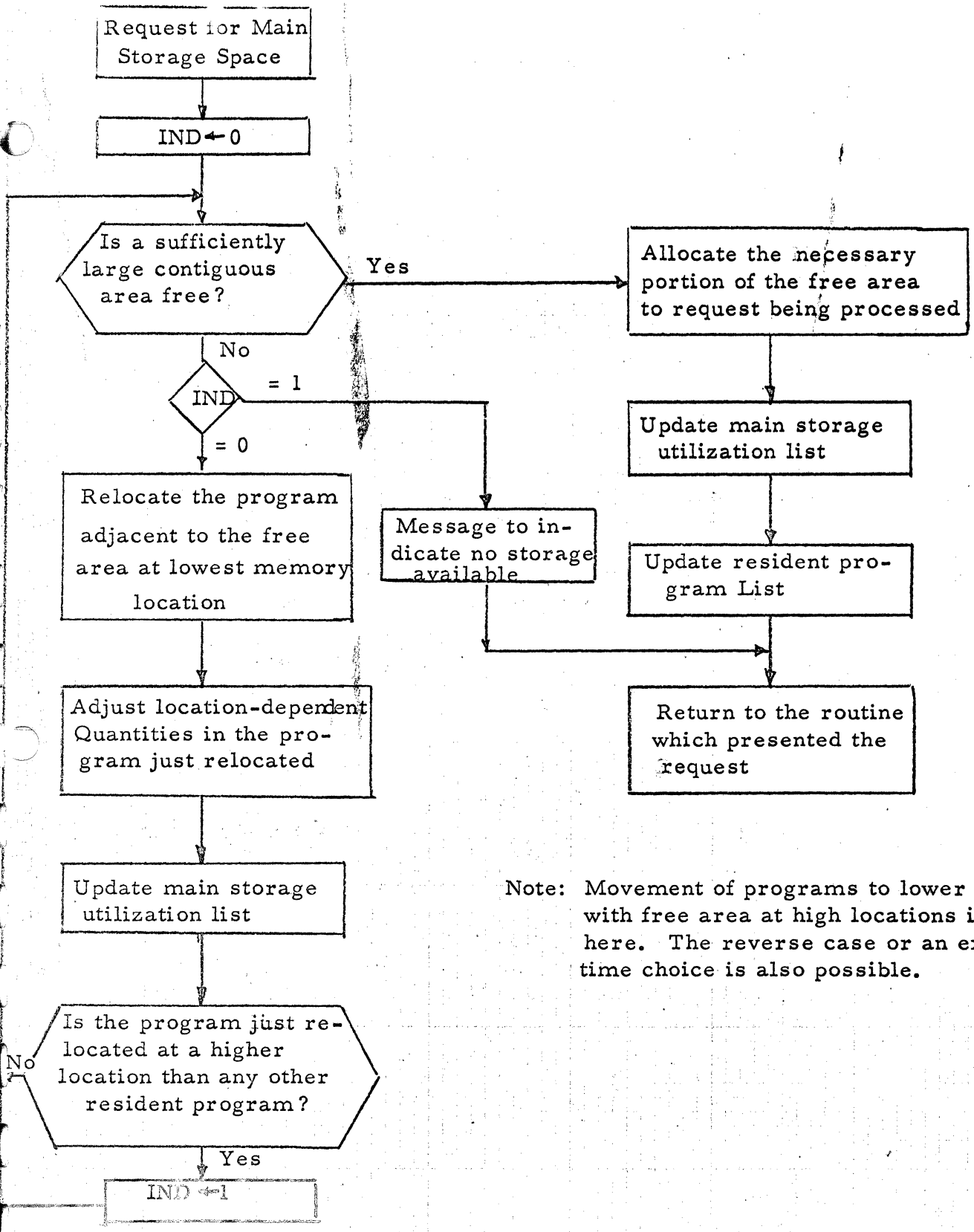
2.3.4.6 Utility Services

Utility services provided consist of re-enterable routines available to applications programs to perform computations such as trigonometric functions, exponentials, logarithms, data conversion, etc. The exact repertoire of such functions provided depends on demands imposed by the mathematical formulation of all mission functions and has not yet been finally determined. Provision of these as system services reduces redundant coding effort, contributes to uniformity of results, and conserves main storage by making it unnecessary to store duplicate copies for separate users.

2.3.5 SIMULATION TOOLS

The third major software system that must be provided for QRGT is that required to perform debug, test and validation of the flight programs. This software is commonly referred to as "simulation tools" and could be categorized as "soft" simulation and "hard" simulation. By "soft" simulation is meant that all of the components of the system (computer, sensors and controllers) are implemented in software. By "hard" simulation we mean that these components are implemented in hardware and operated in a multiprocessor facility.

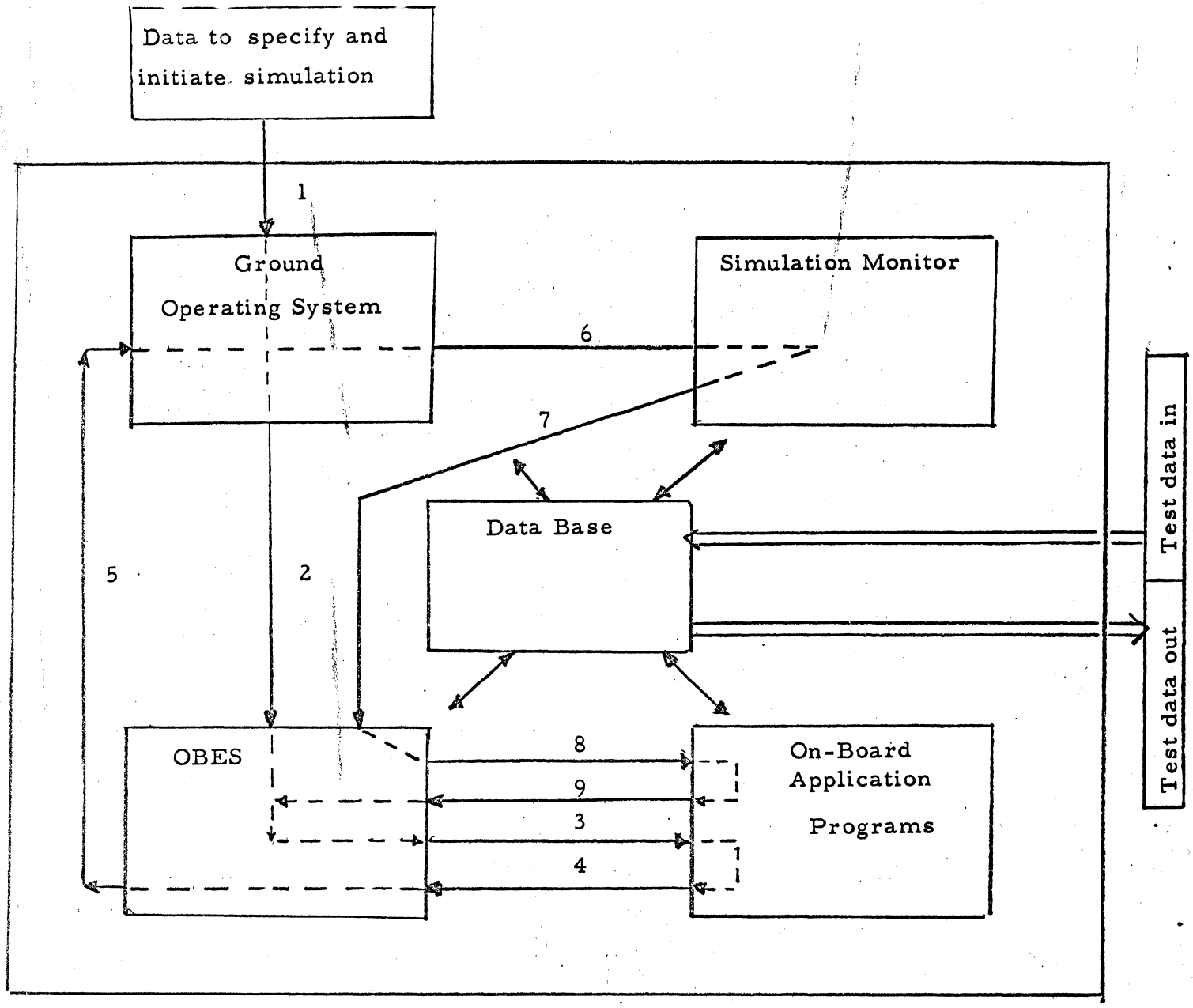
Minimal (soft) simulation is illustrated in Figure 2-18 and a more extensive multiprocessor facility is illustrated in Figure 2-19. The minimal one is capable of basic validation of the on-board software contained in each SVL. The multiprocessor facility performs the validation in a more thorough and realistic manner using an actual on-board computer system connected and operating with a larger ground-based computer. The minimal facility provides basic interface validation of SVL's; the multiprocessor would, in addition, serve as a training/demonstration tool and provide capability for final systems test and qualification of the on-board computer software/hardware system.



Note: Movement of programs to lower locations with free area at high locations is assumed here. The reverse case or an execution - time choice is also possible.

Figure 2 - 17

Main Storage Allocation and Cleanup



1 Initiate Simulation.

2 Initiate On-Board System.

3 Schedule an Applications Program and transfer control to it.

4 Applications Program issues a request for an OBES service.

5 In attempting to provide the service, OBES uses a privileged operation which causes a program interrupt and control to GOS.

6 GOS handles the interrupt by transferring to the simulation monitor which processes the interrupt as it would be on-board.

7 Simulation monitor returns control to OBES to routine processing the request.

8 After OBES completes the request processing, it returns control to the requestor applications program.

9 At segment end, control is returned to OBES.

Figure 2-18

A Minimal Simulation Facility with Sample Flow

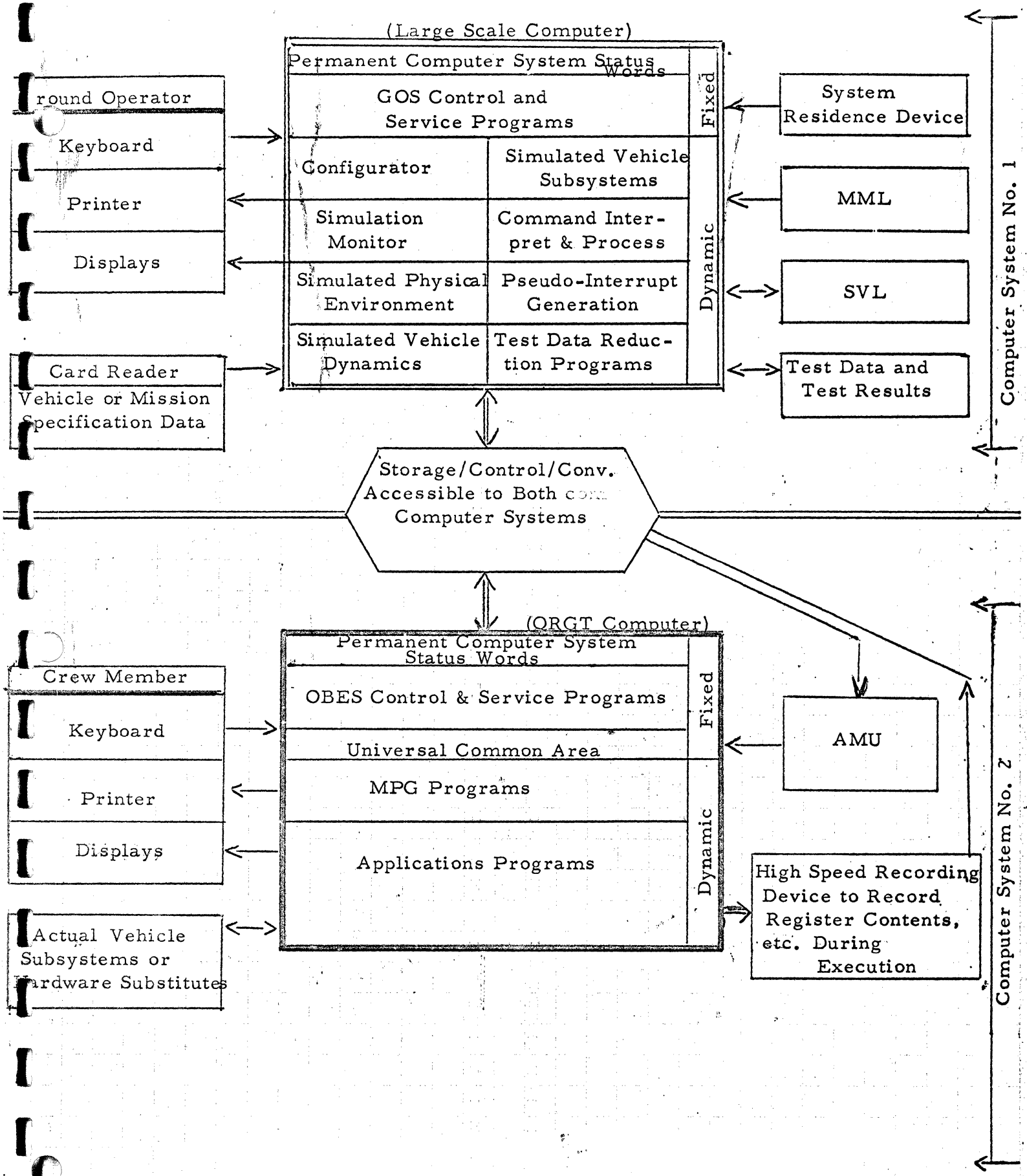


Figure 2-19 A Multiprocessor Simulation Facility

2.3.5.1 Minimal Simulation

The computer used in the minimal system employs four major categories of software:

- A control program nearly identical to the GOS control program.
- A simulation monitor to mediate between the control program and the on-board programs.
- The on-board executive system (OBES).
- The on-board applications programs being validated.

All on-board software including the OBES functions as a problem program. When OBES issues an instruction not permitted in the problem state, (e.g., an I/O instruction which is only executable by a supervisor program in the ground system) the "violation" causes transfer of control to the control program. The control program, however, has been modified to permit such "violations" by transferring control to the simulation monitor for handling as it would be handled on board. The simulation monitor then returns control to OBES via the control program. An example of such interaction is given in Figure 2-18

With compatibility between ground and on-board computers, the simulation monitor serves mainly as a switching center between simulation facility software and the on-board software. In this case, on-board instructions can be executed by the ground computer on a one-for-one basis. Without compatibility, the simulation monitor would have to provide interpretive routines for fetching and simulating on-board instructions.

2.3.5.2 A Multiprocessor Simulation Facility

The multiprocessor facility (Figure 2-19) consists of a ground computer and an on-board computer operating together as a single system through an interface unit consisting of storage, control, and conversion equipment. Its capability includes:

- Generation and maintenance of the MML.
- Extraction of MML subsets to form SVL's.
- Loading of SVL's on an actual AMU device.

- Initial loading from AMU to an on-board main storage unit and initiation of the on-board system.

- Communication of information and control data from the on-board computer to vehicle devices; in some cases actual devices, in others software-simulated devices.

- Execution of on-board executive and applications programs in the same manner that they would be executed on-board.

- Provision for test data input, data reduction, and test result output.

- High-speed recording of on-board computer status at small time intervals (e.g. after each instruction execution) for detailed post-test analysis of the flight program.

- Insert/display capability which functionally provides all such capability available to crew members plus additional facilities for monitoring and controlling simulation runs.

- Use of computer test equipment for the purpose of testing, at the probing level, selected on-board computer test points.

On-board computer programs are executed in an environment which accurately represent their operational environment. Inputs representing the space physical environment, including gravity and atmospheric forces are provided by a simulated physical environment operating in the ground computer. Vehicle control data is input to and processed by the simulated vehicle dynamics program operating in the ground computer system; this same program outputs vehicle attitude reference data to the application programs. Information and control data (e.g. telemetry, discretes, display, autopilot signals) transmitted between application programs and vehicle devices are transmitted from (to) actual devices attached to the on-board computer system where possible; software-simulated devices are substituted where actual devices cannot be provided.

The multiprocessor facility described provides all services necessary to prepare, debug, test, integrate, and validate QRGT software. Its most important purpose is to provide for final systems test and qualification of an integrated software (SVL) hardware system.

2.3.6 COMPUTER SIZING

In order to determine the organization of the flight computer program it is necessary to compile the storage and timing requirements for the functions to be implemented on the flight computer. For the present study a decision had to be made as to the level at which program sizing was to be evaluated. As indicated in the past, the overall sizing study depends heavily on previous evaluations updated to the present time; that is, within this phase of the QRGT study, the only functions being thoroughly studied and defined are the guidance equations, mission planner and the on-board executive system (not to the extent of the guidance equations). The outputs of these studies will provide flow diagrams from which good estimates can be arrived at by trial programming. But in order to obtain an estimate of the realistic computational load it is necessary to consider such things as orbital navigation, digital autopilot, displays, calibration, alignment, etc. To obtain the estimates for these items, maximum use is being made of previous computer sizing efforts on other studies. The principal information comes from studies like SSGS and MOL and is supplemented with information from Gemini, Titan and Saturn where necessary. It follows that the functional level used to evaluate sizing should be compatible with the breakdown of these programs and the level that appears most compatible is the mode level as defined in Appendix A of this report.

It should be noted, however, that it is not necessarily true that the flight program will be structured at this level. Since the present study is to delineate the computer requirements, the method of evaluation must be compatible with the use of a computer with or without auxiliary storage capabilities. This being the case the final computer organization cannot be determined until a more comprehensive computational load is determined. Even then two organizational aims exist. If the computer is to be used without auxiliary memory, the most efficient use of core presupposes that common modules be utilized to the fullest extent feasible, whereas if auxiliary memory is to be used, tradeoffs exist concerning the modular size, the access time and the storage needed for bookkeeping. Because of the absence of a strict limitation on size of an auxiliary memory, a large amount of program redundancy could be desired for this method. For example, it could be conceived that the computer could be loaded by phase rather than by function mode. If this were true there could be considerable redundancy among the modules. Recommendations as to module size will be presented as the study advances but since these recommendations are based on computational load and bookkeeping costs (storage/timing) they will be continually revised as required.

The trial programming is being done using the IBM System/4 Pi Computer (Model EP) using floating point arithmetic. One program, the Gemini Re-Entry, is being written in both fixed and floating point arithmetic. This is being done in order to calculate the ratio between a fixed and floating point program. The ratio, for both storage and major cycle times, has been determined by evaluating the portions of the re-entry program that are primarily computational. It is recognized that this ratio must be used with care. Each of the programs to be sized must be investigated from the standpoint of the amount of logic versus the amount of computation and the ratio then applied only to the computational portions.

The Gemini-Re-Entry program was selected to determine the ratio due to the large amount of information available about it. IBM developed the program for Gemini missions and detail math flows with exact scaling are readily available. The ratio for the fixed to floating point arithmetic has not been applied to the other functions for the present report but will be utilized after further analysis to substantiate an accurate one. Preliminary analysis shows the factor to be approximately 5% for storage and 15% for timing (i. e. fixed point requires 5% more storage and 15% more CPU time).

The sizing effort, to the present time, has investigated available sources for program sizing that are not covered by Task II. Coding has been completed for the Gemini Re-Entry (floating and fixed point) and a Digital Autopilot proposed for the Titan III C vehicle by IBM during a computer competition. Also, the Martin T III digital autopilot equations are being trial programmed, but the results are not yet complete. Many of the functions that IBM analyzed for MOL have been updated and used for sizing purposes. The summary of the storage and timing estimates to date is presented in "QRGT Computer Sizing Estimates and Mission Description" a classified document to be issued under separate cover as a supplement to this report.

The computer requirements analysis will continue with concentration on the guidance equations, mission planning and mathematical subroutines. Effort will also continue on compiling available information on other QRGT functions in order to get a more comprehensive feel for the size and speed of computer requirements. The major functions identified such as alignment routines, orbital navigation, I/O interface routines will be trial programmed to get an accurate estimate. Other areas such as displays, self test, etc. will not be trial programmed during Phase I due to absence of detailed logic/math flows during this phase.

2.4 DISPLAYS (TASK 4)

Several of the missions considered under the QRGT Study involve manned spacecraft. The over all objectives of Task 4 are: (1) to identify the tasks to be performed by the man and (2) establish the information that must be displayed to aid the man. Manned tasks related to guidance and targeting fall in the following categories:

- Mission Planning
- System management in normal operation
- Malfunction detection and corrective action
- Performance of a portion of the navigation guidance or control function to replace failed elements of the system.

Table 4-1 presents a preliminary list of tasks generated to date, together with their basic information requirements. In the next quarter, these tasks and their information requirements will be defined in greater detail. The consequences for computer requirements will be estimated in cooperation with Task 3.

Table 4.1 Crew Activities and Tasks
Preliminary

Crew Activity	Tasks	Information Required
Mission Planning (Prelaunch)	Insert Mission Specification via manual input device (if not automatic function.	Target ephemeris or latitude and longitude constraints to be followed (or ignored), criteria (such as minimum time or maximum probability of success within time limit), mission objectives.
	Check Mission Specification for accuracy and consistency with preliminary briefing.	Stored Mission Specification
	Update Mission Specification (if not automatic function).	Revised part of Specification
	Insert constraints to be followed	Special physical constraints. Other constraints such as minimum time or number of station passes between maneuvers, lighting conditions, safety margins, range safety, launch time.
	Insert cues to guide in generation of mission plan.	
	Examine computer-generated mission plan(s).	Launch window, range safety data, constraints violated, time to mission objective, total mission time, maneuvers required, time between maneuvers, time of station passes, lighting conditions at critical times, ΔV requirements, orbit parameters, etc.
	Modify cues or constraints as desired.	
	Modify portions of proposed plans as required.	
	Construct own mission plan if desired.	Details of part of plan already constructed, mission specification, and information for planning next portion.
	Select a plan.	Details of candidate plans.
	Verify validation.	
Range-Safety (Pre-launch)	Insert launch azimuth constraints and any other constraints which differ from launch site normal.	Constraints may be based on existing air and surface traffic conditions and urgency of mission.
	Monitor existing constraints.	
	Check proposed mission plans against constraints.	Planned and permitted flight paths, launch azimuths, impact points, and overflight conditions.
	Revise mission plan to meet constraints or obtain clearance to violate them.	

Table 4.1 Continued

Crew Activity	Task	Information Required
Mission Planning (Orbital)	Verify or insert current mission status.	Present orbit parameters, remaining capability.
	Modify Mission Specification if required.	Alternate Mission Specification.
	Plan mission as in Pre-Launch Phase, but starting with current status.	
Mission Planning (For Deorbit)	Select or modify criteria for trajectory shaping based on current conditions. Criteria may include heat, acceleration, mechanical stress, time to re-enter, time to landing or splashdown, time to rescue.	Heat shield and vehicle integrity, life support system status.
	Select Landing sites.	Possible sites under existing constraints and criteria as a function of times, consequences of selected site and proposed trajectories.
	Select trajectories for further analysis.	Consequences of selected trajectories including total heat, peak heat rate, and allowed values; peak acceleration magnitude, time, range, and cross range.
	Select re-entry plan.	
Trajectory and Performance Evaluation (Ascent)	Monitor for abnormal performance.	Acceleration, attitude, attitude rates, time of staging, etc.
	Compare trajectory flown with nominal and with safety and range safety limits.	Precomputed trajectory limit data.
	Monitor predicted impact points unless included in above.	Impact points and protected areas.
	If trajectory requires abort, determine optimum time.	Same plus degree of urgency.
	Check parameters of achieved orbit.	
	Determine maneuver required (if any) to achieve safe orbit.	Perigee, apogee, time to apogee, and tolerance to which these are known. ΔV required. Consequences with nominal and non-nominal orbit.
	Make Go/No Go decision.	ΔV required for safe orbit, ΔV to complete mission. ΔV remaining. Achievable nominal and worst case orbit parameters.
Trajectory Evaluation (Descent)	Monitor trajctory consequences.	Present and predicted maximum total heat, peak heat rate, acceleration. Time, range, cross range, altitude.
	Pilot vehicle to landing.	Piloting and navigation instruments and displays.
Abort Planning (Ascent)	Monitor abort mode.	Altitude and ascent events.
	Optimize time of abort.	Degree of danger, achievable landing areas, predicted impact area of debris.

Table 4.1 Continued

Crew Activity	Tasks	Information Required
Navigation	Sensor management.	Sensor capabilities and requirements, navigation requirements.
	Select landmark (or star).	Availability and recognition information.
	Operate landmark telescope and check correction for reasonableness.	Telescope pointing and tracking indication and time to go. Magnitude of correction.
Guidance	Monitor performance of maneuver.	Scheduled and actual time, duration, attitude, ΔV, and results.
	Determine need for corrective maneuver.	Above plus navigation information or position update from ground.
	Determine need for replanning mission.	Existing capability and requirements of present plan. Feasibility of present plan.
	Docking	Range, range rate, angles and rates.
	Manual backup.	Indication of need, maneuver schedule and details. Present state of vehicle.

2.5 ERROR ANALYSIS (TASK 5)

Effort during the past quarter consisted primarily of development of error analysis plans for Phase II.

In the next quarter, guidance error vectors (errors in six orbit parameters at thrust termination) for each guidance source considered will be received from Task 2 personnel, and propagated to critical mission points; i.e., points at which mission success is affected by spacecraft position and velocity accuracy. Mission performance will be compared with mission success requirements. Results of this analysis will be used to establish which guidance errors are acceptably small and which guidance errors need to be reduced by modification of the guidance equations.

Section 3

APPENDICES

APPENDIX A

MODULARIZATION OF MISSIONS

This Appendix includes a definition of a classification hierarchy and indicates the breakdown of QRGT missions within this hierarchy. This breakdown and classification is necessary for Multi-Mission Planning for various, but related, reasons:

- o Determination of the extent of commonality across missions as a function of the classification level.
- o Selection of the "optimal" program module size for the On-board and Off-board (GOS) systems.
- o Determination of the On-board flight program organization as a function of on-board memory capacity and allocation (Core, Auxiliary Storage).
- o Identification of the modules, from those available on-board, necessary to construct a flight program which implements the Mission Planning specification.
- o Determination of an on-board validation procedure and its impact on reaction time and computer loading.

The presently defined hierarchy is: Mission Set, Mission Class, Mission, Phase, Function, Mode, and Element.

Mission Set is the collection of all QRGT missions. The mission set presently being considered consists of the missions described in TOR-669(6730-07)-2 "Mission Analysis and Requirements".

Mission Class is a high-level classification of missions according to trajectory objectives (orbit injection only, orbital rendezvous/intercept, or other) and according to whether the mission is manned or unmanned. Of the six classes defined, one (manned missions with orbit injection only) is empty since no such missions are in the Mission Set.

Mission, as used in this discussion, denotes a generic mission defined by its objective, e.g., orbit injection for an application satellite. Eight different generic missions have been considered in detail. For convenience, they will be referred to as missions 0 through 7, following the terminology of TOR-669(6730-07)-2, the report referenced above. A specific mission is defined by a quantitative

or semi-quantitative mission plan which describes the planned sequence of events (trajectory changes, etc.) from the beginning to end, including contingent events such as abort.

Phase denotes a time-based segment of a mission, with boundaries defined by some change of operating conditions, such as the ignition or cut-off of engines, etc. Major mission phases are: Pre-launch, Ascent, Abort, Parking Orbit and Transfer, Terminal Rendezvous, Orbital Operations, and Re-Entry and Landing. These phases are briefly described in the next section. Some phases do not occur in certain missions; others may occur more than once.

Function denotes the operations necessary to implement the various phases. These functions, which are identifiable with major vehicle systems, are: Navigation, Guidance, Control, Mission Planning, System Readiness, Digital Communication, Display, Acquisition and Tracking, and Mission Support. The individual functions are described below.

Mode denotes the specific application of the function during a particular phase of a given mission and which is compatible with the relevant interfacing systems. For example, the Inertial Navigation Mode is the application of the Navigation Function during the Ascent Phase of all missions.

Element is that part of a mode program which can be identified with a particular implementation technique, subsystem or mission requirement. As an example, the Inertial Navigation Mode may have three elements: Data Processing, Gravity Model, and Integration Algorithm.

This classification concept is illustrated in Figure A1. Figure A2 presents the Phases for all QRGT Missions. The preliminary definitions of Functions and Modes are presented in Figures A3 and A4 respectively.

MISSION PHASES

The mission phases, indicated on Figure A2, are described briefly with a listing of the operations performed in each phase. These operations are generally what will be referred to later as Function Modes.

Pre-Launch Phase - The operations during pre-launch will be fairly standard for all missions; however, the accuracy requirements and time availability varies for specific missions or configurations. The following operations are performed in this phase:

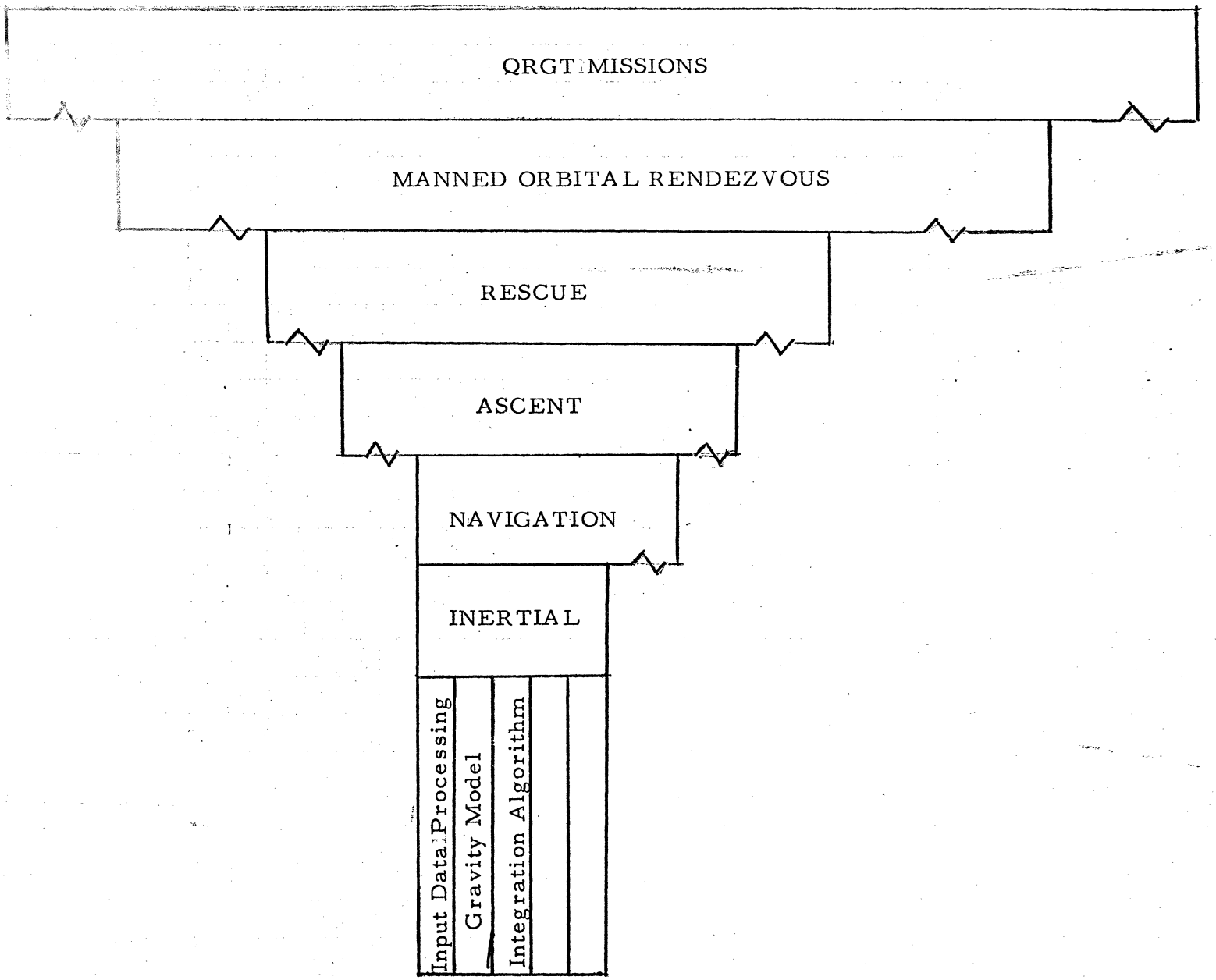


Figure A1 QRGT Mission Classification Hierarchy

Class		Mission Number TOR-669(6730-07)-2 / Phase	Pre-Launch	Ascent	Abort	Parking Orbit and Transfer	Intercept and Terminal Rendezvous	Orbital Operation	Re-Entry and Landing
Unmanned	Orbit Injection	0	x	x	x	x		x	
		4	x	x	x	x		x	
	Orbit Rendezvous	0	x	x	x	x	x	x	
		3	x	x	x	x	x	x	x
		1	x	x	x		x		
	Other	2	x	x	x	x		x	x
Manned	Orbit Injection	Presently no QRGT Mission defined in this Class							
	Orbit Rendezvous	5	x	x	x	x	x	x	x
		7	x	x	x	x	x	x	x
	Other	6	x	x	x	x		x	x

Figure A2 Mission Phases

Phase	Function / Mission Number (TOR-669(6730-07)-2)	Unmanned: Orbit Injection (0 4)	Unmanned: Orbit Rendezvous (0 3 1)	Unmanned: Other (2)	Manned: Orbit Injection (No missions presently defined)	Manned: Orbit Rendezvous (5 7)	Manned: Other (6)
Pre-Launch	Mission Planning	x x	x x x	x		x x	x
	System Readiness	x x	x x x	x		x x	x
	Digital Communications	x x	x x x	x		x x	x
	Display					x x	x
Ascent	Navigation	x x	x x x	x		x x	x
	Guidance	x x	x x x	x		x x	x
	Control	x x	x x x	x		x x	x
	Digital Communications	x x	x x x	x		x x	x
	Display					x x	x
	Mission Planning (Mission Safety)					x x	x
Abort	Navigation					x x	x
	Guidance					x x	x
	Control					x x	x
	Mission Planning					x x	x
	System Readiness					x x	x
	Digital Communications	x x	x x	x		x x	x
	Display					x x	x
Parking Orbit and Transfer	Navigation	x x	x x x	x		x x	x
	Guidance	x x	x x x	x		x x	x
	Control	x x	x x x	x		x x	x
	Mission Planning	x x	x x x	x		x x	x
	System Readiness	x x	x x x	x		x x	x
	Digital Communications	x x	x x x	x		x x	x
	Display					x x	x
	Mission Support	x					
Terminal Rendezvous	Navigation		x x			x x	
	Guidance		x x x			x x	
	Control		x x x			x x	
	System Readiness		x x x			x x	
	Digital Communications		x x x			x x	
	Display					x x	
	Acquisition and Tracking		x x			x x	
Orbital Operations	Navigation			x		x x	x
	Control	x x	x x	x		x x	x
	Mission Planning		x	x		x x	x
	System Readiness	x x	x	x		x x	x
	Digital Communications	x x	x x	x		x x	x
	Display					x x	x
	Acquisition and Tracking		x			x	
	Mission Support	x x	x x	x		x	x
Re-Entry and Landing	Navigation		x	x		x x	x
	Guidance		x	x		x x	x
	Control		x	x		x x	x
	Mission Planning		x	x		x x	x
	System Readiness		x	x		x x	x
	Digital Communications		x	x		x x	x
	Display					x x	x

Figure A3 Mission Functions by Phase

Figure A4 Function Modes by Phase (Sheet 1 of 2)

Function	Mode \ Phase	Pre-Launch	Ascent	Abort	Parking Orbit and Transfer	Terminal Rendezvous	Orbital Operations	Re-Entry & Landing
Navigation	Inertial		x	x	x	x		
	Augmented Inertial							x
	Autonomous				x	x	x	
	Approach and Touch Down (Landing)							x
Guidance	Atmospheric Ascent		x					
	Vacuum Ascent		x					
	Abort		x	x	x			
	Orbit Transfer			x	x			
	Terminal (Rendezvous, Station Keeping)					x		
	Re-Entry			x				x
	Landing			x				x
	Synergistic Plane Change							x
Control	Atmospheric Booster Flight Control		x					
	Vacuum Booster Flight Control		x		x	x		
	Vehicle Attitude		x	x	x	x	x	x
	Sensor Pointing				x	x	x	x
	Re-Entry			x				x
Mission Planning	Preliminary Trajectory Generation	x		x	x	x		x
	Trajectory Optimization	x		x	x	x		x
	Trajectory Selection	x		x	x	x		x
	Targeting	x		x	x	x		x
	Flight Program Identification	x			x			x
	Mission Safety	x	x	x	x			x
	Validation	x			x	x		x
	Plan Update	x	x			x		
System Readiness	Activation	x		x	x	x	x	
	Computer Self-Check	x	x		x	x	x	x
	System/Subsystem Checkout	x		x	x	x	x	
	Calibration	x			x		x	
	Alignment	x			x		x	
	Initialization and Verification	x		x	x	x	x	
	Status Monitoring	x	x	x	x	x	x	x
Digital Comm.	Up-Link		x	x	x		x	x
	Down-Link		x	x	x		x	x
	Manual (Manned Missions)	x	x	x	x	x	x	x
Display (Manned Missions)	System/Subsystem Status	x	x	x	x	x	x	x
	System/Subsystem Modes	x	x	x	x	x	x	x
	Vehicle Attitude		x	x	x	x	x	x
	Trajectory Parameters		x	x	x	x	x	x
	Rendezvous Data					x		
	Re-Entry Footprint			x	x			x
	Insert/Display	x	x	x	x	x	x	x
	Mission Plan Parameters	x		x	x			x

Function	Acquisition and Tracking				Mission Support				
Mode / Phase	Sensor Activation and Initialization	Acquisition	Tracking		Experiment Activation, Initialization and Control	Sensor Activation and Command	Experiment Data Pod Ejection	Payload Preparation and Deployment	
Pre-Launch									
Ascent									
Abort									
Parking Orbit and Transfer									
Terminal Rendezvous	x	x	x						
Orbital Operations	x	x	x		x	x	x	x	
Re-Entry and Landing									

Figure A4 Function Modes by Phase (Sheet 2 of 2)

Mission Planning and Validation
Complete System checkout
Subsystem Calibration and Alignment
Operational Readiness verification
Maintain Digital Communications
Display of checkout and system status data
(Manned missions only)

Figure A5 represents a typical functional flow for this phase.

Ascent Phase - The ascent phase begins at launch and ends at injection into the first orbit which may be a transfer orbit, a parking orbit or the actual working orbit.

The following operations are performed on all missions:

Short-term navigation
Boost Guidance in various modes of operation
Booster Flight Control
Vehicle attitude maneuvers
Digital Communications
Flight conditions monitoring

A representative functional flow for the Ascent Phase is presented in Figure A6.

Abort Phase - The Abort Phase is run in parallel with the Ascent Phase and its objective is crew safety on manned mission and range safety on all missions. Abort of unmanned missions is usually a Range Safety function but an on-board Abort Phase might be responsible for detection of erratic flight and/or malfunctions as an aid to Range Safety operations. Some of the operations necessary in this Phase are:

System Status Monitoring
Ascent Trajectory Monitoring
Digital Communications
Payload Separation and Control
Abort Planning
Abort Guidance
Spacecraft Attitude Control
Re-Entry and Landing
} Manned Missions

Figure A7 represents a functional flow of this Phase for manned missions and is based on the Apollo Abort Mode definitions. [1]

[1] Henderson, M. M., et. al., "Spacecraft Operational Abort and Alternate Mission Studies for AS-204A, Vol. 1 - Abort Studies," MSC Internal Note No. 66-FM-113, October 28, 1966.

Parking Orbit and Transfer Phase - During some missions the vehicle may be inactive in a parking orbit and certain in-flight operations are required to ready the system for operation. This phase is also entered when any orbit transfer maneuver is required (orbit keeping, gross rendezvous, plane changes, de-boost, etc.). The operations performed are as follows:

Activate system and subsystem
Verify total system operation
Establish attitude reference
Establish navigation references using autonomous navigation or ground updates.
Perform autonomous navigation
Perform attitude control and maneuvering in orbit
Perform maneuver planning
Perform orbit transfer guidance and control
Maintain digital communications
Present operator display (manned missions)

This phase is presented in Figure A8.

Terminal Rendezvous Phase - The Terminal Rendezvous phase is unique to certain missions; however, some of the following standard operations shall be performed:

a. Establish attitude reference
b. Perform short-range navigation
c. Accept navigation updates when available
d. Perform attitude control
e. Point acquisition and tracking sensors
f. Perform terminal guidance and control
g. Acquire and track target
h. Activate docking system

These operations are represented on the functional flow Figure A9.

Orbital Operations Phase - The functions performed during Orbital Operations vary more greatly between missions than in any other phase. In many cases these functions are highly mission and hardware dependent. Listed below are some representative operations that might be performed in this phase.

Activate rendezvous system
Activate docking system
Activate and checkout auxiliary equipment
Establish attitude reference
Perform autonomous navigation
Accept ground updates when available
Ready shuttle for re-entry
Maintain digital communications

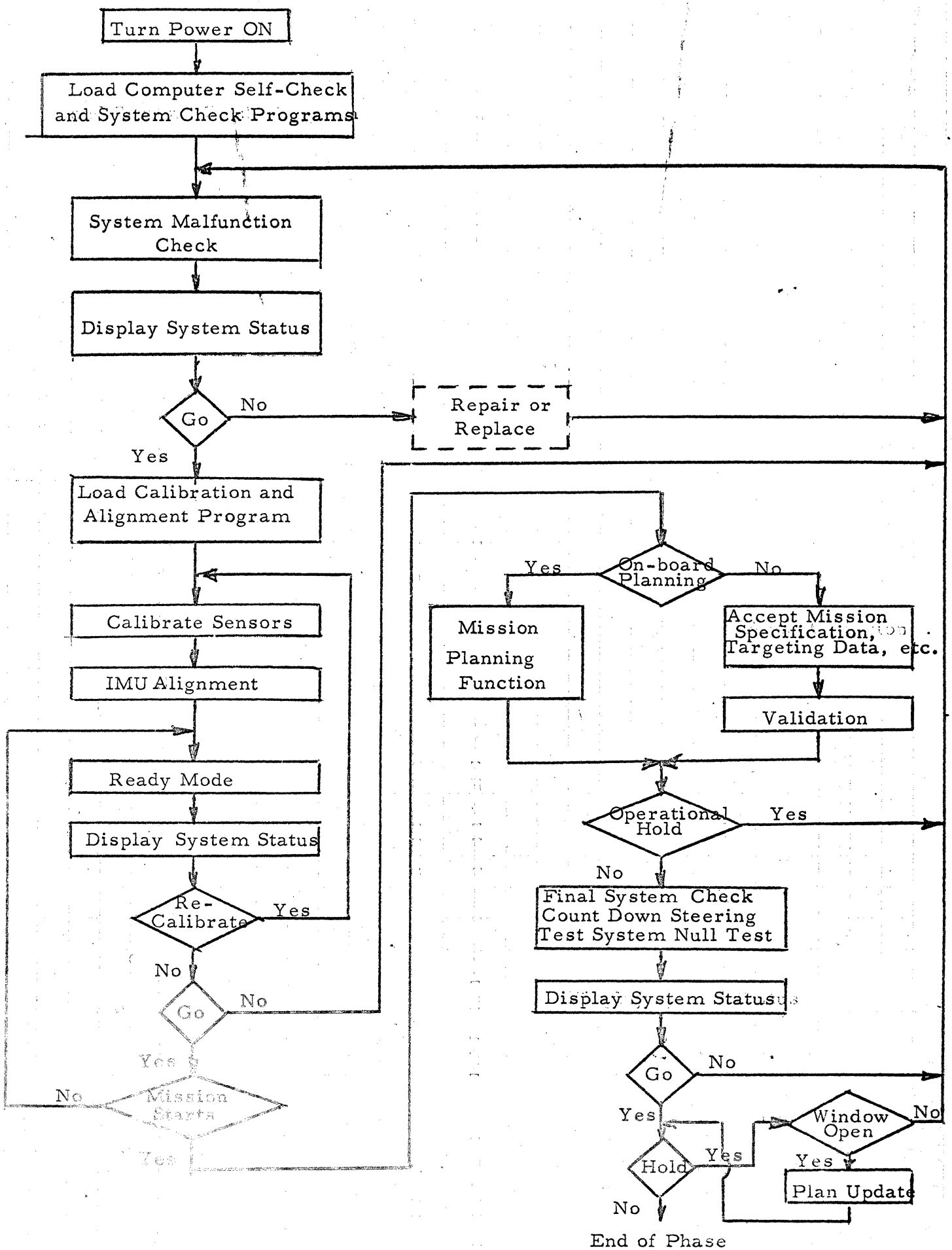


Figure A5 Pre-Launch Phase Functional Flow (all Missions)

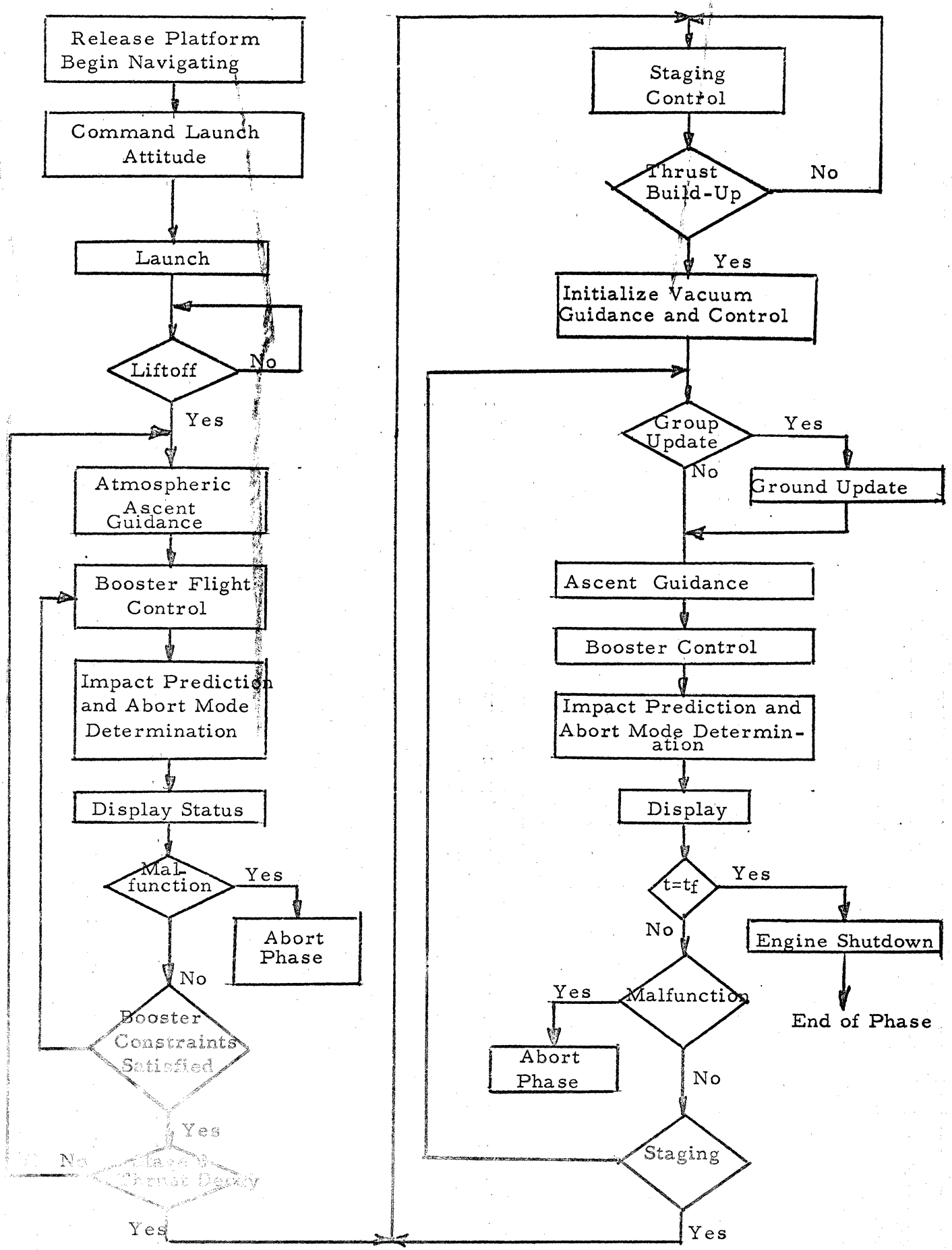


Figure A6 Ascent Phase Functional Flow (All Missions)

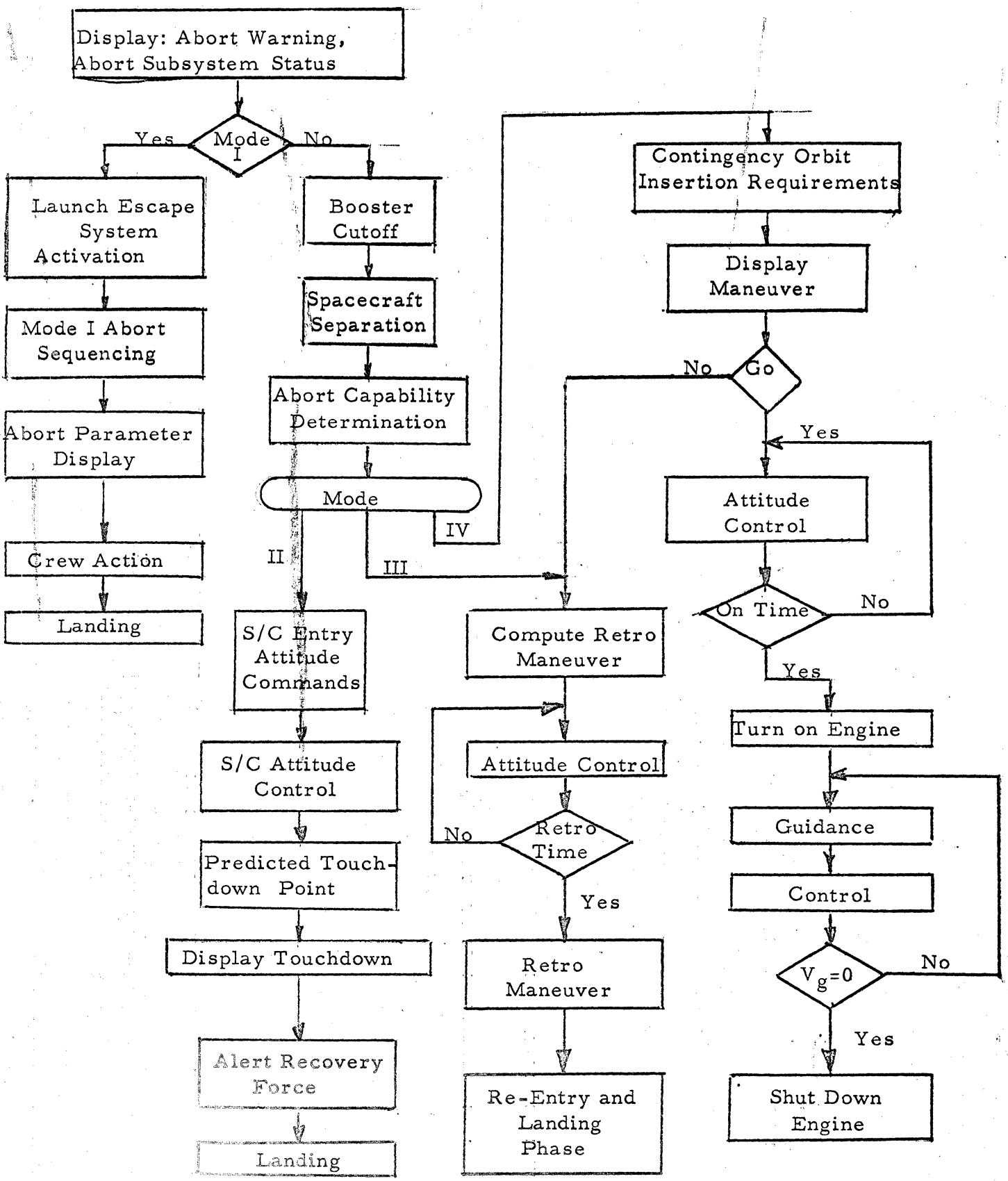


Figure A7 Abort Phase Functional Flow (Manned Missions)

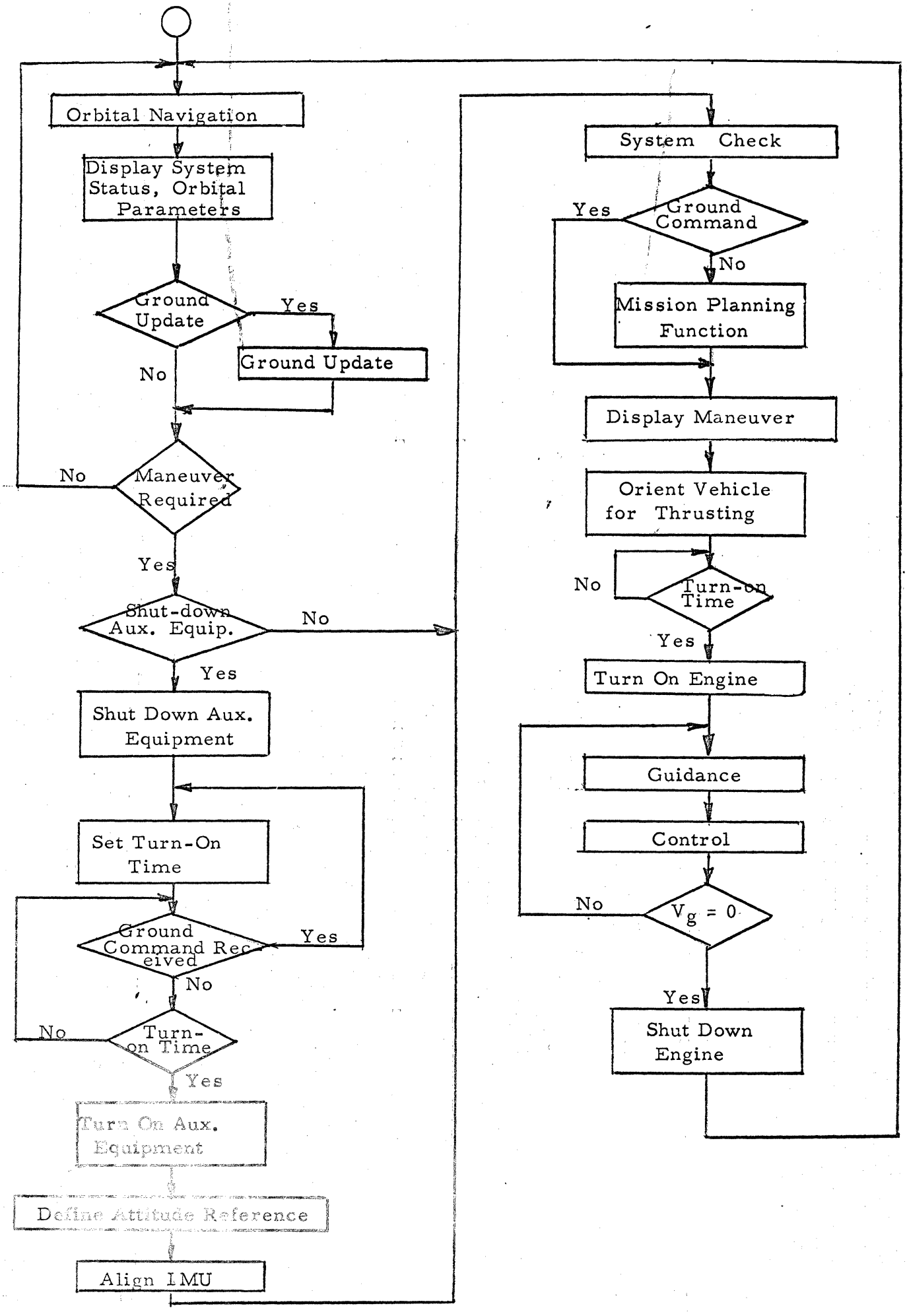


Figure A8 Parking Orbit and Transfer Phase Functional Flow

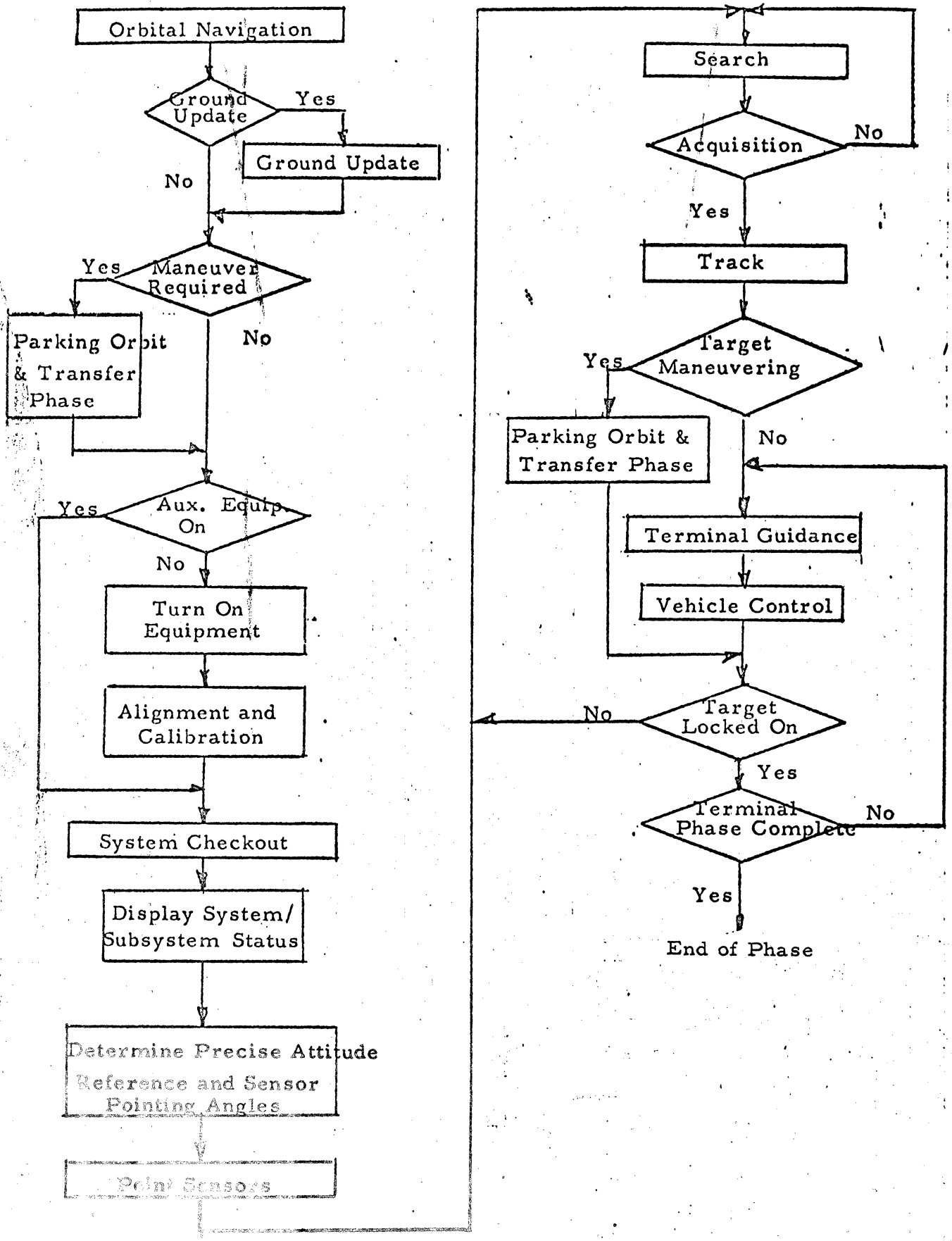


Figure A 9 Terminal Rendezvous Phase Functional Flow

Operate display systems (manned missions)
Activate, initialize, and control experiments

Figure A10 is a generalized function flow of this phase.

Re-Entry and Landing Phase - This phase includes ballistic and low L/D re-entry, and medium L/D re-entry and landing. The preceeding de-boost maneuver is a special case of the Parking Orbit and Transfer Phase. This phase includes:

Landing site selection
Determination of de-boost requirement
Re-entry prediction
Footprint computations
Capsule Ejection Conditions
Navigation
Maintain attitude control
Re-Entry guidance and control, when applicable
Display control
Accept ground updates

Figure A11 represents this phase for ballistic and maneuverable vehicle re-entries.

MISSION FUNCTIONS

To satisfy the requirements of the various QRGT missions requires the capability of performing the major functions of Navigation, Guidance, Control, Mission Planning, System Readiness, Digital Communications, Display Acquisition and Tracking, and Mission Support. These functions, indicated in Figure A3, are described along with preliminary definition of their operating Modes (see Figure A4).

Navigation - The Navigation system shall be capable of performing both long-term and short-term navigation completely independent of any ground-based tracking system but capable of accepting ground updates when available. The navigation capability shall be provided during all mission phases except pre-launch although it may be inoperative during certain coasting periods. Short-term navigation is performed by solving the equations of motion with inputs from the GIMU (Gimbaled Inertial Measuring Unit) and, possibly, other sources (such as radar altimeter). Long-term navigation (autonomous) shall be performed by statistical filtering of navigation sensor data.

The Navigation Modes are:

Inertial
Augmented Inertial
Autonomous
Approach and Touchdown (landing)

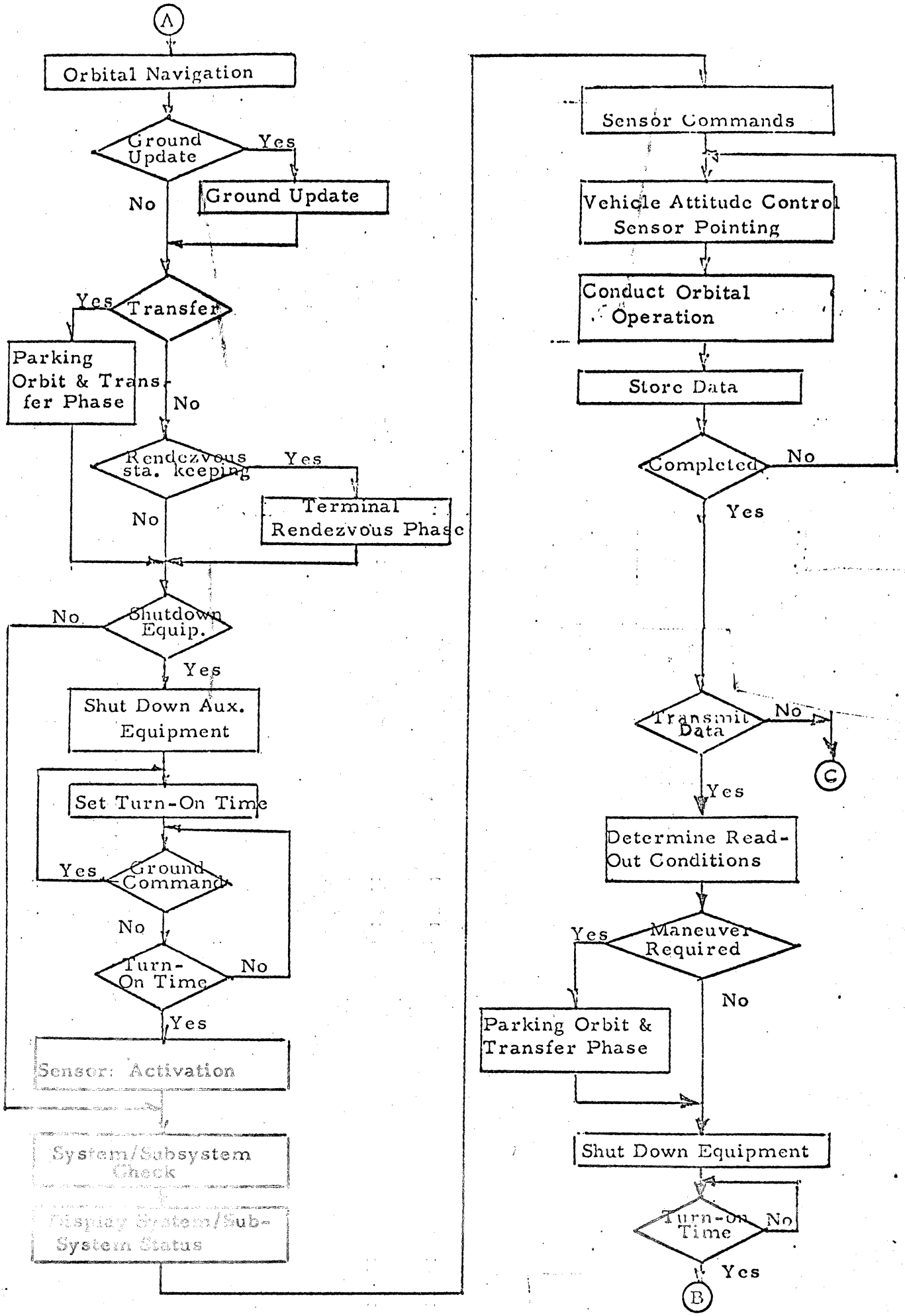


Figure A10- Orbital Operations Functional Flow (Sheet 1 of 2)

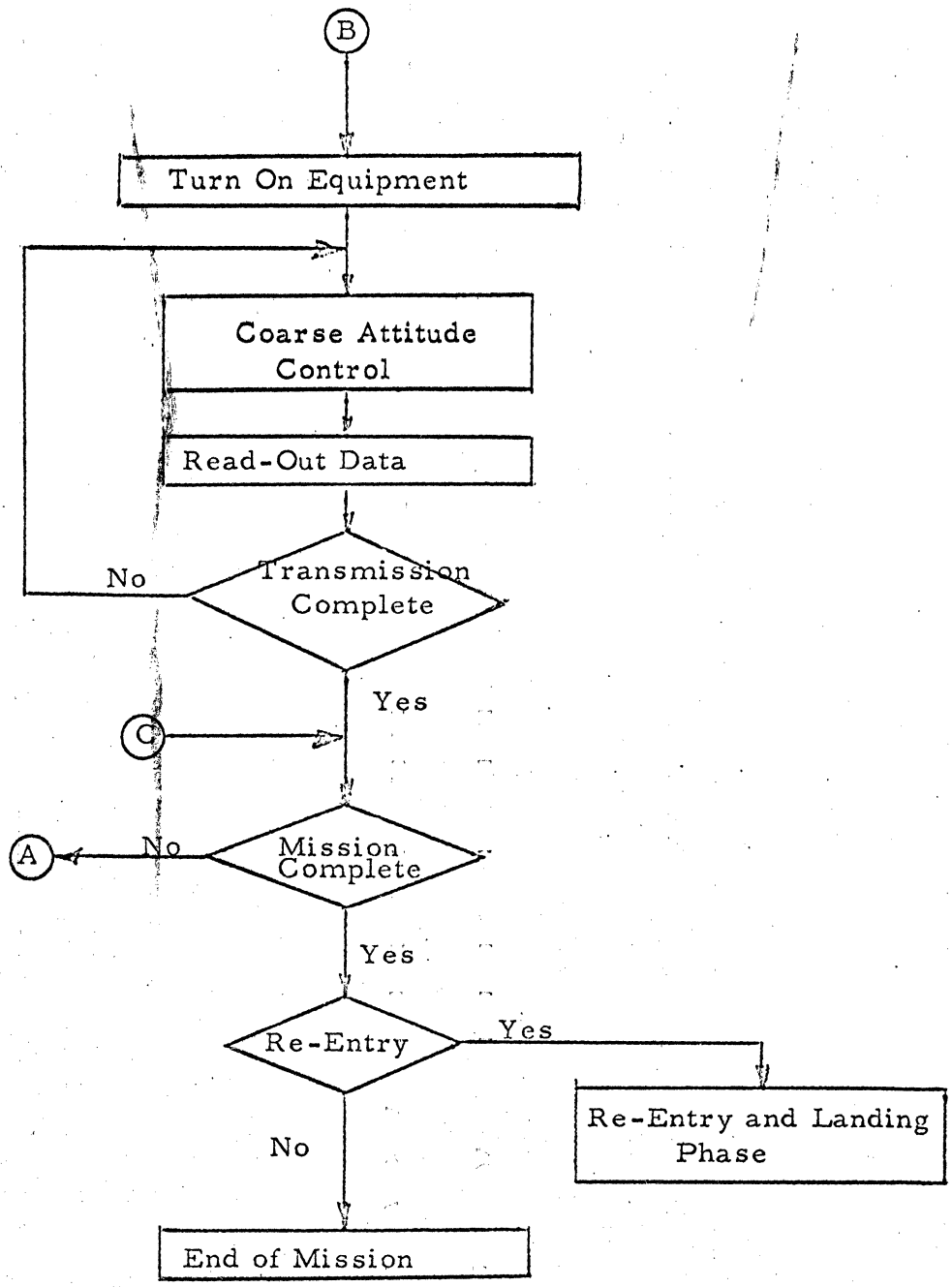


Figure A10- Orbital Operations Functional Flow (Sheet 2 of 2)

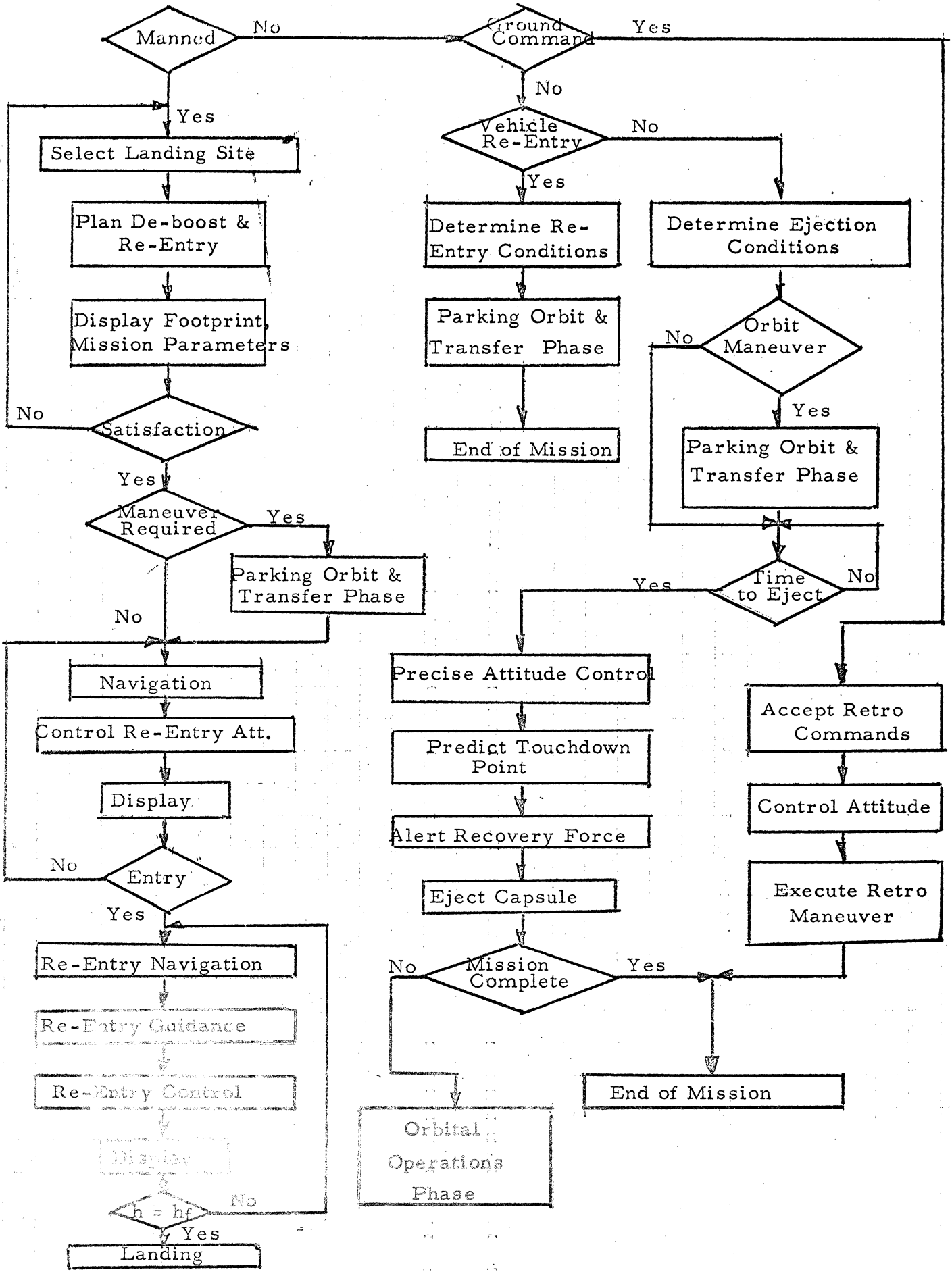


Figure A11 - Re-Entry and Landing Phase Functional Flow

Guidance - The Guidance function supplies commands for the control of vehicle attitude and acceleration during all powered phases and re-entry. The guidance computations shall be independent of ground communication but not exclude it. The inputs to the Guidance Function will be from the Navigation, Mission Planning, and Acquisition and Tracking Function. The Guidance Modes are:

Atmospheric Ascent
Vacuum Ascent
Abort
Orbit Transfer
Terminal (final phase of Rendezvous)
Re-Entry
Landing

Control - The Control function involves trajectory control during powered phases and re-entry, vehicle altitude control during all mission phases except Pre-Launch and vehicle sensor control as required. The Booster Flight Control should be capable of controlling all stages of several booster configurations with various payloads. The Control Modes are:

Atmospheric Booster Flight Control
Vacuum Booster Flight Control
Vehicle Attitude
Sensor Attitude
Re-Entry

Mission Planning - The Mission Planning Function generates, with minimal input, a validated flight program and targeting data based on optimized mission trajectory which satisfies vehicle and mission constraints, safety margins, and mission objectives. This function must also have the capability for re-planning, after launch, the next phase of the remainder of the mission. The Modes of this function are:

Preliminary Trajectory Generation
Trajectory Optimization
Trajectory Selection
Targeting
Flight Program Identification
Mission Safety
Validation
Plan Update

This function and its operating modes is more completely described in Appendix B.

System Readiness - The System Readiness Function is operative during both the Pre-Launch and the [illegible]ht phases and includes, but is not limited to, the following:

Activation
Computer Self-Check
System/Subsystem Checkout
Calibration
Alignment
Initialization and Verification
Status Monitoring

Digital Communications - This function involves: receiving, decoding and transferring digital commands transmitted from ground command stations and formatting, encoding and transferring PCM telemetry data during various mission phases. In addition, manual digital communications will be provided in all manned missions for manual insert of data. The Modes of Operation are:

Uplink
Downlink
Manual

Display - For all manned missions, a Display Function capability will be required. This function, described in Task IV, might include, but is not limited to, the following Modes:

System/Subsystem Status
System/Subsystem Modes
Vehicle Attitude
Trajectory Parameters
Rendezvous Data
Re-Entry Footprint
Mission Plan Parameters

Acquisition and Tracking - The Acquisition and Tracking Function is required in all missions with a Terminal Rendezvous Phase. The system should be capable of acquiring and tracking both cooperative and non-cooperative targets by measuring relative parameters such as range, range rate, angle and angular rate. This information is used in the Guidance Function for Terminal Rendezvous and Docking or Station Keeping. The operating Modes are:

Sensor Activation and Initialization
Acquisition
Tracking

Mission Support - The Mission Support Function is performed during Orbital Operations phases and it provides interface among and computational capability to various systems/subsystems involved in the actual mission objectives. The Modes of operation of this function are highly mission and hardware dependent. Some examples are:

Experiment Activation, Initialization and Control
Sensor Activation and Command
Experiment Data Pod Ejection
Payload Preparation and Deployment

APPENDIX B

ON-BOARD MISSION PLANNING

On-Board Mission Planning represents a new and important addition to the repertoire of space system functions. The role of this function, in the QRGT concept is two-fold:

- Construct, with minimal input, a validated flight program and targeting data based on a self-generated trajectory which satisfies vehicle and mission constraints, safety margins, and mission objectives.

- Provide replanning capability, after launch, for the upcoming phase or for the remainder of the mission in order to incorporate new mission data and/or objectives, plan de-boost and atmospheric maneuvers, and to handle contingencies such as abort.

The input to the On-Board Mission Planning Function is a mission description with related data and the output is a validated flight program with the necessary targeting data (or may be only revised targeting data when used to replan a phase).

The On-Board Mission Planning Function, as presently defined, has eight Modes: Preliminary Trajectory Generation, Trajectory Optimization, Mission Safety, Trajectory Selection, Flight Program Identification, Targeting, Validation, and Plan Update.

The present configuration of the On-Board Mission Planning Function is illustrated by Figure B1 and its various Modes are discussed in the following paragraphs.

MISSION PLANNING MODES

1. Preliminary Trajectory Generation

Purpose

Generation of one or more candidate trajectories as starting points for the optimization routine. The trajectory(ies) may or may not be feasible at this point depending upon the capability of the routines involved.

Input

Initial State
Mission Type
Target Parameters
Mission Constraints
Vehicle Configuration/Characteristics

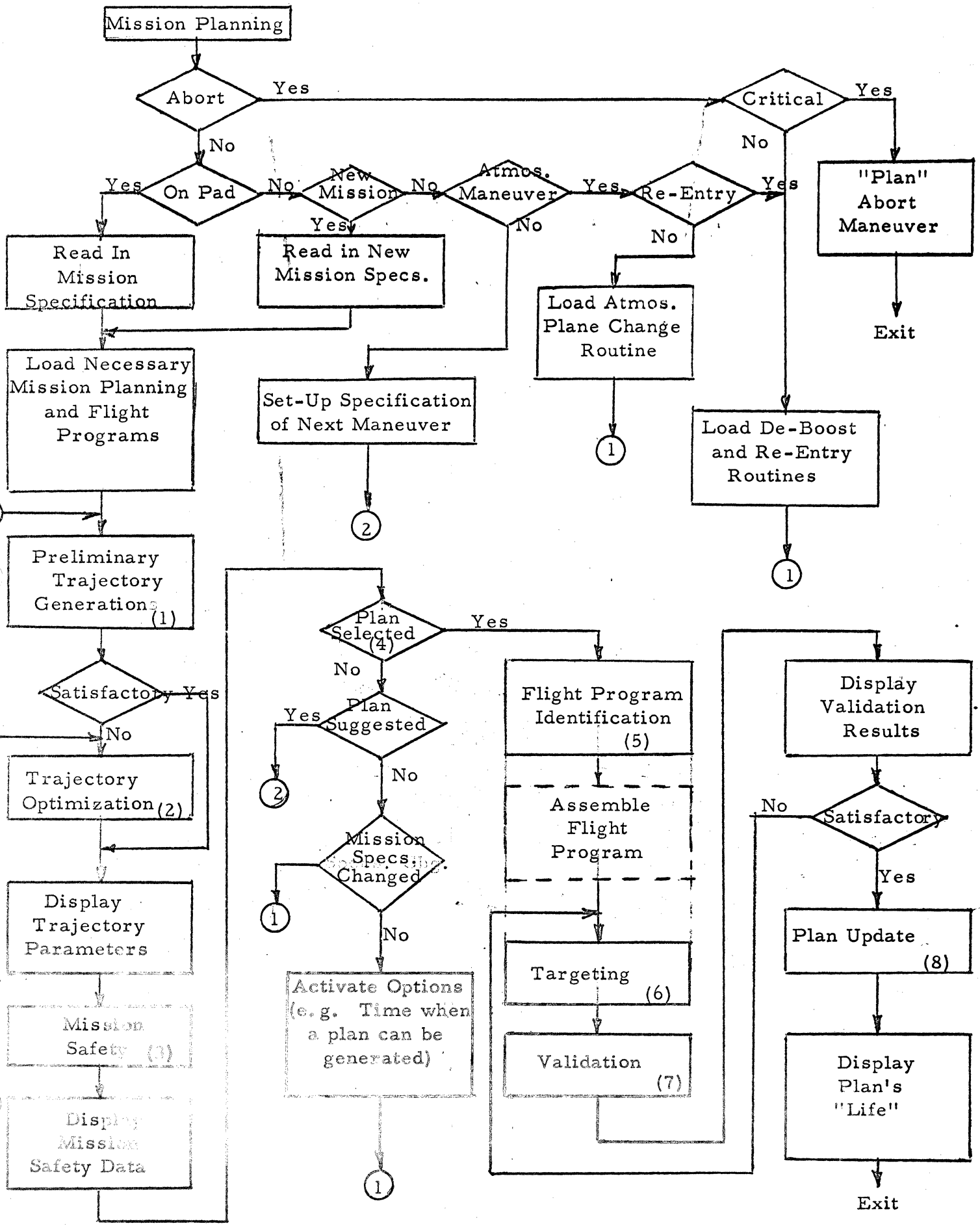


Figure B1 - On-Board Mission Planning Function

Output

Trajectory Parameters - one set per candidate trajectory.

Technique

The preliminary trajectory will be generated from a repertoire of prescriptions which make use of analytical expressions for launch conditions and the atmospheric phase of ascent, explicit guidance equations for the vacuum phase of ascent, impulsive approximations for orbit transfer and orbit plane change, and empirical relations for first-stage burnout conditions, reentry footprints and rendezvous and docking requirements. Various options will be available so that alternate plans can be generated for consideration. The principal elements of this Mode are:

- o Built-in Logic and Routines - this includes the supervisor program and routines particular to mission planning (e.g., launch window determination).

- o Guidance Routines - these are from the flight program library and are used to construct a given mission phase such as ascent, orbit transfer, etc.

- o Prescriptions - the rules for constructing a particular trajectory plan or plans. A mission prescription is dependent upon: the mission class, mission and vehicle constraints, mission objectives, and, possibly, external cues. Its function is to determine options and priorities, initialize guidance routines and select relevant trajectory parameters for the Optimization Mode. The prescriptions will be formulated as a fixed part and a variable part. The fixed portion will consist of a pre-determined table which includes, for each mission class, the necessary information to construct the applicable trajectory and any alternate trajectories.

  The variable portion of the prescription will allow for variations and/or additions to the fixed part due to peculiarities of the particular mission of interest (such as a mission constraint), for re-planning of a mission (due to contingencies, new mission objectives, etc.) or due to external cues.

The final step in the Preliminary Trajectory Generation Mode is to identify the trajectory in terms of relevant parameters, which are then varied in the next Mode, and to assemble the corresponding routines necessary for calculating the pay-off function. The trajectory parameter sets will be predetermined by mission class and mission plan and can be changed as a plan is refined or altered.

Trajectory Optimization

Purpose

Rapid, approximate optimization (local minimum/maximum) of the preliminary mission trajectory to a point which permits: accurate comparison of alternatives, rejection of infeasible trajectories, and generation of targeting parameters.

Input

Trajectory Parameters - one set for each preliminary mission trajectory generated in Mode 1.

Output

Optimized Trajectory Parameter Sets.
Decision Variables - for Mode 3.

Technique

An optimization routine will be employed to systematically vary the mission parameters to minimize (maximize) a mission-dependent "payoff" function while satisfying all mission constraints. This routine is of the "direct search" type. During the optimization mode, trajectories which do not satisfy consrraint might be noted along with the violated constraint(s). This information would be useful in certain cases during Trajectory Selection (See Mode 3) as indicators for possible mission respecification.

Mission Safety

Purpose

Generation of certain Range Safety information during Pre-Launch and Ascent Phases and certain critical trajectory data during the Orbit Transfer and Re-entry Phases.

Input

Vehicle Data
Mission Trajectory Parameters
Mission Safety Routines

Output

Range Safety Data
Critical Trajectory Data

Technique

In the case of the critical trajectory data, this information may be directly or indirectly available from the Trajectory Parameter Set in the form of: transfer trajectory perigee (whether it occurs on the transfer arc or beyond); total mission duration; maximum altitude, etc.

On the other hand, information such as ground coverage during certain mission phases, lighting conditions during terminal maneuvers, etc. will require additional data and routines.

The impact of Range Safety on QRGT is discussed in Section 2.2.6.

Trajectory Selection

Purpose

Selection of the "best" trajectory (if more than one candidate was considered and found to be feasible) including the option of delaying the mission until a more favorable opportunity. For most missions, especially manned ones, this should be an operator (astronaut, control center); decision based on Decision Variables from Mode 2. These variables would include such information as total ΔV required, time required to complete mission and critical mission phases, and safety margins at critical points in the mission.

Input

Decision Variables

Output

Selected Mission Plan, or
Alternate Plan, or
Mission Respecification

Technique

The selection of a mission plan is a trade-off problem involving various factors such as mission type, mission priority and personal preference. The problem is solved by a combination of procedural policies and human judgment.

In the event the plan is unacceptable, Mode 1 can be repeated with contingency options exercised or with operator assistance in the form of an alternate plan or by relaxation of certain mission constraints (which were noted as violated in Mode 2).

Flight Program Identification

Purpose

Configuration of the flight program necessary to carry out the planned mission.

Input

Mission Specification
Vehicle Data
Optimized Trajectory Parameter Set

Output

Flight Program Identifiers
Program Schedules
Discrete Levels

Technique

The mission is synthesized phase by phase from the mission specification and optimized trajectory. Each phase is identified as to its objectives, sequence, and Functions and Modes required. The actual assembly of the flight program is carried out by the on-board Executive routine.

Targeting

Purpose

Determination of targeting parameters associated with the selected trajectory and compatible with the Guidance Function Modes. In some cases these parameters are part of the output of Mode 3, in others they represent effects such as oblate earth and finite duration orbital burns which were not considered in Modes 1 and 2.

Input

Optimized Trajectory Parameter Set - as selected in Mode 3
Vehicle Data
Targeting Routines

Output

Targeting Parameters

Technique

The mission trajectory refinements necessary for targeting purposes can, in most cases, be handled by existing analytical methods which amount to

calculated offsets to compensate for effects neglected in Mode 1. Fast time simulation of the pertinent mission phases might also be employed to generate the targeting data.

Validation

Purpose

Validation of the generated flight program and associated targeting parameters to insure satisfaction of mission objectives and constraints under both nominal and non-nominal flight conditions.

Input

Flight Program
Targeting Parameters
Environment
Vehicle Data
Test Conditions

Output

Selected Data - for validation purposes.

Technique

At one extreme the validation could be completely self-contained with recycling through some or all of the preceeding modes in order to adjust the mission and targeting parameters until all validation criteria are satisfied. This could also include a navigational error analysis of the planned mission. At the other extreme, all validation could be performed externally based on information generated in the previous steps with possible recycling through Mission Planning for adjustment purposes. The more likely solution would be a division of effort with the best possible on-board validation (within time and memory constraints) supplemented with off-board validation procedures. In this concept, a simplified error analysis might be performed and used to adjust an autonomous navigation schedule.

Plan Update

Purpose

Define the "life" of the generated plan (e.g., the launch window during the Pre-Launch Phase) and the information necessary for updating time and/or state dependent parameters that must be varied up to the time of the plan's execution.

Input

Time Dependent Mission Parameters
Maximum Capability Trajectory Parameters

Output

Life Span of Plan
Updating Parameters

Technique

The selected trajectory can be time perturbed (i.e., Launch time for Pre-Launch Planning) to determine maximum allowable timing delays that will result in the mission objectives but with certain mission/vehicle constraints operative. (e.g., $\Delta V_{Total} = \Delta V_{Max.}$)

APPENDIX C

NUMERICAL COMPARISON OF ALTERNATIVE ORBIT TRANSFERS FOR RENDEZVOUS

The Rendezvous-Intercept Routine Option A4, (RIA4) was employed as a prescription for obtaining feasible preliminary two-impulse trajectories for rendezvous starting from a parking orbit. This technique is briefly described in Section 2.2.5. Variable Point Guidance (VPG) is a multi-impulse trajectory planning and guidance concept for near-earth orbit rendezvous developed by RCA (reference 1). Both of these techniques give rise to many possible solutions of the rendezvous problem. Numerical comparison of the least impulse and the least time solutions from each technique will be described in this Appendix.

IBM developed a computer program to simulate the VPG technique. The main features of this program are:

- The spacecraft (interceptor) starts from a circular parking orbit.
- The target (real or fictitious) is in an arbitrary but non-coplanar orbit.
- All burns occur at the line of nodes except the first burn which is positioned to satisfy the next feature.
- The flight path angle at the beginning and end of each burn is unchanged.
- Eight Options are generated for each input. Four options correspond to rendezvous occurring at a given node, the other four to rendezvous occurring at the opposite node. The four options result in Bielliptic-Chase-and-Lob and Full-Orbit-Phasing Chase-and-Lob solutions.
- Bielliptic-Chase denotes a maneuver in which the intermediate burn (on the line of nodes) is inside the target orbit while in Bielliptic-Lob it is outside the target orbit. In the case of Full-Orbit-Phasing, if the period of the phasing ellipse is less than that of the target, the solution is a chase-solution otherwise it is a lob-solution.
- The optimal plane change split subroutine of the program includes an option (input) of considering a pure plane change at the target altitude.

The rendezvous solutions from the Rendezvous Intercept Routine Option A-4 and VPG were obtained for three orbital geometries. The orbits employed in Run #1 were taken from reference 1 and are shown in Figure C1. In Run #2 the inclination of the parking orbit was changed to 0.01 rad. The target orbit was rotated 60° in Run #3 to make the line of apsides perpendicular to the line of nodes. The inclination of the parking orbit in Run #3 was 30°.

The one-dimensional search for the time of the first burn, t_o, which gives a local minimum of the total impulse, was restricted to the interval 0 to 60,000 sec. (The upper limit was chosen so that some rendezvous times would be obtained which were as high as those obtained from VPG.)

Figure C2 is a plot of the total impulse as a function of the true anomaly of the target at epoch (the time when the S/C is at the ascending node). The curves in Figures C2 through C13 should be piecewise continuous functions of the true anomaly of the target at epoch with finite discintinuities. In order to conserve computation time, the argument was incremented in steps of 30°. For each starting condition, the solution which gave the least total impulse is plotted for both the rendezvous intercept routine, Option A-4 (RIA4) and VPG. Figure C3 shows the time of rendezvous for these least impulse solutions. The variation in total impulse between VPG and RIA4 solutions is about ±2.5%. For the most part, the rendezvous via RIA4 occurs sooner than that of VPG.

Figure C4 compares the least time of rendezvous solutions of VPG and RIA4 where only the local minima of the total impulse are chosen as solutions from RIA4. Figure C5 shows the corresponding total impulse. About one-half the time, RIA4 resulted in a faster rendezvous. Sometimes RIA4 gave a quicker rendezvous at a considerable fuel saving.

Figure C6 compares the least impulse solutions of RIA4 and VPG when the relative inclination of the parking orbit is changed to 0.01 radians. Figure C7 shows the corresponding rendezvous times. Figures C8 and C9 compare the least time solutions.

Figures C10 and C11 show the effect of changing the angle between the line of apsides and the line of nodes. In this case RIA4 shows a clear advantage in total impulse bur usually requires more time for completing the rendezvous. Figures C12 and C13 show the least time solutions. RIA4 shows an advantage in time often at the expense of some fuel.

The two-burn technique RIA4 sometimes shows an advantage over the three-burn VPG technique when either the least impulse or least rendezvous time solutions are compared. However, further testing for other orbital geometries is anticipated. Additionally, these techniques will be employed to initialize an optimizer routine. Comparison of the optimized two-burn and three-burn solutions will be made this month.

Ref. 1. "Variable Thrust Guidance Study", SSD-TR-65-100, 15 July 1965.

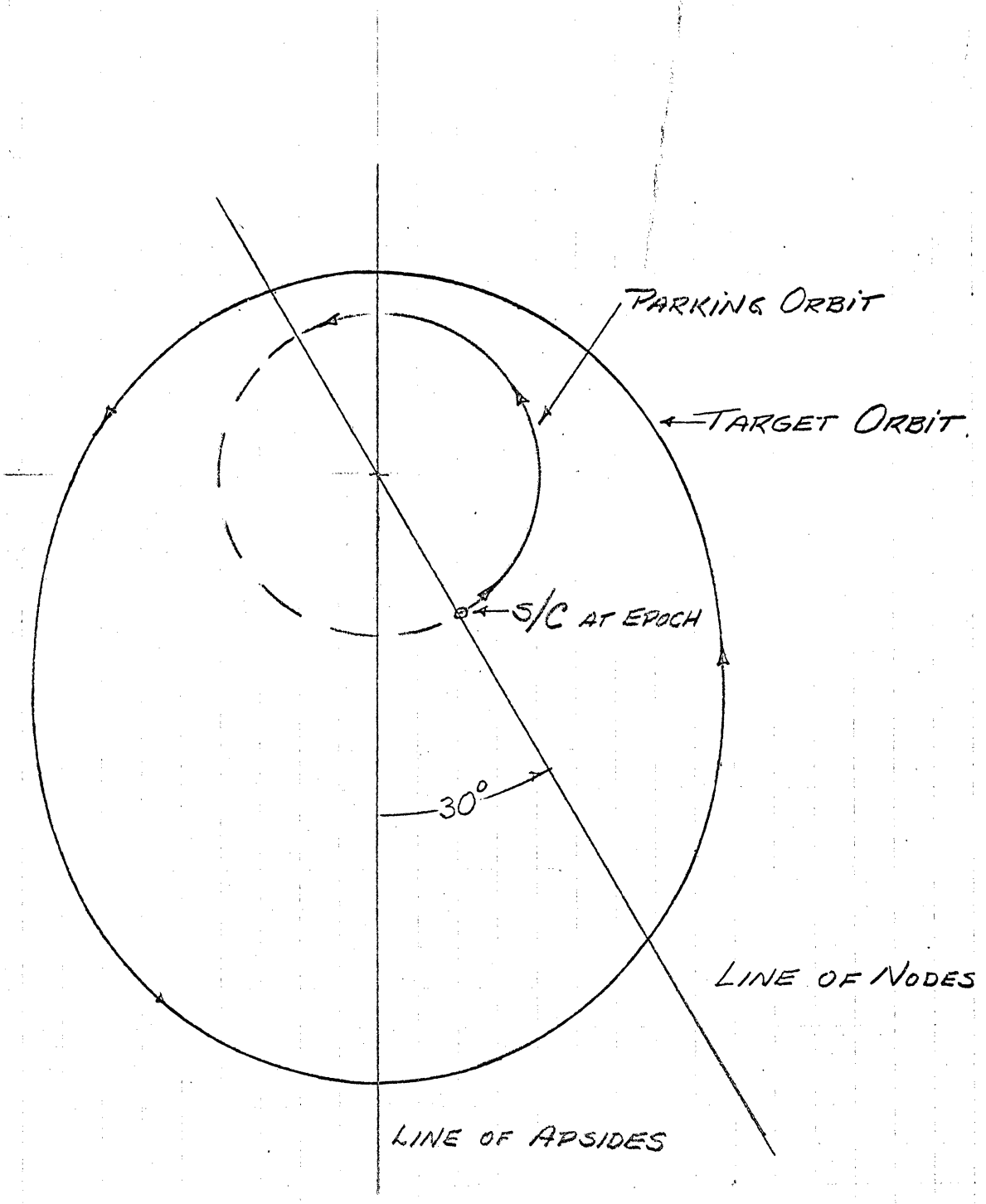
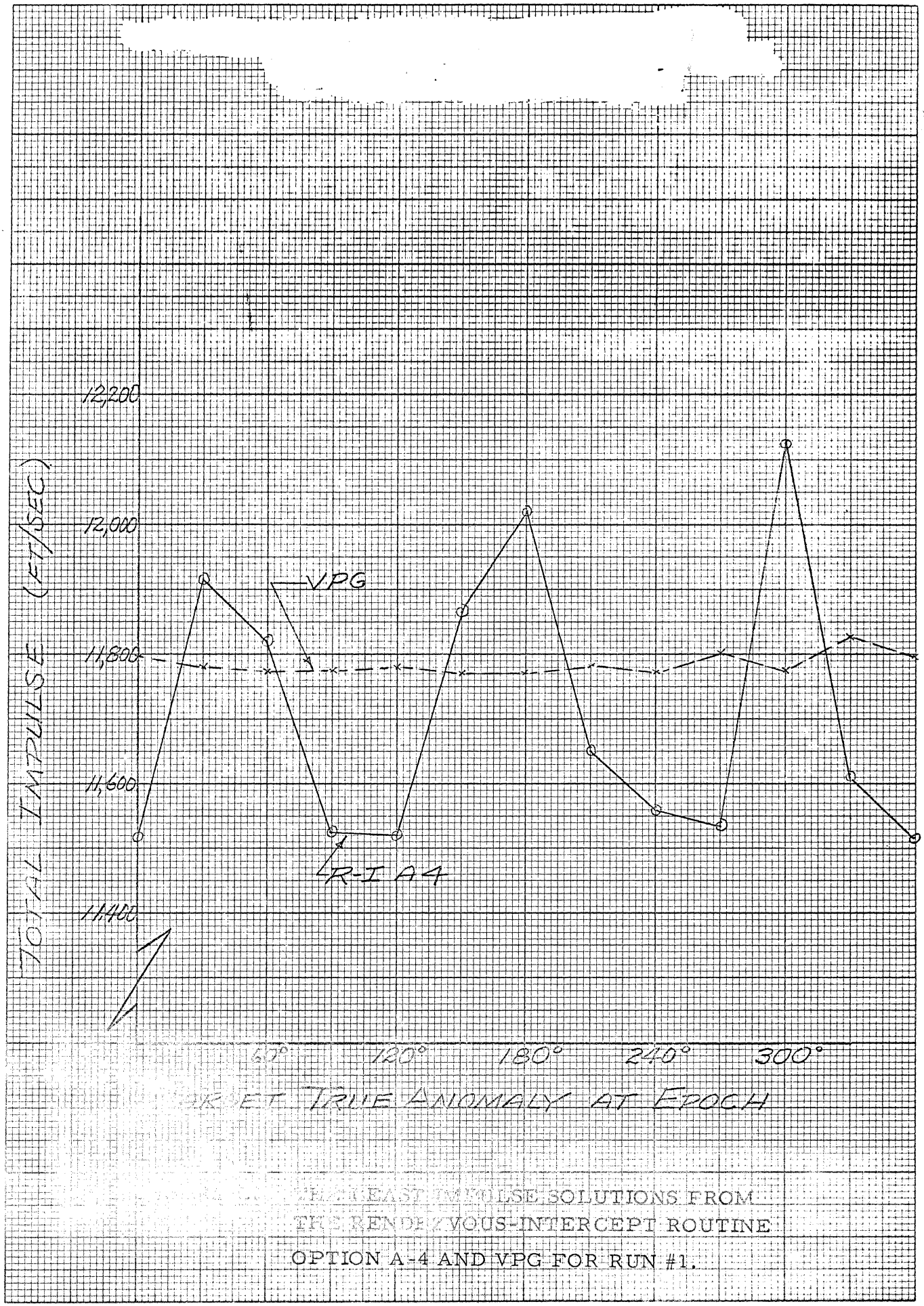


FIGURE C1 ORBITAL GEOMETRY FOR RUN 1. THE TRUE ANOMALY OF THE TARGET AT EPOCH IS VARIED FROM 0° TO 330°



LEAST IMPULSE SOLUTIONS FROM
THE RENDEZVOUS-INTERCEPT ROUTINE
OPTION A-4 AND VPG FOR RUN #1.

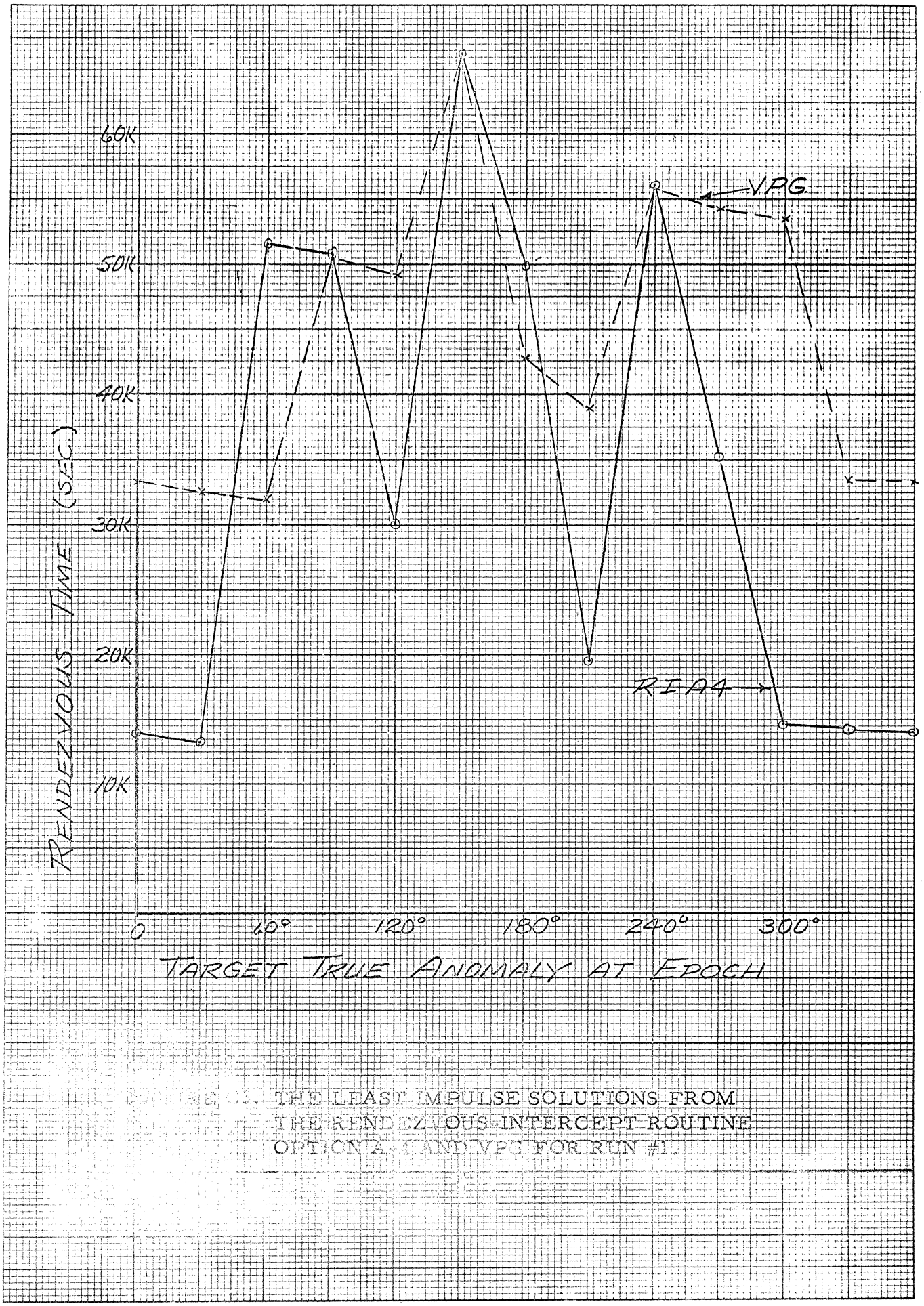


FIG. 5. THE LEAST IMPULSE SOLUTIONS FROM THE RENDEZVOUS-INTERCEPT ROUTINE OPTIONAL AND VPG FOR RUN #1.

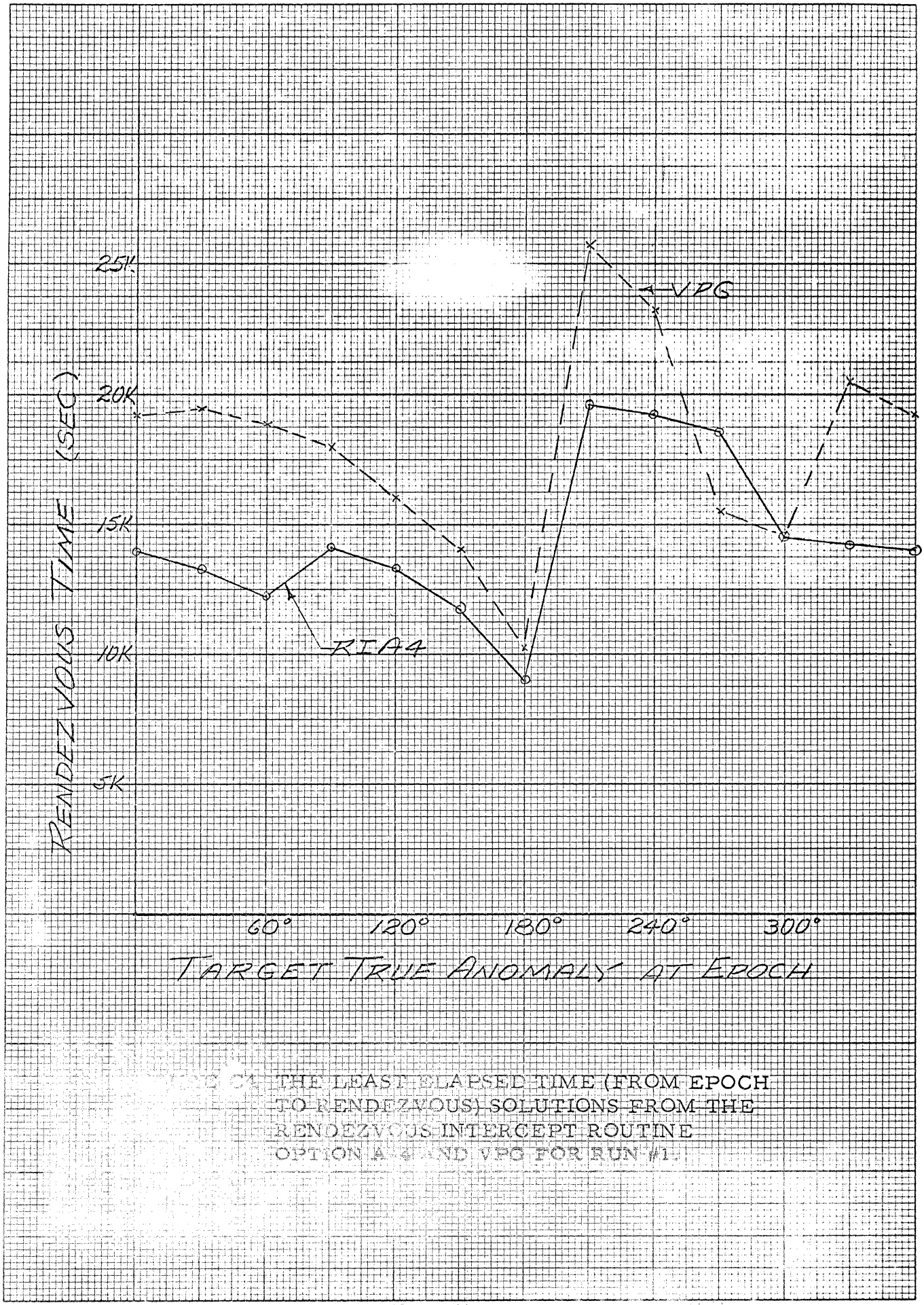


FIG C4 THE LEAST ELAPSED TIME (FROM EPOCH TO RENDEZVOUS) SOLUTIONS FROM THE RENDEZVOUS INTERCEPT ROUTINE OPTION A-4 AND VPG FOR RUN #1.

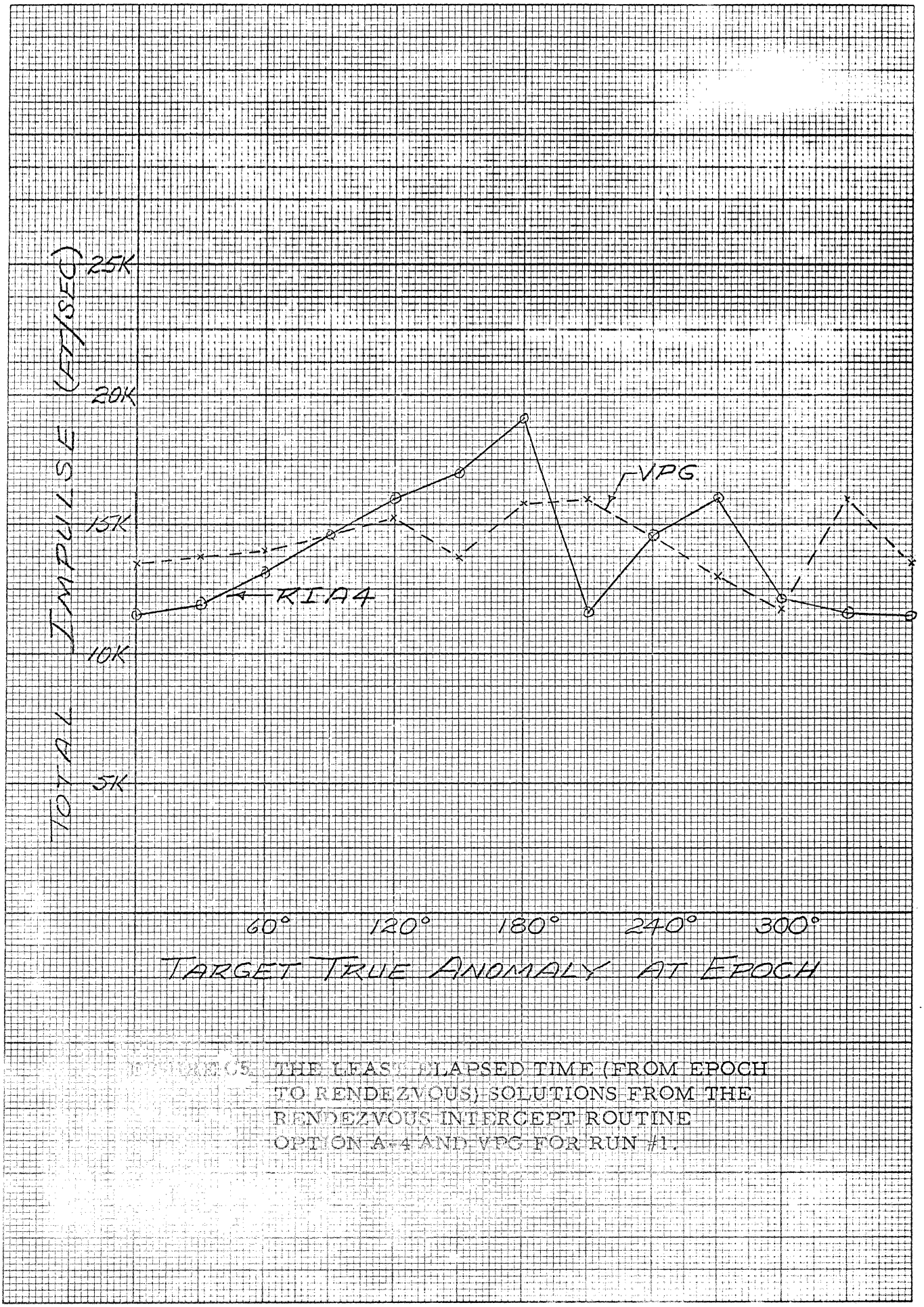


FIGURE C5. THE LEAST ELAPSED TIME (FROM EPOCH TO RENDEZVOUS) SOLUTIONS FROM THE RENDEZVOUS INTERCEPT ROUTINE OPTION A-4 AND VPG FOR RUN #1.

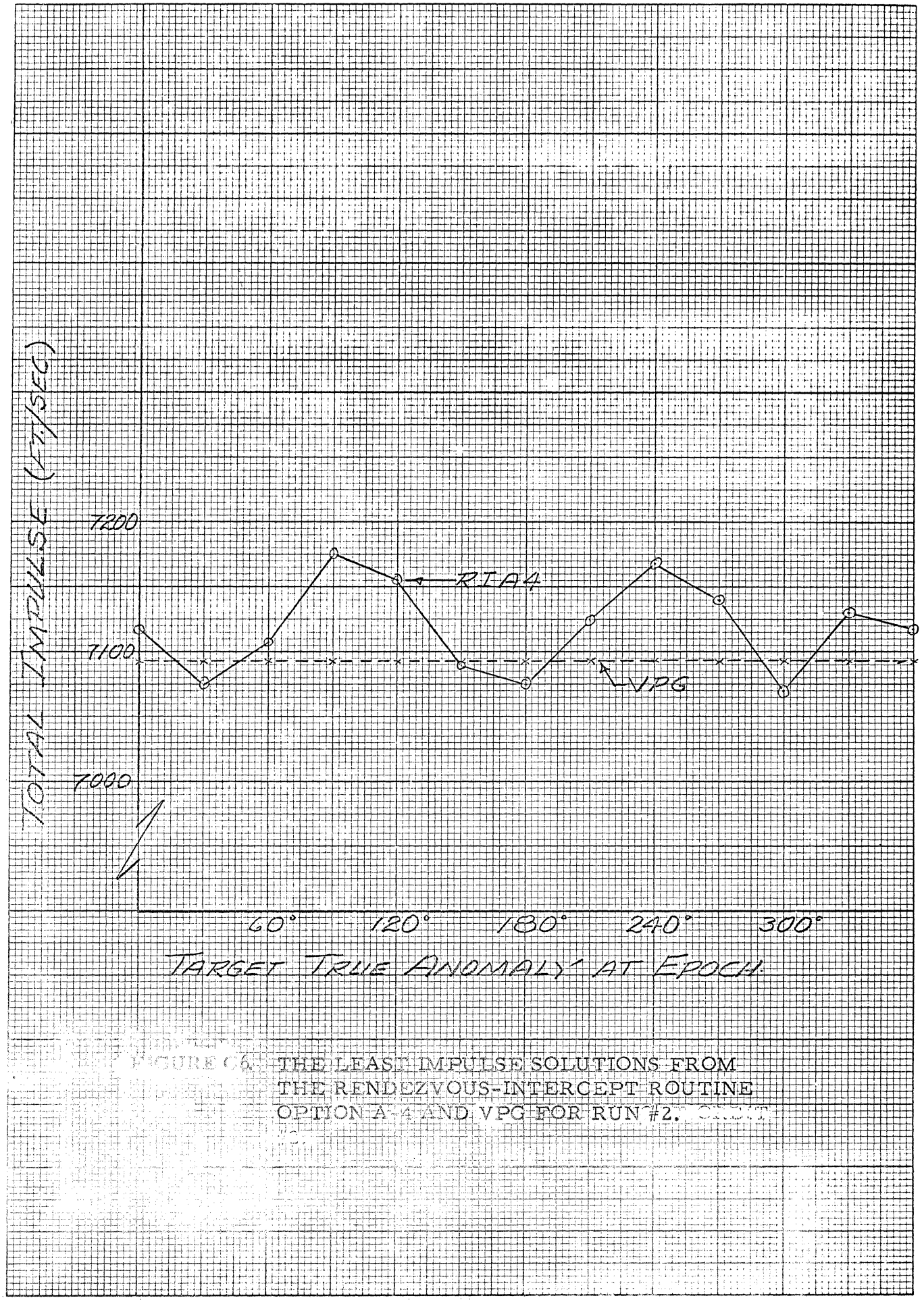


FIGURE C6. THE LEAST IMPULSE SOLUTIONS FROM THE RENDEZVOUS-INTERCEPT ROUTINE OPTION A-4 AND VPG FOR RUN #2.

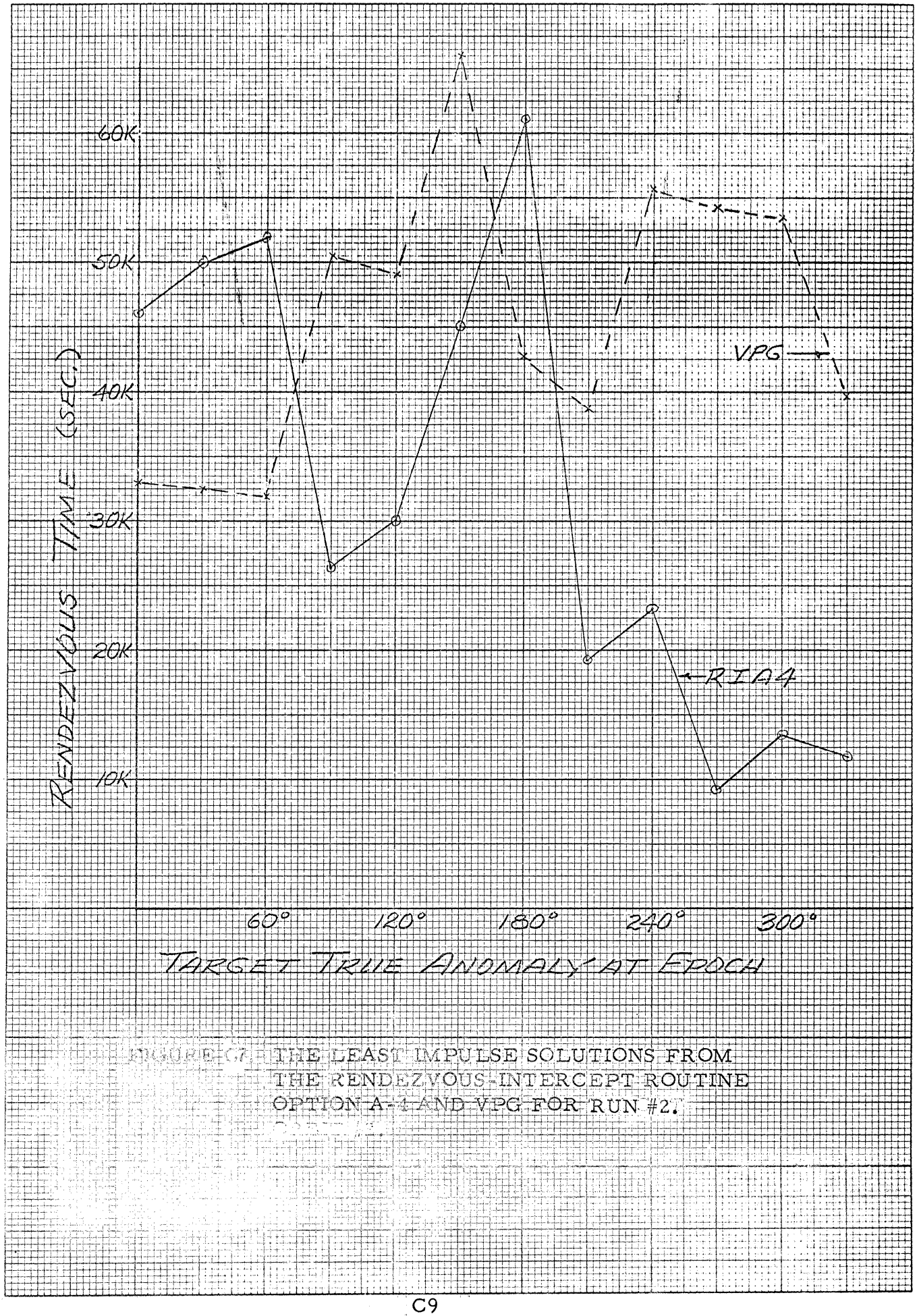


FIGURE C7 THE LEAST IMPULSE SOLUTIONS FROM THE RENDEZVOUS-INTERCEPT ROUTINE OPTION A-1 AND VPG FOR RUN #2.

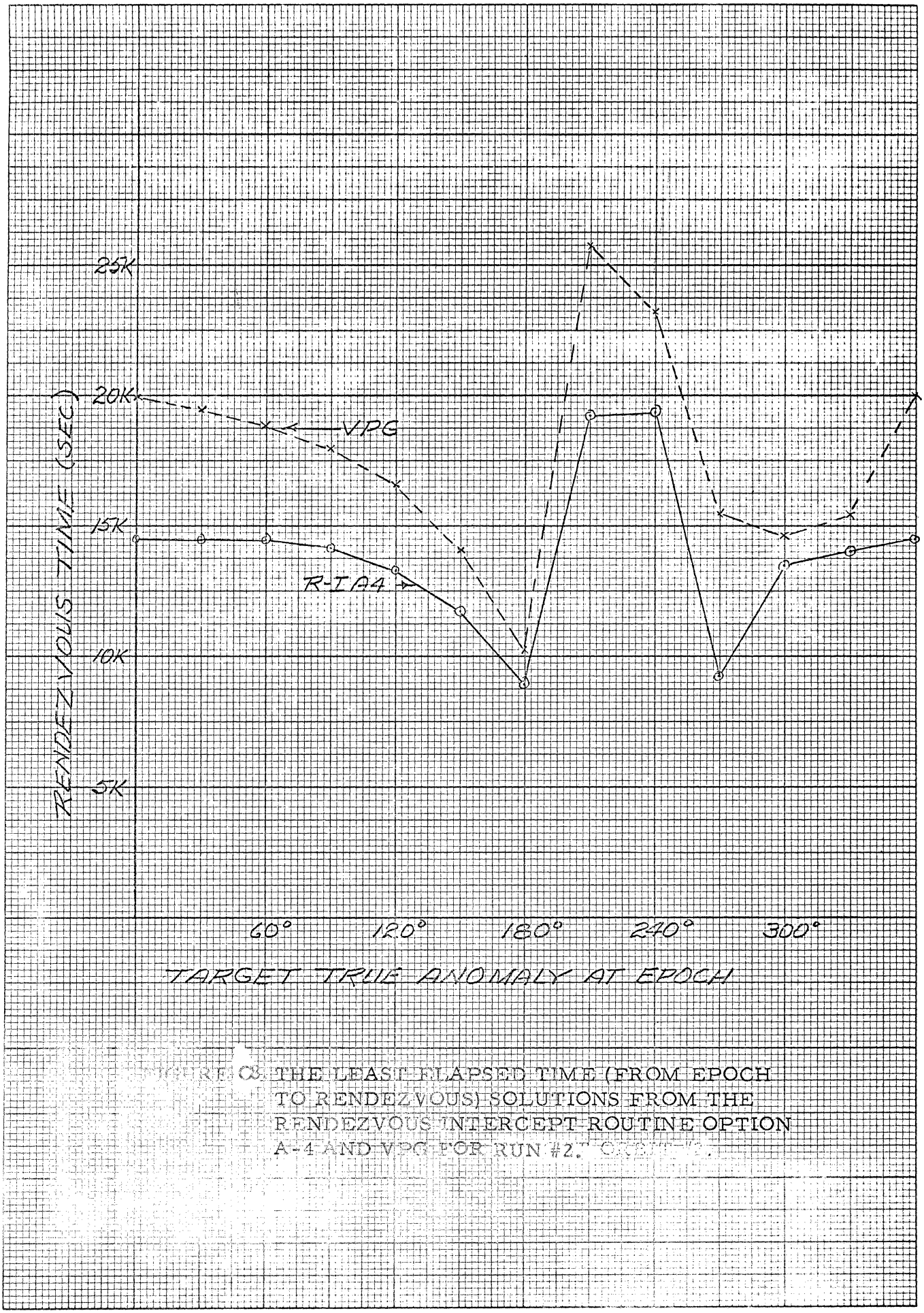


FIGURE C3 THE LEAST ELAPSED TIME (FROM EPOCH TO RENDEZVOUS) SOLUTIONS FROM THE RENDEZVOUS INTERCEPT ROUTINE OPTION A-4 AND VPG FOR RUN #2.

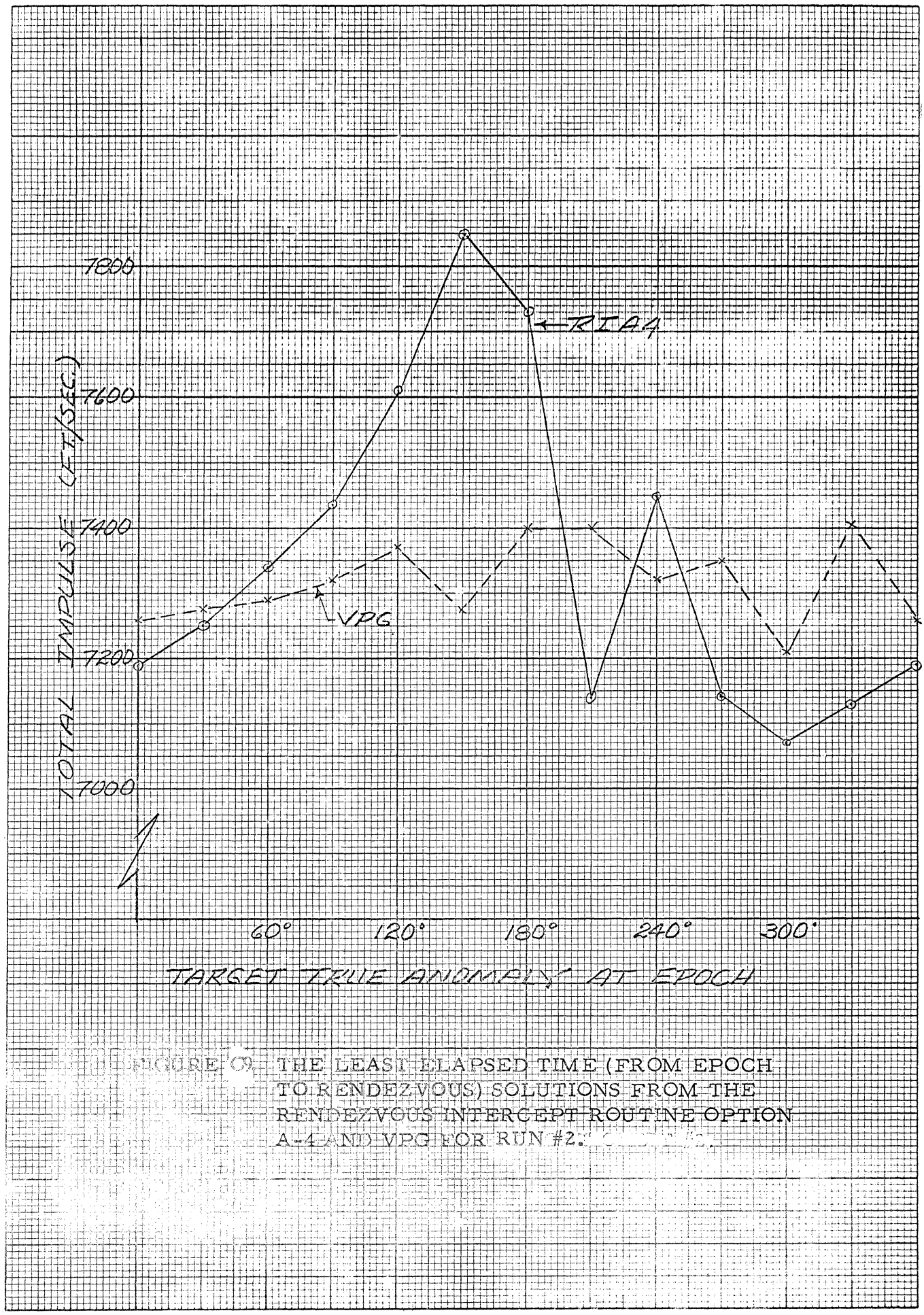


FIGURE C9. THE LEAST ELAPSED TIME (FROM EPOCH TO RENDEZVOUS) SOLUTIONS FROM THE RENDEZVOUS INTERCEPT ROUTINE OPTION A-4 AND VPG FOR RUN #2.

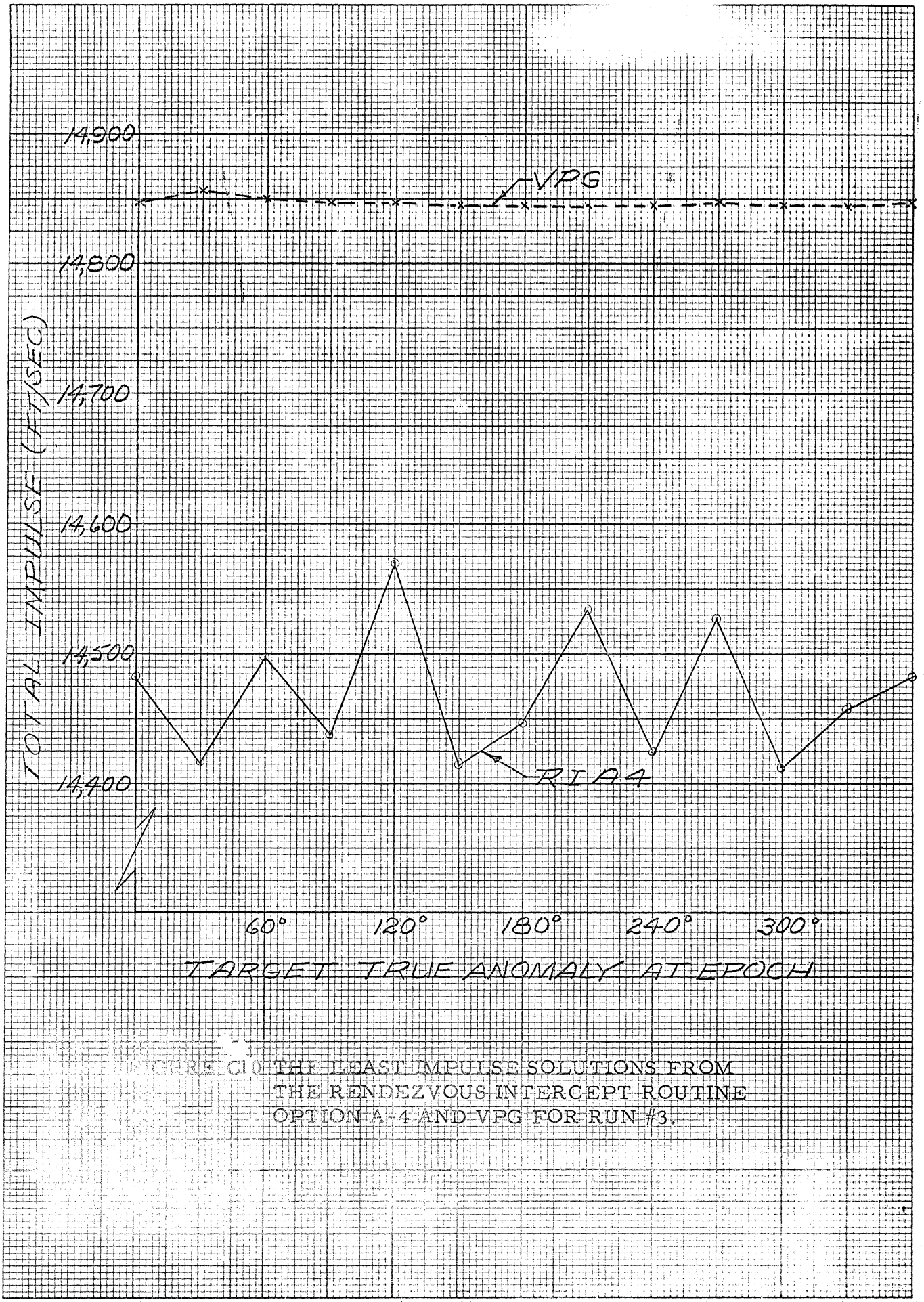


FIGURE C10 THE LEAST IMPULSE SOLUTIONS FROM THE RENDEZVOUS INTERCEPT ROUTINE OPTION A-4 AND VPG FOR RUN #3.

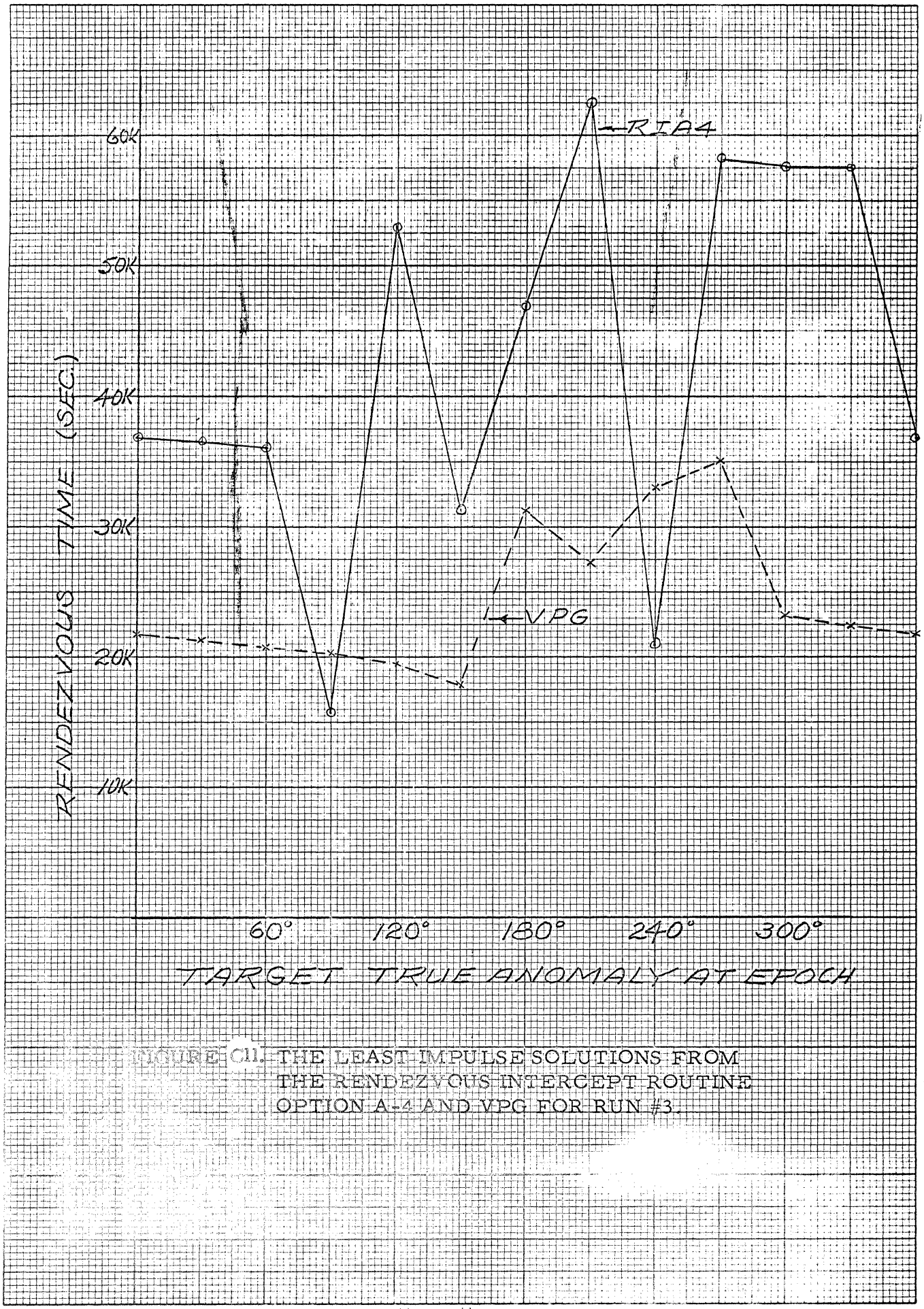


FIGURE C11. THE LEAST IMPULSE SOLUTIONS FROM THE RENDEZVOUS INTERCEPT ROUTINE OPTION A-4 AND VPG FOR RUN #3.

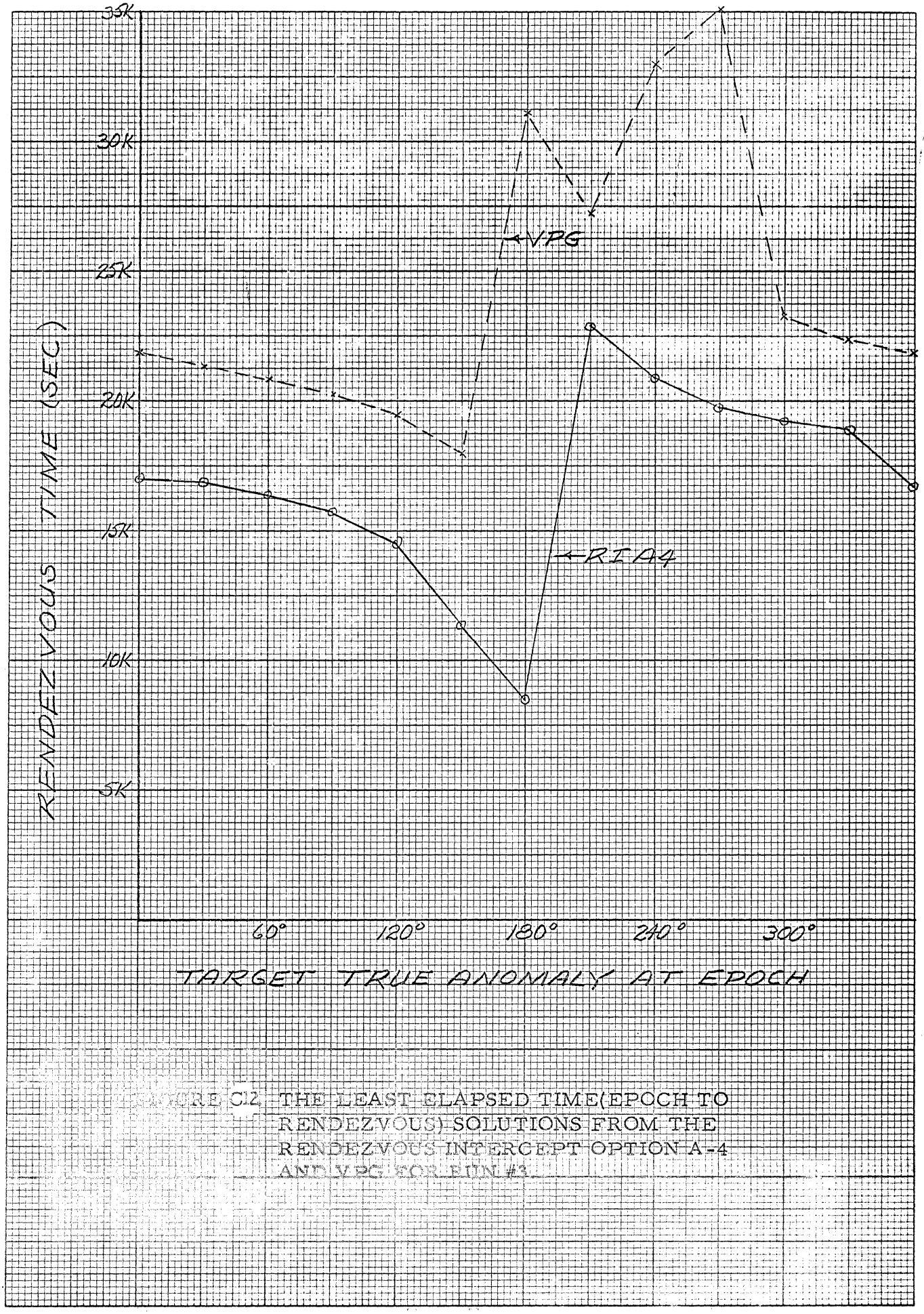


FIGURE C12 THE LEAST ELAPSED TIME(EPOCH TO RENDEZVOUS) SOLUTIONS FROM THE RENDEZVOUS INTERCEPT OPTION A-4 AND VPG FOR RUN #3

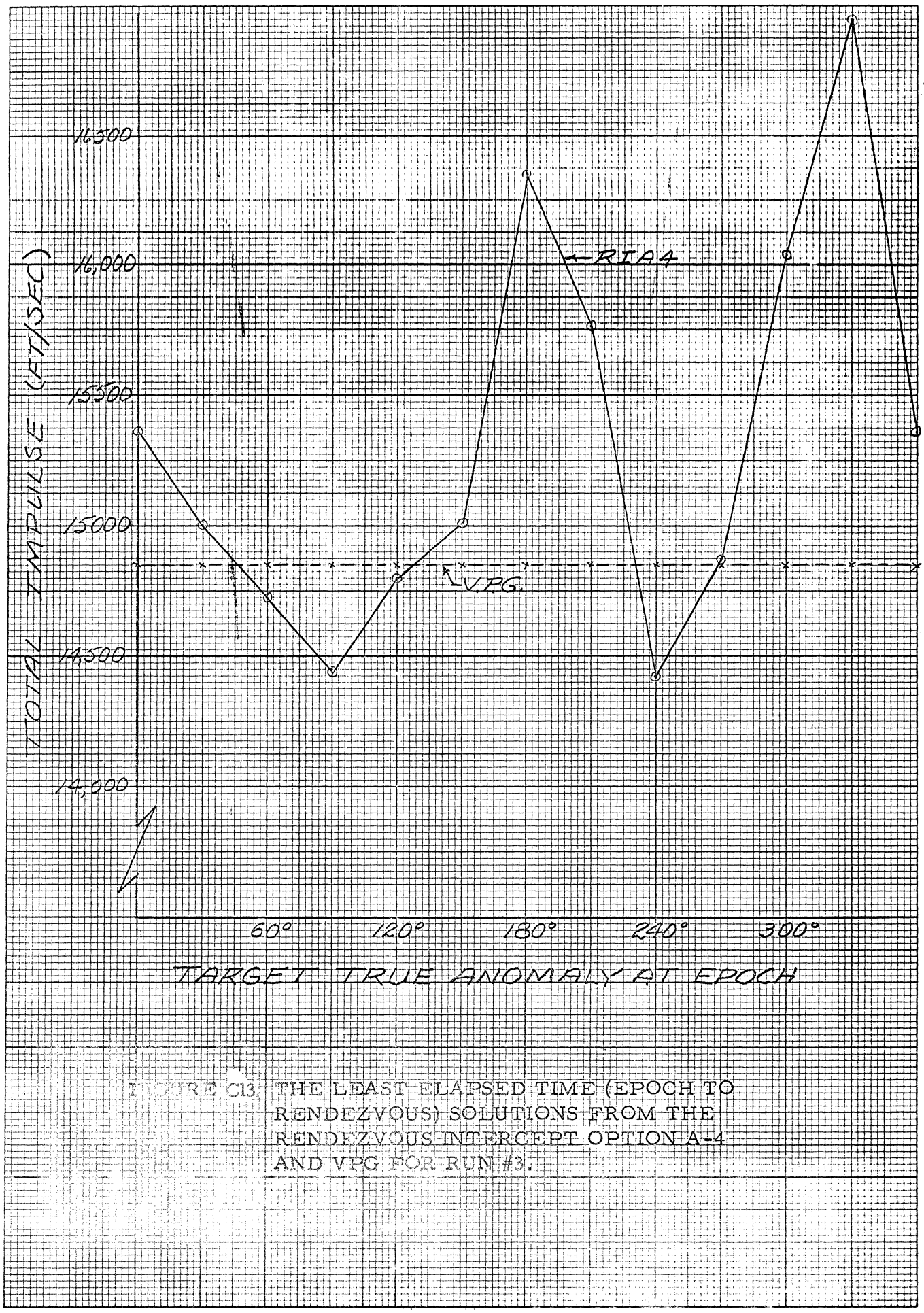


FIGURE C13. THE LEAST ELAPSED TIME (EPOCH TO RENDEZVOUS) SOLUTIONS FROM THE RENDEZVOUS INTERCEPT OPTION A-4 AND VPG FOR RUN #3.

APPENDIX D

EXPERIMENTAL OPTIMIZATION ALGORITHMS FOR QRGTS

INTRODUCTION AND SUMMARY

This appendix describes the optimization algorithms (i.e., algorithms for minimizing a given function of the independent variables, subject to a given set of constraints) which have been developed in the QRGTS study for use in on-board optimization of planned trajectories. These algorithms are experimental, and are still being modified. Experimentation is considerably facilitated by the organization of the optimization program which is called DSOP (Direct Search Optimization Program). DSOP consists of a number of subprograms which can be fitted together in various ways to generate optimization algorithms, plus a number of frequently used test problems.

As the name implies, the optimization algorithms of DSOP are of the "direct search" type; that is, they make no use of analytical derivatives. A test problem (or any problem to which the problem is applied) is, therefore, completely defined by equations for evaluating the function to be minimized and the constraint functions (if any). No equations for partial derivatives of these function are required. This characteristic is highly desirable in an optimizer designed for use on-board a space-carft, since it greatly reduces the number of routines that must be provided if there are a large number of different optimization problems which may require solution. Also, it makes the optimizer applicable to problems where analytic derivatives would be difficult or expensive to generate, and improves flexibility by simplifying the specification of new problems for optimization and the modification of existing ones to accommodate changes in mission objectives and/or constraints, etc.

The principal subprogram of DSOP is called PMS (Pattern Move Search). PMS is a version of the rather famous search technique developed by Hook and Jeeves (Ref. 1). Its logic is explained in the PMS Section.

PMS is not a self-contained optimization algorithm; it requires an "exploratory" subroutine which performs a restricted local search about a given point. DSOP provides a number of different subroutines which can be inserted in PMS for this purpose. Modification of these subroutines, and generation of new ones, is the principal method used for developing, testing, and incorporating improvements. The framework of PMS, and of DSOP itself, remains relatively unchanged; the changes that have been made were such that early versions of the "exploratory"subroutines are still usable.

The principal exploratory subroutines are EMR (Exploratory Move Routine) and UNIVAR. EMR is modeled after the exploratory subroutine used by Hook and Jeeves (Ref. 1). UNIVAR is similar to part of a published optimization algorithm called BRENT UNIVAR (Ref. 2). These subroutines are described in the EMR and UNIVAR Sections.

The PMS program and its associated subroutines were first designed for minimization without constraints, and modified until they accomplished this reliably and moderately efficiently on a set of test problems. From these experiments, UNIVAR appeared to be a better "valley-follower" tham EMR, and was therefore selected for further development: PMS and UNIVAR were modified to provide for the enforcement of specified upper and lower bounds for each component of the argument vector. Also, PMS was modified to incorporate an improved stopping rule. EMR was not modified but EMR and the original version of UNIVAR are still usable for problems where argument bounds are absent, or do not affect the solution. Constraints other than argument bounds are presently handled by penalty functions.

An optimization algorithm developed by Powell (Ref. 3) has recently been incorporated into DSOP as a subprogram. Powell's algorithm is briefly discussed in Section 6. A more complete description is given in the cited reference. Powell's algorithm has no provisions for handling constraints, but shows extremely rapid convergence for unconstrained problems, and can be used for constrained problems by employing penalty functions to enforce the constraints.

Alternatives being considered for the next step in optimization algorithm development are (1) combining the ideas of PMS and of Powell's algorithm, possibly by developing a new new subroutine for PMS which incorporates some of Powell's ideas, (2) modifying Powell's method to handle constraints, or (3) midifying PMS to handle more general constraints than argument bounds.

NOTATION

The principal symbols used are defined below. Additional symbols will be defined as introduced, unless their meaning is clear from the context. In flow diagrams, the operations to be performed are defined partly by equations, partly by English sentences such as "EVALUATE P" or "BOUND $\underline{U}$".

N = dimensionality of vectors denoted by underlined symbols
$\underline{U}$ = argument vector with components U(I),, $1 \leq I \leq N$
$\underline{UA}$, $\underline{UB}$, etc. = remembered past values of $\underline{U}$. $\underline{UB}$ is the "base point" from which exploratory moves begin. $\underline{UA}$ is the "advance point", the best point found to date during the current exploration.
P = objective function evaluated with $\underline{U}$ as argument. Sometimes written P($\underline{U}$) for clarity.
PA, PB, etc. = remembered past values of P.
LC1, LC2, etc. = logical-choice variables with value ±1
$\underline{STE}$, $\underline{STF}$, etc. = vectors used in generating "steps" which change $\underline{U}$.
MINC(I) = minimum magnitude for stepsize in the I^{th} coordinate
UMAX(I) = prescribed upper bound for U(I)
UMIN(I) = prescribed lower bound for U(I)
BOUND $\underline{U}$ = a subroutine which enforces $\underline{UMIN} \leq \underline{U} \leq \underline{UMAX}$.
BVA (I) = binary variable with value 0 or 1 according as U(I) is to be held fixed or varied for minimization. The vector $\underline{BVA}$ is a program input used to control search strategy by providing for preliminary searches over a subspace.

BVB(1) = binary variable which keeps track of whether U(I) is at one of its bounds, or between them.

IFIN = an integer used in an exit test.

PATTERN MOVE SEARCH (PMS)

The basic logic of Pattern Move Search is shown in Figure D1. The program makes use of an exploratory subroutine, which operates in two different modes. The exploratory subroutine starts at a given base point UB, and makes exploratory moves, according to built-in-rules, in a search for a smaller value of P. The output of the exploratory subroutine is an "advance point" UA and function value PA, with PA < PB if the search succeeds, and with PA = PB, UA = UB if it fails. In "Mode Plus", the subroutine keeps trying, varying its stepsizes and/or other search parameters, until it either succeeds, or concludes that PMS has converged to a solution. In "Mode Minus", the subroutine stops after one run through of its exploration sequence, regardless of success or failure.

Pattern Move proceeds by generating a sequence of accepted base points, each of which gives a lower function value than the previous one. A "pattern move" generates a tentative new base point by displacing the newest accepted base point by a vector equal to (or proportional to) the difference between the newest and next-newest accepted base points. The tentative new base point is improved by use of the exploratory subroutine (in "Mode Minus") and the result tested to see if it is an improvement. If so, it becomes the newest accepted base point, and the cycle repeats. If it is not an improvement, i.e., if the pattern move (together with its associated local exploration) is a failure, the program returns to the newest accepted base point (UE) and executes the exploratory subroutine in "Mode Plus". If this fails, the search ends. If it succeeds, the output and input of the subroutine become the newest and next-newest accepted base points respectively.

A special merit of Pattern Move Search is that the tentative new base point generated by a pattern move is not tested for success until after an attempt has been made to improve it by a local exploration. This means that Pattern Move Search can follow a curving valley; even is pattern move misses the valley floor and hence gives a high function value, the subsequent exploration sequence finds the valley again, so the over-all move is a success. Search programs which always reject points that do not show an immediate improvement will sometimes be much less efficient that Pattern Move Search. (Sometimes they may be more efficient; persistence is not always a virtue.)

The detailed logic of Pattern Move Search is shown in Figure D2. Most of this Figure can be understood from the previous discussion, and by comparing with Figure D1. LCCONV is a quality which is set by the exploratory subroutine, and subsequently tested by PMS to decide whether to continue or to terminate the search. Another possible termination of the search is by EXIT-TEST, a subroutine that counts the number of consecutive cycles that have failed to improve the function value by amounts above a given threshold. IFIN is a counter used in this subroutine. A third mode of termination (not shown in the Figure) occurs if the number of function evaluations exceeds a given limit.

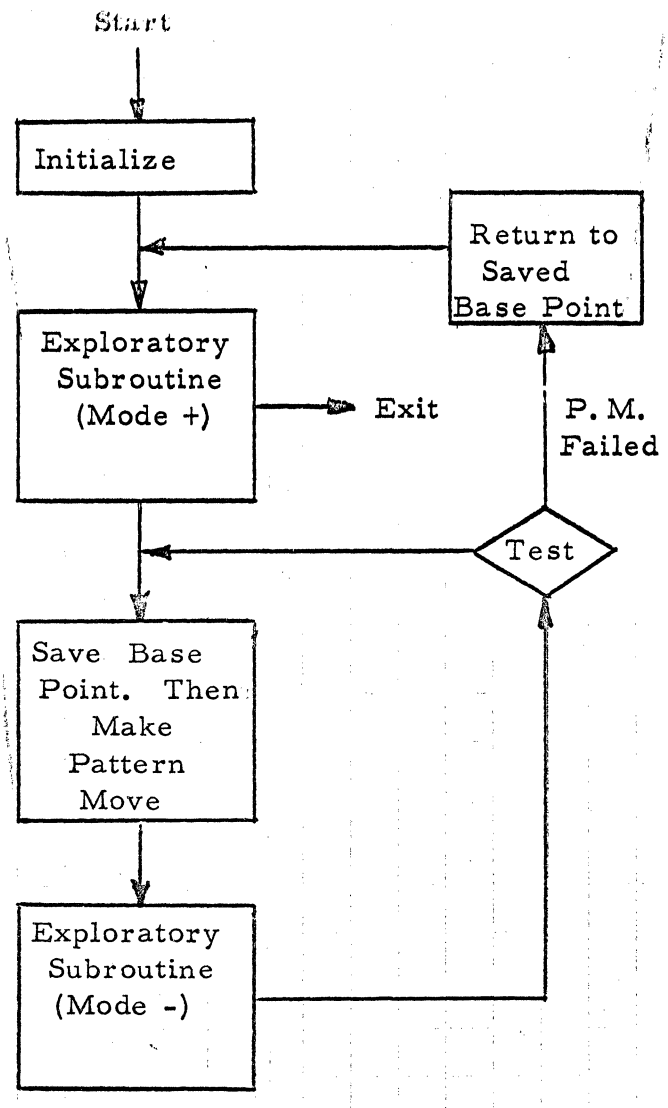


Figure D1. Basic Logic of Pattern-Move Search (PMS)

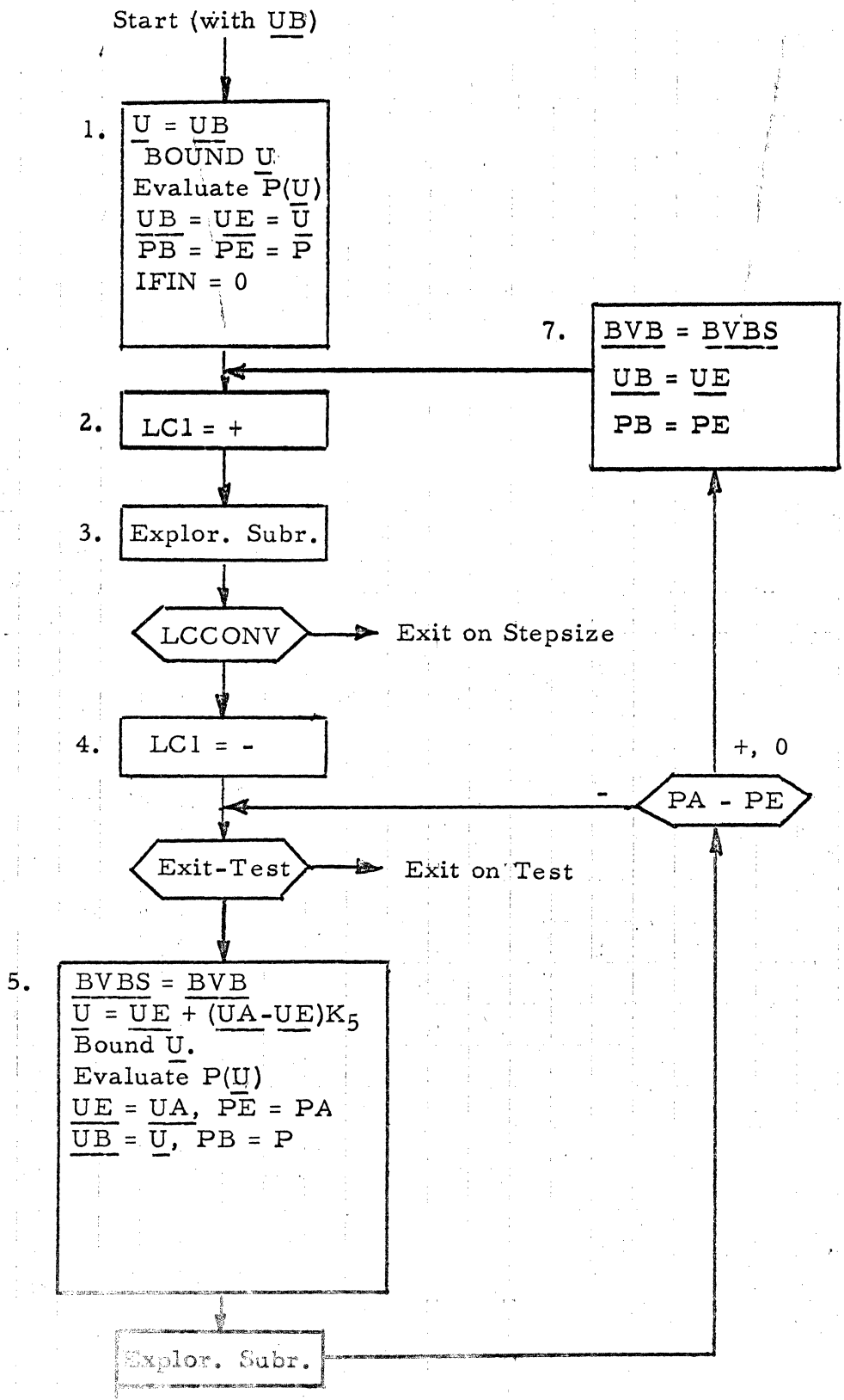


Figure D2. Detailed Logic of PMS

Block 1 in Figure D2 is initialization. Block 5 saves the newest accepted base point (labeling it UE) and performs a pattern move. The coefficient K_5 in Block 5 is currently set at $K_5 = 2$. By choosing K_5 greater than 2, the lengths of a sequence of pattern moves can be made to grow more rapidly, but with increased probability of failure. Block 7 restores the saved base point after failure of the pattern move. The vector BVB, which remembers which components of the argument vector are at their bounds (and hence require different treatment in the exploratory subroutine) is saved and restored along with UE. The operation denoted by BOUND U, in Blocks 1 and 5, is a subroutine that examines each component of U to see if it exceeds the prescribed upper or lower bound. If so, it is set equal to whichever bound was violated, and BVB is changed accordingly.

Pattern Move Search provides a very general framework into which almost any type of optimization algorithm can be fitted as an exploratory subroutine, with minimal changes in PMS itself. For example, if it should be decided to modify the optimizer so it can handle general sets of equality and/or inequality constraints, this could be done by developing a new exploratory subroutine with the desired capabilities. The only changes required in PMS itself would be (a) replacing the operation BOUND U with a more elaborate operation which enforces, or approximately enforces, all active constraints, by projecting the search point onto constraint surfaces, and (b) saving and restoring information which remembers which constraints are active; this requires an extension of the BVB vector.

EXPLORATORY MOVE ROUTINE (EMR)

Components of the argument vector U for which BVA(I) = 0 are treated as constraints; other components of U are varied systematically in a search for a local minimum. For each value of I for which BVA(I) = 1, the routine tentatively changes U(I) in one direction and then (if this fails to reduce the function value) in the opposite direction. The direction tried first is determined by the sign of STE(I), and the magnitude of this quantity determines the step size. If either change succeeds, the value of PA (which is the best value of P found so far) is updated, and the search proceeds from the altered argument vector, by considering the next value of I. If both directions of change along the Ith coordinate axis are unsuccessful, no change is made in U(I), and the routine moves on to the next value of I after saving some auxiliary information.

When all variable components of U(I) have been treated as described above, the routine decides whether to accept the result, or try one additional "interpolatory" step, whose components are given by a vector STF. The value of STF(I) is zero for any value of I for which BVA(I) = 0 or for which either of the steps ±STE(I) succeeded. For the remaining cases (i.e., BVA(I) = 1, and ±STE(I) both unsuccessful), STF (I) is computed by considering the location of the minimum of a quadratic function fitted to the function values found at the three points U(I) and U(I) ± STE (I) with all other components of U fixed. If K_1 were chosen to be 1/2, and no other coordinates were involved, STF (I) would give a move to this minimum. A value of K_1 less than 1/2 is appropriate in multidimensional cases. If P(U) is a quadratic function, and if all coordinate-steps have failed, it can be shown that the interpolatory step always succeeds if $K_1 < 1/N$, and is always too short (undershoots the minimum) if $K_1 < 1/2N$. Therefore

K_1 should be between 1/N and 1/2N.

The decision whether to try the interpolatory step or not is made by comparing the total function-reduction obtained by steps ±STE(I) with the expected further reduction obtainable by the interpolatory step. This expected reduction is proportional to SUM, if interaction between coordinates is ignored. A coefficient K_4 is provided which can be adjusted to bias the decision. If $K_4 \leq 0$, the interpolatory step is never tried unless all the step ±STE(I) failed. If K_4 is made very large, the interpolatory step will always be tried.

The step vector STE is updated during the routine, to provide a faborable step for next time. If step STE(I) is successful the new STE(I) is larger ($K_3 > 1$). If STE(I) fails but the reverse step succeeds, STE(I) reverses sign but does not change magnitude. If both steps ±STE(I) were unsuccessful, the new STE(I) has its sign determined by comparison of the two unsuccessful trials.

An option (LCI) is provided to allow the routine to exit on failure to reduce the function, or try again with reduced step sizes until either success is attained, or the step sizes are reduced to the allowed minimums given by MINC.

The performance ofEMR could probably be improved significantly by adding a one-dimensional search along the direction defined by STF, i.e., parallel to the "interpolatory move". However, this experiment has not been tried.

UNIVAR

The subroutine UNIVAR exists in two versions: an original version which has no provision for bounds on the argument variables, and a revised version that incorporates such bounds. Either version works with PMS.

The original UNIVAR proceeds by making a sequence of one-dimensional searches, each parallel to a coordinate direction,, until all directions have been tried except those for which BVA(I) = 0. Each one-dimensional search begins from the best point found by the previous search. The logic for a one-dimensional search is as follows:

Step in the stored direction (and by the stored amount; both are defined by a component of the step vector STE). If this succeeds, increase stepsize and repeat. If if fails, reverse direction and go the other way. Proceed until the minimum along the search direction has been bracketed, i.e., until a set of three points has been found with the smallest function-value in the middle. This occurs when a success if followed by a failure, or when the first step fails and is immediately followed by a second failure when the reverse direction is tried. Fit a quadratic function to the values found at these three points and try the point where this quadratic is minimal. The search result is either this new point or the previous middle point, whichever has the least function-value. The stepsize is reduced to cancel the last increase and is further reduced by an extra factor of one-half if the last trial (from the quadratic fit) is not a success. However, stepsize is not allowed to go below a prescribed minimum size.

This one-dimensional search procedure is known to be inefficient, since it requires a minimum of three new function-evaluations per direction, whereas a procedure due to Powell (based on remembering and updating an estimate of the second derivative) reduces the minimum to two. Adopting Powell's rules for one-dimensional searches might reduce the number of function evaluations for each execution of UNIVAR by as much as 33%. The average reduction would be less than this, but still significant. This improvement has not been incorporated because other developments have higher priority.

The revised UNIVAR is the same as the original UNIVAR except for extra logic to handle bounds on the independent variables. The binary-valued vector BVB keeps track of which variables are at their bounds. If BVB(I) = 0, indicating that the I^{th} coordinate is in the interior of its allowed range, the search along the I^{th} direction is the same as in the original UNIVAR, unless a tentative step reaches or exceeds a boundary. In this case, a step to the boundary is tried instead. If this step fails, the search along the I^{th} direction will end at an interior point. If it succeeds, BVB(I) is set to 1, STE(I) is set to call for a small step (4 times the minimum size) away from the boundary, and the routine goes on to the next direction.

If BVB(I) = 1, indicating that the I^{th} variable is at a boundary, this coordinate is left unaltered unless LC1 = +, in which case a step away from the boundary is attempted. If this succeeds, BVB(I) is set to 0 and the search proceeds as from an interior point. If it fails, stepsize is reduced to half its previous magnitude, or to the minimum magnitude allowed, whichever is greater.

The motivation for restricting attempted moves away from a boundary to cases when LC1 = + (i.e., to times when the pattern-move sequence is being started for the first time, or is being restarted from an old base point after failure of a pattern move) is to reduce the number of unsuccessful attempts to move away from the boundary that would otherwise occur if the true final solutions calls for one or more variables to be at their bounds. Since search termination on step size is permitted only when LC1 = +, the program cannot exit on step size without having tried moves away from all boundaries. It can "exit on test" if there is a prolonged sequence of pattern moves which are all successful but give very small improvement, but this is highly unlikely; in general, one of the pattern moves will fail, and the next execution of UNIVAR will attempt moves away from the boundaries.

VA04A (POWELL'S 1964 ALGORITHM)

VA04A is the name given by M. J. D. Powell to the function-minimization program developed by him and described in a 1964 paper (Ref. 3). Note: Powell has also published other algorithms in 1962, 1963, and 1965, which are entirely different from this and should not be confused with it. We have converted VA04A from Fortran II to Fortran IV and incorporated it into DSOP.

The [illegible] operating cycle of VA04A has been concisely described in a review paper by Fletcher (Ref. 4) who presents it in pseudo-ALGOL notation as:

```
y: = x;
for i: = 1 step 1 until N do MIN(i);
for i: = 1 step 1 until N-1 do p_i: = p_{i+1};
p_N: = y-x; MIN(N)
```

where x is an N-component argument vector (equivalent to the $\underline{U}$ of our notation), y is the value x had at the beginning of the cycle, and MIN(i) is a subroutine which performs a one-dimensional minimizing search parallel to p_i, starting from the best point found previously. The vectors p_i are originally chosen to be parallel to the coordinate directions. The routine performs a linear search parallel to each p_i in sequence. The net resulting change in x defines a new p-vector, and all the p-vectors shift down in the list, the oldest (the previous p_1) being discarded.

Powell's actual procedure, as described in his 1964 paper, is somewhat more complex than this. It is evident that if his set of vectors p_i ever become linearly dependent, his search process becomes trapped on a hyperplane and will never find the solution if it is is not on this plane. To avoid this, he does not always discard the oldest vector; another vector, chosen by a rather complex set of rules, may be discarded instead. Also, the newly generated direction is not always accepted.

Powell's program performs N+1 one-dimensional searches per cycle, each search requiring at least two new function evaluations. It will find the exact minimum of a quadratic function in N cycles, requiring a total of $N^2 + N$ linear searches. For non-quadratic functions, the exact answer is generally not found, and more than N cycles will usually be required to obtain the accuracy desired. The convergence is ultimately quadratic. That is, if the function to be minimized behaves like a quadratic function in the neighborhood of the optimum, then if the search point is sufficiently close to the solution and at least N cycles have occurred, each subsequent cycle will double the number of significant figures. The number of linear searches required for minimizing a quadratic function could be halved by a slight change in the program logic, but this change would not double the program's efficiency for minimizing non-quadratic functions; at most, it might save half of the first $N^2 + N$ searches that would otherwise be performed, but subsequent operation would not be improved.

VA04A has outperformed the PMS-UNIVAR combination on simple test problems in two and three dimensions. It will next be tried on higher-dimensional examples with penalty functions and on actual mission-planning problems.

VA04A has no provisions for handling constraints (except by using penalty functions). An interesting, and presently unsolved, problem is how to modify this program so it can directly handle general constraints, without losing its quadratic convergence.

References for Appendix D

1. Hooke, R., and Jeeves, T., "Direct Search Solution of Numerical and Statistical Problems," J. Assoc. Computing Mach. 8, 212-229 (1961).

2. Leon, A., "A Comparison Among Eight Known Optimizing Procedures," in Recent Advances in Optimization Techniques (book) John Wiley, 1966.

3. Powell, M. J. P., "An Efficient Way of Finding the Minimum of a Function of Several Variables Without Computing Derivatives," Computer Journal 7, 155-162, 1964.

4. Fletcher, R., "Function Minimization Without Evaluating Derivatives - a Review" Computer Journal 8, 33-41, 1965.

APPENDIX E

GLOSSARY OF TERMS EMPLOYED IN THE QRGT STUDY

Active Guidance	The closed-loop steering of a space vehicle during a propulsive phase, or during a skip or re-entry.
Active State	A term used to describe the status of an on-board applications program which is in main storage, is sharing the CPU, and is not in a wait state.
Actual State	Six parameters which describe the true or actual position and velocity of the spacecraft at a given time. See Reference State and Error State.
Adaptation	The single burn which changes the orbit of the S/C to closely approximate that of the target vehicle. This maneuver is usually made in the vicinity of the target vehicle to decrease the relative velocity, thereby allowing more time for terminal sensors to search for, and acquire the target.
Address Constant	An expression (value, term or combination of values and terms) entering into the computation of a main storage address.
Aerodynamic Velocity Vector	The velocity of a space vehicle relative to that of the neighboring air mass.
Algorithm	A computational procedure.
Allocate	To reserve or grant a computer system resource for a job or task.
AMU	Auxiliary Memory Unit - An airborne computer subsystem which stores programs that can be read into the main memory, Normally a tape or drum.
Anomaly, True	The angle, measured from perigee, that the geocentric position vector of an orbiting vehicle sweep outs.
Apsides, Line of	The line connecting the apogee and perigee of an elliptical orbit.
Assembler	A ground-based support program which accepts programs written in symbolic language and translates these into the machine language of a specific computer.
Bielliptic Transfer	An orbit transfer maneuver involving three burns. VPG Bielliptic denotes a VPG rendezvous maneuver with the second and third burns on the line-of-nodes and 180 degrees apart. See VPG.

Bin Storage	An area of storage partitioned in a manner which permits each partition to be allocated to a job or task to accommodate storage requirements which vary (expand or contract) during execution.
Calling Sequence	The instruction and data words required to establish transfer of control to, and facilitate return from, a software module (routine).
Channel	The portion of a computer system used to control the transmission of data between an I/O device and main storage.
Chase:	A VPG Rendezvous maneuver that occurs between the Interceptor and Target orbits. See VPG.
Compiler	A support program more powerful than an assembler which can translate statements; formulas and operators into an executable sequence of instructions for a given computer. For example, the FORTRAN compiler.
Configuration	A ground-based support program which selects flight program elements from the Multi-Mission Library and configures a set of program modules for a specific vehicle.
Conjugate Directions	A generalization of orthogonal directions. If H is the Hessian matrix (matrix of second partial derivatives) of a quadratic function of n variables, two vectors p and q are conjugate with respect to H if $p^T Hq = 0$. The concept of conjugate directions is useful in finding the minimum of any function which behaves like a quadratic function in the neighborhood of the minimum.
Conjugate Variables	In the Hamiltonian formulation of the theory of optimal control, a Lagrange multiplier P_i (t) is associated with each state variable x_i, and the variables x_i, P_i are said to be conjugate to each other. The Lagrange multipliers are also called adjoint variables or costate variables.
Control Program	A term used to describe collectively all routines which perform scheduling, initiate and terminate execution, allocate resources, and otherwise provide common and conventional services to applications programs.
Control Section	The smallest separately and contiguously loadable unit of code which can be established as an entity at coding time.
Control Unit	That portion of an I/O device which adapts device characteristics to the standard form of channel control provided in the computer system. A CU may be an integral part of a device or a physically separate unit; it may be used for a single device or may be common to several devices.

Conversational Mode	Mode of operation in which the computer using a display, and the operator, using a manual input device, guide each other to the desired end.
CRT	Cathode Ray Tube.
Cutoff State	The state vector at thrust termination.
Data Management	The control program functions which provide access to data according to conventional usage by an applications program. These functions include regulation of I/O device usage, initiation of I/O operations, and avoidance of conflict for device usage requested by various applications programs.
Data Set	Any named collection of instructions or data residing on an external (I/O) device organized and described in a manner which permits access by data management routines. Somewhat synonomous with "file".
Decision Variable(s)	Parameters specifying the candidate trajectories for use is selecting a particular trajectory. These parameters might include: V required, time required, constraint(s) violated, etc.
V	Total impulse; the integral of thrust acceleration.
Direct Search	Any procedure for finding the minimum or maximum of a function without using derivatives (except, possibly, approximate derivatives obtained by finite differences).
Display	The equipment used to present information, or the information presented.
Dog Leg Ascent Trajectory	An ascent trajectory which includes a plane change introduced after the region of high dynamic pressure. It is used principally to avoid launch range obstacles.
Dormant State	A term used to describe the status of an on-board program on the AMU but not in main storage and not currently competing for computer system resources.
DSOP	Direct Search Optimization Program; a computer program developed under the QRGT study. It contains experimental algorithms for optimization (function minimization).
Dynamic Storage Area	The area of storage which may be allocated at execution time to applications programs and executive system utility (service) programs.
ECI	Denotes an Earth Centered Inertial coordinate system.
Element	In discussing mission modularization, a Mode Module which can be identified with a particular implementation technique,

	subsystem, or mission requirement. Presently, the lowest level of the mission hierarchy.
EMR	Exploratory Move Routine, part of DSOP.
Entry Point	Any point in a program which is specified, at coding time, as a point to which control can be passed, at execution time, by another program.
Epoch	An instant of time selected as a reference for past or future events.
ER	Ephemeris Routine; a computer program for predicting the future position and velocity of a spacecraft.
Error Mapping	Evaluation of the error state at time t_2 as a function of the error state at time t_1 is referred to as a mapping from t_1 to t_2.
Error Propagation	Behavior of the error state as a function of time.
Error State	The difference between actual and reference state at a given time. Error state is a six-dimensional vector.
Execution List	A list (queue), used and maintained by the OBES scheduler, of all active and waiting programs sharing CPU time.
Executive System	The control program for the on-board computer.
Explicit Guidance Technique	A guidance algorithm which employs an approximate means of predicting the trajectory that will result from a given steering policy, and which dynamically adjusts the steering coefficients to meet specified terminal conditions. The algorithm requires a minimum of precomputed information.
External Reference	A reference within a software module to a symbol defined in another module.
External Symbol	The name of an entry point, control section, or external reference which appears in an external symbol dictionary.
External Symbol Dictionary	A section of an object or load module which identifies all symbols referred to in the module but external to it.
Fixed Storage Area	The portion of main storage containing permanently - resident executive system routines and imbedded transient areas containing (one at any given time) executive system routines which are not permanently resident.
Footprint	The Ground Area Attainable (GAA) from the current vehicle state. In this report the term refers to the approximation to

	the true footprint defined by: maximum, minimum range and maximum cross-range trajectories.
Full Orbit Phasing	A VPG Rendezvous Maneuver that includes a phasing ellipse tangent to the target orbit at the line-of-nodes. See VPG.
Function	In mission modularization, the major operations necessary to implement the various mission Phases and which are generally identifiable with major vehicle systems. The functions presently defined are: Navigation, Guidance, Control, Mission Planning, System Readiness, Digital Communication, Display, Acquisition and Tracking and Mission Support.
GAA	See Footprint.
GISMO	General Integrated Simulation Model - a versatile point mass vehicle simulation program especially constructed to allow easy implementation of candidate guidance algorithms.
GMM	An acronym for the General Matrix Manipulator.
Ground Operating System (GOS)	A term used to collectively denote all software components used in the ground environment for the preparation, debug, execution, test, maintenance, integration, configuration, and validation of QRGT programs for pre-mission analysis or later on-board use.
Ground Configurator	A GOS program which extracts from the multi-mission library (MML) a subset of all program modules which support all mission functions within the capability of a specified vehicle configuration. A subset formed in this manner is referred to as a specific vehicle library (SVL).
Guidance	The process of generating commands for attitude and thrust control, from navigation information and a definition of the desired final conditions. This definition excludes the execution of the commands (flight control) and the determination of vehicle state (navigation). Guidance is also used in a broader sense, to denote the entire process of automatically directing a vehicle to a desired objective.
Guidance Error	Differences between actual spacecraft position and velocity and desired spacecraft position and velocity are caused by approximations and simplifications in the guidance equations.
Herricks Universal Variable	The independent variable rediscovered (1960) by Samuel Herrick which results in formulae for computing the motion of a space vehicle that is equally valid for elliptic, parabolic, hyperbolic and rectilinear orbits. The discovery of the universal variable is generally credited to Stumpff (1947). It was also independently discovered by Battin (1964).

Impulse Splitting	The application of two co-linear velocity increments separated by one or more complete orbits such that the V for the orbital change is identical to that of a one-burn maneuver.
Impulse, Total	The sum of the magnitude of the velocity increments in any sequence of propulsive maneuvers. Also called V.
Inactive State	A term used to describe the status of an on-board program which is not currently queued in the execution list but is in main storage.
Initial Program Loading (IPL)	An initialization procedure which loads a boostrap loader, which in turn loads the system nucleus and control program routines into the fixed area of main storage and transfers control to the control program. In the ground environment, loading is from any available auxiliary storage device; in the prelaunch and space-borne environments, loading is from the vehicle AMU or from launch support equipment.
Initiator/Terminator	A job management routine within the GOS control program which collects information and resources for a job stepand initiates and terminates each job step.
Input Job Stream	A sequence of job control language statements and data entering the system via a system input device. Applicable to GOS.
Intercept	Used in Mission Planning and Guidance, to denote the passage of the planned trajectory through a prescribed point (which may be time dependent) with no requirement to match velocity. Hence differs from rendezvous which requires a velocity match also. See intercept condition.
Intercept Condition	The requirement that a trajectory satisfy three position constraints at a future time (i.e., pass through a given possibly time-dependent, point in space), but need not satisfy any velocity conditions simultaneously. The corresponding transversality condition is that the printer vector vanish at intercept time.
Interrupt	A computer system signal originating from a source usually external to the CPU which changes the sequence of instruction execution without the use of a pre-coded branch instruction.
I/O Command	A computer word specified by the programmer and decoded by the I/O channel which determines the I/O operation to be performed and the main storage area and extent from (to) which data is to be transmitted.
I/O Instruction	A computer word decoded by the CPU which specifies an I/O device address at which an I/O operation is to be initiated, tested, or stopped.

I/O Order	A specification of functions peculiar to an I/O device such as rewind (tape), lineskip (printer), set density (tape), etc.
I/O Interrupt	A computer system signal originating in the I/O channel most commonly at the completion of an I/O operation. The signal changes the sequence of instruction execution within the CPU.
Jacobian Matrix	An n^{th} order square matrix of the partial derivatives of n functions of n variables with respect to the n variables.
Job	One or more job steps specified externally as a unit of work to be performed by GOS software and the ground-based computer system.
Job Management	A term used to describe collectively all functions of the GOS control program related to job step and task sequencing.
Language Translator	A program which produces machine-language code in one language from statements written in another language - e.g., an assembler or a compiler.
Launch Window	The time interval during which a particular mission plan is applicable.
Light Pen	A manual input device used to specify to the computer an item of information being displayed on a CRT.
Linear Tangent Steering Policy	A means of specifying the time history of the angle between the thrust axis and a reference axis such that the tangent of the angle is a linear function of time.
Linkage	The means by which communication between routines is achieved.
Lob	A VPG Rendezvous Maneuver that goes outside the target's orbit. See VPG.
Macro	A one-word statement of a coded program which is used to generate a sequence of machine instructions at a given point in a program.
Macro Facility	The feature of an assembler which recognizes the use of macros in an application program and produces the proper sequence of machine instructions. Specified by the definition of the given macro.
Matrix Manipulator, General	A set of algorithms programmed for a digital computer to perform vector matrix computations via a simple set of instructions. The program was specifically designed to perform the computations required for the statistical error analysis of Navigation systems.

Mission (as used in hierarchy level)	A generic mission defined by its objectives, e.g., intercept, rescue, shuttle, etc.
Mission Parameters	Replaced by Trajectory Parameters. See Trajectory Parameters.
Mission Planning	See Mission Planning Function
Mission Planning Function	An on-board computer function which constructs with minimal input, a validated flight program and targeting data based on a self-generated, optimized trajectory which satisfies vehicle and mission constraints, and mission objectives.
MML	Multi-Mission Library. A magnetic tape which contains all of the elements which are required to perform any mission for any vehicle configuration. It is the input tape to the Configurator used to construct an SVL.
Mode	In mission modularization, a specific application of a Function during a particular phase of a given mission.
Module	Any discreet programming unit which is identifiable and known by name to the operating system - usually a source, object, or load module.
Multiprogramming	Concurrent fulfillment of more than one computational task using the same computing system.
N-dimensional Search Routine	A computer program which systematically varies N independent variables to find the maximum or minimum of a given function, subject to given constraints.
Navigation	The determination of present vehicle state (position, velocity, and possibly attitude). Also used in a broad sense to denote the entire process of directing a vehicle to a desired objective. In this broad sense, navigation includes guidance or is synonomous with guidance.
Navigation Error	Differences between actual position and velocity and desired position and velocity which are caused by errors in navigation instruments and errors in mathematical models.
Newton-Raphson	A means of solving n simultaneous transcedental equations by employing the Jacobian matrix to compute a correction to the last estimated solution. See Secant Method.
Node, Ascending	The point on the line of nodes at which an orbiting vehicle crosses a reference plane (usually the equatorial or ecliptic plane) from south to north.

Nodes, Line of	(1) The intersection of two orbital planes; (2) the intersection of the orbital plane and the equatorial plane.
OBES	On-Board Executive System - (See executive system).
Object Module	The output from each single successful execution of a language translator. Consists of one or more control sections, is relocatable, and includes control dictionaries as well as text.
Optimization Algorithm	A procedure for minimizing a given function of the independent variables, subject to given constraints.
Overlay	Any technique which permits use of the same main storage area for different portions of the same job.
Parking Orbit	Any coasting orbit.
Payoff Function	A function whose value is to be maximized for optimization. Also called the objective function; however, the latter term may also denote a function which is to be minimized rather than maximized.
Penalty Function	An extra term added to a function which is to be minimized by an optimization algorithm; its presence serves to enforce (approximately) a desired constraint relation.
Phase	A time-based segment of a mission, with boundaries defined by some change of operating conditions, such as ignition or cutoff of engines, etc. The phases presently defined are: Pre-Launch, Ascent, Abort, Parking Orbit and Transfer, Intercept/Rendezvous, Orbital Operations, and Re-Entry and Landing.
PMS	Pattern Move Search; an optimization algorithm included in DSOP.
Predictor-Corrector Integration	A technique used for numerical integration whereby the present and past values of the integrand are used to extrapolate or predict the integrand change over the next increment. Then the predicted present and past values of the integrand are used to improve or correct the integrand change over the same increment.
Preliminary Trajectory Generation	A mode of the Mission Planning Function in which planning Prescriptions and rules-of-thumb (e.g. VPG) are employed to calculate one or more candidate trajectories as starting points for Trajectory Optimization.

Prescriptions (Mission Planning Prescriptions)	The rules for generating one or more preliminary trajectories in the Preliminary Trajectory Generation Mode of the Mission Planning Function.
Primer Vector	A vector, first defined by Derek Lawden, and usually denoted by (t), which occurs in the theory of optimal rocket trajectories. Its direction gives the optimal thrust direction at any instant, and the variations of its magnitude determine (in a slightly more complicated way) the optimal control of thrust magnitude.
Program Control Table	The portion of an on-board applications program which provides the control information needed by OBES to service the program.
Program Status	A computer word which contains information describing the status of the computing system in relation to the program being currently executed.
Quadratic Convergence	A form of rapid convergence of an iterative process, in which the error after an iteration is asymptotically proportional to the square of the previous error.
Range Angle	The angle traversed by a spacecraft; i. e., the difference of the true anomaly at the beginning and end of a transfer arc.
Reaction Time	The time interval between when a mission is specified and when the mission may be executed.
Re-Enterable	The attribute of a load module which permits its use as a system resource by two or more tasks concurrently. A load module which does not modify itself during use.
Reference State	Six parameters which describe the computed or estimated position and velocity of the spacecraft at a given time.
Relocation	Address Constant modification to compensate for a change in the origin (in main storage) of a module.
RIA4	The two-burn trajectory planning technique derived from the Rendezvous Intercept Routine Option A-4 (see QRT&GS Second Quarterly Summary Report, Section 2.2.4).
RIR	Rendezvous Intercept Routine; an algorithm for computing the required velocity vector to intercept a moving target point. This routine has several options which may be employed in active guidance or in the preliminary trajectory planning of orbital maneuvers.

RVVR	Required Velocity Vector Routine: an algorithm for computing the velocity vector that a spacecraft should have to free fall from a given initial point to a desired terminal point. A constraint must be applied to make the free fall trajectory unique.
Safety Margins	The amount by which safety constraints are not violated; e.g., perigee altitude, of a transfer maneuver, which is just above the atmosphere.
S/C	Spacecraft.
Scheduler	An OBES program which (a) activates all programs to be executed, (b) selects the program which is next to be allocated CPU time, (c) transfers control to the program selected, and (d) insures that each program is serviced as specified by its service class and repetition rate.
Secant Method	An iterative method of solving a non-linear equation, or set of equations. It differs from the Newton-Raphson method in that exact derivatives are not used. Instead, approximate derivative information is obtained by finite differences. There are numerous versions of the secant method, which differ in the detailed methods of obtaining and using approximate derivatives.
Segment	The portion of a program to which control is transferred each time the program is scheduled. Each time a program is scheduled, only one segment of the program is executed. Segment break points are established on the basis of CPU time required rather than number of instructions.
Slope at Arrival, m_a	The tangent of the flight path angle at arrival at the aim point; in plane polar coordinates, $m_a = \left(\frac{1}{r} \cdot \frac{dr}{d\theta}\right)_a$
Specific Impulse	Thrust per unit weight rate of flow. Typical units are seconds.
State	An n dimensional vector; often denotes the three components of the position and velocity vectors of a spacecraft.
SVL	Specific Vehicle Library. A magnetic tape which contains the set of program modes or functions which are needed to perform all missions which a specific vehicle can be used for. Used to load the AMU.
Target Offset	The required velocity vector algorithm aims at a fictitious point which is slightly displaced from the desired target point. This displacement termed a target offset is employed to compensate for the neglect of perturbative forces (drag and oblateness).

Targeting (As Employed In Mission Planning)	A mode of the Mission Planning Function in which trajectory-dependent guidance parameters are calculated.
Trajectory Optimization	A mode of the Mission Planning Function in which a direct-search optimization algorithm is used to systematically vary the Trajectory Parameters to minimize a mission-dependent Payoff Function while satisfying all mission constraints.
Trajectory Parameter(s)	Identifier(s) of a trajectory generated in the Preliminary Trajectory Generation Mode of the Mission Planning Function. A trajectory parameter set might include launch time, launch azimuth, parking orbit altitude, engine start times, desired landing site coordinates, etc.
Translator	A ground-support program whose input is a sequence of statements in some language and whose output is an equivalent sequence of statements in another language. For example, translation from FORTRAN statements to symbolic statements.
Transversality Condition	A terminal boundary condition involving both state and co-state (adjoint) variables which is a necessary condition for optimality of a trajectory.
UNIVAR	An optimization algorithm included in DSOP.
Validation (As Used in Mission Planning Function)	A mode of the Mission Planning Function in which the flight program and targeting data are exercised to insure satisfaction of mission objectives and constraints.
Validation	The final verification that a complete software system operating on specified computer equipment functions properly and meets all system specifications.
Variable Point Guidance	See VPG.
V_g	Velocity to be gained; the vector difference between the present velocity and the required velocity. Also, the magnitude of this vector.
VPG	Variable Point Guidance - The mission planning and guidance concept for near-earth rendezvous missions developed by RCA and documented in "Variable Point Guidance Study", SSD-TR-65-100, 15 July 1965. When simulation runs are noted as VPG it denotes the results of an IBM program based on the above reference. The main features of this program (as of February 1, 1967) are:

VPG (Cont'd.)

- The spacecraft (interceptor) starts from a circular parking orbit.
- The target (real or fictitious) is in an arbitrary but not co-planar orbit.
- All burns occur on the Line-of-Nodes except the first burn which is positioned to satisfy the next feature.
- The flight path angle constraint is maintained through all burns (e.g. the flight path angle relative to local horizontal is unchanged).
- Eight options are generated for each input. Four options correspond to rendezvous occurring at a given node, the other four to rendezvous at the opposite node. The four options are Bielliptic Chase and Lob and Full Orbit Phasing Chase and Lob solutions.
- Bielliptic Change denotes that the intermediate burn (which is on the Line-of-Nodes) is inside the target orbit while Bielliptic Lob denotes it is outside it. In the case of Full Orbit Phasing, Chase or Lob denotes if the period of the phasing ellipse is less than or greater than that of the target orbit.
- The optimal plane change split part of the program includes an option (input) of considering a pure plane change maneuver at target altitude.