TID-4500, UC-32
Mathematics and Computers

LAWRENCE LIVERMORE LABORATORY
University of California/Livermore, California/94550

UCRL-51130 Vol. 8

Technical Support

R. E. Aley
A. Schiff

MS. date: October 19, 1971

NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

Preface

Navigation without external reference during a mission requires inertial navigation technology. If the mission length is more than a day or so, the position errors of the inertial navigator can become unacceptably large. However, if the mission includes stationary periods, the navigator's errors can be dramatically reduced by the effective use of these periods.

The idea of using the stationary periods to minimize inertial navigator errors has resulted in a concept called the Geographic Position Locator (GPL). A GPL is a completely automatic system with the following key functions:

- Sensing whether it is moving or stationary.
- Minimizing its position errors during stationary periods.
- Comparing its computed position with stored position data and, as a result of this comparison, executing a prescribed action.

The GPL concept was investigated under the PASSPORT program. PASSPORT was conducted by the Lawrence Livermore Laboratory under the joint sponsorship of the U.S. Atomic Energy Commission's Division of Military Application and the Department of Defense's Advanced Research Projects Agency.

The objective of PASSPORT was to determine the feasibility of the GPL concept and to evaluate the capabilities of a GPL system using commercial-quality inertial components.

This objective was achieved. The excellent performance capabilities of the GPL error reduction techniques that resulted should make them attractive for use in any inertial navigation system that is occasionally stationary during a mission.

The GPL final report, UCRL-51130, consists of the following volumes:

1. Summary
2. Executive Summary
3. System Theory
4. Error Reduction
5. Motion Sensing
6. Performance
7. Field Considerations
8. Technical Support
9. WISPIN—An Inertial Navigation System Simulation Program
10. DINSII—A Navigation Data Reduction Program
11. Data Reduction Kalman Filter—Subroutine
12. Data Reduction A3FIX—Subroutine
13. Vulnerability (SDI)

The GPL development project was primarily staffed by an average of about 10 persons at LLL. In alphabetical order, those persons who were on, or directly associated with, the Livermore staff engaged in this project and who made significant and original contributions to the success of the GPL program were:

R. E. Aley	D. L. Goldman	C. H. Radewan
F. T. Beers	R. H. Henderson	R. M. Refowitz
N. R. Cotter	R. J. Lepre	A. Schiff
F. J. Deadrick	L. W. McCullough	W. M. Stevens
P. D. Franklin	W. A. Niven	

Major offsite support was provided by contracts with the Charles Stark Draper Laboratory of the Massachusetts Institute of Technology (MIT), The Analytic Sciences Corporation (TASC), and EG&G Inc.

As with any project of this scope, the generous assistance of many other people, both within and without the Laboratory, was of great help and is gratefully acknowledged.

J. D. Salisbury
GPL Program Director
X-Division

M. R. Gustavson
Leader
X-Division

Contents

TECHNICAL SUPPORT

Abstract

The Geographical Position Locator "A" (GPL-A) system, consisting of a commercial aviation inertial navigation unit and a small external computer, was installed in an instrumented van so that the mathematical error-reduction techniques developed for the GPL system could be tested under realistic conditions. This volume of the GPL final report gives technical descriptions of the equipment assembled for this testing and discusses the various programs that were written for such functions as controlling the inertial platform, calculating position and velocity information, error reduction, motion sensing, and data logging.

Introduction

This volume describes the equipment used and work performed to support the PASSPORT program test effort. An off-the-shelf navigation system, with a modified computer program, was combined with an external computer to form the Geographic Position Locator "A" system (GPL-A) for the purpose of applying the mathematical techniques developed for error reduction to a real-world environment. The test facility was an instrumented van. The test effort consisted of outfitting the van, providing a programming system to incorporate various error modeling schemes while controlling the system's operation, establishing a test route with known reference locations, conducting the tests, writing programs to reduce the data from the navigation "runs" for analysis, and creating a simulation scheme to permit refinement of system modeling under controlled conditions.

GPL-A tests were conducted for a period of one year (mid-1970 to mid-1971) at the Lawrence Livermore Laboratory and along a nearby highway running essentially north and south in the San Joaquin Valley of northern California. Recorded data were processed at LLL on the CDC 6600 and 7600 scientific computers. Outside computing facilities were used to assemble programs for GPL-A computers, debug these programs, and run simulations to pretest the programs.

Preparation for the tests and operation of the system during the test period required various talents. The effort to put together a working system enlisted three programmers, two engineers, and one technician over a span of four years.

INECS Hardware

Several major items made up the system used by LLL to conduct testing under the PASSPORT program. Known as the Inertial Navigation Executive Computer System (INECS), the test system consisted of GPL-A hardware and ancillary equipment. GPL-A was defined to be the inertial navigation system and an external minicomputer. The navigator was a Litton LTN-51* system,[1,2] chosen for this application to demonstrate the effects of error reduction schemes on off-the-shelf hardware. The external computer was a ruggedized Honeywell DDP-516 digital computer. Its primary function was to provide storage for the Kalman filter schemes under investigation and adequate speed to perform the calculations in real time. Since the main executive routine ran in it, the DDP-516 controlled the entire INECS, including another computer, an Arma Micro-D, located inside the LTN-51.

The interrelationship of major hardware components of INECS is shown in

*Reference to a company or a product name does not imply approval or recommendation of the product by the University of California or the U.S. Atomic Energy Commission to the exclusion of others that may be suitable.

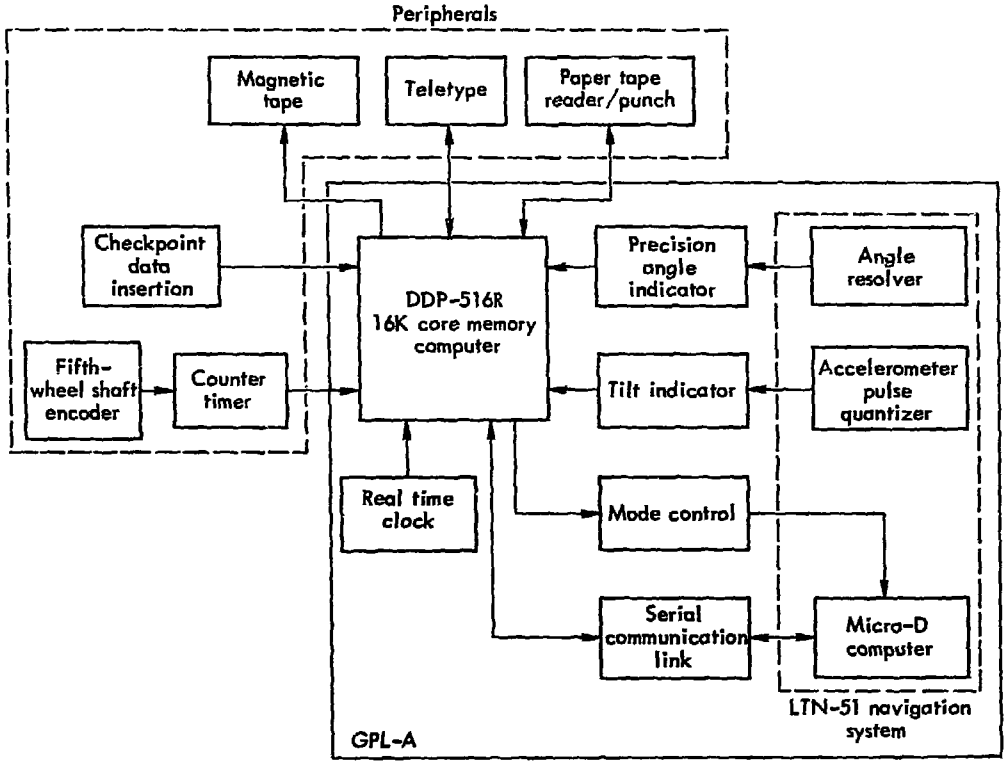

Figure 1. Inertial navigation executive computer system.

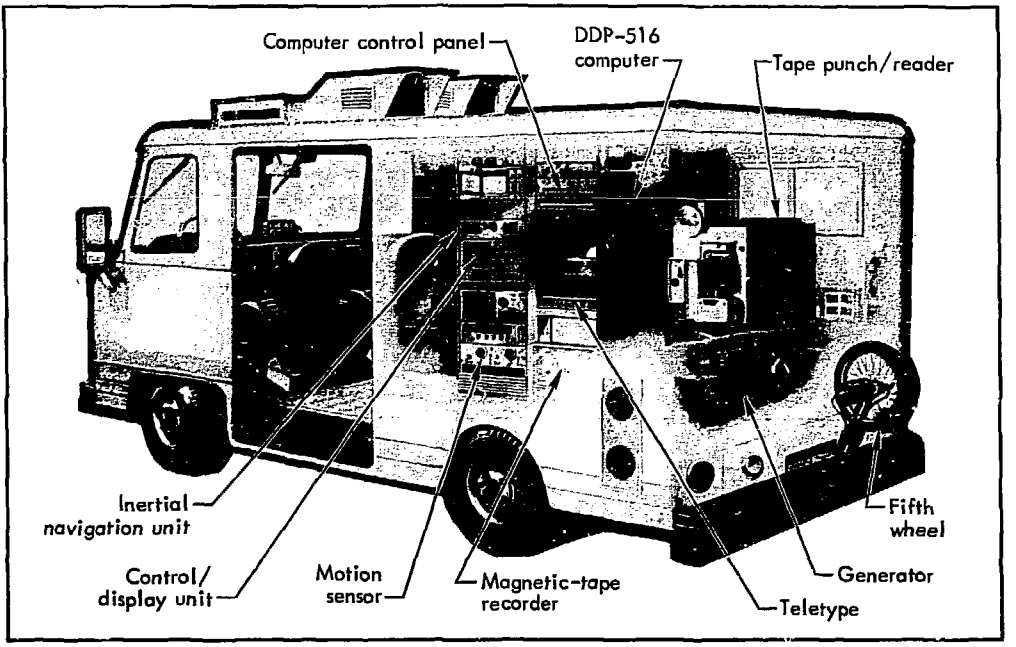

Figure 2. Mobile test unit.

Fig. 1. The two computers communicated via a serial link known as the ARINC channel. A precision angle indicator was used to provide fine (4.4 arc-sec) azimuth resolver data in digital form to the DDP-516. Reference velocity information, using test-vehicle speed, was determined by means of a shaft encoder to digitize fifth-wheel travel. This was converted to reference velocities during data reduction using heading data generated by the stable platform, thereby permitting assignment of direction to the velocity vector quantities. When the system passed by checkpoints along its route, preassigned identification numbers were entered into the computer from a thumbwheel panel. The mode of operation was changed from navigate to calibrate with a fast-acting (millisecond) relay which changed the state of a particular "bit" in a Micro-D computer storage register used to indicate the status of discrete inputs to the navigation system.

The standard peripherals attached to the DDP-516 were a high-speed paper-tape reader/punch, teletype, and magnetic tape recorder. The tape reader permitted programs to be loaded into the computer. The tape punch was used in preparation of new programs or modifications of existing ones. The teletype was used for program initialization, parameter modifications, memory dumps, debugging, and quick-look of run data. The magnetic tape recorder was used solely for data logging.

A drawing of the test vehicle is shown in Fig. 2. The equipment was housed in

a mobile van called the DINSmobile (DINS is an acronym for Digital Inertial Navigation System).

LTN-51 SYSTEM DESCRIPTION (LLL MECHANIZATION)

The LTN-51 is a compact digital inertial navigation system originally designed to fulfill the navigation requirements of modern commercial airliners.* Several Government agencies have procured the system for experimental programs; these are listed in Tables 1 and 2. Its nominal performance in the aircraft environment is an error rate on the order of 1 nm/hr.

*The system meets the requirements set down by the airline industry through the Aeronautical Radio, Inc. (ARINC), Characteristic No. 561

Table 1. Government LTN-51 users—current programs.

Agency	Aircraft	Mission	Systems
U.S. Air Force	VC-137	Air Force One	14
U.S. Air Force	C-135	COM SAT TAC	1
U.S. Air Force	C-130	Classified	23
U.S. Air Force	FB-111	TFRIS	1
U.S. Air Force	CV-880	ILS, ASD	1
U.S. Air Force	C-140	VIP fleet	1
U.S. Air Force	WC-130	Hurricane res	1
U.S. Air Force	C-135	VIP/FLT test	2
U.S. Air Force	C-97	Classified	2
U.S. Coast Guard	G-1159	Department of Transportation	1
Sandia Laboratories	C-135	Classified	1
LLL	Truck	Classified	1
MOT	Comet	Flight testing	2
EG&G	M-404	Classified	1
NASA	CV-990	Navigation/instrumentation	1
NASA	RB-57F	Earth resources	1
NASA	NP-3A	Earth resources	1
NASA	C-130	Earth resources	1
NASA	T-38	Space shuttle	1
NASA	C-440	ILS flight testing	1
U.S. Navy	C-130	Deep Freeze	2
U.S. Navy	P-3A/B	Aries	19
U.S. Navy	P-3A/B	Flight test	1
U.S. Coast Guard	C-130	Polar survey	1
NCAR	Saber	Atmospheric research	1
FAA	Saber	DME update/facilities calibration	1
U.S. Air Force	C-141	ASD/AWLS	1
U.S. Air Force	RB-47	FLAMR	1
SAMSO	C-135	Instrumentation	1
U.S. Air Force	B-737	Navigation trainer	19
Maritime Commission	Surface effect ship	R/D	1

Table 2. Government LTN-51 users—temporary and completed programs.

Agency	Aircraft	Program
MAC	C-141	AWADS, AWS, ARRS
SAC	KC-135	Command and Control
USAFE	EC-135	Command and Control
PACAF	EC-135	Command and Control
TAC	EC-135	Command and Control
USAFE	C-135	Airlift
U.S. Air Force	C-130	Holloman flight test
U.S. Air Force	F-102	Holloman flight test
U.S. Navy	YP-3C	Polar flight
FAA	CV-580	DME update program
CAF	Argus	ASW
U.S. Air Force	H-53	ARRS
NASA, Ames Research Center	Cessna 401	Earth resources

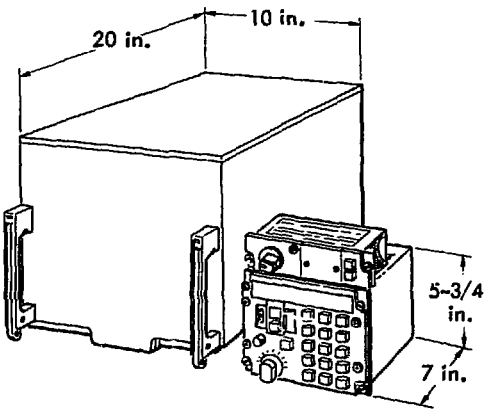

Figure 3. LTN-51 navigation system.

It was selected because of its availability (currently in production), price (much less than military versions), advanced design (incorporating gas bearing gyros and a digital computer), and performance.

The system is shown in outline form in Fig. 3. It consists of three units: the inertial navigation unit (INU), the control/display Unit (CDU) and the mode selector unit (MSU). The INU contains a four-gimbal platform and platform electronics, the digital computer, a digital subsystem, and a power supply.

Acceleration signals are transferred from the platform instruments to the computer via the digital subsystem, where gyro torquing compensations are computed and then returned to the platform. Pitch, roll, and heading are available for observation directly from the platform. The system power supply operates from single-phase 400-Hz power and 28-V dc power. In the event of a power failure, the system automatically switches to a backup battery.

The CDU, shown in Fig. 4, is used to enter initial position and to display navigation data generated by the system. The MSU is used for turning the system on and off and for switching the mode of the system from align to navigate. Self-testing and gyro biasing can also

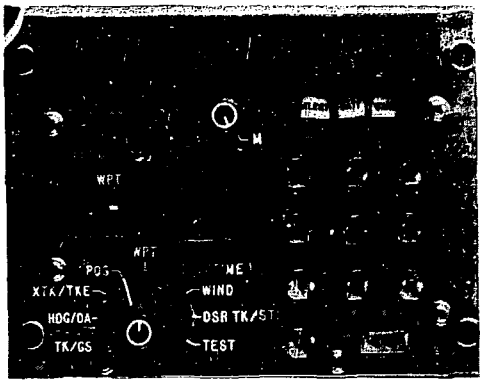

Figure 4. Control/display unit.

be performed through a switch on this panel.

Platform

The four gimbals of the LTN-51 platform are, from the innermost outward: azimuth, inner roll, pitch, and outer roll. The two gyros are Litton's G-1 floated, 2-degree-of-freedom gyros. The G-1 features a self-generating gas bearing on the rotor. The system contains three A-1 accelerometers. They are floated, torque-to-balance instruments, featuring dithered pivots which support the pendulum. Performance parameters for both instruments are given in Tables 3 and 4.

Digital Computer

The navigation system contains an Arma Micro-D computer located within the INU case. The characteristics of the machine are summarized below.

Operation:	Serial
Number system:	Binary, fixed point, fractional, twos complement
Word length:	18 bits
Clock frequency:	1.5 MHz
Instruction set:	18 commands
Add time:	18 μsec
Multiply time:	342 μsec
Memory:	4096 words, ferrite core
Weight:	5.25 lb
Volume:	0.09 ft^3
Power:	40 W

DDP-516 COMPUTER

The DDP-516 computer was used as an experimental device. It was chosen because of its interface flexibility, speed, and ruggedness and because the manufacturer could provide a computer facility convenient to programmers for degugging purposes. The machine was well suited for real-time operation. Major features of the computer are given below.

Operation:	Parallel
Number system:	Binary, fixed point, double precision option, twos complement
Word length:	16 bits
Cycle time:	0.96 μsec
Instruction set:	72 commands
Add time:	1.92 μsec
Multiply time:	5.28 μsec
Memory:	16,384 words, ferrite core
Weight:	>500 lb
Size:	$\approx$6 ft^3
Power:	1000 W

The DDP-516 would not be usable if the system were to be considered for actual field operation. However, there is a miniaturized version of the 516 that features a plated wire memory and is functionally identical to the 516. The machine was designed for use on the Air Force Gimballed Electrostatic Aircraft Navigation System (GEANS).

Table 3. G-1 gyro parameters.

Design parameters	
Type	2 degrees of freedom
Weight	1 lb
Size	3.1 in. diameter, 2.1 in. long
Motor type	3-phase hysteresis, synchronous
Motor bearing	Gas bearing
Pickoff	Moving coil differential transformer
Torquer	Voice coil
Float suspension	Cylindrical jewel, spherical pivot
Angular momentum	0.69×10^6 gm-cm^2/sec
Spin speed	24,000 rpm
Time constant	2000 sec
Transverse inertial of float	470 gm-cm^2
Runup time	<60 sec
Operating temperature	154 to 156°F
Operating temperature variation	±0.2°F
Flotation temperature	157°F
Performance parameters	
Drift	
Random drift	0.007 deg/hr (1σ) horizontal axis 0.01 deg/hr (1σ) vertical axis
Temperature-sensitive drift	0.01 deg/hr/g/°F
Non-g-sensitive drift	<1.0 deg/hr
g-sensitive drift	<8.0 deg/hr/g uncompensated <0.5 deg/hr/g compensated
g^2 drift	<0.1 deg/hr/g^2
Drift repeatability	0.007 deg/hr (1σ) horizontal axis 0.01 deg/hr (1σ) vertical axis
Pressure-sensitive drive	0.001 deg/hr/psi(1σ)
Alignment	
Torquer alignment	80 sec horizontal axis 80 sec vertical axis
Spin-axis alignment	40 sec level axis 40 sec azimuth axis
Float rotational freedom	>3 mrad

Interfaces

Several special interfaces were designed and added to the standard interfaces to meet the particular requirements of this program. The input/output addressing structure permitted up to 64 devices to be contacted, of which 13 were assigned, including 3 with priority interrupt status. These are given in Table 5.

The incremental magnetic tape interface performed the functions of buffering the information because of the difference in speeds between the computer and tape drive, and of formatting to reconcile the

Table 4. A-1 accelerometer parameters.

Design Parameters	
Weight	<200 gm
Size	$1.00 \times 1.135 \times 1.80$ in.
Operating temperature	$155 \pm 5°F$
Flotation temperature	$179 \pm 5°F$
Maximum acceleration	6 g
Float rotational freedom	11 ± 1 mrad
Performance	
Threshold acceleration	0.5×10^{-5} g
Axis alignment	± 50 sec
Scale factor linearity	$\pm 0.05\%$
Bias	$<200 \times 10^{-5}$ g
Bias stability (day to day)	5.0×10^{-5} g

Table 5. DDP-516 interfaces

Name	Address (octal)	Type	Interrupt
Paper tape reader	01	Standard	Standard
Paper tape punch	02	Standard	Standard
Incremental magnetic tape	10	Special	Standard
Teletype	04	Standard	Standard
ARINC	26	Special	Priority
Checkpoint	32	Special	Priority
Real-time clock	20	Standard	Standard
Console rupt	21	Special	Standard
Tilt indicator	43	Special	None
Motion sensor	43	Special	Priority
Mode change relay	56	Special	None
Precision angle indicator	64	Special	None
Fifth wheel	65	Special	None

computer word length to seven-channel IBM-compatible tape "bytes" or "characters." The computer transfers data from the A register to the interface in 2 μsec and holds it there until the tape unit is not busy. It then sends 6-bit "bytes" to the recorder at the rate of 1 byte/msec. The formatting is depicted in Fig. 5.

The ARINC interface sends and receives data between the inertial navigation unit and the DDP-516. Data are transmitted serially on a hardwired link

consisting of three twisted-pair lines: data, clock, and sync. Data are digital, in twos complement binary format. The data words are 32 bits long and are sent at a 13-kHz clock rate. The words are divided into 8 bits for address and 24 bits for data. Signal characteristics of the channel are shown in Fig. 6.

The tilt indicator interface was used to determine the output pulse rate from the navigation system accelerometers. The rate is an indication of the platform "tilt" when the system is stationary. A gyro torquing signal for releveling the stable member could then be more finely computed. This mode of operation reduced nonlinearities because of its smoothing effect and reduced limit cycling during damping phases in the alignment sequence. The interface accepted

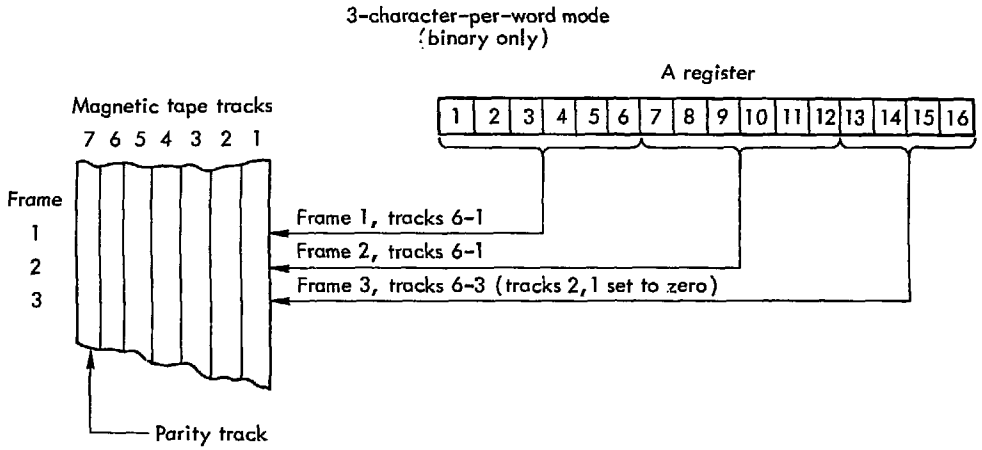

Figure 5. Conversion of DDP-516 words to magnetic tape format.

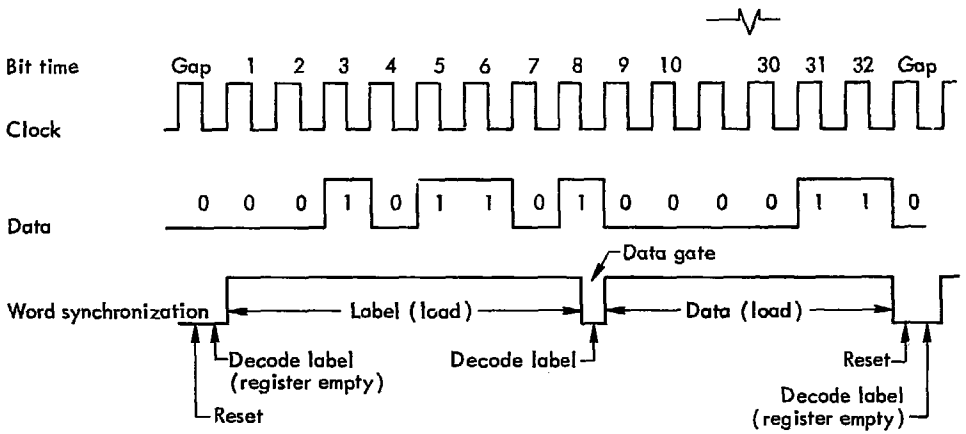

Figure 6. ARINC signal timing and characteristics.

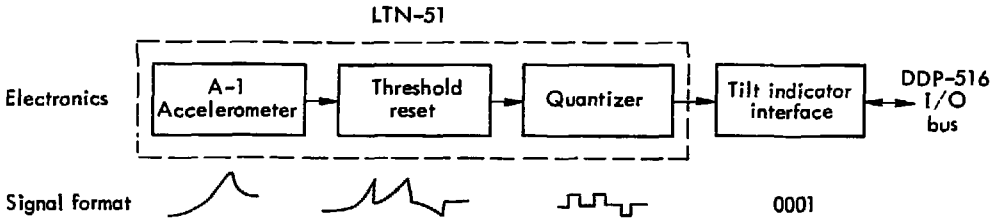

Figure 7. Tilt-indicator configuration.

18	Standby mode
17	Platform ready
16	Navigate mode
15	Display update
14	Memory display
13	Reset mode
12	Up to temperature
11	Velocity buffer overflow
10	Torquer error
9	Test
8	Accelerometer scaling
7	Platform orientation
6	Test
5	Test
4	Test
3	ARINC overflow
2	Auto fill control
1	Auto fill control

Figure 8. Micro-D discrete word.

quantized pulses (1/256 ft/sec/pulse) from the accelerometer north and east channels; its configuration is shown in Fig. 7.

The motion sensor interface was a variation of the tilt indicator. In the motion-sense mode, the counters in the interface were continually compared against threshold limits preset by thumbwheel switches. If either the north or the east velocity channel exceeded the threshold, a priority interrupt would be triggered, indicating the onset of motion. The counters were periodically zeroed out to prevent accumulated pulses due to gyro drifts from causing false triggering.

A special interface was created to actuate a mode control relay. An octal code word in the DDP-516 was used to command the relay to pull in or drop out. The relay was a mechanically latching variety, chosen to ensure that moding would remain unchanged in the event of a momentary power loss by the DDP-516. The relay was used to control the state of a particular bit in a Micro-D register containing bits for discrete functions. In this particular case, bit 13 was used, with the "1" condition meaning navigate and the "0" condition meaning calibrate. In this manner, moding was completely under control of the program in the DDP-516 computer. The mode could then be changed automatically by the motion sensor, by a programmed timer, or by the run operator via a teletype. The other discretes serviced by the Micro-D are displayed in Fig. 8.

Ancillary Equipment

The mobile test unit, shown previously in Fig. 2, served as the test bed for the GPL-A series of experiments. A number of studies were conducted to ensure proper operating conditions for the mobile laboratory. Among the problems considered were fifth-wheel speed resolution, noise levels, checkpoint accuracies, radio communications, critical loading, center of mass, interior layout, air-conditioning requirements, power budget, exhaust venting, power transfer between facility and generator, safety seats, and electronic equipment.

DINSMOBILE

A plan view of the DINSmobile is given in Fig. 9. Four passengers, including the driver, could be accommodated during runs. The vehicle itself was a standard delivery van outfitted locally with air conditioners, a motor/generator, electronic racks, lights, windows, auxiliary gas tanks, and acoustical tiles.

A cooling capacity of 40,000 btu/hr was required to adequately air-condition the van. The exterior was painted white and reflective overlays were put on the windows to help reduce heating by the sun, which was the main heat source during summer. The gross weight of the vehicle reached 13,000 lb; the design limit was 14,000 lb. Care was taken not to overload the axles. Equipment, passengers, and gasoline tanks were placed so as to balance the load.

POWER

A 15-kVA motor/generator was located in a separate shielded, fireproof, and insulated compartment in the rear. A specially designed plenum down-drafted the exhaust from its engine, thereby significantly decreasing the ambient temperature of the compartment. Approximately 80% of the rated power output was required. A bank of relays controlled by a synchronizing relay was used to transfer

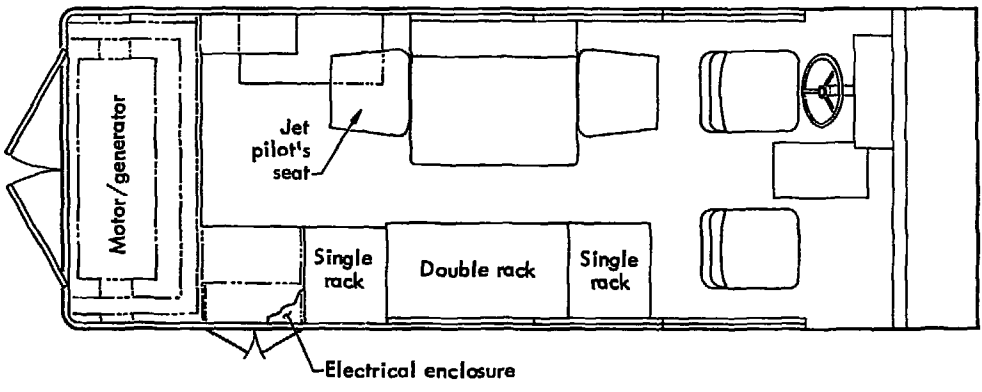

Figure 9. DINSmobile plan view.

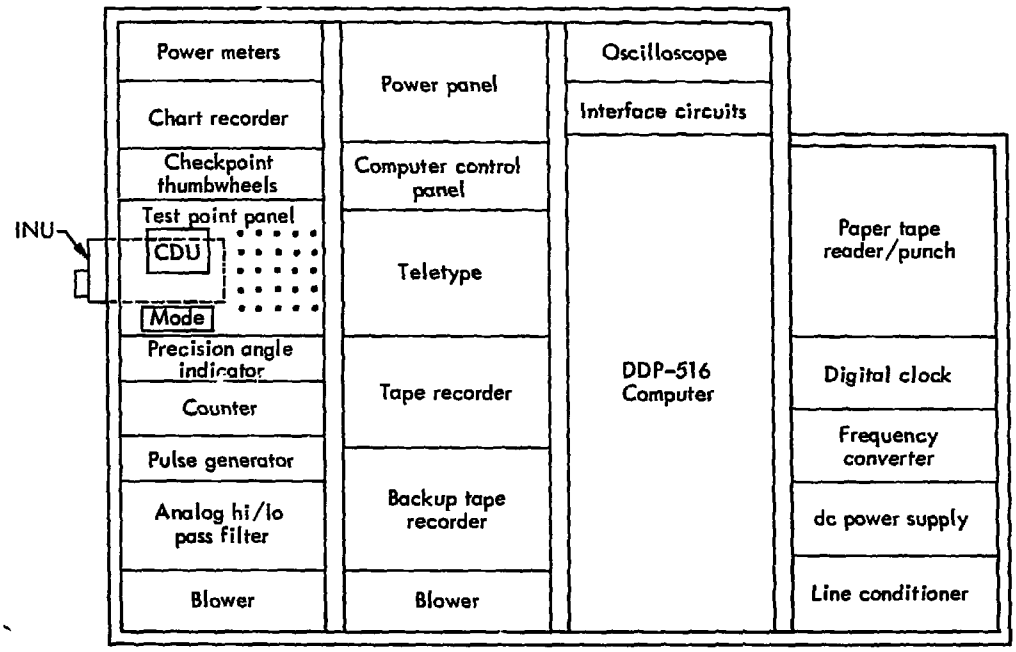

Figure 10. DINSmobile electronic rack layout.

power from the LLL facility to the motor/generator and vice versa without transients. Another relay scheme sensed momentary power interruptions and automatically restarted the DDP-516 program to continue any run in progress. The navigation system had a backup battery which ensured uninterrupted service for up to 15 min. The 115-V ac power had to be converted to 400 Hz to operate the automatic controller for the navigator heaters and to 28 V dc to operate the Micro-D computer and other electronics within the LTN-51. Other electrical equipment sensitive to power variations of the generator were serviced through a line conditioner. Interference from motors in the teletype and chart recorder was eliminated effectively with line filters.

ELECTRONIC EQUIPMENT

The electronic equipment necessary to support GPL-A testing was contained in the racks shown in Fig. 10 and an auxiliary rack shown in Fig. 11.

The functions of the items called out in Fig. 10 are given below:

Chart recorder	Continuous monitoring of V_x and V_y accelerometers
Checkpoint thumbwheels	Initiate record of present reference position
INU	Inertial navigation unit under test
CDU	Control/display unit (of LTN-51)
Mode	Selection of LTN-51 standby, align, or navigate mode
Test point panel	INU signals available for monitoring

Precision angle indicator	Fine azimuth resolver readout
Counter	Fifth-wheel speed indication
Pulse generator	Testing digital circuits
Analog hi/lo pass filter	Suppress selective vibration frequencies coupled into accelerometers
Power panel	Control power transfer between facility and generator
Computer control panel	DDP-516 operator's console: display of computer registers
Teletype	Initialization, memory dumps, quick-look data
Tape recorder	Data logging
Oscilloscope	Debugging circuits
Interface circuits	Communication between DDP-516 and external devices
DDP-516 computer	Control of inertial navigation executive computer system software
Paper tape reader/punch	Loading and modifying programs for DDP-516
Digital clock	Time reference
Frequency converter	400-Hz, 1-kVA, single phase power for LTN-51 heaters
dc power supply	28-V dc, 50-A power for LTN-51 digital circuits
Line conditioner	Regulated power for sensitive equipment

The auxiliary rack of equipment was used to load and debug programs for the Micro-D computer (see Ref. 3 for full details). Because space in the DINSmobile was limited and the auxiliary rack was used infrequently, it was housed in a building at LLL. It was connected to the Micro-D in the van by means of a 12-ft mating cable assembly. The unit provided the following capabilities:

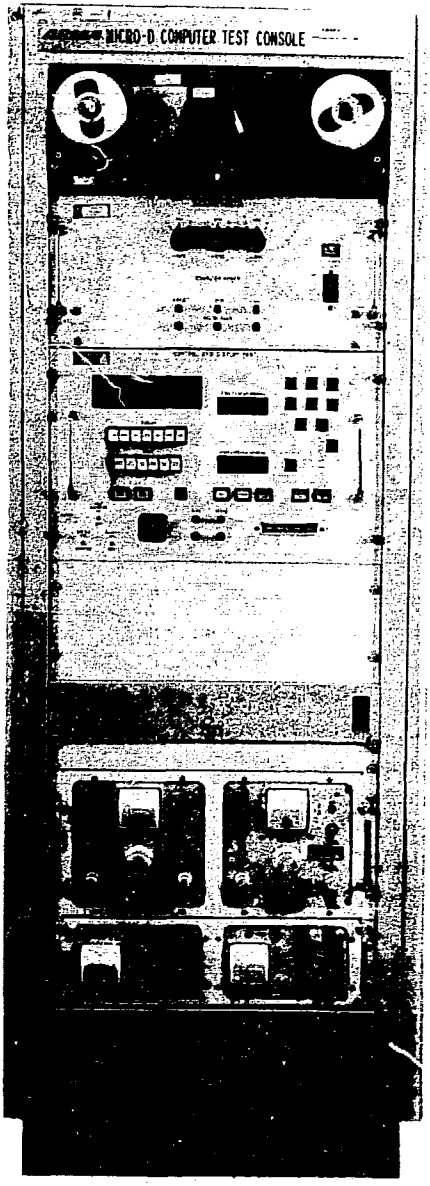

Figure 11. Micro-D computer test console.

- Display of computer registers.
- Loading memory via keyboard or paper tape reader.
- Single-step program control.

- Continuous readout of any memory cell during program execution.
- Test marginal power conditions of computer.

GPL-A Software

The programs developed to use with the INECS hardware are called the GPL-A software. These programs are written for two distinct computers, the Micro-D and the DDP-516. Although the two run in parallel, the DDP-516 program controls the system. They communicate via the "ARINC channel."*

The primary functions of the Micro-D computer program are to control the inertial platform and to calculate position and velocity information. The primary functions of the DDP-516 are error reduction, motion sensing, and data logging.

To explain how the GPL-A software works, we start by giving a brief overview of the functions of each computer's programs. Synchronization of the data flow between them is then explained, after which each computer's software organization is described in greater detail. The implementation of each error reduction program is discussed after the basic DDP-516 programs. The FORTRAN Inertial Navigation Equations System, which is different from the other GPL-A software, is discussed last. Finally, a section on software preparation is included to give the reader some perspective on the tools available for program preparation.

*A serial data transmission link specified in Aeronautical Radio Inc. (ARINC) characteristic No. 561-2 for Air Transport Inertial Navigation system, dated February 1, 1968.

FUNCTIONS OF THE DDP-516 PROGRAMS

The DDP-516 has several functions. One of these is to act as the executive of the system. It makes motion/no-motion decisions based on information it receives from its various peripheral devices (precision angle indicator, tilt indicator, etc.; see Fig. 12), and commands the Micro-D to damp or navigate. In this sense, the Micro-D could be considered the slave to the DDP-516. One of the main functions of the DDP-516 is to automatically calibrate the system during periods of no motion. It does this by "filtering" information it receives from the inertial platform, either directly from one of the peripheral devices or from the Micro-D computer over the ARINC channel, and sending reset information back to the Micro-D over the ARINC channel. Another of the functions of the DDP-516 is to gather data from the peripheral devices and to log it on the teletype and magnetic tape recorder for data reduction and analysis.

FUNCTIONS OF THE MICRO-D PROGRAMS

The Micro-D computer has two primary functions: to control the inertial platform and to provide data to the DDP-516. To control the platform, the software must provide a means of aligning the platform at start-up. After the platform is aligned, it must be kept level and

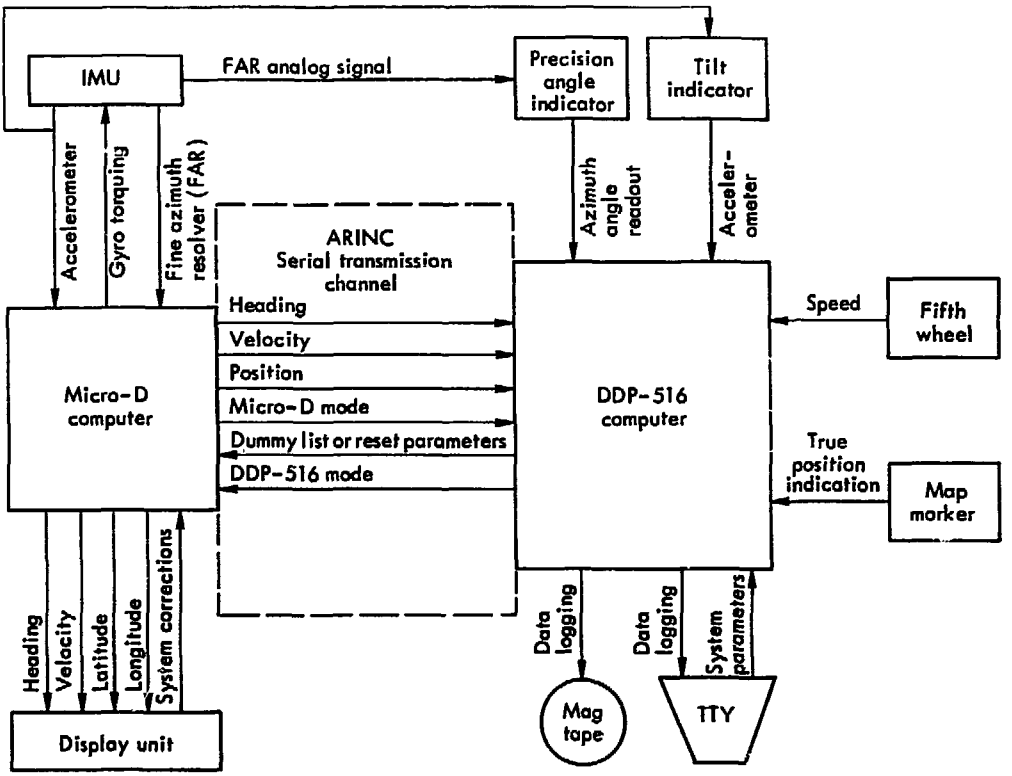

Figure 12. GPL-A system organization.

pointing north. The purpose of a navigator is to provide position and velocity information; after the alignment is completed, the navigator can perform that function. The position and velocity information then becomes the data needed by the DDP-516. This data is sent to the DDP-516 on the ARINC channel.

COMPUTER SYNCHRONIZATION

When two computers are involved in a complex real-time program such as GPL-A, one problem which can occur is that of synchronizing the flow of data between them. Synchronization was achieved in the GPL-A system through a combination of software and operator action. The best way to show how this was done is to go through the system start-up procedure.

In starting up the system, it is necessary to ensure that the desired programs have been loaded, the power is on, and the hardware is operable. The first step is to mode the LTN-51 to standby and enter the present position. This starts the Micro-D computer but does not start the alignment sequence. At this time, the Micro-D starts sending its ARINC output data to the DDP-516 once a second. The transmission takes approximately

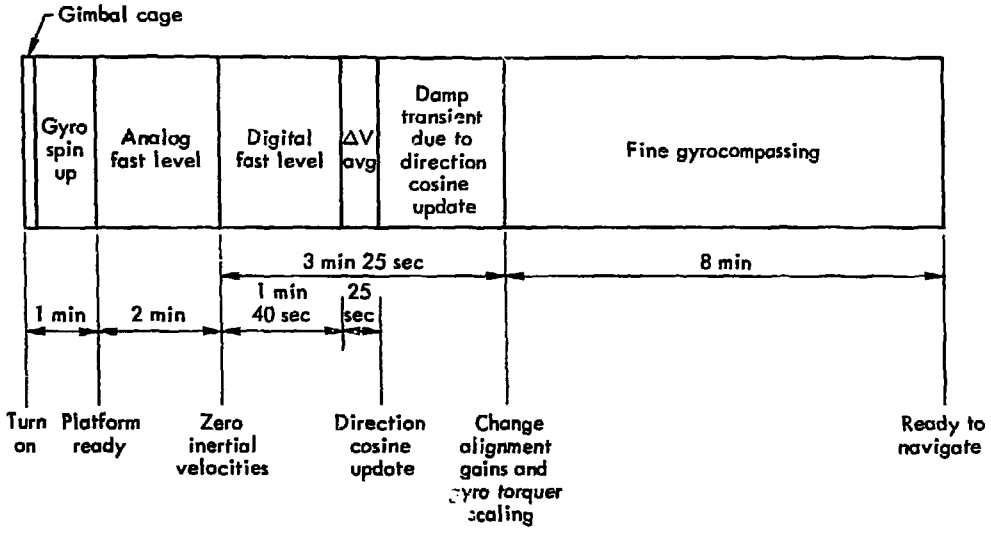

Figure 13. Alignment sequence.

0.3 sec, after which the Micro-D goes into a listening mode waiting to hear from the DDP-516; each second thereafter the cycle repeats.

At this point, the DDP-516 can be started. As part of its initialization, it waits to receive data from the Micro-D; 0.5 sec after receiving a "command word"* from the Micro-D, the DDP-516 will send a "dummy list"† back to the Micro-D. The computers are now synchronized, and a complete two-way transmission will take place every second, with the DDP-516 in the listening mode the first half second and in the transmission mode the second half second, and the Micro-D operating in a complementary fashion.

When the Micro-D is put in the alignment mode (this could be before starting the DDP-516), the normal alignment sequence starts. The timed sequence is shown in Fig. 13. At the time the LTN-51 has been moded to align and ARINC communication has been synchronized, the DDP-516 assumes executive control and the GPL-A is ready for use.

MICRO-D COMPUTER PROGRAM ORGANIZATION

To understand more thoroughly how the GPL-A software functions, it is necessary to consider the individual computer programs separately. First we will consider the Micro-D program.

Most real-time programs are organized around a background program (also called a background loop) and a foreground program. The background cycles

*A list consists of 15 words, which are sent at 10-msec intervals. A "command word" is the first word in the list.

†A "dummy list" is a list that contains no information for the Micro-D to act on. Its only purpose is to inform the Micro-D that the DDP-516 is operating.

continuously, doing things that can be done on a low-priority basis. The functions performed by the foreground, however, generally have to be done promptly or at a regular frequency. The program is informed of one of these conditions by an interrupt. When an interrrupt occurs, the normal flow of the program is broken and control is transferred to some known address. The program that takes the desired action is called an interrupt program or interrupt loop. After the desired action is taken, control returns to the point in the program where the interrupt occurred.

The Micro-D programs are organized around three interrupts (see Table 6): one that occurs every 50 msec, one that occurs every 10 msec, and an ARINC interrupt. Any interrupt will immediately interrupt any program in the machine, including another interrupt program. However, because the 50-msec interrupt uses such a large portion of the CPU time (see Fig. 14), it will allow completion of any ARINC or 10-msec interrupt program which it has interrupted. Except for this, the software services all interrupts on a last-in, first-out basis.

The functions that the background loop and the interrupt programs must perform are:

1. Align the inertial platform.
2. Calculate geographic position.
3. Keep the platform level and pointing north.
4. Provide data to the DDP-516 for error reduction and logging purposes.
5. Provide information on velocity, heading, and geographic position to the operator.

Table 6. Micro-D interrupt list.

Name	Frequency	Servicing time
50-msec interrupt	50 msec	25 msec
10-msec interrupt	10 msec	1 msec
ARINC	10 msec during first 300 msec of last half of every second	1 msec

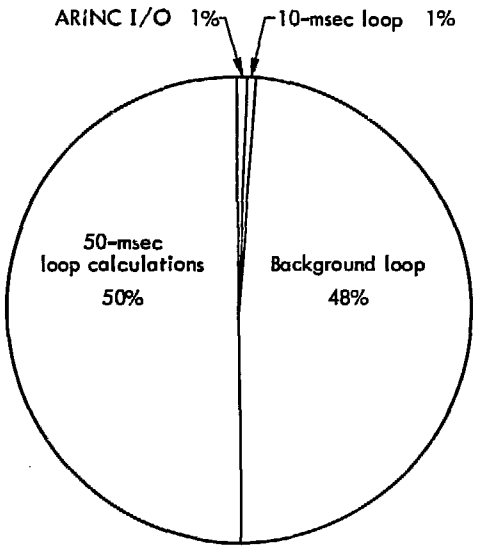

Figure 14. Micro-D CPU time use in the navigation mode.

The first three of these functions are provided by the 50-msec program, the fourth is provided by the 10-msec loop, and the fifth is provided by the background program.

The 50-msec interrupt program (see Fig. 15) is used initially to align the platform and, while in the navigate mode, to

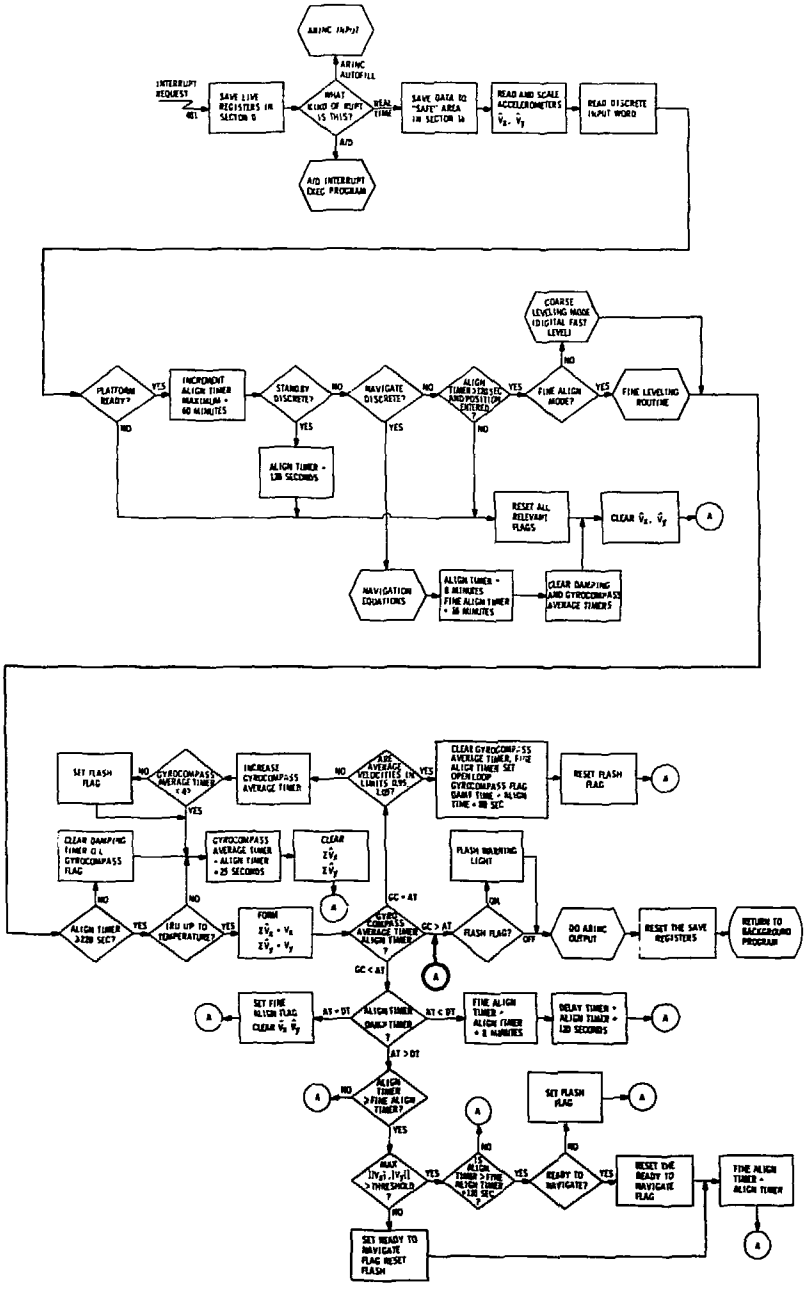

Figure 15. Micro-D 50-msec interrupt loop.

update the direction cosines used for calculating velocities, gyro torquing signals to keep the platform level, and geographic position. The equations as mechanized in the Micro-D are the same as in the basic commercial-aviation LTN-51 system, except that a fixed azimuth mechanization is used instead of the "wander" azimuth.[4] The 50-msec loop is also used to make corrections to position, velocity, and platform leveling when a reset list has been received over the ARINC channel.

The 10-msec interrupt program is used to read the accelerometers and fine azimuth resolver (FAR). It is also used to keep track of time and send ARINC transmissions of filter input data to the DDP-516 at regular intervals.

The ARINC interrupt program is used to receive data from the DDP-516.

When the program is not in an interrupt program, it is executing a program in the background loop. The background loop makes calculations that can be done on a no-priority basis. In the GPL-A this essentially consists of calculating latitude and longitude.[5] Background loop time is also used to service the display unit.

The display unit has two purposes. It displays the system position, velocity, and heading; it is also a means of making changes to Micro-D memory locations while the system is in operation.

As shown in Fig. 12, inputs to the Micro-D come from three different devices, the inertial measurement unit, (IMU), the display unit, and the ARINC channel. Inputs from the IMU are the accelerometer readings (north and east acceleration) and the fine azimuth resolver, which is an angular measure of heading. The inputs from the control/display unit are initial latitude and longitude at start-up. ARINC inputs (from the DDP-516) are used to indicate to the Micro-D that the DDP-516 is operating when a dummy list is sent. When a reset list (containing navigation update data) is sent, the information is used to make corrections to the gyro biases, velocities, and position.

Outputs from the Micro-D go to the same three devices. The outputs to the IMU are the gyro torquing signals. Sent to the display unit are current latitude, longitude, speed, and heading. Outputs to the DDP-516 over the ARINC channel are heading, north velocity, east velocity, latitude, longitude, and the direction cosines.[6]

The Micro-D core (memory) organization is shown in Fig. 16.

DDP-516 COMPUTER PROGRAM ORGANIZATION

All the error reduction schemes consist of filters programmed for GPL-A. Each required a somewhat different organization. Despite this, most functions which the DDP-516 had to perform were the same. The same basic system was set for all the different implementations. The basic functions of the DDP-516 software in the GPL-A system are:

1. Provide gyro bias estimates for the Micro-D.
2. Provide latitude and longitude error estimates to the Micro-D.
3. Determine whether motion is present or not and use this information to select the Micro-D mode.

4. Log desired data in the TTY and magnetic tape for later off-line analysis.

To perform these functions the foreground program uses six interrupts (see Table 7). Although the software services six interrupts, it is organized around two interrupts (the ARINC and clock interrupts) and the background program. All interrupts are organized on a last-in, first-out basis. This type of organization dictates that the amount of time a program spends servicing an interrupt be held to a minimum. In the DDP-516 software, the time to service all interrupts that could occur before finishing the first interrupt never exceeds 6 msec. No conflict results because the shortest interrupt period is 10 msec.

Table 7. DDP-516 interrupt list.

Name	Frequency	Servicing time
Clock	10 msec	4 msec
ARINC	10 msec during first 300 msec of any second	0.5 msec
Teletype	Random	0.5 msec
Magnetic tape	3 msec during data logging	0.1 msec
Checkpoint	Random	0.05 msec
Tilt indicator	Random	1 msec

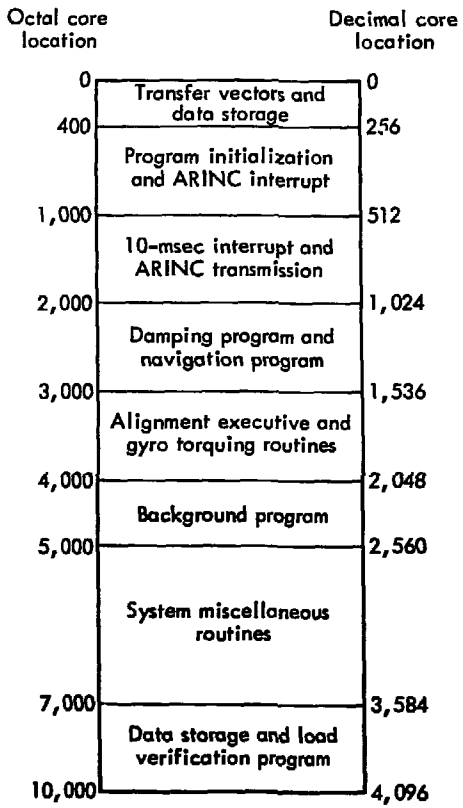

Figure 16. Micro-D core allocation.

The clock interrupt occurs every 10 msec and is used to perform all functions which must be performed regularly. These are:

1. Make ARINC transmissions to the Micro-D.
2. Check for missing ARINC inputs.
3. Make motion decisions.
4. Provide a time reference for logged data.
5. Indicate to the background program it is time to log data.
6. Indicate to the background program to start filter calculations.

Because of the number of functions it must perform, the operation of the DDP-516 program centers on the clock-interrupt routine. The first four of these functions are completed in the clock interrupt. Because of the computing time required to start the logging of data and

make the filter calculations, these functions could not be run directly off the clock interrupt. The program therefore sets a flag which will start the appropriate action when control returns to the background loop.

The other major interrupt is the ARINC interrupt. It receives data from the Micro-D that is used in the filter calculations and for data logging. Certain filter implementations require that the background filter calculation be synchronized to the ARINC interrupts. In RESET-16 (described in the following section), this is done by putting the background program in a loop once a minute, waiting for the "data in" signal to be received by the background from the ARINC interrupt program. When this happens, the filter calculations begin.

The primary functions of the background loop are to gather up data needed for logging and filtering and to make the logs and filter calculations. As can be seen from the figures in this section illustrating how CPU time is used (Figs. 14, 19, 20, 26, and 31), the program spends the majority of the time cycling through the background loop. For most of this time, it is waiting to make filter calculations or log data. Since one of the primary jobs of the background loop is to make filter calculations, each filter implementation requires a somewhat different background-loop approach.

Other interrupts serviced by the software are the checkpoint, tilt indicator, magnetic tape, and teletype. The teletype interrupt is used by the software to schedule teletype input and output efficiently. The magnetic tape interrupt is used to schedule magnetic tape outputs efficiently. The tilt indicator interrupt (used only by the suboptimal filter) is used to indicate the rapid onset of motion. The checkpoint interrupt is used to command a priority data log at a known position.

As executive for the complete GPL-A system, the function of the DDP-516 is to determine what mode both computers of the system should be in, and to take appropriate action to ensure that they are in the proper mode.* Acting in this capacity, the executive receives input from the teletype, clock, and motion sensor. The teletype allows the run coordinator to determine the initial mode and to change it as desired. The clock allows the mode to be changed as a function of time. Built into the system are a series of timers which allow the run coordinator to command mode changes at fixed intervals without being present. The motion sensors† allow the system to operate fully automatically. The executive determines whether the system should be in calibrate or navigate accordingly as the motion sensors indicate no motion or motion.

Since the implementation of each DDP-516 program depends on which filter is being used, a discussion of each filter follows. Details of the programs can be found in Refs. 7 and 8. The computer allocations for each filter are summarized in Table 8.

The large memory allocation for the 16-state filter in the GPL-A system is attributable to several factors. Since it

*The DDP-516 controls the mode of the Micro-D through the mode control relay (see Fig. 1).

†Motion sensing is described in Volume 5 of this report.

Table 8. Computer resources required for GPL-A (filter only).

Resource	16-state filter	Suboptimal filter
Memory (16-bit words)	7680	3072
Frequency of calculation	1 min	20 sec
Duration (of filter calculation)	15 sec	0.015 sec
Word size (for filter calculation)	48 bits	32 bits

was an experimental system the FORTRAN language was used to program the filter, which reduced coding efficiency. Calculations for the 16 states required combining three 16-bit computer words into 48-bit computational words for precision. This tripled the data storage space. In addition, the matrix manipulations for 16 states contributed to the increased memory size.

The suboptimal filter also required more than 16-bit precision. The next multiple of the computer word size was 32 bits, although 24 bits would have been sufficient.

The 16-State Filter (RESET-16)

The first error-reduction scheme programmed was the 16-state Kalman filter. The 16-state filter uses all of the general system described earlier except the motion sensor. In the case of RESET-16, the largest module (the Kalman filter) was implemented in FORTRAN, while the rest of the operational system was written in the DDP-516 assembly language. The use of FORTRAN enabled rapid development of the filter program and provided a program in which experimental changes could be made relatively easily. The use of assembly language on other parts allowed more efficient use of core and CPU time.

The background program for RESET-16 has four functions:

1. Make the Kalman filter calculations.
2. Make the system reset calculations.
3. Log data.
4. Make mode changes (navigate/calibrate).

It is organized around cyclically executed modules. Figure 17 is a general flow chart of the RESET-16 background loop. Figure 18 illustrates how memory is utilized, and Fig. 19 shows how the computer time is used. In the discussion on the background loop that follows, refer to Fig. 17.

When the filter module is called, it will perform one of three functions: initialize, extrapolate, or update. The symbols used in the discussion that follows are defined below*:

x	16-element state vector
P	16 × 16 covariance matrix
Φ	16 × 16 state transition matrix
Φ^T	transpose of Φ
H	3 × 16 measurement matrix
H^T	transpose of H
N	16 × 16 noise matrix
R	3 × 3 measurement noise matrix
z	3-element measurement vector

Initialization

When the initialization call to the filter is made, the Φ, P, x, R, N, and H matrices are set to their initial values.

*For more detail see Ref. 7.

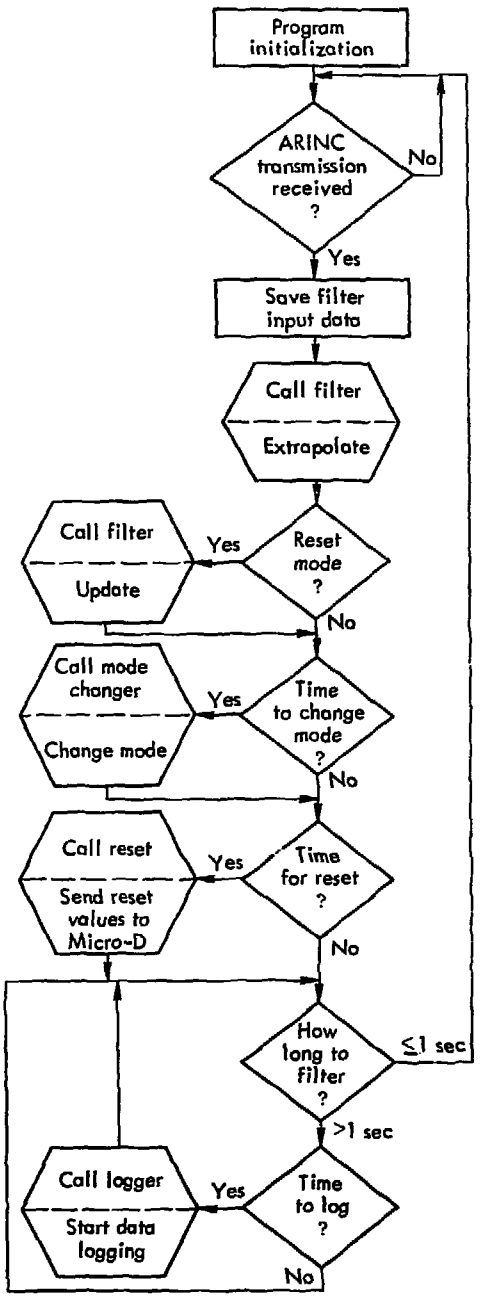

Figure 17. RESET-16 background program in the DDP-516.

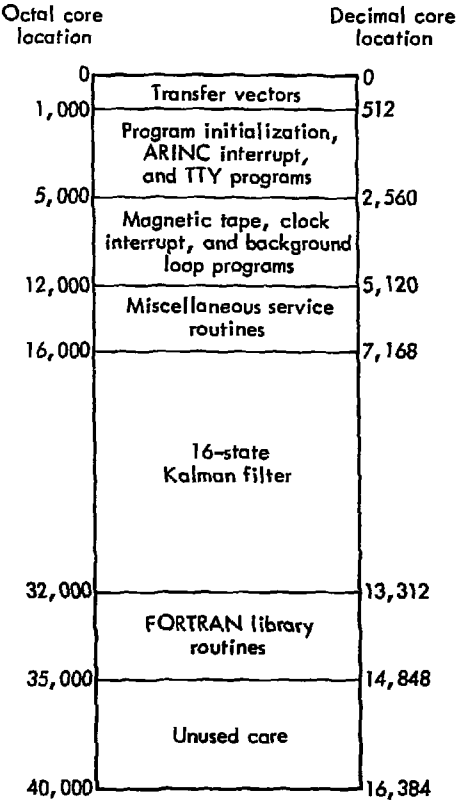

Figure 18. Sixteen-state Kalman filter core allocation.

Extrapolation

When the extrapolation call to the filter is made (this happens in both the navigate and calibrate modes), the Φ and N matrices are extrapolated to their new values according to the computed latitude. The equations solved are:

$$x_{n+1} = \Phi x_n,$$

(Extrapolation)

$$P_{n+1} = \Phi P_n \Phi^T + N.$$

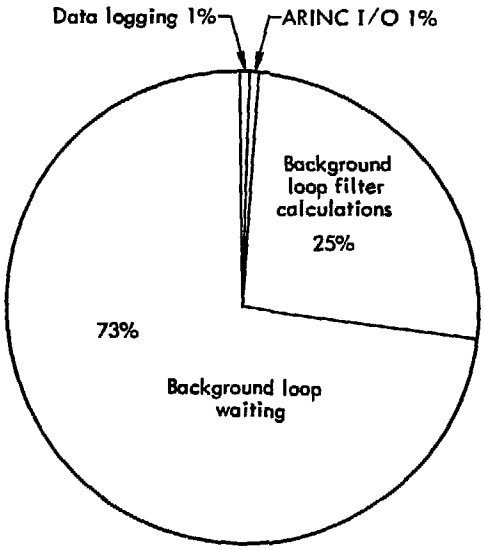

Figure 19. DDP-516 CPU time use during the filter mode with the 16-state filter.

Update

When the update call to the filter is made (this happens only in calibration), the x and P matrices are updated. The equations solved are:*

$$x_{m+1} = x_m + P_m H^T \times (HP_m H^T + R)^{-1}(z - Hx), \quad \text{(Update)}$$

$$P_{m+1} = P_m - P_m H^T \times (HP_m H^T + R)^{-1} HP_m.$$

Reset

When the reset module is called, it has three functions: calculate the reset values, prepare the reset values for ARINC transmission to the Micro-D, and update the state vector of the Kalman filter. The following notation is used in the reset calculations:

γ_r	gamma reset*
λ	latitude from Micro-D
λ_r	latitude reset
Ω	earth spin rate
x_i	ith element of the state vector (see Table 9)
V_{xr}	x velocity reset
V_{yr}	y velocity reset
α_r	alpha reset
β_r	beta reset
ℓ_r	longitude reset
B_{xr}	x bias reset
B_{zr}	z bias reset
R	radius of the earth
S_B	conversion factor, rad/sec to deg/hr
S_A	conversion factor to get resets into Micro-D units

The reset values are calculated according to the following equations:

$$U_x = x_9 + x_{12},$$

$$U_y = x_{10} + x_{13},$$

$$U_z = x_{11} + x_{14} + x_{15},$$

$$U_p = U_z \sin\lambda - U_x \cos\lambda,$$

$$V_r = U_y/(\Omega \cos\lambda) - x_8,$$

*The P_m of this call is the P_{m+1} from the extrapolation call.

*γ_r notation indicates the reset value in the DDP-516 program. γ_{rm} notation indicates the reset value in the Micro-D program.

$V_{xr} = -x_1$,

$V_{yr} = -x_2$,

$\alpha_r = x_6$,

$\beta_r = x_7$,

$\ell_r = x_5$,

$B_{xr} = U_p \cos\lambda$ for drift center resets
$= -U_x$ for state resets,

$B_{zr} = U_p(-\sin\lambda)$ for drift center resets
$= -U_z$ for state resets,

$\lambda_r = U_e/\Omega - x_3$ for drift center resets
$= -x_3$ for state resets.

The values prepared for transmission to the Micro-D will be calculated from

$V_{xrm} = V_{xr} \cdot R$,

$V_{yrm} = V_{yr} \cdot R$,

$B_{xrm} = B_{xr} \cdot S_B$,

$B_{zrm} = B_{zr} \cdot S_B$,

$\alpha_{rm} = \alpha_r \cdot S_A$,

Table 9. RESET-16 state vectors and measurement vectors.

State vector		
x_1	δW_N	North velocity error
x_2	δW_E	East velocity error
x_3	$\delta\lambda$	Total latitude error
x_4	$\delta\lambda_o$	Initial latitude error at start of error reduction mode
x_5	$\delta\ell$	Total longitude error
x_6	α	Vertical error about x axis
x_7	β	Vertical error about y axis
x_8	γ	Azimuth error
x_9	U_{xl}	x-gyro correlated random drift component
x_{10}	U_{yl}	y-gyro correlated random drift component
x_{11}	U_{zl}	z-gyro correlated random drift component
x_{12}	b_x	x-gyro constant drift component
x_{13}	b_y	y-gyro constant drift component
x_{14}	b_z	z-gyro constant drift component
x_{15}	U_{z2}	z-gyro random ramp drift component
x_{16}	K_z	z-gyro random ramp slope
Measurement vector		
1	δW_N	North velocity error
2	δW_E	East velocity error
3	$\Delta\gamma$	Change in azimuth angle since beginning of error reduction mode

$$\beta_{rm} = \beta_r \cdot S_A,$$

$$\gamma_{rm} = \gamma_r \cdot S_A.$$

The state vector will be updated according to the following equations:

$$x_1 = x_1 + V_{xr},$$

$$x_2 = x_2 + V_{yr},$$

$$x_3 = x_3 + \lambda_r,$$

$$x_5 = x_5 + \ell_r,$$

$$x_6 = x_6 + \alpha_r,$$

$$x_7 = x_7 + \beta_r,$$

$$x_8 = x_8 + \gamma_r,$$

$$x_{12} = x_{12} + B_{xr},$$

$$x_{14} = x_{14} + B_{zr}.$$

Data Logging

When the data logging program is called, it converts desired outputs to ASCII code for logging on the teletype and gathers up all data for recording on magnetic tape. After starting the logging on both the teletype and magnetic tape, control returns to the background program, and the appropriate interrupt routine will continue to service the output device.

Mode Control

The mode change logic is simply a timer which is updated by the real-time clock. When the mode change timer indicates that the desired amount of time has elapsed, the mode change program is entered. The mode change program will then change the system mode from navigate to filter or vice versa. Motion sensor control of moding for RESET-16 remained to be implemented at the completion of testing.

Filter Input/Output

Input measurements to the filter are the north and east velocities (system at rest) from the Micro-D, which are transmitted over the ARINC channel, and the azimuth angle readout from the precision angle indicator (PAI). Other inputs to the DDP-516 illustrated in Fig. 12 are used for data logging only.

Output from the filter is the updated state vector. This vector is used by the reset program at regular intervals (nominally one hour) during the calibration mode. This vector is then used to calculate resets which are sent to the Micro-D over the ARINC channel. The only outputs from the DDP-516, then, are the ARINC lists and data logger lists.

The Suboptimal Filter (RESET-3)

The second error reduction scheme implemented was the three-state suboptimal filter. This program was written completely in assembly language, although the FORTRAN library routines were used. Assembly language was used because motion sensing was included in this implementation, requiring faster response time than could be attained with a largely FORTRAN system. Figure 20 shows how the core was allocated to the different modules, and Fig. 21 shows how the CPU time was divided by the more important subprograms.

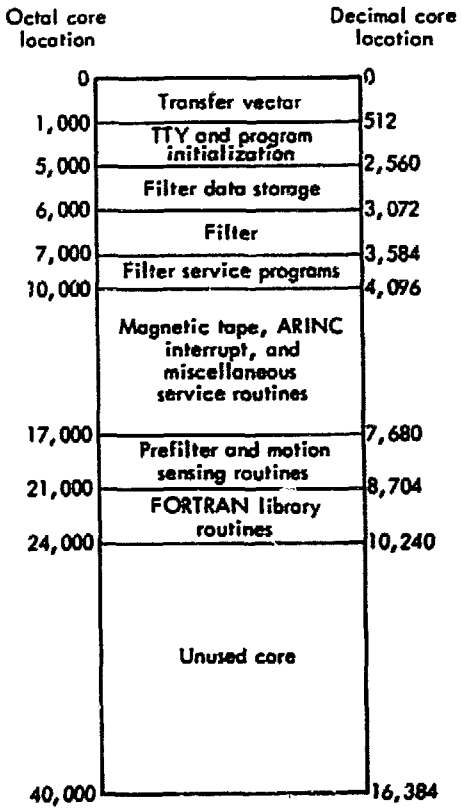

Figure 20. Three-state suboptimal filter core allocation.

RESET-3 as implemented uses all of the general DDP-516 system described earlier and in addition has a few unique features of its own. These features appear in three blocks of coding: the prefilter and motion-sensing block of the real-time clock interrupt program, the background loop, and the tilt indicator interrupt.

Prefilter and Motion Sensing

The prefilter and motion-sensing block is entered every 50 msec on command from the real-time clock interrupt. Figure 21 is a general flow diagram of this block of logic. This block has four primary functions: initialize variables when a mode change is made, determine if motion is present, make prefilter calculations in the calibrate mode, and command the Micro-D to change modes. A general discussion of each of these functions follows. For a more detailed discussion, see Ref. 9.

With the suboptimal filter, there are three DDP-516 modes and two Micro-D modes. The DDP-516 modes are DAMP, FILTER, and NAV. The Micro-D modes are DAMP and NAV. When the DDP-516 is in DAMP or FILTER, the Micro-D must be in DAMP. When the DDP-516 is in NAV the Micro-D is also in NAV. These modes can be changed in two ways, either automatically, depending upon motion-sensor decisions, or on a timed

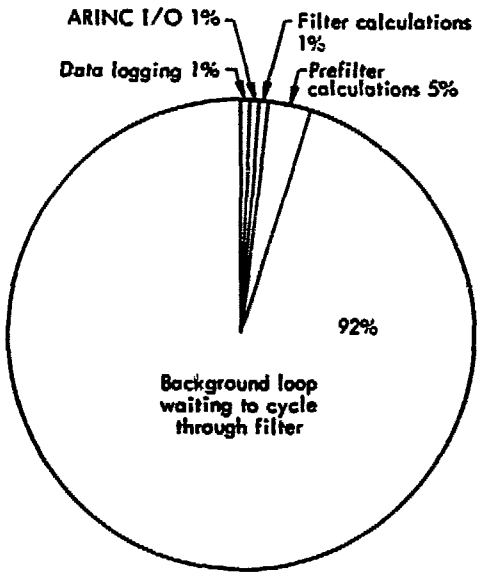

Figure 21. DDP-516 CPU time use during the filter mode with the three-state filter.

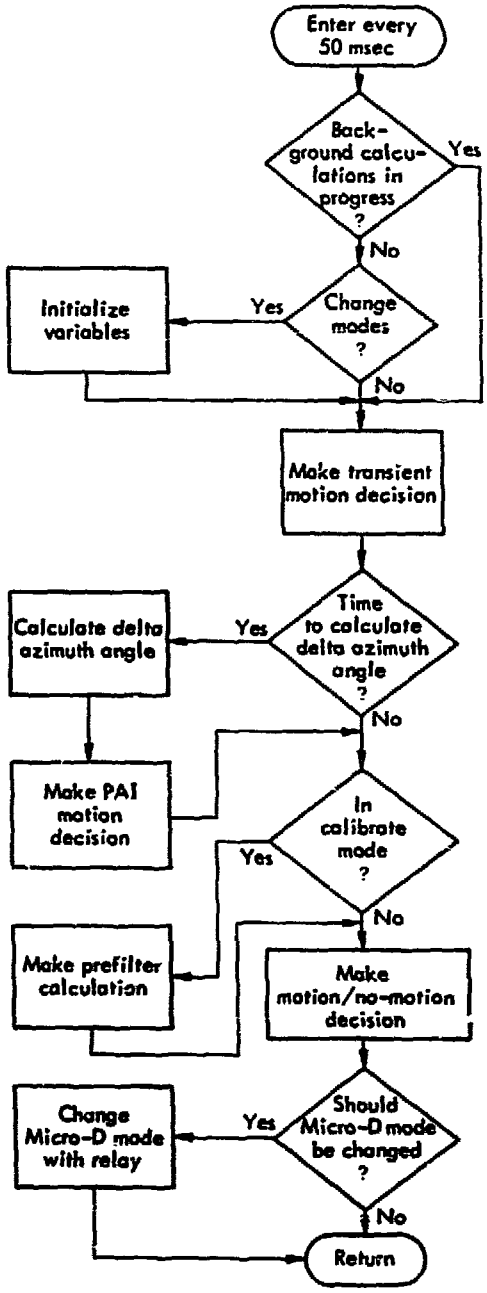

Figure 22. Prefilter and motion-sensing logic.

basis. The job of the mode-change logic is to determine if a mode change should be made and, if so, to make the change in both the DDP-516 and the Micro-D.

Since many variables must be changed when a mode change is made, another function of the real-time interrupt program is to make these changes. The two most important are the setting of motion-sensor thresholds and zeroing the prefilter outputs before the start of further calculations.

Two motion sensors, the transient and PAI motion sensors, are programmed into the real-time interrupt program. The transient motion sensor makes a decision every 20 msec by comparing the absolute value of the sum of the north (also east) velocity pulses* with a threshold determined by the DDP-516 mode. If this sum is greater than the threshold, the transient motion sensor indicates motion. The PAI motion sensor uses changes in azimuth angle to test motion.

The output of the PAI motion sensor is compared to a threshold which is a preset function of the DDP-516 modes. If the output is greater than the threshold, the PAI motion sensor indicates motion. The system will be assumed to be in motion if either the PAI or transient motion sensor indicates motion.

The prefilter calculations are made every 50 msec in the FILTER mode. The equations are shown below:

$x_1(n)$ intermediate state variable

*Each velocity pulse represents an increase or decrease of z^{-8} ft/sec. For example, 2^8 pulses means a change of velocity of 1 ft/sec.

$x_2(n)$ output of the prefilter

T time increment

τ_p prefilter time constant

y_1 obtained from a linear combination of north and east velocities

y_2 azimuth angle

$$x_1(n) = x_1(n-1)\exp(-T/\tau_p) + \left[1 - \exp(-T/\tau_p)\right] \times \left[y_1(n-1) - \frac{y_2(n-1)}{\tau_p}\right];$$

$$x_2(n) = \exp(-T/\tau_p)\left[x_2(n-1) + \frac{T}{\tau_p}\, x_1(n-1)\right] - \frac{T}{\tau_p}\exp(-T/\tau_p)\left[y_1(n-1) - \frac{y_2(n-1)}{\tau_p}\right] + \left[1 - \exp(-T/\tau_p)\right] y_1(n-1).$$

A block diagram of the prefilter is shown in Fig. 23.

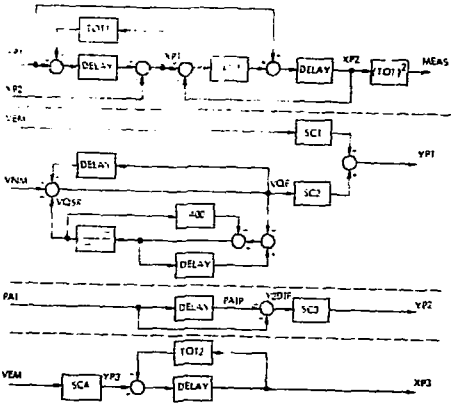

Figure 23. Digital prefilter.

Background Loop

A general flow chart of the background loop for the suboptimal filter is given in Fig. 24. The primary functions of this loop are to prepare for mode changes, make filter calculations, make reset calculations, and start data logging. The cycle time (set in the initialization program of the real-time interrupt) is nominally 36 sec in the DAMP and FILTER modes and 6 min in the NAV mode.*

This system combines the three-state error reduction with motion sensing. Because of this, the background loop must change modes instantly and still be able to function correctly in an error-reduction sense. For this reason the mode-change logic requires the background program to prepare to change modes. As an example, consider the case where the system has been in calibrate for a considerable period of time but no resets have been made. If motion takes place at this point, valuable reset information would be lost unless some method of saving this data were programmed in. It is this type of problem that the "prepare to change modes" logic handles.

Three-State Kalman Filter

The filter calculations can be broken into four matrix equations. These are the extrapolate covariance matrix equation, the Kalman gain equation, the update state vector equation, and the update covariance matrix equation. The equations

*The cycle time is the frequency with which the background program starts again at the beginning; see Fig. 24.

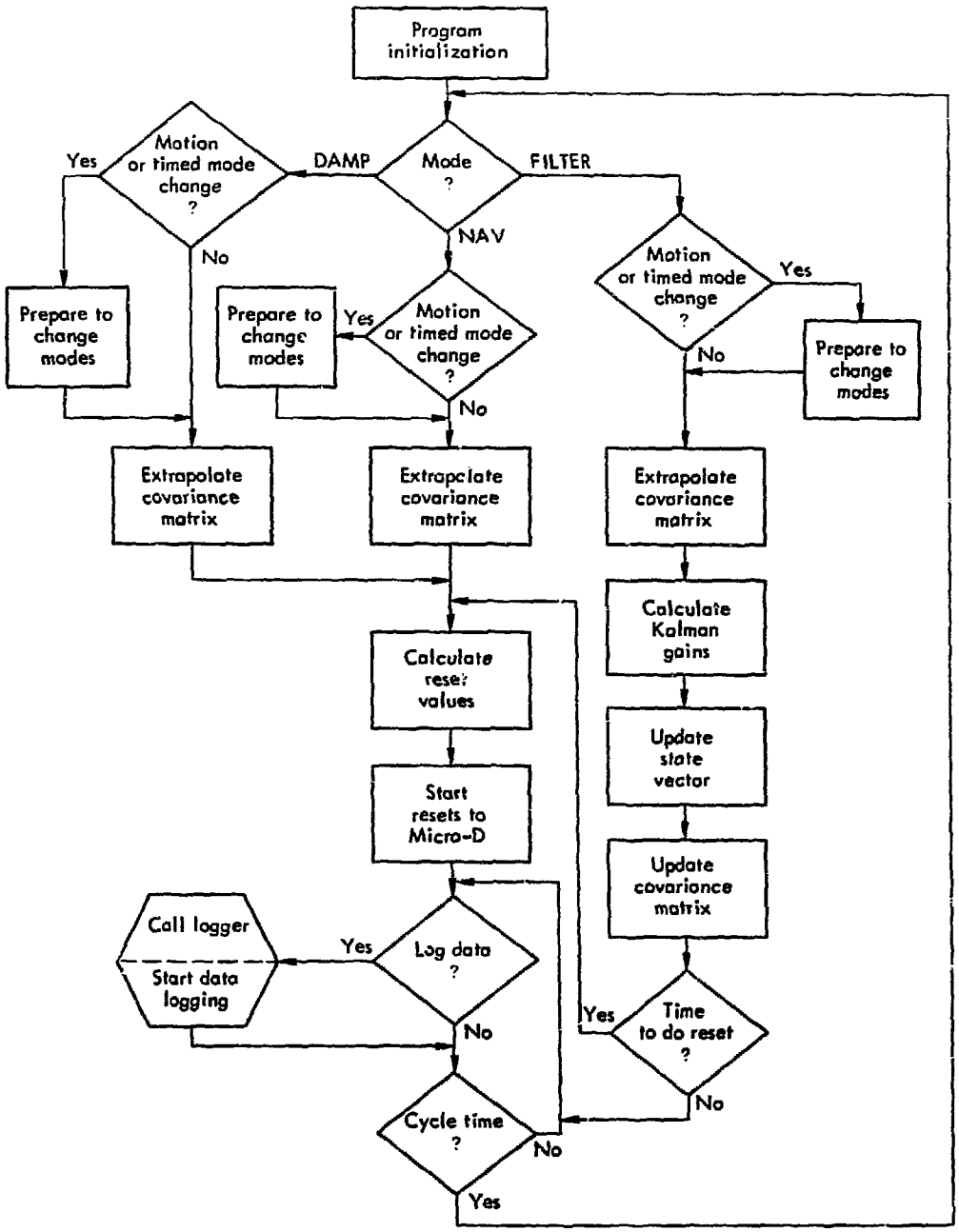

Figure 24. Background loop for the suboptimal filter.

given below are the equations which are mechanized in the DDP-516.* The symbols used are:

P	covariance matrix
Q	process noise matrix
R	measurement of noise matrix
v	measurement vector (actually the $x_2(n)$ of the prefilter)
I	identity matrix
K	Kalman gain matrix
$\hat{X}$	state vector estimate
Φ	state transition matrix
t	time
τ_s	nominal time increment
T_s	cycle time

The extrapolate covariance matrix equation is:

$$P_n = \Phi_{n-1} P_{n-1} \Phi_{n-1} + Q. \quad \text{(Extrapolate)}$$

where

$$\Phi = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & \exp(T_s/\tau_s) \end{bmatrix} \text{ in DAMP and FILTER,}$$

and

$$\Phi = \begin{bmatrix} 1 & -T_s & -\tau_s[1 - \exp(T_s/\tau_s)] \\ 0 & 1 & 0 \\ 0 & 0 & \exp(-T_s/\tau_s) \end{bmatrix} \text{ in NAV.}$$

The Kalman gain equation is:

$$K_n = P_n H_n^T \left(H_n P_n H_n^T + R_n\right)^{-1}. \quad \text{(Kalman gain)}$$

The update state vector equation is:

$$\hat{X}_n = \Phi \hat{X}_{n-1} + K(V - HQ\hat{X}_{n-1}). \quad \text{(Update)}$$

The update covariance matrix equation is:

$$P_n = (I - K_n H_n) P_{n-1}.$$

Note: P_{n-1} is the P_n of the extrapolate covariance matrix equation.

The purpose of the reset program is to calculate the actual reset quantities. These quantities are passed to the ARINC transmission program and sent to the Micro-D. The nominal reset interval during the FILTER mode is 20 min.

Data Logging

The data logger module gathers up pertinent data from the various external devices. The data is passed to the teletype logging program, where it is converted to ASCII format.* The ASCII data output is started on the teletype. The data is also passed to the magnetic tape logger where tape logging is started. The program relies on the magnetic tape and teletype interrupt to continue the data loge.

The Velocity Averager

The third error reduction technique programmed on the INECS hardware was the velocity averager. Essentially, the velocity averager is a simplified version of the suboptimal filter. It uses the same real-time logic (e.g., prefilter). In the velocity averager, no motion sensing

*The equations are not programmed as matrix equations. For a list of the equations as programmed, see Ref. 9.

*ASCII is the standard teletype code.

logic was included. The program uses all of the GPL-A support software described earlier. The primary difference between this program and the suboptimal filter program is the background loop illustrated in Fig. 25. The four functions of the background loop are to initialize variables at mode change time, sum the measurements, calculate the reset values, and log data.

The mode changes in this program always take place in the cycle DAMP to FILTER to NAV to DAMP. When the mode changes from DAMP to FILTER, the measurement summer is zeroed. When it changes from FILTER to NAV, reset values are calculated and sent to the Micro-D. When it changes from NAV to DAMP, prefilter parameters dependent on position are calculated. As is apparent, the function of this logic is to prepare the program for operation in the new mode. As with the suboptimal filter, the actual mode change will take place in the real-time interrupt.

The velocity summer is actually the sum of the output $x_p(n)$ [MEAS].* It is calculated from

$$X(n+1) = X(n) + x_p(n),$$

or,

$$X = \sum x_p(n). \qquad \text{(Velocity summer)}$$

The value used to calculate resets is

$$E_p = \frac{X}{\Delta t}. \qquad \text{(Velocity average)}$$

*MEAS is the polar gyro measurement used by the Kalman filter (see Ref. 10).

where X is the summation of MEAS and Δt is the length of time the filter has been running.

Data logging takes place in the same fashion as data logging in the suboptimal filter. There is a different logging list, however.

The velocity averager was programmed but never fully debugged. This program was not completed because simulations had shown it to be inferior to the suboptimal filter, and implementation would have required nearly as much of the resources of the DDP-516 as the suboptimal filter. This can be seen by comparing Fig. 21 with Fig. 26 and Fig. 20 with Fig. 27.

FORTRAN INERTIAL NAVIGATION EQUATIONS SYSTEM

The FORTRAN Inertial Navigation Equations System (FINES) is somewhat different from the other GPL-A software described above. It was written to enable analysts to study the behavior of the LTN-51 more closely. Accordingly, the decision was made to write the program as much as possible in FORTRAN. (The entire program could not be written in FORTRAN, however, because of the need to operate special hardware.) The use of FORTRAN enabled quick programming changes and use of a substantial library of existing routines. These routines include floating-point arithmetic and the standard FORTRAN trigonometric and exponential packages.

To derive maximum benefit from the use of FORTRAN, it was necessary to use it in both the interrupt and background loops. This brings up the so-called reentrant problem, which occurs when a

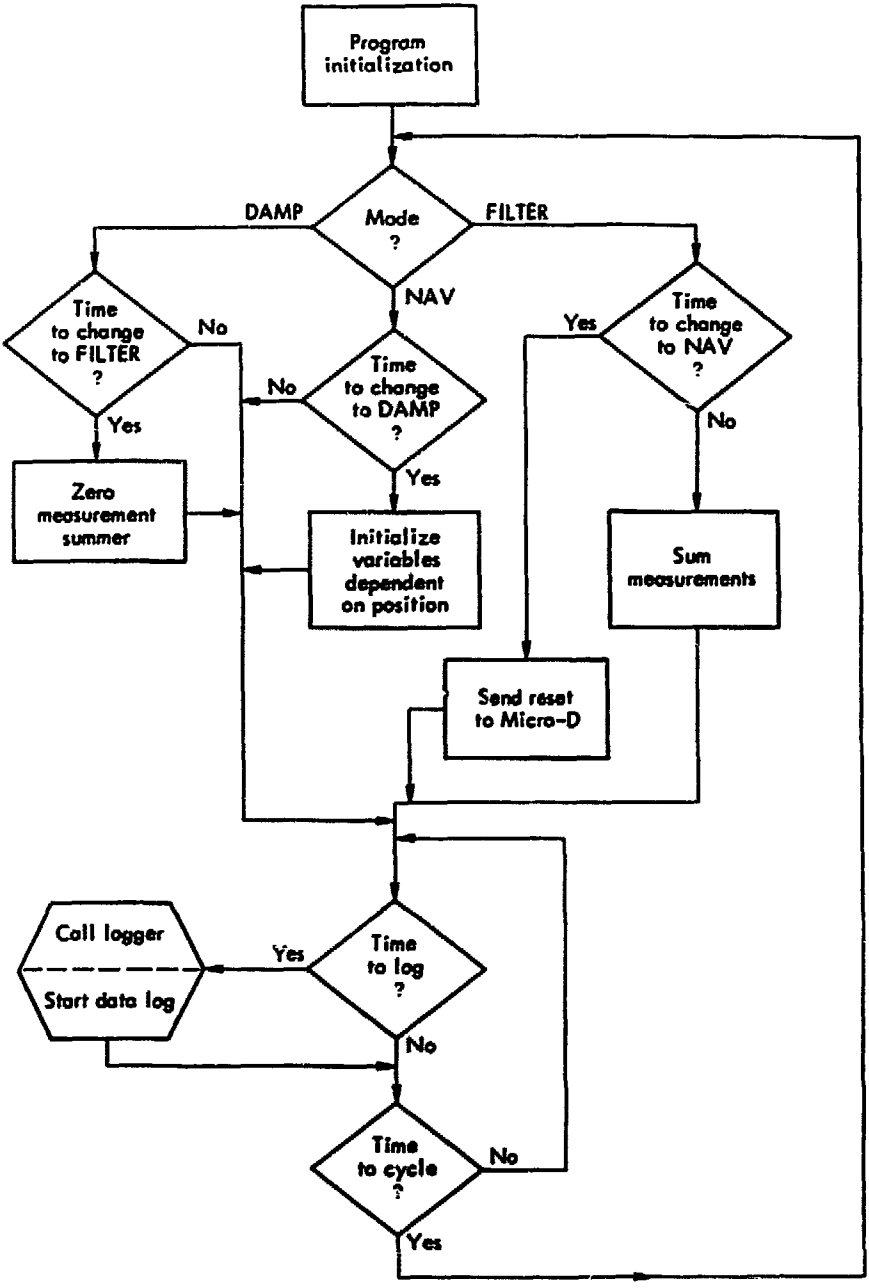

Figure 25. Velocity averager background loop.

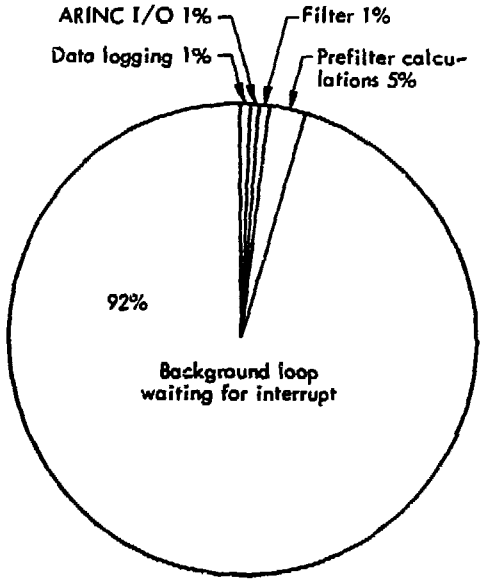

Figure 26. DDP-516 CPU time use during the filter mode with the velocity averager.

subprogram is being executed and an interrupt program calls the same subprogram. Since interrupts normally are organized on a last-in, first-out basis, the call to the subprogram will be satisfied first and will operate correctly; however, after completion of the interrupt, control will return to the same subprogram. When this happens, any variables referenced in the program will have the values left from the interrupt call and not the values calculated during the first call. This will of course cause erroneous results to be returned by the subprogram. There are two ways to solve this problem: either write a re-entrant code or else load multiple copies of those routines where this might happen. In the FINES system, the latter approach was taken; hence Fig. 28 shows two copies of the FORTRAN library.

In the FINES system, the navigation equations were removed from the Micro-D computer. In the NAV mode the Micro-D reads the accelerometers every 50 msec and sums them for 1 sec. At the end of this period, the Micro-D sends these summed values to the DDP-516 over the ARINC channel. The DDP-516 uses these numbers to calculate position and gyro-torquing signals. The torquing signals are then sent back to the Micro-D over the ARINC channel and are used to torque the gyros during the next second. In this system, the program is put into the NAV mode by entering a command in

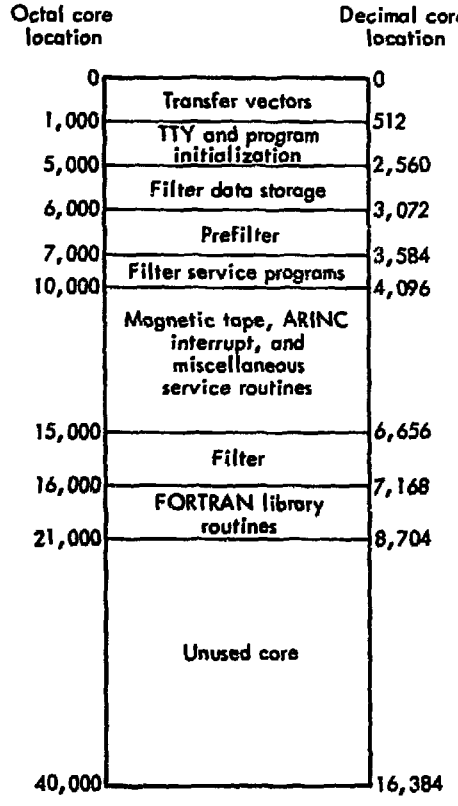

Figure 27. Velocity averager core allocation.

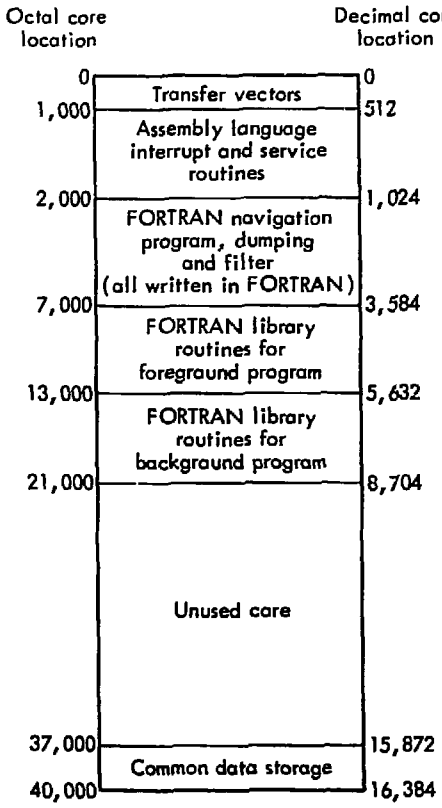

Figure 28. FINES core allocation.

the Micro-D via the control/display unit, and the Micro-D program synchronizes the computers by sending the accelerometer inputs to the DDP-516 once a second.

The operation of FINES can best be understood by considering the function of each computer in the alignment and navigation modes. While in the alignment mode, the Micro-D employs the normal LTN-51 fixed-azimuth alignment sequence, except there is no ARINC transmission taking place. When the platform is aligned, the READY NAV light will come on. When the system is put in NAV, it starts summing the accelerometers and ARINC transmission. The receipt of the y-accelerometer causes the DDP-516 to start the navigation calculations.

The DDP-516 program is organized around two modules: the ARINC interrupt loop and the background loop. Since these two modules are loaded essentially as two separate programs, all communication between them is via the calling sequence to the FORTRAN NAV program and through labeled common blocks.

The ARINC input program is shown in Fig. 29. It shows how the different parts of the DDP-516 program fit together and how the FORTRAN portions interface with the assembly language portions. The program shown in Fig. 29A is written in assembly language, and that shown in Fig. 29B is written in FORTRAN. The FORTRAN block has four distinct functions: 1) determine which mode the DDP-516 requires (at this point, the Micro-D has no mode; it is simply summing accelerometer readings and torquing the gyros as determined by the DDP-516) and make the necessary calculations; 2) make any error reduction calculations; 3) indicate to the background program to log on the teletype; and finally 4) to start logging on magnetic tape.

As mentioned earlier, the primary function of FINES was to provide analysts with a tool for studying the system. The design of the ARINC interrupt block allowed a quick and easy implementation of motion sensing. When the motion sensor determined that motion was present, the navigation calculations were made; when no motion was present, the damping calculations were made. The use of FORTRAN permitted the implementation of several different damping and

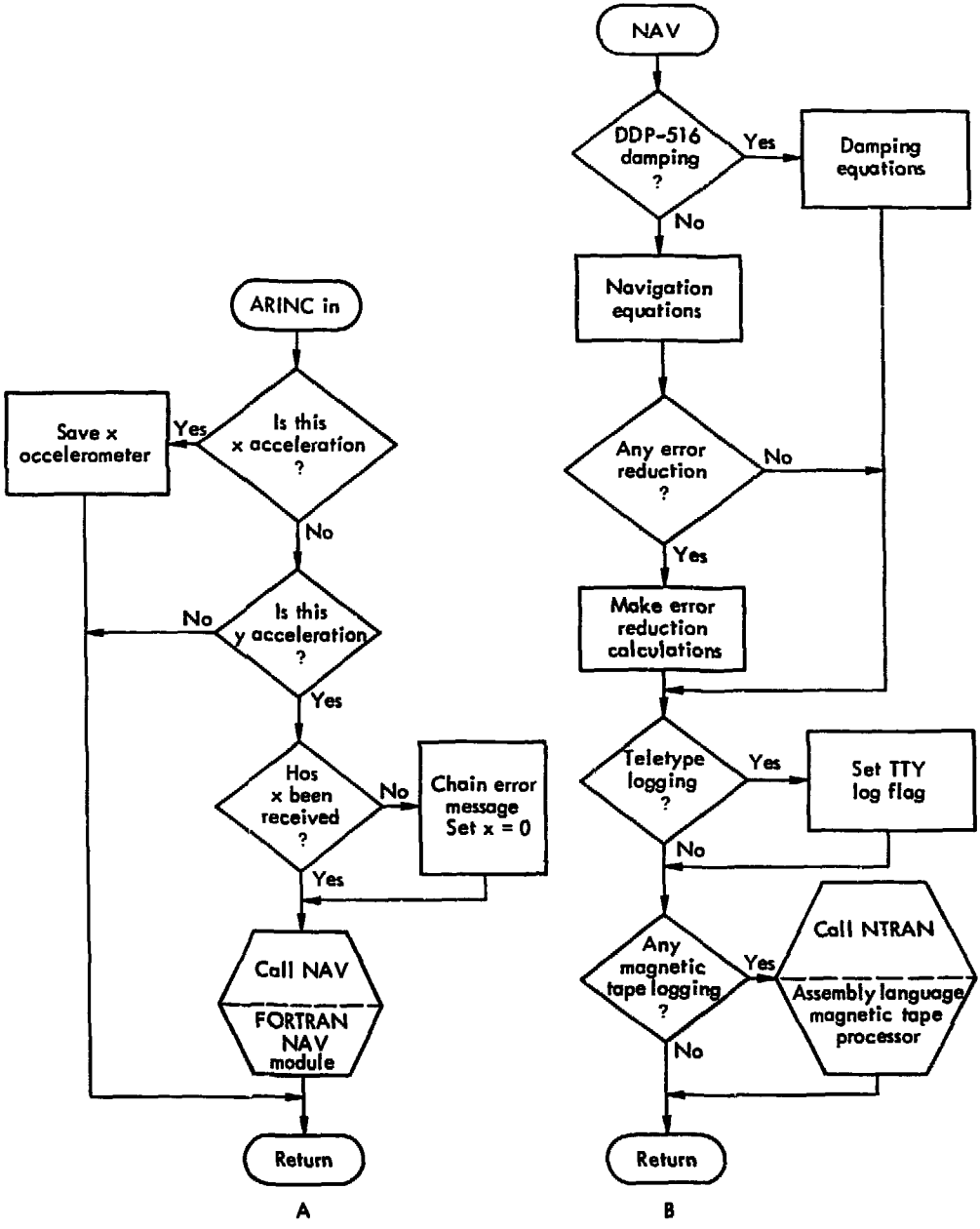

Figure 29. FINES ARINC input interrupt.

navigation calculation techniques. In general, they followed the basic equations described in Ref. 1.

The FINES program provided a place in the ARINC interrupt program to make error reduction calculations. Since it was in the ARINC interrupt program, error reduction calculations were made with the same frequency as the navigation or damping calculations. In one case, this logic permitted the development of a deterministic method of calculating gyro drift rates.

The ARINC interrupt program logic permitted data logging on the teletype and on magnetic tape. Logging data on magnetic tape in the ARINC loop was very advantageous; since it permitted logging every call to the program (once a second), various system parameters could be studied in detail.

The FINES program serviced two other interrupts, the clock interrupt and the magnetic tape interrupt. The clock interrupt's sole function was to indicate if a transmission from the Micro-D was lost (e.g., if more than 1.2 sec had passed without receiving y-accelerometer data). The magnetic tape interrupt permitted a more efficient use of CPU time. Using this interrupt allowed the software to keep the magnetic tape output going after it had been started in the ARINC interrupt logic.

The background loop for the FINES program is shown in Fig. 30. The background has three functions: log data on teletype, calculate resets, and allow operator modifications to memory locations. The background cycles continuously, waiting to perform these functions. Figure 31 shows how CPU time was used in the FINES system.

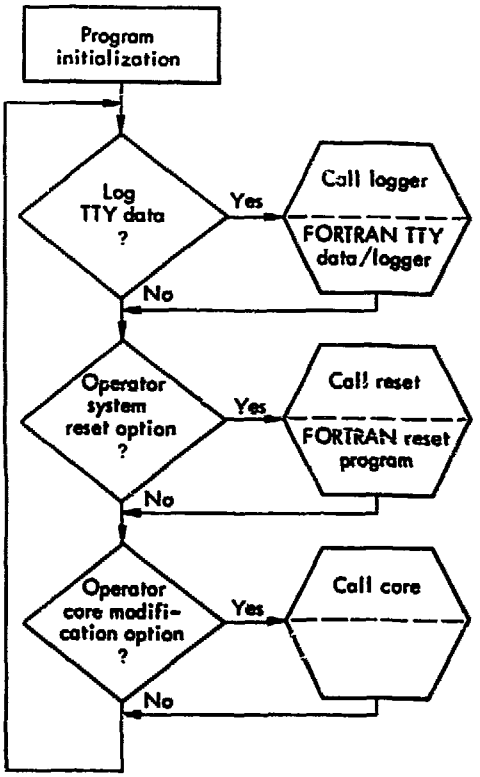

Figure 30. FINES background loop.

SOFTWARE PREPARATION

Executable instructions for any computer can be generated in two ways. One way is to write programs in machine language and enter them into the computer via the console. This method is satisfactory for very short programs only, since it becomes exceedingly difficult for the programmer to correlate all data words and operations when programs grow large. Also, the problem of entering erroneous instructions soon becomes overwhelming.

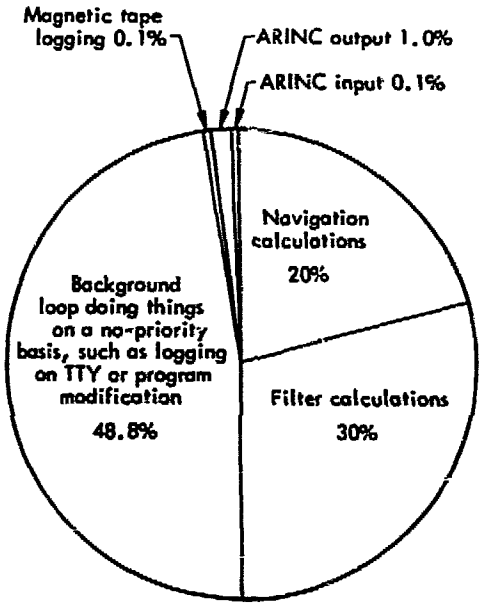

Figure 31. DDP-516 CPU time use in the FINES system.

The other way is to write these programs in a format more convenient for the programmer and to allow the computer to generate the executable instructions. This is accomplished on the Micro-D through the use of an assembler and on the DDP-516 through the use of the DAP-16 assembler or the FORTRAN compiler.

The Micro-D is a small computer with very limited input/output capacity. Programs must be prepared on a larger machine. An assembler was written which runs on an IBM 360 computer. The assembler accepts input from cards and punches a paper tape in the appropriate format for loading the computer through the Micro-D test console.

The DDP-516 is a much larger and more versatile machine than the Micro-D. All the programs used in generating executable instructions for the DDP-516 can be processed by the machine itself. The names of the programs used in generating DDP-516 software are SSUP, DAP-16 assembler, FORTRAN compiler, LDR, APM, and PAL-AP.*

The DAP-16 assembler and FORTRAN compiler used in the GPL-A system receive punched paper tape as input and generate relocatable code that is punched on a paper tape by the high-speed paper tape punch. This output is called an object tape. To generate executable code, another program, called a loader, is used. In the DDP-516 system, this program is called LDR-APM. The loader will load and set up the necessary linkage for one or several independently compiled programs. This is the desired executable program. At this point, PAL-AP can be used to generate a self-loading (generally referred to as a bootstrap) tape. Once a bootstrap tape has been created, LDR-APM can be bypassed in subsequent loads. The procedure for creating a production program is illustrated in Fig. 32.

Once a source program exists on paper tape, it often becomes necessary to alter it. Since recreating the source tape on an off-line ASR would be a time-consuming process, a program called SSUP exists to assist in this process. SSUP receives as input a paper tape in some source language (DAP-16 or FORTRAN) and input

*For details of usage, see Ref. 11.

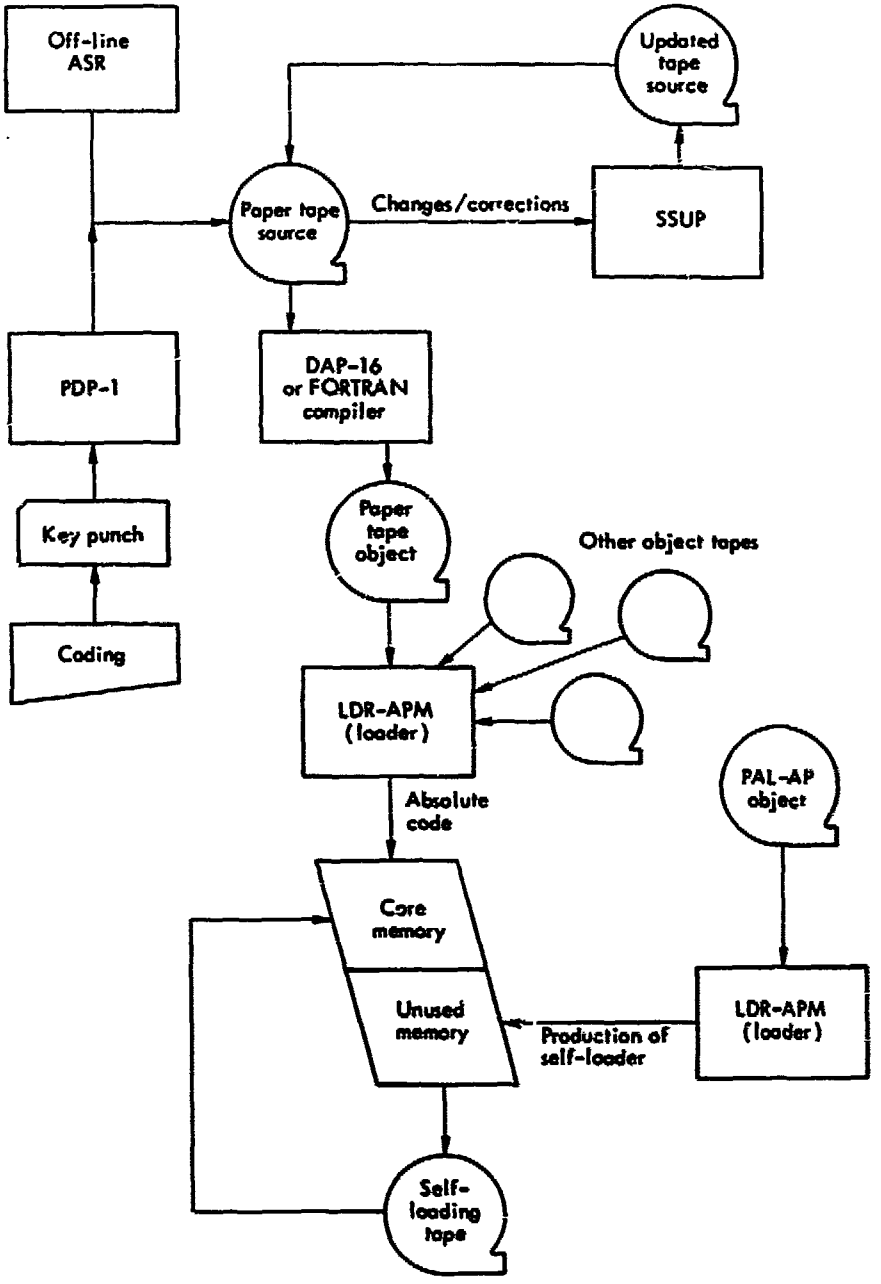

Figure 32. Production of a program for use on the DDP-516.

from the teletype in the form of delete, insert, or copy commands. These two inputs are then used by SSUP to punch out an updated source program appropriate for inputting to the assembler or compiler.

GPL-A Field Test Route

ROUTE LOCATION

Dynamic (motion) testing of the GPL-A requires a field-test route with landmarks whose geographic coordinates (latitude and longitude) are well known. The performance of the GPL system is demonstrated by the accuracy with which it indicates the position of these checkpoints. This section describes the test route and illustrates how the geographical locations of the checkpoints were determined.

The test route consists of segments of the Interstate highway system near Livermore, California. The primary requirements for a test route are convenience, minimum traffic, and well-established landmarks. The 75-mile route goes from the Lawrence Livermore Laboratory to Los Banos, California, and traverses some 60 miles of I-5, a new, lightly traveled, and essentially straight highway. The truck-mounted GPL-A system typically completes a one-way passage of the route in 2 hr. There are enough exits, undercrossings, and overpasses along this route to establish checkpoints at 1- or 2-mile intervals. This path was also selected, in part, because of the availability of roadside rest areas and a suitable place to stop over in Los Banos. The location of the test route in relation to Livermore is shown in Fig. 33.

Once the test route had been selected, the geographic positions of the checkpoints had to be determined. The simplest and most direct source of this data would have been the U.S. Geological Survey (USGS) 7.5- and 15-min* quadrangles. Locations fixed with the aid of these maps are accurate to within about 0.01 arc-min (60 ft). This is more than satisfactory, since GPL-A resolves position to only the nearest 0.1 arc-min. However, since the USGS maps predate the construction of Highway I-5, alternative sources of position data had to be found.

The California Coordinate System used by the California Division of Highways was the prime source of alternative position data. Initially, it appeared that the design plans for I-5 drawn in these coordinates would suffice, but it was found that these plans showed no features that could be located on the older, more complete USGS quadrangles. The only correspondence between the coordinate systems is through the Index Sheets issued by the Division of Highways. These sheets reference the locations of proposed road construction to the location of various features shown on the USGS maps. The location of a highway can then be transferred from one map to another by the suitable adjustment of scales and careful photography.

This approach proved unacceptable for several reasons. First, it offered much

*One minute of latitude is 1 nm, while one minute of longitude at 37° latitude is approximately 0.8 nm.

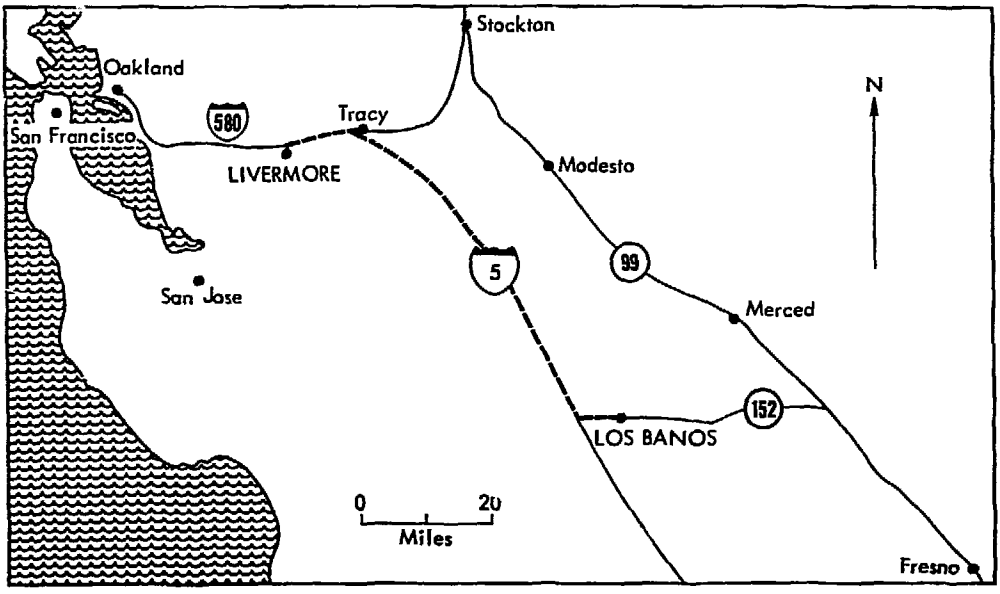

Figure 33. GPL-A field-test route (dashed line) between Livermore and Los Banos, California.

less accuracy than the 0.01 arc-min of the USGS maps. Second, the highways shown on the Index Sheets were just sketched in and did not exactly correspond to the roads as they are actually built. Finally, the scales used on the Index Sheets were unsuitable.

The checkpoints were eventually located through the use of the site-view plans for the highway. These are a set of detailed, large-scale (typically 1 in. = 50 ft) drawings that illustrate the actual location of the road. While the plans are drawn in California Coordinates,* the conversion to latitude and longitude is straightforward. It employs an algorithm[12] that is easily implemented on a computer. The overall accuracy of the checkpoints determined in this fashion is on the order of 0.01 arc-min (60 ft).

GRAVITY ANOMALIES

Gravity anomalies (deflections of the vertical) introduce oscillatory errors into inertial navigation systems. These are similar to accelerometer measurement errors in that they excite Schuler (84-min) oscillations but not 24-hr oscillations. The velocity channel oscillations are integrated with respect to time and incorporated into the system position error. The main effect of gravity anomalies is to cause a small error in the Kalman filter's estimate of system error sources at the first part of the calibration

*The California Coordinates are a Lambert Conformal Grid system used to identify position in rectangular coordinates (X and Y). This is preferred to the more common spherical latitude and longitude system because it avoids tedious problems in spherical trigonometry.

(rest) period. However, this error diminishes rapidly with time in calibration to a relatively insignificant value.

The residual effect of gravity tilt, after the effect on the oscillatory errors has subsided, is a small error in position if the present gravity tilt is changed from that at the beginning of the mission. Compared to long-term mission errors the position errors due to a change in tilt can probably be ignored, since they are normally on the order of a few tenths of an arc-minute.

Off-Line Computer Programs

Two extensive computer programs were written to support the GPL program. One was for data reduction (DINSII) and the other was for simulation studies (WISPIN). Both were written to operate on in-house large scientific computers in higher-level languages to provide versatility in analyzing GPL results. For less-formalized side calculations a Hewlett Packard calculator/plotter and APL time-sharing service[13] were used. WISPIN and DINSII are treated in detail in Volumes 9 and 10 of this report; they are described briefly here, together with a simulation program developed by an outside contractor, The Analytical Sciences Corporation (TASC), for this project.

DATA REDUCTION (DINSII)

DINSII is a computer program written in FORTRAN for use on the LLL CDC 7600 computer. It was developed for the processing and analysis of data generated by the GPL-A system. In one mode (straight data reduction), the program analyzes navigation run data. In the other mode (called postprocessing) it subjects the data to different mission profiles and estimates what would have happened given each profile.

Data Flow Through DINSII

Figure 34 shows how the different parts of the DINSII program interact. Data input to DINSII is recorded on a magnetic tape by the GPL-A system. DINSII reads the data from tape, converts it to CDC 7600 word format, analyzes it by mathematical techniques to determine if data in a record is bad (this is called glitch detection), and places the processed data on the disk. The program then can read data from disk for plotting of useful parameters such as velocity errors, latitude errors, longitude errors, and gyro drifts, as calculated by the GPL-A error reduction technique in use. The DINSII Kalman filter and 3-fix* subroutines will also use this data in their estimation of system errors and gyro drifts. After the data has been saved on disk, it can be put on a master tape for permanent storage and future processing.

DINSII Program Organization

The DINSII code consists of an executive routine, MAIN, and various subroutines that are used to perform the necessary functions. These routines can

*"3-fix" is a technique for determining gyro drifts by using three latitude measurements.

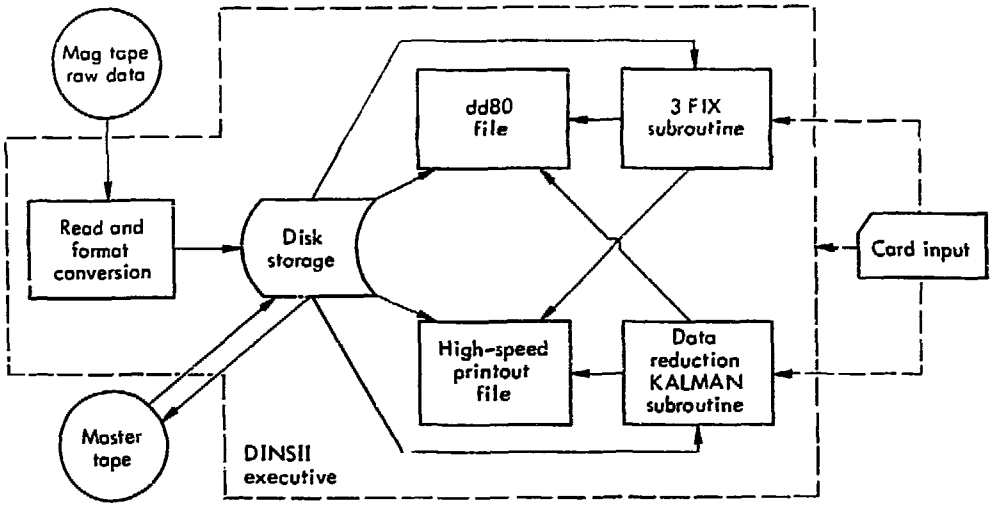

Figure 34. DINSII data system.

store data in three places: large core memory (LCM), small core memory (SCM), and disk files. Most of the large arrays are assigned addresses in LCM. Whenever possible, arrays used most frequently are assigned to SCM because of its faster access time. Data can be placed on the disk in any one of the four files NAV, RESET, STANDBY, and KALMAN. Each piece of data is assigned a channel number and placed on one of these files. Data may be actual data generated, such as system north velocity; data computed from run data, such as north velocity error; or data resulting from analysis computations, such as Kalman filter estimate of east velocity.

DINSII contains two processors: the raw processor for data from a tape generated by GPL-A, and a master tape processor for data from a master tape created by previous raw processing of one or more test runs of GPL-A.

The raw processor will read a record of data from the tape and convert the data in each record (Verify, System Error, Nav, and Reset) from Micro-D and DDP-516 words to CDC 7600 words. Data is then passed through a glitch detector to determine if a record contains bad data. If so, the record is automatically deleted. Converted data (if not deleted) is then used to compute other parameters such as velocity errors, latitude, longitude, radial errors (position errors are computed only at checkpoints where true latitude and longitude are known), and gyro drift estimates. As data is processed, it is placed into a buffer in the assigned channel. When buffers are full, they are emptied onto the proper disk file. Once the data has been processed and placed on the disk, the following options are available:

- Plotting data from NAV and RESET files.

- Computing gyro drifts and other parameter estimates using a Kalman filter and 3-fix analysis.
- Printing motion-sensing data.
- Printing verify list.
- Printing system error list.
- Creating master tapes.
- Postprocessing using the Kalman filter.

The master tape processor will read the processed data for a specified run from the master tape and put the data into the proper disk files. Since the data is already processed, no further treatment of the data is required. Once the data is on disk the various options can be executed as in the raw processor, except for the creation of master tapes.

DINSII also allows running non-real-time error reduction experiments. Data from the GPL-A in the navigate mode can be reprocessed using various assumed rest/motion profiles and filter parameters. The results are essentially identical to a real-time run made under the assumed conditions. The method is called post-processing and is discussed further at the end of this section.

Data Reduction Kalman Filter (KDR)

To properly evaluate the various error reduction techniques, it was necessary to have some measure of gyro behavior that is more accurate than the estimates provided by the GPL-A error reduction filters. It was found desirable to apply another Kalman filter, with augmented measurements, during data reduction to achieve a good measure of the GPL-A filter performance.

The data reduction filter makes use of all the information on system errors recorded during a GPL-A test run. The information available on any one data record varies during the run according to test conditions when the record was made. The possible combinations of measurements are as follows:

- GPL-A moving at checkpoint—velocity and position errors.
- GPL-A stationary at checkpoint—velocity, position, and change azimuth errors.
- GPL-A moving, no checkpoint—velocity errors.
- GPL-A stationary, no checkpoint—velocity and change in azimuth errors.

The 11-state data reduction Kalman filter accommodates these changes by updating the state estimates and covariance matrix with a measurement matrix H and measurement noise covariance matrix R, selected according to the information available in the data record. Because it uses much more data, the Kalman filter has an advantage over the 3-fix data reduction approach, which uses only position and, when stationary, change-in-azimuth errors from data taken at a checkpoint.

The data reduction filter can operate with a variable update interval. This feature allows it to extrapolate the estimates over missing data records and arbitrary measurement times.

The filter has been programmed to accommodate changes in the navigation system's dynamic error equations due to damping and GPL-A velocities. This is accomplished by adding appropriate terms to the differential equations that model system errors. The velocity terms provide more accurate error estimates when

the GPL-A is in motion. The damping terms allow the data reduction filter to process data from periods when the navigator is in the calibrate mode with those error reduction techniques that use damping.

3-Fix Drift Rate Recovery

The ability to recover the gyro drift rate histories was exceedingly valuable in characterizing the base performance of the GPL-A system. Not only did it prove useful in diagnosing abnormal system behavior during motion, but it also permitted a better understanding of the character of the gyro drift rates in the system. The latter is of particular importance in properly defining the statistical models that are used in the Kalman filter error-reduction schemes that have been designed for GPL-A

A 3-fix capability to estimate the gyro drift rates was developed early in the testing program for GPL-A. It makes use of test data to experimentally verify concepts in the GPL program. This data analysis capability is embodied in the A3FIX subroutine that forms a part of the DINSII data reduction code and WISPIN simulation code.

The GPL-A data is processed in the A3FIX subroutine as illustrated in Fig. 35. The position errors are first filtered* to remove the Schuler oscillations. The filtered position data is then used to estimate the effective north and vertical drift rates.

Precision azimuth indicator (PAI) data can be used with the position errors to compute the effective north gyro drift rate and a less noisy estimate of the vertical gyro drift rate. In addition, it permits the effective east gyro drift rate to be estimated to within an unknown constant; i.e., it can resolve slow changes in the east gyro drift rate but not the absolute magnitude.

For diagnostic purposes, it is necessary to relate the gyro drift rate estimates to the instruments on the platform. The estimates are calculated in the north-east-down coordinate system. Therefore, the system is initially aligned along one of the cardinal headings so that the level gyros point along the north-south and east-west axes (this is called a fixed azimuth mechanization). A coordinate transformation can then be calculated to relate the drift rate estimates to the instruments.*

Two alternative methods are used in the A3FIX subroutine to recover the drift rates, depending on the input data that is available. The most accurate and complete estimates are derived if the inertial platform's azimuth error changes and position errors are known. For this method, the system must be stationary.

*Filtering was achieved using a "sliding window" technique which averaged measurements for 84.4-min intervals.

*The wander angle changes during static navigation according to the relation $A = \delta\ell \sin\lambda$. The longitude error that develops will therefore cause the platform heading to change slightly during a test run. For example, typical longitude errors that develop at the end of 60 to 80 hr of navigating are 20 arc-min. At LLL's latitude (37°41'), this typically produces a change in A_α of only 12 arc-min. The coordinate transformation can therefore be calculated with the initial value of A_α, since any subsequent changes are negligible.

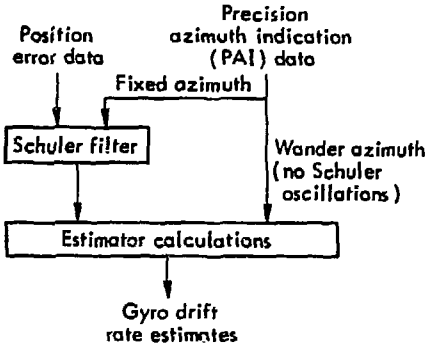

Figure 35. Basic organization of the A3FIX subroutine.

The alternative method is required in the absence of azimuth error change data. In this case, the east gyro drift rate profile cannot be determined, and the quality of the estimate of vertical drift rate degrades. This second method does have the advantage that it can be applied during motion periods, with some small loss of accuracy for a slowly moving vehicle.

Postprocessing

The KALMAN subroutine can also be used to conduct computer experiments with an 11-state Kalman filter. This method, called postprocessing, is used to run additional experiments on data generated by RESET-16 real-time runs, in contrast with the RESET-16 error reduction Kalman filter (ERKF) in the test van (GPL-A system). Since the KDR has access to all measurement data, its assessment of test run performance should be the best possible. If we supply it only with the same data that the ERKF got, we should get exactly the same estimates as the ERKF gave.

However, if we assume that the test data was taken under a motion profile different from that in the actual experiment and process the data with this constraint, we should get precisely the same results that we would have got had we run the original experiment with this assumed motion profile. This means that one set of test data can be used under a variety of assumed motion profiles, which gives us system performance comparisons with and without error reduction under a variety of assumptions.

The postprocessing can be done using two possible combinations of measurements:

- Method 1—velocity.
- Method 2—velocity errors and change in azimuth errors.

A special motion profile is selected by selecting navigate-calibrate sequences and frequency of measurements, the maximum frequency being the frequency of recording.

WISPIN SIMULATION

WISPIN is a computer program written in FORTRAN for use on the CDC 6600 and 7600 computers at LLL. The WISPIN computer program was written to simulate an inertial navigation system for surface navigation. The basic equations and their modifications for platform damping and error reduction are solved to determine system performance and stability. Such system simulations aid greatly in judging the relative merits of different mechanizations for platform control, where the effects of error sources applied singly or in combination may be shown and compared with a perfect system.

WISPIN was found to be especially useful in simulating systems under more

realistic conditions than those assumed by the analytical procedures used to design them. For example, simulation of proposed damping methods helped in the development of the GPL damping schemes. Other work done with WISPIN included preliminary studies of the motion sensor (for automatically switching between calibration and navigation depending on the state of motion), the use of internal signals to determine motion, and the effects of interruption of the partial calibration procedure (causing transient and long-term effects).

WISPIN was also useful in debugging various data analysis methods. The A3FIX and KALMAN data reduction subroutines used in DINSII were added to WISPIN. Since WISPIN has controlled inputs such as gyro drifts, it was very useful in determining the accuracy of the drift estimates of the subroutines.

WISPIN is organized into subroutines that simulate the various parts of the system and subroutines that carry out supporting tasks. The accelerometers are modeled in terms of their physical parameters. The gyros are simulated in terms of their random drift rates—the most important long-term source of system errors. The navigation computer is simulated to continuously solve the navigation equations for values of computed position and torquing commands to keep the platform horizontal and aligned to north. The function of the motion evaluator is simulated in order to detect motionless periods, stop normal position calculations, and initiate the calibration sequence.

The supporting subroutines translate input data for program use, transform vehicle motion into inertial and gravitational forces, compute the state of the

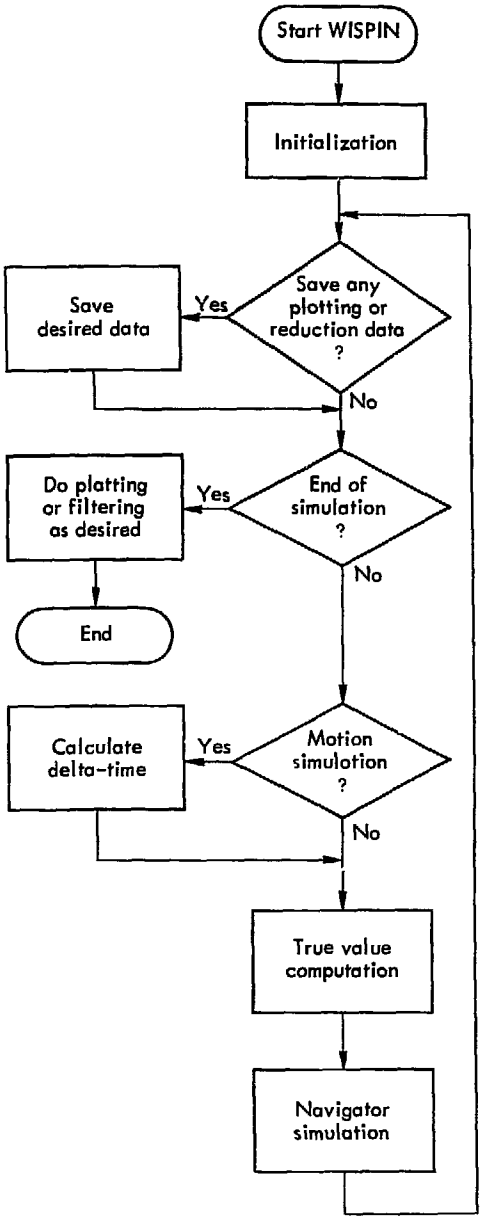

Figure 36. General flow of WISPIN.

error-free system, and make plots of the resultant errors.

WISPIN is divided into five groups of subroutines: 1) initialization, 2) computation of true values, 3) solution of navigation equations, 4) output, and 5) data analysis. The principle functions of each group are described briefly here. A general flow diagram is shown in Fig. 36.

Input

Data cards are read to provide the problem description, system characteristics, initial error values, gyro drift rates, and vehicle motion description. Several sets of data may be stacked behind the program deck. (Refer to the WISPIN Input and Output section of Volume 9 for input variables and their data card formats.) Data quantities are changed by the program from customary units to feet, pounds, seconds of time, and radians for program use. Constants are defined and variable quantities are initialized.

True-Value Computation

The motion description given by the input data is inspected at each computation cycle to determine the true heading and true acceleration. Vehicle velocity, geographic position, and rotation of the gravitational vector are then computed from the heading and acceleration for comparison with results from the solution of the navigation equations. The north and east acceleration components including vibration, plus the apparent force due to the rotation of the earth (Coriolis force), are supplied to the accelerometer simulation subroutine. This acceleration is the only vehicle motion input to the simulation of the navigation system.

Navigator Simulation

Mechanical operations of the accelerometers and gyros of the reference platform are simulated. The velocity change measured by the accelerometers is sent to the "system" computer simulation, where the navigation equations are solved for computed velocity, computed geographic position, and gyro torquing signals. A separate logic section of the system computer sets the values of the coefficients that modify the basic equations for platform damping. After the platform oscillations are damped, new gyro bias torques are computed for partial calibration.

Rotation rates of the platform about the three orthogonal axes are set to the gyro drift rates plus the computed torquing signals. The random gyro drift rates are generated from a mathematical model.

Output

Computed values from solution of the navigation equations are compared with the true values from the input data. Results of the simulation run are the four errors in the north and east components of velocity and geographic position and the three platform error angles about the local vertical and geographic axes. These values are stored during computation and plotted when the simulation run is completed.

Data Reduction

These subroutines consist of the two data reduction codes A3FIX and KALMAN,

which are used in DINSII for state estimates. WISPIN proved useful for debugging these two codes by providing controlled "data" input for the codes to operate on. The results could then be compared with those expected, since the simulated system's performance was totally known.

TASC SIMULATION

Another digital computer simulation developed for the purpose of testing the GPL-A system design was the TASC simulation. The two program vehicles used to generate this simulation are the continuous system modeling program (CSMP)[14] and the linear covariance program.[15] The simulation developed in the early part of the GPL program was simple and was intended for the study of specific problems or system concepts. The simulation was gradually expanded and integrated into a composite simulation of the complete GPL-A system with the suboptimal filter and the velocity averager for error reduction.

CSMP

CSMP is an IBM System/360 program for the simulation of continuous systems that combines a functional block modeling feature with a powerful algebraic and logical modeling capability. The input language enables a user to prepare structure statements describing a physical system, starting from either a block diagram or a differential-equation representation of that system. The program provides a basic set of functional blocks, together with means for the user to define functions specially suited to his particular simulation requirements. Application-oriented input statements are used to describe the connections between these functional blocks. CSMP also accepts FORTRAN statements, thereby allowing the user to readily handle complex nonlinear and time-variant problems. A translator converts these structure statements into a FORTRAN subroutine, which is then compiled and executed alternately with a selected integration routine to accomplish the simulation.

Input and output are simplified by means of a free format for data entry and user-oriented input and output control statements. Data and control statements may be entered in any order and may be intermixed with structure statements. Output options include printing of variables in standard tabular format, print-plotting in graphic form, and preparation of a data set for user-prepared plotting programs.

Two important features of CSMP are statement sequencing and a choice of integration methods. With few exceptions, structure statements may be written in any order and, at the user's option, may be automatically sorted by the system to establish the correct information flow. Centralized integration is used to ensure that all integrator outputs are computed simultaneously at the end of the iteration cycle. A choice may be made between the fifth-order Milne predictor-corrector, fourth-order Runge-Kutta Simpson's, second-order Adam's, trapezoidal, and rectangular integration methods. The first two methods allow the integration interval to be adjusted by the system to meet a specified error criterion.

CSMP provides a complement of 34 functional blocks (also called functions) for modeling a continuous system. These functions include such conventional analog computer components as integrators and relays plus many special-purpose functions such as delay time, zero-order hold, dead space, and limiter functions. This complement is augmented by the FORTRAN library functions such as cosine, tangent, and absolute value. In addition, the user can define functional blocks specially suited to his own application area. This definition can be accomplished either through FORTRAN programming or, more simply, through a macro-capability that permits individual existing functions to be combined into a larger functional block.

TASC GPL-A Simulation Program

The GPL-A simulation includes all of the logic and computations contained in the DDP-516 computer program detailed in the section on GPL-A software except for the azimuth motion sensor. In addition, the simulation includes a nonlinear model of the INS and the required GPL-A system peripheral hardware. The salient features of the INS model are:

- Acceleration reset integrator.
- Velocity quantizers.
- Navigate and damping modes.
- x-gyro, z-gyro, longitude, and latitude reset capability.
- Fine azimuth resolver quantization.
- Accelerometer dead zone.
- Gyro bias and Markov drift models.
- Background acceleration noise.
- Tilt indicator.
- Micro-D mode-control logic.

Since the simulation must include models of the system hardware and is constrained by certain CSMP limitations, slight modifications of the program designed for the DDP-516 are necessary. These modifications include such things as floating-point rather than fixed-point tests and computations. The real-time loop has a cycle time of 0.2 sec rather than 0.05 sec to reduce running time. A modified form of the INS damping scheme was also required to circumvent simulation integration-interval problems.

A general flow diagram of the simulation is given in Fig. 37. The background and real-time loop logic and computations are detailed in the section on the suboptimal filter (p. 26); this section is concentrated on the simulation operation and the hardware models.

Figure 38 is a block diagram of the GPL-A INS model and the tilt-indicator hardware model in FORTRAN and CSMP language notation. The navigation mode is entered by setting K_b K_z and K_{ze} (if used) equal to zero. K_v and NAVSW are set equal to unity. K_{ze} is set equal to K_z if latitude settling is implemented in the navigator during the damping mode. When the damping mode is entered from the navigation mode, the initial damping switch is closed for the first third of the duration of the damping mode. The closing of this switch results in an initial damping interval that is analog in nature. If this scheme is not used, the acceleration reset integrator requires a reset time which is much smaller than normal, resulting in an excessive amount of simulation running time. After the initial large transients have settled, the switch is opened and the remainder of the interval uses the digital damping configuration.

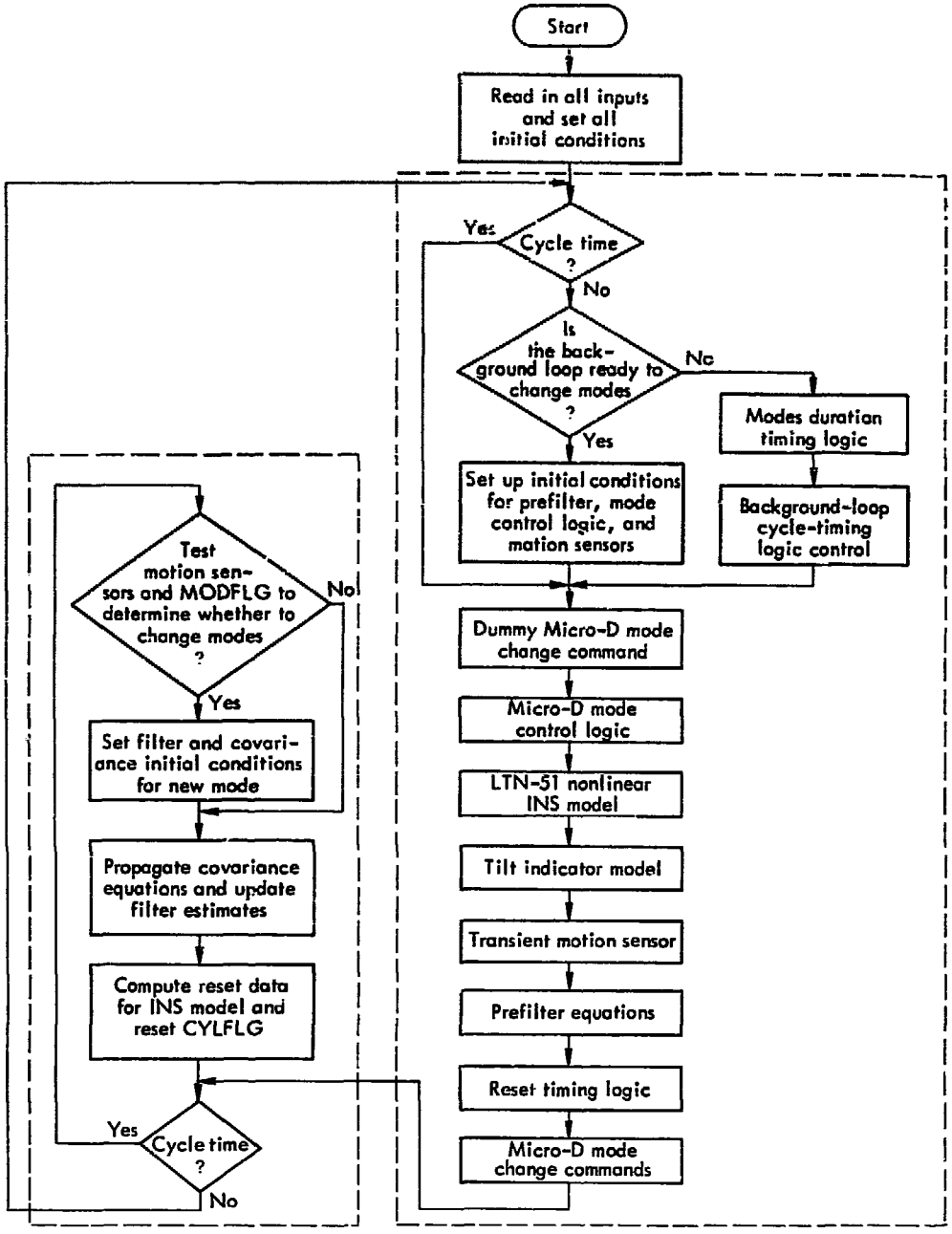

Figure 37. General flow diagram of the GPL-A CSMP computer simulation.

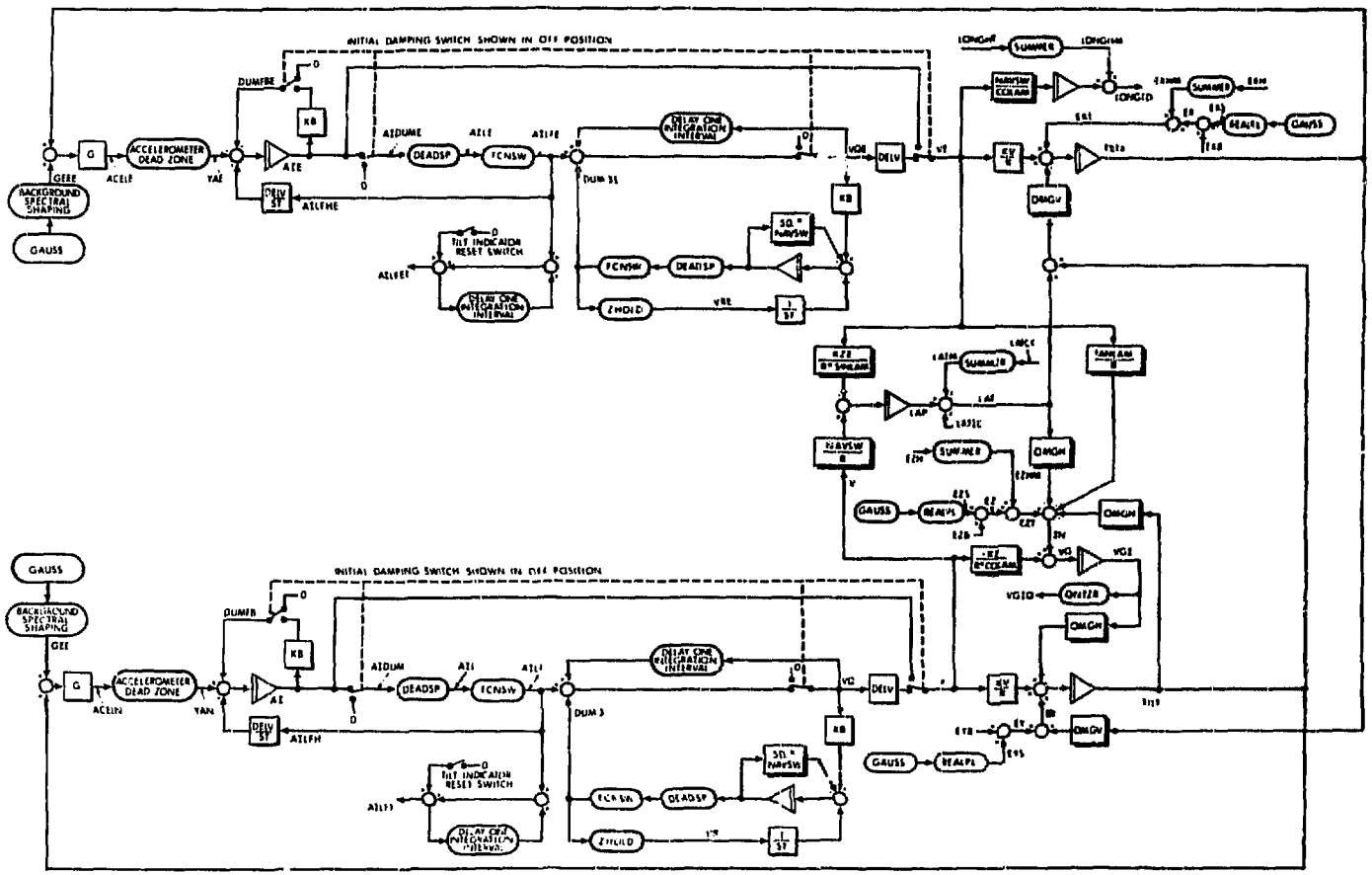

Figure 38. CSMP simulation model for the GPL-A inertial navigation system.

The gyro errors are modeled as a bias plus a Markov drift which is generated by driving a single-pole filter with the output of a Gaussian random-number generator. The incremental gyro resets from the Kalman filter are summed and added to the output of the gyro error model. The latitude and longitude incremental resets are implemented in the same manner.

The prefilter input variables are the outputs of the north and east tilt indicators, AILFT and AILFET, and the quantized azimuth angle, VGIQ. Each time the tilt indicators are sampled, they are reset to zero as illustrated by the tilt indicator reset switch.

The background noise in each channel is generated by passing the independent output sequence from a Gaussian random-number generator through the required spectral shaping filter. For white noise, the shaping filter is not used. It should be noted that all of the random-number generators are independent of each other.

References

1. Technical Description, Litton LTN-51 Inertial Navigation System for Commercial Aviation, Litton Industries Commercial Avionics Guidance and Control Systems Division, Woodland Hills, Calif., Publication No. 6363 Revision B (December 1967).
2. Litton LTN-51 Inertial Navigation System for Commercial Aviation, Theory of Operation, Litton Industries Commercial Avionics Guidance and Control Systems Division, Woodland Hills, Calif., Publication No. TT-300-2, Vol. II (November 1968).
3. Micro-D Computer Instruction Manual, American Bosch Arma Corporation, Arma Division, Document MD-1476 (March 1969).
4. Ref. 1, Section 4.
5. Ref. 1, p. 4-9.
6. Ref. 1, p. 4-4.
7. GPL-A Computer Program Description, Tape 10XX, Charles Stark Draper Laboratory, Massachusetts Institute of Technology, Cambridge, Mass.
8. GPL-A Computer Program Description, Tape 20XX, Charles Stark Draper Laboratory, Massachusetts Institute of Technology, Cambridge, Mass.
9. W. F. O'Halloran, R. S. Warren, and D. B. McLaughlin, Studies on Error Reduction and Motion Sensing for the Geographical Position Locator, The Analytic Sciences Corporation, Reading, Mass., Rept. TR-199-1 (UCRL-13516-1) (May 1971).
10. Ibid., p. G-3.
11. H316 and DDP-516 Programmers Reference Manual and 316/516 Users Guide, Honeywell, Inc., Computer Control Division, Document Nos. 70130072156C and 130071627.
12. Plane Coordinate Projection Tables (California Lambert), U.S. Government Printing Office Special Publication No. 253.
13. A. D. Falkoff and K. E. Iverson, APL/360 User's Manual, IBM Corporation (August 1968).
14. System/360 Continuous System Modeling Program (360A-CX-16X) User's Manual, IBM Corporation Publication H20-0367-2.
15. W. O'Halloran, Error Reduction for the Geographical Position Locator, The Analytical Sciences Corporation, Reading, Mass., (LLL document No. UCRL-13450)(January 1970).

NOTICE

"This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately-owned rights."

Printed in USA. Available from the National Technical Information Center, National Bureau of Standards, U. S. Department of Commerce, Springfield, Virginia 22151
Price: Printed Copy $3.00; Microfiche $0.65.