MASSACHUSETTS INSTITUTE OF TECHNOLOGY

APOLLO

GUIDANCE, NAVIGATION AND CONTROL

Approved David G. Hoag Date 21 Jul 71
D.G. HOAG, DIRECTOR
APOLLO GUIDANCE AND NAVIGATION PROGRAM

Approved Ralph R. Ragan Date 21 Jul 71
R.R. RAGAN, DEPUTY DIRECTOR
THE CHARLES STARK DRAPER LABORATORY

R-700

MIT's ROLE IN PROJECT APOLLO

FINAL REPORT ON CONTRACTS
NAS 9-153 AND NAS 9-4065

VOLUME V

THE SOFTWARE EFFORT

by
Madeline S. Johnson
with
Donald R. Giller

JULY 1971

MIT CHARLES STARK DRAPER LABORATORY

CAMBRIDGE, MASSACHUSETTS, 02139

ACKNOWLEDGMENTS

This report was prepared under DSR Project 55-23890, sponsored by the Manned Spacecraft Center of the National Aeronautics and Space Administration through Contract NAS 9-4065.

The author expresses appreciation to her many colleagues, both within and without the Draper Laboratory, who gave of their time and contributed essential information to this Final Report; these persons are cited in a section entitled, "A Note on Sources". She offers special thanks to Joseph A. Klawsnik, who, though with this effort only briefly, left considerable impact on the format and contents of Sections II and III herein. And finally, she is indebted to Donald R. Giller, whose sensitive editorial judgment and good humor well serve his creative cultivation of words.

The publication of this document does not constitute approval by the National Aeronautics and Space Administration of the findings or conclusions contained herein. It is published for the exchange and stimulation of ideas.

© Copyright by the Massachusetts Institute of Technology
Published by the Charles Stark Draper Laboratory of the Massachusetts Institute of Technology
Printed in Cambridge, Massachusetts, U.S.A., 1970

FOREWORD

The title of these volumes, "MIT's Role in Project Apollo", provides but a modest hint of the enormous range of accomplishments by the staff of this Laboratory on behalf of the Apollo program. Man's rush into spaceflight during the 1960s demanded fertile imagination, bold pragmatism, and creative extensions of existing technologies in a myriad of fields. The achievements in guidance and control for space navigation, however, are second to none for their critical importance in the success of this nation's manned lunar-landing program, for while powerful space vehicles and rockets provide the environment and thrust necessary for space flight, they are intrinsically incapable of controlling or guiding themselves on a mission as complicated and sophisticated as Apollo. The great achievement of this Laboratory was to supply the design for the primary hardware and software necessary to solve the Apollo guidance, navigation and control problem. It is to the credit of the entire team that this hardware and software have performed so dependably throughout the Apollo program.

The quantum leap in technology nurtured by the Apollo program has been and should continue to be of immensely significant benefit to this country—socially, economically and in terms of its national esteem. It is the responsibility of all those who contributed to the proud achievements of Apollo to convince their countrymen of the directions this nation ought to follow in implementing these newly gained—and hard fought for—advances.

C. Stark Draper, President
Charles Stark Draper Laboratory

REPORT R-700

MIT's ROLE IN PROJECT APOLLO

Final Report on Contracts
NAS 9-153 and NAS 9-4065

VOLUME V
THE SOFTWARE EFFORT

ABSTRACT

Seventy-six days after the President of the United States committed the nation to a massive lunar-landing program, the Charles Stark Draper (formerly Instrumentation) Laboratory of the Massachusetts Institute of Technology received the first major contract of the Apollo program. This volume of the Final Report discusses the efforts of Laboratory personnel in developing the specialized software for the Guidance, Navigation and Control System. (The Guidance, Navigation and Control System hardware was also designed at the Draper Laboratory and is reported in other volumes of this Final Report.) Section I presents the historical background of the software effort. Section II discusses the software architecture developed for the Apollo Guidance Computer. Section III treats the methods of testing and verification of the flight programs, and the Laboratory's mission-support activities. Four appendices present functional descriptions of some major program capabilities —coasting-flight navigation, targeting, powered-flight navigation and guidance, and the digital autopilots.

CONTENTS

CONTENTS (Cont.)

CONTENTS (Cont.)

CONTENTS (Cont.)

CONTENTS (Cont.)

CONTENTS (Cont.)

ILLUSTRATIONS

ILLUSTRATIONS (Cont.)

PREFACE

Rarely has mankind been so united as in its awe at one man's step onto the lunar surface. When Neil Armstrong placed his left foot in the dust of the moon, engineers and scientists at the Massachusetts Institute of Technology Instrumentation Laboratory felt a special pride for their significant contribution to this accomplishment in the design of the Primary Guidance, Navigation and Control System for the Apollo spacecrafts.

This volume discusses the efforts of Instrumentation Laboratory personnel in developing the special software for the Guidance, Navigation and Control System. Although it is part of a multi-volume series documenting the total Project Apollo efforts of the Instrumentation Laboratory, this section may be read independently of the other volumes; the author intends it to be meaningful to the general reader who may or may not have read the preceding volumes.

In January 1970, this facility became the Charles Stark Draper Laboratory, named in honor of its founder and current President. Throughout this report, "MIT" and "Draper Laboratory" are used interchangeably, in reference to the former Instrumentation Laboratory.

SECTION I

HISTORY OF THE SOFTWARE EFFORT

1.1 Introduction

Seventy-six days after John Fitzgerald Kennedy committed the United States to participation in a massive lunar-landing program, the Instrumentation Laboratory* of the Massachusetts Institute of Technology received the first major contract of the Apollo program. Steps leading to this award, however, did not begin 25 May 1961—the day of the President's special message to Congress; the footprints of this history trace back at least several years earlier.

In the Fall of 1957, a group of scientists and engineers at MIT began the investigation of a recoverable interplanetary space vehicle. Under contract to the U.S. Air Force, the MIT group collaborated with AVCO Corporation, the Reaction Motors Division of Thiokol Chemical Corporation, and MIT's Lincoln Laboratory. As reported in the MIT/IL document R-235, "A Recoverable Interplanetary Space Probe", this investigation established the feasibility of designing a vehicle which would journey to a neighboring planet, take a high-resolution photograph there, and return for recovery on earth. The investigators studied the navigational techniques and interplanetary orbits which would be required for a variety of such missions. This study served to bring the engineering problems of interplanetary navigation, attitude control, communications, atmospheric reentry, and space exploration into sharp focus. R-235 argued that the "early execution of a recoverable interplanetary space probe is an effective means for advancing the state-of-the-art in self-contained interplanetary navigation and control needed for later scientific and military achievements". Furthermore, the report stressed that the "successful physical recovery of a small vehicle which has navigated itself around the solar system and which brings back photographic evidence of its close and well-controlled passage by another planet is certain to enhance the prestige of this nation".

---

*As explained in the Preface, the Laboratory was renamed the Charles Stark Draper Laboratory in January 1970.

Following the publication of the study in July 1959, the newly established National Aeronautics and Space Administration undertook its first contract with the Draper Laboratory. In September 1959, MIT agreed to investigate guidance and navigation concepts for a variety of interplanetary missions. Placing emphasis on unmanned missions, the Draper Laboratory devised a system for automatic guidance, including the design of an automatic sextant. Upon completion of this contract in March 1960, several months of discussion ensued between representatives of MIT and NASA's Space Task Group, headquartered at Langley Field, Virginia. A second study contract resulted, this one for another six-month effort: MIT was to present a preliminary guidance and navigation design for a manned lunar-landing mission. This study ran concurrently with several industry investigations of the overall Apollo spacecraft mission.

Although work on the preliminary guidance and navigation design for a manned mission began in late 1960, the actual contract was not announced until 7 February 1961. Midway through the contractual period, President Kennedy declared that a manned lunar landing and return would be a national goal for the 1960s. The President's decision opened the way for formal contractual designations by NASA for design, development and manufacture of the various Apollo spacecraft systems.

Thus, by the time of the Presidential message to Congress, the Draper Laboratory had demonstrated scientific and engineering competence in three space studies: the early recoverable space-vehicle investigation; the six-month unmanned guidance and navigation study; and the preliminary manned guidance and navigation examination. Another factor which proved influential in NASA's assessment of MIT's capabilities was the Laboratory's responsibility for the design and development of guidance and navigation systems for the Polaris guided missile. MIT's experience with the U.S. Navy's Polaris project included engineering and managerial techniques which, it appeared, might be implemented during Project Apollo. Indeed, during the month of July 1961, representatives of NASA and the Laboratory studied the development and scheduling of the Polaris guidance and navigation system, from original conception through production. The group plotted a rough schedule for a similar program on Apollo. NASA representatives also expressed interest in MIT's subcontractor philosophy on Polaris: through significant support by subcontractors, the Draper Laboratory had been able to build up a working force and achieve substantial results in a relatively short period of time. Thus, though Project Apollo would undoubtedly prove to be a much larger and more complex task than Polaris, MIT had demonstrated achievement on a qualitatively similar project.

As a result of the preliminary manned guidance and navigation study, NASA's Space Task Group recommended that the guidance and navigation portion of the Apollo spacecraft mission be negotiated as a contract separate from the development of the Apollo spacecraft. Shortly after this decision was made, and following a noncompetitive, sole-source procurement procedure, the Space Task Group designated MIT to implement the guidance and navigation system of the Apollo spacecraft. Announced 9 August 1961, the first major Apollo contract awarded by NASA called on MIT to conduct a Navigation and Guidance System Development Program which would "meet the intermediate as well as the ultimate objectives of Project Apollo", and which would "provide a general on-board guidance capability for the various earth-orbital and cislunar missions".

Although, by the end of 1961, a great deal of theorizing and experimenting had already been accomplished, and the major Apollo spacecraft contractors had been chosen, a significant unknown remained to be answered: how would men actually land on the moon—and equally important, how would they return to earth? The time had come to forecast the amount of rocket power that could be achieved by the end of the decade, to estimate how much weight the lunar surface could actually support, and to devise a means for leaving the moon after a safe landing.

By early 1962, three types of mission plans were being discussed by NASA planners. These methods were called direct ascent, earth-orbit rendezvous (EOR) and lunar-orbit rendezvous (LOR).

The direct-ascent scheme would place a 150,000-lb manned spaceship directly into lunar trajectory, using the boosting power of a still-to-be-developed rocket with an initial thrust of about 12 million lb. From lunar trajectory, the spacecraft would enter lunar orbit; braking rockets would fire and the vehicle would back down toward the lunar surface. The same vehicle would later blast off the surface and land back on earth from an earth orbit. But two problems faced this type of mission. First, there was considerable doubt that the necessary rocket power could be harnessed by 1970. The so-called "Nova" would have required about twice as much power as any rocket then being discussed. Second, planners were concerned that so large a spacecraft might break through the lunar crust—or, indeed, that its high center of gravity (the spacecraft itself would have measured about 90 ft) would cause it to topple upon landing.

A second method of lunar landing and earth return avoided the requirement of so massive an initial rocket thrust. Earth-orbit rendezvous would have placed <u>two</u> payloads in orbit around the earth. First, a "tanker" rocket would be launched,

containing fuel that would eventually be fed into the second payload. After the tanker had achieved its requisite orbit, the second payload would be launched; this would be the manned Apollo spacecraft, propelled by a "Saturn V" rocket whose third stage lacked the liquid-oxygen fuel necessary for the lunar trip. After the payloads had rendezvoused, the spacecraft would dock with the tanker, and the fuel delivery would be accomplished. The advantage of this method was that it involved rocket power then considered likely by the end of the decade. But the same problems of landing on the lunar surface as faced the direct-ascent method still remained.

The third method of lunar landing at first appeared the least likely, probably because it intuitively seemed the most risky. A Saturn V rocket would propel an Apollo vehicle containing three astronauts, plus something new—a detachable craft designed specifically for landing on the moon (e.g., it would possess a low center of gravity and special landing "legs"). After stabilizing in earth orbit, the combined spacecraft and landing vehicle would enter a lunar trajectory and finally stabilize into a lunar orbit. At that point, two astronauts would move into the lunar landing craft, detach it from the mother ship, and descend toward the moon's surface. To rejoin the orbiting Apollo vehicle, the two astronauts would fire rockets for the lunar craft to reinsert into lunar orbit. After the two vehicles had rendezvoused and docked, the astronauts would reenter the main Apollo spacecraft, the landing vehicle would be scuttled, and the Apollo ship would fire its rockets for a return to earth.

The differences between earth-orbit and lunar-orbit rendezvous were immense: EOR plotted a rendezvous in earth orbit before embarking onto a lunar trajectory; LOR involved rendezvous in lunar orbit after the actual landing. The idea of doing a rendezvous (which itself at the time seemed a hazardous maneuver) so far away from earth as planned in the LOR method was initially a frightening proposition. Eventually, however, a team of Langley scientists and engineers demonstrated that, despite outward appearances, LOR would result in substantial savings in earth boost requirements. In addition, it would offer substantial simplification in all phases of a mission—development, testing, manufacture, erection, countdown, launch and flight operations.

With the selection of the lunar-orbit rendezvous method in July 1962, NASA filled in the most significant void then facing the major Apollo contractors. The myriad of scientists and engineers planning for man's eventual landing on the moon could now follow a specific plan. More specifically, the software effort ongoing at MIT at last was able to proceed toward a specific goal. For the most part, conception and development of the Guidance, Navigation and Control hardware did not depend

upon the specific mission plan chosen; software, on the other hand, most assuredly had been hampered by the lack of a definitive goal. Landing on the moon and returning via lunar-orbit rendezvous—this was the Apollo mission; the software effort could now begin in earnest.

1.2 Software Programs for the Apollo Missions

The Draper Laboratory's software efforts culminated in a series of flight programs for the Apollo Primary Guidance, Navigation and Control System. Each flight required its own set of software, defined by the mission objectives and constraints. In general, however, the flight programs were comprised of mission programs and routines which remained rather fixed in approach and technique. Thus, such mission programs as rendezvous, targeting and landing are now part of every lunar-landing flight; their underlying techniques are relatively constant, but, in general, control data change with each mission.

Before work could begin on the first flight program—indeed, even before the Apollo mission had been finalized—basic software techniques had to be developed. Many of these early software efforts are briefly discussed in Section 2.2.1. A completed flight program represents the assembly of mission programs and routines. In common parlance, the completed assembly of hard-wire fixed and erasable memory is known as a "rope", a name taken from the weaving process by which the fixed memory is manufactured; the result of this weaving process actually resembles a rope.

An intriguing aspect of the rope developmental history is the means by which the ropes acquired their given names. At first, virtually all of the rope names derived from their association with the name given the entire lunar-landing mission —Apollo: Greek god of the sun*. Those early ropes without "SUN" in their name generally related to astronomical phenomena: thus, ECLIPSE (developed at the time of a major solar eclipse, in 1963), CORONA and AURORA. (RETREAD was an extensively revised version of SUNRISE.) Assigning the early rope names was the treasured prerogative of those most intimately concerned with each rope's development. After the succession of the "SUN" names given the next ropes—SUNDIAL, SUNSPOT, SUNBURST, SUNDISK and SUNDANCE (and SOLARIUM, with its direct sun association, as well)—it became somewhat difficult to remember which of the

*No satisfactory explanation has yet been offered for naming a project aimed at landing on the moon after the sun god. Apollo's sister, Diana (also called Artemis), goddess of the moon, might well feel offended.

ropes were for the Command Module and which were for the Lunar Module. Accordingly, NASA requested, and MIT agreed, that all Command Module ropes begin with a "C", and all ropes for the Lunar Module with an "L". After a lively intramural competition, the names finally chosen for the LM and CM series were LUMINARY and COLOSSUS, respectively (but not until such names as "Lewis" and "Clark" and "Lemon" and "Coughdrop" had been, for more or less obvious reasons, disqualified).

The following sections summarize the development of flight programs for the Apollo Guidance Computer (AGC). As the result of a NASA decision emanating from a Guidance and Navigation System Implementation Meeting (see Section 1.3.1.2), MIT began digital-autopilot design in late 1964. Two decisions—to integrate an autopilot function into the Guidance, Navigation and Control System, and to enlarge and redesign the AGC—occurred at about the same time, requiring software to fit that computer. Thus, two basic designs of the AGC evolved. Ropes for the earlier, Block I computer, are discussed in Section 1.2.1. The next section discusses the programs developed for the Block II AGC. Section 1.2.3 presents a summary of the Apollo flights, including the names of the flight programs, the launch dates and crews, and flight descriptions. Figure 1.2-1 depicts the interrelationship of the Block I and Block II ropes discussed in Sections 1.2.1 and 1.2.2.

1.2.1 Block I Rope Summary

ECLIPSE is generally ascribed as the first test program designed for use in an early Block I Apollo Guidance Computer. ECLIPSE was, in fact, an assembly of many fundamental routines. It brought together such routines as the Executive, Interpreter and Waitlist. (See Section II for a description of the AGC computer architecture.) In addition, ECLIPSE included Program PINBALL GAME BUTTONS AND LIGHTS, which processes the buttons and illuminates the lights of the spacecraft's Display and Keyboard. Because ECLIPSE was intended only as a test of the Block I AGC, it contained no routines to exercise the Guidance, Navigation and Control System (GN&CS) hardware.

By adding fundamental guidance and navigation functions to ECLIPSE, MIT engineers designed and developed SUNRISE, the first G&N systems-test program for the Block I computer. SUNRISE was the first Block I program suitable for operation in a laboratory-based guidance system. Included in SUNRISE were such G&N-specific routines as an Inertial Measurement Unit mode-switching program, interface-monitoring programs, down telemetry, and routines to measure gyro-drift coefficients and the bias and scale factors of the three accelerometers. SUNRISE

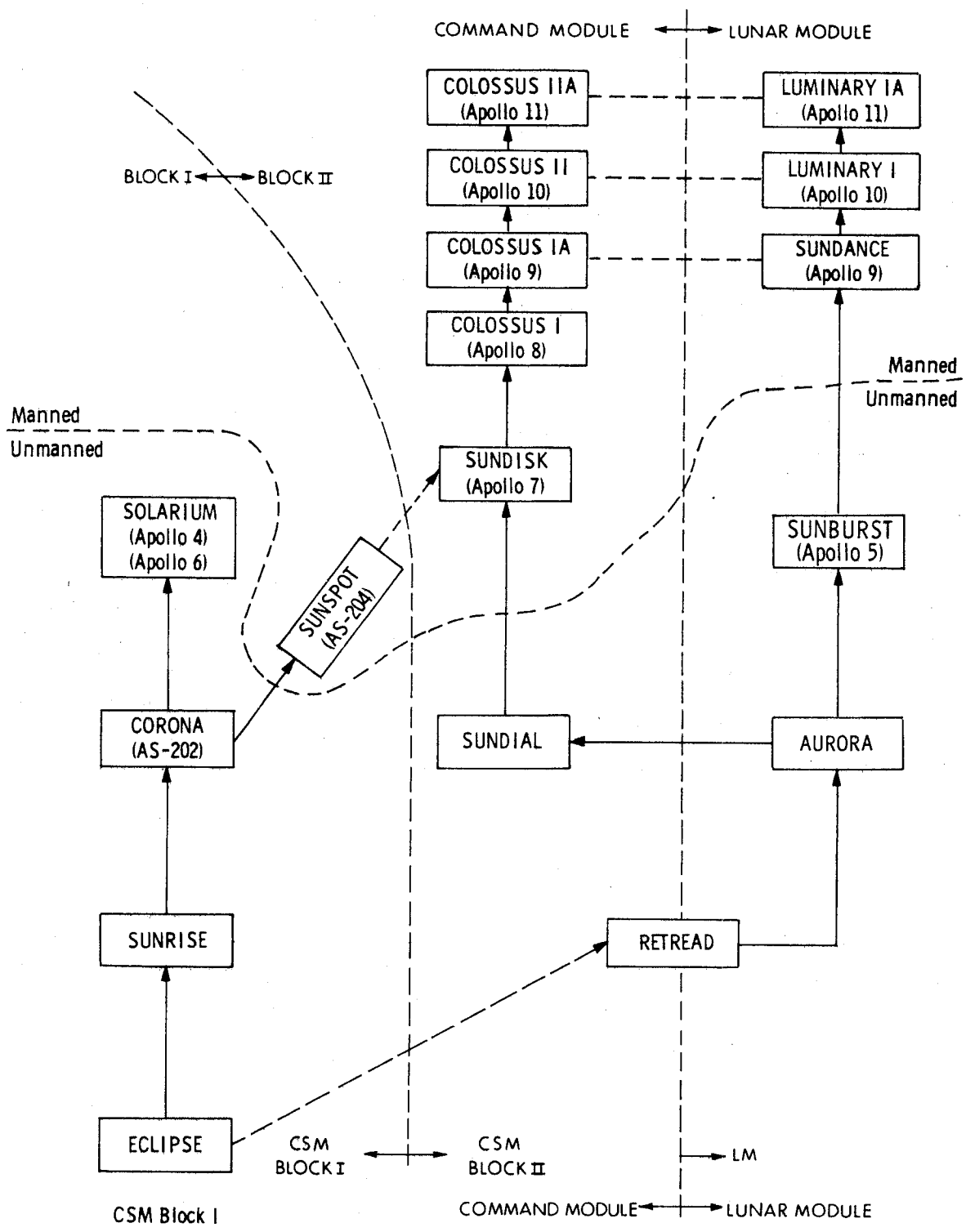


Figure 1.2-1 Rope Tree

also contained a program for prelaunch alignment. Although not destined for an actual mission, SUNRISE served as a building block for the first flight programs that followed. Programs under development could be interfaced with SUNRISE, and thus tested and changed in a working computer environment.

The program designed for the first Apollo flight was known as CORONA; it was used on the unmanned mission, AS-202. CORONA interacted with an onboard Mission Control Programmer, a series of relays connected to the computer interface to simulate certain astronaut functions. Also, CORONA included an earth-orbital reentry program which served as the model for all future such programs.

Two developmental extensions of CORONA occurred at about the same time. The more straightforward evolution led to SOLARIUM, the flight program for the unmanned missions, Apollo 4 and Apollo 6. SOLARIUM contained few major changes from CORONA, except that rescaling occurred to replace the elliptical trajectory of CORONA with parabolic and hyperbolic reentry trajectories for SOLARIUM.

The second evolution from CORONA led to SUNSPOT, the program intended for what would have been the first manned mission, AS-204. The major change represented by SUNSPOT allowed for elaborate astronaut-interface display programs. Whereas programs were sequenced automatically in the previous unmanned missions, allowing only a certain preordained series of events, SUNSPOT introduced the flexibility of astronaut selection of programs. Most of the automatic sequences provided in CORONA were removed in SUNSPOT.

1.2.2 Block II Rope Summary

To a great extent, the development of programs for the Block II Apollo Guidance Computer resembled the path taken in developing the Block I programs (Fig. 1.2-1). The most obvious differences resulted from the added presence of the Lunar Module (LM), which was to contain an Apollo Guidance Computer identical to that in the Command Module (CM). Following the testing of a Block II program which contained basic guidance and navigation functions, programs for the CM and LM computers evolved simultaneously.

For the initial development of a Block II program, the basic Block I systems-test programs were adapted and assembled into the rope appropriately known as RETREAD. Because the Block II computer contained a larger and more powerful instruction repertoire than that of the Block I AGC, recoding of the basic Block I programs resulted in increased speed and efficiency. Analogous to Block I's

ECLIPSE, Block II's RETREAD contained the system-software programs required to test the potential of the computer—Executive, Waitlist, Interpreter. As in ECLIPSE, no provision for mission- or spacecraft-specific programs was included in RETREAD.

From RETREAD evolved the main on-line ropes, beginning with AURORA. In many ways equivalent in purpose to the Block I SUNRISE, AURORA included programs which interfaced with LM GN&CS hardware. AURORA included the monitoring routines for the Inertial Measurement Unit, prelaunch alignment programs, radar-manipulation routines, and various means to control the Display and Keyboard logic, altitude and altitude-rate meters, and the turn-on and turn-off processes. Like the Block I SUNRISE, AURORA provided a software environment for testing and development of future ropes.

As an offshoot from AURORA, a rope called SUNDIAL tested the GN&C System for the Command Module. SUNDIAL naturally resembled AURORA, except that the LM-specific functions of AURORA were replaced with the CM-specific functions of SUNDIAL. SUNDIAL and AURORA both grew out of RETREAD and they "fathered" two lines of flight programs specific, respectively, to the Command and Lunar Module computers.

The first rope for a manned mission using the Block II AGC was SUNDISK, developed for Apollo 7. Although this program was developed for an earth-orbital flight, it contained many translunar programs in their formative stages. COLOSSUS I, the rope for Apollo 8, the first mission to orbit the moon, included operational cislunar and return-to-earth targeting and navigation programs. Apollo 8 orbited the moon without a Lunar Module, however. CSM/LM rendezvous programs were exercised in earth orbit in COLOSSUS IA, the rope developed for the Apollo 9 mission. COLOSSUS II, developed for the Apollo 10 mission, allowed for the first CSM/LM rendezvous in lunar orbit and included a revised model of the lunar-gravity potential. COLOSSUS IIA, flown on Apollo 11, was virtually the same as COLOSSUS II.

Programs for the LM Apollo Guidance Computer evolved from the early AURORA assembly. SUNBURST was developed for Apollo 5, an unmanned flight test of the Lunar Module and its flight rope. The SUNDANCE rope was developed for the first manned Lunar-Module flight, Apollo 9. Although the Apollo 9 mission was strictly earth-orbital, SUNDANCE exercised lunar-landing, lunar-ascent and rendezvous routines for the first time.

Employing the rope LUMINARY I, Apollo 10 marked the first low pass (to 50,000 ft) over the lunar surface by a solo LM. LUMINARY I represented a refinement

of SUNDANCE, and included scaling for the lunar descent. On 20 July 1969, LUMINARY IA finally guided the Lunar Module to its safe touchdown on the moon's surface, thus fulfilling the nation's commitment to a lunar landing in the 1960s.

1.2.3 Overview of the Apollo Flights

Figure 1.2-2 is a summary of the missions flown during Project Apollo, through the flight of Apollo 11. Included are the flight name, the flight program(s) employed, a description of the objectives, the launch date and the crew for each flight. For those flights where two ropes are listed, the first is for the Command Module and the second for the Lunar Module*.

1.3 Control of the Software Effort

This section describes the various means by which MIT's software activities were monitored—internally, through several operating committees; and externally, through formal contact with the customer, NASA. Linking these types of control was the Guidance System Operations Plan—a multi-volumed document that served several functions, including specification control of each succeeding mission flight plan. This document was prepared by the Draper Laboratory for NASA approval, and reflected the technical decisions emanating from internal and external monitoring operations. Section 1.3.1 below discusses external control; Section 1.3.2 describes the Guidance System Operations Plan; and Section 1.3.3 comments upon other types of control, including internal control.

1.3.1 Control by NASA

During the buildup of the Apollo program, NASA issued a number of specifications and contractual documents, few of which had any influence upon the software. One exception, the Interface Control Documents, enforced compatibility between the G&N hardware/software (e.g., scaling and polarity) and the many other external systems with which the G&N system interfaced. Another exception was the Software Requirements Documents, which attempted to establish all of the operational requirements necessary to accomplish the flight-test objectives of the Apollo mission.

Most of NASA's control of MIT's software activities occurred in the form of regular series of meetings conducted among representatives of NASA, MIT, North

*For an insight into all of the phases which comprise a lunar-landing mission, the reader may choose at this time to continue with Section 2.2.2.

Flight	Flight Program Name	Description	Launch Date	Crew
AS-202	CORONA	Suborbital; supercircular entry with high heat load	8-25-66	unmanned
Apollo 4	SOLARIUM	High apogee; suborbital; supercircular entry at lunar return velocity	11-9-67	unmanned
Apollo 5	SUNBURST	First Lunar Module flight; earth orbital	1-22-68	unmanned
Apollo 6	SOLARIUM	High apogee; suborbital; supercircular entry at lunar return velocity; verification of closed-loop emergency detection system	3-4-68	unmanned
Apollo 7	SUNDISK	First manned Apollo flight; earth orbital	10-11-68	Schirra Eisele Cunningham
Apollo 8	COLOSSUS I	First manned lunar-orbital flight; first manned Saturn V launch	12-21-68	Borman Lovell Anders
Apollo 9	COLOSSUS IA SUNDANCE	First manned Lunar Module flight; exercise of lunar landing, ascent and rendezvous techniques in earth orbit; EVA (Extra Vehicular Activity)	3-3-69	McDivitt Scott Schweikart
Apollo 10	COLOSSUS II LUMINARY I	First lunar-orbit rendezvous; Lunar descent to 50,000 ft	5-18-69	Stafford Young Cernan
Apollo 11	COLOSSUS IIA LUMINARY IA	First lunar landing (7-20-69)	7-16-69	Armstrong Aldrin Collins

Figure 1.2-2 The Apollo Flights

American Rockwell, Grumman Aircraft Engineering Corporation, and other relevant contractors and subcontractors. These meetings served as a vehicle for communications among the prime contractors and the customer, and apparent conflicts were often settled through unhampered discourse. When contractors were unable to agree on technical issues or future directions, NASA would often use the forum of these meetings to issue its decisions on such matters.

1.3.1.1 G&N System Panel Meetings

The earliest series of discussions was known as G&N System Panel Meetings. This series occurred from August 1962 through February 1964, under the direction of the Apollo Systems Project Office of NASA/MSC. Participants represented NASA, MIT, North American Rockwell, Grumman and Bellcomm. Throughout this period, three subseries of Panel Meetings met regularly, each focusing on a separate issue: lunar-orbit operations of the Lunar and Command Modules; earth-orbit and cislunar activities of both vehicles; and the reentry activities of the CSM. Through the medium of vigorous discussion and debate, these meetings collated the technical decisions being made in the design and development of the Guidance, Navigation and Control System.

1.3.1.2 G&N System Implementation Meetings

The next set of meetings served to define the required interfaces between the GN&C System and the spacecraft. The Guidance and Navigation System Implementation Meetings were a means of negotiating the Interface Control Documents (ICDs) which were binding upon all contractors. Implementation Meetings focusing on interfaces for the CSM occurred from June 1964 through February 1965. Implementation Meetings responsible for LM interfaces occurred from September 1964 through April 1966. In addition to physical interfaces, among the topics discussed were kinds of data being sent across the interfaces; the formating of data transmission; data rates; and accuracies of data.

The Implementation Meetings monitored the integration of guidance, navigation and control. Out of these discussions came a decision which had a major impact on MIT's Apollo responsibilities. Originally, the Apollo Command Module autopilot function had been the responsibility of the Honeywell Corporation, under subcontract to North American Rockwell. The Honeywell autopilot was analog and was deemed by the NASA monitors to lack the flexibility and versatility required for the complex Apollo mission plan. Consequently, NASA directed the Draper Laboratory to develop a digital autopilot which would have none of these limitations. The existing Block I

computer hardware did not have sufficient storage capacity to accommodate an addition of such import; however, at about this same period, another significant decision was made to enlarge the computer capacity and at the same time make its computer architecture more powerful than had heretofore been possible. Therefore, through the forum of the G&N System Implementation Meetings, the Block II Apollo Guidance Computer and the digital autopilots were conceived.

1.3.1.3 Data Priority Meetings

As a means of relieving the problem of customer-contractor and inter-contractor communications, the concept of Data Priority Meetings emerged in 1967. The Planning and Analysis Division of NASA/MSC regularly gathered together the flight crews, flight directors, flight controllers, various MSC software, hardware and analytical specialists, and appropriate contractor representatives. There were thus brought into a single room three significant components: those with questions; those with answers; and those with authority to render decisions.

The group meticulously reviewed the guidance and control details for each succeeding mission. Data Priority Meetings defined how the various data could be used and the priority which could be imposed to effect the nominal and backup execution of each mission phase.

MIT's role was restricted to the Guidance, Navigation and Control System, but this was one of the most complex subsystems in the Apollo spacecraft, interacting with most other spacecraft systems. MIT's representatives to the Data Priority Meetings oversaw the Laboratory's follow-up to each meeting. Questions arising from these meetings elicited formal responses, usually in the form of Mission Techniques Memoes.

1.3.1.4 "Tiger" Teams

A fourth type of NASA control of MIT's software activities occurred through a means less formal than that of an organized meeting. In late 1967, the Flight Operations Directorate of NASA/MSC organized so-called "Tiger" Teams to hasten technical decisions on MIT's rendezvous and display techniques. The Tiger Teams were aptly named, for despite their relatively informal approach, they were extremely effective. The first Tiger Team spent several days in Cambridge in a successful attempt to clarify the rendezvous displays and operations. Display interfaces between

the crew and the landing and rendezvous maneuvers were determined, and rendezvous-display compatibility (e.g., scaling, polarity) between the LM and the CM were established. Targeting programs were made consistent from one program to the next. The second Tiger Team addressed itself to the same issues, but since the decisions of greatest import had already been made, its impact was less pervasive —hence, this Tiger Team was dubbed the "Pussycat" Team.

1.3.1.5 "Black Friday" Meetings

Shortly after MIT evidenced its dismay over the rapidly-saturating fixed-memory storage capacity of the AGC, joint MIT/NASA meetings were held to purge the mission programs then under development of any routines deemed "nonessential". Three such meetings took place—on 13 May 1966, 13 January 1967 and 28 August 1967. These meetings became emotional because of disagreement about what was, in fact, nonessential. Nonetheless, difficult compromises resulted in the then-current fixed-storage usage being reduced sufficiently to allow inclusion of every essential routine.

1.3.2 GSOP Concept and History

Beginning with CORONA, the computer program for the AS-202 mission, a document known as the Guidance System Operations Plan (GSOP) served as the specification toward which the software efforts were directed. Development and control of the GSOP were important activities in planning the release of a flight program. The format for the GSOP evolved through a series of discussions among key personnel at NASA and the Draper Laboratory. During preparation of the CORONA rope, several alternative mission profiles had been considered: orbital, short-ranged suborbital, and long-ranged suborbital. MIT provided NASA with estimates of navigational difficulty that might be encountered on each type of mission, whereupon NASA chose the long-ranged suborbital trajectory. The CORONA GSOP represented an integration of inputs from MIT, NASA and North American Rockwell (the manufacturer of the CSM spacecraft), further defining the mechanics of achieving such a trajectory. NASA reviewed the document, modified it where necessary, and finally approved it as the specification for MIT's software effort.

In comparison to the GSOP format which would follow, the AS-202 document was relatively informal, encompassing in one small volume the same type of information which would later require six separate volumes for each rope. The CORONA GSOP discussed the general description of the mission, the logic diagrams defining the operation of the Apollo Guidance Computer, the uplink and downlink

that would interface with the guidance system, and the guidance equations and routines which MIT considered of potential interest to NASA.

Further evolution of the GSOP structure resulted from the additional requirements, constraints and capabilities of later missions. For instance, the SUNSPOT rope developed for the AS-204 mission was the first to allow for manned Apollo flight. With astronauts involved for the first time, more time was required for the GSOP discussions, and more personnel participated in the GSOP development. SUNBURST, the rope for the Apollo 5 flight, contained the first routines developed specifically for the Lunar Module, and thus the GSOP for SUNBURST was the first of the LM GSOPs. Beginning with SUNDISK (Apollo 7, CSM) and SUNDANCE (Apollo 9, LM), successive GSOPs generally represented merely changes from the preceding version, and did not require a completely fresh start. Most of the GSOP effort for later missions involved only minor changes, with relatively little rewriting.

As mentioned above, the GSOP is published separately for the Lunar Module and the Command Module, and is updated with each new program release, thus providing NASA with current and accurate control over the software and system operations. In addition to these functions, the GSOP has served as an internal working document to coordinate the efforts of the various MIT groups, and as a testing guide for simulation personnel. Finally, the GSOP served as a GN&C software description and a crew training aid for MSC personnel and contractors. A more detailed description of the GSOP is contained in Section III.

1.3.3 Additional Software Control

The Draper Laboratory monitored the incorporation of mission-program requirements into the mission programs through the actions of a Mission Design Review Board (MDRB), a formally-constituted group comprised of the directors of all software groups. Under the direction of each rope's Project Manager, the MDRB approved, internally, all mission-related documentation. The Project Manager was charged with the responsibility for MDRB coordination and participation to ensure proper processing of control documentation. The specific function of the MDRB was to provide a mechanism for internal control and coordination of mission-related activities. Program Change Requests (PCRs) and Program Change Notices (PCNs) were used as interim revisions of the GSOP, and to document departures from the published GSOP until such a time as MSC-approved changes were incorporated in official GSOP revisions. A NASA-comprised group known as the Software Control Board (with representatives of MIT) initiated and approved specific change concepts, whereupon the MDRB would monitor MIT compliance with these changes.

1.4 Man and Machine Loading Requirements

The story of Project Apollo's successful completion represents, in the end, a myriad of individual successes, most of which were based upon an intricately-tuned interaction among men and machines. For its own part, the story of MIT's software-development effort demonstrates the essential interdependence of talented scientists, engineers, mathematicians and technicians with increasingly complex, versatile and powerful computing equipment. As the tempo of the Laboratory's involvement in software tasks changed, these changes were reflected in the number and types of personnel participating in the effort, and in the power and speed of computers which the Laboratory acquired. This section discusses, in general terms, the history and philosophy of MIT's personnel and computing requirements.

1.4.1 History of Man Loading

1.4.1.1 Initial Philosophy

At the beginning of the Laboratory's participation in Project Apollo, a simple philosophy guided the staffing of the software-development group. Essentially, this philosophy placed a premium on engineers and scientists who, in addition to original, conceptual work, would put their own ideas into a form which machines could understand. Thus, in the early days of the Apollo work, there were no "programmers", as such. Instead, engineers and scientists learned the techniques of programming. At this stage, a relatively small group was thought capable of handling what was then considered a practicable task. It was believed that competent engineers with a credible, solid mathematical background could learn computer programming much more easily than programmers could learn the engineering aspects of the effort. The small size of the initial staff dictated that integration of engineering and programming talents in a few individuals would be preferable to attempts at intercommunication by individual engineers and programmers. Thus, the original intent was to have the project's basic core of engineers follow the program through, from conception to actual flight support.

With the passage of time, however, it became clear that the philosophy could best be followed in spirit, rather than in letter. As desirable as it might be to have a staff composed solely of multidisciplinary personnel, it was clearly impossible to shape such a staff beyond a certain size. Individuals talented in both engineering and in computer programming were not readily available. Also, as the software tasks became better delineated, it was apparent that a major underestimation of program-testing requirements had initially occurred. Because the Apollo Guidance

Computer has a comparatively small erasable memory, the problem of having various people using the same registers for different tasks, the problem of overlaying memory —these all required extensive precautionary measures to avoid conflict. Optimally, one dedicated engineer/programmer assumed responsibility for ensuring that no erasable-memory conflict occurred, and for integrating the individual flight programs.

1.4.1.2 Creative Use of Subcontractors

Part of the solution to the problems discussed in the preceding section developed through the extensive use of subcontracted personnel. From the very beginning of MIT's participation in Project Apollo, the Laboratory had stressed that its frequent and extensive use of subcontractors would allow it manpower leverage essential to its responsibilities under the Apollo contract. Through the use of subcontracted personnel, the Laboratory would not be required to assemble and disassemble its own staff to meet the time-varying responsibilities of the Apollo program. Subcontractors would serve as a buffer for the Laboratory's staffing requirements. Importantly, Draper Laboratory personnel have traditionally enjoyed the benefits of long-term employment, so the use of subcontractors would permit Laboratory management to carry a mainline staff of a size that would assure maximum security to all personnel. As detailed in Section 1.1, MIT's extensive hiring of subcontractors during the Polaris project had been a strong point in its presentation to NASA in advance of the Apollo program. Thus, when it became apparent that work loads were greater than initially estimated—especially in the areas of testing and verification—subcontracted personnel were made available for virtually immediate deployment.

Throughout MIT's participation in Project Apollo, subcontractors have served in a variety of roles. They have provided a complement to the talents of the Laboratory's own staff. Except in the area of direct administration, subcontractors have played parts in virtually every phase of the software effort, including design, analysis, testing, verification and simulation. Perhaps most significantly, the ready availability of subcontracted personnel facilitated quick solutions to unexpected personnel requirements, since the Laboratory could hire such personnel without necessarily promising any long-term commitment. The costs—direct and indirect —relating to in-house staffing levels were therefore kept to a mimimum throughout.

1.4.1.3 Review of Man Loading

Figure 1.4-1 depicts the man-loading history of the Apollo program at MIT from September 1961 through March 1970. As well as containing a curve for the

total personnel levels, the figure shows separate breakdowns for subcontracted hardware and software and total hardware and software levels.

Inclusion of the hardware-personnel figures demonstrates the relative personnel requirements for the hardware and software tasks under MIT's Apollo contract. Thus, the project manpower resources were concentrated on developing system hardware from 1961 through 1965. In 1966, this hardware-development effort rapidly tapered off, and the requirements for designing and developing the mission computer programs increased. In November 1966, the software effort captured precedence as the primary task of the Laboratory's Apollo division.

Figure 1.4-2 demonstrates some of the reasons for the rapid buildup of software personnel. In 1966, no fewer than five ropes were being developed at one time. During the following year, when much of the software buildup had already occurred, six ropes were worked on simultaneously. Figure 1.4-2 is also a milestone chart of the many decisions and events germane to the Apollo software efforts of MIT.

1.4.2 History of Digital Machine Loading

Digital-computation facilities have played a significant role in MIT's development of software for the Primary Guidance, Navigation and Control System of the Apollo spacecraft. As will be discussed in Sections II and III, digital computers were used in simulating the Apollo Guidance Computer during design, verification and testing of software. (In addition to this so-called "All-Digital Simulator" function of the digital computer, it served as a basis for the Engineering Simulator, also described in Section III. A Hybrid Simulator and a Systems Test Laboratory also assisted in the test and verification of computer software and are also discussed in Section III.) This section discusses, in chronological sequence, the four types of digital computers around which the Draper Laboratory fashioned its digital-computation facilities. These computers are the IBM 650, the Honeywell 800, two Honeywell 1800s, and two IBM 360/75s.

During the period in which MIT has participated in the Apollo program, the computing facilities described in this section have served other Draper Laboratory groups in addition to the Apollo division. However, Apollo activities have accounted for about 90 percent of the total use of these facilities.

1.4.2.1 IBM 650

When the Draper Laboratory received its first contract from NASA, in September 1959, an IBM 650 provided the Laboratory with its in-house computing

also prevents the load factor from exceeding 8g. When the relative velocity decreases to 1000 ft/sec, the guidance ceases to generate new steering commands, but maintains the most recent value for the CM terminal roll attitude. P67 is terminated at crew option following drogue-chute deployment, although for telemetering coverage, termination is delayed as long as possible.

p 174

C.5 Lunar-Landing Guidance and Navigation

One of the most important phases of the ApQllO mission is the guidance and navigation of the Lunar Module during the deceleration maneuvers prior to touchdown on the lunar surface. This section discusses the guidance and navigation capabilities onboard the LM for this powered-landing phase, which is presented in the context of the complete mission in Section 2.2.2.7.

The basic function of the LM guidance and navigation systems during powered landing is to take the spacecraft from a nominal initial altitude of about 50,000 ft and a velocity of approximately 5600 ft/sec and safely land it at an assigned site on the moon with virtually zero touchdown velocity. Several condition-s and constraints govern the means by which the powered landing is accomplished:

a. - The Descent Propulsion System (DPS) propellant must be utilized in an efficient manner, i.e., the required velocity increment should be as small as possible.

b. The selected landing site must be visible to the astronaut through the window of the LM for a tIme interval of at least 75 sec immediately prior to touchdown.

c. New state-vector estimates and steering commands for the LM cannot be obtained more frequently than once every 2 sec.

d. The DPS must be operated either at a fixed high-throttle setting (close to maximum thrust) or as a continuously-throttle able engine over a limited range of lower throttle settings-with the direction of applied thrust essentially parallel to the longitudinal axis of the LM.

e. The astronaut must have the capability of manually redesignating the landing site during the interval when the site is visible. (Conditions prevailing at 500-ft altitude must be "comfortable'~ such that manual takeover can be accomplished with ease.)

The navigation and guidance systems each perform different functions during ~wered landing maneuvey, as indicated in Fig. C,5-l. The navigation system ~ly determines (estimates) the state of the vehicle, i.e., its position and velocity. ~idance system uses the navigation information to compute specific-force ands £or use in steering the vehicle. The

following two parts of this section Ibe the guidance and navigation systems which enable the LM to accomplish tted objectives during the powered-landing maneuver.

Guidance System Description

~he mission requirements of efficient fuel utilization and a 75-sec landing- site ity interval during the powered-landing maneuver are in direct conflict. For el management during the powered maneuver, the Descent Propulsion System be operated at the largest permissible thrust level, with the thrust direction y above the local horizontal. For the astronaut to have visibility of the landing ough the LM window, however, the longitudinal axis of thevehicle (and hence, ust direction) must be in a nearly vertical orientation. Continuous throttle is required during the final part of the landing maneuver, moreover, to ly shape the trajectory (to meet visibility requirements) and to achieve the i terminal conditions (position and velocity) at touchdown.

:~ accomplish these objectives in an efficient manner, the landing maneuver led into three major phases, as indicated in Fig. C.5-2. For convenience, hree phases are referred to as the "braking" and "visibility" phases, and minal-descent (hover) maneuver. The major deceleration of the vehicle is 'lished during the braking phase, which is typically about 470 sec in duration. aking phase is the longest with respect to time and range covered. The)bjective of this phase is to establish the desired initial altitude and velocity)ns for the Tollowing visibility phase within efficient propellant usage limits; ~ of the ~pacecraft 1s orbital velocity is removed during the braking phase. bility (approach) phase is designed such that the commanded vehicle attitude ed engine-throttle setting allows the astronaut to view the landing area for t time. During this second phase the astronaut can-if he so desires-riate the landing site toward which the GN&C System is controlling the Lunar

A further objective of the visibility phase is to establish and maintain a ry from which the astronaut can easily take over control when he desires. the third and final terminal-descent phase, the astronaut manually controls through the final vertical descent to lunar touchdown. For convenience, the

*

braking-phase terminal point is referred to as the "High-Gate" point, and the visibility-phase terminal point is called the "Low-Gate" point. The terminal point for the hover is, of course, the landing site.

The guidance system solves the powered-landing guidance problem essentially as successive two-point boundary-value guidance problems. First of all, the guidance system executes an algorithm which determines the exact time and spacecraft attitude required at ignition. Ignition nominally occurs at the periapsis of the prior orbit. The spacecraft is maneuvered to that desired attitude and, 35 sec prior to lighting the engine, the DSKY is blanked for five sec to notify the astronaut that everything is proceeding smoothly. Ullage is commanded Tor 75 sec prior to ignition-the beginning of the braking phase. Fur 26 sec the engine operates at minimum thrust, about 12 percent of the engine's rated thrust of 10,500 pounds.

With sufficient propellant, the guidance can permit substantial landing-site redesignation; as much as 7000 ft forward or to either side, or about 4500 ft backward. The guidance system requires no fixed landing site and indeed provides a relatively gentle standard approach to Low Gate whether the landing site has been redesignated or not.

Nominally, the descent trajectory is planar; however, redesignation results in a nonplanar trajectory. The descent trajectory during both the hr aking and visibility phases is provided by a vector polynominal which determines a three-dimensional line in space.

During almost the entire visibility phase, the guidance system maintains spacecraft attitude so that the landing site nearly coincides with the reticle (Landing Point Designator) etched on the Lunar Module window; the computer displays on the DSKY a number which informs the astronaut where along this reticle to look to see where the computer is taking him.

*

The "target" for the braking phase lies very near the lunar surface, projected forward about 62 sec past High Gate, but the phase actually ends at High Gate. The reason for avoiding the designated target (other than the obvious intersection with the lunar terrain) is that as time approaches zero with the guidance equations, the gain of the guidance equations approaches infinity. To avoid that gain variation, the targets of the braking phase are projected about a minute downstream from the

phase lies some 10 sec beyond Low Gate.
desired braking-phase terminus. For the same reason, the target for the visibility

When the Low-Gate aim conditions have been met, the Lunar Module begins the so-called terminal-descent maneuver, which is nearly vertical. The point where this maneuver is started is dependent upon mission ground rules, crew option, and the erasable load provided for the specific mission.

The terminal-descent program in Apollo 11 and 12 automatically nulled the horizontal components of velocity and provided a 3 ft/sec rate of descent. In Apollo 13 and subsequent Tlights, the astronaut must specify the rate of descent by means of his rate-of-descent (ROD) controller; there is also an attitude-hold mode in which spacecraft attitude (and hence horizontal velocity) can be manually controlled-not necessarily nulled-to provide the desired translation across the lunar terrain.

The fuel allotment of the DPS provides for hovering immediately prior to ~ouchdown. If the spacecraft does not enter this terminal descent phase, an abort ~s initiated on the ascent stage (see the next section of this Appendix).

Thrust magnitude and orientation time histories are shown in Fig. C.5-3 for ~ typical simulated landing trajectory. As can be seen, the throttle is operated at ~aximum thrust for most of the braking phase. During this period the DPS is

)riented along the direction of the command specific force even though, due to

imitations , the throttle cannot follow the thrust command. The DPS is operated

hereafter in the continuously-throttleable range between 12 and 58 percent of ~aximum thrust.

.5.2 Navigation System Description

The navigation concept used by the LM GN &CS controls the braking and visibility)hases of the lunar-landing maneuver and then continues to provide altitude and ~elocity data to the astronaut in the final hover phase. This approach employs an nertial navigation system updated by a Doppler landing radar. The first three to our minutes of the braking phase is completely controlled by the inertial navigation ystem. When the altitude has decreased to between 40,000 and 30,OQO ft, as llustrated in Fig. C.5-2 for the Apollo 11 mission, the landing radar is activated Lnd initial altitude updates are accepted by the navigation system. When the velocity as decreased to less than 2000 ft/sec, landing-radar velocity corrections are next ncorporated. It is important that the landing radar altitude and velocity updates

Due to a hard mechanical stop, the engine is incapable of delivering more than 94 ercent of the rated thrust. ~e achieved during the braking phase for two reasons: first, to guarantee a safe ~ltitude condition at the start of the approach phase, and second, to make the major rajectory changes before the visibility phase, so that the commanded vehicle altitude vill be relatively free of control transients during this second phase, thus allowing he astronaut to visually determine and evaluate the landing area and have time to edesignate the landing site, if necessary. The visibility phase lasts only 100 sec ~r less, so major navigation-update changes resulting in large commanded vehicle Lttitude changes are undesirable.

The landing-radar sensor used to update the inertial navigation system during Lie landing maneuver is a four-beam Doppler radar with the beam configuration hown in Fig. C.5-4. Three beams are used for velocity determination and one for ange. The landing-radar antenna is mounted at the base of the LM descent stage nd can be oriented in one of two fixed positions. The first antenna position is sed during the braking phase when the ve~icle attitude is essentially horizontal, nd the second is used in the approach and landing phases as the LM orientation ears a vertical attitude.

The navigation problem during the powered landing is significantly different

om that during the coasting cislunar and rendezvous navigation phases discussed

1 Appendix A. In addition to gravitational forces, a large thrust acceleration acts

~ the vehicle at all times during the landing maneuver. Another significant difference that the guidance system controlling the vehicle thrust and attitude continuously ~es navigation data throughout the maneuver, as opposed to the intermittent updates aneuvers typical for cislunar and rendezvous phases. Despite these major fferences, the general navigation concept used during landing is very similar to at used during coasting phases.

A simplified landing navigation and control functional diagram is illustrated Fig. C.5-5. During the landing maneuver, the inertial measurement unit is active all times and provides the specific force data necessary for extrapolation of the ate-vector estimates. During the initial part of the braking phase, the IMU is the ly navigation sensor employed, and the estimated vehicle-state vector is used to 'mm and required vehicle attitudes and and engine throttle levels so that the desired rminal -conditions for the mission phase are achieved. When the landing radar is tivated later in the braking phase, the vehicle altitude above the lunar surface is timated in the computer and then compared with the landing-radar measured range ta converted to altitude. The difference between these two parameters is tomatically checked in a landing radar (LR) data test to verify that the LR is crating normally. If the data check satisfactorily, a correction term is computed

*

by applying a precomputed weighting factor to the altitude difference , as shown in Fig. C.5-5. This correction is then used to update the state-vector estimate and to correct the landing trajectory in the next guidance-equation computation cycle. After velocity lock-on is achieved by the LR at a lower altitude and velocity, a similar operation is conducted for velocity updates to the state vector. When both altitude and velocity updating are being done, the~avigation-measurement data from the LR are controlled by the onboard computer. Altitude is updated every two sec, and each component of velocity every six sec.

With reference to Fig. C.5-5, it should be noted that LR measurement data are used to update only four of the six components of the estimated state vector, i.e., altitude and the three velocity components. The down-range and cross-track position estimation errors are not updated by LR measurements. The astronaut can correct these two horizontal-plane position errors during the visibility phase by incrementally retargeting the guidance equations. In this operation, the landing site to which the GN&CS is controlling the vehicle is displayed to the astronaut by a DSKY number referenced to a grid pattern on the LM window. As previously mentioned, it is important that the major altitude and velocity corrections to the state vector be completed before the visibility phase so that the astronaut can effectively assess the landing area and correct the cross-track and down-range deviations, if required.

The weighting functions used to compute the state-vector ~pdates in Fig. C.5-5 are significantly different from the time-varying statistical weighting functions used in the cislunar and rendezvous navigation phases. These powered-landing weighting functions are linear approximations to the statistical optimum navigation-filter weighting functions based upon inertial and LR sensor accuracies, lunar terrain uncertainties, and measurement bias errors. Since computation time is a critical parameter during the landing maneuver, the LR weighting functions are

precomputed and stored. The altitude weighting function is a linear Tunction relative to estimated altitude. Since altitude updates are typically started some 90 nmi from the landing site, lunar terrain altitude variations relative to the landing site were important factor in shaping this function. The velocity weighting function is linearized with respect to estimated velocity; it is truncated to fixed small values at the low-velocity conditions to minimize commanded attitude changes during the more critical terminal phases of the landing maneuver, and to avoid incorpoTating large LR velocityupdates

*

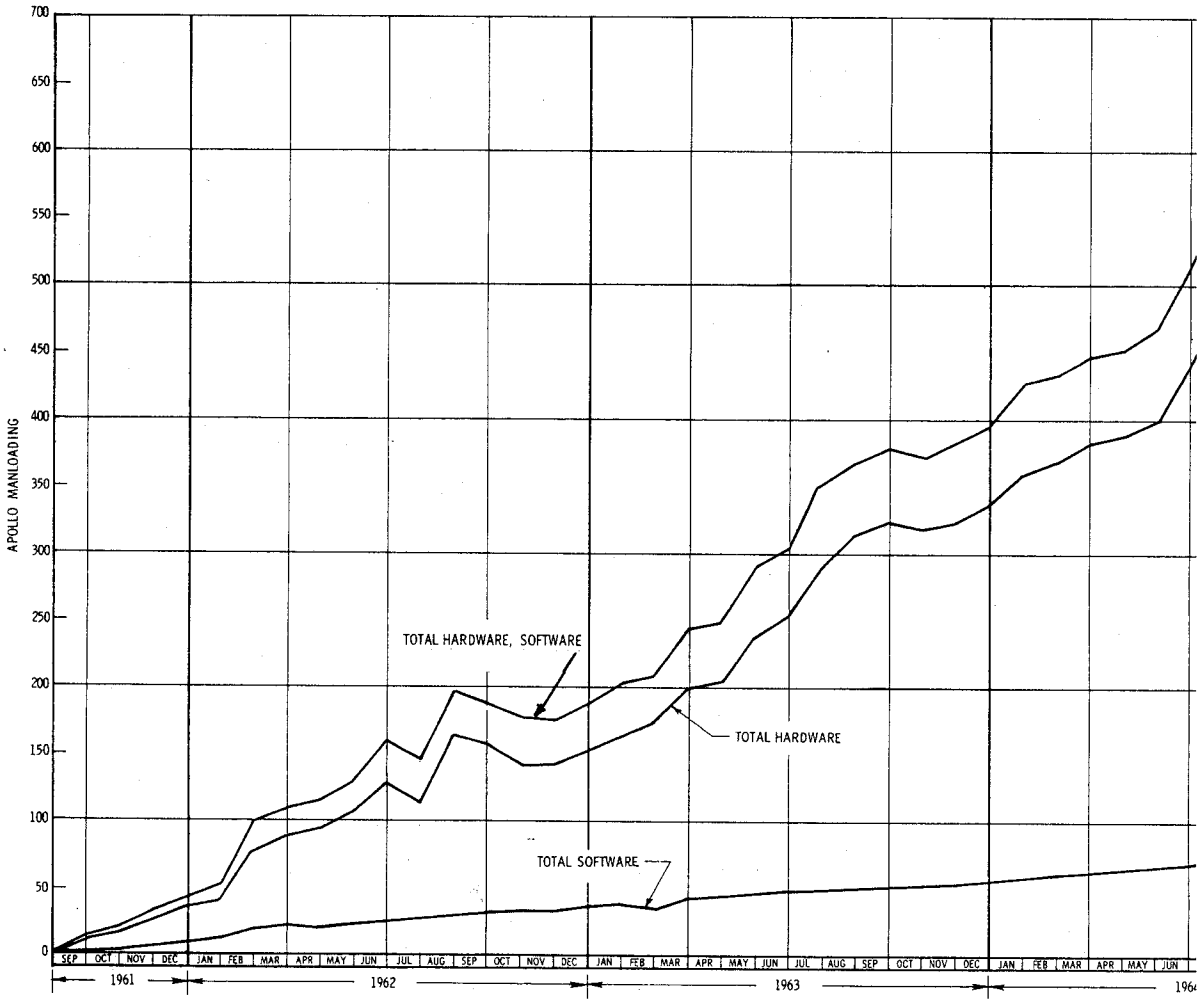
If the LR data test is failed a given consecutive number of times, the DSKY alerts the astronaut to this fact; he can force acceptance of the rejected data should he desire, but this option would not normally be used.

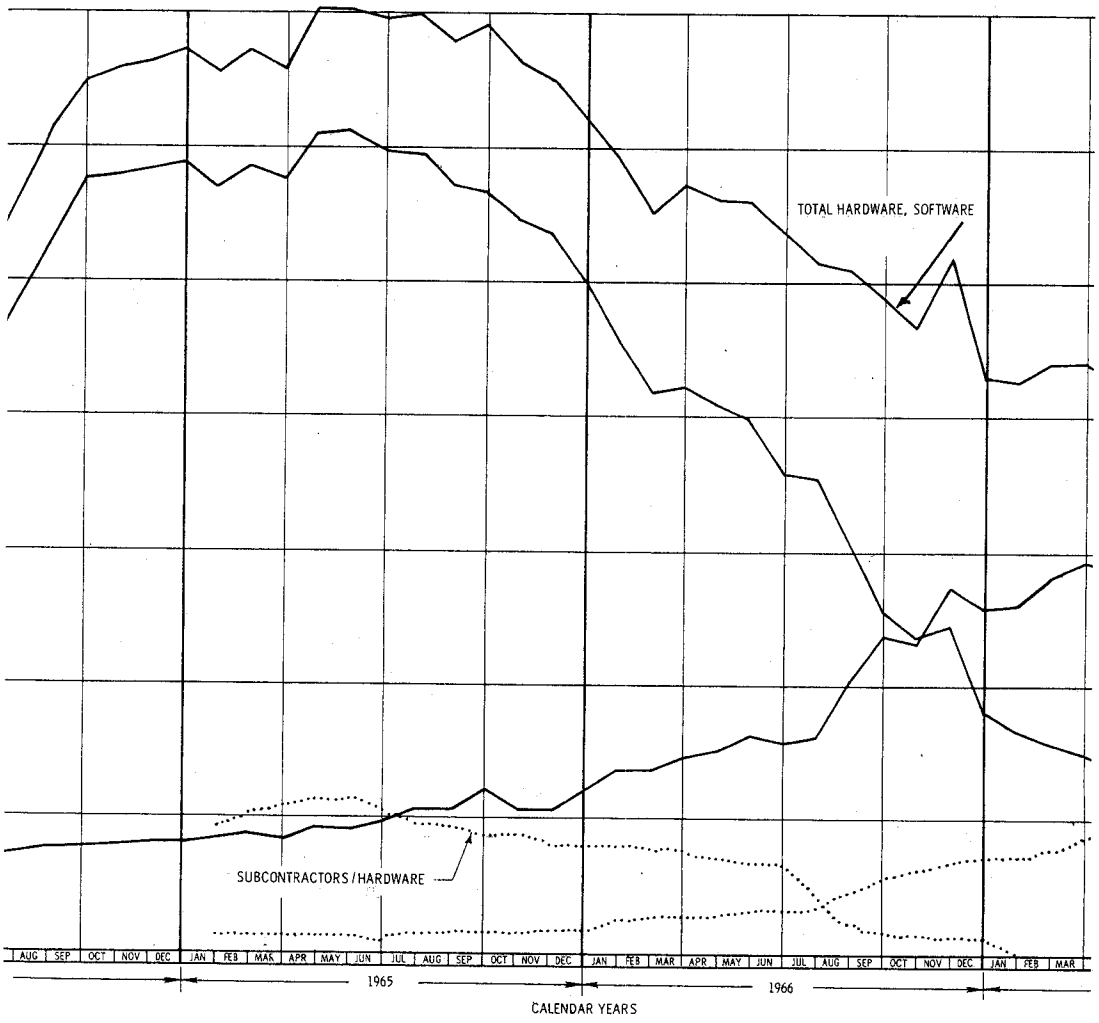
n the velocity region where LR dropouts can occur due to near-zero velocity onditions along various radar beams. The LR measurement weighting functions re uncoupled or noncorrelated with any measurement direction other than that along ihich the LR measurem-ent is being made. Whereas correlation in the navigation-up-late weighting functions is very important in the cislunar and rendezvous navigation oncepts, it is intentionally avoided in the landing maneuver navigation since rnplementation of such a correlation function was not considered practical due to ~odeling uncertainties and G&N system computation time limitations in the landing ~ aneuver.

Several important differences exist between the operations of power-landing avigation and those of cislunar and rendezvous navigation. First, the state-vector pdating procedure and monitoring in powered landing is completely automatic, since ~e navigation is time-critical. Next, the navigation-measurement weighting function sed for the landing navigation is a precomputed linearized approximation to a tatistically optimum weighting function, and is uncorrelated between state-vector omponents. Finally, the landing-navigation function updates only four of the six tate-vector components, and the astronaut is required to manually correct the emaining horizontal position deviations during the final two phases of the landing laneuver.

p185

.


APOLLO MANLOADING
700
650
600
550
500
450
400
350
300
250
200
150
100
50
0
TOTAL HARDWARE, SOFTWARE
TOTAL HARDWARE
TOTAL SOFTWARE
SEP OCT NOV DEC JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC JAN FEB MAR APR MAY JUN
1961
1962
1963
1964


TOTAL HARDWARE, SOFTWARE
SUBCONTRACTORS / HARDWARE
AUG SEP OCT NOV DEC JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC JAN FEB MAR APR MAY JUN JUL AUG SEP OCT NOV DEC JAN FEB MAR
1965
1966
CALENDAR YEARS

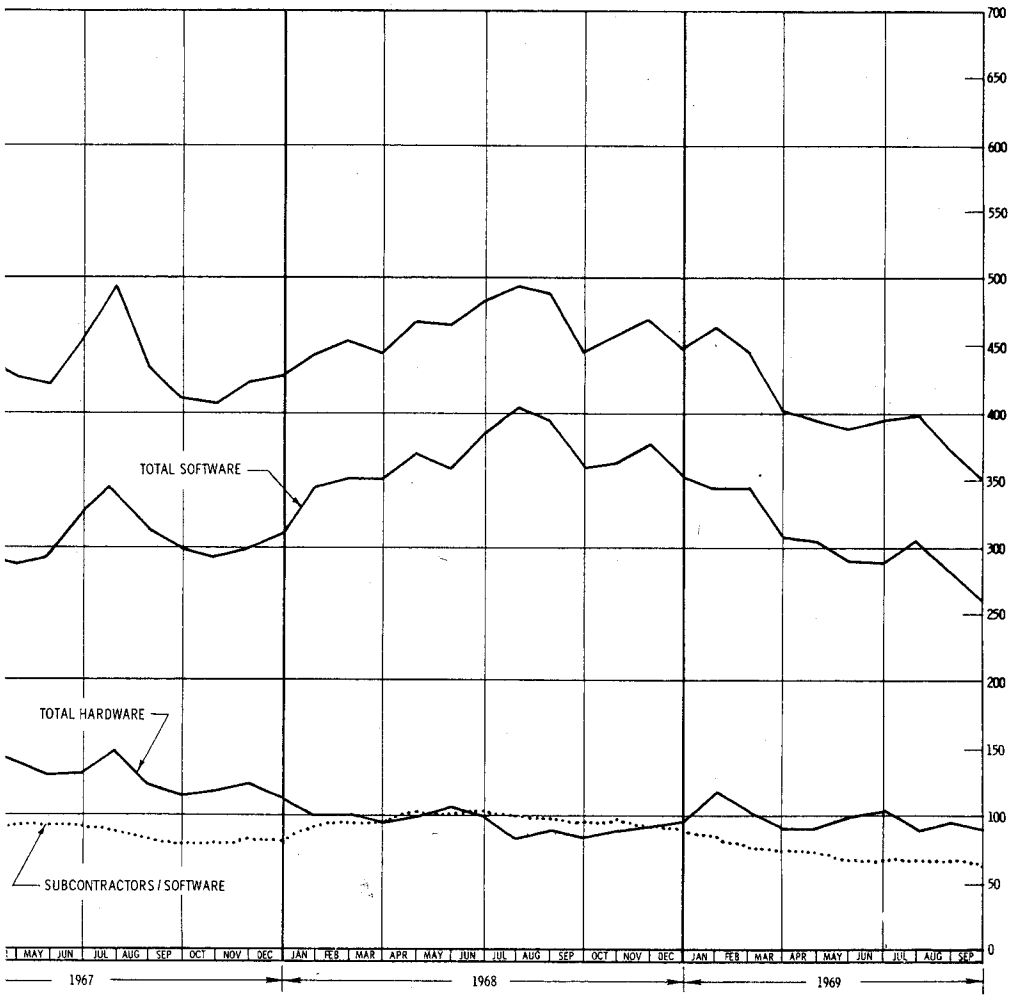


Figure 1.4-1 APOLLO Manloading - Charles Stark Draper Laboratory

57												58												59												60								
J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O

Work started on "A Recoverable Interplanetary Space Probe"

"A Recoverable Interplanetary Space Probe" published (R-235)

NASA contracts MIT for guidance and control study for various space missions

Space-study report started in September 1959 is issued

57												58												59												60								
J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O	N	D	J	F	M	A	M	J	J	A	S	O

A
61
62
N D J F M A M J J A S O N D J F M A M J J A S O N D J F M A M
ECLIPSE
6-month NASA contract to MIT for preliminary G&N design
NASA announces Apollo GN&CS contract award to MIT
Software effort rolling-investigating software techniques • Apollo bidders conference in Washington
Earth-orbit rendezvous, launch-timing, translunar-injection studies begin
"Statistical Optimization for Space Flight" published (R-341)
Preliminary computer program, Executive, Waitlist, Interpreter, reentry, guidance, extra-atmospheric abort studies begin
"Analysis of Guidance Techniques for Achieving Orbital Rendezvous" published (E-1106)
Industrial support bidders meetings
"Earth Orbital Rendezvous" published (E-1195)
Work in following areas: "MAC" compiler, AGC simulation for H-800, booster-monitoring, integration, reentry, telemetry, rendezvous, orbital navigation, lunar-landing techniques, etc.
Decision to use micrologic in the AGC
NASA announces lunar-orbit rendezvous method • MAC for H-800 complete • "G&N for LM" published (R-373)
G&N Systems Panel Meetings begin • Software & utility programs virtually complete • MIT orders H-1800
MIT study shows LM computer can be virtually same as CM computer
Software effort phases from technique investigation to actual program development
"Universal Formulae for Conic Trajectory Calculations" published (R-382)
First MSC/MIT radar meeting
"Application of Midcourse Guidance Techniques to Orbit Determination" published (E-1261)
"A List Processing Interpreter for AGC-4" published (AGC-4 Memo No. 2)
"Backup Thrust Vector Control" published (E-1287)
"Logic Description for AGC-4" published (R-393)
"Radar Requirements for Primary G&N Operations" published (R-404)
"Sampled-Data Velocity Vector Control of a Spacecraft" published (T-355)

APOLLO HISTORY "MILESTONES"

66
67
68
J F M A M J J A S O N D J F M A M J J A S O N D J F M A M J J
AREL 202
REREL 202
AS-202 8/25/66
54 55
APOLLO 4 11/9/67
BREL 204
AREL 204
AS-204 1/25/67 ACCIDENT
APOLLO 6 4/4/68
SUNDISK
282
ROPE SPLIT
FACI
CARR
COLOSSUS
COL I
ROPE
TO THE PRESENT
TO THE PRESENT
PERSONNEL BUILDUP
LUMINARY
LUM 1
SUNDANCE
ROPE SPLIT
292
302/1
CA
SUNBURST
ROPE SPLIT
116
120
APOLLO 5 1/22/68
AS-204 GSOP published (R-507)
TVC DAP progressing • NASA management changed from G&C to FOD
Apollo 4 nominal simulation complete • G&N Systems Implementation Meetings end
"Apollo CSM Reaction Control by DAP" published (E-1964)
5/13/66 First "Black Friday" fixed-memory storage meeting • Major reorganization – Group 23B formed
Apollo 5 GSOP published (R-527)
Apollo 4 GSOP published (R-537)
Separate work into subroutines • Apollo 7 GSOP published (R-547)
1/13/67 Second "Black Friday" fixed-memory storage meeting
Verification plans – definitions of levels of testing
"Critique of Apollo G&N Design" published (E-2119)
8/28/67 Third "Black Friday" fixed-memory storage meeting
PCR procedures adopted
Major reorganization: project managers & DAP group
Tiger Team arrives to change Apollo 9 rendezvous
NASA review of LM DAP
COLOSSUS GSOP published (R-577)
SUNDANCE GSOP published (R-557)
66
67
68
J F M A M J J A S O N D J F M A M J J A S O N D J F M A M J J

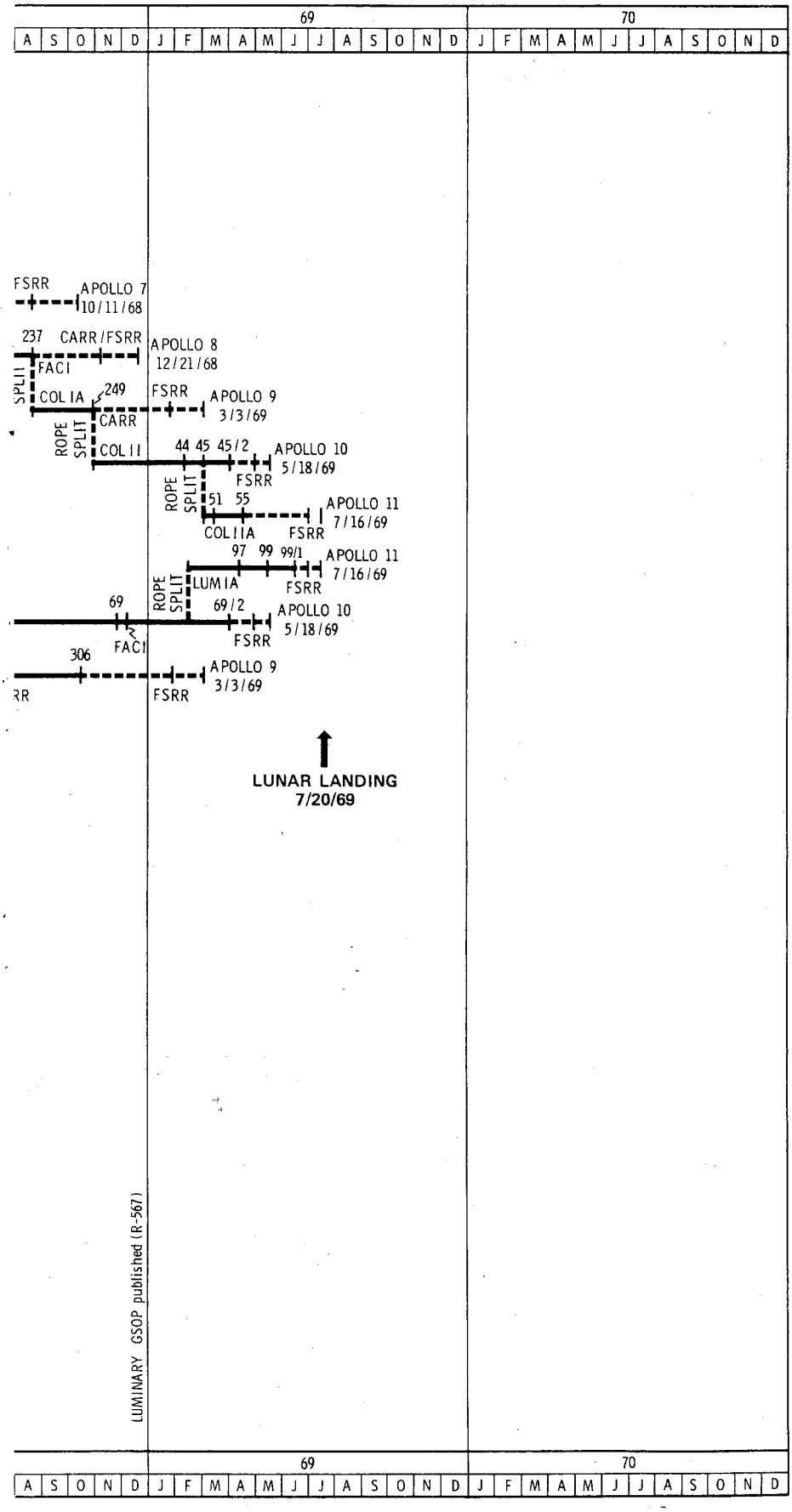


Figure 1.4-2 Apollo History "Milestones"

capability. The IBM 650 was a 2000-word-drum central processor, with 60 words of core storage. One tape drive and a disc bar were the only pieces of peripheral equipment. Programmers would write in MAC*, and the IBM 650 was used primarily to compile these programs for computation on much faster and more powerful outside equipment, such as the IBM 704, 709 and 7090. Toward the end of 1959, the burden of the NASA pre-Apollo workload, added to the much larger workload of the Laboratory's Polaris project, stimulated investigation into the possibility of providing additional in-house equipment to accommodate all the work then done by the IBM 650 and the outside rented machines.

1.4.2.2 Honeywell 800

The Honeywell 800 was ordered during Summer 1960, with delivery occurring in December 1961. Based upon the workload of mid-1960, the H-800 was predicted to run about 4 hours per day and to cost no more than the previous total of in-house IBM 650 and outside rented time. By the time the H-800 was placed in production** —in May 1962—it was apparent that even greater speed and power were necessary. Rather than the expected four hours per day, two operator shifts (16 hours/day) were required for the initial H-800 workload. Despite the unexpected demands which the H-800 faced immediately upon being placed in production, it represented approximately a threefold increase over the capabilities of the IBM 650.

To overcome the inadequacy of the Honeywell 800, two approaches were undertaken simultaneously in mid-1962. First, additional memory and peripheral equipment were acquired for the H-800; second, an order was placed for the Honeywell 1800, with expected delivery 18 months later.

The Honeywell 800 had been delivered with a 16,000-word memory, each word having 48 bits. It included a printer, six tape drives and a card reader/punch. To upgrade the H-800 while awaiting delivery of the H-1800, the memory was doubled, additional tape drives and a printer were acquired, and a disc file and a graphic plotter were added.

---

*MAC is a high-level programming language for general-purpose computers, developed at MIT for scientific application. It is not to be confused with MIT's Project MAC. The latter was named independently some years later and is unrelated to the MAC language.

**"In production" implies that the equipment included a MAC Compiler capable of a complete AGC simulation. Computers were "in operation" before they could be "in production".

1.4.2.3 Honeywell 1800

Honeywell's 1800 possesses a 2-μsec access-to-memory, while the H-800's access was on the order of 6 μsec. The H-1800's delivered memory size was 32,000 words, double that of the H-800. These capabilities rendered the delivered H-1800 roughly three times as powerful as the H-800.

Although the Laboratory's H-1800 was delivered in January 1964, it was not until the following May that the system was in total production. In the meantime, failures in hardware necessitated total replacement of the machine's memory. As a result of these difficulties with the new system, between the months of January and May, no in-house digital-computing facilities were available, since the H-800 had been removed to provide space for the H-1800. Consequently, time was rented on outside equipment during this period.

By October 1964, it was becoming apparent that the H-1800 was computing much more rapidly than its peripheral equipment could provide input and output. At that time, Honeywell announced its Model 200 computer, a small machine that could do much of its own computation, could provide its own input and output, and could serve as a buffer for the much more powerful H-1800. MIT ordered a Model 200 for delivery in October 1965.

Two decisions were reached in Summer 1965 regarding the need for additional computing facilities: a second Honeywell 1800 was ordered in June; and a study was begun of the potential advantages offered by even more powerful computers. The second H-1800 was delivered and placed in production in March 1966. The investigation into other computers resulted in the Laboratory's decision to order an IBM 360, Model 75.

The original H-1800's memory had been increased in size from 32,000 to 48,000 words. The second H-1800 was delivered with the larger memory. By the time of the second H-1800's acceptance, a second Model 200 had also been acquired. The final upgrading of the H-1800 facilities occurred with the delivery of a Honeywell Model 2200, a system approximately equivalent to two Model 200s. It was estimated that the addition of the Model 2200 increased the capability of the H-1800 facilities by about 20 percent.

1.4.2.4 IBM 360/75

When the Summer-1965 study of large computing systems began, several systems were under consideration. One was highly valued, but doubts existed that

it would ever be manufactured. Another system was by far the fastest machine under consideration, but Laboratory officials were concerned that internal parity checking would not reach the standard necessary to ensure the safety of astronauts —the ultimate customers of the Laboratory's services. Still another system was rejected primarily because it did not allow eventual expansion into an even larger system. Finally, the IBM 360, Model 75 (360/75) was chosen because of its relatively high speed, its degree of internal error checking, and the availability of the more powerful Model 91, should the need for expansion occur. It was estimated that a single IBM 360/75 would be roughly equivalent to four Honeywell 1800s.

The IBM 360/75 was delivered in October 1966, and it became operational two months later. During the first eight months of operation, three basic activities consumed most of the machine's availability: MAC language was adapted for the 360/75, system software was developed, and simulation software was implemented. During these first months of IBM 360/75 operation, it was concluded that the CPU's 512,000-byte memory would not suffice for simulation purposes; memory size was thereafter doubled. Not until September 1967, about ten months after delivery, was the IBM 360/75 in total production for general simulations.

By the time the IBM 360/75 came into total production, the need for a second IBM 360/75 was already recognized. Accordingly, the Honeywell 1800s would be removed. Removal of the second-delivered H-1800 occurred in December 1967, and the original H-1800 was removed in April 1968 to make way for the second IBM 360/75, to be delivered the following month. Thus, during the last quarter of 1967, three complete systems were in operation— the IBM 360/75 and the two H-1800s.

The second IBM 360/75 was placed in total production a mere two weeks after delivery, primarily as a result of the experience gained through the lengthy break-in procedures on the first IBM 360/75. By the time the second system was placed in production, the peripheral equipment originally delivered had also been expanded in power and capacity. For instance, the six original IBM 2311 disc packs were increased to ten. Two printers were added to the original two, and additional tape drives and a card reader were acquired. Finally, three IBM 2314 disc packs were gained, each of which was roughly equivalent to four IBM 2311s.

In August 1969, following Apollo 11's successful lunar mission, the second-delivered IBM 360/75 was removed, thus leaving the original IBM 360/75 and the systems's peripheral equipment as the remaining digital-computing facility of the Draper Laboratory. Although the remaining IBM 360/75 was deemed adequate for the needs after the lunar landing, within seven months it also reached saturation.

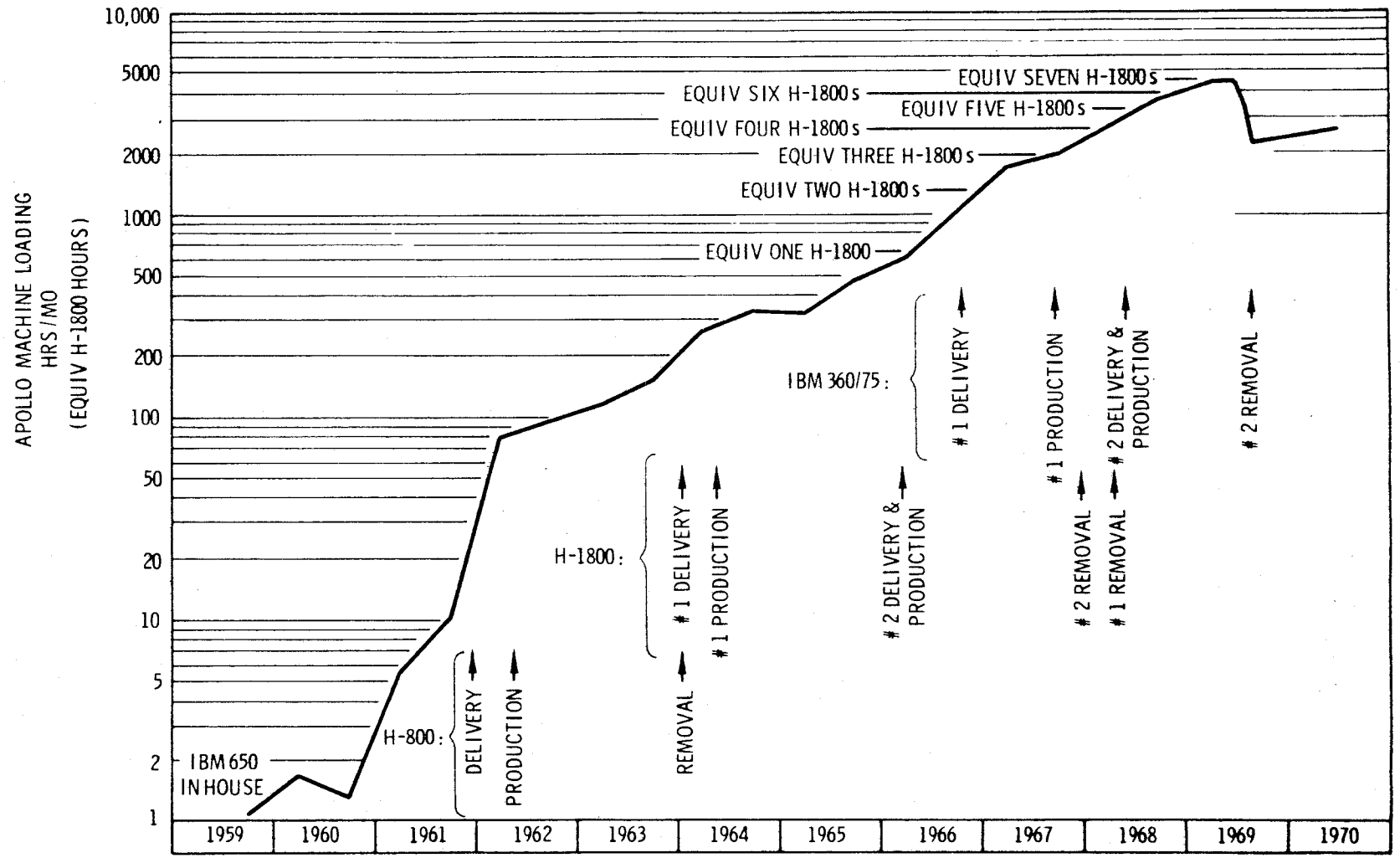


Figure 1.4-3 Apollo Machine Loading

1.4.2.5 Loading of the Digital Computing Facilities

Figure 1.4-3 charts the monthly load which was logged on the Laboratory's digital-computing facilities, expressed in equivalent Honeywell-1800 hours. In this figure, monthly saturation of a Honeywell 1800 is 660 hours (=22 hours/day x 30 days). Since the Honeywell 800 was roughly a third as powerful as the H-1800, saturation of the H-800 occurs at 220 hours/month. The IBM 650 was, in turn, about one third as powerful as the H-800—or a ninth as powerful as the H-1800; thus on this graph its saturation is 73.3 hours/month. The IBM 360/75 is roughly four times as powerful as the H-1800, and thus its saturation occurs at 2640 hours/month. This figure also indicates the dates of computer acquisitions and removals.

1.5 Major Recurrent Problems

With the manned lunar landing and return accomplished in July 1969, Project Apollo met the national goal enunciated eight years earlier. Through its design, development and implementation of the Primary Guidance, Navigation and Control System for the Apollo spacecrafts, MIT's Draper Laboratory shared in that eminent success. Along the course of its participation in the Apollo adventure, MIT experienced the kinds of technical and managerial difficulties that can only be expected in undertaking so massive a program—but that nevertheless create uneasiness at the time of their occurrence. This section focuses on the two problems which caused the greatest difficulty in the software effort. Difficulties were encountered in the estimate of time and manpower schedules and in the control of accurate, up-to-date spacecraft data. Both of these problems continually plagued MIT's software efforts, since neither their cause nor their solutions could be found within the Laboratory, alone; ultimate solution would require an extraordinarily well-tempered orchestration among NASA and all of its contractors and subcontractors.

1.5.1 Difficulty in Estimating Time and Manpower Schedules

Throughout much of the Apollo software effort at MIT, managers have experienced difficulty in estimating the time and manpower requirements to design, test and verify the successive mission-flight programs. At the commencement of work on a new flight program, it is advantageous—perhaps even essential—to break down the total required effort into a series of smaller tasks, each fitting into a preplanned sequence of steps leading to the required whole. Specialists in each of the subdivided tasks can then be assigned stated responsibilities within a specified

time constraint. This description fits the optimal situation—the situation in which the Draper Laboratory more nearly finds itself today than it has in the past.

It is more likely that at the commencement of work on an entirely new mission-flight program, the separate tasks required to lead to the assembled program cannot be known in advance. Indeed, this was the case with virtually every program up to the revisions in COLOSSUS and LUMINARY. Part of the development process included the understanding of what these basic steps should be. In brief, at the beginning no one could forecast all the little pieces which would eventually be required, and thus predicting accurate work schedules was almost a priori an impossible task.

Another probable cause of this overall scheduling problem was that subtasks required an ordered interrelationship. Not all of the tasks could occur simultaneously; some took precedence over others, and certain later tasks could not proceed until the completion of earlier tasks. In other words, the entire sequence of tasks could be completed no sooner than the time required to complete perhaps a certain few "pacemaker" tasks. Perhaps the most difficult estimate to be made in advance was the amount of time required for iteration and retests. Thus, to forecast accurate work schedules adequately, the manager would have had to predict not only all the necessary subtasks, but, in addition, which few of these subtasks would be the pacemakers and which would later be redesigned and require further testing.

Another cause of the work-schedule problem related to the vagaries of personal dynamics. Throughout much of the software effort, management encountered a problem of deadline definition; that is, when a deadline for rope release became known, a number of intermediate deadlines or goals had to be established, particularly for pacemaker tasks, to ensure that the final deadline be met. After all of the deadlines had been assigned, it was sometimes difficult to convince software personnel of the importance of meeting the earlier deadlines; the tendency was strong for those with the earlier tasks to aim toward the deadline for the completed flight program. Consequently, management was continually required to reemphasize the importance of meeting each assigned deadline.

A final cause of the work-schedule problem also related to the area of human dynamics. The communication of "bad news"—e.g., news of imminent delays—slowed as it went up the line of management. This difficulty derived from the basic human drive to prefer the communication of positive tidings to that of negative findings. Both the bearer and the receiver of bad news feel uneasy with the experience, but management must encourage its personnel to communicate the bad with the good.

When the person responsible for one of the subtasks recognizes that his schedule must slip, it is human nature to defer passing along word of the delay. As this one piece of bad news progresses up the ladder of administrative responsibility, communication of the bad news is further impeded. As the initial step in rationalization, each person along the line attempts to discover for himself whether the bad news is as bad as anticipated—or if, perhaps, some degree of overstatement has occurred. Only through conscious recognition of this process by all personnel could this problem be alleviated.

Thus, four separate causes combined to render the estimation of work schedules an especially vexing problem:

a. the difficulty of predicting all of the required subtasks;
b. the difficulty of pinpointing and hastening pacemaker subtasks;
c. the difficulty of meeting deadlines for individual subtasks;
d. the difficulty of communicating "bad news" quickly through the line of management.

As MIT gained experience through its successive responsibilities in the Apollo program, the work-schedule problem became increasingly more routine—and less annoying. Nevertheless, small remnants of this problem continue to cause occasional difficulties in the scheduling of current ropes.

1.5.2 Control of Timely Spacecraft Data

The second major problem encountered by MIT software planners relates to the acquisition of complete and up-to-date data on spacecraft parameters. In the design, verification and testing of guidance, navigation and control software, it was essential that the responsible MIT engineers possess the most current data obtained by other NASA contractors in the development of the spacecraft components. From the beginning, it was clear that a mechanism for such data exchange was of prime importance.

One of the initial responsibilities of liaison personnel was the development of a data-exchange mechanism. For instance, North American Rockwell's liaison with MIT was to record the most up-to-date information on the Command and Service Modules, and the liaison from Grumman was to do the same for the Lunar Module. In practice, however, this official mechanism broke down quickly, since spacecraft engineers were reluctant to formally release data on parameters still undergoing development, measurement or testing. Such virtually universal reluctance to commit

preliminary data, even to discretionary use, rendered the officially-recognized channels rather dinosauric in current-information content. During years of effort to establish a smoothly-functioning, up-to-date data-exchange program, MIT software personnel resorted to other means for learning the parameters and tolerances to which they should design their software and simulations.

As MIT software personnel became acquainted with their peers at the other relevant spacecraft contractors, an informal network of data exchange developed. Rather than relying upon the official mechanism of liaison contact, the engineers responsible for the development of software would place strategic telephone calls to learn up-to-the-minute data being used in the development of the spacecraft systems. Although this informal method of data exchange possessed the disadvantage of consuming much valuable time, it produced the distinct benefit of collecting the most timely information available.

In an attempt to formalize the person-to-person method of data control, a "Data Book" which listed current data was organized at MIT. There were two sections within this document: Class A data, which were official and verified by an authority at the originating contractor; and Class B data, the type generally received through telephone and person-to-person communication, but which lacked official verification. But the Data-Book mechanism required personal enthusiasm for the task of collecting data—enthusiasm which virtually all dedicated software engineers feel should better be devoted to the task of designing software.

All of the parameters and tolerances to which the software and simulations were designed were published in Chapter 6 of the GSOP. (See Section 3.2.1 of this report.) In this fashion, the Laboratory kept NASA continuously and officially apprised of MIT's current information—information which could be approved along with NASA's general approval of GSOP revisions.

By no means was the problem of timely data control ever completely solved, but solely because of MIT's increased familiarity with the spacecraft components, it became somewhat less of a problem. Just as there were elements of human dynamics in the problem of time and manpower scheduling above, so, too, did personal vagaries play a role in this difficulty: people are unwilling to divest themselves of data which they consider not yet final. And the very qualities of technical competence and conscientiousness which one needs to invest in the area of data exchange are difficult to come by, since individuals so endowed generally prefer to apply these qualities in the actual software development.

SECTION II

AGC SOFTWARE

This section describes the software which controls the LM and CM guidance computers. The computer is the heart of the Apollo Guidance, Navigation and Control System. The software maintains knowledge of the vehicle's position and velocity in space, determines the path to a desired destination, and during a maneuver, steers the spacecraft along that path by sending commands to the engines. It communicates with the astronauts and the ground, and monitors the performance of the GN&C System.

Mission programs, such as rendezvous, targeting and landing, control some of the phases of an Apollo flight. However, before these can be discussed, it is necessary to examine the underlying computer organization which allows the mission program to operate. Thus, Section 2.1 describes the basic machine architecture, the Executive and service programs which control AGC operations, and the input/output functions which allow the computer to monitor the GN&CS and to communicate with the astronauts and the ground. Although the CM and LM computers satisfy different mission requirements, the underlying system software is quite similar for the two vehicles. Hence Section 2.1 presents a generalized Apollo Guidance Computer, and specific differences are noted when they apply.

Section 2.2 includes a general description of all the phases of the Apollo mission and of the major mission-program capabilities required for that mission. The design effort which produced these mission programs has been a long and challenging task. This report will not attempt to give a complete discussion of this effort, since it has been documented in other sources; however, the rope design philosophy and the problems encountered as it finally evolved are discussed in Section 2.2.3, and the major mission-program capabilities are described in greater detail in the appendices to this report.

2.1 Computer Capabilities

2.1.1 Storage and Manipulation of Computer Instructions

The AGC contains two distinct memories, fixed and erasable, as well as hardware logic circuits. The fixed memory is stored in a wire braid which is manufactured and installed in the computer. This memory cannot be changed after manufacture and it can only be read by the computer. Fixed memory contains 36,864 "words" of memory grouped into 36 banks. Each word contains 15 bits of information (a sixteenth bit is used as a parity check). The word may contain either a piece of data, or an instruction which tells the computer to perform an operation. A series of instructions forms a routine or a program. In addition to storing programs, the fixed memory stores data such as constants and tables which will not change during a mission.

The erasable memory makes use of ferrite cores which can be both read and changed. It consists of 2048 words divided into 8 banks. Erasable memory is used to store such data as may change up to or during a mission, and is also used for temporary storage by the programs operating in the computer.

Included in the hardware logic circuits is a Central Processing Unit (CPU). The CPU performs all the actual manipulation of data, according to the instructions designated by a program. The 34 possible machine instructions include arithmetic operations (add, multiply, etc.) as well as logical operations, sequence control, and input/output operations. Also included are a limited number of "double-precision" instructions which permit two words of data to be processed as a single "word" of greater precision.

The memory cycle time (MCT) in the AGC is 11.7 μsec. Most single-precision instructions (e.g., addition) are completed in two MCTs; most double-precision machine instructions are completed in three MCTs. The unconditional transfer-control instructions, however, operate in one MCT.

To be used as an instruction, a computer word must specify the operation to be performed and give the location of the data to be operated on. However, a 15-bit word does not contain enough information to specify 34 operations and 38,912 fixed and erasable locations. In fact, 15 bits cannot even specify 38,912 locations unambiguously. It is for this reason that both the fixed and the erasable memories are grouped into banks. An instruction may specify any address within its own bank, and may also address the first four banks of erasable and the first two banks

of fixed memory. Access to other banks is accomplished using bank-selection registers in the CPU. In many cases a program exists entirely within one bank of memory, so that bank switching is not required.

Many of the tasks the AGC performs can be adequately carried out by machine instructions. However, for extensive mathematical calculations—in such areas as navigation—the short word length of the AGC presents difficulties. It limits the number of instructions available, the range of memory that can be addressed without switching banks, and the precision with which arithmetic data can be stored and manipulated. To alleviate these problems, nontime-critical mathematical calculations are coded in "interpretive language" and are processed by a software system known as Interpreter. Each Interpreter instruction is contained in two or more consecutive computer words. The increased information available allows more possible instructions and a greater range of memory addressable without bank switching. In fact, with some exceptions, all of erasable memory and fixed memory may be addressed directly. Among the available Interpreter instructions are a full set of operations on double-precision quantities, including square root and trigonometric functions, some triple-precision instructions, and a set of vector instructions such as cross product, dot product, matrix multiply, and vector magnitude. Interpreter routines translate an Interpreter instruction into an equivalent series of machine instructions to be performed by the CPU. Thus, one Interpreter instruction may be equivalent to many machine instructions, and much storage space is saved in the computer. The Interpreter also contains software routines for the manipulation and temporary storage of double- and triple-precision quantities and vectors.

Interpreter expands the processing capabilities of the CPU hardware. However, its operation is quite slow, since the CPU must perform all the actual operations, and much time is spent in the translation of instructions and the manipulation of data. Although processing time is slower, much storage space is saved in fixed memory by the more powerful Interpreter instructions; thus, the vast majority of nontime-critical mathematical computations are coded using interpretive language.

2.1.2 Timing and Control of the Computer

Two of the more stringent requirements placed upon the AGC are the need for real-time operations and the necessity for time-sharing of multiple tasks.

Certain computer functions must occur in real time. For example, certain input data must be stored or processed immediately upon receipt; and outputs, such as those which turn the jets on and off, must occur at precisely the correct time.

An interrupt system causes normal computer operation to be suspended while performing such time-critical tasks.

Several programs, which are less time-critical, may all be required during a phase of the mission. Time sharing between these programs is controlled by a software executive system which monitors the programs and processes them in order of priority. The Executive can stop one job when a higher priority job is necessary, then resume the low-priority job when time is available.

2.1.2.1 Interrupt System

To permit quick response to time-dependent requests, the AGC has a complex interrupt structure. There are two classes of interrupts, counter interrupts and program interrupts. Counter interrupts have the highest priority of all AGC operations. Counters are locations in erasable memory which can be modified by inputs originating outside the CPU. Some counters are used as clocks, while others interface with spacecraft systems to receive or transmit sequences of data pulses. The counters respond to a set of involuntary instructions called counter interrupts, which may increment, decrement, or shift the contents of the counters. A counter interrupt suspends the normal operation of the CPU for one MCT, while the instruction is being processed. Except for the short time loss, the ongoing program is not affected by the counter interrupt; in fact, it is not aware that the interrupt has occurred. These interrupts are used solely for counter update and maintenance; their priority assures that no information will be lost in the counters.

The use of counters as input/output devices will be described in Section 2.1.3.1; it is appropriate now, however, to discuss the six counters which are used for timing purposes. Two counters, designated TIME1 and TIME2, form a double-precision master clock in the AGC. TIME1 is incremented at the rate of 100 counter interrupts per second. Overflow of TIME1 triggers a counter interrupt to increment TIME2. Since total time that must elapse before TIME2 overflows exceeds 31 days, TIME1 and TIME2 are thus able to keep track of total elapsed mission time.

The remaining clock-counters, designated TIME3 through TIME6, measure time intervals needed by the AGC hardware and software. For example, autopilot computations must be processed periodically whenever the autopilot is in use. Before reaching completion, these computations preset the TIME5 counter so that it will overflow at a specified time in the future. TIME5 is incremented at the rate of 100 counter interrupts per second. When TIME5 overflows, a signal sent to the CPU causes a "program interrupt" which interrupts the program in process and begins the autopilot computations once again.

Program interrupts have lower priority than counter interrupts, but greater priority than normal program operation. Unlike counter interrupts, the purpose of program interrupts is to alter the normal processing sequence. There are 11 program interrupts; they may be triggered by a clock-counter overflow, as in the example given above, or by externally generated signals, such as the depression of a key on the Display and Keyboard (DSKY) by an astronaut. The occurrence of a program interrupt causes the computer to suspend normal operation at the end of the current instruction. The current CPU data are saved, the computer is placed in interrupt mode, and control is passed to a preassigned location in fixed memory. This preassigned location is the beginning of a program which performs the action appropriate to the interrupt. While the interrupt program is running, the computer remains in interrupt mode, and no additional program interrupts will be accepted, although counter interrupts can still occur. (Requests for other program interrupts are stored by the hardware and processed before returning to normal operation.) At the conclusion of the interrupt program, a "resume" instruction is executed. If there are no other program interrupts, the CPU is taken out of interrupt mode, the original contents are restored, and the program returns to the point at which it was interrupted. One program interrupt (restart) takes precedence over all the others, and can even interrupt an interrupt. It results from various kinds of computer malfunctions. (This interrupt will be discussed in Section 2.1.4.)

A computation which takes place by means of a program interrupt is called a task. Since tasks may not be interrupted, they must be short to avoid delaying other tasks. This speed requirement precludes the use of interpretive language.

One class of tasks is initiated by overflow of time counters TIME3, TIME4, TIME5, and TIME6. These are considered time-dependent tasks. The TIME5 interrupt, described above, initiates autopilot computations at precise periodic intervals. TIME6 controls the timing of the autopilot RCS jet firings. TIME4 initiates a series of routines which periodically monitor the IMU, radar, etc., and process input/output commands. The TIME3 counter is under the control of the software executive system (described below). It is available for general use by any program needing to schedule a task for a specific time.

A second class of tasks is initiated by interrupts caused by external action. For example, depressing a DSKY key initiates a task that begins processing DSKY readings and storing the information for later processing. Telemetry and the radar also cause interrupts that initiate tasks to receive or transmit the next data word.

2.1.2.2 Software Executive System

Computation in the AGC is managed by a software executive system comprised of two groups of routines, Executive and Waitlist. This system controls two distinct types of computational units, jobs and tasks. In its normal operating mode, the computer processes jobs. These are scheduled by the Executive, according to a priority system. The Waitlist uses the TIME3 interrupt to schedule tasks for a specific time in the future. (Tasks originated by the other program interrupts take place independently of the software executive system.)

Most AGC computations are processed as jobs. Division of a program into discrete jobs is at the discretion of the programmer, who also assigns a priority to each job indicative of its importance. The Executive can manage up to seven jobs (eight in the LM program) simultaneously.

To schedule a job, the Executive places the job's priority and beginning location on a list, assigning the job a set of working storage locations called a core set. In addition, if a job requires a larger working storage, as in the use of interpretive language, a second area, called a VAC area, may be assigned. The Executive is capable of maintaining seven core sets (eight in the LM program) and five VAC areas as each is assigned to a job, and of redesignating them as available when the job is finished.

A job in process must periodically call Executive to scan the list of waiting jobs, thus determining if any scheduled job has a priority higher than itself. If so, the job currently active is suspended and the higher priority job is initiated. To permit suspension of a job and subsequent resumption at a point other than its beginning, the working storage associated with the job is saved when the job is suspended and restored when the job is reinstated. A suspended job is returned to the job list and is not reinstated until it has the highest priority on the list. Eventually, a given job will run to completion, at which time it is removed entirely from consideration. When all jobs on the list have run to completion, a "DUMMYJOB" with zero priority constantly checks to see if new jobs have appeared. (The computer also performs a self-check, as described in Section 2.1.4.)

The relative importance of a job may change for various reasons. When this is the case, Executive changes the priority list and rechecks the list for the job of highest priority. Many times it is desirable to purposely suspend the execution of a job, but not to terminate it completely. Temporary suspension is desirable to await an event such as the input or output of data, or for the availability of a

nonreenterable subroutine currently in use. To accomplish temporary suspension, Executive saves the job's interrupted registers and sets its priority to a negative value. Because the interrupted job has a negative priority, DUMMYJOB has priority over it. As a result, the job is, in effect, suspended indefinitely. Eventually, Executive is called to restore the job, usually by the event for which the job is waiting. Executive restores the original priority and again checks the list for the highest priority job.

Waitlist allows any program to schedule a task to occur at a specified time in the future. The TIME3 clock interrupts the job in process at the correct time and initiates the task. (As mentioned before, tasks initiated by the other program interrupts are not controlled by the Executive.)

To schedule a new task, Waitlist requires the starting address of the task and the amount of time which must elapse before execution. Waitlist maintains a list of tasks waiting to run in the order in which they will be performed and a list of time differences between adjacent items on the task list. It determines when the new task will run in relation to others on the list, placing it appropriately in the list.

The TIME3 counter counts the time to the first item on the list. When this time arrives, the TIME3 program interrupt occurs. TIME3 is immediately set to overflow when the time has elapsed for the next task on the list, and all tasks and times move up one position on the list. The computer remains in interrupt mode until the task is completed. It is then free to process other interrupts or return to the original job.

Since TIME3 is a single precision AGC word (15 bits) that is incremented 100 times a second, Waitlist can process tasks up to 162.5 sec in the future. For longer delays, a routine called LONGCALL processes a single task—the repeated calling of Waitlist. LONGCALL can schedule tasks for as long as 745 hours in the future, a time span larger than an entire Apollo mission.

2.1.2.3 Sequence Control

In normal AGC operation, the Executive maintains a constant background of activity, while program interrupts break in for short, time-critical bursts. The execution of a job is subject to numerous interruptions. A counter interrupt may occur after the completion of any instruction. Program interrupts stop the job in process. While the computer is in interrupt mode, any further program interrupts

are saved by the hardware and processed one at a time before returning to the job. Under control of the Executive, high-priority jobs also steal time from a job in process. This control system of interrupts and priorities ensures that in times of heavy load, the most critical computations for the mission will be processed first.

Normally, the CPU does not stop during periods of low activity. If no jobs or tasks are being executed, the CPU executes a short loop of instructions (DUMMYJOB) which continually looks for jobs to initiate. Periodically, TIME4 overflows, initiating a task to monitor various GN&C subsystems. If an autopilot is in operation, TIME5 triggers other interrupts for autopilot functions. In addition, periodic counter interrupts will occur as counter input is received and clock counters are updated. More extensive computer activity awaits action by the astronaut, as described in the following section.

2.1.3 Computer Interfaces

To perform its various functions, the AGC must interact with the other spacecraft systems, the astronaut, and the ground. External to the AGC are the various sensors and controls which provide inputs, and the spacecraft systems and displays which receive outputs. Figures 2.1-1 and 2.1-2 illustrate the signal interconnections between the computer and the external hardware for the CM and LM systems, respectively. This report will not, in general, discuss these external equipments, except as they apply to specific AGC programs. (See functional description treated in Part 1, Chapter II, and Part 2, Chapter I of this Apollo Final Report.)

Within the AGC, the actual transmission of data is accomplished through special registers known as counters and channels, as discussed below. Various AGC programs process the input and output data. A mission program such as rendezvous will interrogate selected counters and channels for the specific input data it requires. The program will, in turn, issue commands by means of these interfaces. The operation of the mission programs is discussed in Section 2.2 and in the appendices. In addition to the mission programs, there are also special programs designed to process input/output information for purposes of telemetry and communication with the astronauts. These interfaces are discussed in the present section.

2.1.3.1 Counters and Channels

All AGC input/output takes place through counters and channels. Counters are used for the transmission and reception of numeric data; channels are used for the communication of discrete* data.

*The AGC has 15 input and output channels whose bits are individually distinct (i.e., discrete). Each bit either causes or indicates a change of state, e.g., liftoff, zero optics, SPS-engine on, RCS-jet on.

Channels are solid-state registers in the CPU that do not form part of memory. They cannot be referenced by most machine-language instructions, but are read and in some cases written into by means of special channel instructions. Each channel can consist of up to 15 separate bits or discretes. For input channels, the discretes are set by external G&N hardware and may be read by the computer. The input channels inform the computer of the state of the hardware, such as a hand controller out of detent, or the last key depressed on the DSKY. Output channels are written into by the computer to command external hardware functions, such as turning jets on or off, changing the DSKY display, or turning on panel lights. The AGC reads or writes into channels only when instructed to do so—either by the ongoing program or by a program interrupt. For example, pressing a key on the DSKY changes the information in channel 15; it also initiates the KEYRUPT1 program interrupt which causes the computer to read channel 15.

Counters are used for the input and output of numerical information. As described in Section 2.1.2, counters can be changed by programs as if they were ordinary erasable locations, but the counters also respond to counter interrupts which are not under program control.

For input, a typical operation requires that a counter first be set to zero under program control. The counter may then be incremented or decremented, one count at a time, via counter interrupts triggered by an external device. Thus, a counter is able to keep track of the state of the external device. An example of this kind of counter is that used with the Coupling Data Units (CDUs), the interfaces between the Inertial Measurement Unit and the AGC. For each 39.5-arcsec change in a particular gimbal angle, the CDU generates a signal to the AGC which causes a decrement or increment counter interrupt to the appropriate counter.

The output counters function in a similar way. The program sets the counter to an initial value which is later "enabled" via a channel discrete. Following the initialization, all action is automatic and not under program control. A series of counter interrupts decrement the counter toward a value of zero. For each interrupt, a signal of appropriate sign is sent to an external device. When the counter reaches zero, another signal is generated which stops the counting process. Thus, the number of signed pulses sent out is equal to the original contents of the counter. For example, signed pulses torque the gyros or control the optics shaft and trunnion drives.

For telemetry input, counter interrupts shift a pattern of bits into the counter. Selective use of two types of interrupts achieves the desired pattern after the counter has been cleared under program control.

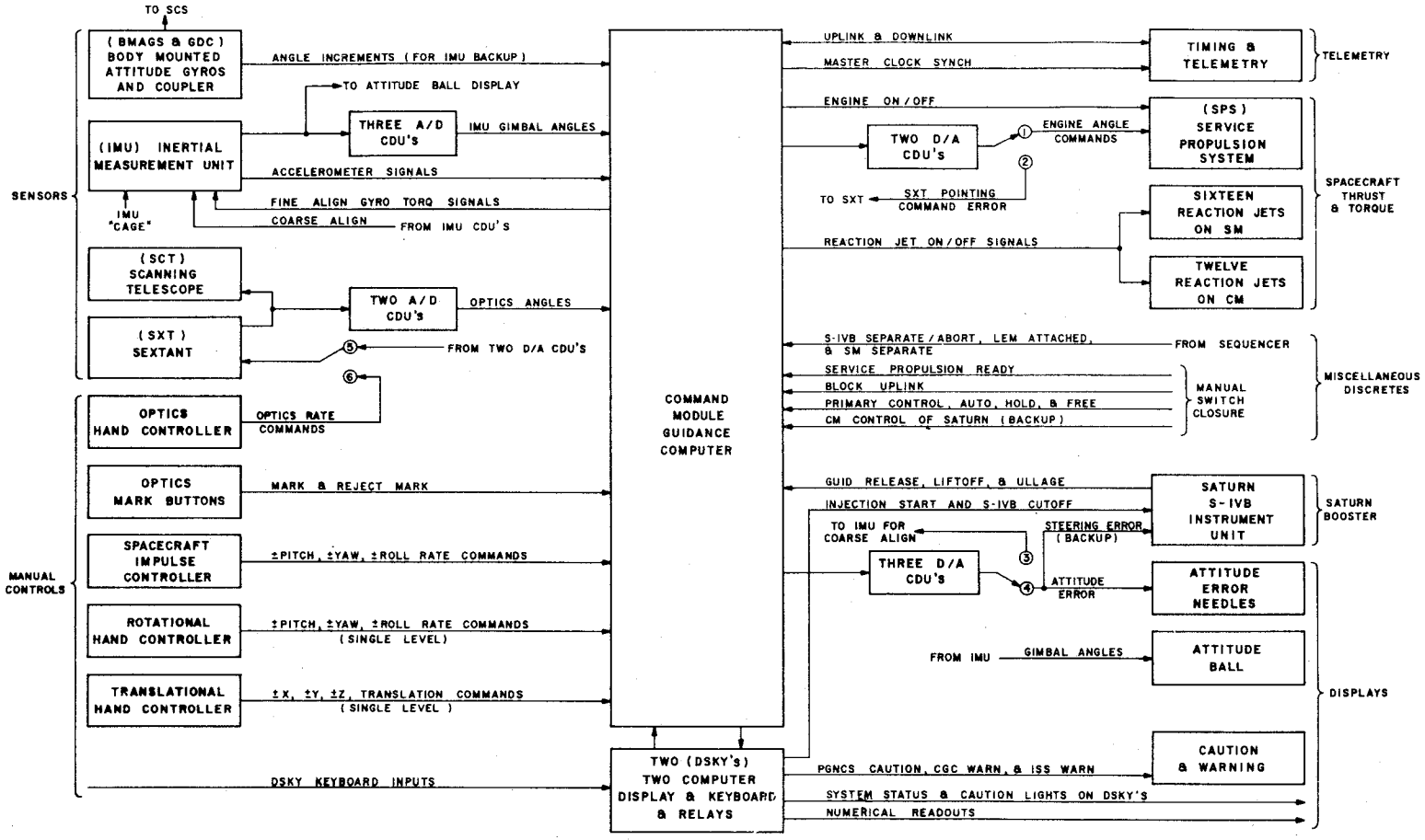


Figure 2.1-1 Guidance, Navigation and Control Interconnections in the Command Module

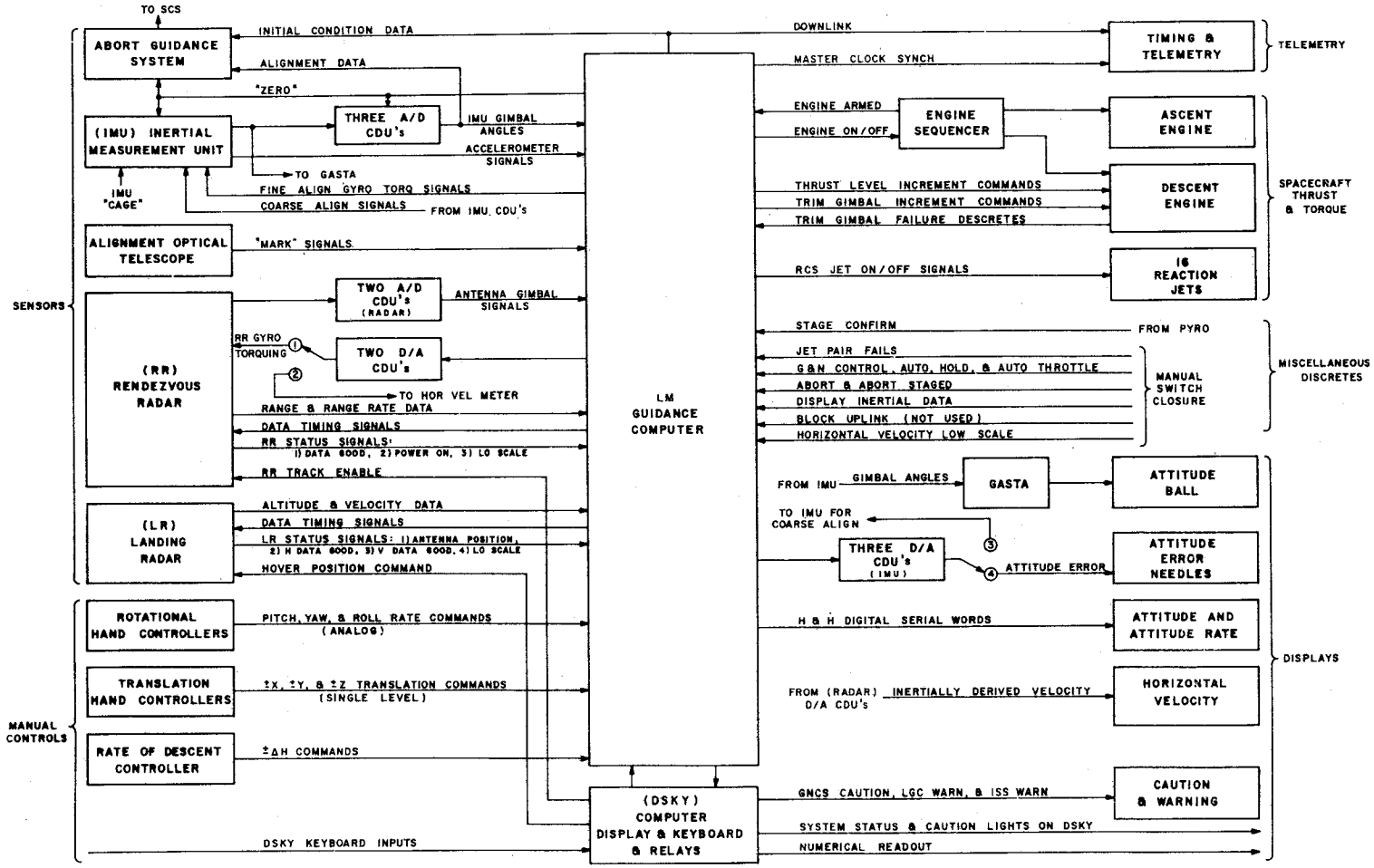


Figure 2.1-2 Guidance, Navigation and Control Interconnections in the Lunar Module

2.1.3.2 Cockpit Displays and Controls

The Apollo Guidance, Navigation and Control System has been designed to utilize the best features of man and machine. Many mission tasks are best left to the computer, such as those that are extremely tedious or that require accurate response too rapid to lie within man's capabilities. However, man's judgment and adaptability, his decision-making capability in reacting to unanticipated situations, and his unique ability to recognize and evaluate patterns are all necessary for mission success. The Apollo displays and controls have therefore been designed to provide the crew with the most flexibility in monitoring and controlling the spacecraft. The astronaut can choose to be directly involved in the procedures, or to allow automatic operation which he can monitor.

Displays available to the crew in both the CM and LM are the attitude ball, attitude-error needles, attitude-rate needles, caution and warning lights, and a DSKY. The LM has additional displays which give the astronaut essential information during the descent to the lunar surface; these are the altitude/altitude-rate, horizontal-velocity, and thrust-level meters and the Landing Point Designator.

Several manual controllers enable the astronaut to become directly involved in spacecraft control. Both the CM and LM have rotational and translational hand controllers. The LM has a rate-of-descent controller. In the CM, additional controllers are used in conjunction with the optics; these are the minimum-impulse and optics hand controllers and the optics mark buttons. In the LM, a DSKY command can convert the rotational hand controller to a minimum impulse controller. All of these controllers make available to the astronaut a large repertoire of manual maneuvers.

The basic man/computer interface device is the DSKY (shown in Fig. 2.1-3). Through the DSKY the astronaut can initiate, monitor, or change programs being processed by the computer. He can request the display of specific data or enter new data. Communication with the DSKY is two-way: the astronaut can exercise command via the DSKY and the computer can request the astronaut to monitor, approve, or enter data when necessary. There are two DSKYs available in the CM and one in the LM. Each DSKY has a keyboard, several electroluminescent displays, and activity and alarm lights. The activity lights are for the computer and the telemetry uplink, and the alarm lights are for the computer and inertial subsystems. These aid the astronaut in monitoring the status of the G&N system. The alarm lights indicate equipment-failure and program alarms. There are two levels of program alarms. The more serious type of alarm either terminates all but the

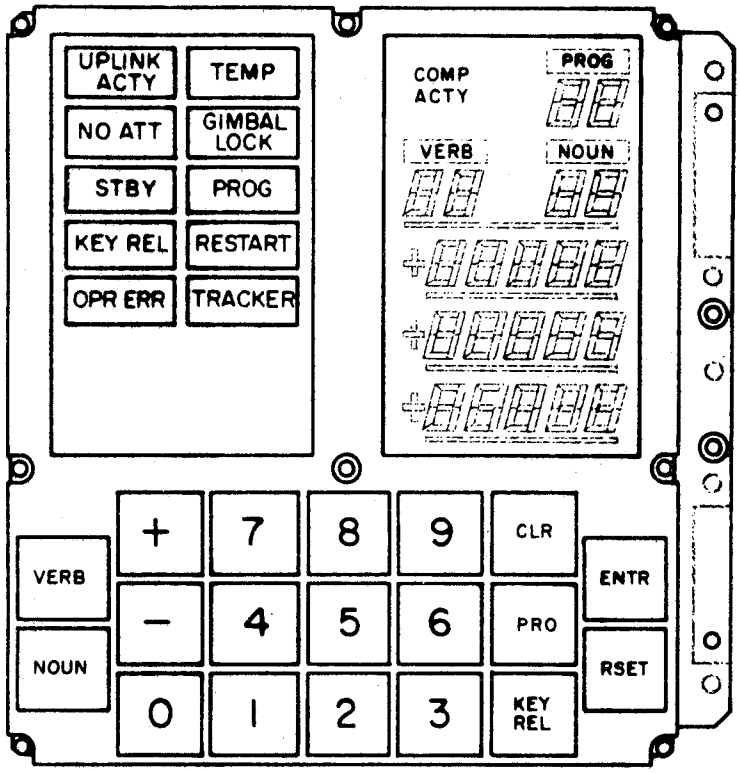


Figure 2.1-3 Display and Keyboard

most necessary program activities or terminates all current program activities and requests astronaut action. The latter is accomplished by a preemptive flashing display of an error code indicating the cause of the alarm. The other type of program alarm is also indicated by the program-alarm light, but in this case the program in process continues without change. Should the astronaut wish to interrogate the cause of this alarm, he can key in a request to the computer to display the error code. The DSKY keyboard and displays are discussed in the next section.

2.1.3.3 PINBALL and DSKY Displays

The AGC program which responds to DSKY buttons and requests illumination of the DSKY lights is called PINBALL GAME BUTTONS AND LIGHTS—or PINBALL, for short. PINBALL is under Executive control and enables communication between the computer and the astronaut. As mentioned in the previous section, communication can be initiated by operator action or by an internal computer program. Four modes of operation are associated with PINBALL—internal data display, external data loading, systems-test usage, and initiation of large-scale mission phases. Internal data can be displayed once for verification (e.g., the ascent-injection parameters for lunar ascent) or periodically updated and displayed for monitoring (e.g., time-to-go to main-engine ignition). External data are displayed in the appropriate display-panel register as they are keyed into the DSKY. The data for the loading (external) and displaying (internal) modes can be presented in octal or decimal format; if internal data are presented in decimal format, the program supplies the appropriate scale factors for the display. PINBALL can also initiate a class of routines used for systems-test functions which might require operator interaction to determine whether to stop or continue the routine. The final mode of PINBALL is initiation of large-scale mission phases by operator action, i.e., by changing the mission program via the DSKY. (Fig. 2.1-4 lists the AGC programs for the CM and LM available during a lunar mission.)

The DSKY keyboard contains the following notations: VERB, NOUN, +, -, the numerical characters 0 through 9, CLR (clear), ENTR (enter), RSET (error reset), PRO (proceed), and KEY REL (key release). Each of these notations is internally represented by a 5-bit binary code which is transmitted and recognized by the computer. When the operator depresses any one of these buttons on the keyboard, an interrupt program called KEYRUPT enters a request to the Executive for another program that decodes and stores the key code in an input register of the AGC.

The numeric sections of the DSKY panel form three data-display registers, R1, R2 and R3, which can contain up to five numerals each, and three control display

Command Module AGC Programs	Lunar Module AGC Programs
00 CMC* Idling	00 LGC** Idling
01 Prelaunch Initialization	06 GNCS Power Down
02 Gyro Compassing	
03 Verify Gyro Compassing	12 Powered Ascent Guidance
06 CMC Power Down	
07 IMU Ground Test	20 Rendezvous Navigation
	21 Ground Track Determination
11 Earth Orbit Insertion (EOI) Monitor	22 Lunar Surface Navigation
17 Transfer Phase Initiation (TPI) Search	25 Preferred Tracking Attitude
	27 LGC Update
20 Rendezvous Navigation	
21 Ground Track Determination	30 External ΔV
22 Orbital Navigation	31 Lambert Aimpoint Maneuver
23 Cislunar Midcourse Navigation	32 Coelliptic Sequence Initiation (CSI)
27 CMC Update	33 Constant Delta Height (CDH)
	34 Transfer Phase Initiation (TPI)
30 External ΔV	35 Transfer Phase Midcourse (TPM)
31 Lambert Aimpoint Maneuver	38 Stable Orbit Rendezvous (SOR)
32 Coelliptic Sequence Initiation (CSI)	39 Stable Orbit Midcourse (SOM)
33 Constant Delta Height (CDH)	
34 Transfer Phase Initiation (TPI)	40 DPS
35 Transfer Phase Midcourse (TPM)	41 RCS
37 Return to Earth (RTE)	42 APS
38 Stable Orbit Rendezvous (SOR)	47 Thrust Monitor
39 Stable Orbit Midcourse (SOM)	
	51 IMU Orientation Determination
40 SPS	52 IMU Realign
41 RCS	57 Lunar Surface Align
47 Thrust Monitor	
	63 Braking Phase
51 IMU Orientation Determination	64 Approach Phase
52 IMU Realign	65 Landing Phase (Auto)
53 Backup IMU Orientation Determination	66 Landing Phase (ROD)
54 Backup IMU Realign	67 Landing Phase (Manual)
	68 Landing Confirmation
61 Maneuver to CM/SM Separation Attitude	
62 CM/SM Separation & Preentry Maneuver	70 DPS Abort
63 Entry-Initialization	71 APS Abort
64 Entry-Post 0.05 g	72 CSM CSI Targeting
65 Entry-Up Control	73 CSM CDH Targeting
66 Entry-Ballistic	74 CSM TPI Targeting
67 Entry-Final Phase	75 CSM TPM Targeting
	76 Target ΔV
72 LM Coelliptic Sequence Initiation (CSI)	78 CSM SOR Targeting
73 LM Constant Delta Height (CDH)	79 CSM SOM Targeting
74 LM TPI Targeting	
75 LM TPM Targeting	
76 Target ΔV	
77 LM TPI Search	
78 LM SOR Targeting	
79 LM SOM Targeting	

*CMC is Command Module Computer (CM AGC)

**LGC is Lunar Guidance Computer (LM AGC)

Figure 2.1-4 Programs for a Lunar-Landing Mission

registers, VERB, NOUN, and PROG (program), of two numerals each. Each of the three data display registers has a sign section which displays a plus sign, a minus sign or nothing at all (blank). The PROG register indicates the mission program currently operating; the VERB and NOUN registers indicate the display and load activity initiated by the operator or by the computer. All information necessary to operate the display panel on the DSKY is transmitted from the computer through an output register which activates two display characters at a time. The basic language used for communication between the operator and PINBALL is a pair of two-character numbers that represents a verb/noun combination. The verb code indicates the operation to be performed, while the noun code indicates the operand to which the operation (verb) applies. Typical of the verb codes used are those for displaying and loading data. Noun codes call up groups of erasable registers within computer memory. Figures 2.1-5 and 2.1-6 give a list of the verbs and nouns available in the AGC for the CSM program COLOSSUS. (The LM program, LUMINARY, has a similar list.)

In addition to the numeric buttons and verb/noun control buttons, PINBALL responds to the other control buttons found on the DSKY. The RSET button usually turns off the alarm lights on the panel. Should any of these alarm lights remain on after the RSET button is depressed, the condition causing the alarm persists. The ENTR button has two functions: it causes the AGC to execute the verb/noun combination appearing in the VERB and NOUN registers or to accept a newly-entered data word. The CLR button is used to blank R1, R2, or R3 during a data-loading sequence, thus allowing reloading of a data word. The KEY REL light is turned on when an internal program attempts to use the DSKY but finds that the astronaut has not released it for internal use; depressing the KEY REL button performs this release. Thus, the operator has control over the displays he wishes to observe, without being interrupted by an internal request. As will be shown in a discussion which follows on multi-level displays, the KEY REL button can also be used to reestablish displays which have been temporarily suspended.

While the astronaut communicates with the computer by entering information in the DSKY, the computer communicates with the astronaut by a flashing or nonflashing verb/noun display. The loading of data registers provides an example of two-way communication. To load three registers of data, the astronaut selects VERB 25 NOUN XX ENTR, where NOUN XX describes the data involved. He then depresses the ENTR button and the computer responds by flashing VERB 21, telling him to load register R1, which has been blanked. After the astronaut keys in the initial data, he keys ENTR. The computer responds with a flashing VERB 22, indicating that it is ready to accept data in the second register. The process is

THIS LIST REPRESENTS THE VERBS USED IN PROGRAM COLOSSUS

REGULAR VERBS

00 NOT IN USE
01 DISPLAY OCTAL COMP 1 IN R1
02 DISPLAY OCTAL COMP 2 IN R1
03 DISPLAY OCTAL COMP 3 IN R1
04 DISPLAY OCTAL COMP 1,2 IN R1,R2
05 DISPLAY OCTAL COMP 1,2,3 IN R1,R2,R3
06 DISPLAY DECIMAL IN R1 OR R1,R2 OR R1,R2,R3
07 DISPLAY DP DECIMAL IN R1,R2
08 SPARE
09 SPARE
10 SPARE
11 MONITOR OCTAL COMP 1 IN R1
12 MONITOR OCTAL COMP 2 IN R1
13 MONITOR OCTAL COMP 3 IN R1
14 MONITOR OCTAL COMP 1,2 IN R1,R2
15 MONITOR OCTAL COMP 1,2,3 IN R1,R2,R3
16 MONITOR DECIMAL IN R1 OR R1,R2 OR R1,R2,R3
17 MONITOR DP DECIMAL IN R1,R2
18 SPARE
19 SPARE
20 SPARE
21 LOAD COMPONENT 1 INTO R1
22 LOAD COMPONENT 2 INTO R2
23 LOAD COMPONENT 3 INTO R3
24 LOAD COMPONENT 1,2 INTO R1,R2
25 LOAD COMPONENT 1,2,3 INTO R1,R2,R3
26 SPARE
27 DISPLAY FIXED MEMORY
28 SPARE
29 SPARE
30 REQUEST EXECUTIVE
31 REQUEST WAITLIST
32 RECYCLE
33 PROCEED
34 TERMINATE
35 TEST LIGHTS
36 REQUEST FRESH START
37 CHANGE PROGRAM
38 SPARE
39 SPARE

EXTENDED VERBS

40 ZERO CDU
41 COARSE ALIGN CDU (W N20,N91)
42 PULSE TORQUE GYRO
43 LOAD FDAI ATT ERROR NEEDLES (TEST ONLY)
44 SET SURFACE FLAG
45 RESET SURFACE FLAG
46 ACTIVATE DAP
47 SET LM STATE VECTOR INTO CSM STATE VECTOR
48 LOAD DAP DATA (R03)
49 START CREW DEFINED MANEUVER (R62)

Figure 2.1-5 Verbs Used in Program COLOSSUS

50 PLEASE PERFORM

51 PLEASE MARK

52 MARKED ON OFFSET LANDING SITE

53 PLEASE MARK ALTERNATE LOS

54 START REND BACK UP SIGHTING MARK (R23)

55 INCREMENT CMC TIME (DECIMAL)

56 TERMINATE TRACKING

57 START REND SIGHTING MARK (R21)

58 RESET STICK FLAG AND SET V50N18 FLAG

59 PLEASE MARK (OPTICS CALIBRATION)

60 SET ATTITUDE ERROR REFERENCE TO PRESENT ATTITUDE.

61 SELECT MODE 1 (DISPLAY DAP ATTITUDE ERROR)

62 SELECT MODE 2 (DISPLAY TOTAL ATTITUDE ERROR (N22-N20))

63 SELECT MODE 3 (DISPLAY TOTAL ASTRONAUT ATTITUDE ERROR (N17-N20))

64 START S-BAND ANT CALC (R05)

65 START OPTICAL VERIFICATION OF PRELAUNCH ALIGNMENT (P03)

66 SET CSM STATE VECTOR INTO LM STATE VECTOR

67 START W-MATRIX RMS ERROR DISPLAY

68 SPARE

69 RESTART

70 UPDATE LIFTOFF TIME (P27)

71 UNIVERSAL UPDATE-BLOCK ADR (P27)

72 UNIVERSAL UPDATE-SINGLE ADR (P27)

73 UPDATE CMC TIME (OCTAL) (P27)

74 INITIALIZE ERASABLE DUMP VIA DOWNLINK

75 BACKUP LIFTOFF

76 SET PREFERRED ATTITUDE FLAG

77 RESET PREFERRED ATTITUDE FLAG

78 UPDATE PRELAUNCH AZIMUTH

79 START BARBECUE MODE ROUTINE (R64)

80 UPDATE LM STATE VECTOR

81 UPDATE CSM STATE VECTOR

82 REQUEST ORBIT PARAM DISPLAY (R30)

83 REQUEST REND PARAM DISPLAY #1 (R31)

84 SPARE

85 REQUEST REND PARAM DISPLAY #2 (R34)

86 REJECT REND BACK UP SIGHTING MARK

87 SET VHF RANGE FLAG

88 RESET VHF RANGE FLAG

89 START REND FINAL ATTITUDE MANEUVER (R63)

90 REQUEST REND OUT OF PLANE DISPLAY (R36)

91 BANKSUM

92 SPARE

93 ENABLE W MATRIX INITIALIZATION

94 ENABLE CISLUNAR TRACKING RECYCLE

95 SPARE

96 TERMINATE INTEGRATION AND GO TO P00

97 THRUST FAIL DISPLAY

98 SPARE

99 ENABLE ENGINE IGNITION

Figure 2.1-5 (cont.) Verbs Used in Program COLOSSUS

THIS LIST REPRESENTS THE NOUNS USED IN PROGRAM COLOSSUS.

00	NOT IN USE	
01	SPECIFY ADDRESS (FRAC)	.XXXXX FRAC .XXXXX FRAC .XXXXX FRAC
02	SPECIFY ADDRESS (WHOLE)	XXXXX. INTEG XXXXX. INTEG XXXXX. INTEG
03	SPECIFY ADDRESS (DEGREE)	XXX.XX DEG XXX.XX DEG XXX.XX DEG
04	SPARE	
05	ANGULAR ERROR/DIFFERENCE	XXX.XX DEG
06	OPTION CODE	OCT OCT
07	CHANNEL/FLAGWORD/ERASABLE OPERATOR	OCT OCT OCT
08	ALARM DATA	OCT OCT OCT
09	ALARM CODES	OCT OCT OCT
10	CHANNEL TO BE SPECIFIED	OCT
11	TIG OF CSI	00XXX. HRS 000XX. MIN 0XX.XX SEC
12	OPTION CODE	OCT OCT
13	TIG OF CDH	00XXX. HRS 000XX. MIN 0XX.XX SEC
14	SPARE	
15	INCREMENT ADDRESS	OCT
16	TIME OF EVENT (USED BY EXT VERB ONLY)	00XXX. HRS 000XX. MIN 0XX.XX SEC
17	ASTRONAUT TOTAL ATTITUDE (USED IN MODE 3 NEEDLES (V63))	XXX.XX DEG XXX.XX DEG XXX.XX DEG
18	BALL ANGLES AUTO MANEUVER	R XXX.XX DEG P XXX.XX DEG Y XXX.XX DEG
19	SPARE	
20	PRESENT ICDU ANGLES	R XXX.XX DEG P XXX.XX DEG Y XXX.XX DEG
21	PIPAS	X XXXXX. PULSES Y XXXXX. PULSES Z XXXXX. PULSES
22	NEW ICDU ANGLES	R XXX.XX DEG P XXX.XX DEG Y XXX.XX DEG
23	SPARE	
24	DELTA TIME FOR CMC CLOCK	00XXX. HRS 000XX. MIN 0XX.XX SEC
25	CHECKLIST (USED WITH V50)	XXXXX.
26	PRIO/DELAY,ADRES,BBCON	OCT OCT OCT
27	SELF TEST ON/OFF SWITCH	XXXXX.
28	SPARE	
29	XSM LAUNCH AZ	XXX.XX DEG
30	TARGET CODE (GYROCOMPASSING VERIFICATION)	XXXXX. XXXXX. XXXXX.
31	SPARE	
32	TIME FROM PERIGEE	00XXX. HRS 000XX. MIN 0XX.XX SEC

Figure 2.1-6 Nouns Used in Program COLOSSUS