PROCEEDINGS

OF THE

FOURTH INTERNATIONAL WORKSHOP

ON

LASER RANGING INSTRUMENTATION

Software Sessions
1981 October 13-15

Edited

by

Peter J. Shelus
Astronomy Department and McDonald Observatory
University of Texas at Austin
Austin, Texas 78712 (USA)

1982 August

Sponsored by:

University of Texas at Austin
IAG Special Study Group 2.33

Dedicated to

Peggyann
Peter
and
Jonathan

PART I

TLRS: A Data General NOVA-Based Ranging System

by

Ronald W. Heald
Randall L. Ricklefs
Department of Astronomy
University of Texas
Austin, Tx 78712

ABSTRACT

The Transportable Laser Ranging System is a software-intensive system centered around a Data General NOVA III mini-computer operating under RDOS. Programming is done in NOVA assembly language where required for speed, and in FORTRAN otherwise. The operating system is partitioned into foreground and background. The monitor is constantly running in foreground and supplies updates and displays for time, clock differences, weather parameters, and telescope position. The other programs, which run in background, assist in acquiring data for mount modeling and solving for mount parameters, integrating satellite state vectors for pass-by-pass ephemeris generation, and acquiring, displaying, and copying satellite ranging data. Significant use is made of RDOS' multi-tasking, overlaying, and inter-ground communications capabilities. Minor changes to RDOS were made to implement a CRT monitor and CAMAC as system devices.

1 Introduction

The Transportable Laser Ranging System (TLRS) built by the University of Texas at Austin uses a Data General NOVA III computer to control its operation. The computer uses the Real-time Disk Operating System (RDOS) also supplied by Data General. The following briefly describes how the computer hardware and software are able to collect ranging data at a rate of 10 Hz while controlling beam director (telescope) movement and maintaining other station functions. For more information on RDOS and the pertinent languages, see the manuals listed below.

2 Hardware

The computer has a semi-conductor memory size of 48K 16 bit words. The central processing unit (CPU) has optional hardware to increase the speed of floating point number operations, both normal and double precision. The system also has 5 million 16 bit words of rotating disk storage, half of that being on a removable pack. A Tektronix 4006 storage-tube graphics terminal provides the operator console. Other available peripherals are a Texas Instruments Silent 700 thermal printer terminal with cassettes for off-site communications, and a Digi-Data one-half inch magnetic tape drive used for disk backup and data transfer.

To enable the computer to communicate with the timing and beam director electronics, a CAMAC crate was chosen. CAMAC is European and IEEE standard with manufacturers world-wide supplying modules of various functions. The TLRS CAMAC crate contains a blend of purchased and custom-built modules.

3 RDOS Multi-Tasking and Dual-Programming

To increase system resource utilization RDOS permits multiple tasks to execute with apparent simultaneity within a program. Each task is a logically complete, asynchronous execution path. Tasks can execute separate paths, the same reentrant path, or any combination of these.

The RDOS task scheduler is responsible for deciding which task has CPU control at any given moment. To enable the task scheduler to function each task is assigned a priority. The task scheduler always gives CPU control to the highest priority task that is ready. If more than one task is assigned the same priority these tasks take turns receiving CPU control.

A task can exist in any of four states: it may be in control of the CPU and executing its assigned path; it may be ready and awaiting its turn to gain CPU control; it may be suspended and unable to compete for control until it again becomes ready; or it may be dormant since it has not yet been initiated into one of the other four states.

An executing task becomes suspended whenever it makes a call to the operating system. An example would be a task desiring to read a block from the

Table of Contents

PREFACE

The Fourth International Workshop on Laser Ranging Instrumentation was held at the University of Texas in Austin, Texas (USA) during the week 12-16 October 1981. In addition to the usual hardware-oriented sessions, the organizers of the Workshop planned a concurrent "Software Workshop" whereby, in a "Poster Session" type of an environment, programmers and analysts from the various laser ranging stations around the world could meet to discuss the manner in which various tasks were currently being handled by software developed at those stations. The idea behind such a session was simply that, in the main, all stations had similar tasks to be performed and similar problems to be solved. Therefore, this kind of a session could serve as a clearing house for ideas and their implementation in software.

The ground rules for presentations were simple. Presenters would be asked to describe the ways in which they had succeeded in handling some task or problem, with software. The routines presented would have been totally operational and would include, so far as possible, adequate documentation so that source code could be transferred to other stations with minimal effort. This documentation was also to provide sample data together with sample results so that, upon implementation of a particular package at another site, the test case could be run and results compared to the standard to verify correct implementation.

Further, to provide for and to encourage the widest dissemination of these operational procedures, proceedings of this "Poster Session" would be produced and copies would be provided to all attendees. Wherever possible, it was planned that papers together with source code, test data, and test results would be provided by the author to the editor in machine readable form. In this way all editing could be performed using standard word processors and text formatters and a suitable document could be created completely "on-line" with a minimum of expense and delay.

The presentations have been arbitrarily divided into two very broad categories. In Section 1 are presented those descriptions of complete operating systems and the philosophies behind them. In large part, because the papers of this first section describe entire operating systems, code, together with test data and test results, could not be provided in a reasonable amount of space. In these instances the interested individual reader should direct his questions and inquiries directly to the authors of those papers. In Section 2 are presented various useful utilities needed to perform specific tasks at a station. In these cases it is usual that actual source code with test data and results are provided so that other systems can indeed implement such procedures at their own facilities. To this end, if requested, machine-readable copies of such source code could be provided at cost by the editor. Of course, this does not preclude contact with the authors themselves to assure the latest version of each utility. Within each section the papers are presented in alphabetical order with respect to the name of the first author.

It is felt by the editor that the above goals and aspirations have in fact been accomplished in spite of the fact that nearly a full year has gone by before copies of the Proceedings could be disseminated. The delay was not the result of problems with editing or reproduction. Rather it was the result of not having found the right "tool" to accomplish the task. It was originally envisioned that either the DEC-10 or DEC-20 systems of the University of Texas would be able to provide sufficient facilities to accomplish the job. This proved to be not the case because of inflexibility of the system rather than the absence of suitable text editing and formatting procedures.

The breakthrough came when, under the suggestion of Dr. G. F. Benedict, the editor transferred all activity to the University of Texas Astronomy Department's Digital Equipment Company VAX 11/780. The ease of file interaction and manipulation using the UNIX (Berkeley Version 4.1) operating system and standard utilities was all that was required. The correct tool had been found and the task could be pursued with all due haste. All but only a small amount of the textual material was entered directly into the VAX from the media provided by the authors. The document produced was totally edited using the 'vi' text editor and text formatting was performed using SCRIBE (Unilogic, Ltd.). Output of material was provided for by a DIABLO (Xerox Corporation) Series 1300 HYTYPE Printer".

The learning curve was steep but once the skills were mastered the work proceeded quickly and smoothly. I believe that when a second attempt is made to produce meeting proceedings using this method, the time taken to provide distributable material will be quite a bit shorter than it has been this first time around. The editor wishes to thank the readers for having waited so long for the appearance of these proceedings without complaint. My thanks also go to Dr. G. F. Benedict for his original suggestion for VAX usage and for the ensuing help along the learning curve by him and his most competent staff. Also to be acknowledged are Randall Ricklefs, Nelson Zarate, and Arline Tompkins for their time and effort contributed to the completion of this task. Of course, the editor would be remiss not to have acknowledged Eric C. Silverberg and IAG Special Study Group 2.33 Chairman Peter Wilson, the driving forces behind the 4th Laser Ranging Workshop.

disk. The task remains suspended until the call is complete. In the example the task would not be readied until the disk block contents are transferred to memory. During the period that the task is suspended the task scheduler will assign CPU control to the next highest priority task which is ready.

To further increase system utilization RDOS permits two programs, foreground and background, to be simultaneously memory-resident and execute concurrently. The operating system switches control between the two programs according to a predetermined priority. Dual programming allows two unrelated collections of tasks to be executed, sharing all system resources. While this scheme is not as general as a multi-programming system it adequately meets the needs of the TLRS as will be described. Each program is truly independent, but can establish a communication area whose contents are accessible to the other program.

4 TLRS Software concepts

TLRS software uses the dual-programming facility to perform two types of functions. Those functions which are executed continuously are the first type and are performed by the foreground program. They consist of displaying and placing in a communication area information from the clocks, beam director position encoders, station level transducers, and weather transducers. The hand-paddle must also be constantly monitored to take appropriate action when the buttons are activated.

Functions which are performed sequentially form the second function type. Examples are as complex as determining beam director orientation parameters (program ORIENT), integrating a satellite orbit for range and point-angle predictions (INITSAT), and acquiring ranging data (RANGE and HORIZON), or as simple as stowing the beam director (STOW) and setting the system clock (SETCLK). The console operator decides when to perform these functions and is usually required to interact with the program to set parameters. These functions are performed by programs in the background partition.

Operator interaction receives special attention from TLRS software since operators are assumed to have little or no computer background. Programs query the operator for each needed parameter. Responses are checked for correctness, and the operator is informed if a problem is found. Only appropriate operator inputs are accepted so format or value errors cannot cause the program to fail.

Tcomputer hardware of TLRS was designed to provide an event-driven system. No polling loops are used by the software as it relies completely on interrupts to tell it of outside events. This greatly increases system efficiency.

All coding for TLRS was done in FORTRAN except where assembly language was required to minimize memory usage and execution time. During TLRS software development constant improvements and changing function definitions proved many times the value of using a high-level language. If the poor efficiency of

compiled code can be tolerated, the benefits in ease of coding and making changes are significant.

Figure 1 is an overview of the TLRS software system.

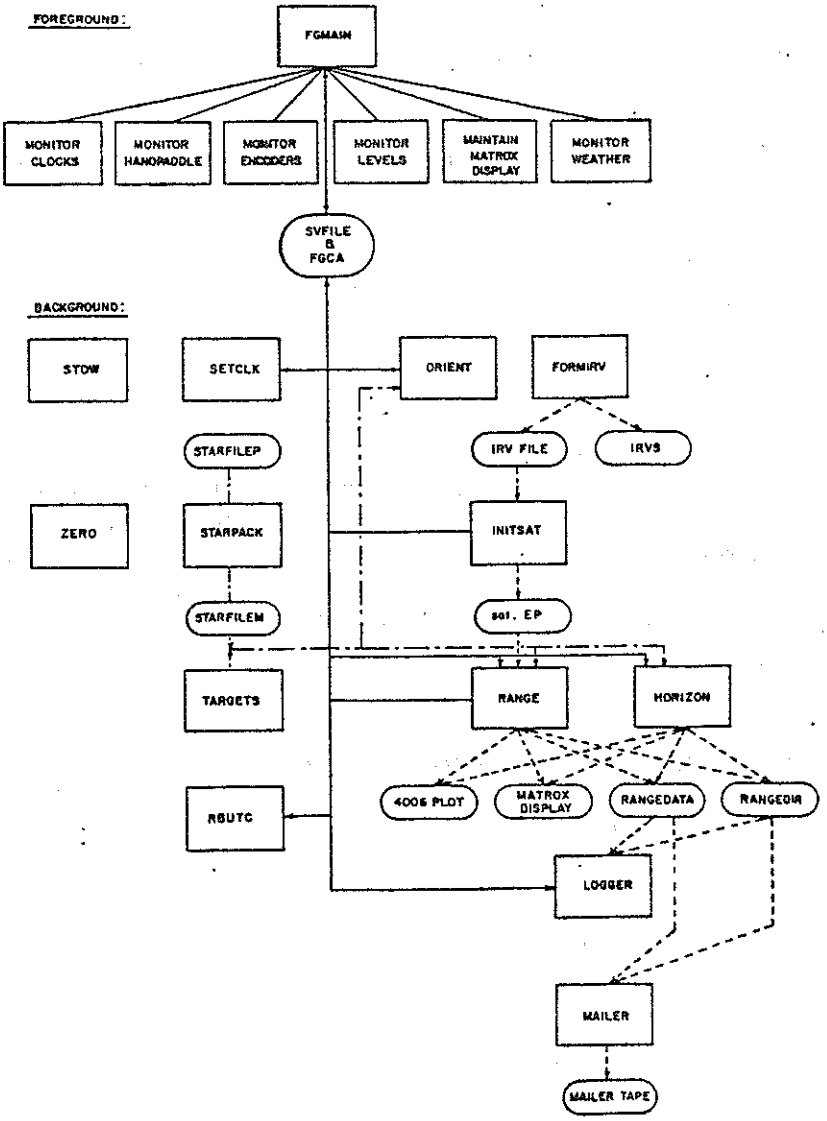


Figure 1: TLRS Software System Overview

The foreground/background communications through the foreground communication area (FGCA) and the station variables file (SVFILE) are shown by solid arrows. Broken arrows and elliptical boxes indicate files passed between programs or to peripherals. Rectangles in foreground indicate functions within the monitor program, while in the background rectangles indicate functions implemented in separate programs.

5 Foreground Program

Both foreground and background programs begin continuous execution when the TLRS computer system is initiated. The program priorities are set so that if either program attempts to dominate CPU control equal time will be given to the other program. The foreground program consists of four concurrently executing tasks and interrupt service routines (ISR) for the system 1 Hz tick and hand-paddle buttons.

On each 1 Hz pulse from the system rubidium clock the appropriate ISR reads and places in the foreground communication area (FGCA) the current beam director position. This records positions at precise intervals necessary for proper beam director guiding. The ISR also increments the FGCA time and readies the clock task. When the clock task gains CPU control it updates the time and beam director position on the status display.

The hand-paddle ISR executes whenever any hand-paddle button is pressed. It readies the hand-paddle task. When the hand-paddle task gets control it performs the function(s) indicated by the buttons and suspends itself when they are completed.

Another foreground program task takes readings from the station levels and weather transducers. A final task determines differences between the system clock and the two other 1 Hz sources. Both these tasks place their results in the FGCA and on the display.

The foreground program was written in assembly language, and makes use of memory overlays and reentrant paths. This was done to minimize memory usage and execution time leaving as much of these resources as possible for the background program.

6 Background Program

The background program begins by executing the Data General supplied command line interpreter (CLI). The CLI will perform a variety of functions for the console operator, but its main function for TLRS operation is to bring in programs which perform TLRS-unique functions. When these functions are completed the CLI is restored. Examples of other CLI functions available to the operator are disk directory and file maintenance, executing utilities used for software development, and controlling output device spooling.

The TLRS programs used to determine the beam director orientation parameters and perform ranging use the RDOS multi-tasking capabilities. Only the ranging program task functions will be described as they are typical of both programs.

The range data acquisition program has as many as four tasks running at once They are described in order of ascending priority. The starting or main task is always running. It begins by asking the operator for the required parameters, and then starts other tasks to track the target, perform ranging,

and transfer the data to disk. While ranging is occurring the main task displays the data on the operator console. When the ranging burst is complete the task loops to begin again.

The task to transfer the range data to disk is readied just after the ranging burst begins. It continues transferring data during ranging until it detects the end-of-burst mark in the data at which point it suspends itself.

The target tracking task begins execution when the operator has selected a target. It is readied on each 1 Hz tick of the system clock. Its function is to compare the actual and desired beam director positions and send the hardware the needed corrections.

The task which controls the timing hardware during ranging runs at the highest priority. It repetitively performs the following: first the hardware is instructed to fire the laser and the actual fire time is recorded. The task then instructs the hardware to open the return window and records the return time if a return occurs. This task is interrupt driven and written in assembly language to keep its execution time small.

7 RDOS Modifications

Several minor changes and additions were made to the operating system, to increase its usefulness to TLRS. The most extensive of these was to handle interrupts from the CAMAC crate and add a device for the status display. RDOS source was purchased to allow full access to the operating system for such changes.

The CAMAC crate controller provides an interface between the NOVA and CAMAC busses. When the NOVA receives an interrupt from the crate, it must poll the controller to determine from which station the interrupt originated. This code must reside in the operating system, since the station must be known to pass control to the correct ISR in the foreground or background program.

The status display is driven from a custom-built module in the CAMAC crate. Since this module is similar to Data General output devices, a device driver was written for the display and placed in the operating system. This allows normal system calls and FORTRAN write statements to be used when updating the display.

Another change which will be implemented in the operating system will allow more satisfactory handling of short ground target ranges. RDOS priority interrupt handling does not pass control from one ISR to a higher priority ISR until the first ISR has released it. This causes some slow interrupt response times in a multiple interrupt system. Also, for RDOS to give a foreground program's ISR control over a timing interrupt requires a memory remap, which also takes considerable time. The response time is currently too slow to permit use of the same timing hardware for obtaining fire and return times when ranging ground targets. Hence some of the hardware was duplicated to obtain these ranges. Only one of these problems can be alleviated in

so^tware: the remap problem can and will be handled by moving the timer ISR's into the operating system.

8 Conclusions

In retrospect it is very difficult to judge needed computer memory size, instruction set power, execution speed, and data path width given only the system definition. The main reason for choosing Data General equipment for TLRS was historical: McDonald Observatory had always used Data General computers and personnel were familiar with its programming and maintenance. However it has proven to be an adequate choice to make TLRS a workable system.

9 References

Real Time Disk Operating System (RDOS) Reference Manual, 093-000075-08, Southboro, Data General Corporation, 1979.

NOVA-LINE FORTRAN IV User's Manual, 093-000053-09, Southboro, Data General Corporation, 1978.

Macro Assembler User's Manual, 093-000081-04, Southboro, Data General Corporation, 1975.

Telescope Control and Data Handling at Dodaira Station

by

Tomohiro Hirayama and Tai Kanda
Tokyo Astronomical Observatory
Mitaka, Tokyo, 181 Japan

1 COMPUTER AND PERIPHERALS

HEWLETT-PACKARD 2100S
PURCHASED IN 1974
MEMORY 32KW (1K=1024, 1 W = 16 BITS (+PARITY BIT))
16K OF CORE + 16K OF IC MEMORY (W/BATTERY BACK-UP)
NO MEMORY MAPPING, ADDRESS 15 BITS, SO THIS IS MAX

(OCTAL)	(DECIMAL) WORDS	
0- 1	2	A & B REGISTERS
2- 1077	574	INTERRUPT CELLS, LINKAGE (OS)
1100- 1777	448	0-PAGE LINKAGE FOR USER
2000-15777	6144	OS
16000-77633	25500	COMMON (OPTIONAL) USER PROGRAM & LIBRARIES OVERLAID SEGMENTS AREA
77634-77677	36	CONSOLE INPUT BUFFER
77700-77777	64	INITIAL LOADER (WRITE PROTECT)

TIMING
0.98 US MEMORY CYCLE TIME
1.96 US ADD (16 BIT INTEGER)
51.94-55.86 US FLOATING DIVIDE
DATA FORMAT
INTEGER: 2'S COMPLEMENT -32768 TO 32767
ASCII: FIRST CHARACTER MSB, SECOND LSB.
FLOATING: 1.5E-39 TO 1.7E+38 BINARY NORMALIZATION.
SINGLE: 23 BIT FRACTION (7 DECIMAL DIGITS)
DOUBLE: 39 BIT FRACTION (11-12 DEC DIGITS)
(DOUBLE IS SUPPORTED BY SOFTWARE)
DISCS 5 MB
1 TRACK = 48 SECTORS (HARDWARE 2 TRACKS EACH SIDE)
1 SECTOR = 128 WORDS = 256 BYTES
REMOVABLE DISC
TRACK
0 DIRECTORY

FIXED DISC

0->20	SYSTEM
21	USER DIRECTORY
22->198	USER FILES (*)
199->22	JOB BINARY AREA FOR JUST COMPILED CODE
200->202	SPARE

FILE ASSIGNMENT BY SECTOR.
DISCS ARE ALWAYS 'PACKED.'
SOURCE FILES ARE OF VARIABLE LENGTH RECORD FORMAT.
UNUSED TRACKS ARE USED AS WORK AREA.
(*)NOT USED FOR DURABLE FILES. THIS AREA USED FOR FILE BACK-UP (REM->FIX->ANOTHER REM) ABOUT ONCE A MONTH.

HP DOT BI-DIRECTIONAL PRINTER
HP CRT TERMINAL
HP PAPER TAPE READER
TELETYPE TTY (ORIGINALLY CONSOLE, NOW ONLY FOR PUNCH)
DRUM XY PLOTTER 25CM X 30M (SECOND-HAND)
HIGH SPEED PAPER TAPE PUNCH 6/8 HOLE (SECOND-HAND)
PAPER TAPE READER FOR 6 HOLE JAPANESE TELEX (SECOND-H)
PROM WRITER ASSEMBLED AT DODAIRA
INTERFACES FOR LASER OBSERVATION (INCLUDING HP-IB I/F)

2 OPERATING SYSTEM

HP DOS-IIIA
SINGLE USER, SINGLE PROGRAM.
NOW NOT SUPPORTED BY HP. PERIPHERAL DRIVERS (EXCEPT FOR DISC AND TAPE READER) ARE WRITTEN BY US. SO SOME NON-STANDARD BUT USEFUL FEATURES COULD BE INCORPORATED. FOR INSTANCE, MONITORING CONSOLE (CRT TERMINAL) I/O BY PRINTER, WHILE THE LATTER IS IN USE AS SYSTEM OUTPUT DEVICE.
I/O ARE IN PRINCIPLE UNDER CONTROL OF OS, BUT IT CAN BE OVERRIDDEN. TELESCOPE CONTROL ETC ARE OPERATED BY THIS ARTIFICE.
IN FUTURE, AFTER REPLACEMENT OF THE COMPUTER, RTE-IV WILL BE USED AS OS. BEING MULTI-PROGRAM SYSTEM, IT DOES NOT ALLOW USE OF I/O INSTRUCTIONS OUTSIDE OS. SUITABLE DRIVERS SHOULD BE INCLUDED IN SYSTEM GENERATION.

3 LANGUAGES

ASSEMBLER: ONLY FOR I/O AND A FEW SUBPROGRAMS.
MASM (META-ASSEMBLER) OF UNIVAC 1100/80B AT TAO CAN CROSS-ASSEMBLE HP'S SOURCE (SOME LIMITATIONS).
FORTRAN IV: CHIEFLY USED.
OCTAL CONSTANTS (E.G. 123456B) USEFUL FOR US.
ALGOL: HAD BEEN USED.
NICE COMPILER. NO BUGS.
WHILE ... DO
DO ... UNTIL
CASE
STATEMENTS ENABLE GOTO-LESS PROGRAM.
BUT NOT UPDATED FOR NEWER HP COMPUTER FEATURES, SUCH AS >64K PROGRAM AREA.

4 OUTLINE OF TELESCOPES

SATELLITES' TRANSMITTER & RECEIVER, MOON'S TRANSMITTER:
XY-MOUNT, X-AXIS POINTS TO THE NORTH POINT.
AZ,EL OF X-AXIS (IDEALLY 0,0) & ERROR FROM PERPENDICULARITY BETWEEN X,Y-AXES AND BETWEEN Y-AXIS & TELESCOPE ARE CONSIDERED IN CALCULATION OF PREDICTION.
THESE ERRORS ARE DETERMINED BY STAR OBSERVATIONS.
SMALL CATALOG OF SOME 500 BRIGHT STARS IS IN DISC.
MORE COMPLICATED ERRORS WHICH PROBABLY ARISE FROM FLEXURE SEEM TO EXIST. ANALYSIS OF THIS EFFECT HAS NOT BEEN COMPLETED.
MOON'S RECEIVER: ALTAZIMUTH MOUNT.

5 CONTROL TIMING

THE CONTROL SYSTEM (NOT THE COMPUTER) HAS A CLOCK AND AT EVERY 1/10 SECOND READS TELESCOPE ANGLES & OUTPUTS VOLTAGE (SET BY THE COMPUTER) TO THE TORQUE MOTORS, WHICH ARE ON THE TELESCOPE AXES.

6 ELEMENTS ARE TELEXED FROM SAO EVERY WEEK

TAPE FROM TELEX IS READ BY 6 HOLE TAPE READER.
TRANSMISSION ERRORS ARE RARE. MOST ERRORS CAN BE CORRECTED, BUT SOMETIMES CORRECTION IS DIFFICULT (USUALLY IN CASE OF TRIVIAL ERROR, E.G. AT LEAST SIGNIFICANT DIGITS). WE HAVE A PROGRAM TO FIX UP TO 3 SUCH ERRORS IN A LINE.
IT WOULD BE HELPFUL IF CHECK LETTERS OR DIGITS COULD SHOW MORE READILY WHERE THE ERROR IS (AS IN ASTROGRAMS).

7 OBSERVATION PLANNING

SATELLITE PATHS ARE PLOTTED ON XY-PLOTTER WITH 'OBSERVABLE' WINDOWS.

8 PREDICTION CALCULATION

COEFFICIENTS OF CHEBYSHEV POLYNOMIALS WHICH
REPRESENT X,Y, AND RANGE, ARE COMPUTED AND
STORED IN ONE OF THE 10 PREDICTION FILES.

9 TEST OBSERVATION

TEST OBSERVATION CAN BE MADE, IF OBSERVER SPECIFIES
A START TIME.

10 TELESCOPE CONTROL

TELESCOPE CONTROL IS RATHER PRIMITIVE. AT EVERY
1/10 SECOND AXES POSITIONS ARE READ (TO 0.0005 DEG)
AND AT THE NEXT 1/10 SECOND INTERRUPT VOLTAGES
PROPORTIONAL TO THE O-C ARE OUTPUT.
SOMETIMES COUNTER GIVES 123.0000 INSTEAD OF
124.0000 (CARRY WAS SLOW), SOFTWARE AMENDS
THIS SITUATION. D/A CONVERTER ERRORS ARE ALSO
CORRECTED.

11 OBSERVATION

TELESCOPE DIRECTION CAN BE ADJUSTED BY A HANDSET.
FOUR BUTTONS AND A SWITCH (SLOW/QUICK)
ARE SCANNED BY THE PROGRAM. THE SHIFTS ARE
ALONG OR CROSS THE PATH.
MEASUREMENTS (FLIGHT TIME & LASER SHOT DELAY)
ARE STORED INTO DISC WORK AREA (BY 'NO WAIT' I/O),
AND TRANSFERRED INTO ONE OF 10 RESULT FILES
AFTER THE PASS.
TELESCOPE FIELD IS MONITORED BY TV.

12 RESULT SCREENING

THE FLIGHT TIME ARE PLOTTED ON PRINTER.
O-C AND CORRESPONDING EARLY/LATE VALUES ARE SHOWN.

13 SATELLITE TRACKING

```
FTN4
      PROGRAM SATR
C      SATELLITE TRACKING 76-02-23(MON)
C       EXTERNAL  RSB20(SUB20,SUB21,SUB22)
C       EXTERNAL  DISP(DISP)
C       EXTERNAL  .IN17(IN17)
C       EXTERNAL  ROT17(OUT17)
C       EXTERNAL  BITOP(IRIGH,LEFT)
C       EXTERNAL  DATE,DINT(DATE)
C      (EXTERNAL) ECHEB(ECHE.(ASMB) O TUZUKETE ASMB)
C      REV 76-03-23(TUE)
C      FILE "OBS" (:ST,B,OBS,48) NI KANSOKU O KAKIKOMU.
C      KANSOKU-TYUU WA WORK AREA NI KAKIKOMU.
C      1. ZIKOKU (5 SEC TAN'I)
C      2. COUNTER 16 KETA (BCD)
C      3. COUNTER  8 KETA (BCD)
C      REV 76-05-03(MON)
C      HANDSET KARA NO SINGOO O IRERU.
C      1. X... SINKOO-HOOKOO NI OFFSET (CW: SUSUMU; CCW: OKURERU)
C      2. Y... SORE TO SUITYOKU NI OFFSET
C      3. MED... X: 1 STEP (0.1 SEC NO UGOKI BUN)
C      Y: 1/4 STEP
C      4. LOW... X: 1/4 STEP
C      Y: 1/16 STEP
C      PARAMETER(0--9) DE CHEBF & OBS FILE O
C      "CHEBF","CHE01",...,"CHE09",
C      "OBS  ","OBS01",...,"OBS09" TO ERABU.
C      REV 76-05-05(WED)
C      RANGE GATE ZIDOO-SYUTURYOKU (ATT WA 0).
C      REV 76-08-14(SAT)
C      RENSYUU NO TOKI BETU NO ZIKOKU NI YARERU
C      REV 76-10-21(THU)
C      RANGE GATE MARGIN O INPUT
C      REV 76-10-24(SUN)
C      DATE TIME O FILE KARA YOMU.
C      REV 81-03-27(FRI)
C      S-REG BIT 15 ON -> XOFF,YOFF=0 (LABELS 7,77)
      DIMENSION COEFX(19),COEFY(19),COEFD(19),BUF(64)
      REAL MJD
      INTEGER DASC(5),DASC1,DASC2,DASC3,DASC4,YEAR,NYOOBI(2,7)
      DIMENSION IPARAM(5)
      INTEGER LASRIO(3),CHEBF(3),OBS(3),SECTOR(256)
      INTEGER H(4),IBUF64(2)
      LOGICAL HOLD,SAVED
      DIMENSION S(4),A(7)
      EQUIVALENCE (COEFX,BUF(2)),(COEFY,BUF(21)),(COEFD,BUF(40))
      EQUIVALENCE (IBUF64,BUF(64))
      EQUIVALENCE (DASC1,DASC(1)),(DASC2,DASC(2)),
     1  (DASC3,DASC(3)),(DASC4,DASC(4))
      DATA HOLD,SAVED/2*.FALSE./
      DATA LASRIO,CHEBF/2HLA,2HSR,2HIO,2HCH,2HEB,2HF /
```

```
      DATA OBS/2HOB,2HS ,2H  /,SECTOR/256*-1/,IS2/0/
      DATA S/4*0.0/
      DATA A30,A32,A40,A42,A31,A41/4*0.0,2*1.0/
      DATA PI,Y0/3.141593,600.0/
      DATA XOLD,XOFF,YOFF/3*0.0/
      DATA K/20000B/
      DATA NYOOBI
     1 /2HWE,1HD,2HTH,1HU,2HFR,1HI,2HSA,1HT,2HSU,1HN,2HMO,1HN,2HTU,1HE/
      CALL RMPAR(IPARAM)
      IF(IPARAM(1).GT.9)GO TO 3500
      IF(IPARAM(1).EQ.0)GO TO 3510
      CHEBF(2)=2HE0
      CHEBF(3)=2H0 +IPARAM(1)*400B
      OBS(2)=2HS0
      OBS(3)=2H0 +IPARAM(1)*400B
      GO TO 3510
 3500 WRITE(1,3501)
 3501 FORMAT("FILE # TOO LARGE")
      STOP 7777
 3510 CONTINUE
      CALL DISP(0)
      CALL EXEC(23,LASRIO)
      CALL EXEC(14,102B,BUF,128,CHEBF,0)
      WRITE(1,6)IBUF64(2)
    6 FORMAT(10H SATELLITE,I6)
      MJD=BUF(59)
      CALL DATE(MJD,YEAR,MONTH,DAY)
      IDAY=DAY
      IYOOBI=IFIX(AMOD(MJD,7.0))+1
      WRITE(1,6000)MJD,YEAR,MONTH,IDAY,NYOOBI(1,IYOOBI),NYOOBI(2,IYOOBI)
 6000 FORMAT(F6.0,I6"-"I2"-"I2"("A2,A1")")
      T2=BUF(60)
      T3=BUF(61)
      T4=BUF(62)
      TI1=BUF(63)
      TI2=IBUF64(1)
      IT2=T2
      IT3=T3
      IT4=T4
      ITI1=TI1
      ITI2=TI2
      WRITE(1,6010)IT2,IT3,IT4,ITI1,ITI2
 6010 FORMAT(I2,4I3" _")
      TX2=-1.0
      TX4=0.0
      READ(1,*)TX2,TX3,TX4
      TTT=0.0
      IF(TX2.NE.-1.0)TTT=(T2*60.0+T3)*60.0+T4
     1-((TX2*60.0+TX3)*60.0+TX4)
      IF(TTT.LT.0.0)TTT=TTT+86400.0
      TI=TI2*0.5
      T0=((T2*60.0+T3+TI)*60.0+T4)/60.0
      TT2=T2
      TT3=T3+TI2
```

```
      TT4=T4
      IF(TT3.LT.60.0)GO TO 110
      TT3=TT3-60.0
      TT2=TT2+1.0
      IF(TT2.LT.24.0)GO TO 110
      TT2=TT2-24.0
  110 CONTINUE
      WRITE(1,1000)
 1000 FORMAT("EARLY/LATE ??? SEC   _")
      READ(1,*)EARLY
      IF(EARLY.EQ.0.0)GO TO 60
   40 WRITE(1,1010)
 1010 FORMAT("SAT EARLY OR LATE?   _")
      READ(1,1020)IERLT
 1020 FORMAT(A2)
      IF(IERLT.EQ.2HEA)GO TO 60
      IF(IERLT.EQ.2HLA)GO TO 50
      GO TO 40
   50 EARLY=-EARLY
   60 CONTINUE
      WRITE(1,1050)
 1050 FORMAT("VOLTS/DEG FOR X,Y ?")
      V3=-99.0
      V4=-99.0
      READ(1,*)V4,V3
      IF(V4.NE.-99.0)GO TO 70
      V4=0.0
      V3=0.0
      GO TO 80
   70 CONTINUE
      IF(V3.EQ.-99.0)V3=V4
   80 CONTINUE
      WRITE(1,1060)
 1060 FORMAT("RANGE GATE MARGIN IN MILLISECONDS ?")
      RGM=1.0
      READ(1,*)RGM
      RGM=ABS(RGM)
      CALL EXEC(17,IFTRK,ILTRK,ISIZE)
      CALL EXEC(16,1,IFTRK,ISTRK)
      IF(ISTRK.EQ.0)STOP 5555
      DO 500 NSECT=0,46,2
        CALL EXEC(2,2,SECTOR,256,ISTRK,NSECT)
  500 CONTINUE
      NSECT=0
      CALL SUB21
  200 CALL SUB20(S(1),A(1))
      IF(TTT)768,768,767
  767 A7=A(7)+TTT
      N7=A7/60.0
      B7=N7
      A(7)=A7-B7*60.0
      A6=A(6)+B7
      N6=A6/60.0
      B6=N6
```

```
      A(6)=A6-B6*60.0
      A(5)=AMOD(A(5)+B6,24.0)
  768 CONTINUE
      IF(A(5).NE.T2.OR.A(6).NE.T3.OR.A(7).NE.T4)GO TO 200
  300 CALL SUB20(S(1),A(1),H(1))
      IF(TTT)778,778,777
  777 A7=A(7)+TTT
      N7=A7/60.0
      B7=N7
      A(7)=A7-B7*60.0
      A6=A(6)+B7
      N6=A6/60.0
      B6=N6
      A(6)=A6-B6*60.0
      A(5)=AMOD(A(5)+B6,24.0)
  778 CONTINUE
      IF(HOLD)GO TO 600
      IF(AMOD(A(7),10.0).NE.1.0)GO TO 390
      ASSIGN 310 TO KKK
      IS2=IS2+1
      SECTOR(IS2)=IS(A(5),A(6),A(7))
      IF(IS2.EQ.128)GO TO 340
      IF(IS2.GT.255)GO TO 350
  310 CONTINUE
      ASSIGN 330 TO KKK
      DO 330 I=1,6
        DO 320 J=1,4
          ISEC1=ISEC1*16+IN17(0)
  320   CONTINUE
        IS2=IS2+1
        SECTOR(IS2)=ISEC1
        IF(IS2.EQ.128)GO TO 340
        IF(IS2.GT.255)GO TO 350
  330 CONTINUE
  390 CONTINUE
      IF(A(5).EQ.TT2.AND.A(6).EQ.TT3.AND.A(7).GE.TT4)HOLD=.TRUE.
      XX=(((A(5)*60.0+A(6))*60.0+A(7)+EARLY)/60.0-T0)/TI
      CALL ECHEB(XX,COEFX,X)
      CALL ECHEB(XX,COEFY,Y)
      IA7=A(7)*10.0
      IF(MOD(IA7,100).NE.90)GO TO 690
      XX=0.01666667/TI+XX
      CALL ECHEB(XX,COEFD,D)
      D=(D-RGM)*10000.0
      CALL CODE
      WRITE(DASC,2000)D
 2000 FORMAT(F10.1)
      CALL OUT17(IRIGH(8,IAND(DASC1,7400B))
     1 +LEFT(4,IAND(DASC1,17B))+K+100000B)
      K=IAND(NOT(K),20000B)
      CALL OUT17(IRIGH(8,IAND(DASC2,7400B))
     1 +LEFT(4,IAND(DASC2,17B))+K+100000B)
      K=IAND(NOT(K),20000B)
      CALL OUT17(IRIGH(8,IAND(DASC3,7400B))
```

```
     1 +LEFT(4,IAND(DASC3,17B))+K+100000B)
       K=IAND(NOT(K),20000B)
       CALL OUT17(IRIGH(8,IAND(DASC4,7400B))
     1 +LEFT(4,IAND(DASC4,17B))+K+100000B)
       K=IAND(NOT(K),20000B)
       CALL OUT17(K+100000B)
       K=IAND(NOT(K),20000B)
   690 CONTINUE
       IF(XOLD)700,710,700
   700 GO TO (9,9,9,9,9,9,9,9,9,10,11,12,13,9,9),H(3)
    10 YOFF=YOFF-0.25
       GO TO 9
    11 YOFF=YOFF-0.0625
       GO TO 9
    12 YOFF=YOFF+0.25
       GO TO 9
    13 YOFF=YOFF+0.0625
     9 GO TO (8,8,8,8,8,8,8,8,8,20,21,22,23,8,8),H(4)
    20 XOFF=XOFF-1.0
       GO TO 8
    21 XOFF=XOFF-0.25
       GO TO 8
    22 XOFF=XOFF+1.0
       GO TO 8
    23 XOFF=XOFF+0.25
     8 DIFX=X-XOLD
       DIFY=Y-YOLD
       XOLD=X
       YOLD=Y
       IF(ISSW(15))7,77
     7 XOFF=0.0
       YOFF=0.0
    77 CONTINUE
       X=DIFX*XOFF+DIFY*YOFF+X
       Y=DIFY*XOFF-DIFX*YOFF+Y
   400 CONTINUE
       A33=A32+A32-A31
       IF(ABS(A31+A31-A30-A32).GT.0.0050 .OR.
      1   ABS(A33-A(3)).LT.0.0050)A33=A(3)
       A30=A31
       A31=A32
       A32=A(3)
       A43=A42+A42-A41
       IF(ABS(A41+A41-A40-A42).GT.0.0050 .OR.
      1   ABS(A43-A(4)).LT.0.0050)A43=A(4)
       A40=A41
       A41=A42
       A42=A(4)
       DY=Y-A33
       DX=X-A43
       S(3)=DY*V3
       S(4)=DX*V4
       CALL DISP(IFIX(SQRT((SIN(ABS(Y-Y0)*PI/180.0)*DX)**2
      1     +DY**2)*1000.0+0.5))
```

```
      GO TO 300
  710 XOLD=X
      YOLD=Y
      GO TO 400
  340 CALL EXEC(2,20002B,SECTOR,128,ISTRK,NSECT)
      NSECT=NSECT+1
      GO TO KKK
  350 CALL EXEC(2,20002B,SECTOR(129),128,ISTRK,NSECT)
      IS2=0
      NSECT=NSECT+1
      GO TO KKK
  600 CONTINUE
      IF(SAVED)GO TO 400
      CALL OUT17(0)
      SAVED=.TRUE.
      IF(IS2.EQ.0.OR.IS2.EQ.128)GO TO 620
      IF(IS2.GT.128)GO TO 610
      DO 605 I=IS2+1,128
        SECTOR(I)=-1
  605 CONTINUE
      CALL EXEC(2,2,SECTOR,128,ISTRK,NSECT)
      GO TO 620
  610 CONTINUE
      DO 615 I=IS2+1,256
        SECTOR(I)=-1
  615 CONTINUE
      CALL EXEC(2,2,SECTOR(129),128,ISTRK,NSECT)
  620 CONTINUE
      DO 630 NSECT=0,46,2
        CALL EXEC(1,2,SECTOR,256,ISTRK,NSECT)
        CALL EXEC(15,2,SECTOR,256,OBS,NSECT)
  630 CONTINUE
      GO TO 400
      E N D
      FUNCTION IS(H,M,S)
      REAL H,M,S
C
C      H,M,S ---> IS (5 SEC TAN'I)
C
      IH=H
      IM=M
      IM=(IH*60+IM)*12
      IS=IFIX(S)/5+IM
      RETURN
      E N D
$
```

14 EXAMPLE OF GARBLED MESSAGE

```
   1  RGXV   1 SATELLITE 6503201
```

```
 2  LJNB   2 EPOCH 44841 0.
 3  UPFX   3 POLY
 4  BXHQ   4 ELEMENTS
 5  XGMB   5  295.8909993368 5.1750954333
 6  XPWW   6  -285.5727377827 -4.2580895975
 7  BBQV   7  41.1852040842 -0.0000703896
 8  JSFX   8  0.0245612908 -0.0000007863
 9  PVBE   9  0.443794058 13.361609698 0.1547E-05
10  KBQK  10 LP 1
11  WEYS  11 LPTERMS
12  DIEA  12  -0.5155487E-05 0.6940138E-06 0.1495698E-07
13  ILBS  13  0.1582802E-02 0.1126232E-06 -0.136886E-06
14  JIXC  14  -0.9106587E-03 0.8305953E-05 -0.1207308E-06
15  VSUZ  15  0.5150395E-05 -0.5442665E-06 0.103492E-07
16  QJZJ  16  -0.2335753E-04 0.7230575E-06 0.1098189E-07
17  TIWX  17  0.5663218E-03
18  WQGP  18 SATELLITE 6508901
19  PLUW  19 EPOCH 44841 0.
20  VAIR  20 POLY 2 4 6 8 11
21  WTUU  21 ELEMENTS
22  WICD  22  320.9798549986 0.6529992864
23  HUPD  23  -264.9220084507 -2.2468187079
24  LRUG  24  59.381665911 -0.0001357304
25  KKFU  25  0.0716018693 -0.0000006387
26  GVUP  26  0.9488806765 11.9685182113 -0.1701E-07
27  YNYP  27 LP 1
28  BBKA  28 LPTERMS
29  OLMT  29  0.343096E-04 -0.5956032E-05 -0.5543068E-06
30  QFDS  30  0.3797774E-01 -0.7719765E-03 -0.3109254E-04
31  MFXZ  31  -0.3016641E-02 -0.8239531E-04 0.4745482E-05
32  CGEJ  32  -0.3432969E-04 0.2017524E-05 -0.4559234E-06
33  UQQA  33  0.1017943E-04 -0.9858999E-05 -0.3389772E-06
34  XNOX  34  0.124764E-02
35  CEEN  35 SATELLITE 7501001
36  JBMJ  36 EPOCH 44841 0.
37  JVUZ  37 POLY 2 4 6 8 11
38  AFCI  38 ELEMENTS
39  KHMO  39  4.4138266603 3.301446
40  LJZI  40  -104.1761189576 -3.9452107312
41  TIWZ  41  49.8219749607 -0.0000945507
42  MNGS  42  0.0206116614 -0.0000003924
43  LXYG  43  0.7487951892 13.820516052 0.3974E-06
44  ZAFS  44 LP 1
45  RTEH  45 LPTERMS
46  KKEK  46  0.176083E-05 0.2704648E-06 -0.9331303E-09
    FBJC

47  RNMC  47  0.3507322E-02 -0.7539671E-04 -0.6386387E-06
48  KCGT  48  -0.7784168E-03 -0.1737059E-05 0.3379114E-07
49  JZER  49  -0.1766713E-05 -0.3389569E-06 0.4769446E-08
50  BRDR  50  -0.1926351E-04 0.3931076E-06 0.3612812E-08
51  LARR  51  0.7825221E-03
52  SDKC  52 SATELLITE 7502701
53  NIGW  53 EPOCH 44841 0.
```

```
54  RAFY  54 POLY 2 6 12 20 31
55  LZJP  55 ELEMENTS
56  HCVV  56  -13.9675565466 -0.3505029
57  RHUS  57  108.5475194522 2.7289262833
58  XBQM  58  114.9918738264 -0.0001086945
59  SJBX  59  0.0011423637
60  EQFL  60  0.8548656536 14.1561613895 0.3753E-05
61  KXWZ  61 LP 1
62  YEUY  62 LPTERMS
63  HEVZ  63  0.2317628E-07 0.267506E-07 -0.2413003E-10
64  LDSD  64  -0.4964935E-02 0.6802461E-06 0.4182253E-09
65  JRFK  65  -0.2458245E-04 0.1280424E-08 -0.6838164E-11
66  PVRX  66  -0.2365297E-07 -0.2718816E-07 -0.1108268E-10
67  ETJV  67  -0.4758698E-04 -0.1022795E-07 -0.9547856E-11
68  UWMA  68  -0.8252701E-03
69  TJSS  69 SATELLITE 7603901
70  HPRT  70 EPOCH 44841 0.
71  KAEX  71 POLY 2 4 6 8 11
72  WPAW  72 ELEMENTS
73  XDKA  73  -153.3435731194 -0.2113375921
74  NQSS  74  332.7680027039 0.3428141931
75  AFNL  75  109.8718835741 0.0000516362
76  YXQZ  76  0.0044076345 0.0000001557
77  ANVL  77  0.4294374541 6.386634719 -0.2181E-08
78  HJEE  78 LP 1
79  DBNG  79 LPTERMS
80  OEYK  80  -0.2199088E-06 0.1054664E-08 0.3301661E-11
81  BKOT  81  -0.2120306E-03 0.1327438E-06 0.4854907E-09
82  JLTC  82  0.4925433E-04 -0.201312E-07 0.2751856E-10
83  SSJM  83  0.2191907E-06 -0.4506719E-09 0.4509785E-11
84  BUJC  84  0.1374206E-05 0.2731741E-08 0.7344228E-11
85  YENQ  85  0.5392146E-03
86  PZQK  86END
```

15 TELEX CORRECTION

```
FTN,L
      PROGRAM CHECK
      DIMENSION N(4)
      COMMON II,J,K1,K2,K3,K4,KK1,KK2,KK3,KK4,LINE(64)
      CALL INIT
      II=1
      K1=0
      K2=0
      K3=0
      K4=0
2000  CONTINUE
      WRITE(1,6500)
6500  FORMAT("> ")
      READ(1,5000)KX1,KX2,KX3,KX4,LINE
```

```
5000  FORMAT(68R1)
      KX1=KX1-101B
      KX2=KX2-101B
      KX3=KX3-101B
      KX4=KX4-101B
      DO 1000 I=64,1,-1
      IF(LINE(I).NE.40B)GO TO 2100
1000  CONTINUE
      STOP
2100  CONTINUE
      IQ=0
      DO 1100 IX=I,1,-1
      IF(LINE(IX).NE.77B)GO TO 1100
      IF(IQ.EQ.4)STOP 7777
      IQ=IQ+1
      N(IQ)=IX
1100  CONTINUE
      IF(IQ.EQ.0)GO TO 2000
      NN=10**IQ-1
      J=I
      DO 1200 NNN=0,NN
      NA=NNN
      DO 1150 NB=1,IQ
      LINE(N(NB))=MOD(NA,10)+60B
      NA=NA/10
1150  CONTINUE
      CALL CHEK
      IF(KK1.NE.KX1)GO TO 1200
      IF(KK2.NE.KX2)GO TO 1200
      IF(KK3.NE.KX3)GO TO 1200
      IF(KK4.NE.KX4)GO TO 1200
      WRITE(1,6000)NNN
6000  FORMAT(I5)
1200  CONTINUE
      GO TO 2000
      END
$
```

16 PREDICTION FOR OBSERVATION

```
FTN
      PROGRAM CHEBT
C      EXTERNAL RCHEB
C     (EXTERNAL) ECHEB(ECHE.(ASMB) O TUZUKETE ASMB)
C      REV 76-05-03(MON)
C      PARAMETER(0--9) DE CHEBF FILE O
C      "CHEBF","CHE01",...,"CHE09" TO ERABU.
C      REV 76-10-23(SAT)
C      YOHO(SS:KZ1) NI AWASERU.
C      TIME O FILE NO USIRO NI NOKOSU.
```

```
C       ZANSA O INSATU.
C       REV 77-04-09(SAT)
C       BUF(64) _ IFIX(TI20) & ID
C       REV 78-01-25(WED)
C       CHEB3 (SEGMENT)
C       REV 80-06-29(SAT)
C       SEE LOG BOOK
C       REV 80-11-29(SAT)
C       ADAPT TO NEW ELEMENTS FORMAT
C       (MEMORANDA 1 JUN 79 FM PEARLMAN,
C                  15 AUG 79 FM ROMAINE/LATIMER;
C        "NON-SINGULAR VARIABLES FOR EPHEMERIDES" 22 MAY 79)
C       REV 81-04-27(MON)
C       2ND WORDS OF BUF(20),BUF(39),BUF(58) _ TO (EPOCH)
C       REV 81-07-31(FRI) SOLAR ECLIPSE
C       LATITUDE 36.0058 -> 36.0059 (LINES 339,340)
C       LONGITUDE 139.1917 -> 139.1920 (LINES 341,342)
C       X -0.613091 -> -0.613098 (LINE 497)
C       Z 0.584687 -> 0.584693 (LINE 499)
      DOUBLE PRECISION X(30,3),FXJ(3),GC(39),DXX
      DIMENSION COEFX(19),COEFY(19),COEFD(19),BUF(64)
      DIMENSION IPARAM(5)
      INTEGER SEG1(3),LASRIO(3),CHEBF(3),NSECT(2),IBUF64(2)
      INTEGER SEG3(3)
      LOGICAL TEAST
      INTEGER N(5)
      DIMENSION ELEM(5,4),EL(5),AMP(6)
      DIMENSION ITO(3),IBUF20(2),IBUF39(2),IBUF58(2)
      DOUBLE PRECISION ANOM(5),T,TO,SID,T1,T11,ANOM1,XY(3)
      EXTERNAL YOHO
      COMMON A,A8,A9,AI2,AI3,AI4,AI8,AI9,AK8,AK9,AL,AL1,ANOM,ANOM1,AR,C,
     1       C22,C2A,C31,CA,CAL,CAL1,CAZERR,CC,CELERR,CF,CF1,CF2,CF3,
     2       CI1,CI2,CI3,CL1,CL2,CL3,CL4,CO1,CO2,CO3,CO4,CR1,CR2,CR3,
     3       CR4,CS,E,E12,E2,ED,EL,ELEM,ELEM1,ELEM2,ES,FAC,M,ID,NI,J,K,
     4       O2,O3,P2,PAI,Q2,Q3,R,RE,S2,S22,S2A,S3,SA,SAL,SAL1,SAZERR,
     5       SC,SELERR,SF,SF1,SF2,SF3,SI,SIO,SID,SL,SM,SS,T,TO,T1,T11,
     6       U2,U3,U4,TEAST,TI,UI1,UI2,TM,V,XO,X1,X2,X3,X4,XS,XY,Y,YO,
     7       Y1,Y2,Y3,Y4,YS,Z,Z1,Z2,Z3,Z4,ZS,P,Q
     8       T10,T20,T30,T40,TI10,TI20
      COMMON N,AMP
      EQUIVALENCE (NSECT,BUF),(IBUF64,BUF(64)),
     1            (COEFX,BUF(2)),(COEFY,BUF(21)),(COEFD,BUF(40))
      EQUIVALENCE (ITO,TO),(IBUF20,BUF(20)),(IBUF39,BUF(39)),
     1            (IBUF58,BUF(58))
      DATA EPS,NPLMAX,N2/0.2938736E-38,30,39/
      DATA SEG1/2HCH,2HEC,2H1 /,LASRIO,CHEBF/2HLA,2HSR,2HIO,
     1          2HCH,2HEB,2HF /,SEG3/2HCH,2HEC,2H3 /
      CALL RMPAR(IPARAM)
      CALL TTY
      IF(IPARAM(1).GT.9)GO TO 3500
      IF(IPARAM(1).EQ.0)GO TO 3510
      CHEBF(2)=2HE0
      CHEBF(3)=2H0 +IPARAM(1)*400B
      GO TO 3510
```

```
3000 CONTINUE
     STOP 7777
3510 CONTINUE
     CALL EXEC(8,SEG1)
     CALL EXEC(8,SEG3)
  10 WRITE(1,1004)
1004 FORMAT("#?")
     NPL=12
     READ(1,*)NPL
     NPL=MIN0(NPL,18)
     CALL CHEBY(3,NPL,NPLMAX,N2,YOHO,X,FXJ,GC)
     DO 20 I=1,NPL
  20 WRITE(6,1000)X(I,1),X(I,2),X(I,3)
1000 FORMAT(2F10.3,F13.6)
     DO 30 I=1,NPL
     NPLA=NPL+1-I
     COEFX(I)=X(NPLA,1)
     IF(COEFX(I).EQ.0.0)COEFX(I)=EPS
     COEFY(I)=X(NPLA,2)
     IF(COEFY(I).EQ.0.0)COEFY(I)=EPS
     COEFD(I)=X(NPLA,3)
     IF(COEFD(I).EQ.0.0)COEFD(I)=EPS
  30 CONTINUE
     COEFX(NPL+1)=0.0
     COEFY(NPL+1)=0.0
     COEFD(NPL+1)=0.0
     BUF(59)=T10
     BUF(60)=T20
     BUF(61)=T30
     BUF(62)=T40
     BUF(63)=TI10
     IBUF64(1)=TI20
     IBUF64(2)=ID
     IBUF20(2)=IT0(1)
     IBUF39(2)=IT0(2)
     IBUF58(2)=IT0(3)
     CALL EXEC(23,LASRIO)
     CALL EXEC(15,2,BUF,128,CHEBF,0)
     T2=T20
     T3=T30
     T4=T40
     TI1=TI10
     TI2=TI20
     IT20=T20
     IT30=T30
     IT40=T40
     ITI10=TI10
     ITI20=TI20
     WRITE(6,1010)IT20,IT30,IT40,ITI10,ITI20
1010 FORMAT(I2,4I3)
     DX=TI1/TI2/30.0
     II=2.0/DX
     DO 40 I=0,II
     XX=DX*I-1.0
```

```
      CALL ECHEB(XX,COEFX,XXX)
      CALL ECHEB(XX,COEFY,YYY)
      CALL ECHEB(XX,COEFD,DDD)
      DXX=XX
      CALL YOHO(DXX,FXJ)
      ERRORX=XXX-FXJ(1)
      ERRORY=YYY-FXJ(2)
      ERRORD=DDD-FXJ(3)
      ERRORX=ERRORX*1000.0
      ERRORY=ERRORY*1000.0
      ERRORD=ERRORD*1000.0
      IERRRX=SIGN(ABS(ERRORX)+0.5,ERRORX)
      IERRRY=SIGN(ABS(ERRORY)+0.5,ERRORY)
      IERRRD=SIGN(ABS(ERRORD)+0.5,ERRORD)
      WRITE(6,1200)XXX,IERRRX,YYY,IERRRY,DDD,IERRRD
 1200 FORMAT(3(F8.3"("I3") "))
   40 CONTINUE
      GO TO 10
      END
      PROGRAM CHEC1,5
      LOGICAL TEAST
      INTEGER PN(5),N(5)
      DIMENSION ELEM(5,4),EL(5),AMP(6)
      DOUBLE PRECISION ANOM(5),T,T0,SID,T1,T11,ANOM1,XY(3)
      COMMON A,A8,A9,AI2,AI3,AI4,AI8,AI9,AK8,AK9,AL,AL1,ANOM,ANOM1,AR,C,
     1       C22,C2A,C31,CA,CAL,CAL1,CAZERR,CC,CELERR,CF,CF1,CF2,CF3,
     2       CI1,CI2,CI3,CL1,CL2,CL3,CL4,CO1,CO2,CO3,CO4,CR1,CR2,CR3,
     3       CR4,CS,E,E12,E2,EE,EL,ELEM,ELEM1,ELEM2,ES,FAC,I,ID,II,J,K,
     4       O2,O3,P2,PAI,Q2,Q3,R,RE,S2,S22,S2A,S3,SA,SAL,SAL1,SAZERR,
     5       SC,SELERR,SF,SF1,SF2,SF3,SI,SI0,SID,SL,SM,SS,T,T0,T1,T11,
     6       T2,T3,T4,TEAST,TI,TI1,TI2,TM,X,X0,X1,X2,X3,X4,XS,XY,Y,Y0,
     7       Y1,Y2,Y3,Y4,YS,Z,Z1,Z2,Z3,Z4,ZS,P,Q,
     8       T10,T20,T30,T40,TI10,TI20
      COMMON N,AMP
      INTEGER OPNTB(128),TRBUF(256),NOTRB(2),ERRNO,FNAME(3),
     1        PAKNO,SCODE,LINE(36,4)
      LOGICAL DEFINE
      EQUIVALENCE (NOTRB(2),MAXPK)
      DATA NOTRB/1/,MAXPK/1/,DEFINE/.FALSE./,PAKNO,SCODE/9,0/
    1 ID=0
      IGEN=0
      WRITE(1,2)
    2 FORMAT(" SATELLITE? _")
      READ(1,*)ID,IGEN
      IF(ID.EQ.0)GO TO 4
      IF(IGEN.LT.0 .OR. IGEN.GT.9)GO TO 1
      IF(DEFINE)GO TO 3
C      DEFINE
      CALL EXEC(24,1,OPNTB,128,TRBUF,NOTRB,2,ERRNO)
      IF(ERRNO.NE.0)STOP 6666
      DEFINE=.TRUE.
    3 CONTINUE
C      OPEN
      FNAME(1)=(ID/1000+60B)*256+MOD(ID,1000)/100+60B
```

```
      FNAME(2)=(MOD(ID,100)/10+60B)*256+MOD(ID,10)+60B
      FNAME(3)=(IGEN+60B)*256
      CALL EXEC(24,4,FNAME,PAKNO,1,SCODE,1,ERRNO)
      IF(ERRNO.NE.0)GO TO 1
C      READ
      CALL EXEC(24,6,FNAME,1,LINE,ERRNO)
      IF(ERRNO.NE.0)STOP 6666
      CALL CODE
      READ(LINE,*) ID,TO,PN
      CALL EXEC(3,1106B,-2)
      WRITE(6,6) ID
      N(1)=PN(1)
      DO 5 I=2,5
    5 N(I)=PN(I)-PN(I-1)
      CALL EXEC(24,6,FNAME,2,LINE(1,1),ERRNO)
      IF(ERRNO.NE.0)STOP 6666
      CALL EXEC(24,6,FNAME,3,LINE(1,2),ERRNO)
      IF(ERRNO.NE.0)STOP 6666
      CALL CODE
      READ(LINE,*) ((ELEM(I,J),J=1,N(I)),I=1,4)
      CALL EXEC(24,6,FNAME,4,LINE,ERRNO)
      IF(ERRNO.NE.0)STOP 6666
      CALL CODE
      READ(LINE,*) (ANOM(I),I=1,N(5))
      CALL EXEC(24,6,FNAME,5,LINE(1,1),ERRNO)
      IF(ERRNO.NE.0)STOP 6666
      CALL EXEC(24,6,FNAME,6,LINE(1,2),ERRNO)
      IF(ERRNO.NE.0)STOP 6666
      CALL CODE
      READ(LINE,*) AMP
      GO TO 8
    4 CONTINUE
      READ(5,*) ID,TO
      CALL EXEC(3,1106B,-2)
      WRITE(6,6) ID
    6 FORMAT(10H SATELLITE,I6)
      READ(5,*) PN
      N(1)=PN(1)
      DO 7 I=2,5
    7 N(I)=PN(I)-PN(I-1)
      READ(5,*) ((ELEM(I,J),J=1,N(I)),I=1,4)
      READ(5,*) (ANOM(I),I=1,N(5))
      READ(5,*) AMP
    8 CONTINUE
      FAC=360./6.2831853
      PAI=6.2831853
      P2=0.00108264
      READ(1,*) T1,T2,T3,T4,TI1,TI2
      T10=T1
      T20=T2
      T30=T3
      T40=T4
      TI10=TI1
      TI20=TI2
```

```
      CALL EXEC(29)
      CALL CHEBT
      END
      PROGRAM CHEC3,5
      LOGICAL TEAST
      INTEGER N(5)
      DIMENSION LINES(108),LINEAE(36),LINEXY(36),LINEPQ(36),LINEF(3)
      DIMENSION ELEM(5,4),EL(5),AMP(6)
      DOUBLE PRECISION ANOM(5),T,T0,SID,T1,T11,ANOM1,XY(3)
      DOUBLE PRECISION TT(4)
      COMMON A,A8,A9,AI2,AI3,AI4,AI8,AI9,AK8,AK9,AL,AL1,ANOM,ANOM1,AR,C,
     1       C22,C2A,C31,CA,CAL,CAL1,CAZERR,CC,CELERR,CF,CF1,CF2,CF3,
     2       CI1,CI2,CI3,CL1,CL2,CL3,CL4,CO1,CO2,CO3,CO4,CR1,CR2,CR3,
     3       CR4,CS,E,E12,E2,EE,EL,ELEM,ELEM1,ELEM2,ES,FAC,I,ID,II,J,K,
     4       O2,O3,P2,PAI,Q2,Q3,R,RE,S2,S22,S2A,S3,SA,SAL,SAL1,SAZERR,
     5       SC,SELERR,SF,SF1,SF2,SF3,SI,SI0,SID,SL,SM,SS,T,T0,T1,T11,
     6       T2,T3,T4,TEAST,TI,TI1,TI2,TM,X,X0,X1,X2,X3,X4,XS,XY,Y,Y0,
     7       Y1,Y2,Y3,Y4,YS,Z,Z1,Z2,Z3,Z4,ZS,P,Q,
     8       T10,T20,T30,T40,TI10,TI20
      COMMON N,AMP
      EQUIVALENCE (LINES,LINEAE),(LINES(37),LINEXY),(LINES(73),LINEPQ)
      DATA LINEF/2HLI,2HNE,2HF /
      WRITE(1,1000)
1000  FORMAT("AZ,EL?")
      READ(1,5000)LINEAE
5000  FORMAT(36A2)
      DO 1100 I=1,35
      IF(LINEAE(I).NE.2H  )GO TO 2300
1100  CONTINUE
      CALL EXEC(14,2,LINES,108,LINEF,0)
      DO 1200 I=35,2,-1
      IF(LINEAE(I).NE.2H  )GO TO 2000
1200  CONTINUE
      I=1
2000  CONTINUE
      WRITE(1,5000)(LINEAE(J),J=1,I)
      CALL CODE
      READ(LINEAE,*)AZERR,ELERR
      WRITE(1,1001)
1001  FORMAT("X,Y?")
      DO 1300 I=35,2,-1
      IF(LINEXY(I).NE.2H  )GO TO 2100
1300  CONTINUE
      I=1
2100  CONTINUE
      WRITE(1,5000)(LINEXY(J),J=1,I)
      CALL CODE
      READ(LINEXY,*)X0,Y0
      WRITE(1,1004)
1004  FORMAT("P,Q?")
      DO 1400 I=35,2,-1
      IF(LINEPQ(I).NE.2H  )GO TO 2200
1400  CONTINUE
      I=1
```

```
2100 CONTINUE
     WRITE(1,5000)(LINEPQ(J),J=1,1)
     P=0.0
     Q=0.0
     CALL CODE
     READ(LINEPQ,*)P,Q
     GO TO 2400
2300 CONTINUE
     LINEAE(36)=2H,,
     CALL CODE
     READ(LINEAE,*)AZERR,ELERR
     WRITE(1,1001)
     READ(1,5000)LINEXY
     LINEXY(36)=2H,,
     CALL CODE
     READ(LINEXY,*)XO,YO
     WRITE(1,1004)
     READ(1,5000)LINEPQ
     LINEPQ(36)=2H,,
     P=0.0
     Q=0.0
     CALL CODE
     READ(LINEPQ,*)P,Q
     CALL EXEC(15,2,LINES,108,LINEF,0)
2400 CONTINUE
     SAZERR=SIN(AZERR/FAC)
     CAZERR=COS(AZERR/FAC)
     SELERR=SIN(ELERR/FAC)
     CELERR=COS(ELERR/FAC)
     WRITE(1,1002)
1002 FORMAT("E,W?")
     READ(1,1003) IEAST
1003 FORMAT(A1)
     TEAST=IEAST.EQ.1HE
     T11=T1+(T2+(T3+T4/60.)/60.)/24.
     RE=TI2*60./TI1
     II=INT(RE)
     TI=TI1/24./60./60.
     T1=T11-TO
     TM=T1+TI2/60./48.
     AR=ELEM(1,1)+ELEM(1,2)*TM
     AR=AR/FAC
     SA=SIN(AR)
     CA=COS(AR)
     TT(2)=T1
     TT(3)=TM**2
     TT(4)=TM**3
     DO 8 I=1,4
     DO 8 J=2,N(I)
   8 ELEM(I,1)=ELEM(I,1)+ELEM(I,J)*TT(J)
     DO 9 J=2,N(5)
   9 ANOM(1)=ANOM(1)+ANOM(J)*TT(J)
     TT(2)=TM
     DO 4 J=3,N(5)
```

```
 4 ANOM(2)=ANOM(2)+FLOAT(J-1)*ANOM(J)*TT(J-1)
   ELEM(2,1)=ELEM(2,1)+AMP(2)*CA
   ELEM(3,1)=ELEM(3,1)+AMP(3)*SA
   XE=ELEM(4,1)*CA+AMP(1)*COS(ELEM(1,1)/FAC)
   ET=ELEM(4,1)*SA+AMP(4)*SIN(ELEM(1,1)/FAC)+AMP(6)
   ELEM(4,1)=SQRT(XE**2+ET**2)
   ELEM(5,1)=ATAN2(ET,XE)*FAC
   ANOM(1)=ANOM(1)+AMP(5)*CA/PAI-(ELEM(5,1)-ELEM(1,1))/360.
   ELEM(1,1)=ELEM(5,1)
   AI8=COS(36.0059/FAC)
   AI9=SIN(36.0059/FAC)
   AK8=COS(139.1920/FAC)
   AK9=SIN(139.1920/FAC)
   AI2=COS(ELEM(3,1)/FAC)
   AI3=SIN(ELEM(3,1)/FAC)
   E=ELEM(4,1)
   E2=E**2
   E12=1.-E2
   ES=SQRT(E12)
   T4=T4-TI1
   A=(17.04355/SNGL(ANOM(2)))**0.6666667
   C=1.0-1.5*P2*(1.0-1.5*AI3**2)*ES/(A*E12)**2
   A=(17.04355*SQRT(C)/SNGL(ANOM(2)))**0.6666667
   AI4=AI3**2
   P2=P2/(A*E12)**2
   CL1=0.75*P2*(4.-5.*AI4)*ELEM(4,1)
   CL2=-0.25*P2*(3.-5.*AI4)*E
   CL3=-0.25*P2*(3.-7.*AI4)
   CL4=-0.25*P2*AI2**2*E
   CR1=-0.5*P2*(1.-1.5*AI4)*A*E12
   CR2=CR1*(1.-ES)/E
   CR3=-CR1/ES
   CR4=0.25*A*AI4*P2*E12
   CI1=0.75*AI2*AI3*P2
   CI2=E*CI1
   CI3=CI2/3.
   CO1=-1.5*P2*AI2*E
   CO2=0.75*AI2*P2
   CO3=CO2*E
   CO4=CO3/3.
   K=0
   SID=6.6521845/24.0+1.0027379093D0*(T11-42778.0D0)
10 IF(SID-0.5) 12,11
11 SID=SID-1.0D0
   GO TO 10
12 IF(ANOM(1)-0.5) 14,13
13 ANOM(1)=ANOM(1)-1.0D0
   GO TO 12
14 SI=SID*PAI
   SL=-78.24+0.985647*(T11+TI2/60./48.-42780.0)
   SM=SL+77.49
   SL=SL/FAC
   SM=SM/FAC
   SL=SL+0.03344*SIN(SM)
```

```
      XS=COS(SL)
      YS=COS(23.442/FAC)*SIN(SL)
      ZS=SIN(23.442/FAC)*SIN(SL)
      C22=2.3E-06*ANOM(2)/(A*E12)**2
      S22=-1.33E-06*ANOM(2)/(A*E12)**2
      C31=2.03E-06*ANOM(2)/(A*E12)**3
      C41=0.53E-06*ANOM(2)/(A*E12)**4
      S41=0.75*C41
      AL=SI-ELEM(2,1)/FAC+TI2*PAI/60./48.
      C2A=COS(2.0*AL)
      S2A=SIN(2.0*AL)
      CC=C22*S2A+S22*C2A
      SS=C22*C2A-S22*S2A
      CA=COS(AL)
      SA=SIN(AL)
      CC4=S41*CA+C41*SA
      SS4=S41*SA-C41*CA
      CS=AI4*(1.0+3.0*AI2)-0.8*(1.0+AI2)
      SC=AI4*(1.0-3.0*AI2)-0.8*(1.0-AI2)
      AL=AL-ELEM(1,1)/FAC
      AL1=AL+2.0*ELEM(1,1)/FAC
      SAL=SIN(AL)
      CAL=COS(AL)
      SAL1=SIN(AL1)
      CAL1=COS(AL1)
      ELEM(2,1)=ELEM(2,1)+FAC*(CC*AI2+SS4*(4.-29.*AI4+28.*
     1AI4**2)/AI3)
      ELEM(3,1)=ELEM(3,1)+FAC*(SS*AI3+CC4*AI2*(4.-7.*AI4))
      E=E+E12*C31*(CAL*CS-CAL1*SC)
      ELEM(1,1)=ELEM(1,1)+CC*(1.5*AI4-AI2**2)*FAC
     1+C31*(CS*SAL+SC*SAL1)*FAC/E
     2-SS4*FAC*(4.-69.*AI4+98.*AI4**2)*AI2/AI3
      ANOM(1)=ANOM(1)+ES*(1.5*CC*AI4-C31*(CS*SAL+SC*SAL1)/E)/PAI
      IF(ID.NE.6589) GO TO 15
      ELEM(1,1)=ELEM(1,1)-0.38E-04*FAC*DSIN(ANOM(1)*PAI
     1-11.*ELEM(1,1)/FAC-12.*AL)
   15 CONTINUE
      ANOM1=ANOM(1)
      ELEM1=ELEM(1,1)
      ELEM2=ELEM(2,1)
      SI0=SI
      CALL EXEC(29)
      CALL CHEBT
      END
      PROGRAM CHEC2,5
      LOGICAL TEAST
      INTEGER N(5)
      DIMENSION ELEM(5,4),EL(5),AMP(6)
      DOUBLE PRECISION ANOM(5),T,T0,SID,T1,T11,ANOM1,XY(3)
      COMMON A,A8,A9,AI2,AI3,AI4,AI8,AI9,AK8,AK9,AL,AL1,ANOM,ANOM1,AR,C,
     1       C22,C2A,C31,CA,CAL,CAL1,CAZERR,CC,CELERR,CF,CF1,CF2,CF3,
     2       CI1,CI2,CI3,CL1,CL2,CL3,CL4,CO1,CO2,CO3,CO4,CR1,CR2,CR3,
     3       CR4,CS,E,E12,E2,EE,EL,ELEM,ELEM1,ELEM2,ES,FAC,I,ID,II,J,K,
     4       O2,O3,P2,PAI,Q2,Q3,R,RE,S2,S22,S2A,S3,SA,SAL,SAL1,SAZERR,
```

```
     5       SC,SELERR,SF,SF1,SF2,SF3,SI,SI0,SID,SL,SM,SS,T,T0,T1,T11,
     6       T2,T3,T4,TEAST,TI,TI1,TI2,TM,X,X0,X1,X2,X3,X4,XS,XY,Y,Y0,
     7       Y1,Y2,Y3,Y4,YS,Z,Z1,Z2,Z3,Z4,ZS,P,Q,
     8       T10,T20,T30,T40,TI10,TI20
      COMMON N,AMP
      TI=(1D0+T)/2880D0*TI2
      ANOM(1)=ANOM(2)*TI+ANOM1
      ELEM(1,1)=ELEM(1,2)*TI+ELEM1
      ELEM(2,1)=ELEM(2,2)*TI+ELEM2
      SI=1.0027379*PAI*TI+SI0
      EL(1)=ELEM(1,1)/FAC
      EL(5)=ANOM(1)*PAI
      EL(3)=EL(5)
   16 EL(4)=EL(5)+E*SIN(EL(3))
      EE=ABS(EL(4)-EL(3))
      IF(EE-0.00001) 18,17
   17 EL(3)=EL(4)
      GO TO 16
   18 X=A*(COS(EL(4))-E)
      Y=A*SQRT(1.0-E**2)*SIN(EL(4))
      Z=ATAN(Y/X)
      R=SQRT(X**2+Y**2)
      IF(X) 19,20
   19 Z=Z+PAI/2.0
   20 EE=Z-EL(5)
      IF(ABS(EE)-1.0) 24,21
   21 IF(EE) 22,23
   22 EE=EE+PAI
      GO TO 24
   23 EE=EE-PAI
   24 EL(4)=Z+EL(1)
      EE=EE/E
      SF=SIN(Z)
      CF=COS(Z)
      AL=2.0*EL(4)
      CF2=COS(AL)
      SF2=SIN(AL)
      AL=AL-Z
      CF1=COS(AL)
      SF1=SIN(AL)
      AL=AL+2.0*Z
      CF3=COS(AL)
      SF3=SIN(AL)
      EL(4)=EL(4)+CL1*(EE+SF)+CL2*SF1+CL3*SF2+CL4*SF3
      R=R+CR1+CR2*CF+CR3*R/A
      R=R+CR4*CF2
      EL(2)=ELEM(2,1)/FAC+CO1*(EE+SF)+CO2*SF2+CO3*SF1+CO4*SF3
      EL(3)=ELEM(3,1)/FAC+CI1*CF2+CI2*CF1+CI3*CF3
      O2=COS(EL(4))
      O3=SIN(EL(4))
      Q2=COS(EL(2))
      Q3=SIN(EL(2))
      AI3=SIN(EL(3))
      AI2=COS(EL(3))
```

```
      X=R*(O2*Q2-O3*Q3*AI2)
      Y=R*(O2*Q3+O3*Q2*AI2)
      Z=R*O3*AI3
      S2=COS(SI)
      S3=SIN(SI)
      X1=S2*X+S3*Y+0.613098
      Y1=S2*Y-S3*X-0.529362
      Z1=Z-0.584693
      X2=Y1*AK8-X1*AK9
      Y2=Z1*AI8-(X1*AK8+Y1*AK9)*AI9
      Z2=Z1*AI9+(X1*AK8+Y1*AK9)*AI8
        A8=SQRT(X2**2+Y2**2)
      A9=(ATAN(Z2/A8)+0.00029*A8/Z2)*FAC
      XY(3)=SQRT(X1**2+Y1**2+Z1**2)*42.5505
      A8=ATAN(X2/Y2)*FAC
      IF(Y2) 25,26
   25 A8=A8+180.
   26   Y2=COS(A9/FAC)
      X2=COS(A8/FAC)*Y2
      Y2=SIN(A8/FAC)*Y2
      Z2=SIN(A9/FAC)
      X3=X2*CAZERR+Y2*SAZERR
      Y3=Y2*CAZERR-X2*SAZERR
      Z3=Z2
      X4=X3*CELERR-Z3*SELERR
      Y4=Y3
      Z4=Z3*CELERR+X3*SELERR
      X=ATAN2(Z4,-Y4)*FAC
      Y=ATAN2(SQRT(Y4**2+Z4**2),-X4)*FAC
      DELTAX=(-Q-P*COS(Y/FAC))/SIN(Y/FAC)
      IF(TEAST) GO TO 262
      XY(1)=XO+(180.0-X)+DELTAX
      XY(2)=YO-Y
      CALL EXEC(29)
  262 XY(1)=XO-X-DELTAX
      XY(2)=YO+Y
      CALL EXEC(29)
      CALL CHEBT
      END
      SUBROUTINE YOHO(ST,SXY)
      DOUBLE PRECISION ST,SXY(3)
      INTEGER SEG2(3)
      LOGICAL TEAST
      INTEGER N(5)
      DIMENSION ELEM(5,4),EL(5),AMP(6)
      DOUBLE PRECISION ANOM(5),T,TO,SID,T1,T11,ANOM1,XY(3)
      COMMON A,A8,A9,AI2,AI3,AI4,AI8,AI9,AK8,AK9,AL,AL1,ANOM,ANOM1,AR,C,
     1       C22,C2A,C31,CA,CAL,CAL1,CAZERR,CC,CELERR,CF,CF1,CF2,CF3,
     2       CI1,CI2,CI3,CL1,CL2,CL3,CL4,CO1,CO2,CO3,CO4,CR1,CR2,CR3,
     3       CR4,CS,E,E12,E2,EE,EL,ELEM,ELEM1,ELEM2,ES,FAC,I,ID,II,J,K,
     4       O2,O3,P2,PAI,Q2,Q3,R,RE,S2,S22,S2A,S3,SA,SAL,SAL1,SAZERR,
     5       SC,SELERR,SF,SF1,SF2,SF3,SI,SIO,SID,SL,SM,SS,T,TO,T1,T11,
     6       T2,T3,T4,TEAST,TI,TI1,TI2,TM,X,XO,X1,X2,X3,X4,XS,XY,Y,YO,
     7       Y1,Y2,Y3,Y4,YS,Z,Z1,Z2,Z3,Z4,ZS,P,Q,
```

```
     8       T10,T20,T30,T40,TI10,TI20
      COMMON N,AMP
      DATA SEG2/2HCH,2HEC,2H2 /
      T=ST
      CALL EXEC(8,SEG2)
      SXY(1)=XY(1)
      SXY(2)=XY(2)
      SXY(3)=XY(3)
      RETURN
      END
$
```

17 SATELLITE PREDICTION

```
FTN4,L
      PROGRAM YOHO
      INTEGER PN(5),N(5)
      DIMENSION ELEM(5,4),EL(5),AMP(6)
      DOUBLE PRECISION ANOM(5),T,T0,SID,T1,T11,TT(4)
      READ(5,*) ID,T0
      WRITE(6,6) ID
    6 FORMAT(10H SATELLITE,I6)
      READ(5,*) PN
      N(1)=PN(1)
      DO 5 I=2,5
    5 N(I)=PN(I)-PN(I-1)
      READ(5,*) ((ELEM(I,J),J=1,N(I)),I=1,4)
      READ(5,*) (ANOM(I),I=1,N(5))
      READ(5,*) AMP
      FAC=360./6.2831853
      PAI=6.2831853
      P2=0.00108264
      READ(1,*) T1,T2,T3,T4,TI1,TI2
      T11=T1+(T2+(T3+T4/60.)/60.)/24.
      RE=TI2*60./TI1
      II=INT(RE)
      TI=TI1/24./60./60.
      T1=T11-T0
      TM=T1+TI2/60./48.
      AR=ELEM(1,1)+ELEM(1,2)*TM
      AR=AR/FAC
      SA=SIN(AR)
      CA=COS(AR)
      TT(2)=T1
      TT(3)=TM**2
      TT(4)=TM**3
      DO 8 I=1,4
      DO 8 J=2,N(I)
    8 ELEM(I,1)=ELEM(I,1)+ELEM(I,J)*TT(J)
      DO 9 J=2,N(5)
    9 ANOM(1)=ANOM(1)+ANOM(J)*TT(J)
```

```
      TT(2)=TM
      DO 4 J=3,N(5)
    4 ANOM(2)=ANOM(2)+FLOAT(J-1)*ANOM(J)*TT(J-1)
      DO 3 I=1,6
      IF(AMP(I).NE.0.0)GO TO 2
    3 CONTINUE
      GO TO 1
    2 CONTINUE
      ELEM(2,1)=ELEM(2,1)+AMP(2)*CA
      ELEM(3,1)=ELEM(3,1)+AMP(3)*SA
      XE=ELEM(4,1)*CA+AMP(1)*COS(ELEM(1,1)/FAC)
      ET=ELEM(4,1)*SA+AMP(4)*SIN(ELEM(1,1)/FAC)+AMP(6)
      ELEM(4,1)=SQRT(XE**2+ET**2)
      ELEM(5,1)=ATAN2(ET,XE)*FAC
      ANOM(1)=ANOM(1)+AMP(5)*CA/PAI-(ELEM(5,1)-ELEM(1,1))/360.
      ELEM(1,1)=ELEM(5,1)
    1 CONTINUE
      AI8=COS(36.0059/FAC)
      AI9=SIN(36.0059/FAC)
      AK8=COS(139.1920/FAC)
      AK9=SIN(139.1920/FAC)
      AI2=COS(ELEM(3,1)/FAC)
      AI3=SIN(ELEM(3,1)/FAC)
      E=ELEM(4,1)
      E2=E**2
      E12=1.-E2
      ES=SQRT(E12)
      T4=T4-TI1
      A=(17.04355/ANOM(2))**(2.0/3.0)
      C=1.0-1.5*P2*(1.0-1.5*AI3**2)*ES/(A*E12)**2
      A=(17.04355*SQRT(C)/ANOM(2))**(2.0/3.0)
      AI4=AI3**2
      P2=P2/(A*E12)**2
      CL1=0.75*P2*(4.-5.*AI4)*ELEM(4,1)
      CL2=-0.25*P2*(3.-5.*AI4)*E
      CL3=-0.25*P2*(3.-7.*AI4)
      CL4=-0.25*P2*AI2**2*E
      CR1=-0.5*P2*(1.-1.5*AI4)*A*E12
      CR2=CR1*(1.-ES)/E
      CR3=-CR1/ES
      CR4=0.25*A*AI4*P2*E12
      CI1=0.75*AI2*AI3*P2
      CI2=E*CI1
      CI3=CI2/3.
      CO1=-1.5*P2*AI2*E
      CO2=0.75*AI2*P2
      CO3=CO2*E
      CO4=CO3/3.
      K=0
      SID=4.5833510/24.0+1.0027379093D0*(T11-43477.0D0)
   10 IF(SID-0.5) 12,11
   11 SID=SID-1.0D0
      GO TO 10
   12 IF(ANOM(1)-0.5D0) 131,13
```

```
 13 ANOM(1)=ANOM(1)-1.0D0
    GO TO 12
131 IF(ANOM(1)+0.5D0) 132,14
132 ANOM(1)=ANOM(1)+1.0D0
    GO TO 131
 14 SI=SID*PAI
    SL=-78.24+0.985647*(T11+TI2/60./48.-42780.0)
    SM=SL+77.49
    SL=SL/FAC
    SM=SM/FAC
    SL=SL+0.03344*SIN(SM)
    XS=COS(SL)
    YS=COS(23.442/FAC)*SIN(SL)
    ZS=SIN(23.442/FAC)*SIN(SL)
    C22=2.3E-06*ANOM(2)/(A*E12)**2
    S22=-1.33E-06*ANOM(2)/(A*E12)**2
    C31=2.03E-06*ANOM(2)/(A*E12)**3
    C41=0.53E-06*ANOM(2)/(A*E12)**4
    S41=0.75*C41
    AL=SI-ELEM(2,1)/FAC+TI2*PAI/60./48.
    C2A=COS(2.0*AL)
    S2A=SIN(2.0*AL)
    CC=C22*S2A+S22*C2A
    SS=C22*C2A-S22*S2A
    CA=COS(AL)
    SA=SIN(AL)
    CC4=S41*CA+C41*SA
    SS4=S41*SA-C41*CA
    CS=AI4*(1.0+3.0*AI2)-0.8*(1.0+AI2)
    SC=AI4*(1.0-3.0*AI2)-0.8*(1.0-AI2)
    AL=AL-ELEM(1,1)/FAC
    AL1=AL+2.0*ELEM(1,1)/FAC
    SAL=SIN(AL)
    CAL=COS(AL)
    SAL1=SIN(AL1)
    CAL1=COS(AL1)
    ELEM(2,1)=ELEM(2,1)+FAC*(CC*AI2+SS4*(4.-29.*AI4+28.*
   1AI4**2)/AI3)
    ELEM(3,1)=ELEM(3,1)+FAC*(SS*AI3+CC4*AI2*(4.-7.*AI4))
    E=E+E12*C31*(CAL*CS-CAL1*SC)
    ELEM(1,1)=ELEM(1,1)+CC*(1.5*AI4-AI2**2)*FAC
   1+C31*(CS*SAL+SC*SAL1)*FAC/E
   2-SS4*FAC*(4.-69.*AI4+98.*AI4**2)*AI2/AI3
    ANOM(1)=ANOM(1)+ES*(1.5*CC*AI4-C31*(CS*SAL+SC*SAL1)/E)/PAI
    EN=ANOM(2)*TI
    ELEM(1,2)=ELEM(1,2)*TI
    ELEM(2,2)=ELEM(2,2)*TI
    SI1=1.0027379*PAI*TI
    IF(ID.NE.6589) GO TO 15
    ELEM(1,1)=ELEM(1,1)-0.38E-04*FAC*DSIN(ANOM(1)*PAI
   1-11.*ELEM(1,1)/FAC-12.*AL)
 15 EL(1)=ELEM(1,1)/FAC
    EL(5)=ANOM(1)*PAI
    EL(3)=EL(5)
```

```
   16 EL(4)=EL(5)+E*SIN(EL(3))
      EE=ABS(EL(4)-EL(3))
      IF(EE-0.00001) 18,17
   17 EL(3)=EL(4)
      GO TO 16
   18 X=A*(COS(EL(4))-E)
      Y=A*SQRT(1.0-E**2)*SIN(EL(4))
      Z=ATAN(Y/X)
      R=SQRT(X**2+Y**2)
      IF(X) 19,20
   19 Z=Z+PAI/2.0
   20 EE=Z-EL(5)
      IF(ABS(EE)-1.0) 24,21
   21 IF(EE) 22,23
   22 EE=EE+PAI
      GO TO 24
   23 EE=EE-PAI
   24 EL(4)=Z+EL(1)
      EE=EE/E
      SF=SIN(Z)
      CF=COS(Z)
      AL=2.0*EL(4)
      CF2=COS(AL)
      SF2=SIN(AL)
      AL=AL-Z
      CF1=COS(AL)
      SF1=SIN(AL)
      AL=AL+2.0*Z
      CF3=COS(AL)
      SF3=SIN(AL)
      EL(4)=EL(4)+CL1*(EE+SF)+CL2*SF1+CL3*SF2+CL4*SF3
      R=R+CR1+CR2*CF+CR3*R/A
      R=R+CR4*CF2
      EL(2)=ELEM(2,1)/FAC+CO1*(EE+SF)+CO2*SF2+CO3*SF1+CO4*SF3
      EL(3)=ELEM(3,1)/FAC+CI1*CF2+CI2*CF1+CI3*CF3
      O2=COS(EL(4))
      O3=SIN(EL(4))
      Q2=COS(EL(2))
      Q3=SIN(EL(2))
      AI3=SIN(EL(3))
      AI2=COS(EL(3))
      X=R*(O2*Q2-O3*Q3*AI2)
      Y=R*(O2*Q3+O3*Q2*AI2)
      Z=R*O3*AI3
      S2=COS(SI)
      S3=SIN(SI)
      X1=S2*X+S3*Y+0.613098
      Y1=S2*Y-S3*X-0.529362
      Z1=Z-0.584693
      X2=Y1*AK8-X1*AK9
      Y2=Z1*AI8-(X1*AK8+Y1*AK9)*AI9
      Z2=Z1*AI9+(X1*AK8+Y1*AK9)*AI8
A8=SQRT(X2**2+Y2**2)
      A9=(ATAN(Z2/A8)+0.00029*A8/Z2)*FAC
```

```
      D=6378.16*SQRT(X1**2+Y1**2+Z1**2)
      XS1=S2*XS+S3*YS
      YS1=S2*YS-S3*XS
      DIR=(XS1*X1+YS1*Y1+ZS*Z1)/(D/6378.16)
      SS=XS1*(X1-0.613098)+YS1*(Y1+0.529362)+ZS*Z
      SS=SQRT(R**2-SS**2)-1.0
      D=2.*D/299.7925
      A8=ATAN(X2/Y2)*FAC
      IF(Y2) 25,26
   25 A8=A8+180.
   26 T4=T4+TI1
  265 IF(T4-60.) 30,27
   27 T4=T4-60.
      T3=T3+1.
   28 IF(T3-60.) 265,29
   29 T3=T3-60.
      T2=T2+1.0
      GO TO 265
   30 WRITE(6,31) T2,T3,T4,A9,A8,D,DIR,SS
   31 FORMAT(3F4.0,3F8.3,2F7.3)
      IF(K-II) 32,33
   32 K=K+1
      ANOM(1)=ANOM(1)+EN
      ELEM(1,1)=ELEM(1,1)+ELEM(1,2)
      ELEM(2,1)=ELEM(2,1)+ELEM(2,2)
      SI=SI+SI1
      GO TO 15
   33 STOP
      END
      END$
```

HP-Based Ranging System Software for Graz-Lustbuhel

by

G. Kirchner and P. Pesec
Graz, Austria

ABSTRACT

The paper describes the software for the Austrian Laser ranging system being under development at present. The software is based on a HP-1000/40 computer system supported by a HP-Signal Measurement and Control Processor and allows for complete control of the laser emitters and all laser sub-systems as well as for guidance of the mount in closed and semi-closed loop operation. A computer link between HP-1000 and UNIVAC 1100/81 enables suitable splitting of prediction and control software. The software is written in FORTRAN, for critical time-dependent operations HP-assembler is planned. All system components except primary I/O devices are connected to the HP-1000 system via 2 HP-IB bus (IEEE 488). Although some of the programs have been tested the whole program package is still under development. Therefore, computer listings cannot be distributed at present, but can be placed at disposal only after completing all tests.

1 Introduction

End of 1978 the Austrian Science Research Council decided to support the installation of a Laser-Ranging facility at the observatory Lustbuhel near Graz, Austria. Because of the a priori financial restrictions it was not possible to order a complete system including system software, which means that the whole system integrations had to be undertaken by the Graz-group.

This led to the philosophy to minimize hardware integration in favor of an extended software integration. The obvious consequence was to look for a homogeneous set of equipments wherever possible and to urge all other firms to provide their products with compatible interfaces.

It was decided to install an HP-1000/40 computing system, which gave, moreover, the possibility to integrate already available HP-devices and to use the already existing computer link to the main computer UNIVAC 1100/81 of the Rechenzentrum Graz. All other equipments, such as mount and laser, can be controlled by subroutines residing in the standard RTE IVB system software via HP-IB bus partly supported by the extended Measurement and Control Processor HP-2240A.

Figure 1 shows the system configuration as it is available at the moment, a direct 2400 baud connection between HP-1000 and UNIVAC is planned for the near future.

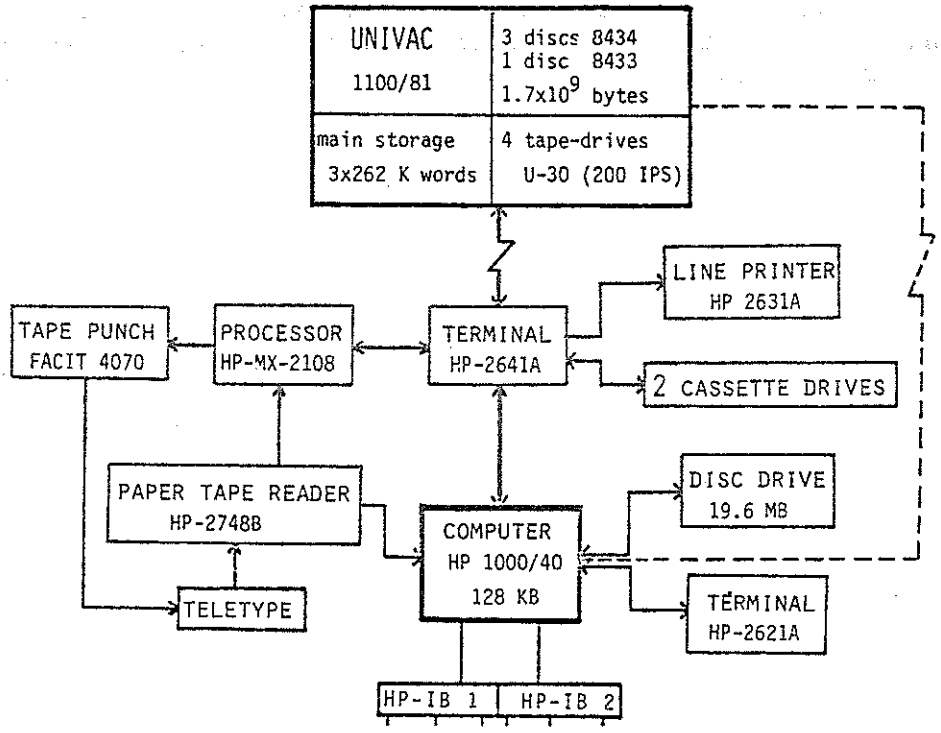


Figure 1:

Part of the programs and subroutines described in the next chapters have been tested, some programs are in development. Actually it was planned to present

a well-proved software package. However, the delayed delivery of the mount and shortage of manpower also delayed software developments.

2 Data Flow

As already indicated in Figure 1 the main advantage of the system configuration is the close connection of the HP-1000 with the UNIVAC via an internal telephone line. This gives the possibility to divide the whole software into two parts which can be operated from a common terminal. A separate system console is used for the actual laser observations.

According to Figure 1 the general data flow starts at the teleprinter. Incoming orbital data are converted by HP MX-1508 (program DECLS) into ASCII and stored on a cassette of HP-2641A. After transmission to UNIVAC satellite positions are calculated (program AIMLASER) and converted to cubic spline-functions. These functions are transmitted back onto terminal cassettes where they are at disposal for direct input into the HP-1000 system. These procedures can be run some days before the actual observations, but in spite of the extensive AIMLASER-run they can be usually run also shortly before the observations by special agreement reducing work in case of uncertain weather predictions. The general data flow for the pre-observational phase is shown in Figure 2.

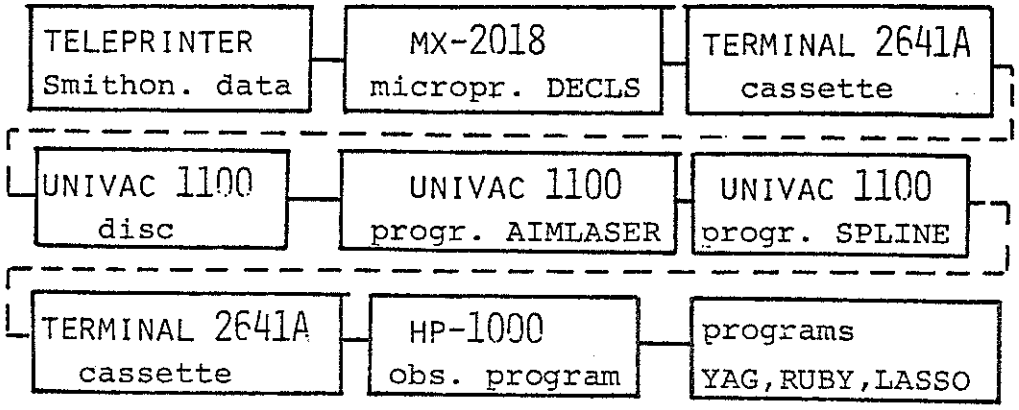


Figure 2:

3 Pre-Observational Phase

3.1 Program DECLS

Program DECLS is written in object code for an HP MX-2108 (16 KB) since no operating system is available for the HP MX-2108. It contains the driver subroutines READ (read in from tape reader), OUTPUT (output to terminal via asynchronous interface) and the subroutine DECODE, which converts 5-channel teleprinter tape to ASCII code. As a data block a decode-matrix is included. ASCII data are stored on a data cartridge using the edit mode of the terminal.

3.2 Program AIMLASER

Program AIMLASER has been placed at our disposal by Delft University of Technology, Dept. of Aerospace Engineering /1/. This program was operating on an IBM 370/158. It has been modified for the UNIVAC 1100 in January 1981. The main changes concerned the I/O operations and those parts of the subroutines MYORB and NMYORB, where free field Smithsonian data are read in (different internal representation of hollerith constants in both machines).

Some features were added in order to use the above-described data-flow scheme: Formatted data input can be optionally changed to free field input which is very convenient for execution in the demand mode. In addition to the primary output 1 second satellite positions (azimuth, elevation, range) are stored on a separate file for later use.

3.3 Program SPLINE

Program SPLINE performs a cubic spline-fit on the equally spaced satellite positions (azimuth, elevation, range). In order to avoid the inversion of large matrices (the dimension of normal equation depends on the number of supporting points used and may be up to 500) the determinant A and the co-factors $A^{-1}_{i,j}$ have been approximated analytically:

$$A^{-1}_{i,j} = (-1)^{i+j}.2^{i-j-1}.Q_i(1/2).Q_{n-j}(1/2)/\det A \qquad 0 < i \le j \le n$$

$$\simeq (-1)^{i+j}/2\sqrt{3}.\left(b^{i-j} - b^{-i-j-2} - b^{i+j-2n-2} \right) \qquad \text{for } n > 20$$

$$A^{-1}_{o,j} = (-1)^{j}.2^{-j-1}.Q_{n-j}(1/2)/\det A \qquad 0 < j \le n$$

$$\simeq (-1)^{j}.b^{-j-1} \qquad \text{for } n > 20$$

$$\det A = 2.Q_n(1/2) - 1/4.Q_{n-1}(1/2)$$

$$\simeq 1/\sqrt{3}.2^{-n-2}.b^{n+2} \qquad \text{for } n > 20$$

where

$$Q_n = 2.D_{n-1} + 1/4.D_{n-2} \quad , \quad D_n = 1/\sqrt{3}.2^{-n-1}.(b^{n+1} - b^{-n-1}) \quad , \quad b = 2+\sqrt{3} \; .$$

This approximation introduces some small oscillations at the starting and end points of the fit, gives however the advantage to choose the interval between the supporting points freely as a function of the highest elevation of the pass. For highest elevations up to 60 degrees 10 sec-spaced supporting points are sufficient, with 4 sec spacing passes up to 85 degrees can be fitted in azimuth without affecting the desired pointing accuracy for worst case of low flying satellites.

Subroutine COFF recomputes satellite positions for 1 sec intervals from the spline fit and compares the results with the original positions lying

inside the spline interval. The largest residual of each interval is printed out for azimuth, elevation and range and gives the indication whether the interval for supporting points has been chosen properly.

Spline coefficients are usually calculated for elevations higher than 30 degrees in order to provide the possibility to encounter satellite delays and to correct them prior to the actual laser observations starting at 45 degrees.

Spline coefficients are outputted on a permanent file and transmitted back onto a cartridge of the HP-2641A terminal at the observatory.

4 Observational Phase

As already indicated the design of the complete observation software has not been finished up to now. Part of the subprograms and subroutines are finished and tested, another part is under development. Completely new subroutines may be introduced within the next months.

In Figure 3 all programs and subroutines available or under development are listed. It seems to be too early to state flow-chart diagrams at the present stage.

MAIN PROGRAMS: YAG , RUBY , LASSO
SUB PROGRAMS: AIROK,CAL,MINIT,TRACK,SEARCH,CLASS
SUBROUTINES : RSPL,POS,MMOD,MOUT,MINP,RD309,RD28,RD70A

Figure 3:

4.1 Main Programs

In principle 3 operating modes are possible with the laser ranging facility at Graz. For these modes the main programs YAG, RUBY, LASSO are presently under development.

- Program YAG controls observations carried out with the Nd-YAG laser at a repetition rate of about 5 Hz.
- Program RUBY controls observations carried out with the Ruby laser at a repetition rate of 0.25 Hz.
- Program LASSO controls observations to geo-stationary satellites.

All 3 main programs have only organizing character and are used to schedule subprograms and subroutines. Program RUBY is nearly finished the programs YAG and LASSO are in the initial stage.

4.2 Sub Programs

4.2.1 Program MINIT

Program MINIT initializes the mount. It takes care that the mount is prepositioned in azimuth and elevation to a defined unambiguous position (the azimuth range is 540 degrees). It further gives advice to the operator how to operate the front panel and in which sequence to enable the different switches.

4.2.2 Program AIROK

Program AIROK tests the operation of the aircraft detection system. It moves the mount sequentially to a series of terrestrial and/or celestial targets and checks the ability of the system to interrupt laser measurements. As program MINIT program AIROK makes extensive use of the mount subroutines MMOD, MOUT, MINP. The whole system is configured in such a way that without positive reply the main program stops unconditioned.

4.2.3 Program CAL

Similar to program AIROK program CAL performs the necessary calibration tests. It checks the calibration of the mount by a terrestrial reference point, which can be visual determined with the ISIT camera. It, furthermore, determines the calibration values for the laser ranges. Program CAL is presently under development.

4.2.4 Program TRACK

Program TRACK guides the mount. First it reads spline data from the terminal cassette (RSPL) and stores the data in a labeled common block. In order to avoid the ambiguity in azimuth it moves the mount to the azimuth of closest approach in two steps. After that the starting position (usually 30 degrees elevation) and the starting epoch is calculated (POS) and the mount is moved to that position. The operator has now the possibility to enter initial corrections determined from prior passes. At a convenient time before the starting epoch he starts tracking by pressing RETURN. The mount is changed from position mode to rate mode and the time is checked continuously for the starting epoch t_o. At t_o the first computed rate is transmitted to the mount and the next position and rate is calculated (usually in 1 sec steps). At the next second the mount position is read out, compared with the theoretical position and the corresponding rate- correction computed. The corrected rate is then immediately outputted to the mount. This game is repeated until elevation reaches a predefined value, where the mode is again changed to position mode.

4.2.5 Program SEARCH

Program SEARCH is an extension of program TRACK. It allows for scanning a time interval of $\pm$ 10 sec referred to t_o. This is done by increasing or

decreasing the a priori rate in such a way that the satellite image crosses the monitor screen in about 5 sec, which enables the operator to set an interrupt when the image is near the center. At the interrupt the mount position is read out and compared with the computed position. The offset is calculated and transferred to rate change, which is efficient within a precalculated time interval. In the moment SEARCH is acting only at the beginning of the pass between elevations of 30 to 45 degrees. The possibility to correct passes during the laser observations will be included at a later time.

4.2.6 Program CLASS

Program CLASS is a demonstration program. It shows the use of CLASS-I/O. CLASS-I/O enables input of parameters via terminal into a scheduled and running program. The program picks up and uses these parameters, but there is no stop or waiting for the parameter input operation. The program continues with old parameters if there is no input, it continues with new parameters if there is any input.

The demonstration of the continuation feature of programs is done by resetting the HP5370 counter every second. Demonstration of input of the new parameters is done by asking the terminal for trigger inputs and displaying these trigger levels on the HP-5370 counter.

4.3 Subroutines

4.3.1 Subroutines RSPL, POS

These subroutines read in spline function data from terminal cassette into a labeled common block and calculate satellite positions and rates for arbitrary epochs. The underlying mathematical scheme is the standard procedure of cubic spline functions.

4.3.2 Subroutines MMOD, MOUT, MINP

MMOD is used for changing the mode of the mount (off-mode, position-mode, rate-mode). For these commands secondary addressing is used.

MINP converts computed data (azimuth, elevation) to an ASCII string and puts them out to the mount.

MOUT reads in the position information from the mount and converts it from ASCII string to a variable.

RD309 reads HP309 clock, converts ASCII string (hour, minutes, seconds) into time of day (seconds).

RD28 reads in data from universal counter HP-5328 and returns the data to

the main program.

RD70A reads in data from universal counter HP-5370A and returns the data to the main program.

5 Summary

All programs and subroutines addressed above are available or presently under development. Since it was not possible up to now to check the whole program system under real conditions it is not possible in the moment to distribute parts of the programs for further use. Computer print-outs can however be presented for discussion. We hope to finish the first operable program package until end of 1981, and we are ready to distribute interesting parts of this package at that time.

An Overview of a DEC-based Satellite Ranging System

by

David W. Morrison
Drews Systems, Inc.
599 North Matilda Avenue
Suite 245
Sunnyvale, CA 94086 USA

1 THE PDP-11/45 COMPUTER AND RSX-11M

SEVERAL FEATURES OF THE DEC COMPUTER AND ITS OPERATING SYSTEM HAVE STRONGLY SHAPED THE SATELLITE RANGING SYSTEM (SRS) LOCATED IN WETTZELL, BUNDESREPUBLIK DEUTSCHLAND. THE FIRST SECTION DISCUSSES THOSE FEATURES, THE SECOND DESCRIBES SRS ITSELF, AND THE THIRD DEALS WITH THE DESIGN AND CODING METHODS THAT WERE USED IN THE CREATION OF SRS.

1.1 THE EQUIPMENT

FIGURE 1.1-1 SHOWS THE CURRENT AND ORIGINAL COMPUTER CONFIGURATIONS IN WETTZELL.

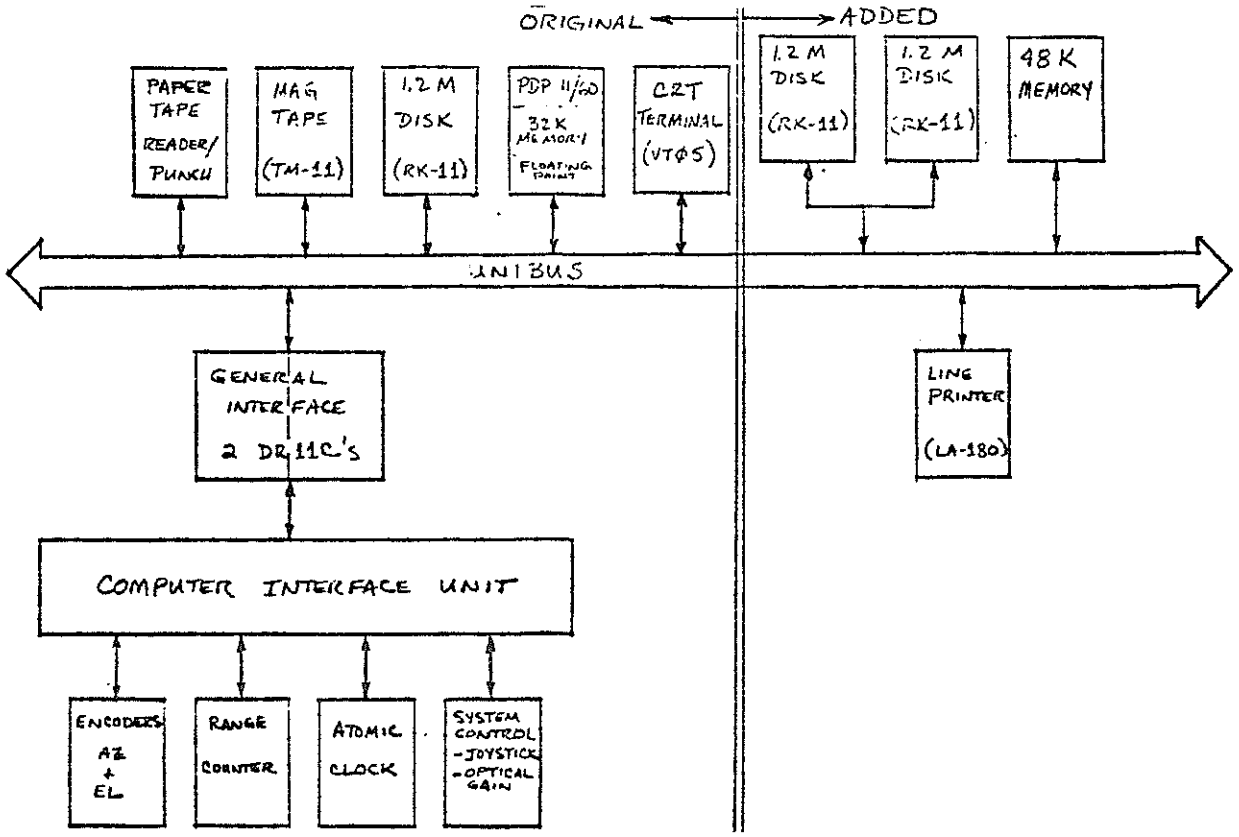


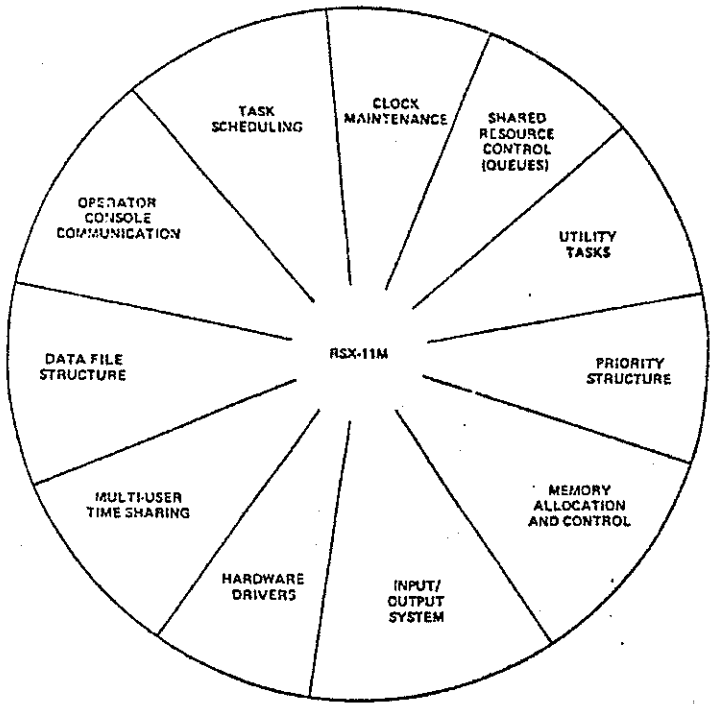
Figure 1.1-1

THE FACILITY HAS BEEN SIGNIFICANTLY EXPANDED SINCE THE ORIGINAL SRS WAS WRITTEN PARTICULARLY BY ADDING MEMORY, DISK STORAGE, AND A FAST LINE PRINTER.

1.2 RSX-11M--THE OPERATING SYSTEM

FIGURE 1.2-1 DEPICTS THE MAJOR FEATURES OF THE RSX-11M OPERATING SYSTEM.

MOST PLAYED A ROLE IN SRS DEVELOPMENT. SEVERAL ARE DISCUSSED IN SUBSEQUENT ARTICLES.



RSX-11M OPERATING SYSTEM

NOTE:
RSX PROVIDES THE FUNCTIONS SHOWN IN THE DIAGRAM. APPLICATION PROGRAMS MAKE "EXECUTIVE" CALLS TO COMMUNICATE WITH RSX AND INVOKE ONE OR MORE OF THESE FUNCTIONS.

Figure 1.2-1

1.3 PROGRAM DEVELOPMENT UTILITIES

FIGURE 1.3-1 SUMMARIZES THE RELATIONSHIPS BETWEEN THE UTILITY PROGRAMS, COMPILER, AND ASSEMBLER WHICH ARE USED TO CREATE SOURCE TEXT, OBJECT FILES, LIBRARIES, AND EXECUTABLE PROGRAMS.

ANY FILE CAN BE COPIED, DELETED, MERGED OR LISTED USING PIP (PERIPHERAL INTERCHANGE PROGRAM.) PIP IS FOUND IN SEVERAL DEC SYSTEMS. IT IS NOT PART OF THE I/O SYSTEM (CALLED FILE CONTROL SERVICES) THAT IS AVAILABLE TO INDIVIDUAL PROGRAMS. IT GENERALLY PROVIDES WAYS FOR MANIPULATING FILES AS A WHOLE AS OPPOSED TO MAKING CHANGES WITHIN THEM.

- EDI AND EDT ARE TWO TEXT EDITORS THAT PROVIDE THE NORMAL MEANS OF SOURCE ENTRY AND MAINTENANCE.

- F4P IS A SEPARATELY AVAILABLE FORTRAN COMPILER WHICH GENERATES EFFICIENT, INLINE CODE.

- MAC IS THE ASSEMBLER FOR THE DEC ASSEMBLY LANGUAGE CALLED MACRO.

- TKB BUILDS EXECUTABLE PROGRAM IMAGES (CALLED TASKS) FROM RELOCATABLE

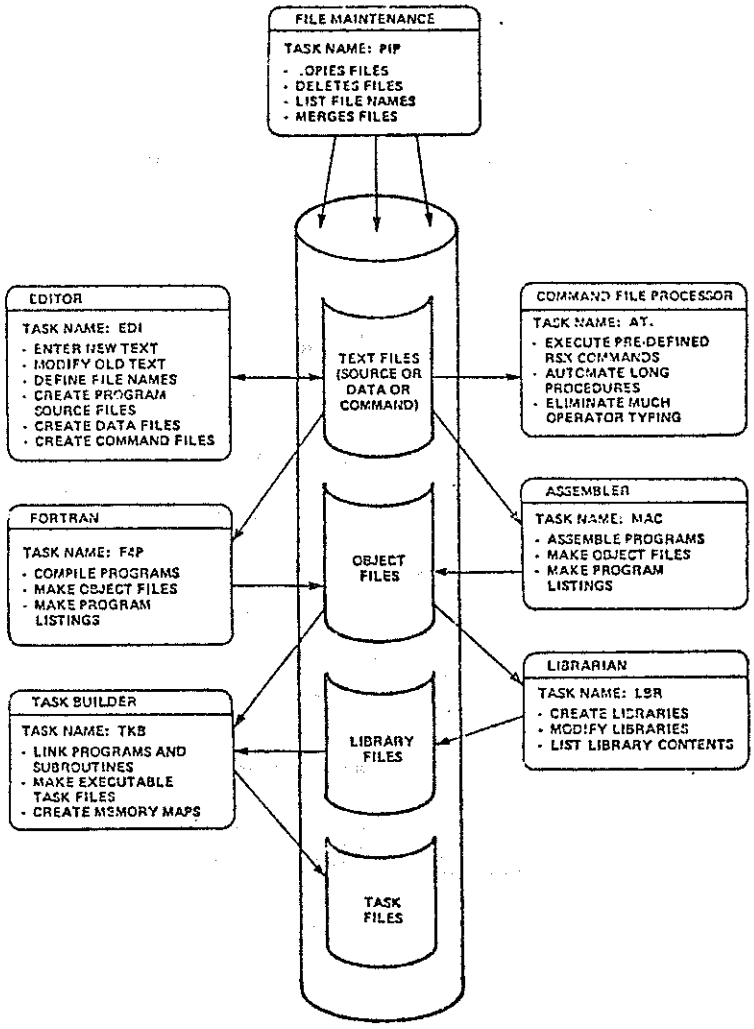


Figure 1.3-1

OBJECT FILES AND OBJECT MODULE LIBRARIES.

- LBR IS THE LIBRARIAN UTILITY USED TO CREATE AND MAINTAIN OBJECT MODULE LIBRARIES.

- AT. IS A COMMAND LANGUAGE UTILITY THAT ALLOWS LENGTHY CONTROL SEQUENCES TO BE PRE-DEFINED IN FILES. THE RSX KEYBOARD MONITOR ACCEPTS SUCH FILES AS DO ALL THE MAJOR UTILITY PROGRAMS. MUCH OPERATOR TYPING AND TIME IS SAVED THROUGH THE USE OF AT. SYSTEM-WIDE PROCEDURAL STANDARDS CAN ALSO BE MORE EASILY IMPOSED.

1.4 EVENT FLAGS

EVENT FLAGS ARE ONE OF THE MAIN SYNCHRONIZING AGENTS IN THE RSX SYSTEM. THEY CAN BE ASSOCIATED WITH AN ASYNCHRONOUS EVENT (FOR EXAMPLE, THE END OF A TIME INTERVAL.) WHEN THE EVENT OCCURS, THE FLAG'S STATUS GENERALLY CHANGES FROM "CLEAR" TO "SET". EVENT FLAGS CAN ALSO BE SET AND CLEARED BY TASKS AS GATING INDICATORS. EACH TASK HAS A LOCAL SET OF EVENT FLAGS WHILE ANOTHER

GROUP IS COMMON TO ALL TASKS.

EVENT FLAGS ARE MAINTAINED BY THE RSX EXECUTIVE VIA EXECUTIVE REQUEST CALLS FROM THE APPLICATION TASKS. CAPABILITIES INCLUDE SETTING, CLEARING, AND TESTING THE STATE OF AN ARBITRARY EVENT FLAG; WAITING FOR AN EVENT FLAG OR FLAGS TO BE SET; AND ASSOCIATING AN ARBITRARY EVENT FLAG WITH CERTAIN EVENTS. FIGURE 1.4-1 SUMMARIZES THE CAPABILITIES.

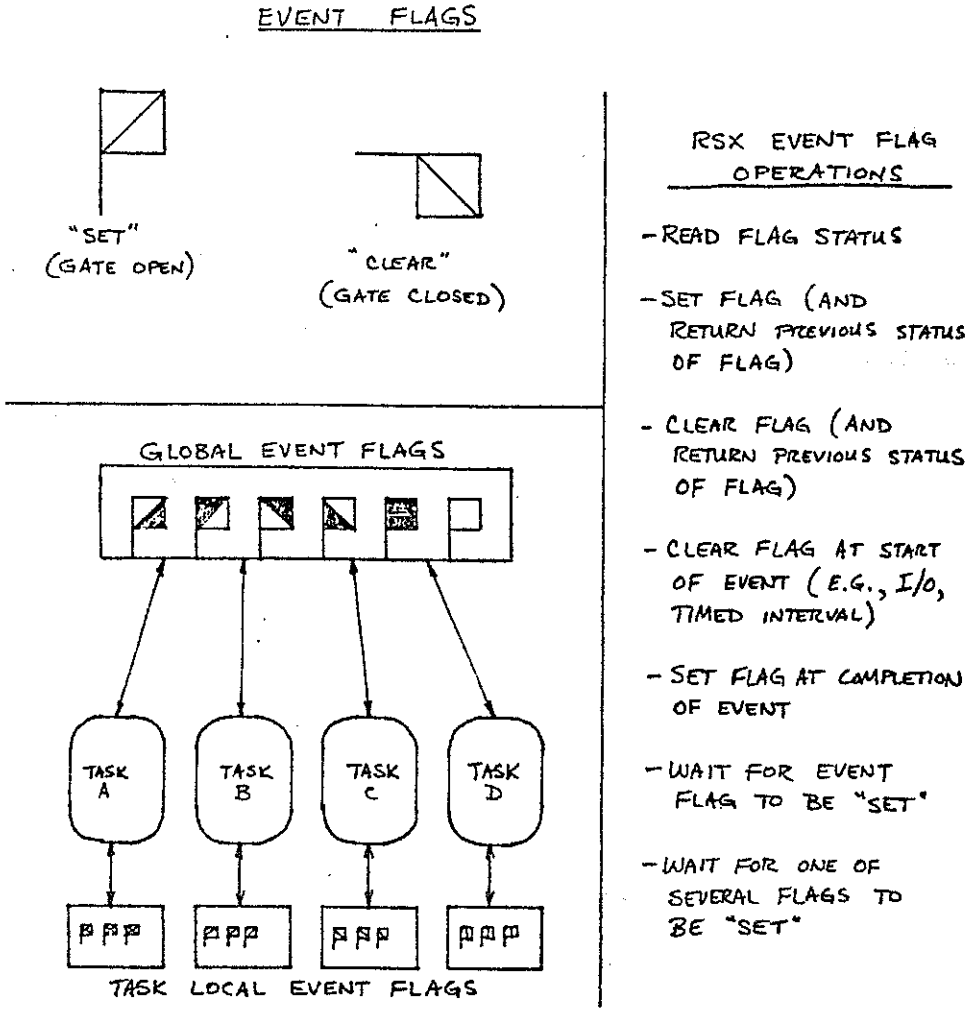


Figure 1.4-1

EVENT FLAGS ARE OFTEN USED TO SYNCHRONIZE INPUT/OUTPUT OPERATIONS WITH EXECUTION OF THE REQUESTING TASK. THE I/O CAN PROCEED IN PARALLEL WITH TASK OPERATION. THE EVENT FLAG IS SET BY THE RSX EXECUTIVE WHEN THE I/O COMPLETES. THE TASK CAN CHECK THE EVENT FLAG STATUS WHENEVER AND WHEREVER THE SYNCHRONIZATION MUST OCCUR. IT IS ALSO POSSIBLE TO CHECK THE FLAG'S STATUS WITHOUT SYNCHRONIZING.

A SECOND COMMON USE FOR EVENT FLAGS IS TO GATE ACCESS TO COMMON DATA BASES. USUALLY A CLEAR FLAG MEANS THE DATA IS WRITE-LOCKED WHILE SET MEANS THE DATA IS READ-WRITE ENABLED. COMPETING TASKS WAIT FOR THE EVENT FLAG TO BE SET (THE TASK USUALLY RELINQUISHES THE CPU WHILE IT WAITS), THEN TRY TO CLEAR

THE FLAG AND CHECK ITS PREVIOUS STATE. IF THE FLAG WAS PREVIOUSLY CLEAR THEN ANOTHER, HIGHER PRIORITY TASK WO' THE RACE AND LOCKED THE DATA BASE FIRST. THE LOWER PRIORITY REQUESTER MUST AGAIN WAIT FOR THE SET CONDITION AND THEN REPEAT THIS ACCESSING PROCESS.

EVENT FLAGS ARE USED BY SRS IN ALL THE WAYS MENTIONED HERE. FOR EXAMPLE, DURING THE TRACKING OPERATION EVENT FLAGS ARE USED BY

1. THE CRT TASK TO REQUEST MORE DISPLAY DATA FROM THE TRACKING TASK;
2. ALL REAL-TIME TASKS TO CHECK FOR THE END-OF-TRACK CONDITION;
3. THE CRT TASK TO SYNCHRONIZE ITS OUTPUT TO THE OPERATOR CONSOLE;
4. THE TRACKING TASK TO SYNCHRONIZE ITS READING OF PERIODIC DATA.

1.5 I/O REQUESTS AND DATA TRANSFERS

INPUT AND OUTPUT IN THE RSX SYSTEM CAN BE FULLY ASYNCHRONOUS RELATIVE TO THE REQUESTING TASK'S EXECUTION. THE PRINCIPAL I/O INITIATOR ROUTINE FOR RSX IS NAMED QIO. FIGURE 1.5-1 SHOWS THE INITIATION OF AN I/O REQUEST BEGINNING WITH THE QIO CALL IN THE APPLICATION TASK.

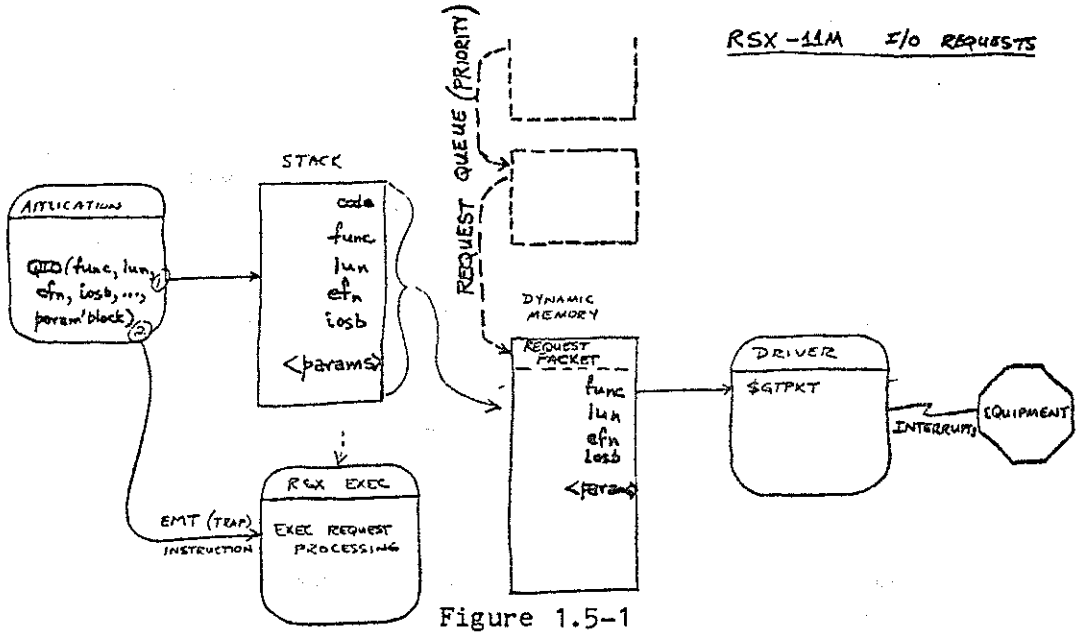


Figure 1.5-1

QIO PLACES THE CALLING PARAMETERS ON THE STACK AND EXECUTES A TRAP INSTRUCTION WHICH BEGINS RSX EXECUTIVE EXECUTION. THE EXECUTIVE PLACES THE PARAMETERS IN A QUEUE AND CALLS THE DRIVER. THE DRIVER READS THE QUEUE AND BEGINS INTERFACING TO THE APPROPRIATE EQUIPMENT.

FIGURE 1.5-2 SHOWS TRANSFERRAL OF DATA IN RESPONSE TO A QIO CALL. THE DRIVER NORMALLY WRITES DATA DIRECTLY TO (OR READS DATA DIRECTLY FROM) THE APPLICATION BUFFER ALTHOUGH IT IS ALSO POSSIBLE TO TRANSFER DATA THROUGH

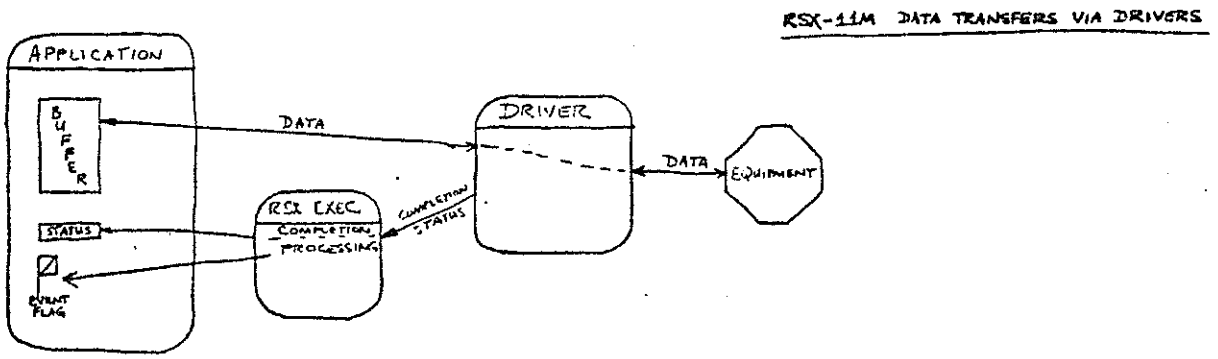


Figure 1.5-2

DYNAMIC MEMORY THUS ALLOWING THE APPLICATION TO BE NON-MEMORY RESIDENT DURING THE BUFFER-HARDWARE TRANSFER. THE RSX EXECUTIVE IS NOTIFIED BY THE DRIVER WHEN I/O COMPLETES. IT, IN TURN, SETS SYNCHRONIZING EVENT FLAGS AS REQUIRED AND MAKES THE APPLICATION TASK ELIGIBLE FOR EXECUTION.

FIGURE 1.5-3 DEPICTS THE STANDARD RSX EQUIPMENT HANDLER ORGANIZATION.

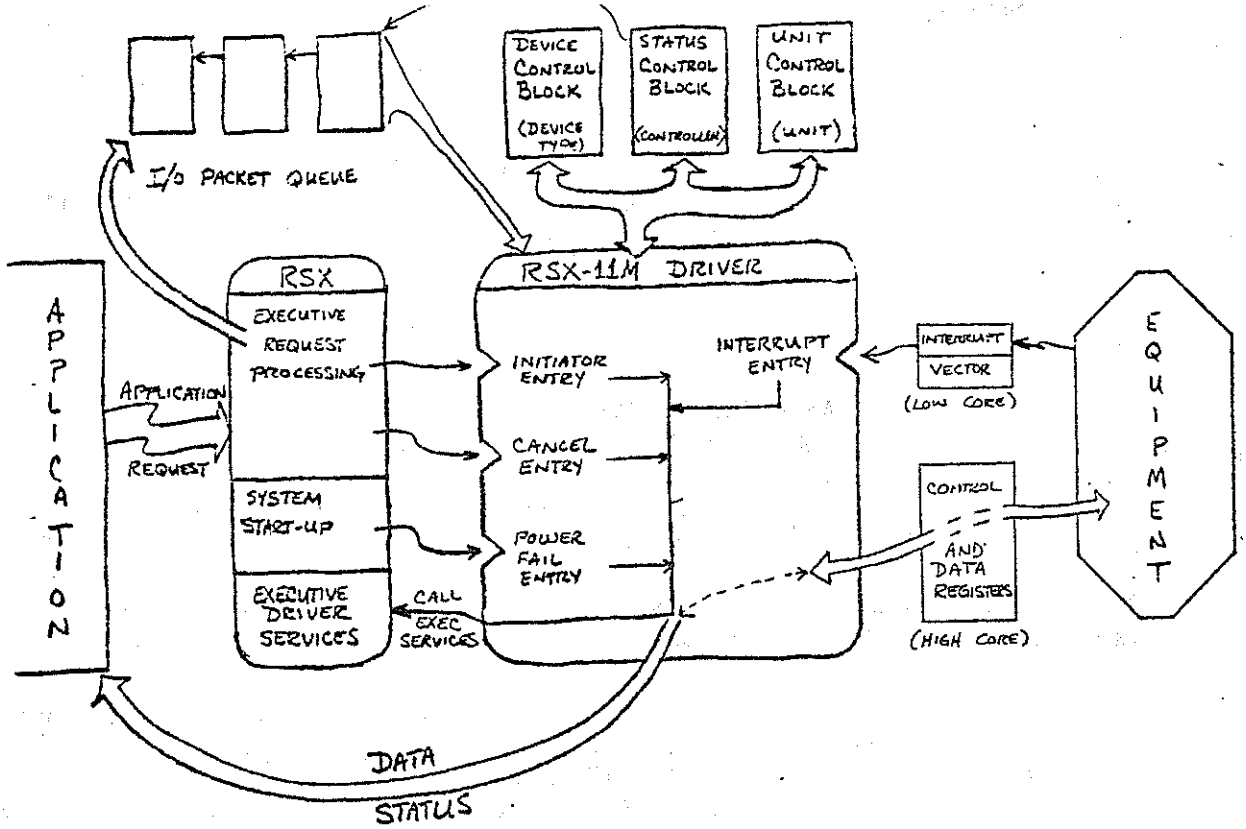


Figure 1.5-3

ESSENTIAL POINTS INCLUDE

1. THE EXECUTIVE REQUEST PROCESSING WHICH IS THE INTERFACE BETWEEN THE APPLICATION AND THE DRIVER-LEVEL SOFTWARE,

2. THE DEVICE CONTROL BLOCK (DCB), THE STATUS CONTROL BLOCK (SCB) AND THE UNIT CONTROL BLOCK (UCB) WHICH REFLECT THE STATUS OF THE HARDWARE,

3. THE I/O PACKET QUEUE WHICH CONTAINS PRIORITY-ORDERED REQUESTS FOR DRIVER ACTION,

4. THE INITIATOR ENTRY POINT WHERE THE EXECUTIVE STARTS DRIVER EXECUTION,

5. THE INTERRUPT ENTRY POINT WHERE HARDWARE INTERRUPTS ARE PROCESSED,

6. THE CONTROL AND DATA REGISTERS WHERE THE FUNCTION CODES, DATA, AND STATUSES ACTUALLY CHANGE HANDS BETWEEN COMPUTER AND EXTERNAL HARDWARE,

7. THE INTERRUPT VECTOR WHICH GUIDES TRANSFER OF COMPUTER CONTROL WHEN AN INTERRUPT OCCURS,

8. THE EXECUTIVE DRIVER SERVICES WHICH PERFORM STANDARD FUNCTIONS FOR THE DRIVER,

9. THE TRANSFERRAL OF DATA AND STATUS INFORMATION BETWEEN APPLICATION AND DRIVER.

SRS MAKES EXTENSIVE USE OF THIS I/O SYSTEM PARTICULARLY THROUGH THE COMPUTER INTERFACE UNIT DRIVER PROGRAM (CUDRV) WHICH IS THE MAIN INTERFACE ROUTINE TO THE TRACKING HARDWARE. CUDRV IS DISCUSSED FURTHER IN SECTION 2.

1.6 FILE STRUCTURE

RSX SUPPORTS A REASONABLY EXTENSIVE FILE STRUCTURE ON DISKS. THE ENTIRE DISK IS MAPPED FOR BLOCK ALLOCATION TO NAMED FILES. FILES MAY BE SEQUENTIAL OR RANDOM, FORMATTED OR BINARY, OR ANY COMBINATION. FILES ARE IDENTIFIED WITH A 1-9 CHARACTER NAME, A 3-CHARACTER TYPE, AND A VERSION NUMBER. FILES ARE STORED IN DIRECTORIES WHICH ARE WRITE LOCKED TO ALL USERS EXCEPT THOSE WHO HAVE ENTERED THE PROPER DIRECTORY NUMBER IN ANY OF SEVERAL WAYS. RSX SUPPORTS FURTHER FILE PROTECTIONS AND MULTI-USER FEATURES, BUT BECAUSE THE SRS ENVIRONMENT IS "FRIENDLY," NO USE IS MADE OF THEM.

FILES MAY BE OPENED AT ANY OF SEVERAL LOGICAL LEVELS. FORTRAN PROVIDES AN OPEN STATEMENT AT THE HIGHEST LEVEL. THE FILE CONTROL SERVICES (FCS) PACKAGE PROVIDES MORE FLEXIBILITY AT THE ASSEMBLY LANGUAGE LEVEL. AT THE LOWEST, STEP-BY-STEP LEVEL THE FILE CONTROL PRIMITIVES SUPPORT NOT ONLY FILE ACCESSES BUT FILE DIRECTORY MANIPULATIONS AS WELL.

SRS USES ALL LEVELS OF FILE ACCESS. FORTRAN IS USED FOR BOTH SEQUENTIAL AND RANDOM ACCESS WHENEVER ITS LIMITED ALTERNATIVES AND RELATIVELY SLOW SPEED CAN BE TOLERATED. FCS PROVIDES OCCASIONAL SPECIAL ACCESS (USUALLY IN THE FORM OF A FORTRAN-CALLABLE ASSEMBLY LANGUAGE ROUTINE.) FINALLY, SOME PRIMITIVES ARE USED BY THE TRACKING TASK TO ACHIEVE MAXIMUM SPEED AND PARTICULARLY TO

AVOID THE MEMORY COST OF THE FORTRAN AND FCS EXECUTION TIME ROUTINES.

1.7 FORTRAN COMPILER OBJECT CODE

THE FORTRAN-IV-PLUS COMPILER (F4P) GENERATES INLINE, MACHINE LANGUAGE CODE. MANY OTHER MINICOMPUTER FORTRANS GENERATE "THREADED" CODE; THAT IS, A SERIES OF CALLS TO EXECUTION-TIME ROUTINES FOR ALL OPERATIONS NO MATTER HOW SIMPLE. THREADED CODE ALLOWS FAST COMPILE TIMES BUT IS MUCH TOO SLOW FOR MOST REAL-TIME APPLICATIONS. THUS THE INLINE CODE OF F4P IS AN ESSENTIAL FEATURE OF RSX WHICH HAS SIGNIFICANTLY AIDED THE CODING, IMPLEMENTATION, AND MAINTENANCE OF THE SRS SYSTEM.

FIGURE 1.7-1 IS A PARTICULARLY IMPRESSIVE EXAMPLE OF F4P'S CODE QUALITY, IN THIS CASE INVOLVING TRIPLE SUBSCRIPTING.

NOTE:
given: INTEGER X(A,B,C)
then: $X(I,j,k) = locX + 2[A(j+Bk) - A(1+B) - (I-1)]$

```
      INTEGER TAPEBF(36,8,2)
C
      TAPEBF(PSTATS,NXSEPT,CRNTPT)=
     *      IAND(TAPEBF(PSTATS,NXSEPT,CRNTPT),'77777)
      CALL UPDATN(TAPEBF(PSTATS,NXSEPT,CRNTPT),
     !            TAPEBF(PCVPWR,NXSEPT,CRNTPT)
```

```
MOV     CRNTPT,R0
ASH     #3,R0          ; Bk ... NOTE USE OF SHIFT TO MULTIPLY - BY POWER OF 2.
ADD     NXSEPT,R0      ; j+Bk
MOV     R0,R1
MUL     #110,R1        ; 2[A(j+Bk)]
BIC     #100000,TAPEBF-1210(R1) ; R1 = 2[A(1+B)+I-1]
                       ; 0043
MOV     R1,$IDATA+12
ADD     #TAPEBF-1210,$IDATA+12 ; loc(ARG1)
ADD     #TAPEBF-1140,R1
MOV     R1,$IDATA+14   ; loc(ARG2)
MOV     #$IDATA+10,R5
JSR     PC,UPDATN      ; CALL UPDATN (---)
```

Figure 1.7-1

1.8 OVERLAYS

THE RSX TASK BUILDER AND RUN-TIME LOADER SUPPORT PROGRAM OVERLAY STRUCTURES. AT THE EXPENSE OF EXECUTION SPEED, OVERLAYS ALLOW VERY LARGE

PROGRAMS TO BE EXECUTED IN RELATIVELY SMALL MEMORY AREAS WITHOUT THE NEED FOR BREAKING THE APPLICATION INTO MANY SMALL, INTERACTING TASKS. BECAUSE SRS WAS ORIGINALLY WRITTEN FOR AN APPLICATION MEMORY SPACE OF ONLY 18,000 WORDS, THIS FEATURE WAS EXTENSIVELY USED IN SEVERAL TASKS.

RSX SUPPORTS TWO METHODS OF OVERLAY LOADING--MANUAL AND AUTOMATIC. THE FIRST IS FASTER BUT REQUIRES EXPLICIT CODE IN THE PROGRAM TO LOAD EACH OVERLAY. THE SECOND IS SLOWER BUT REQUIRES NO KNOWLEDGE OF THE OVERLAY STRUCTURES IN THE PROGRAMS. THE FIRST LOADING METHOD IS USED BY THE SRS TRACKING TASKS, THE SECOND BY MOST OTHER SRS TASKS WHICH USE OVERLAYS.

1.9 MEMORY MAPPING REGISTERS

THE DEC HARDWARE USES EIGHT MEMORY MAPPING REGISTERS TO MAP UP TO 32K WORDS OF LOGICAL ADDRESS SPACE ONTO ACTUAL PHYSICAL MEMORY (FIGURE 1.9-1.)

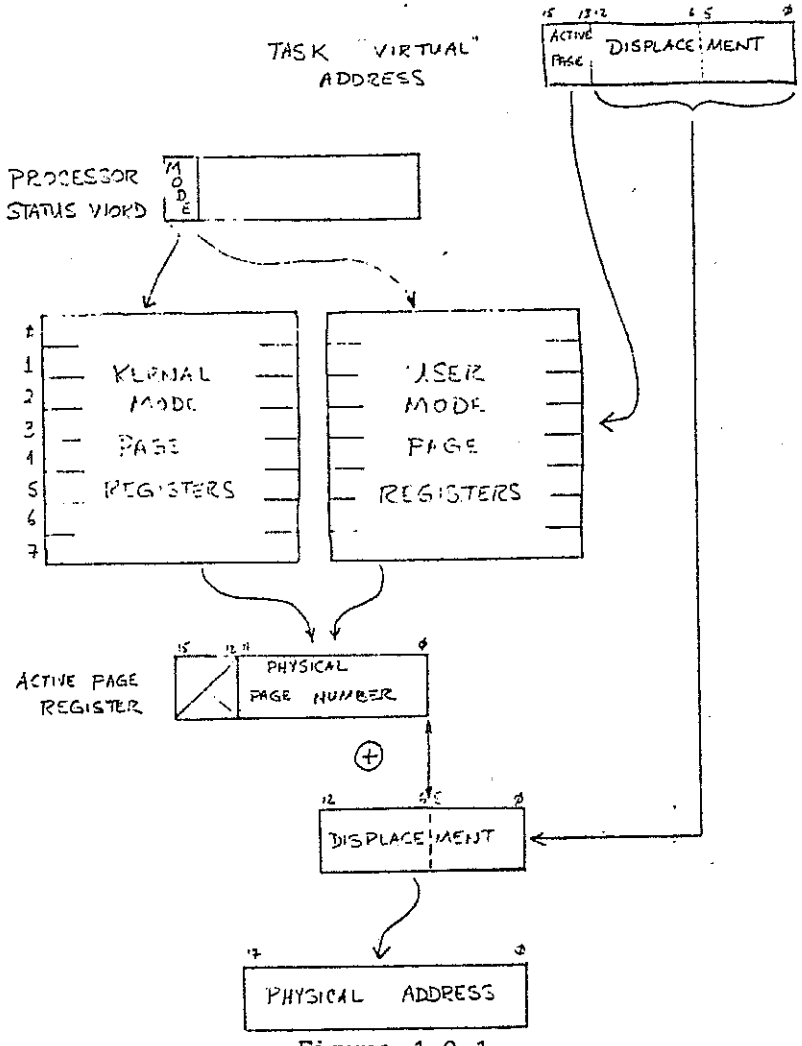


Figure 1.9-1

THIS MEANS THAT LOGICAL MEMORY CAN ONLY BE ALLOCATED IN 4K-WORD BLOCKS. THIS RELATIVELY LARGE INCREMENT IS CONFINING IN SOME CASES AND WASTEFUL OF LOGICAL MEMORY SPACE IN OTHERS.

EXAMPLES: (1) THE GLOBAL DATA AREA IS ABOUT 2.5K WORDS LONG WHICH MEANS THAT, BECAUSE MEMORY CAN ONLY BE ALLOCATED IN 4K BLOCKS, ABOUT 1.5K WORDS OF LOGICAL ADDRESS SPACE IS LOST IN EVERY TASK WHICH LINKS TO THIS AREA; (2) FOR THIS REASON THE DIAGNOSTIC MODE TASK MUST BE BUILT IN A SPECIAL WAY TO AVOID LINKING TO THE GLOBAL AREA.

2 SRS ORGANIZATION

THE SRS SYSTEM IS BROKEN INTO SEVERAL RELATED UNITS CALLED "MODES". EACH MODE IS IMPLEMENTED AS A SEPARATE PROGRAM CALLED A "TASK". FIGURE 2.0-1 DEPICTS THE RELATIONSHIPS BETWEEN ALL THE TASKS.

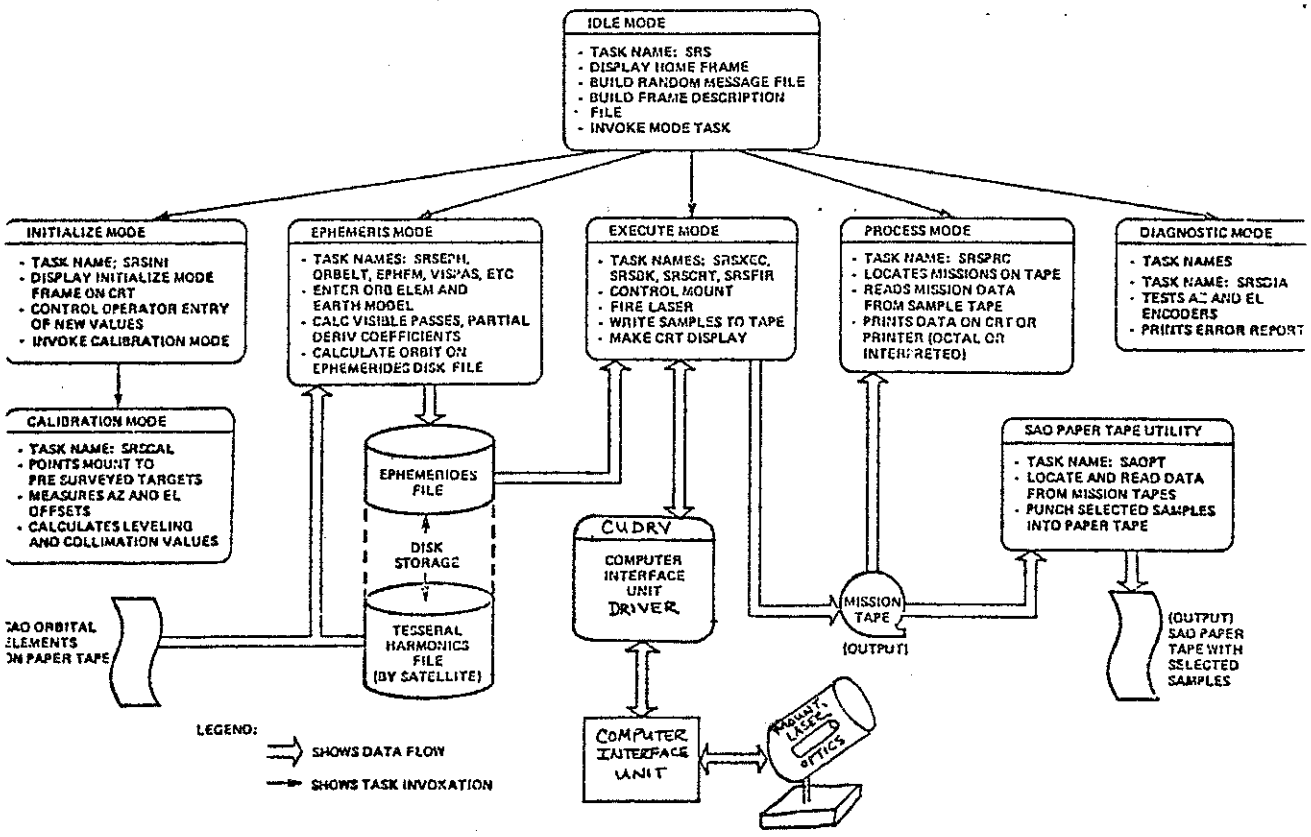


Figure 2.0-1

FURTHER ARTICLES IN THIS SECTION SUMMARIZE EACH MODE PLUS RELATED UTILITIES AND THE COMPUTER INTERFACE UNIT DRIVER PROGRAM.

2.1 IDLE MODE

- PURPOSE:

1. ALLOW THE OPERATOR TO SELECT A MAJOR OPERATING MODE,

2. WHEN NECESSARY, READ THE MESSAGE FILE AND CREATE THE RANDOM MESSAGE FILE AND THE FR..ME DESCRIPTOR FILE.

- COMMENTS:

* THE IDLE MODE TASK--NAMED SRS--IS CALLED WHEN THE OPERATOR TYPES "RUN SRS". WHENEVER THIS CALL IS MADE, A MAJOR SUBFUNCTION OF THE IDLE MODE NAMED THE "MESSAGE HANDLER" IS CALLED. THE MESSAGE HANDLER IS RESPONSIBLE FOR PREPARING THE MESSAGES AND DISPLAYS WHICH WILL BE USED DURING EXECUTION OF THE SRS SYSTEM.

* THE MESSAGE HANDLER BEGINS BY ATTEMPTING TO OPEN A FILE NAMED "FRAMEF.SRS". THIS FILE CONTAINS DESCRIPTIVE INFORMATION FOR THE VARIOUS CRT FRAMES IN SRS. IF THE FILE IS SUCCESSFULLY OPENED, THE MESSAGE HANDLER HAS NO MORE RESPONSIBILITIES. IT CLOSES THE FILE AND RETURNS TO THE IDLE MODE. IF, HOWEVER, "FRAMEF.SRS" IS NOT SUCCESSFULLY OPENED--THAT IS, NO FILE BY THAT NAME EXISTED ON THE DISK--AN ASCII TEXT FILE CALLED THE "MESSAGE FILE" IS READ AND TWO NEW FILES ARE CREATED:

  1. A RANDOM FILE CONTAINING THE MESSAGES WHICH CAN BE DISPLAYED DURING SRS EXECUTION (RANDOM.SRS);

  2. A FILE CONTAINING DESCRIPTIONS OF THE DISPLAY FRAMES IN THE SYSTEM (FRAMEF.SRS.)

* THIS PROCESSING WILL NOT BE REPEATED ON SUBSEQUENT EXECUTIONS OF SRS UNLESS FRAMEF.SRS IS DELETED OR DESTROYED (SEE FIGURE 2.1-1.)

* AFTER THE ABOVE PROCESSING HAS BEEN COMPLETED, THE HOME FRAME (FIGURE 2.1-2) IS DISPLAYED TO ALLOW THE OPERATOR TO SELECT A SPECIFIC PORTION OF THE SYSTEM FOR FURTHER PROCESSING.

* THE FRAME DATA CREATED BY THE MESSAGE HANDLER IS ACCESSED THROUGHOUT SRS VIA A COLLECTION OF ROUTINES CALLED THE "DISPLAY PACKAGE." THE DISPLAY PACKAGE CAN BE INCLUDED IN ANY TASK WHICH MUST CREATE AND INTERACT WITH A STANDARD SRS DISPLAY FRAME.

* THE MESSAGE HANDLER AND DISPLAY PACKAGE PROVIDE SEVERAL ADVANTAGES FOR SRS. THEY SERVE AS A STRONG IMPETUS FOR STANDARDIZING ALL DISPLAYS. THIS IN TURN PROVIDES A COHERENT "FEEL" TO THE ENTIRE SYSTEM. THEY GREATLY SIMPLIFY CREATION OF NEW DISPLAYS AS WELL AS MODIFICATION OF EXISTING ONES. THEY ALLOW FRAME CONTENTS TO BE STORED ON DISK UNTIL NEEDED THUS SAVING MUCH VALUABLE MEMORY SPACE FOR EXECUTABLE CODE.

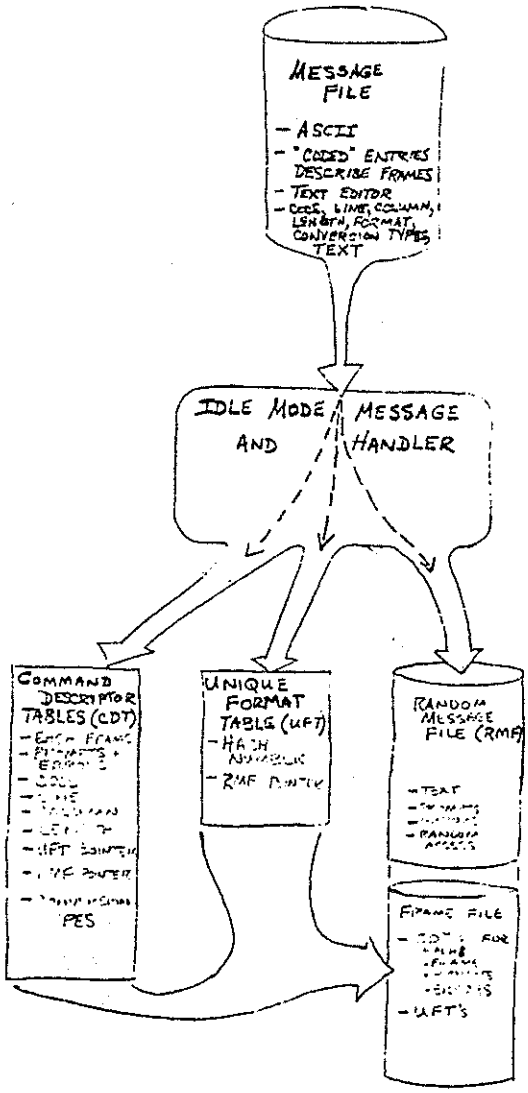


Figure 2.1-1

HOME FRAME

PREPARED BY: _____ CHARGE NUMBER: _____

DATE: _____ PAGE _____ OF _____

```
< OPERATOR ENTRY LINE >
ENTER COMMAND+CR OR CNTL/E+CR TO END          < ERROR AREA >

***** SATELLITE RANGING SYSTEM - MODE SELECTION *****

IN-INITIALIZE
EX-EXECUTE
PR-PROCESSING
DI-DIAGNOSTICS
EP-EPHEMERIDES
```

Figure 2.1-2

2.2 INITIALIZE MODE

- PURPOSE:

  * DEFINE AND VERIFY HARDWARE AND SOFTWARE PARAMETERS WHICH WILL BE USED WHEN TRACKING A SATELLITE.

- COMMENTS:

  * INITIALIZE MODE DISPLAYS THE INITIALIZE FRAME (FIGURE 2.2-1) AND ALLOWS THE OPERATOR TO CHANGE ANY OF THE DISPLAYED PARAMETERS.

INITIALIZE FRAME

PREPARED BY _____ CHARGE NUMBER: _____

DATE: _____ PAGE ___ OF ___

```
< OPERATOR ENTRY LINE >
ENTER COMMAND+CR  OR CNTL/E+CR TO END
***** SATELLITE RANGING SYSTEM - INITIALIZATION *****
*** CIU VALUES ***                    *** CORE VALUES ***
FV-FIELD OF VIEW-MR          XX.X     LA-LEVELING AMPLITUDE-DG   ±XXX.XXXX
AT-ATTENUATION-DB            XX.X     LP-LEVELING PHASE-DG       ±XXX.XXXX
DV-DIVERGENCE-MR             X.XXX    CA-COLLIMATION ANGLE-DG    ±X.XXXXX
ST-START THRESHOLD-MV        XXX.X    RF-RANGE OFFSET-M          ±XXX.XXX
SP-STOP THRESHOLD-MV         XXX.X    AF-AZIMUTH OFFSET-DG       ±XXX.XXXX
TF-TIME OFFSET-SEC           ±XXX.XXX EF-ELEVATION OFFSET-DG     ±XXX.XXXX
MR-MINIMUM RANGE-KM          XXXXX.XX SG-SAG-DG                  ±X.XXXX
RG-RANGE GATE WIDTH-KM       XXXX.XXX MA-MINIMUM ATTENUATION-DB  XX.XX
                                      RV-RCVR PWR REF-DB         ±XX.XX
                                      FI-LASER FIRE INTVL-MS     XXXX
                                      RC-REFRACTION CONSTANT     X.XXX
                                      SA-RANGE SURVEY AZ-DG      ±XXX.XXX
                                      SE-RANGE SURVEY EL-DG      ±XXX.XXX
CL-CALIBRATION                        SC-RANGE SURVEY SAMPLE COUNT XXXXX
```

Figure 2.2-1

TWO MAJOR TYPES OF PARAMETERS ARE SHOWN, VALUES WHICH ARE CIU HARDWARE SETTINGS AND VALUES WHICH ARE USED INTERNALLY BY THE COMPUTER PROGRAMS DURING TRACKING.

* THE INITIALIZE MODE TASK IS HEAVILY OVERLAID. IN GENERAL, EACH COMMAND CAUSES AN OVERLAY TO BE LOADED FROM DISK.

* INITIALIZE MODE WILL INVOKE THE CALIBRATION MODE TASK IN RESPONSE TO THE APPROPRIATE COMMAND.

* TYPING "CNTL/E" CAUSES THE CURRENT PARAMETERS TO BE SAVED AND

THE IDLE MODE TASK TO BE RE-INVOKED.

2.3 CALIBRATION MODE

- PURPOSE:

  * CONTROL MOUNT LEVELING AND COLLIMATION.

- COMMENTS:

  * CALIBRATION MODE USES CODE WHICH IS A SIMPLIFIED VERSION OF THE MOUNT POINTING SUBROUTINES IN EXECUTE MODE. UP TO TEN PRE-SURVEYED TARGETS CAN BE UTILIZED. COMMANDS ARE PROVIDED TO DEFINE NEW SITES, POINT IN EITHER DIRECT OR DUMPED POSITION TO A TARGET, DELETE OR CHANGE EXISTING TARGET PARAMETERS, TAKE OFFSET MEASUREMENTS, AND CALCULATE LEVELING AND COLLIMATION CORRECTIONS (SEE FIGURE 2.3-1.)

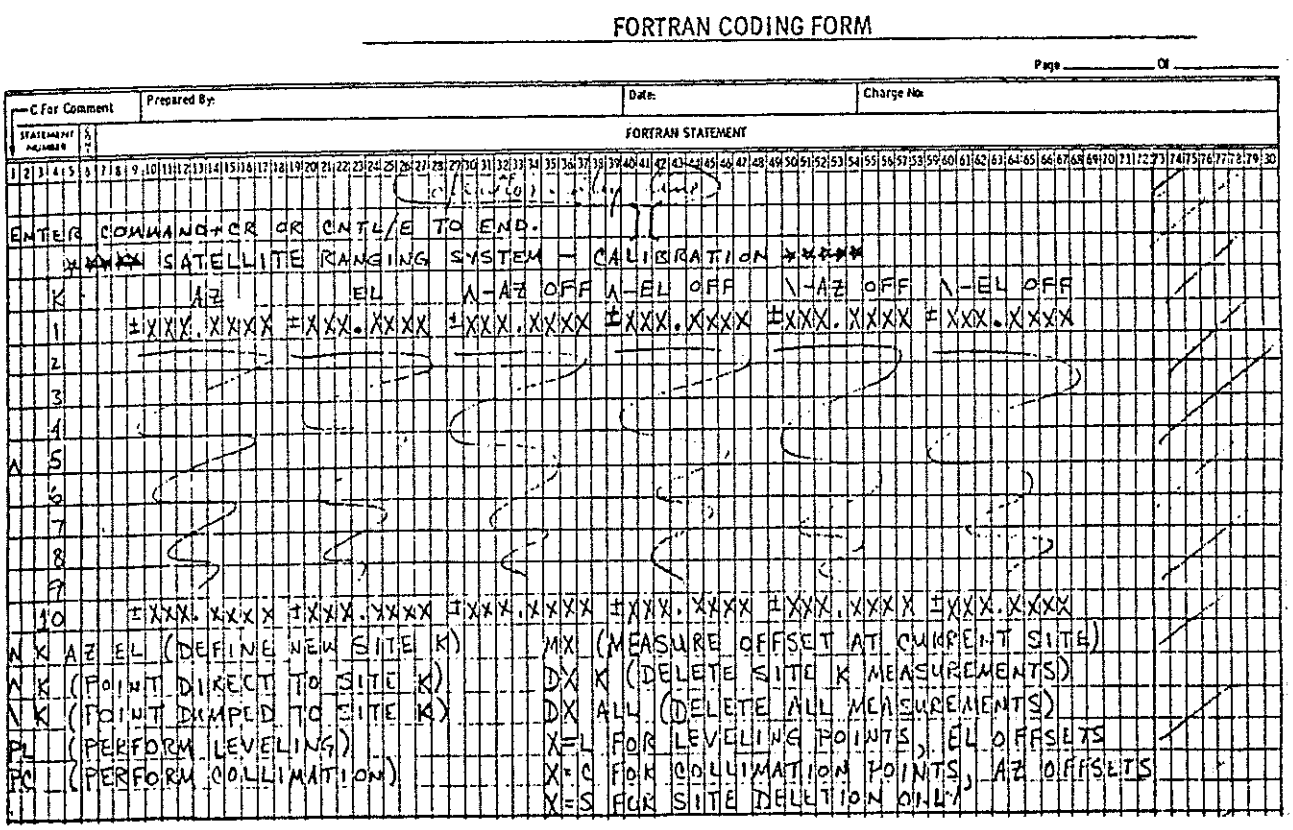


Figure 2.3-1

  * THE CALIBRATION MODE TASK IS NOT OVERLAID.

  * TYPING "CNTL/E" CAUSES THE INITIALIZE MODE TASK TO BE RE-

INVOKED.

2.4 EPHEMERIDES MODE

- PURPOSE:

  * CONTROL PREPARATIONS FOR AND CALCULATIONS OF SATELLITE EPHEMERIDES.

- COMMENTS:

  * EPHEMERIDES MODE CONSISTS OF SEVERAL TASKS--EACH OF WHICH PERFORMS A PARTICULAR PART OF THE OVERALL EPHEMERIS FUNCTION (SEE FIGURE 2.4-1.)

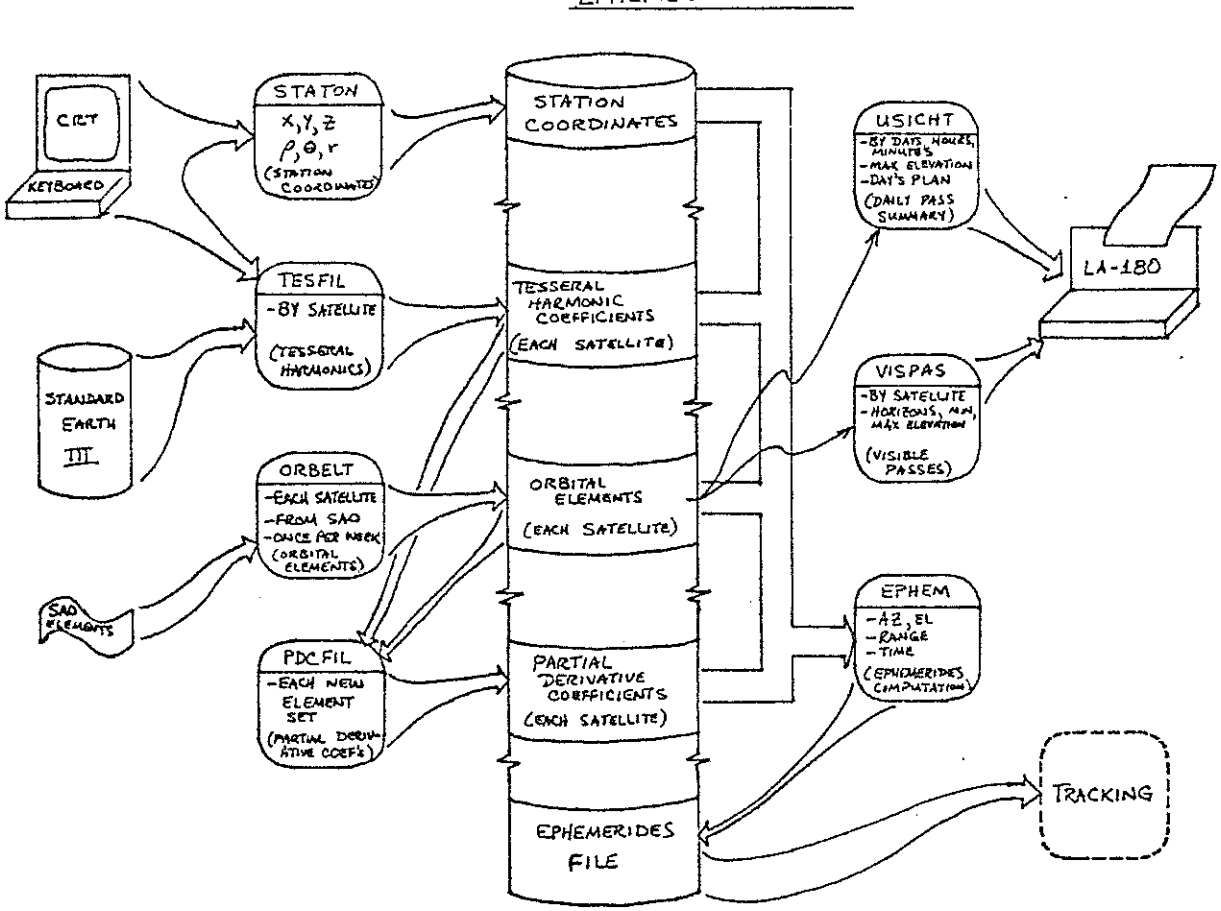


Figure 2.4-1

  * "SRSEPH" (NOT DEPICTED IN FIGURE 2.4-1) DISPLAYS A CRT FRAME WHICH ALLOWS THE OPERATOR TO SELECT THE PARTICULAR TASK WITH WHICH HE WISHES TO WORK. THE TASKS ARE SUMMARIZED IN THE FOLLOWING PARAGRAPHS.

* "STATON" ACCEPTS THE STATION COORDINATES FROM THE OPERATOR.

* "TESFIL" CONTROLS CREATION OF THE TESSERAL HARMONICS FILE FOR EACH SATELLITE. TESSERAL HARMONICS ARE CONSTANTS AND COEFFICIENTS WHICH ARE USED IN EQUATIONS WHICH DESCRIBE THE EARTH'S GRAVITA- TIONAL FIELD. THIS FUNCTION IS USUALLY PERFORMED ONCE PER YEAR OR LESS.

* "ORBELT" CONTROLS ENTRY OF THE ORBITAL ELEMENTS FROM THE SMITHSONIAN ASTROPHYSICAL OBSERVATORY. THIS FUNCTION IS USUALLY PERFORMED ONCE PER WEEK.

* "PDCFIL" CALCULATES COEFFICIENTS OF PARTIAL DIFFERENTIAL EQUATIONS WHICH WILL BE USED IN COMPUTING SPECIFIC SATELLITE ORBITS. THIS FUNCTION MUST BE PERFORMED WHENEVER NEW ORBITAL ELEMENTS HAVE BEEN ENTERED AND BEFORE ANY NEW ORBITS ARE CALCULATED USING THOSE ELEMENTS.

* "VISPAS" CREATES A LISTING OF VISIBLE PASSES FOR ANY SATELLITE FOR ANY TIME PERIOD. THE CALCULATIONS ARE APPROXIMATE AND FASTER THAN FULL ORBITAL COMPUTATIONS. THIS FUNCTION IS USUALLY PERFORMED ONCE FOR EACH SATELLITE WHEN NEW ORBITAL ELEMENTS ARE ENTERED.

* "USICHT" IS SIMILAR TO VISPAS EXCEPT THAT ITS REPORT PROVIDES A DAY-BY-DAY SUMMARY OF SATELLITE PASSES.

* "EPHEM" CALCULATES ONE FULL, VISIBLE PORTION OF A SATELLITE ORBIT. OUTPUT FILES FROM TASKS STATON, TESFIL, ORBELT, AND PDCFIL ARE USED AS INPUTS TO EPHEM. ALSO, THE OPERATOR MUST ENTER SEVERAL PARAMETERS INCLUDING START TIME, DATE, SATELLITE NUMBER, AND 72 CHARACTERS OF FREE-FORM TEXT (A SPECIAL FORMAT FOR THIS TEXT MUST BE USED IF SELECTED MISSION SAMPLES ARE TO BE PUNCHED USING THE SAOPT UTILITY PROGRAM. SAOPT IS DISCUSSED IN ANOTHER PORTION OF THIS DOCUMENT.) EPHEM WRITES THE PARAMETERS, TEXT, CALCULATED TIMES, AZIMUTHS, ELEVATIONS, AND EXPECTED RANGES ONTO A DISK FILE CALLED THE EPHEMERIDES FILE. THIS FILE WILL BE READ BY THE EXECUTE MODE TASKS DURING ACTUAL SATELLITE TRACKING. THIS FUNCTION IS PERFORMED ONCE BEFORE EACH SATELLITE TRACK.

.5 EXECUTE MODE

- PURPOSE:

  * CONTROL SATELLITE TRACKING AND SAMPLE DATA COLLECTION.

- COMMENTS:

  * EXECUTE MODE CONSISTS OF FOUR TASKS, TWO WHICH ARE TIME

CRITICAL, ONE THAT IS BACKGROUND, AND ONE WHICH ALWAYS EXECUTES WHENEVER NONE OF THE OTHER THREE REQUIRE THE COMPUTER (SEE FIGURE 2.5-1.)

EXECUTE (TRACKING) MODE

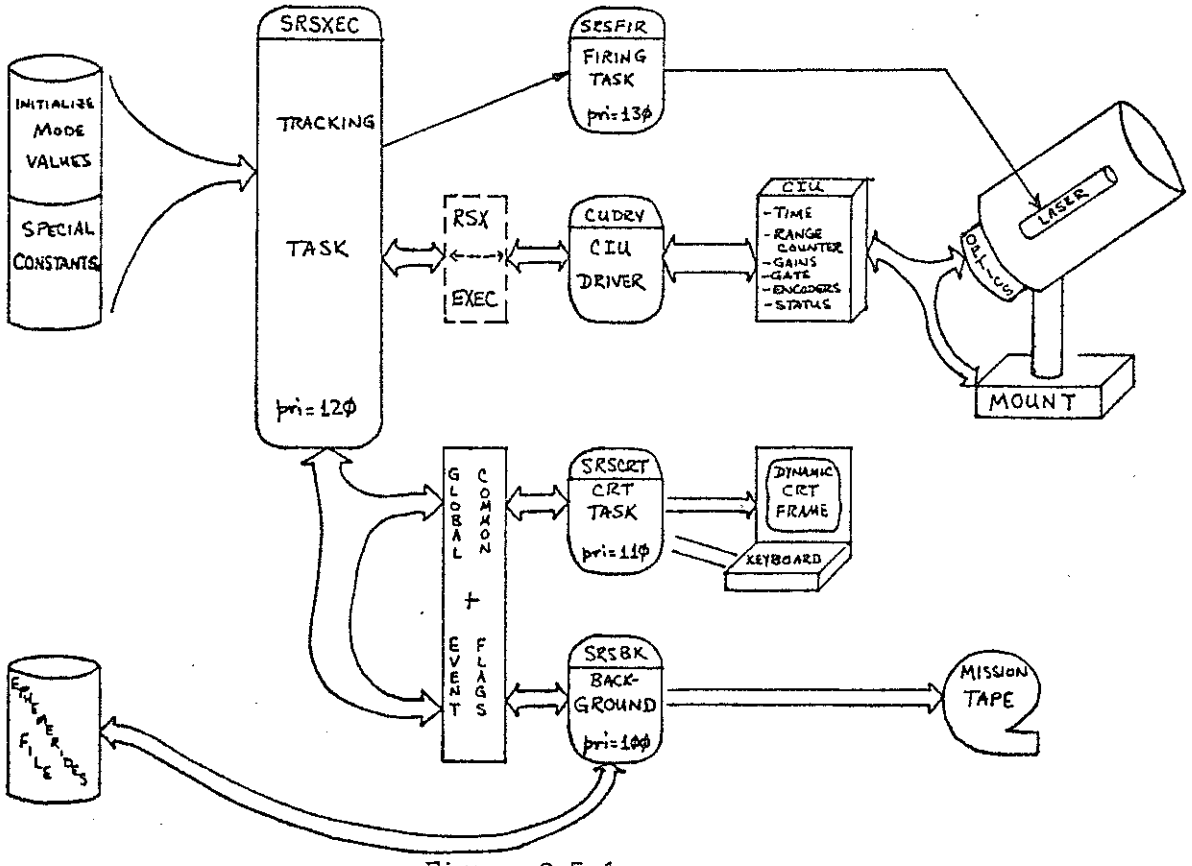


Figure 2.5-1

* "SRSXEC" IS THE TASK WHICH CONTROLS ALL MOUNT AND LASER FUNCTIONS AND GATHERS SAMPLE DATA. IT IS CODED IN FORTRAN BUT USES ONLY SELECTED FORTRAN LIBRARY ROUTINES AND NO FORTRAN I/O. A SPECIAL SUBROUTINE CALLED "CRAMP" IS USED TO AVOID LOADING THE NORMAL FORTRAN OBJECT TIME ROUTINES. THUS THE FORTRAN COMPILER IS USED IN THIS CASE AS AN EFFICIENT GENERATOR OF IN-LINE MACHINE LANGUAGE CODE. THIS IS ESSENTIAL FOR ACHIEVING THE TIME- CRITICAL PROCESSING WHICH IS REQUIRED FOR REAL-TIME CONTROL. SRSXEC HAS SEVERAL OVERLAYS. HOWEVER, DURING REAL-TIME OPERATION ALL REQUIRED CODE IS RESIDENT IN MEMORY SO THAT NO OVERLAY LOADING IS NECESSARY UNTIL THE TRACK HAS FINISHED. SRSXEC DOES NOT DIRECTLY LOAD THE EPHEMERIDES FILE OR WRITE THE SAMPLE DATA TO THE FINAL MISSION TAPE. INSTEAD IT SHARES COMMON BUFFER AREAS WITH SRSBK (DISCUSSED BELOW) AND EXPECTS THAT TASK TO PERFORM THE READ/WRITE FUNCTIONS WHEN REQUIRED.

* "SRSFIR" FIRES THE LASER. IT IS SCHEDULED BY SRSXEC TO BE RUN PERIODICALLY AT A RATE WHICH IS EQUAL TO THE LASER FIRING

INTERVAL AS SPECIFIED ON THE INITIALIZATION FRAME.

* "SRSBK" IS THE BACKGROUND TASK WHICH READS THE EPHEMERIDES FILE AND WRITES THE COLLECTED SAMPLE DATA TO THE MISSION TAPE. IT ALSO PERFORMS DYNAMIC MOUNT LEVELING ADJUSTMENTS ON AN AS-FAST- AS-POSSIBLE BASIS. IT COMMUNICATES WITH SRSXEC VIA GLOBAL DATA BUFFERS AND SOME ASSOCIATED POINTER VARIABLES.

* "SRSCRT" EXECUTES WHENEVER NONE OF THE THREE TASKS DISCUSSED ABOVE REQUIRE THE PROCESSOR. SRSCRT CONTROLS DISPLAY OF ALL DATA VALUES ON THE CRT DURING SATELLITE TRACKING (FIGURE 2.5-2.)

DYNAMIC FRAME

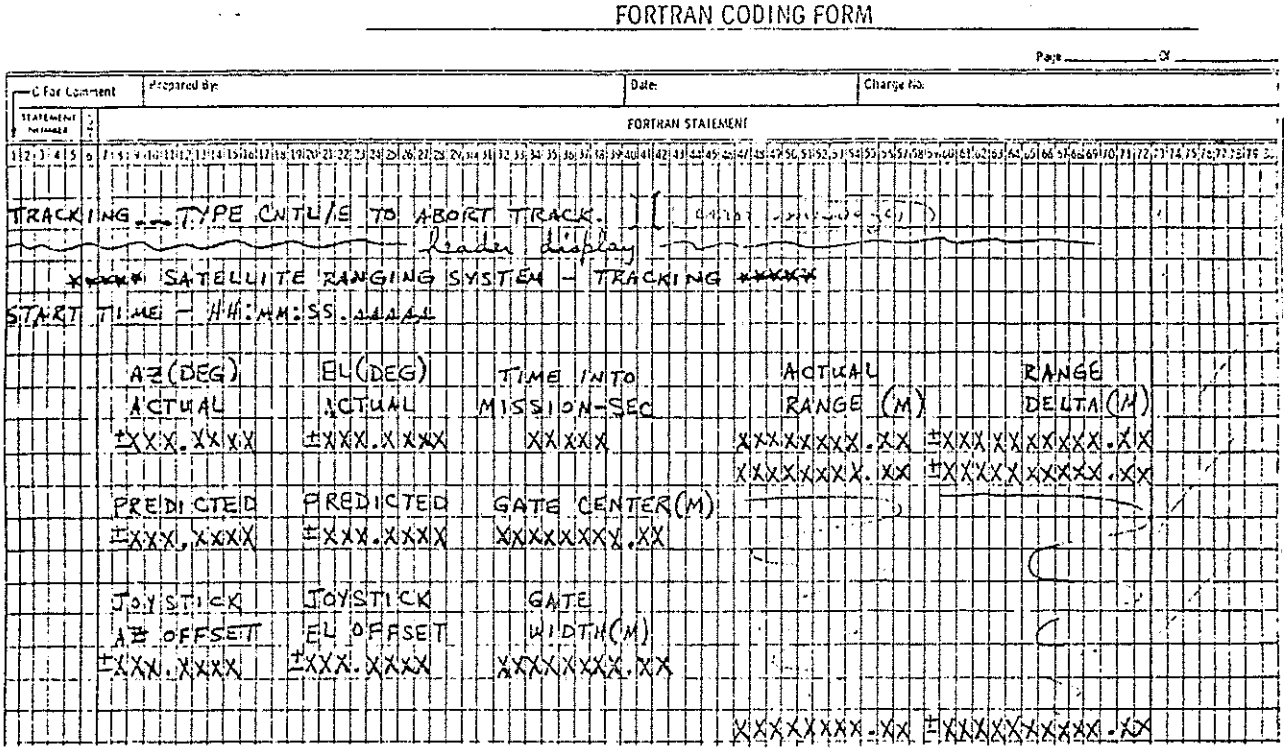


Figure 2.5- 2

IT SHARES A GLOBAL DATA AREA WITH SRSXEC THROUGH WHICH THE DISPLAY VALUES ARE OBTAINED.

2.6 PROCESS MODE

- PURPOSE:

  * EXAMINE SELECTED SAMPLES FROM A MISSION TAPE.

- COMMENTS:

* AFTER DISPLAYING ITS MENU OF OPTIONS (FIGURE 2.6-1) PROCESS MODE RESPONDS TO OPERATOR COMMANDS TO POSITION THE TAPE, LOCATE A MISSION, LOCATE DATA WITHIN A MISSION, AND DUMP DATA IN OCTAL OR INTERPRETED FORMAT TO EITHER THE PRINTER OR CRT.

PROCESSING FRAME

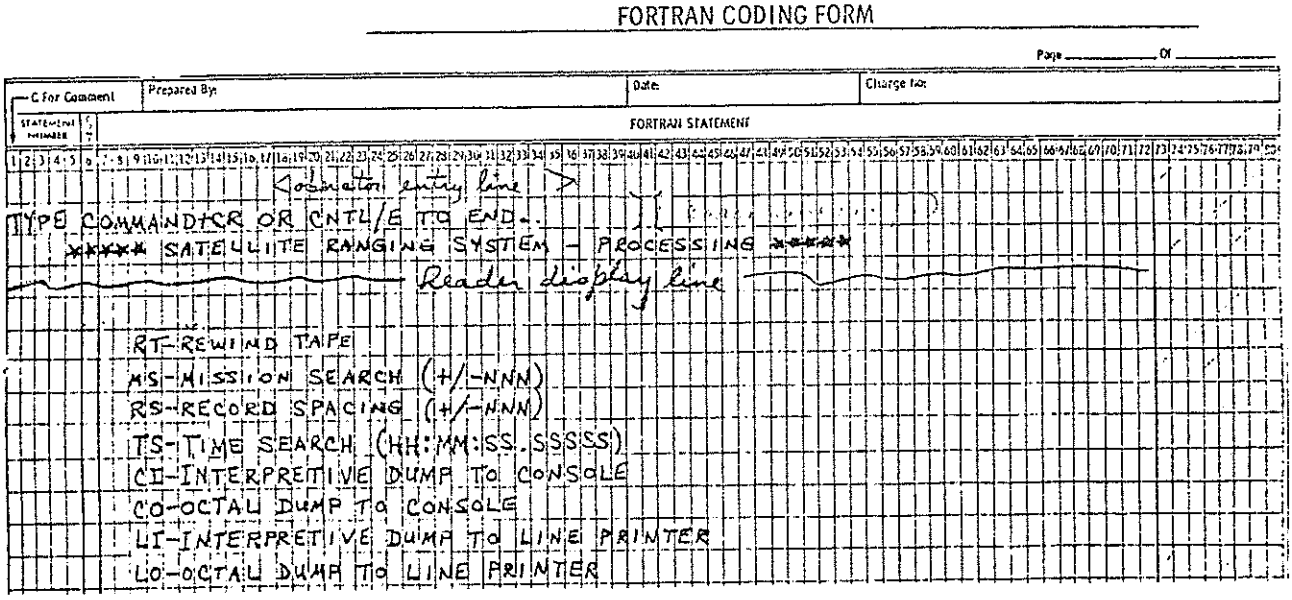


Figure 2.6-1

* THE PROCESS MODE TASK IS OVERLAID. MANY, BUT NOT ALL, OF THE INDIVIDUAL FIELDS WITHIN A SAMPLE REQUIRE A SEPARATE OVERLAY WHEN AN INTERPRETIVE DUMP IS BEING PERFORMED.

* OCTAL DUMPS SHOW EACH WORD OF EACH SAMPLE IN UNINTERPRETED, OCTAL FORMAT. EACH BIT OF DATA IS AVAILABLE FOR INSPECTION. NO SELECTION OF VALUES IS POSSIBLE. LINE SPACING IS USED TO IMPROVE READABILITY. THE LENGTH OF EACH PRINTED LINE IS ADJUSTED TO FILL EITHER THE PRINTER OR CRT LINE SIZE.

* INTERPRETED DUMPS SHOW SELECTED VALUES FROM EACH SAMPLE AFTER THEY HAVE BEEN CONVERTED TO DECIMAL FORM IN MEANINGFUL UNITS. THE SELECTION IS MADE ON THE PROCESS FRAME BEFORE THE DUMP IS STARTED. TIME AND RANGE FIELDS ARE ALWAYS PRINTED. LINE SPACING IS USED TO IMPROVE READABILITY. THE LINE LENGTHS ARE ADJUSTED TO FILL EITHER THE PRINTER OR CRT LINE SIZE.

2.7 DIAGNOSTIC MODE

- PURPOSE:

  * TEST THE ENCODERS FOR PROPER FUNCTION.

- COMMENTS:

  * EITHER AN AZIMUTH OR ELEVATION TEST IS PERFORMED. TO START THE TEST, THE MOUNT IS MOVED TO A LIMIT. IT IS THEN DRIVEN SMOOTHLY IN ONE DIRECTION UNTIL 360 DEGREES HAVE BEEN COVERED. THE ENCODERS ARE SAMPLED RAPIDLY ENOUGH TO DETECT CHANGE IN THE LEAST SIGNIFICANT BIT. THE TEST FAILS IF NON-MONOTONICITY OF THE ENCODER READINGS IS DETECTED. UP TO TWELVE ERRORS ARE REPORTED IN A CRT LISTING (FIGURE 2.7-1.)

DIAGNOSTIC FRAME

FORTRAN CODING FORM

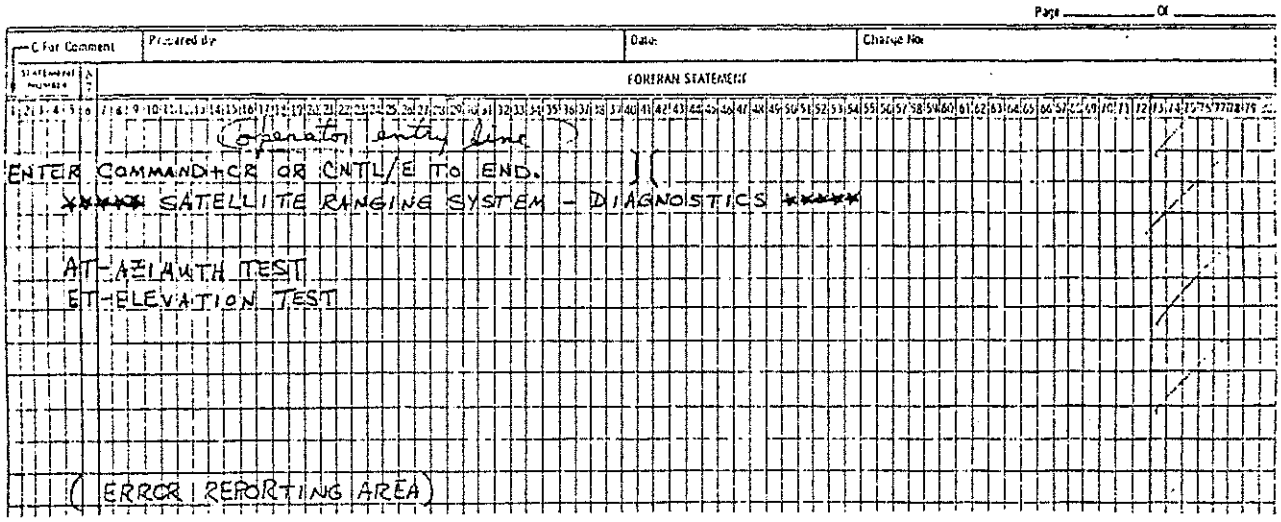


Figure 2.7-1

  * DUE TO LOGICAL MEMORY MAPPING LIMITATIONS, THIS TASK DOES NOT REFERENCE CERTAIN RESIDENT LIBRARIES AND GLOBAL DATA AREAS BUT DEFINES AND/OR LOADS THESE VALUES LOCALLY.

2.8 SAO "QUICK-LOOK" PAPER TAPE PUNCHING UTILITY

- PURPOSE:

  * PUNCH SELECTED SAMPLES FROM A MISSION INTO PAPER TAPE FOR TRANSMITTAL TO SMITHSONIAN ASTROPHYSICAL OBSERVATORY IN CAMBRIDGE, MASSACHUSETTS.

- COMMENTS:

* "SAOPT" DISPLAYS A MENU OF COMMANDS (FIGURE 2.8-1.).

72-Character Header Format

Column(s)	Contents
1-7	Satellite identification number
8	blank
9-13	Station identification number
14	blank
15-16	Year of measurement
17-19	Day of measurement (1 ≤ day ≤ 366)
20	blank
21-25	Pressure in millibars
26	blank
27-30	Temperature in degrees C
31	blank
32-35	Humidity in percent
36	blank
37-45	Calibration Range in meters
46	blank
47-55	Clock Correction in seconds
56-72	Text entered by operator or blanks

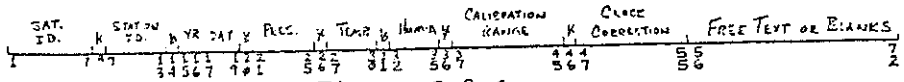


Figure 2.8-1

COMMANDS ARE AVAILABLE FOR LOCATING A MISSION, DEFINING SAMPLE SELECTION PARAMETERS, COUNTING THE NUMBER OF SAMPLES WHICH FALL WITHIN SPECIFIED BOUNDS, AND PUNCHING THE TAPE.

* IT IS AN SAOPT REQUIREMENT THAT THE 72-CHARACTER TEXT FIELD IN THE HEADER BLOCK OF THE MISSION BE DEFINED ACCORDING TO THE FORMAT GIVEN IN FIGURE 2.8-2.

2.9 CIU DEVICE HANDLER (CUDRV)

- PURPOSE:

* DIRECT HARDWARE HANDLING OF THE COMPUTER INTERFACE UNIT (CIU).

SAOPT CONTROL FRAME

```
ENTER COMMAND+CR OR CNTL/E+CR TO END

***** SATELLITE RANGING SYSTEM - SAO PAPER TAPE PUNCHING *****

RT-REWIND TAPE
MS-MISSION SEARCH (+N)
MN-MINIMUM RANGE DELTA BOUND  ±XXXXXXXXXX.XX
MX-MAXIMUM RANGE DELTA BOUND  ±XXXXXXXXXX.XX
SD-SELECTION DELTA                      XXXX
RM-RANGE SURVEY MEAN                 XXXXXX.X
CO-COUNT IN BOUNDS SAMPLES
PT-PUNCH TAPE
```

Figure 2.8-2

- COMMENTS:

  * "CUDRV" SERVES AS THE INTERFACE BETWEEN APPLICATION PROGRAMS ON THE ONE HAND AND THE LASER, MOUNT AND OPTICAL HARDWARE ON THE OTHER. CUDRV IS THE PROGRAM WHICH SENDS FUNCTION CODES DIRECTLY TO THE CIU AND WHICH RECEIVES INTERRUPTS, DATA AND STATUS CODES DIRECTLY FROM IT.

  * CUDRV IS CODED AND INSTALLED AS AN INTEGRAL PART OF THE RSX-11M OPERATING SYSTEM. THIS ALLOWS CUDRV TO TAKE ADVANTAGE OF RSX FEATURES SUCH AS REQUEST QUEUING, I/O PAGE ACCESS, AND INTERRUPT SERVICING. CODING IS IN MACRO ASSEMBLY LANGUAGE TO ACHIEVE THE FASTEST POSSIBLE EXECUTION SPEED.

  * TO FURTHER INCREASE SPEED THE OVERHEAD OF SOME I/O REQUESTS TO CUDRV IS ELIMINATED BY THE USE OF "DIRECTIVE PACKETS." DIRECTIVE PACKETS ARE A SUPERSET OF THE NORMAL I/O REQUEST MECHANISM. NORMALLY A SINGLE I/O REQUEST SPECIFIES THE TRANSFER OF AT MOST ONE CONTIGUOUS BLOCK OF DATA. THE PARAMETERS FOR THE TRANSFER ARE STORED DIRECTLY IN THE I/O PACKET WHICH IS PLACED BY RSX IN THE DRIVER'S QUEUE. REQUESTS TO CUDRV, HOWEVER, ADD ONE LAYER TO THIS STRUCTURE. THE PARAMETERS IN THE QUEUED I/O PACKET ONLY LOCATE A "DIRECTIVE PACKET" IN THE APPLICATION'S ADDRESS SPACE. THE DIRECTIVE PACKET CONTAINS A BLOCK OF PARAMTERS WHICH WILL CONTROL THE CIU TRANSFER. THESE PARAMETERS MAY SPECIFY SEVERAL EXCHANGES OF CONTIGUOUS BLOCKS OF DATA STARTING AT RANDOM CIU ADDRESSES. CUDRV WILL COMPLETE ALL OF THEM BEFORE PERFORMING THE I/O COMPLETION SEQUENCE WHICH ALERTS THE APPLICATION TASK THAT THE TRANSFER IS COMPLETE. THUS, DIRECTIVE PACKETS BYPASS THE RSX OVERHEAD OF SEVERAL I/O REQUESTS.

3 . RS DESIGN AND CODING PROCEDURES

SEVERAL EXPLICIT GUIDELINES WERE FOLLOWED IN THE DETAILED DESIGN AND CODING PHASES OF THE WETTZELL PROJECT. IN PARTICULAR, THE THEORIES OF STRUCTURED, TOP-DOWN PROGRAMMING WERE CONVERTED TO SPECIFIC METHODS WHICH WERE FOLLOWED RIGOROUSLY. A GREAT DEAL OF DETAILED DESIGN WAS HAND-WRITTEN IN A DESIGN (OR META) LANGUAGE AND THOROUGHLY CHECKED FOR CONSISTENCY, PARTICULARLY IN THE REAL-TIME PART OF THE SYSTEM. DESIGN AND CODE WAS REVIEWED BY DIFFERENT PROJECT PROGRAMMERS BEFORE THE ORIGINATOR MOVED TO THE NEXT PHASE OF HIS WORK.

THIS PROCESS WAS A TIME CONSUMING ONE AND CREATED INITIAL FEELINGS OF DISTRUST ON THE PART OF PROJECT MANAGEMENT WHO WERE ANXIOUS FOR END RESULTS. IN THE END, HOWEVER, SEVERAL GENERALLY AGREED UPON REWARDS EMERGED FROM THIS APPROACH:

1. A TRUE AND EARLY COST PICTURE FOR THE SOFTWARE DEVELOPED--THE HARDWARE COSTS WERE MUCH LONGER IN MATERIALIZING;

2. THE SOFTWARE WAS RELATIVELY EASY TO IMPLEMENT;

3. THE SOFTWARE WAS DEVELOPED IN A TIME FRAME REASONABLY CLOSE TO A ONCE-REVISED SCHEDULE;

4. THE SOFTWARE WAS EXTREMELY ADAPTABLE TO NEW REQUIREMENTS BOTH WITHIN THE PROJECT AND LATER, IN WETTZELL, WHEN THE SYSTEM WAS EXPANDED FOR LUNAR RANGING.

THE PROCEDURE OR SUBROUTINE UNIT WAS ALWAYS OF A LIMITED SIZE AND SINGLE PURPOSE. THE PURPOSE WAS EXPLICITLY WRITTEN OUT AS THE FIRST STEP IN PRODUCING EACH ROUTINE. THAT ACT OF WRITING-- OF BEING CONSCIOUS OF THE PURPOSE OF THE ROUTINE ABOUT TO BE PRODUCED--WAS PERHAPS THE SINGLE MOST IMPORTANT TECHNIQUE USED IN CREATION OF THE WETTZELL SYSTEM.

THE OTHER GUIDELINES WERE NOT ELABORATE. YET THEY PROVED TEDIOUS AT TIMES. THE NEED FOR SO MUCH WRITING TO CREATE EVEN SMALL ROUTINES--EVEN THOUGH THE GUIDELINES WERE INTENTIONALLY MADE "BAREBONES"--WAS SOMETIMES FRUSTRATING. IN RETROSPECT, HOWEVER, THE MAJOR ADVANTAGE IN FOLLOWING THE STRICT RULES WAS THAT THE PROGRAMMER'S THOUGHTS AND ACTIVITIES WERE THEREBY CHANNELLED, DIRECTED, CONTROLLED.

THE REMAINDER OF THIS SECTION SUMMARIZES THE RULES THAT WERE FOLLOWED. THE READER SHOULD REMEMBER THAT THEY WERE SUCCESSFUL ON A PROJECT WHICH HAD AT MAXIMUM ONLY THREE PROGRAMMERS. LARGER PROJECTS WILL NO DOUBT DICTATE MODIFICATIONS OF THESE PROCEDURES. SMALLER PROJECTS MAY SAFELY CIRCUMVENT CERTAIN FEATURES. YET THE BASIC CONCEPT OF A SINGLE, WRITTEN PURPOSE FOR EACH ROUTINE (BEWARE THE WORDS "AND" AND "OR") AND A CAREFUL, STEP-BY-STEP, TOP-DOWN APPROACH TO DESIGN ARE VALID NO MATTER WHAT SIZE THE PROJECT.

3.. SRS DESIGN GUIDELINES

- EVERY PROCEDURE HAS ONE AND ONLY ONE FUNCTION.
- NO PROCEDURE DESIGN LARGER THAN A SINGLE SHEET OF PAPER.
- TOP-DOWN DESIGN (PROGRESS FROM HIGHEST CONTROL LEVELS TO MOST SPECIFIC.)
- DO ONLY THE REQUIRED PROCESSING AT EACH LEVEL. DO NOT TRY TO DO TOO MUCH IN ANY ROUTINE.
- NO "GO TO" STATEMENTS IN THE DESIGN (NOTE THAT "GO TO" STATEMENTS ARE PERMITTED UNDER CERTAIN CONDITIONS IN THE FINAL FORTRAN SOURCE CODE.)
- HAVE THE DESIGN REVIEWED BY AT LEAST ONE OTHER PROGRAMMER BEFORE CODING.
- USE ONLY THE DESIGN LANGUAGE ELEMENTS GIVEN IN THE NEXT ARTICLE EXCEPT IN VERY SPECIAL CASES.
- DO THE FIRST DESIGN WITHOUT ARGUMENT LISTS ON PROCEDURE CALLS. MAKING THE NAMES OF PROCEDURES MORE MEANINGFUL WILL HELP. ADD ARGUMENTS AFTER THE LOGICAL FLOWS OF CONTROL ARE ESTABLISHED.
- EVERY COMMON (THAT IS, GLOBAL) AREA HAS ONE AND ONLY ONE REASON FOR BEING CREATED.

3.2 SRS DESIGN LANGUAGE ELEMENTS

STATEMENT *********	EXPLANATION/COMMENTS **********
PROCEDURE	WILL BECOME A "SUBROUTINE".
MAIN PROCEDURE	WILL BECOME THE "PROGRAM" ROUTINE.
<TYPE> PROCEDURE	WILL BECOME A "FUNCTION". <TYPE> IS IN THE SET (INTEGER, LOGICAL, REAL, DOUBLE PRECISION)
BEGIN	STARTING POINT OF A COMPOUND STATEMENT OR OF A PROCEDURE.
END	STOPPING POINT OF A COMPOUND STATEMENT OR OF A PROCEDURE.
"STMT"	ONE OR MORE EXECUTABLE EXPRESSIONS. COMPOUND STATEMENTS ARE ENCLOSED BY

	A BEGIN...END PAIR. SINGLE STATEMENTS ARE TERMINATED BY A SEMI-COLON (";").
"COND"	A LOGICAL EXPRESSION ("CONDITION").
WHILE "COND" DO "STMT"	EXECUTE "STMT" AS LONG AS "COND" IS TRUE. "COND" IS TESTED BEFORE "STMT" IS EXECUTED.
REPEAT "STMT" UNTIL "COND"	SAME AS WHILE...DO.... EXCEPT THAT "COND" IS TESTED AFTER EXECUTION OF "STMT".
IF "COND" THEN "STMTT" ELSE "STMTF"	WHEN "COND" IS TRUE, "STMTT" IS EXECUTED. WHEN "COND" IS FALSE, "STMTF" IS EXECUTED.
CASE "STMT" OF BEGIN VAL1: "STMT1" VAL2: "STMT2" . . . VALN: "STMTN" END	"STMT" IS EVALUATED TO OBTAIN AN INTERNAL ANSWER (CALL IT "VAL".) THEN WHEN "VAL"="VAL(I)", "STMT(I)" IS EXECUTED. CONTROL THEN PASSES TO THE LINE AFTER "END".
VARIABLE'NAMES'CAN'BE'LONG PROCEDURE'NAMES'CAN'BE'LONG	VARIABLE OR SUBROUTINE NAMES MAY BE ANY NUMBER OF CHARACTERS AND/OR WORDS LONG. WORDS ARE CONCATENATED USING A SINGLE APOSTROPHY AS SHOWN.
PROCEDURE CALLS	IT IS NOT NECESSARY TO WRITE "CALL" BEFORE A PROCEDURE NAME. SIMPLY WRITING THE NAME IN-LINE WITH OTHER DESIGN CODE IS ENOUGH TO INDICATE INVOKATION OF THE PROCEDURE.
GLOBAL VARIABLES AND BLOCK DEFINITIONS	DEFINE GLOBAL BLOCKS ON A SEPARATE PAGE. TO SHOW INCLUSION OF THAT BLOCK IN A PROCEDURE, WRITE THE BLOCK NAME IMMEDIATELY AFTER THE "PROCEDURE" STATEMENT.
PROCEDURE ARGUMENTS	ALL ARGUMENTS MUST APPEAR IN EXPLICIT TYPE STATEMENTS.
FOR VAR:=VAR1 TO VARN STEP VARS DO "STMT"	EXECUTE "STMT" WITH "VAR" SET TO "VAR1", "VAR1+VARS", "VAR1+2*VARS", AND SO ON UNTIL VAR > VARN.
<<COMMENTS>>	DOUBLE BRACKETS ENCLOSE COMMENTS.

3.3 SRS CODING GUIDELINES

SRS IS WRITTEN IN FORTRAN WITH ONLY OCCASSIONAL REVERSION TO ASSEMBLY LANGUAGE. THE FOLLOWING GUIDELINES WERE FOLLOWED TO IMPOSE CONTROL AND A DEGREE OF UNIFORMITY ON THE CODE.

- NO ROUTINE LONGER THAN 60 LINES, MOST LESS THAN 30.
- INCLUDE THE STANDARD HEADER (SEE FOLLOWING ARTICLE) BEFORE EVERY SUBROUTINE.
- NO "DROP THROUGH" CODE INTO A LABELLED STATEMENT. WRITE A "GO TO" STATEMENT. THE FORTRAN COMPILER IGNORES SUCH "GO TO" STATEMENTS BUT THE CODE IS CLEARER TO READ.
- EVERY VARIABLE EXCEPT SIMPLE LOOP COUNTERS MUST BE IN AN EXPLICIT TYPE STATEMENT.
- FORTRAN "COMMON" STATEMENTS ARE DEFINED IN ONE AND ONLY ONE "INCLUDE" FILE WHICH CAN BE COMPILED AS PART OF ANY ROUTINE. (THIS IS A FEATURE OF THE DEC FORTRAN COMPILER.)
- NEW LIBRARY ROUTINES OR GLOBAL DATA DEFINITIONS ARE SUBMITTED TO THE SYSTEM MANAGER FOR INCLUSION IN THE SYSTEM.
- INCLUDE EXPLANATORY COMMENTS IN EACH ROUTINE TO EXPLAIN WHAT IS HAPPENING, WHY, AND ANY SPECIAL CONSIDERATIONS.
- AVOID ALL "TRICKY" CODE WHEREVER POSSIBLE. COMMENT IT WELL WHEN IT MUST BE USED.
- EVERY ROUTINE IS ENTERED ONLY THROUGH THE "SUBROUTINE" OR "FUNCTION" STATEMENT, NEVER THROUGH AN "ENTRY" STATEMENT. NEVER USE AN "ENTRY" STATEMENT.
- EVERY ROUTINE CONTAINS ONLY ONE "RETURN" STATEMENT AT THE END OF THE ROUTINE IMMEDIATELY BEFORE THE "END" STATEMENT.

3.4 SRS STANDARD SUBROUTINE HEADER

THE FOLLOWING FORM WAS USED FOR CREATING A HEADER BEFORE EVERY SUBROUTINE. FOR MACRO (THAT IS, ASSEMBLY LANGUAGE) ROUTINES, THE "C" IN COLUMN 1 IS REPLACED WITH A SEMI-COLON. THE ARTICLE NUMBERS ARE AN AID FOR COMPUTER DOCUMENTATION. VARIABLE PARTS OF THE FORM ARE INCLUDED IN <BRACKETS>.

```
C**
C     .0 <ROUTINE NAME>
C          .1 FUNCTION
C             <SINGLE PURPOSE FUNCTION>
```

```
C         .2 CALLING SEQUENCE
C            CALL <ROUTINE NAME AND CALLING SEQUENCE>
C                 <EACH PARAMETER IS EXPLAINED BRIEFLY.>
C         .3 GLOBAL INPUTS
C            <ALL INPUT VARIABLES THAT ARE NOT PART OF THE CALLING>
C            <SEQUENCE ARE EXPLAINED HERE.>
C         .4 UNIQUE VARIABLES/DATA
C            <SPECIAL NOTES THAT EXPLAIN UNUSUAL OR DIFFICULT>
C            <FEATURES OR DATA IN THE ROUTINE.>
C         .5 OUTPUTS
C            <EXPLANATION BY VARIABLE NAME OF EACH CHANGE>
C            <WHICH THIS ROUTINE MAKES ON DATA ITEMS.>
C
        SUBROUTINE <NAME AND CALLING PARAMETER LIST>
C
C       <THE DATA DEFINITION STATEMENTS GO HERE.>
C
C
C
C       <THE EXECUTABLE STATEMENTS GO HERE.>
C
C
C FINISHED.
C
900     CONTINUE
        RETURN
        END
```

HP 9825B-based Software System for Laser Radar

by

R.Neubert
Central Institute for Physics of the Earth
Potsdam, GDR

ABSTRACT

The software system described here has been designed for automatic control of the Potsdam laser station. Minimum computer requirements could be attained by rather simple approximations. Nevertheless reasonable reliable blind tracking has been obtained during the MERIT short campaign for all satellites included in that program. The laser beam divergency used is 1 mrad or even less. In the following we outline the method used for the main parts of the system. For more detailed information the reader is referred to reference [4] of this paper which contains commented program listings and examples.

1 Satellite Position Prediction

The predictions are calculated on the basis of SAO orbital elements consisting in a polynomial part and additional long periodic terms. The polynomial part is linear for all elements except mean anomaly, which is given up to 3rd degree for low orbiting satellites. Among the 16 long periodic terms supplied by the prediction center we use the 4 most important only. These are the terms B_{21} (Ω correction), B_{31} (i correction), B_{51} (χ correction), and the constant part of the η correction. The latter refer to the new SAO element format [1].

The calculation of the satellite position is carried out in two steps.

- In the first step, the Keplerian elements are evaluated for an epoch very near or identical to the time for which the position is wanted (for a pass prediction the culmination time is used in general). In this step (routine "BE(T)") the polynomial part and long periodic terms are taken into account. The major semi axis is evaluated using the well-known first order approximation [2].

- In the second step ("SAPOS") linear changes to the node and perigee and short periodic J_2-perturbations are applied and the satellite position is calculated. In this part the topocentric position including refraction correction is determined as well.

The transformation into the coordinate system determined by the orientation of the 4-axis mount is done by a separate routine if necessary. For this the orientation of the 3rd mount axis (polar axis, main tracking axis) has to be known or calculated before. For a given pass this orientation is determined from 3 predicted topocentric satellite positions around the culmination point fulfilling the geometrical condition, that the 3 angles between the 3rd axis and all the position vectors should become equal (first part of routine "EPHB").

2 Culmination Time Estimation

To print a list of culmination times , we use a very approximate but fast method which is well-known [2]. The calculation is carried out in 3 steps:

Step 1: Let t_0 be the start time of the search, for instance MJD of first day. Then the time difference to the next crossing of the station through the orbital plane (ascending part) is:

$$t_1 = (\lambda - \lambda_b + \Omega_{t_0} - \theta_{t_0})/(\theta_1 - \Omega_1)$$

$$\sin(\lambda) = \tan(\phi)/\tan(i),\ \lambda = 90^\circ \text{ for } i < \phi$$

where

λ_b = longitude of the station
ϕ = latitude
Ω = longitude of ascending node
Ω_1 = rate of change of Ω ($^\circ$/day)
θ = sidereal time angle
θ_1 = rotational speed of the Earth ($^\circ$/day)
i = orbital inclination

For the descending part of the orbit we have:

$$t_{12} = t_1+(180^\circ-2\lambda)/(\theta_1-\Omega_1), \quad t_{12} = t_1 \text{ for } i < \phi$$

Step 2: In the next step the time is added, which is necessary for the satellite to move from its position at t_0+t_1 to the crossing point:

$$t_2 = (M - M_{t_1})/n$$
$$M = \nu - e*\sin(\nu)$$
$$M_{t_1} = M_{t_0} + n*t_1$$
$$\nu = \lambda - \Omega$$
$$\sin(\lambda) = \sin(\phi)/\sin(i), \quad \lambda = 90^\circ \text{ for } i < \phi$$

where

Ω = argument of perigee
n = dM/dt = anomalistic frequency.

Step 3: In the last step of approximation the Earths rotation during the time t_2 is taken into account. Using plane geometry instead of spherical we obtain:

$$t_3 = t_2*((\theta_1-\Omega_1)*\cos(i))/360n, \quad (\theta_1 \text{ and } \Omega_1 \text{ are in } ^\circ\text{/day})$$

The culmination time $t_0+t_1+t_2+t_3$ is accurate to about 1 min. In the "KULM"- routine, t_1 is calculated for the first day only. For the next day a multiple of $360/(\theta_1-\Omega_1)$ is added to t_1 of first day and the other steps are repeated.

3 Real Time Control of the Equipment

For real time control, the computer outputs an instruction each second via the same channel to the control equipment. This repetition frequency is completely sufficient, because the laser shots are spaced by 6 sec. at our station.

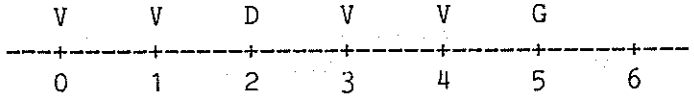
In the normal mode

a), the following types of output information are used: angular velocity along track, cross track correction angle, and receiver gate time (range). The type of information is coded by the first digit of the seven digit word. The remaining 6 digits contain the value of the information (see [4]). Not all of the possible 9 code digits are necessary in the normal mode. The remaining are used to switch the control logic to other modes of operation. In this way the computer can switch to:

b) Start-stop drive of both axes. In this mode being introduced for the planned LASSO experiment, the laser can be programmed to shot at an arbitrary time within the 6 sec. interval using an additional output information (in normal mode the laser repetition rate is fixed).

c) Continuous drive of both axes. This mode is useful for star tracking.

In the following we consider the normal mode a) only. The cross track information and gate time have to be output once each in a 6 sec interval. Therefore, we have the opportunity to change the along track velocity 4 times in each cycle. This is important for a fast and stable adaptation of the system after a perturbation by a manual correction. The tracking algorithm insures, that at the next laser shot both the position and the velocity of the satellite are approximated within the accuracy level of the (shifted) prediction. Let us consider the control cycle in more detail according to the following time diagram:



The full seconds of a cycle are denoted by 0 to 6, where 6 is identical with point 0 of the next cycle. In the upper half of the diagram the type of information accepted by the control circuit in the corresponding point is printed. V stands for velocity, D for cross track correction (δ), and G for gate time. In the moment of acceptance the instruction is immediately executed. For "V" this means, that the velocity is switched to the new value, for "G" the laser is triggered and the gate counter started. In the foregoing cycle the computer calculates the satellite positions in points 4 and 5 taking into account a time shift, which may be changed by the observer in real time. The "real" position of the laser beam at point 0 is calculated by the computer by integrating the velocity values given before. Now the condition is set,

that the laser beam should be directed to the predicted positions in both the points 4 and 5. To obtain a definite solution, the simple additional requirement was introduced, that the same velocity change X should be applied at points 0 and 1, and another change Y at points 3 and 4. Denoting the velocity at point 0 by V_0 and the position in point i by S_i, we obtain for the velocity changes:

$$X = (2(S_4-S_0)-(S_5-S_4)-7V_0)/12$$

$$Y = ((S_5-S_4)-V_0)/2-X$$

In these equations the time interval is removed for simplicity, because it is assumed to be the time unit for velocity (1 sec. in our case). The second equation is identical with the condition, that the total sum of velocity changes should be equal to the difference of the velocities at points 5 and 0. If the required total velocity change but not the positional displacement is very low, the changes X and Y are approximately equal in absolute value, but opposite in sign. This is the case for time corrections near the culmination.

The limited interrupt possibilities of the desktop computer require, that there is no routine in the control program consuming more than 1 sec. for execution. This is no problem at present, because the most critical routine "SAPOS" is well below this limit. So the full program could be used in real time and interpolation techniques have been unnecessary. Some care has been given to the distribution of the computing time to the intervals. The reading and printing of the measurements for instance had to be placed after point 6 (1 sec. after the laser shot).

4 Filtering of the measurements

The selection of the laser returns from noise pulses is done mainly by comparing to predictions. The raw data stored after observation on tape cartridge contain all the measurements. They are read in again by the analysis program and compared to predictions. The differences are below 1000 m. in general for real returns. For best fitting of the predictions to the measurements a time shift may be introduced (Sect. 6.1). The remaining differences are weakly dependent on time and may be easily represented by a low degree polynomial. In this way the instrumental noise level down to some 10 cm. rms is easily reached for the residuals and the false alarm points may be reliably eliminated if the total number of returns is not too low. The filtered measurements are corrected for instrumental effects (calibration, time of flight counter frequency error, eccentric correction of the mount) and stored on a separate tape cartridge.

5 .rediction Improvement Using Own Laser Data

Using data from only one station, prediction improvement has some limitations. The results are very stable, however, if only the mean anomaly is changed. Fortunately, the SAO elements are leading to accurate predictions of the cross track components even if using the simple algorithm outlined. On the other hand, strong along track errors are observed for low orbiting satellites and, therefore, updating of mean anomaly coefficients is useful and necessary.

5.1 Time Shift

The most simple improvement is a constant correction of mean anomaly or an equivalent time shift. For this it is completely sufficient to use two points of one pass only. Two methods have been used in parallel. The first is the standard least square differential improvement leading to the time shift dt:

$$dt = ((D_1-S_1)Q_1+(D_2-S_2)Q_2)/(Q_1^2+Q_2^2)$$

where:

S_1, S_2 = predicted ranges for the two epochs
D_1, D_2 = measured ranges
Q_1, Q_2 = time derivatives of range

This formula gives good results if the remaining range differences (cross track error) are small. If this is not the case, the points should be chosen symmetrically to the culmination at approximately equal range. If measurements are available only before or after the culmination, the following method may lead to better results. It is deduced from the condition, that after introduction of the time shift the range differences should become equal, i.e.:

$$S_1 + Q_1*dt - D_1 = S_2 + Q_2*dt - D_2$$

or:

$$dt = (D_1-S_1-D_2+S_2)/(Q_1-Q_2)$$

This method tends to bring the measured and predicted times of minimum range into close agreement. The latter is exactly the case, if the measured and predicted range versus time curves are identical in shape differing in a constant offset only. The results of this formula may be bad, however, if the points are too close together.

Both formulas are nearly identical, if the points are symmetrically located to the culmination on opposite sides. This may be verified easily by

introducing $Q_1 = -Q_2$ in both equations.

5.2 Orbital Element Improvement

When data from several passes are available, the time shift values may be plotted versus time and extrapolated to estimate the time shift of future passes. More convenient and powerful is a least square adjustment of mean anomaly coefficients using data from all passes simultaneously. For this, the routine "IMPROVE" is included in the analysis program. It uses the first and last point from each pass and minimizes the least square sum of range by standard iterative methods. For each pass, the orbital program has to be called 4 times to evaluate the time derivatives of range consuming about 3 sec. of computing time. Thus for 20 passes about 1 min. is necessary per iteration. Two or even one iteration are sufficient in most cases only.

Let us consider as an example GEOS C using data collected during the spring of 1980. In the following table time shifts (ms) determined by the two point method (Sect. 6.1) are printed as a convenient measure of the along track error. In the first column original SAO elements (reference epoch 44358) are used. It is obvious, that the prediction center used observations from the period of about 2 weeks before the reference epoch. Outside of this period the prediction error increases strongly. For comparison, in the next column the time shifts relative to AIMLASER-predictions are printed. These shifts were calculated treating the ranges predicted by AIMLASER like measured ones in formula 2 of Sect. 6.1. Not for all the passes AIMLASER predictions were available but it is obvious, that an almost constant shift of about 100 ms. exists between the programs. This indicates that the strongly increasing prediction error is mainly due to the SAO elements itself. The opposite sign of the time shift in the past and future, respectively, is caused by the 3rd order term in mean anomaly. Removing this term, the third column results. The divergency is reduced somewhat, but remains strong. In the next 2 columns the time shifts are given for elements updated using all passes before MJD=44359.9 and using the third order term or not. It is seen that the 3rd order polynomial gives slightly better extrapolation behavior in this case. Current practice shows, that the most stable method seems to be the use of 2nd order connected with frequent readjustment of the mean anomaly coefficients as soon as new data are obtained. As an example in the next column the results of fitting to passes up to MJD=386.9 (pass No. 1 to 18) are given. Second order is much better in this case. Using frequent updating, we used the same SAO element set for 6 weeks or even more with no significant loss in accuracy.

5.3 Comparison to AIMLASER

Let us now compare the two programs more directly than in the last section. GEOS C is again considered, because it is the most critical laser satellite with respect to predictions. In Table 2, angular differences are given in seconds of arc and in addition range differences in meters. In the first set, original elements are used, in the second a time shift of 67 ms. is introduced in the Potsdam program.

Table 1: Time Shifts for Different Variants of Improvement.
GEOS C. Reference Epoch: 44358

No	MJD	Orig. elem.	AIM-Po	M3=0	M3=0 No1-10	M3#0 No1-10	M3=0 No1-18	M3#0 No1-18
1	334.8	2210	92	-3199	- 14	- 12	- 47	- 9
2	334.9	2114		-3151	- 30	- 28	- 61	- 25
3	335.8	1832		-2885	- 17	- 19	- 34	- 24
4	344.8	210	131	- 773	12	8	67	2
5	344.9	243		- 740	46	42	102	37
6	345.8	147	135	- 629	- 1	- 3	54	- 4
7	349.8	93	131	- 140	7	10	48	27
8	350.8	63	97	- 94	- 30	- 24	4	- 6
9	358.9	76	113	77	5	7	- 76	- 18
10	359.9	25	95	28	7	5	- 93	- 35
11	361.9	- 119		- 94	26	15	- 120	- 69
12	362.9	- 139	139	- 90	118	101	- 47	- 7
13	363.8	- 275		- 190	126	103	- 70	- 39
14	363.9	- 202	144	- 113	213	188	13	43
15	364.9	- 346		- 207	236	204	11	25
16	365.0	- 314	139	- 164	309	274	71	84
17	368.9	-1241		- 690	396	318	37	- 57
18	368.9	-1187	107	- 629	461	383	104	8
19	372.0	-2361	90	-1191	547	418	73	- 165
20	373.0	-2840	87	-1406	575	427	60	- 238
21	373.9	-3392	52	-1659	562	394	9	- 352
22	379.0	-6851	70	-2929	802	496	10	- 796
23	379.0	-6965		-2955	824	515	25	- 792

This table shows that the cross track angular differences are indeed very small. The along track differences are significant, but, as already pointed out in Sect 6.2, they are usually small compared to the real prediction error. Using updating of elements, long periodic effects are easily absorbed in the polynomial coefficients.

6 Coordinate Transformations Related to Star Observations

Star observations are necessary to check and maintain the pointing accuracy of the mount. This is becoming increasingly important in connection with narrower laser beams. The first two axes of the mount used at our station can be set with a precision of 1 minute of arc only. The mechanical quality of the step motor driven 3rd and 4th axes permits pointing to 5 arcsec. on the other hand. Thus, a possibility to achieve this accuracy would be the observation of a few reference stars immediately before the satellite tracking.

For flexible programming we found it useful to have a set of subroutines for the orthogonal transformations between the astronomical systems. The following systems need to be considered:

1. The equatorial system with x-axis to the equinox at 1950.0

Table 2: Comparison to AIMLASER
GEOS C, Reference Epoch: 44358

		No time shift			Time shift =67ms		
		Angul. along track	Diff. cross track	Range	Angul. along track	Diff. cross track	Range
80 4 12	20 33 45	85	14	-674	16	14	-426
	20 35 5	144	16	-401	53	16	-349
	20 36 45	148	12	97	73	13	-120
80 4 20	20 13 15	88	12	-825	30	13	-471
	20 15 15	221	13	-350	106	13	-284
	20 17 35	157	3	520	94	2	185
80 5 1	20 48 50	59	31	-338	-15	31	-118
	20 50 10	93	29	-163	0	29	-159
	20 51 50	87	21	114	18	21	-141
80 5 10	21 52 50	27	12	-327	-47	12	- 98
	21 54 10	59	10	-243	-34	10	-228
	21 55 50	68	5	- 50	- 2	5	-299
80 5 20	0 35 5	69	25	-697	0	25	-401
	0 36 45	158	27	-368	51	27	336
	0 38 45	135	18	237	67	18	- 61
	rms	118	18	424	52	19	276

2. The equatorial system with x-axis to instantaneous equinox

3. The horizontal system

4. The "SBG"-system defined by the setting of the 1st and 2nd axis of the mount.

For all practical problems it is sufficient to have transformations between all the 3 pairs of systems adjacent in this list (forward and backward). Rectangular coordinates are used throughout introducing no singularities in the vicinity of the pole. In the following we comment the approximation used for the transformation from system 1 to 2 and vice versa only. It is not a clean rotation of the coordinate system because of inclusion of seasonal aberration. The diagonal elements of the precession- nutation matrix are set to unity simply and the non diagonal are approximated by:

$$m_{12} = -m_{21} = -6.119E\text{-}7*t+7.67E\text{-}5*\sin(F)$$

$$m_{13} = -m_{31} = -2.6604E\text{-}7*t+3.33E\text{-}5*\sin(F)$$
$$m_{23} = -m_{32} = -4.47E\text{-}5*\cos(F)$$

$F = 12.11279^{\circ} - 0.0529539(^{\circ}/day) * t$
$t = MJD - 33282$ (days after 1950.0).

For the inverse transformation the signs of the non-diagonal elements have to be reversed.

For aberration, the simple vector relation holds:

$$dX_i = (V/c)(V_i - (V_k X_k) X_i), \quad i,k = 1,2,3$$

In this equation the vectors are represented by their rectangular coordinates using lower case letters as indices. The summation convention over equal indices occurring in a product is adopted. The used symbols are:

V = velocity of the station in the used inertial reference system
c = velocity of light
V_i = unit vector of station velocity
X_i = unit vector directed from the station to the star
$(V_k X_k)$ = scalar product
dX_i = aberration correction

In the equatorial system the direction of the orbital velocity of the Earth is approximately:

$V_1 = \sin(L)$
$V_2 = -\cos(L)\cos(\varepsilon)$
$V_3 = -\cos(L)\sin(\varepsilon)$

where:

L = longitude of the sun in the ecliptic
$\varepsilon = 23.4425$ = inclin. of the ecliptic

For L the approximation is used:

$$L = 360(t/365.242 - 0.222).$$

7 References

[1] M.R. Pearlman: "Updating Ephemerides for Laser Tracking Memorandum directed to SAO elements users June 1, 1979

[2] K. Arnold: "Methoden der Satellitengeodaesie" Berlin (Akademieverlag) 1970

[3] R. Neubert, I. Prochazka: "Software for Automatic Laser Satellite Tracking" Submitted to the Intern. Scient. Conf. Sect.6 INTERCOSMOS, Albena (Varna) ,Sept. 15-21, 1980

[4] R.Neubert: "Laser Radar Software (Listings, Examples)" Report ZIPE, Sept.1981

Software Package for Station SAO No. 7831

by

A. Novotny and I. Prochazka
Czech Technical University
Brehova 7
115 19 Prague 1, Czechoslovakia

ABSTRACT

This software package was developed for the satellite laser ranging station no. 7831 in Helwan, Egypt. It guarantees the computer control of the automatic tracking system. The simple method for on-site prediction correction is included. The computer hardware used includes:

- Hewlett-Packard 2100S processor,
- 64 KByte of internal memory,
- magnetic disc HP7900 with capacity of 5 MByte,
- standard I/O paper tape devices,
- the Multiprogrammer HP6940B and communication channel HP-IB (IEEE 488).

The RTE II real time disc operating system is applied. The laser radar electronics is connected to the CPU partly via the HP-IB channel, partly via Multiprogrammer special interfaces. The whole software package consists of four principal program blocks:

1. programs for satellite position prediction,
2. programs for system calibration and check,
3. satellite tracking and ranging programs,
4. postpass ranging data handling and analysis.

All the programs were written in FORTRAN IV language.

1 Satellite Position Prediction Software Package

For blind, fully automatic satellite tracking and laser ranging the position of the satellite must be predicted with the accuracy better than is the beam divergence of the laser beam transmitted: two arcmin for low-orbit and one arcmin for Lageos satellite. The standard algorithm of evaluation of the satellite position of the AIMLASER program is used [see 1,2]. This algorithm is based on the use of so-called Kozai mean elements, which are routinely provided by SAO and distributed via telex. To fit into the minicomputer HP2100 with only 16k words of user-available memory, the original program was significantly modified and divided into several separate programs. All the input/output procedures and transfers of the data from one program to another one are carried out via the disc, partly within the file structure, partly on the dedicated data region used for direct data access (DDA). Each of the programs may be used separately, in an interactive mode, or may be scheduled by the transfer file in sequence in the batch processing mode. The satellite position prediction software package was constructed to deal with the set of maximum 6 satellites. The standard ones are: Geos A, Geos C, Beacon C, Starlette and Lageos. The satellites are identified within the program by their numbers. To simplify this, only the first four digits of them are used. Any time the satellite identification number is mentioned in the following sections, only these first four digits of the original id. number are kept in mind.

1.0.1 Prediction Software Scheme

The prediction procedure consists of four groups of programs:

- Group 1 - culmination times evaluation programs

  * SEEKC: computes the culmination times for one satellite within given period of time.

  * ARANG: rearranges the culmination times for all satellites in chronological order, checks the daytime conditions.

  * CULM: computes the rough values of azimuth and elevation in satellite culmination and prints out a "time table" for the satellite ranging on the station.

- Group 2

  * PDCTS: computes the slowly varying coefficients for evaluation of perturbations due to tesseral harmonics.

- Group 3 - position prediction programs

  * AIMFL: computes position of the satellite in 10 points near the culmination taking into account all the perturbations available.

* SCHED: small help program which schedules the program AIMFL.

* ATS: computes the time shift parameter from the foregoing satellite ranging for prediction accuracy improvement.

- Group 4 - input/output data sets handling and help programs.

* ORBEL: interactive input of satellite orbital elements.

* CORCT: test of the data for orbital elements by means of four letter checkword provided together with data, correction of bad digits.

* TSHAR: reformating of the tesseral harmonics coefficients and storing them on the direct disc access area.

* TESFL: interactive creating of special tesseral harmonics set from Standard Earth III for special purposes.

* STATN: the station coordinates files manipulation program.

* INFOR: informs about the program, satellite name and program phase just being executed in CPU.

On the end of the prediction procedure the user gets the table of satellite passes for a given station, period of time and satellites and the table of universal times and positions of these satellites in the x,y,z coordinate system in 10 points within each pass. The actual position az/el/rang of the satellite is computed in the on-line procedure by means of the Chebychevian approximation.

1.0.2 Prediction software data structure

- ELWORK: file, created by the ORBEL program, contains orbital elements, which are just being entered.

- EL++++: orbital elements set for satellite with id. number equal to ++++ .

- ST****: station coordinates file for station with id. number equal to ****.

- SE++++: culmination times for satellite ++++.

- CTIMES: file with culmination times for all 6 satellites arranged in the chronological order.

- TTABLE: the "time schedule table" for satellite laser ranging on the desired station.

- DATA3: file with tesseral harmonics coefficients set, contains the

Standard Earth III coefficient set and special tesseral harmonics sets for some satellites.

- TRDATA: file with computed positions of the satellites.

1.0.3 Dedicated disc area structure

To simplify the program PDCTS and AIMFL, the direct disc access to several data sets is arranged. Totally, the first 21 disc tracks on the non-system disc cartridge are used for this purpose. They are protected against wrong manipulation by means of initialization of the corresponding cartridge from track no. 21. The scheme of the *DDA* area is on fig. 2. Generally, there exists 3 data blocks in the DDA region:

1. intermediate results for program PDCTS and AIMFL,
2. six different tesseral harmonics coefficients sets,
3. values of slowly varying coefficients for computation of perturbations due to tesseral harmonics for six different satellites.

2 Description of Programs

2.1 Group 1

2.1.1 Program SEEKC

It computes the times of culminations of the satellite above the given place on the Earth. The searching algorithm described in [4,5] is used. This algorithm seems to be a good compromise between the computing accuracy and the computing time required. Program SEEKC runs separately for each satellite. Input parameters date, no. of days, and satellite id. no. are entered via a RMPAR routine to the program. The program calculates all the possible culmination times during one day and stores these times in a linear array. Routine *RRRR* is responsible for it. Then the positions of the satellite are computed for these culmination times in the *SATPS* routine omitting all the perturbations. The elevation angles are then computed and compared with the minimal values required for corresponding satellites (default values are 50° for Lageos, 30° for all the other satellites). The culmination times, in which the satellites culminate above the limit, are stored on the file SE++++, where ++++ is the satellite id. number. The times are stored in the form of non-integer value of modified julian date. The program is started as follows:

RU,SEEKC,iyr,mo,idy,nday,nsat

where

- iyr,mo,idy.... is the date of beginning of prediction,

- nday..... is the no. of day,

- nsat.....satellite id. no.

2.1.2 Program ARANG

It rearranges the culmination times for all the satellites into one file in a time increasing order. All the files SE++++ are opened, read and stored in one two-dimensional array. The elements of this array are moved to another linear array in a time increasing order (thus, the smaller elements are moved earlier). The culmination times for one and the same satellite which differs only in some minutes of time are detected and replaced by a single value obtained as an average of these values. (This culmination time duality may occur for the searching procedure used.) Then, the culmination times are converted into the form of the date, hour and minute. Optionally, the daytime passes are suppressed. The sequence of the culmination times is stored on the file *CTIMES*.

2.1.3 Program CULM

This program computes the position (azimuth, elevation) of the culmination for satellite passes listed in the file *CTIMES*. Only the mean orbital elements are used, all the perturbation effects are omitted. As an output the program CULM prints out a table with the time schedule for satellite ranging. Rough position of the culmination is included to guess the observation priority in the case that several satellite passes overlap. When the program CULM is started in a batch processing mode, the results are stored on the file TTABLE. The print out sample of the file TTABLE is on fig.3. The program can be scheduled by the command:

RU,CULM,irec

where

- irec.... is the no.of record of the CTIMES file from which the procedure should start (default 1).

2.2 Group 2

2.2.1 Program PDCTS

This program computes the slowly varying coefficients for computation of perturbations due to tesseral harmonics. From the theory of Kaula [7], it can be gathered that analytical expressions, describing the short-period tesseral harmonics effects, consist of very slowly changing functions of the mean elements to be multiplied among others by several trigonometrical terms in the

tr ıe and the mean anomaly. It can be shown that the slowly varying terms may be considered to be constant for periods of time of the order of several weeks. Therefore, the so-called intermediate results are computed only once for one orbital elements set for the time period of the next two weeks.

Program PDCTS is composed of the main program and two program segments. The main program contains only the common block area declaration and the next segment call. First segment PDCT1 reads the satellite id. number and the optional print out parameter from the RMPAR routine. Then the appropriate orbital elements data set is read from the file EL++++ and the corresponding set of the tesseral harmonics coefficient from the DDA area. The epoch, for which the computation is carried out, is set to be equal to the epoch of the orbital elements. Optionally, the orbital elements, the tesseral harmonics coefficients and the convergence criteria used are printed out. Then the PDCT2 program segment is called. Segment PDCT2 calls, via some help routines, the function GRAHV for evaluation of the slowly varying coefficients. Finally, these coefficients are stored on the DDA area. Although the algorithm is the same as in the original Aimlaser program, several changes were to be made in the version for the minicomputer to reduce the memory size required and to overcome the problems of the restricted dynamical range of computation (only 48 bits in double precision in contrast with 64 bits on the CDC or the IBM computer).

1. The original function TESSRB was shortened to its initial part, to computing of the coefficients and storing them on the DDA area, only.

2. The two dimensional array CLMP, in original version dimension of 6x300, is not present in the computer memory in its original form. Only one column of it is present, while all the others are stored on the DDA area. For fast and effective handling with the elements of the matrix CLMP two special routines were created. Routine *CLMPD* reads the specified column of the matrix from the DDA area, the *CLMPT* routine stores the values of one column of the CLMP matrix on the DDA area. To reduce the number of disc transfers, the data are transferred to the disc not before the new CLMP matrix column is dealt with. Otherwise, the values are stored in a help 6 element array within the *CLMPT* routine.

3. In the GRAHV, BAT and FECK routines some changes were made, according to [3], to overcome the troubles with the restricted dynamical range of computation. Namely, in evaluation of binomial coefficients, factorial computation.

4. For sequential reading and writing of the results on the DDA area, two new routines were created: DWRIT and DREAD. The function of both of them is quite self-explaining.

Analogically to the original Aimlaser program, different reduced tesseral harmonics sets may be used for computation. Subroutine *SIZES* determines which of six sets present will be used. One additional output parameter was implemented to the original SIZES routine: NKAT. It denotes the sequence

number of the tesseral harmonics coefficients set within the catalogue (NKAT=1,...,6). Satellites Be-A, Ge-A, Starl., Ge-C and Lageos have the number 1 to 5, respectively, to all other satellites the tesseral harmonics coefficients set present on the sixth position in the catalogue will be assigned. On this position the Standard Earth III set or any reduced set (created by the special program *TESFL*) may be placed. Running the program:

RU,PDCTS,nsat,iprint

where

- nsat .... is satellite id. number
- iprint .. is optional print out parameter (default zero)

2.3 Group 3

2.3.1 Program AIMFL

It computes the position of the satellite in the Cartesian coordinate system x,y,z in 10 time points equidistantly spaced round the culmination time. The program consists of the main program and three program segments. The main program declares all the common block variables and calls the first segment AIMF1, only. In segment AIMF1 the satellite id. number, the date, hour and minute of culmination time is read from the file CTIMES. The number of the record to be read from this file is entered to the program via the *RMPAR* routine together with code number identifying the perturbations which are to be used in computation. Then the appropriate satellite orbit elements set, station coordinates and the tesseral harmonics coefficients are read from the corresponding files. The beginning time of the pass and the time step for position evaluation are calculated from the culmination time and the rough duration of the pass.

The second program segment *AIMF2* evaluates the positions of the satellite. With comparison to the original Aimlaser program, several significant modifications were made in the program structure to reduce the memory size required and to decrease the level of nesting of the routines.

For purpose of the segment AIMF2 the routine *INST* was written. In fact, it is separated fourth part of the *MYORB* routine, only. It evaluates the instantaneous values of orbital elements. In the segment AIMF2 the time for position computation is calculated, then the instantaneous values of the orbital elements are calculated within the INST routine. According to the original program version, the call to the function *ZATPZB* should follow. In AIMFL program, the routines ZATPZB, SIDTIM and TSSRLB were introduced directly into the program segment AIMF2. This modification saved roughly 300 words of the computer memory required at the expense of the program elegance. The original function *TESSRB* was shortened. Only the part, which reads the already computed partial derivative coefficients and evaluates the

perturbations due to the tesseral harmonics is used. This modified routine is called *TESE*. Within computation in TESE function and some help routines, the routines for the direct disc access are used analogically to the *PDCTS* program. Some small corrections of the algorithm to overcome the dynamical range problem were made as well.

The computed coordinates of the satellite are stored in the two dimensional array *UU*, the corresponding values of the time are stored in the array *VV* and they are transferred to the next program segment. To reduce the computer memory required, no formated input/output statements are used in this segment. All the I/O procedures are carried out via a common block from the first to the third program segment or by means of the direct disc access. Program segment AIMF3 stores the computed values on the disc file *TRDATA* only. The number of the record of this file, on which the data are stored, is equal to the record number of the input data file CTIMES for this calculation. Program AIMFL may be started by the command:

RU,AIMFL,irec,kode

where

- irec ... is the record number of the CTIMES file to be used (default 1)
- kode ... indicates which perturbation must be included. (default 7... all)

2.3.2 Program SCHED

This is the help program to schedule the program AIMFL in the sequence, using the operator command:

RU,SCHED,ir,kode

Program AIMFL will be periodically scheduled with *RMPAR* parameters irec equal to ir, ir+1, ir+2, ... sequentially. When the final record of the CTIMES file is found, this sequence is terminated.

2.3.3 Program ATS (Along Track Shift)

See Section 4.1.

2.4 Group 4 - Programs for input & output data file manipulation

2.4.1 Program ORBEL

Program for manual interactive input of the orbital elements of the satellites. The data are stored on the help file *ELWORK* in the format acceptable for the *MYORB* routine in all the prediction programs. The key word for the mean elements is supposed to be in the form: 2 4 6 8 12. If some of the elements required by the *ORBEL* program are not supported, they must be entered as zeroes. The program may be started by the command:

RU,ORBEL

Then, the full satellite identification number, the epoch, the argument of perigee, ... are to be entered. Entering the data into the file ELWORK by the ORBEL program, the file must be listed and the data checked. Then, the file ELWORK must be copied to the corresponding EL++++ file.

2.4.2 Program CORCT

Checks the data set with the orbital elements received from SAO via Telex using the four letter checkword [6]. The incorrect lines are printed out. The missing one or non trustful digits or signs may be replaced by "*" by the interactive editor program. In the next program CORCT pass, the correct form of the line will be found and printed out. Up to 4 digits may be corrected in one line, all the combinations of characters corresponding to the given checkword (usually only one) are printed out.

2.4.3 Program TSHAR

It reformats the tesseral harmonics coefficients sets from the file *DATA3* and stores them on the appropriate place in the second block of the DDA area. (see fig. 2) In fact, the *DATA3* file contains many special tesseral harmonics coefficient sets, but only the sets for standard satellites and the Standard Earth III are used, the remaining being ignored. The program was arranged for input from the standard input device (lu=5). If input from the disc file is required, the program must be executed in the batch processing mode with the appropriate starting file.

2.4.4 Program TESFL

This is an interactive program for creating of the reduced set of tesseral harmonic coefficients from the set placed on the sixth position in the second block of the DDA area (default, the Smithsonian Standard Earth III set). The results are stored on the corresponding position in the DDA area according to the satellite identification number. Optionally, the convergence criteria may be modified, too. The program may be started by the command:

RU,TESFL

Then, the identification number of the satellite and the convergence criteria are entered from the console. Then the pairs of the subscripts of the

tesseral harmonics coefficients to be included in the reduced set may be entered. The subscript pair 0,0 terminates the interactive procedure, the reduced set is stored on the DDA area.

2.4.5 Program STATN

This is the interactive program for the station coordinates files manipulation. The program creates the file ST####, where #### is the SAO station id. number. During the program execution the station id. number, name and cartesian coordinates are entered. Then the geographic latitude, longitude and the distance from the earth centre is computed and stored on the file. Running the program :

```
RU,STATN
```

The station files are created automatically. If the corresponding file already exists, it can be or modified or purged by this program.

2.4.6 Program INFOR

This is the help program, which informs the computer operator about the program name, segment number, satellite id.number and the program phase just being executed in *CPU*. As mentioned above, the satellite position prediction software is a rather difficult system of several segmented programs, some of them having no output or control message on the console for relatively long time (up to 5 hours for the PDCTS program). That is why, a simple method of display of the basical information about the program just running was introduced. The *S* register of the computer is used for this purpose. Its value may be set by the programs using the system function *ISSR* call. Independently, the value of the *S* register may be read into another program. Thus, each noninteractive program, which is executed routinely for the prediction procedure, sets on some value on the S register. The meaning of the bits in the register:

```
bits    0-5  ... program computing phase
        6-8  ... program segment number
        9-11 ... satellite id.number
       12-14 ... program name identifier
```

The program may be started by the command :

```
RU,INFOR
```

The program writes on the operator console name of the program, segment and satellite number and program computing phase just executed.

2.5 Preparing the Prediction

1. All the files required must be created. For this purpose the transfer file #FILES may be used.

2. The station coordinate file must be created by the STATN program.

3. The tesseral harmonics coefficients sets must be placed into the DDA area by the program TSHAR.

4. The satellite orbital elements must be entered to the file ELWORK. This may be done interactively by the program ORBEL or by reading of the telex tape with the elements by the program TLXR and by editing of the resulting file.

5. The elements on the file ELWORK must be checked simply by visual comparison to the original telex print-out or using the CORCT program.

6. The file ELWORK must be copied to the appropriated EL++++ file.

7. If another satellite is to be dealt with, go to item 4.

8. The program SEEKC must be started consequently for each satellite. Then,the program ARANG and program CULM must be executed.

9. Program PDCTS must be started consequently for each satellite.

10. Program SCHED must be started to schedule repeatedly the program AIMFL.

On the end of this procedure, the user gets in the file TTABLE table of the satellite passes over given station, period of time and satellites and in the file TRDATA the table of universal times and positions of these satellites in the x,y,z coordinate system. It is obvious, that for routine prediction on the station, the sequence listed above is started once a week from the item no. 4. For automatical scheduling of this sequence items 8-10 the transfer file may be used.

2.6 Prediction procedure computer requirements

From the point of view of computer memory and computing time requirements, only the programs PDCTS and AIMFL are of interest. Program PDCTS requires roughly 15k words of computer internal memory . The time required for computation strongly depends on the tesseral harmonics coefficients set used. For LAGEOS satellite the program requires roughly 0.5 hour of computing time, for GEOS-C satellite roughly 5 hours. Running the PDCTS program for all the standard satellites takes about 12 hours of computing time.

The program AIMFL requires 16k words of computer internal memory. The

computing time required for 1 satellite pass is 100 seconds. The time consumption of all the other programs is of order of seconds or minutes of time. Thus, the off-line prediction procedure for one station, 5 standard satellites and one week period of time, takes about 14 to 16 hours of computing time.

2.6.1 Computer operating systems

Generally, the HP2100S computer is used for two rather different jobs - for satellite position prediction calculation, and for on-line laser radar control. The operating system requirements for these two jobs are quite different as well. The former needs as large computer memory as possible, the latter many special input/output facilities, foreground program availability, etc. To fulfill these requirements, two different operating systems were generated, they are denoted as "A" and "C" system for calculation and control, respectively. In practice the system "A" is used for the program AIMFL run, only. All the other programs, including the PDCTS program, may be executed in the "C" operating system.

The systems may be exchanged by the SWTCH system program. No disc cartridge exchange is required. The "A" system is routinely loaded only once a week, to compute the prediction for the next 7 days. The most time consuming program - PDCTS may be executed in the system "C" within the time gaps between the satellite laser ranging passes. This way, the prediction procedure for one week may be completed within one day (24 hours) on the station computer without affecting the plan of observations.

3 Control Software Package

This control software package was developed for the computer hardware described in section 1.1 and the laser radar hardware consisting of :

1. Laser clock: a multipurpose device which serves as a time central for the whole laser radar. The data to and from the laser clock are transferred via a HP-IB channel (IEEE-488 or IMS-2 standard).

2. Ranging counter: the HP5360 counter with time interval measurement unit HP5379A is used. The measured data are transferred from the counter to the computer via a special interface in the grammer device.

3. Time gate: resolution 100 ns, the window width and the delay time are programmable via a standard 8-bit communication interface in the Multiprogrammer device.

4. Step Motor Control Unit: (SMCU) the control device generating the sequences of pulses to control the step motor power supplies for continuous satellite tracking.

The SMCU is connected to the computer via a HP-IB channel. The data transfer

to and from the devices via a HP-IB is ordered by means of ordinary input/output commands to transfer the ASCII characters (formated READ/WRITE commands in the FORTRAN IV language). For data transfers via the Multiprogrammer device the special routines were written: the *GATE, *BUFR*, RESET ... routines for time gate device programming and data collecting from the range counter, respectively. The function of these routines, using the description comments on the heads of them, is quite self-explaining.

3.1 Programs for system calibration and check

3.1.1 Program TIME

It is an interactive program for programming and set-on of the laser clock unit. There is a closed loop inside this program, the user can choose one of the seven program options:

1. Start the laser clock. The starting values (hour,minute) are entered from the console. The clock is started by the next coincidence of the "1 min" and "1 sec" pulse from the station master clock.

2. Reading and display of the current value of the time from the laser clock and the computer internal clock.

3. Synchronising of the computer internal clock to the laser clock. The former is synchronized by help of the system command TI executed through the function MESSS. The computer internal clock may be synchronized with accuracy bether then 10 ms.

4. Programming of the repetition rate for the laser trigger, the laser period 1 to 9 seconds may be set on.

5. Laser control - by the following characters ASCII :

   - A ... start of the HV power supplies,
   - B ... stop of the power supplies,
   - T ... single shot trigger of the laser,
   - Q ... start of the lasing sequence with period programmed on the item 4,
   - R ... stop of the lasing sequence.

6. The laser clock synchronization check. The epoch of arriving of the one second pulse from the cesium master clock is determined. The fractional part of second of this epoch determines the time shift between the cs.clock and the laser clock.

7. The end of the program.

3.1.2 Program KALIB

It is the program for the fixed target ranging calibration procedure. It can be started by the command:

RU,KALIB,iopt,lu

where

iopt ... option code 0...calibration only(default)
1...precalibration
2...postcalibration
lu ... number of the lu output device for the result list. (default 1)

The program starts the laser power supplies, sets the time gate and completes the total of 23 laser rangings to the target. The ranging results, which differs more than ±20 ns from the usual value are rejected, the error message being transferred to the operator console. The mean range value and the RMS are calculated and displayed. If the option code equals 1 or 2, the values of the temperature, air humidity and pressure must be entered. The mean range value and the atmospheric data are then stored on the record no. 2 of the file RANGE or the file RANG** for pre- and post-calibration, respectively.

3.1.3 Program MOUNT

It is an interactive program for the mount positioning, alignment and pointing accuracy check. The mount motion is controlled through the step motor control unit (see 9). This device generates the sequence of pulses for the step motors power stages. The frequency of stepping is obtained by the digital frequency division of the 1 MHz frequency from the high stability quartz oscillator. The division ratio is programmable, together with some other functions of SMCU, via a HP-IB channel. There are two counters within the SMCU, which count the total number of steps already generated. The stepping frequency of the both drivers is set on each one second.

There exists a set of several routines, which are available for the mount control. Except for the direct SMCU programming, the algorithm is identical for both the step drivers. All the data transfers within this set of control routines is carried out via one two dimensional, double precision array *DM*.

- Subroutine STEPS: the principal routine, which computes the division ratio to be programmed for the mount setting from the initial to the final position. Inside this routine the stepping frequency is evaluated in such a way, that it is from the interval between the minimal and the maximal frequency, the resonant frequencies of the

mechanics and the maximal acceleration allowed is taken into account. The algorithm is resistant to the rounding errors. The meaning of the variables used is described in the head of this routine. (This description is valid for the input/output values only. Within the computation, the variables may be used to other purpose, too!)

- Function MAXA: its value is equal to the maximal acceleration allowed for current frequency of stepping. In the first call, the value of MAXA is equal to 10 to guarantee proper start of the drivers.

- Subroutine REZON: it checks the input stepping frequency with respect to the the resonant frequencies. If the input frequency falls within the resonant region, the output value of frequency will be set lower, then is the lower limit of the resonant frequency region.

- Subroutine SECND: it guarantees the time synchronization of the SMCU programming with respect to the synchro pulses from the laser clock. When this routine is called, it periodically tests if the new synchro pulse was detected in the SMCU. As soon as it occurs, the test is terminated and the control returns to the calling program.

- Subroutine CTI: reads the values of the built-in step counters. Calling the CTI routine, the caller gets the values of the step counters, which were achieved in the moment of arriving of the last synchro pulse.

- Subroutine DRIVE: guarantees the communication with the SMCU. It has 3 separate parts, which are executed according to the value of the input code: the initial reset of the SMCU, the start of stepping and setting the stepping frequency, stop of the stepping.

The routines STEPS and DRIVE are called each second to compute and to set on the division ratios for the SMCU. Consequently, the other help routines CTI, MAXA, SECND, REZON are called as well.

- Subroutine POSIT: is the main routine, which calls periodically the STEPS routine. This routine guarantees the positioning of the mount from one position to another one at the shortest time possible. The input values are the relative shift angles for both axes (in degrees) only. The positioning procedure from one position to another one is carried out in three phases:

  * first phase: the highest rate acceleration up to the maximal speed, motion with the maximal speed for N seconds until one half of the motion required is completed.

  * second phase: motion with the maximal speed for the next N seconds.

* third phase: the highest rate decreasing speed down to minimal start-stop frequency, positioning up to the desired final position.

Then, the total number of steps carried out in each coordinate (the content of the step counters) is compared to the value required according to the input. The difference is compensated by the "single step mode" stepping. Running this program:

RU,MOUNT RU,MOUNT

The program has several user selectable options: definitions of the starting and the final position of the mount and a direction of revolving of the mount round the azimuthal axis (not to twist the cables and pipes inside the mount too much). The fixed, preprogrammed, initial/final coordinates of the position reference target, ranging target and the reference stars position may be used or any other coordinates may be entered manually from the console. The option "aim to the reference star" may be used for absolute accuracy check.

3.1.4 Program FVT (Field of View Test

This is the program for the receiver efficiency and the field of view test. The program controls the mount in such a way that it scans a small area round the initial position. Simultaneously, the receiver noise frequency is recorded. The scanning step is 0.01 degree in the azimuth direction and 0.075 degree in elevation. The current value of the noise frequency is displayed on the console. Scanning the area, the results are printed out together with simple graphs of dependence of the receiver noise on the coordinate in the form of densitographs. As a reference light source the POLAR star may be used. Measuring on the 1PE detection level, the receiver channel overall efficiency may be determined from the difference between the maximal and minimal values of the noise. Running this program:

RU,FVT

3.2 Satellite tracking and ranging programs

3.2.1 Program SATEL

This program controls the laser radar system during the satellite ranging procedure. It is composed of the short formal main program and three program segments for the prepass radar preparation, ranging and postpass radar control. The data for the satellite automatical tracking are evaluated on-line in this program. As the input data, the results of the AIMFL program are used. The computed positions of the satellite in the x,y,z coordinate system are approximated by the Chebychevian polynomial [3]. For computing of the position of the satellite at any time during the pass, the corresponding value of this polynomial is evaluated for the time required taking into account the time shift, computed by the program ATS. The resulting position in the x,y,z

coordinates is then transformed to the azimuth, elevation and range. The routines INTER, AZALT, SIDTM, JNTMT and DASIN are responsible for this computation. The INTER routine has two program options, controlled by the KODE variable. For KODE = 1, the actual beginning of the pass and the associated values of azimuth and elevation are computed. The time is rounded to two tenths of seconds. This option is used in the prepass set procedure in the first program segment. For kode = 2, the azimuth, elevation and range of the satellite are evaluated for the given time. This option is used in the on-line satellite tracking procedure. The program SATEL may be started by the command:

RU,SATEL

Then, the sequence number of the satellite pass must be entered. The tracking input data, computed by the AIMFL program, are read from the corresponding record of the TRDATA file. The station coordinates are read from the file. Then, the actual beginning of the pass is computed together with the starting position for the tracking. The time gate window width and the initial value of the along track tracking delay must be entered.

The proper function of the laser clock unit is then tested, the availability of the help program TRACK is checked. The mount is set to the initial position with help of the POSIT routine. Each 10 seconds, the message to the operator console about the time left to the start of the pass is sent. Twenty seconds before the start, the first program segment is completed and the segment SATE2 is loaded and started.

The segment SATE2 locks the computer to be unaccessible for other foreground programs. The laser power supplies are switched on, the SMCU step counters are reset. At the moment of beginning of the pass, the closed program loop of on-line control is started. The loop runs with the period of lasing - once in 4 seconds. Within it, the laser is triggered, the help program TRACK is scheduled, the ranging results of the foregoing cycle are stored on the disc file RANGE, the value of the step counters is read and compared to the value required and the optional correction of the stepping is introduced. The ranging results: the epoch of the laser fire and the propagation time are read from the laser clock and the range-counter, the range to the satellite is evaluated for the epoch of lasing in the INTER routine, the difference (prediction - measurement) is displayed on the console. Then, the positions of the satellite within the next 4 seconds are evaluated and the corresponding control commands for SMCU are created, the range counter is reset. Then, the loop is scheduled again. This is repeated until the satellite does not decrease below the observation limit(28 degrees).

The programming of the SMCU in this segment rather differs from that in the MOUNT program or the segment SATE1. As the period of ranging (4 sec.) differs from the period of the SMCU control, special algorithm was to be applied. The SMCU is not programmed directly from the program segment SATE2, but via a small help program TRACK. This program transfers the already created control commands to the SMCU only. Program TRACK is loaded as a foreground program, thus, it can be present in the computer internal memory together with the background program SATEL/SATE2. The control commands for the SMCU are

created in the SATE2 program segment for the next four one-second intervals. They are transferred to the TRACK program via a system COMMON block area. The TRACK program is scheduled to run each one second with the highest priority.

Due to the mechanical construction of the mount, there exists a small near-zenith area of the sky, to which the mount can not be aimed. Subroutine CULM guarantees, that the step drivers will not try to set the mount to this position, but to the nearest available one. The execution of the program loop in the SATE2 program segment may be broken manually by the operator command:

BR,SATEL.

To compensate the along track prediction error, the time shift may be introduced into the tracking procedure. The S register switches are used for this purpose. Setting on the bit 0 or 1, the tracking delay will be increased or decreased at one elementary time interval, respectively. The elementary tracking time interval is evaluated with respect to the satellite angular velocity and the laser beam divergence used.

When the program loop of the SATE2 program segment is terminated, the laser HV power supplies are switched off, the motion of the mount is smoothly stopped, the next program segment is loaded. Program segment SATE3 stores the ranging results on the disc file RANG**, where ** is two digits sequential number of the file, generated automatically. Then, the mount is positioned to its initial position and the program is terminated.

4 Programs for Postpass Ranging Data Handling and Analysis

4.1 Program ATS (Along Track Shift)

This program evaluates the along track time shift between the satellite position predicted and the actual measured value. This parameter may be used in the next automatical tracking of the same satellite. The formula derived in [4] is used for the time shift calculation. As the input, two satellite rangings, together with the corresponding epochs are used. The predicted range and the radial velocity of the satellite for these two epochs is then evaluated by the algorithm identical to that in the satellite tracking program. The routines INTER, AZALT, etc. are used ,as well. The program may be started by the command:

RU, ATS

The identification number of the satellite pass must be entered together with the two epochs and measured distances of the satellite. To increase the accuracy of evaluation, it is recommended to use the points of roughly the same satellite range, one before and the other after the culmination. The computed time shift in seconds is displayed on the operator console.

4. Program RDF (Ranging Data Fit)

This is an interactive program for the ranging data handling, noise rejection, ranging accuracy check, etc. The program fits the measured ranging data with a polynomial of chosen degree (1 to 8) and prints out the difference between the measurement and the fitting polynomial. Optionally, the "range to prediction residuals" may be fitted. This option is highly appreciated in the case of very low signal to noise ratio. For example, using the 3rd order polynomial, the residuals for 6 minutes Ge-A pass may be fit up to about +/-30 nsec. This way, the noise points may be rejected effectively. The total number of the rangings and the standard deviation is printed out as well. The program may be started by the command:

RU,RDF,lu,iout

where

lu ... is the no. of terminal (default console)
iout... is the no. of the output device(default lu)

4.3 Program RDT (Ranging Data Transmission)

This is the program for measured ranging data transmission in the SAO standard quick-look data format. The program reformats the ranging data files into the file TELEX. This file may be then punched out in the telex code by the TLXW program. Starting the program by the command "RU,RDT", the user may choose if the quick-look only or the whole output file is to be created. (In the quick-look format only 15 measurements are selected from the full ranging data set.) Then the number of the ranging data file must be entered. The no. 00 terminates the program.

4.4 Program TLXW (TeLeX Write)

This is the program for punching of the paper tape in the standard telex code. It may be started by the command:

RU,TLXW

The user must answer the name of the file, from which the data for punching are to be read.

5 References

[1] Smithsonian Standard Earth, SAO special report no.200

[2] B.Ambrosius, H. Piersma, K. F. Wakker, Description of the Aimlaser

Satellite Orbit Prediction Program and its Implementation on the Delft University IBM Computer, Report LR-218, Delft, May 1976

[3] P. Wilson: private communication

[4] R. Neubert, I. Prochazka, A Simplified Satellite Orbit Prediction Program FJFI-CVUT, Prague 1979, Report no. 79/83

[5] Arnold, Methoden der Satellitengeodaesie, Berlin, Akademie-Verlag, 1978

[6] M. R. Pearlman, Memorandum, Updating Ephemerides Software for Laser Ranging, SAO, 1 June 1979

[7] W. M. Kaula, Analysis of Gravitational and Geometrical Aspects of Geodetics Utilization of Satellites, Geophysical Journal, Vol. 5, 1961

[8] I. Prochazka, P. Sobek: Step Motor Control Unit for Continuous Satellite Tracking, FJFI Report, no. 81/130, Prague, 1981

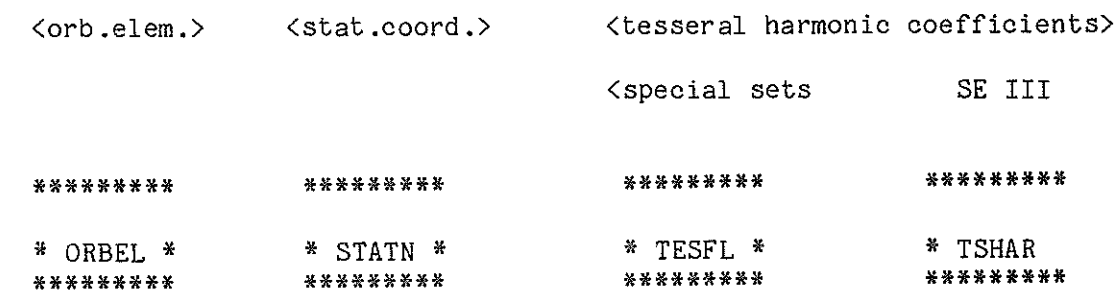
I. Appendix A

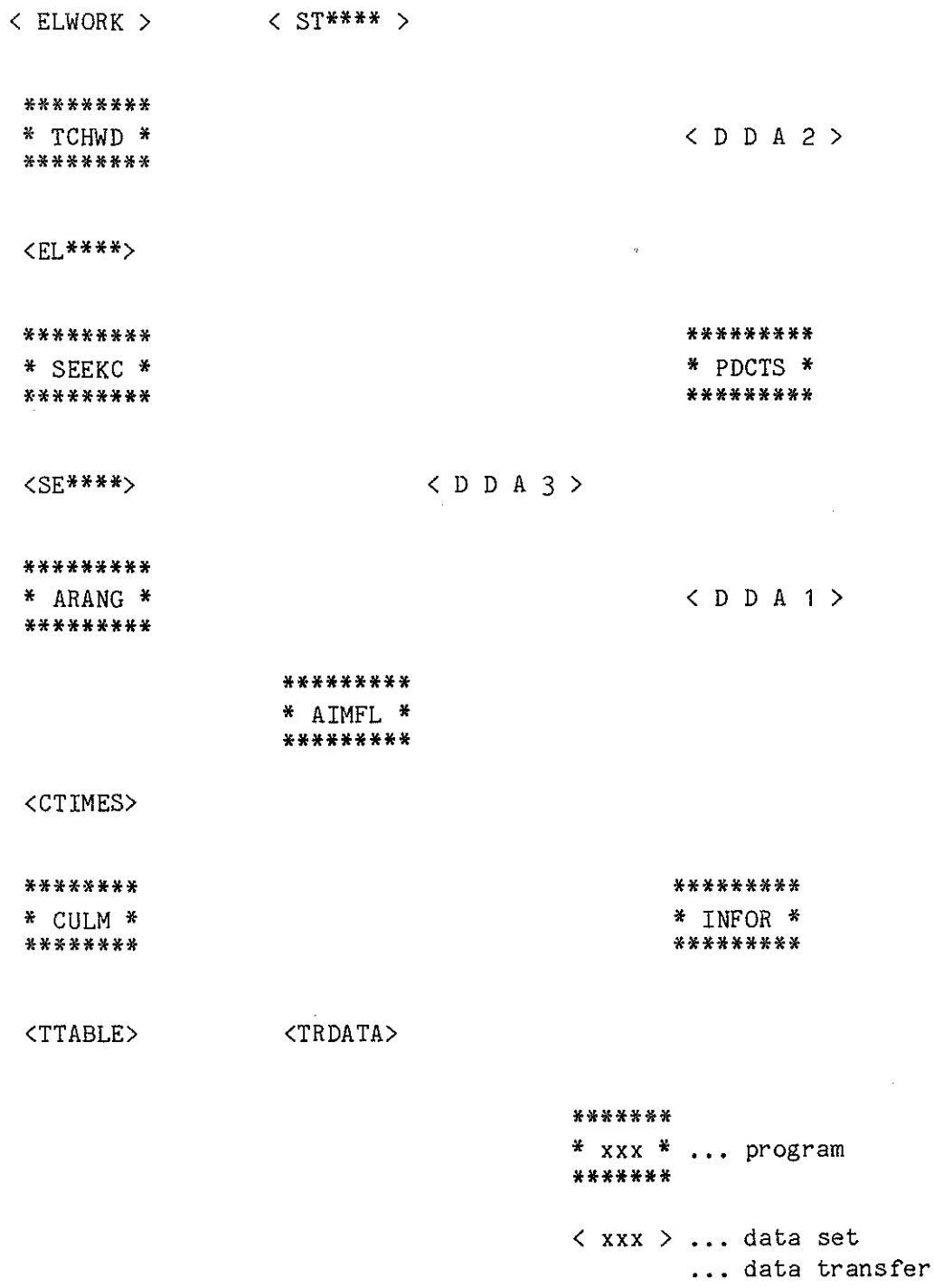
Dedicated disc area (DDA) structure :

track	beg.sector	satellite	content
0- 1	0	all	CLMP matrix.
2- 3	0	Beacon c	slowly
4- 5	0	Geos a	varying
6- 7	0	Starlette	coeff.
8- 9	0	Geoc c	for t.h.
10-11	0	Lageos	perturb.
12-13	0	spare	calcul.
14	1/10	Beacon c	A/ITESS
15	1/10	Geos a	A/ITESS
16	1/10	Starlette	A/ITESS
17	1/10	Geos c	A/ITESS
18	1/10	Lageos	A/ITESS
19	1/10	spare	A/ITESS
20	not used		

fig.2

SATELLITE POSITION PREDICTION SOFTWARE SCHEME




< ELWORK >
< ST**** >
****
* TCHWD *
****
< D D A 2 >
<EL****>
****
* SEEKC *
****
****
* PDCTS *
****
<SE****>
< D D A 3 >
****
* ARANG *
****
< D D A 1 >
****
* AIMFL *
****
<CTIMES>
****
* CULM *
****
****
* INFOR *
****
<TTABLE>
<TRDATA>
****
* xxx * ... program
****
< xxx > ... data set
... data transfer

Multiprocessor-Based Mobile System Software Design

by

K.H. Otten
Working Group for Satellite Geodesy
Kootwijk
Delft University of Technology
Delft, The Netherlands.

ABSTRACT

Considering that the satellite tracking activity can be divided into three major tasks, a choice of three separately operating computer systems has been made. The predictions of the satellite points as well as the corrections are supplied to the servo system of the telescope by the predictor micro processor. The observation data is collected and formatted by a second micro processor and transmitted to a main processor. The main processor hosts a monitor program during observation activity. The monitor program takes care of data transmission and schedules real-time tasks. Programs for initial satellite predictions and observation data screening are included in the main processor's application software package. A brief description of the system hardware is given to clarify the interaction with the computer configuration.

1 Introduction

The laser ranging system described in this paper is a mobile one and therefore can be brought into action in remote locations. This requires it to be almost completely self-supporting for a certain period of time. Concerning the observation activities, this means that the system must be equipped with all facilities to do a maximum number of valuable measurements (on even low satellites like STARLETTE) within a short campaign and with few contacts to the outside world as possible. To meet these requirements a considerable amount of time has been and still will be spent on investigation and development. The hardware system has been designed in more or less conveniently arranged segments with only a few interconnections and a great deal of independence, which will be helpful in locating errors during malfunctioning. For low satellite tracking (STARLETTE), a method has been developed to calculate orbital elements, valid for a whole week of observations and needing no more than seven numbers of data transmission for one pass. Detectors, indicating the rough position of the satellite in the telescope's field of view and the range-gate window, will aid in finding and tracking the satellite.

2 System Description and Functioning

In general, the ranging system consists of four compound units, namely:

- the telescope with the optical system and the laser,
- the mount positioner and laser control unit,
- the detection system including the station UTC-clock,
- the computer configuration with its peripherals.

The mount control and detection units will be described briefly, only to clarify the interaction with the computer configuration, which is the main subject of this paper.

2.1 Mount Positioner and Laser Control Unit

The mount positioner and laser control unit has the responsibility to direct the telescope to the object to be observed and to enable or disable the laser. At this point it may be mentioned, that the laser is continuously triggered at a 10Hz rate during ranging operation.

-Servo system.

The telescope's axes are driven by a servo system using DC-servo motors. Speed and orientation information for both axes are received from the prediction processor at a rate of 10Hz. Every tenth of a second, the actual and predicted positions are compared and differences will be automatically

corrected for. This feature makes special measures unnecessary for the observation of zenith passes. When the speed predictions are set equal to zero, the system chooses its own speed and stops at the desired position.

-Shutter.

A shutter is located in the oscillator cavity to disable the laser, without influencing the operation of the laser itself.

2.2 Detection System

The detection and the optical system are the most vital parts of the ranging system: they offer the means to find and track the satellite, to improve the measurements and to meet some safety requirements (Visser,1981).

-Position, time and meteo-data keeping.

For completeness, it must be mentioned that firing time (UTC), meteorological data and the actual position of the telescope are recorded. Some of the information will also be used during real-time operation of the computer system.

-Simultaneous calibration timer.

A fraction of the laser beam will be reflected after passing the start detector and is directed to the main photo-multiplier (PMT), which in turn stops the simultaneous calibration timer. This gives an indication of the stability of the basic detection system.

-Time interval counter.

The received signal will be attenuated to the single photo electron level by reflecting a part of the beam to the quadrant detector and transmitting the remainder to the main PMT which stops the time interval counter. This counter will deliver the round travel times to the satellite.

-Quadrant-detector timers.

The reflected part of the received signal, described in the previous section, is split up into four equally sized segments and directed to separate PMT's, which stop the quadrant detector timers. These time intervals are related to the start of the window and are expected to give an idea, in which quadrant of the field of view and where in the window the satellite might be found.

-Multiple-stop timer.

The multiple-stop timer consists of four, serially arranged timers,

linked to the main PMT. This configuration allows the recording of four events, relative to the beginning of the window. Speeding up of radial searching of the satellite's position or improvement of the remaining tracking parameters after the satellite has been found, is possible.

-Safety detectors.

An aircraft and a sun detector will be installed to meet government safety regulations and to protect sensitive system components. There will be no discontinuity in tracking when the telescope is passing the sun (although no observations are made).

2.3 Computer Configuration

In analyzing the tracking activity, which is the most critical task the computer system has to support, it can be easily seen that division into three more or less logical parts of cooperating functions is possible:

- final prediction and correction of direction and speed of the telescope,
- collection and formatting of observation data,
- monitoring and data recording.

Considering this division, a choice of three separately operating computer systems, each of them executing one task, is desirable. Since the prediction and formatting sections are quite unlikely subject to changes and not too complex, they will be installed in PROM and running on relative small 8-bit micro-processors: the predictor and the formatter. Unfortunately, micro's are slow in handling floating point data. Therefore a fast floating point processor has to be added to the predictor to handle this type of computations. The two micro's will be slaved to a 16-bit main computer system. The main processor will support extensive software for program development and data-handling. Knowing that computers are overloaded within a short period of time after installation, a choice of the main system will be made very carefully. To enable data communication, the predictor and the formatter are connected to each other by 8-bit bi-directional parallel data lines. The predictor is linked to the mount positioner and the formatter to the detection system. The main processor will have a HP-IB controller, used for data transmission to and from the slave processors. The main processor may be extended by a number of processors, if necessary. As peripheral devices, there will be two diskette units, a graphics display console and an optional KSR-terminal with an acoustic coupler, all of them interfaced to the main system. The UTC-time source will be used for epoch timing and synchronization of computer operation by means of a 10Hz interrupt supply. Both slave processors, in cooperation with the main processor, will execute test routines during system start-up/reset to guarantee the functioning of data communication and the processors themselves. Also the firm-ware will include defined checkpoints, which can be initialized during test mode of operation.

At these points the micro processors will transmit intermediate computational results and memory dump data to th main processor.

-Main processor.

In the computer configuration, the main processor must have the property of a general purpose development system with multi-programming/tasking capabilities, offering all the facilities for real-time and off-line operation. In the basic concept, the main processor will host the monitor program only, which initializes the slave processors, handles their status and message information, supplies the prediction processor with the predictions and corrections, receives the observation data from the formatter and records it on a diskette. This leaves most of the capacity of the main processor for real-time programs, which will be used to improve the ranging system's adjustments. Years of satellite ranging have shown, that supporting software is changed very frequently. Because the monitor is the only more or less time-critical program in the main processor, replacing and adding real-time software can easily be done without disturbing the tracking operation itself. In times, when the main processor is not involved in satellite ranging, it will be used to run application programs for astro-positioning, initial prediction calculation, data validation and general utilities.

-Prediction processor.

The prediction processor provides the mount positioner at 10Hz frequency with the desired direction of the telescope (azimuth, elevation), speed of the two axes, range-gate delay, window and laser control information. Because this action is very time-critical, it is controlled by an interrupt routine, which is initiated by a 10Hz signal from the UTC-system clock. The interrupt routine receives the UTC-time and status information from the formatter. Also it transmits the prediction status to the formatter. The UTC-time is used for internal time-keeping and -verification. Two modes of pointing are supported by the firm-ware, i.e. tracking and positioning. In positioning mode, the final predictions for the mount positioner will not include speed information. This allows pointing to a fixed target for a certain period of time. The initiation of the telescope's positioning can be done automatically by time values included in the predictions or manually by operator intervention. When the processor is running in tracking mode, the interrupt routine will be provided with updated pointing information before an interrupt occurs, whereas in positioning mode it receives one prediction for every predicted telescope direction only. The predictor also has the option to enable or disable the laser during ranging operation by using the information from the formatter, which indicates whether the telescope's orientation is within limits of the predictions or not. The laser will be enabled in positioning mode, when the telescope is pointed at the desired target and in tracking mode, when the minimum elevation is exceeded. After start-up, the predictor is idle, with only the interrupt routine working, waiting for initial predictions. When an input-request is received from the main processor, it reads a complete set of topocentric predictions for one observation activity, i.e. tracking one satellite pass or pointing to one or more targets. Tracking predictions are divided into groups of four satellite points, enclosing three subsequent intervals. For every group, a right-handed rectangular topocentric coordinate

system is defined with the x-axis pointing to the first satellite position and the y-axis being in the plane of the horizon. The remaining satellite positions are transformed into this system and the mean direction of the satellite track is determined. Using the transformed coordinates, the coefficients of a third degree Newton collocation polynomial are calculated. The elements of the rotation matrix (azimuth and elevation), the track direction and the polynomial coefficients replace the original predictions. This method allows real-time interpolation of the direction at tenth of a second intervals with no more than 0.001 degree deviation relative to the initially predicted track. For the range-gate delay, the same type of polynomial is used, resulting in a deviation of no more than 5 ns. The accuracy of the interpolation does not really require a third degree polynomial. However with a lower degree collocation polynomial the available RAM would have been exceeded (more groups per satellite pass). The predictor also applies along- and across-track, wire, delay and window corrections, which are received from the main processor. When the observation activity has ended, the system is idle again.

-Formatter processor.

Like in the prediction processor, an interrupt routine is handling the data communication to the hardware system. At a 10Hz rate, the detection system transmits the UTC-time, telescope direction, meteo-data and the various counter- and time-information to the formatter, even if there is no observation activity. The formatter compares the actual and the predicted direction of the telescope. The result of the comparison together with the time information is transmitted to the predictor. After the formatter has been initiated for an observation activity, relevant data is selected from the received information, formatted and collected in blocks of 1.6 second time periods and placed into an output buffer, to be transmitted to the monitor. To meet a delay in the response of the main processor, the buffer can accommodate several blocks.

3 Software Support

The software of the mobile ranging system must support all activities, which range from astro-positioning to quick-look selection. Careful segmentation and structuring is essential to facilitate future replacement and improvement of the programs. Interaction with the operator will be kept to a minimum, because the station crew may consist of less experienced personnel.

3.1 Astro-Positioning and Orientation

When installed on-site the first time or periodically, it will be necessary to determine the azimuth reference of the telescope and the latitude and the longitude of the ranging system. Preliminary values, obtained from e.g. compass and maps will be corrected iteratively using star observations.

-Star selection.

A catalogue, containing FK3 stars with FK4 accuracy, stored in a random-access file on a diskette will be searched for suitable stars. Depending on the type of parameter to be solved for and adjustment method to be used, the stars must satisfy some conditions for the time of observation, e.g. they must be near the prime vertical or the elongation and be the brightest stars in the near surrounding of the telescope's direction to ease the observation. The star coordinates will be transformed into the topocentric system and stored on a diskette in a manual or automatic positioning mode prediction file.

-Observation of stars.

During the observation of the stars, the ranging system is operating in the real-time mode. The star positions are transmitted to the prediction processor and pointing of the telescope to the stars is initiated automatically or manually by the operator. The observer at the telescope's eyepiece has the ability to correct the direction of the telescope by hand-paddle and to trigger the system by push-button. The observation data is recorded on a diskette.

-Parameter approximation.

The parameters are solved for separately in a non-linear least-square approximation. It might be necessary to repeat the whole cycle before the desired accuracy of the parameters is obtained.

3.2 Satellite Tracking

In this section the system software involved in activities necessary to enable satellite tracking in a remote location will be described. It will include software, which already has been developed and tested and software, that will be implemented after some experience has been gained during operation.

-Orbital elements.

Once a week, for every satellite and pass, a new set of osculating orbital elements is received via public telecommunication facilities and stored in the bubble memory of the KSR-terminal. This message is retransmitted to the prediction center for a transmission check and recorded on a diskette of the computer system, ready to be used by the prediction program.

-Satellite predictions.

The prediction program selects the orbital elements from the diskette, calculates the topocentric coordinates of the satellite positions for one pass and writes it onto an empty diskette. A fourth-order Cowell integration method is used to integrate the equations of motion in a gravity field comprising coefficients up to degree and order 4. The predictions are produced with a 5 or 30 seconds time-interval for STARLETTE or LAGEOS respectively,

rea y to be used for the real-time interpolation operation of the prediction processor (Vermaat,1981).

-Real-time observation scheme.

The real-time operation is initiated by the monitor program, when it transmits the predictions to the prediction processor. From this moment on all observation data is recorded on the same diskette the predictions are read from, except the information from the quadrant detector and the multiple-stop timer, which will be available in the main processor only. When after system calibration tracking has started, the data from the quadrant detector and the multiple-stop timer is displayed on the console to aid in finding the satellite. Corrections will be applied to the predictions, until sufficient returns have been received from the satellite. Subsequently a task is scheduled to select the returns. These observations will be used in a least-squares approximation of Keplerian orbital elements, to extrapolate the range-gate delay and to possibly correct the range-gate delay and window. The information from the quadrant detector will be helpful to correct the predictions in such a way that the satellite stays near the optical axis of the telescope. The real-time operation is terminated after the post-calibration has been completed and corrections have been saved. These corrections will be relayed to the operator as initial values to be used for the next pass.

-Data validation and quick-look selection.

In periods of less activity, a final screening of the calibration and ranging data will be separately performed, one pass at the time. The screening of the pre- and post-calibration data is done in a least-squares sense including testing procedures. For the screening of the satellite observations, a Keplerian model is used including the short-periodic J2-effect. An iterative method of statistical testing is applied, leading to converging solutions even in case of high noise rates (Vermaat,1978). A number of more or less uniformly distributed and reliable observations is selected for the quick-look data report to be transmitted to the prediction center (before the new orbital elements are received).

3.3 Utilities and Diagnostics

The ranging system's software will also include utilities for data-handling, as preparation of a ground-target prediction file and copy operations. A number of diagnostic programs will be available for trouble shooting in case of system malfunctioning.

4 References

Vermaat, E. Laser Data Preprocessing at the Kootwijk Observatory. Proceedings of the Third International Workshop on Laser Ranging Instrumentation. Lagonissi, May 1978.

Vermaat, E. On site Numerical Integration of the STARLETTE Orbit in a Taylored Force Field. Presented at the Fourth International Workshop on Laser Ranging Instrumentation. Austin, October 1981.

Visser, H. and F.W. Zeeman. Detection Package for the German/Dutch Mobile System. Presented at the Fourth International Workshop on Laser Ranging Instrumentation. Austin, October 1981.

Preliminary Data Handling at Borowiec

by

S. Schillak and E. Wnuk
Astronomical Observatory
Poznan University
Sloneczna Str. 36
Poznan, Poland

ABSTRACT

The results of the satellite laser ranging must be initially prepared for their further utilization. The main purpose of these computations is elimination of the erroneous points, determination of the standard deviation of residuals for the satellite pass, and initial corrections of the results. The presented method was prepared especially for calculating of the results of Borowiec laser system. The following conditions were taken into account in this method:

- small number of points per satellite pass,
- possibility of appearance of the great number of noise points,
- quick and simple calculations on small computer,
- good estimation of the standard deviation.

The method can be used also for estimation of the results from other satellite laser ranging stations.

1 Elimination of the Erroneous Points

For the epochs obtained from observations we calculate expected satellite ranges on the base of actual orbital elements. The satellite range is calculated with secular and long-period perturbations. The difference in range between the observation and ephemeris gives possible the rejection of very big errors, of the order of several kilometers and it is the first step for further computation. The main purpose is to bring to minimum the difference between ephemeris and observation. Ephemeris, in consequence of inaccurate orbital elements and not taken into account all factors perturbing the satellite motion from epoch elements to epoch of observation is burden with the systematical error in epoch ΔT and range ΔR (Fig.1).

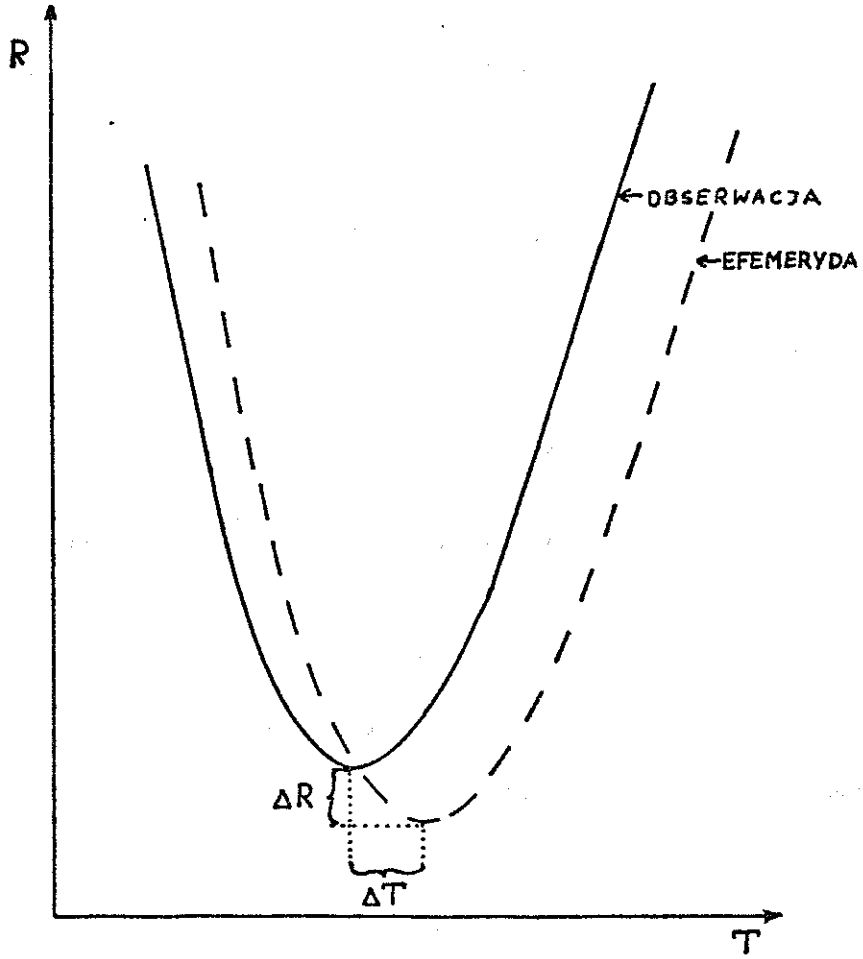


Figure 1: Systematic ephem. error in epoch (ΔT) and range (ΔR).

For the purpose of eliminating of this error, the method supposes step by step change of the epoch so far as both curves will be in agreement. For every step is calculated the sum of squares of the residuals from mean difference between observation and ephemeris. The epoch changes are performed with the 0.1 millisecond step in the direction to diminish the sum square. As final result we obtained the difference in the epochs between ephemeris and observation, the mean value of the difference in the range, it means systematical difference in the satellite range for corrected ephemeris epochs, and residuals from this mean for particular points. The continuous change of

these residuals depends on accuracy of the orbital elements and it comprises in the limit from several meters to several hundred meters (Fig.2).

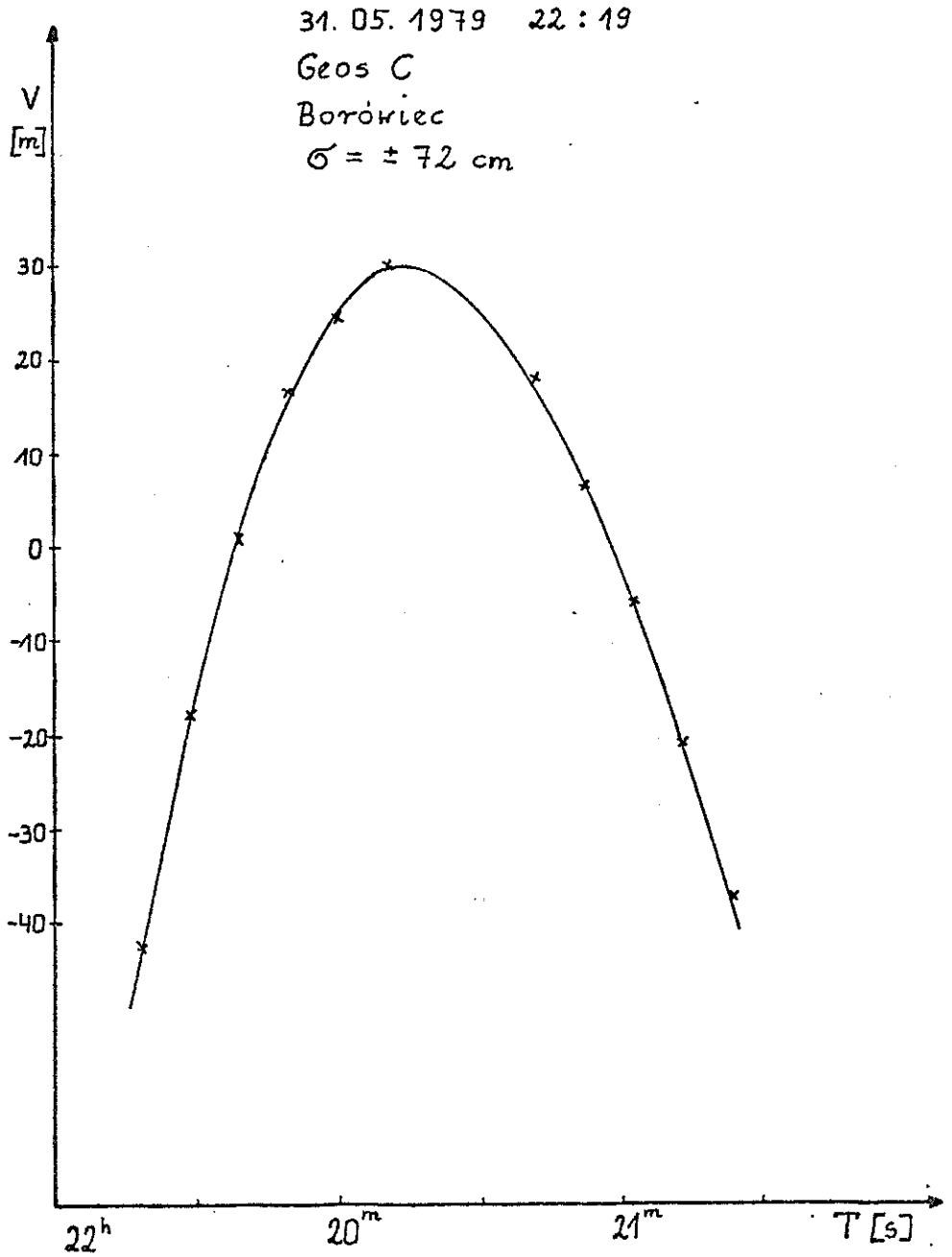


Figure 2: Residuals from the mean dif. observation-ephemeris.

The diagram of the residuals, or visual computer monitor control gives the possibility of easily founding erroneous points, even if errors are small, of the order of several meters, and their number is great (Figs.3,4,5).

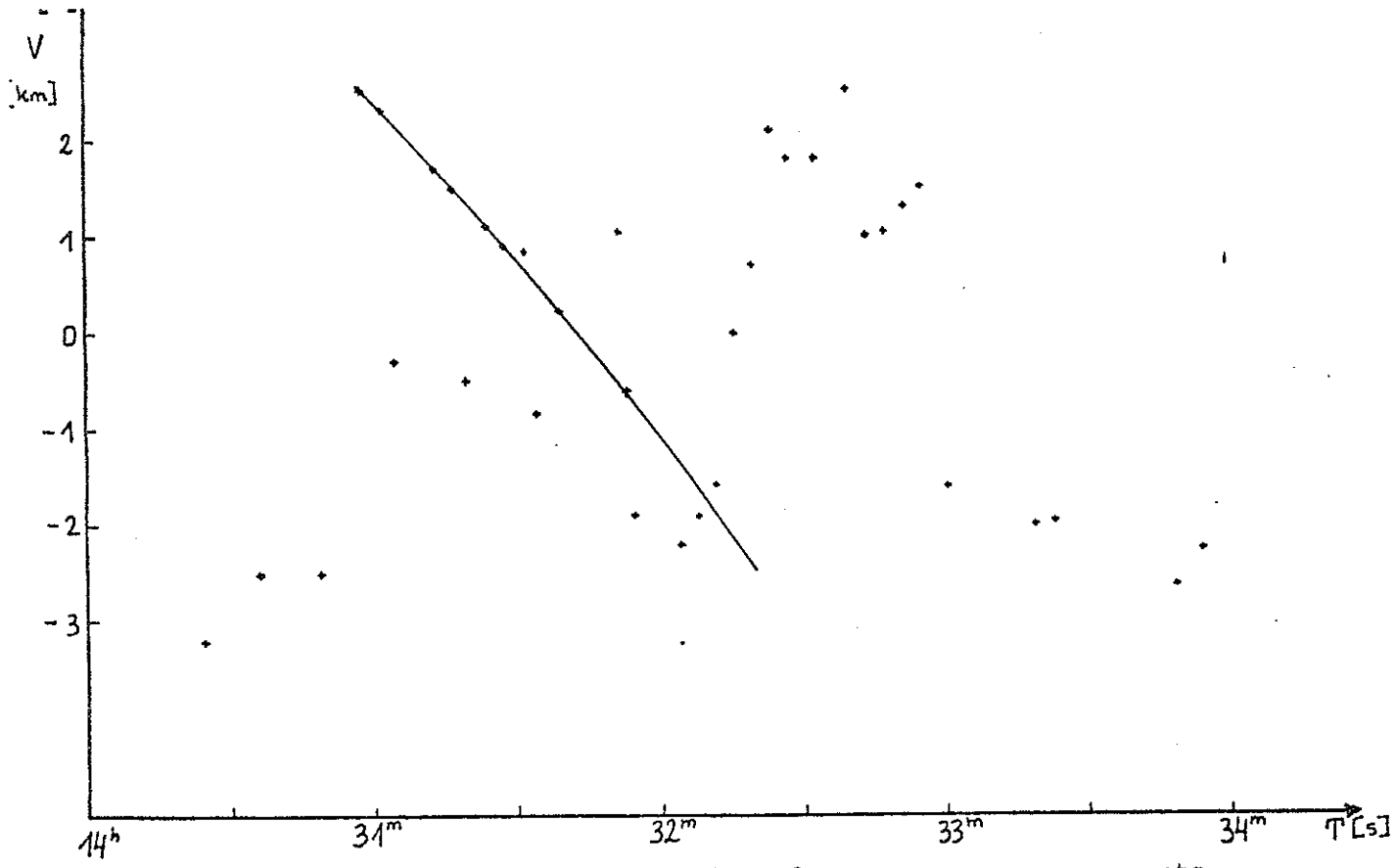


Figure 3: Determination of good points from erroneous measurements

2 Estimation of the Ranging Precision

The important problem for estimation of the observation is possible accurate determination of the standard deviation as result of random dispersion of the observing points for particular pass. For this purpose we used the smoothing of the differences between observation and ephemeris by method of least squares with Tchebychew polynomials.

$$\sigma^2 = \Sigma\, v_p^2 / (n-p-1)$$

The essential element of this method is the determination of the polynomial degree. For this purpose we used statistical method based on Fisher's random variable, which gives the possibility of determination of the polynomial degree for condition when dispersion of the results achieve random character (Schillak & Wnuk, 1977). Fisher's variable we calculate for polynomial degree 'p' by the formula:

$$F = \Sigma\, v_{p-1}^2 - \Sigma\, v_p^2 / \sigma_p^2$$

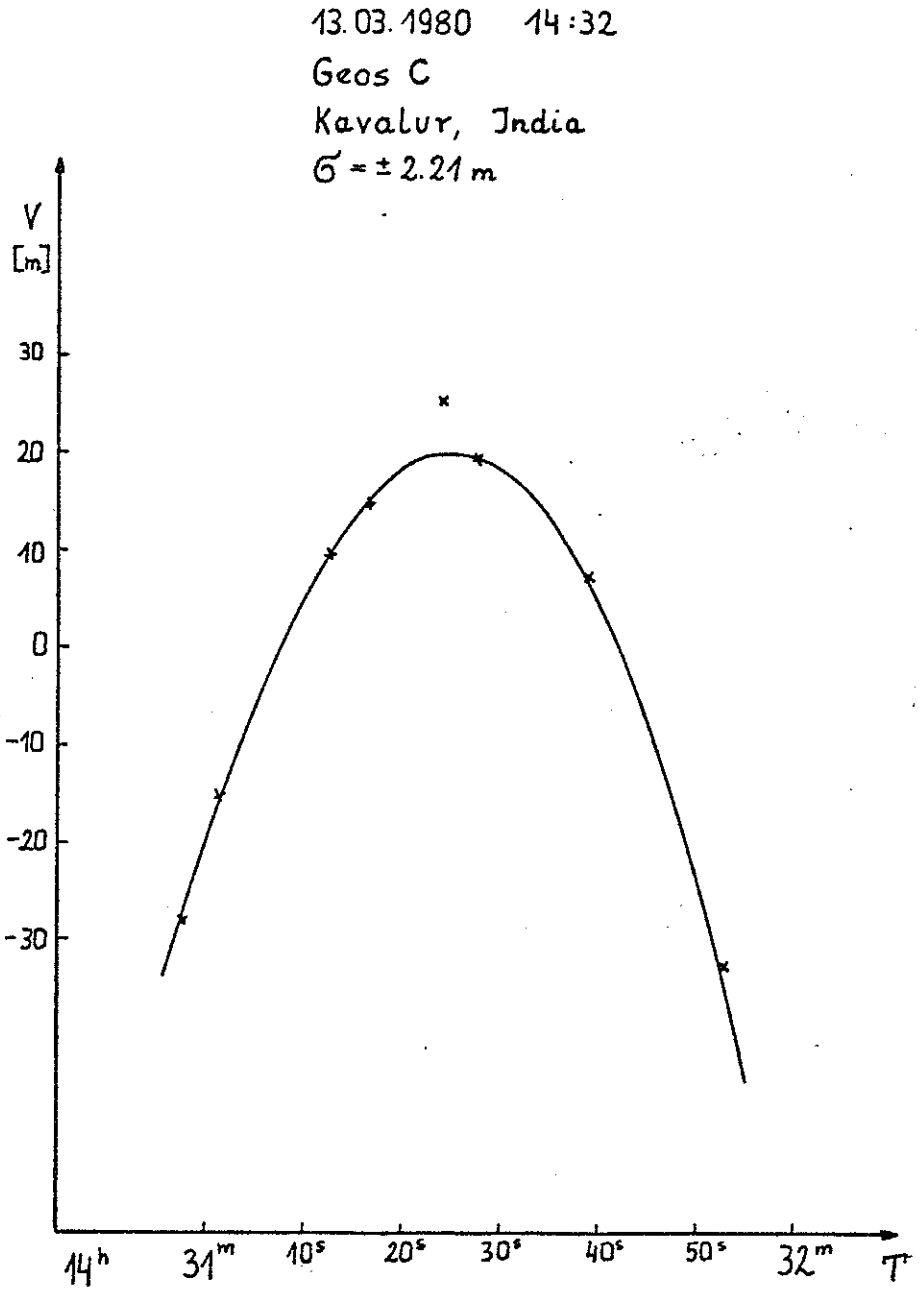


Figure 4: Residuals for good points from Fig. 3, σ = ±221 cm.

where:

- v = residuals from 'p' or 'p-1' polynomial
- n = number of points
- p = polynomial degree

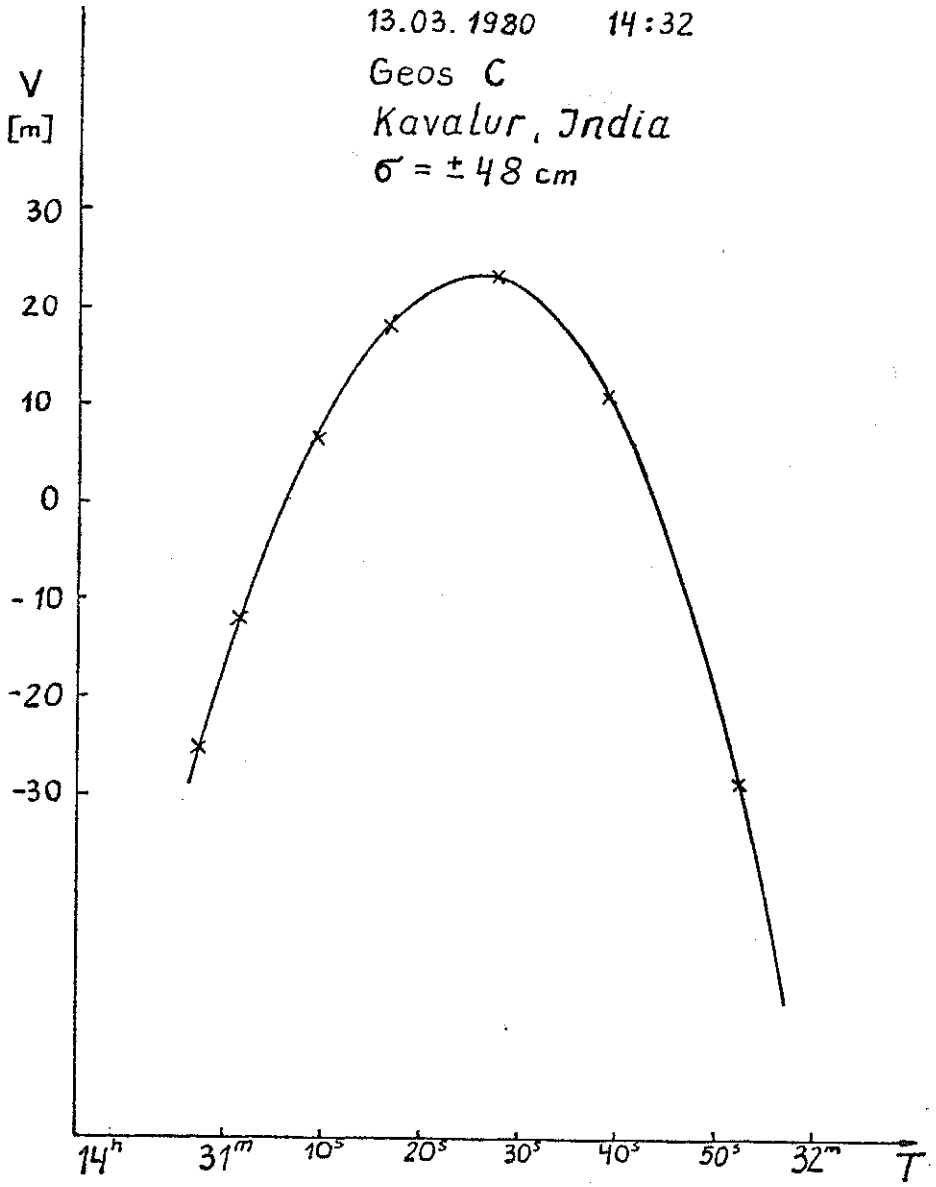


Figure 5: Final residuals for pass, σ = ±48 cm.

The smoothing is stopped, when for successive two polynomial degrees $F < F_o$, where F_o is definite for significance level 0.05 and degree of freedom $\nu_1 = 1$ and $\nu_2 = n-p-1$, it means when the difference between three successive standard deviations is not statistically significant.

3 Correction of the Results

The results of the laser ranging are computed in the two forms. The first one is without any corrections in the SAO 33333 Quick Look Format - epochs for transmitting moment in UTC and two-way range in nanoseconds. This format is used for quick send of the results to cooperation stations and centers of the laser data. In the second form are computed the results with necessary corrections.

1. Calibration Correction - is calculated from the mean of about 20 measurements before and after satellite pass. The ground target distance is corrected for changes of the light velocity in actual atmospheric conditions on the base of the knowledge of ground temperature and pressure by used SAO formula (Gaposchkin, 1973).

2. Atmospheric Correction - is calculated for every point according to ground temperature, pressure and elevation angle of satellite by using simplified SAO formula (Gaposchkin, 1973). This correction results from the changes of light velocity in two-way pass across the atmosphere.

3. Epoch Correction - for the moment of reflecting the laser beam from satellite.

4 Final Data

Computer data consist of all main informations about the satellite pass;

- calibration measurements (mean calibration, standard deviation, calibration correction),
- meteorological data (temperature, humidity, pressure, weather),
- apparatus data (laser voltage, PMT voltage, transmitting and receiving amplitudes, start and stop channel sensitivities),
- epochs and ranges in centimeters with above corrections,
- atmospheric corrections and elevation angle of satellite for every point,
- differences observation-ephemeris,
- residuals from mean difference between observation and ephemeris for correcting ephemeris epochs,
- systematical ephemeris error in range and epoch,
- residuals of the smoothing points for succeeding polynomial degrees, standard deviations and index Fisher's test,
- 33333 Quick Look Format.

5 References

Gaposchkin E. M. - SAO Special Report No. 353, 1973

Schillak, S., E. Wnuk -Veroff. des Zentralinstituts fur Physik der Erde, Nr 52, Teil 3, Potsdam 1977.

6 \ppendix A

```
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 V1(30),V2(30),T(100),RO(100),RRA(100),EL(5,6),RC(100),
     1      RC1(100),RC2(100),TT(100),PP(100),VA(100,20),
     1   SIG(100),D1(100),AUP(100),ARP(100),WS(20)
      INTEGERIV(100,20),IS(20),ID(20),JX(20)
200   FORMAT('1',22X,'ASTRONOMICAL LATITUDE OBSERVATORY')
201   FORMAT(' ',33X,'AT BOROWIEC')
202   FORMAT(' ',30X,'LASER OBSERVATIONS'/////)
203   FORMAT(' ',10X,'SERIAL NUMBER',5X,I5)
204   FORMAT(' ',10X,'DATE',13X,3I5)
205   FORMAT(' ',10X,'CULM. EPOCH',7X,I2,'/',I2)
206   FORMAT(' ',10X,'SATELLITE',9X,I7)
207   FORMAT(' ',10X,'OPERATOR',10X,2A8/////)
208   FORMAT(' ',10X,'OBSERVER',10X,2A8)
209   FORMAT(' ',40X,'C A L I B R A T I O N'/////)
210   FORMAT(' ',34X,'PRE CALIBR.',13X,'POST CALIBR.')
211   FORMAT('0',10X,'CALIBR. EPOCH',9X,I2,'/',I2,20X,I2,'/',I2)
212   FORMAT(' ',10X,'DIAPHRAGM',12X,I2,23X,I2)
213   FORMAT(' ',10X,'TRANS. AMPLITUDE',6X,F3.1,22X,F3.1)
214   FORMAT(' ',10X,'RETURN AMPLITUDE',6X,I3,22X,I3)
215   FORMAT(' ',10X,'NUMBER OF MEASUR.'5X,I3,22X,I3)
216   FORMAT(' ',30X,I2,F8.1,F7.1,10X,2F7.1)
217   FORMAT('0',10X,'MEAN',19X,2F7.1,10X,2F7.1)
218   FORMAT(' ',10X,'STANDARD DEVIATION',4X,F9.2,F6.1, 9X,F9.2,F6.1)
219   FORMAT(' ',10X,'MEAN ERROR',12X,F9.2,15X,F9.2)
220   FORMAT('0',10X,'MEAN CALIBRATION',21X,2F9.1)
221   FORMAT(' ',10X,'CALIBRATION CORRECTION',15X,F9.1)
222   FORMAT('0',10X,'NOTICE')
223   FORMAT('1',34X,'O B S E R V A T I O N'//)
224   FORMAT('0',10X,'METEO DATA',25X,'APPARAT. DATA')
225   FORMAT('0',10X,'TEMPERATURE',F8.1,16X,'LASER VOLTAGE',5X,I6)
226   FORMAT(' ',10X,'HUMIDITY',3X,I8,16X,'PMT VOLTAGE',7X,I6)
227   FORMAT(' ',10X,'PRESSURE',3X,I8,16X,'TRANSM. AMPLITUDE',1X,F6.1)
228   FORMAT(' ',10X,'WEATHER',4X,I8,16X,'RETURN AMPLITUDE',2X,I6)
229   FORMAT(' ',10X,'MOON',7X,A8,16X,'START SENSITIVITY',1X,F6.1)
230   FORMAT(' ',10X,'VISIBILITY',1X,I8,16X,'STOP SENSITIVITY',2X,I6)
231   FORMAT(' ',10X,'TRACKING',3X,I8,16X,'CLOCK CORRECTION',2X,F6.1//)
232   FORMAT('0',11X,'NO',10X,'UTC',12X,'H',3X,'M',6X,'S',15X,'RO'//)
233   FORMAT(' ',10X,I3,F18.10,6X,I2,I4,F12.6,6X,F11.5)
234   FORMAT('1',32X,'COMPARISON WITH EPHEMERIES'//)
235   FORMAT('0',10X,'EPOCH OF ELEMENTS',I7,2I4)
236   FORMAT('0',11X,'NO',4X,'H',7X,'PA',8X,'ROA',10X,'REF',11X,'DRO'//)
237   FORMAT(' ',10X,I3,F7.1,F9.5,3F13.5)
238   FORMAT('1',15X,'DETERMINATION OF THE SYSTEMATIC CORRECTION IN EPOC
     1H AND RANGE')
239   FORMAT('0',11X,'NO',8X,'RC',12X,'ROA-RC',11X,'V'//)
240   FORMAT(' ',10X,I3,3F15.5)
241   FORMAT('0',10X,'EPOCH CORRECTION',4X,F10.4)
242   FORMAT(' ',10X,'RANGE CORRECTION',4X,F10.4)
243   FORMAT(' ',10X,'SQUARE SUM V',8X,F10.4)
```

```
241   FORMAT('1',27X,'DETERMINATION OF THE OBSERVATION ERROR'//)
245   FORMAT('0',11X,'NO',16X,'DEVIATION (V) FOR POLYNOMIAL STEPS')
246   FORMAT('0',20X,I2)
247   FORMAT(' ',18X,I3,8I8)
248   FORMAT(' ',//)
249   FORMAT(' ',10X,I3,I10)
250   FORMAT(' ',10X,I3,8I8)
251   FORMAT('0',10X,'SQUARE SUM',10X,F10.1)
252   FORMAT(' ', 8X,'VARIAN.',I6,8I8)
253   FORMAT(' ', 8X,'ST.DEV.',I6,8I8)
254   FORMAT('1',37X,'SAO 33333 FORMAT'////)
255   FORMAT('+',22X)
256   FORMAT('+',8X)
257   FORMAT('+',23X)
260   FORMAT('1',10X,'ELEMENTS OF ORBIT'///)
261   FORMAT('0',2F20.8,4D16.5)
262   FORMAT('0',10X,'INITIAL DATA'//)
263   FORMAT(' ',10X,I6,2F20.1)
264   FORMAT(' ',8X,'WS',3X,9F8.1)
100   FORMAT(7I10)
      READ(1,100) NRPRZ,NROK,NMC,NDZ,NGZ,NMIN,NRSAT
101   FORMAT(2A8,4X,2A8)
      READ(1,101) AOBS,AOB31,AOPER,AOPER1
102   FORMAT(3I10,F10.1,2I10)
      READ(1,102) K1G,K1M,IPRZ1,AMN1,IAO1,N1
103   FORMAT(F10.1,3I10,2X,A8,2I10)
104   FORMAT(2I10,F10.1,I10,F10.1,I10,F10.1)
105   FORMAT(6F10.1)
      WRITE(3,200)
      WRITE(3,201)
      WRITE(3,202)
      WRITE(3,203) NRPRZ
      WRITE(3,204) NROK,NMC,NDZ
      WRITE(3,205) NGZ,NMIN
      WRITE(3,206) NRSAT
      WRITE(3,208) AOBS,AOB31
      WRITE(3,207) AOPER,AOPER1
      WRITE(3,209)
      WRITE(3,210)
      IPAT=1
      PI=3.1415926535898D0
      RD=180.0D0/PI
      AE=6378.14D0
      S1=0D0
      AKT1=0D0
      AMO1=0D0
      AMK1=0D0
      IF(N1.EQ.0) GO TO 501
      DO 401 K=1,5
      K1=(K-1)*6
401   READ(1,105) (V1(K1+J),J=1,6)
      DO 502 K=1,N1
      V1(K)=V1(K)*5D0
502   AKT1=AKT1+V1(K)
```

```
      AKT1=AKT1/N1
      DO 503 K=1,N1
      V1(K)=AKT1-V1(K)
503   S1=S1+V1(K)*V1(K)
      AM01=DSQRT(S1/(N1-1))
      AN1=N1
      AMK1=AM01/(DSQRT(AN1))
501   READ(1,102) K2G,K2M,IPRZ2,AMN2,IA02,N2
      S2=0D0
      AKT2=0D0
      AM02=0D0
      AMK2=0D0
      IF(N2.EQ.0) GO TO 504
      DO 402 K=1,5
      K1=(K-1)*6
402   READ(1,105) (V2(K1+J),J=1,6)
      DO 505 K=1,N2
      V2(K)=V2(K)*5D0
505   AKT2=AKT2+V2(K)
      AKT2=AKT2/N2
      DO 506 K=1,N2
      V2(K)=AKT2-V2(K)
506   S2=S2+V2(K)*V2(K)
      AM02=DSQRT(S2/(N2-1))
      AN2=N2
      AMK2=AM02/(DSQRT(AN2))
504   CONTINUE
      READ(1,103) TEMP,IWIL,ICIS,IPOG,AKSIE,IWID,IPROW
      READ(1,104) NAPLAS,NAPPMT,AMPNAD,IAMPOD,CZUNAD,ICZUOD,POP
      IF(N1.GT.0.AND.N2.GT.0) AKT=(AKT1+AKT2)/2D0
      IF(N1.GT.0.AND.N2.EQ.0) AKT=AKT1
      IF(N2.GT.0.AND.N1.EQ.0) AKT=AKT2
      WA=0D0
      IF(IPAT.EQ.1) WA=80.29D-6*ICIS/(TEMP+273.2D0)
      AKC=1478.913D0*(1D0+WA+6.917D-4)/0.15D0
      PK=AKT-AKC
      IF(N1.GT.N2) GO TO 507
      NK=N2
      GO TO 508
507   NK=N1
508   CONTINUE
      WRITE(3,211) K1G,K1M,K2G,K2M
      WRITE(3,212) IPRZ1,IPRZ2
      WRITE(3,213) AMN1,AMN2
      WRITE(3,214) IA01,IA02
      WRITE(3,215) N1,N2
      DO 509 K=1,NK
      IF(K.GT.N1) V1(K)=0D0
      IF(K.GT.N2)V2(K)=0D0
      DV1=AKT1-V1(K)
      DV2=AKT2-V2(K)
509   WRITE(3,216) K,DV1,V1(K),DV2,V2(K)
      WRITE(3,217)AKT1,S1,AKT2,S2
      CM01=AM01*15D0
```

```
      CM02=AM02*15D0
      WRITE(3,218) AM01,CM01,AM02,CM02
      WRITE(3,219) AMK1,AMK2
      WRITE(3,220) AKT
      WRITE(3,221) PK
      WRITE(3,222)
      WRITE(3,223)
      WRITE(3,224)
      WRITE(3,225) TEMP,NAPLAS
      WRITE(3,226) IWIL,NAPPMT
      WRITE(3,227) ICIS,AMPNAD
      WRITE(3,228) IPOG,IAMPOD
      WRITE(3,229) AKSIE,CZUNAD
      WRITE(3,230) IWID,ICZUOD
      WRITE(3,231) IPROW, POP
      WRITE(3,232)
108   FORMAT(2F10.1)
107   FORMAT(4I10)
      READ(1,107) IAMP,ITPH,ITPM,ITPS
      TP=(ITPH+(ITPM+ITPS/60D0)/60D0)/24D0
      AMP=IAMP
106   FORMAT(I10)
      READ(1,106) N
      DO 510 K=1,N
      READ(1,108)U,R
      AUP(K)=U
      ARP(K)=R
      U=U/1D4-AMP+ POP/1D3
      U=U/864D2+TP
      RR=R*5D-9
      RRA(K)=RR*149896.229D0
      R=(R*5D0-PK)*1D-9
      U1=U+R/1728D2
      R=R*149896.229D0
      RO(K)=R
      T(K)=U1
      U1=U1*2D0*PI
      CALL RHMS(U1,UH,UM,US)
      IUH=UH
      IUM=UM
510   WRITE(3,233)K,T(K),IUH,IUM,US,R
      WRITE(3,222)
109   FORMAT(4F15.6)
      READ(1,109)XA,YA,ZA,DLU
      DLU=DLU/RD
110   FORMAT(2F20.9)
      READ(1,110) AJDE,DJDE
111   FORMAT(2F15.8,4D12.5)
      DO 511 K=1,4
511   READ(1,111)(EL(K,J),J=1,5)
      READ(1,111)(EL(5,J),J=1,6)
      WRITE(3,260)
      WRITE(3,110) AJDE,DJDE
      DO 560 K=1,4
```

```
56)   WRITE(3,261) (EL(K,J),J=1,5)
      WRITE(3,261) (EL(5,J),J=1,6)
      WRITE(3,262)
      WRITE(3,203) NRPRZ
      WRITE(3,248)
      DO 561 K=1,N
561   WRITE (3,263) K,AUP(K),ARP(K)
      AKE=107.0883D0
      ANDZ=NDZ
      NMC1=NMC
      CALL JDAY(NROK,NMC,ANDZ,AJD)
      AJDEE=IDINT(AJDE+0.5D0+DJDE)
      CALL DATE(AJDEE,IARKE,IAMCE,IADZE)
      WRITE(3,234)
      WRITE(3,235) IARKE,IAMCE,IADZE
      WRITE(3,236)
      S1=0D0
      S2=0D0
      DO 512 K=1,N
      TA=T(K)
      CALL SMC(AJD,TA,DLU,S)
      S=S-2D0*PI*(IDINT(S/(2D0*PI)))
      DT=AJD-AJDE+TA-DJDE
      CALL ELEM(EL(1,1),EL(1,2),EL(1,3),EL(1,4),EL(1,5),
     1          EL(2,1),EL(2,2),EL(2,3),EL(2,4),EL(2,5),
     2          EL(3,1),EL(3,2),EL(3,3),EL(3,4),EL(3,5),
     3          EL(4,1),EL(4,2),EL(4,3),EL(4,4),EL(4,5),
     4          EL(5,1),EL(5,2),EL(5,3),EL(5,4),EL(5,5),EL(5,6),
     5          DT,WM,WD,AI,E,AM)
      CALL KEPL(AM,E,ED)
      V=2D0*DATAN((DSQRT((1D0+E)/(1D0-E)))*DTAN(ED/2D0))
      IF(V.LT.0D0) V=V+2D0*PI
      UM=V+WM
      IF(UM.GT.2D0*PI)UM=UM-2D0*PI
      A=EL(5,2)+EL(5,3)*DT
      A=(AKE/(A*36D1 /RD))**(2D0/3D0)
      DCAI=DCOS(AI)
      DCAI=DCAI*DCAI
      A=A*(1D0+0.1082637D-2*(1D0-3D0*DCAI)/(4D0*A*A*((1D0-
     1  E*E)**1.5D0)))
      CALL WSPOL(A,E,AI,UM,ED,WD,R,XM,YM,ZM)
      S=S-DLU
      XD=(XA*DCOS(S)-YA*DSIN(S))/AE
      YD=(XA*DSIN(S)+YA*DCOS(S))/AE
      ZD=ZA/AE
      ROA=DSQRT((XM-XD)*(XM-XD)+(YM-YD)*(YM-YD)+(ZM-ZD)*(ZM-ZD))
      AH=XD*(XM-XD)+YD*(YM-YD)+ZD*(ZM-ZD)
      BH=DARSIN(AH/ROA)
      RC(K)=ROA*AE
      WA=0D0
      IF(IPAT.EQ.1) WA=0.0414D0*ICIS/(TEMP+273.2D0)
      PA=(2.219D0+WA)/(DSIN(BH)+1D0/(1D3*DTAN(BH)))/1D3
      ROA=RO(K)-PA
      DRO=RO(K)-PA-RC(K)
```

```
      RO(K)=ROA
      S1=S1+RO(K)-RC(K)
      RC2(K)=DRO
      BH=BH*RD
512   WRITE(3,237) K,BH,PA,ROA,RC(K),DRO
      WRITE(3,222)
      TX=0D0
      DO 515 K=1,N
515   S2=S2+(RO(K)-RC(K)-S1/N)*(RO(K)-RC(K)-S1/N)
      NK=0
      DO 525 J=1,5
      J1=J-1
      TX=TX+1D0*(10D0**(-J1))
      NK=0
514   TZ=TX/864D2
      NK=NK+1
      X1=0D0
      X2=0D0
      DO 513 K=1,N
      TA=T(K)+TZ
      CALL SMC(AJD,TA,DLU,S)
      S=S-2D0*PI*(IDINT(S/(2D0*PI)))
      DT=AJD-AJDE+TA-DJDE
      CALL ELEM(EL(1,1),EL(1,2),EL(1,3),EL(1,4),EL(1,5),
     1          EL(2,1),EL(2,2),EL(2,3),EL(2,4),EL(2,5),
     2          EL(3,1),EL(3,2),EL(3,3),EL(3,4),EL(3,5),
     3          EL(4,1),EL(4,2),EL(4,3),EL(4,4),EL(4,5),
     4          EL(5,1),EL(5,2),EL(5,3),EL(5,4),EL(5,5),EL(5,6),
     5          DT,WM,WD,AI,E,AM)
      CALL KEPL(AM,E,ED)
      V=2D0*DATAN((DSQRT((1D0+E)/(1D0-E)))*DTAN(ED/2D0))
      IF(V.LT.0D0) V=V+2D0*PI
      UM=V+WM
      IF(UM.GT.2D0*PI)UM=UM-2D0*PI
      A=EL(5,2)+EL(5,3)*DT
      A=(AKE/(A*36D1 /RD))**(2D0/3D0)
      DCAI=DCOS(AI)
      DCAI=DCAI*DCAI
      A=A*(1D0+0.1082637D-2*(1D0-3D0*DCAI)/(4D0*A*A*((1D0-
     1  E*E)**1.5D0)))
      CALL WSPOL(A,E,AI,UM,ED,WD,R,XM,YM,ZM)
      S=S-DLU
      XD=(XA*DCOS(S)-YA*DSIN(S))/AE
      YD=(XA*DSIN(S)+YA*DCOS(S))/AE
      ZD=ZA/AE
      ROA=DSQRT((XM-XD)*(XM-XD)+(YM-YD)*(YM-YD)+(ZM-ZD)*(ZM-ZD))
      RC(K)=ROA*AE
513   X1=X1+RO(K)-RC(K)
      X1=X1/N
      DO 516 K=1,N
516   X2=X2+(RO(K)-RC(K)-X1)*(RO(K)-RC(K)-X1)
      IF(NK.GT.1) GO TO 517
518   IF(X2.LT.S2) GO TO 520
      ZN=-1D0
```

```
      TX=TX-2D0*(10D0**(-J1))
      DO 519 K=1,N
519   RC1(K)=RC(K)
      GO TO 514
520   ZN=1D0
517   IF(X2.LE.S2) GO TO 521
522   IF(J1.EQ.4) GO TO 525
      TX=TX-1D0*ZN*(10D0**(-J1))
      DO 523 K=1,N
523   RC1(K)=RC(K)
      GO TO 525
521   S1=X1
      S2=X2
      TX=TX+1D0*ZN*(10D0**(-J1))
      DO 524 K=1,N
524   RC1(K)=RC(K)
      GO TO 514
525   CONTINUE
      WRITE(3,238)
      WRITE(3,239)
      DO 526 K=1,N
      ROB=RO(K)-RC1(K)
      ROC=RO(K)-RC1(K)-S1
526   WRITE(3,240) K,RC1(K),ROB,ROC
      WRITE(3,241) TX
      WRITE(3,242) X1
      WRITE(3,243) X2
      DO 530 K=1,N
      TT(K)=T(K)*1440D0
530   PP(K)=1D0
      WRITE(3,244)
      WRITE(3,245)
      L1=N-1
      CALL INTERF(L1,N,TT,RC2,PP,VA,SIG,D1,WS,M)
      MP1=2
      WRITE(3,248)
      IF(M.GT.9) MP1=M-7
      DO 532 J=MP1,M
 532  JX(J)=J-1
      WRITE(3,247) (JX(J),J=MP1,M)
      WRITE(3,248)
      DO 531 I=1,N
      DO 536 J=MP1,M
536   IV(I,J)=VA(I,J)*1D5
531   WRITE(3,250)I,(IV(I,J),J=MP1,M)
      WRITE(3,248)
      DO 534 J=MP1,M
534   IS(J)=SIG(J)*1D10
      WRITE(3,252)(IS(J),J=MP1,M)
      DO 535 J=MP1,M
535   ID(J)=D1(J)*1D5
      WRITE(3,253)(ID(J),J=MP1,M)
      WRITE(3,264) (WS(J),J=MP1,M)
      WRITE(3,254)
```

```
      NSTA=7811
      NMC=NMC1
      CALL FORMSA(NSTA,NRSAT,NROK,NMC,NDZ,IWIL,TEMP,ICIS,AKT,N,T,RRA)
      END
      SUBROUTINE RHMS(A,A1,A2,A3)
      IMPLICIT REAL*8(A-H,O-Z)
      PI=3.1415926535898D0
      RD=180.0D0/PI
      A0=(A*RD)/15.0D0
      IA1=IDINT(A0)
      A1=IA1
      IA2=IDINT((A0-A1)*60.0D0)
      A2=IA2
      A3=((A0-A1)*60.0D0-A2)*60.0D0
      RETURN
      END
      SUBROUTINE JDAY(Y,M,D,JD)
      REAL*8 D,JD,C1,J1,J3,J5
      INTEGER Y,M,C,YA
      IF(M.GT.2) GO TO 10
      M=M+9
      Y=Y-1
      GO TO 20
10    M=M-3
20    C1=Y/100.0
      C=IDINT(C1)
      YA=Y-C*100
      J1=(146097D0*C)/4.0
      J2=IDINT(J1)
      J3=(1461D0*YA)/4.0
      J4=IDINT(J3)
      J5=(153D0*M+2)/5.0
      J6=IDINT(J5)
      JD=J2+J4+J6+D+1721118D0+0.5D0
      RETURN
      END
      SUBROUTINE DATE(JD,Y,M,D)
      REAL*8 Y1,D1,J1,M1,JD,D2
      INTEGER Y,M,D,J
      JD=JD-1721119D0
      Y1=(4*JD-1D0)/146097.0D0
      Y=IDINT(Y1)
      JD=4D0*JD-1D0-146097.0D0*Y
      D1=JD/4.0D0
      D2=D1/1000.0D0
      D=IDINT(D2)
      D2=D1-D*1000.0D0
      M=IDINT(D2)
      D1=D*1000.0D0+M
      J1=(4D0*D1+3)/1461.0D0
      J=IDINT(J1)
      D1=4*D1+3D0-1461*J
      D2=(D1+4D0)/4.0D0
      D=IDINT(D2)
```

```
      M1=(5D0*D-3D0)/153.0D0
      M=IDINT(M1)
      D=5D0*D-3D0-153D0*M
      D1=(D+5D0)/5.0D0
      D=IDINT(D1)
      Y=100D0*Y+J
      IF(M.LT.10) GO TO 10
      M=M-9
      Y=Y+1D0
      GO TO 20
10    M=M+3
20    RETURN
      END
      SUBROUTINE SMC(JD,T,LA,S)
      REAL*8 JD,T,LA,S,T1,S0
      T1=(JD-2415020.0D0)/36525.0D0
      S0=(99.6909833D0+36000.7689D0*T1+0.00038708D0
     1*T1*T1)/57.29577951D0
      S=S0+LA+T*6.30038808D0
      RETURN
      END
      SUBROUTINE ELEM(WM0,DWM,WM1,WM2,WM3,WD0,DWD,WD1,WD2,WD3,I0,DI,
     1I1,I2,I3,E0,DE,E1,E2,E3,M0,DM,DM1,M1,M2,M3,DT,WM,WD,I,E,M)
      REAL*8 PI,RD,U,M4,WM,WD,I,E,M
      REAL*8 WM0,DWM,WM1,WM2,WM3,WD0,DWD,WD1,WD2,WD3,I0,DI,I1,I2,I3
      REAL*8 E0,DE,E1,E2,E3,M0,DM,DM1,M1,M2,M3,DT
      PI=3.141592653589D0
      RD=57.2957795D0
      U=(WM0+DWM*DT)/RD
      WM=WM0+DWM*DT+WM1*DCOS(U)+WM2*DSIN(2D0*U)+WM3*DCOS(3D0*U)
      WD=WD0+DWD*DT+WD1*DCOS(U)+WD2*DSIN(2D0*U)+WD3*DCOS(3D0*U)
      I=I0+DI*DT+I1*DSIN(U)+I2*DCOS(2D0*U)+I3*DSIN(3D0*U)
      E=E0+DE*DT+E1*DSIN(U)+E2*DCOS(2D0*U)+E3*DSIN(3D0*U)
      M4=M0+DM*DT+DM1*DT*DT+M1*DCOS(U)+M2*DSIN(2D0*U)+M3*DCOS(3D0*U)
      M=(M4-IDINT(M4))*360D0
      WM=WM/RD
      WD=WD/RD
      M=M/RD
      I=I/RD
      RETURN
      END
      SUBROUTINE KEPL(M,EE,E)
      REAL*8 S,E,M,E1,EE
      S=0.0D0
      E=M
30    E1=M+EE*DSIN(E)
      S=S+1D0
10    E=E1
      IF(S.GT.100.0D0) GO TO 10
      IF(DABS(E-E1).GT.10D-10) GO TO 20
      GO TO 40
20    E=E1
      GO TO 30
40    RETURN
```

```
      END
      SUBROUTINE WSPOL(A,EE,I,U,E,W,R,X,Y,Z)
      REAL*8 A,EE,I,U,E,W,R,X,Y,Z
      R=A*(1.0D0-EE*DCOS(E))
      X=R*(DCOS(U)*DCOS(W)-DSIN(U)*DSIN(W)*DCOS(I))
      Y=R*(DCOS(U)*DSIN(W)+DSIN(U)*DCOS(W)*DCOS(I))
      Z=R*DSIN(U)*DSIN(I)
      RETURN
      END
      SUBROUTINE INTERF(L1,N,T,Y,P,V,SIG,D1,WS,M)
      IMPLICIT REAL*8 (A-H,O-Z)
      REAL*8 T(1),Y(1),P(1),V(100,10),SIG(1),D1(20),X(100),F(100,20),
     1 C(20),SIGMA2(20),D2(20),A(20,20)
      REAL*8 F01(34),SKW(20),FP(20),WS(20)
      DATA F01/161.0,18.51,10.13,7.71,6.61,5.99,5.59,5.32,5.12,4.96,
     *4.84,4.75,4.67,4.60,4.54,4.49,4.45,4.41,4.38,4.35,
     *4.32,4.30,4.28,4.26,4.24,4.22,4.21,4.20,
     *4.18,4.17,4.08,4.00,3.92,3.84/
      WS(1)=-1.0
      WS(2)=-1.0
      IF(L1.GT.15) L1=15
      SY=0D0
      SL=0D0
      SM=0D0
      AX=T(N)-T(1)
      DO 10I=1,N
      X(I)=(T(I)-T(1))/AX
      F(I,1)=1.0D0
      SL=SL+P(I)*Y(I)
      SM=SM+P(I)
10    SY=SY+Y(I)
      YB=SY/N
      SY=0D0
      DO 11 I=1,N
11    SY=SY+(Y(I)-YB)*(Y(I)-YB)
      S2=SY/N
      S=DSQRT(S2)
      C(1)=SL/SM
      DO 12 I=1,N
12    V(I,1)=Y(I)-C(1)
      DO 50 J=2,L1
      J1=J-1
      DO 40 K=1,J1
      SL=0D0
      SM=0D0
      DO 21 I=1,N
      SL=SL+P(I)*(X(I)**J1)*F(I,K)
21    SM=SM+P(I)*F(I,K)*F(I,K)
40    A(J1,K)=SL/SM
      DO 23 I=1,N
      SA=0D0
      DO 22 K=1,J1
22    SA=SA+A(J1,K)*F(I,K)
23    F(I,J)=X(I)**J1-SA
```

```
      SL=0D0
      SM=0D0
      DO 24 I=1,N
      SL=SL+P(I)*Y(I)*F(I,J)
24    SM=SM+P(I)*F(I,J)*F(I,J)
      C(J)=SL/SM
      DO 26 I=1,N
      SA=0D0
      DO 25 K=1,J
25    SA=SA+C(K)*F(I,K)
26    V(I,J)=Y(I)-SA
      SL=0D0
      SM=0D0
      DO 27 I=1,N
      SL=SL+P(I)*V(I,J)*V(I,J)
27    SM=SM+P(I)*F(I,J)*F(I,J)
      SM=SM*(N-J)
      SKW(J)=SL
      SIGMA2(J)=SL/SM
      SIG(J)=DSQRT(SIGMA2(J))
      D2(J)=SL/(N-J)
      D1(J)=DSQRT(D2(J))
      DS=D2(J)/S2
      SIG(J)=D2(J)
      IF(J.GT.2) GO TO 81
      GO TO 50
  81  FP(J)=(SKW(J-1)-SKW(J))/(SKW(J)/(N-J))
      K=N-J
      WS(J)=-1.0
      IF(FP(J).LE.F01(K)) WS(J)=1.0
      IF(WS(J).GT.0.0.AND.WS(J-1).GT.0.0) GO TO 60
50    CONTINUE
60    M=J
      RETURN
      END
      SUBROUTINE FORMSA(NSTA,NRSAT,NROK,NMC,NDZ,IWIL,TEMP,ICIS,
     1                  AKT,N,T,RO)
      IMPLICIT REAL*8(A-H,O-Z)
      REAL*8 T(1),RO(1)
      PI=3.1415926535898D0
301   FORMAT('0',10X,'33333',I5,I1,3I2)
      NROK=NROK-1900
      IDE=NROK/10
      IDK=NROK-10*IDE
      WRITE(3,301) NSTA,IDE,IDK,NMC,NDZ
302   FORMAT(' ',10X,I5,I3,'0',2I2,I3,'0',I5,'0 ',I5,' 00000')
      IDE=NRSAT/100
      IDK=NRSAT-IDE*100
      KT=10*AKT
      ITE=0
      IF(TEMP.LT.0D0) ITE=1
      JTE=10*TEMP+0.1
      WRITE(3,302) IDE,IDK,IWIL,ITE,JTE,ICIS,KT
303   FORMAT(' ',10X,2I2,I1,' ',I5,' 00000 0',I4,' ',I4,'0')
```

```
      DO 310 K=1,N
      U=T(K)
      R=RO(K)
      U=U*2D0*PI
      CALL RHMS(U,UH,UM,US)
      IUH=UH
      IUM=UM
      US=US-R/299792.458D0
      IUS=US/10
      IUSP=(US-IUS*10)*10000+0.1
      R=R/149896.229D0
      R=R*1D9
      IDE=R/10000
      IDK=R-IDE*10000+0.1
310   WRITE(3,303)IUH,IUM,IUS,IUSP,IDE,IDK
      RETURN
      END
```

PART II

Ephemeris Reconstruction Software

by

Brian D. Cuthbertson
Departments of Astronomy & Aerospace Engineering
University of Texas
Austin, Texas 78712

1 INTRODUCTION

Like many software development efforts, the Transportable Laser Ranging System (TLRS) satellite ephemeris reconstruction software discussed in this paper is a creature of evolution. Its earliest ancestor was installed in the newly-completed TLRS Nova minicomputer in late 1979, and performed serviceably, if slowly, during the initial testing of the TLRS system. Since that time there have been several descendents, each of which has evolved to enhance the speed and/or accuracy of its predecessor. The present software satisfactorily fulfills present TLRS requirements for tracking the LAGEOS satellite, and in fact is adequate for use with other lower satellites as well. This paper summarizes the operational features of the present software, and the environment in which the software operates as part of the overall TLRS tracking system.

2 T.L.R.S. OPERATING ENVIRONMENT

The TLRS ephemeris reconstruction software can perhaps be best introduced through an explanation of its roll in the larger TLRS software tracking system. Naturally, the overall goal of the tracking system is to provide accurate real-time coordinates of a satellite (LAGEOS) during a given pass. The process may be outlined as follows: First, a LAGEOS ephemeris must be supplied to the TLRS crew. This ephemeris is presently generated at the University of Texas Department of Aerospace Engineering through the UTOPIA orbital analysis system, using full geopotential fields and detailed perturbation forces. The ephemeris is accurate for up to several months, and provides state vectors in a metric INTER RANGE VECTOR format at periodic intervals. Originally this interval was 3 hours, but integrator package accuracy improvements have allowed for an increase to 1 day for LAGEOS. Preceding a given LAGEOS pass, the TLRS crew will use the ephemeris and the ephemeris reconstruction software to generate a computer file of predicted altitudes, elevations, and ranges at 1-minute intervals. Often this will be done at the beginning of a day for all passes to be observed. During an actual pass, the TLRS tracking software will use the computer file and an interpolation algorithm to provide continuous real-time altitude, elevation, and range estimates to the tracking hardware. Hence the ephemeris reconstruction software itself is not required to provide real-time performance during actual tracking operations.

3 ACCURACY REQUIREMENTS

Prior to development of the initial 1979 software package, accuracy requirement discussions led to adoption of a roughly 100-meter total maximum pointing error. This pointing accuracy must naturally be coupled with a reasonable integration period. (Any ephemeris reconstruction package can meet any pointing accuracy requirement for a short enough integration time.) The minimum tolerable integration time period was thus set at 3 hours, since 3 hours appeared to be the shortest period desirable for successive state vectors on a LAGEOS IRV ephemeris. As mentioned previously, this 3 hour minimum period proved to be too conservative. The present reconstruction software maintains accuracy well within the 100-meter limit for periods on the

order of 24 hours. Hence, the 24-hour period has been adopted as the standard interval for ephemeris state vectors.

4 FORCE MODELING

Any satellite ephemeris reconstruction software package is essentially completely defined by the force models and the integrator it uses. For the TLRS software, both of these components have evolved since the development of the initial version in 1979. The force modeling in the initial model consisted strictly of a geopotential field model using conventional terms including up to 4th degree and order. This field was evaluated using a Pines (rectangular) geopotential formulation [Pines,1973]. Though adequate for 3-hour integrations, that first force model has been considerably expanded in the present "24-hour" version. In particular, the geopotential model now uses normalized terms including up to 7th degree and order. This allows inclusion of several higher order terms of particular significance to LAGEOS. The field is evaluated using a new Pines normalized geopotential formulation developed especially for this application by Dr. Bob Schutz of the University of Texas Department of Aerospace Engineering. In addition to geopotential perturbations, the present package also models lunar and solar perturbations using an analytical model that does not require lunar and solar ephemerides. The combination of a 7 by 7 geopotential field with lunar and solar ephemerides produces the accuracies desired in the present "24-hour" ephemeris reconstruction software.

5 INTEGRATOR

The integrator used in the TLRS software was originally a Runge Kutta second order integrator. Although adequate in terms of accuracy, it was relatively slow; a 24-hour integration required on the order of 20 minutes in the first software package run on the TLRS Nova minicomputer. In the first revision of the software package, the Runge Kutta integrator was replaced with the much more sophisticated Krogh-Shampine-Gordon integrator, a 14th order multi-step integrator very similar to the one used in the U.T. Aerospace Department's UTOPIA orbital analysis system [Lundberg, 1981]. Although the new integrator is much larger in terms of source code, it is also much faster since it employs a table interpolation scheme. A 24-hour integration, for example, requires on the order of only 5 minutes of TLRS Nova minicomputer time.

6 OTHER FEATURES

In addition to force modeling and the integrator itself, there are several other features worth noting. First, the mean equatorial system has been chosen as the inertial system of integration. In the initial reconstruction software an arbitrary inertial system coincident with the initial state vector was used, but this later became awkward when evaluations of lunar and solar perturbations were added. Second, pseudo-evaluations have been incorporated into the routine calculating perturbing accelerations.

In ended as a time saver, this feature saves each time and total perturbing acceleration for possible re-use during the next integrator call, thus preventing unnecessary and time-consuming re-evaluations of the geopotential and lunar/solar perturbations at duplicate times. Third, the software takes advantage of predicted X and Y polar motion data, and earth rotation rate change data, that are provided in the U.T. Aerospace ephemeris in addition to the IRV state vector for each ephemeris time point. The X and Y polar motion data are used in the conversion of the satellite pseudo body-fixed state to true body-fixed state for the calculation of satellite azimuths, altitudes, and ranges. It might also be noted that the predicted range is corrected for refraction effects. The earth rotation rate change is applied to a constant base, which is then used in various coordinate system transformations. The fact that both polar motion and earth rotation predictions appear on U.T. Aerospace ephemerides reflects on the significant work put forth at U.T. during the past few years on extrapolating these values, work done in conjunction with the TLRS software development effort. Fourth and finally, it might be mentioned that the geopotential field used in the system at present is the GEM-10 normalized field. One flexible feature of the software is that the geopotential field coefficients, as well as other basic geophysical constants, are read from a data file, thus allowing an update of geopotential field or constants without re-compilation of code.

7 NOTES ON SOURCE CODE

The development of the TLRS ephemeris reconstruction software package was not done directly on the TLRS Nova minicomputer on which it was installed, but rather on a similar sized PDP 11/60 minicomputer in the U.T. Dept. of Aerospace Engineering. The Aerospace system offered the benefits of ease-of-access plus a direct link to the powerful U.T. Cyber computers, on which the UTOPIA orbital analysis system was used in generating satellite ephemerides needed for the development work. The FORTRAN source code available with this paper is taken from the PDP 11/60 development system, and reflects the "test bed" orientation of that system. Options allow either interactive or file input of initial, intermediate, and final comparison state vectors, and allow output of radial, transverse, and normal residuals, or of altitude, elevation, and ranges. Program size permits operation without overlays with 32000 16-bit words of memory.

8 CONCLUDING REMARKS

The TLRS ephemeris reconstruction software has proved itself an effective tool during TLRS field operations over the past two years, particularly when used for LAGEOS satellite operations, given LAGEOS ephemeris information generated by the University of Texas Department of Aerospace Engineering. Its performance is satisfactory enough that no modifications to the existing software are anticipated in the near future. The package is also being installed in the University of Texas Astronomy Department's Mobile Laser Ranging System (MLRS), now in place at McDonald Observatory, for satellite and lunar laser ranging work.

9 REFERENCES

Lundberg, J.B., "Multistep Integration Formulas for the Numerical Integration of the Satellite Problem," Inst. for Advanced Study in Orbital Mechanics, The University of Texas at Austin, TR81-1, 1981

Pines, S., "Uniform Representation of the Gravitational Potential and its Derivatives," AIAA J. 11 (11), 1508, 1511, 1973

Spencer, J.L., "Pines Nonsingular Gravitational Potential: Derivation, Description, and Implementation," McDonnell Douglas Technical Services Company, Inc., Report MDC W0013, NASA contract NAS 9-13970, 1976.

1C APPENDIX A

****
TAPE FORMAT:-
9-TRACK, 1600 BPI
LOGICAL RECORD SIZE = 90 ASCII BYTES/RECORD
BLOCK SIZE = 100 RECORDS/BLOCK
(THE LAST BLOCK OF EACH FILE MAY BE SHORT)
****
TAPE CONTENTS:

TAPE CONTAINS FILES OF FORTRAN SOURCE CODE WHICH CAN GENERATE TWO PROGRAMS, IRVINT AND RESGEN. THE FORTRAN SOURCE CODE WAS DESIGNED TO RUN ON A DIGITAL EQUIPMENT CORPORATION PDP 11/60 MINICOMPUTER UNDER THE RSX11-M OPERATING SYSTEM. FOR ADDITIONAL INFORMATION ON EITHER PROGRAM OR ITS OPERATION, CONTACT DR. BOB SCHUTZ, RICHARD EANES, OR BRIAN CUTHBERTSON, AT THE UNIVERSITY OF TEXAS DEPARTMENT OF AEROSPACE ENGINEERING, UNIVERSITY OF TEXAS AT AUSTIN, AUSTIN, TEXAS 78712, U.S.A.

PROGRAM IRVINT:
(IRVINT IS AN ACRONYM FOR INTER-RANGE VECTOR INTEGRATION PROGRAM)
THIS PROGRAM CONTAINS AN INTERACTIVE VERSION OF THE SATELLITE INTEGRATION PACKAGE DEVELOPED FOR FIELD MINICOMPUTER USE IN LASER-RANGING OPERATIONS USING THE TLRS (TRANSPORTABLE LASER RANGING SYSTEM) AND MLRS (MOBILE LASER RANGING SYSTEM) OPERATED BY THE UNIVERSITY OF TEXAS MCDONALD OBSERVATORY. ONCE INTEGRATION ALGORITHMS ARE VERIFIED IN THIS PACKAGE, THE RELEVANT ROUTINES ARE TRANSFERRED TO THE TLRS AND MLRS NOVA MINICOMPUTERS FOR INSTALLATION IN THE FIELD SOFTWARE PACKAGES.

PROGRAM RESGEN:
(RESGEN IS AN ACRONYM FOR RESIDUAL-GENERATION PROGRAM)
THIS PROGRAM IS A MODIFIED VERSION OF IRVINT, USED AT THE U.T. DEPT. OF AEROSPACE ENGINEERING TO DO PRELIMINARY COMPARISONS OF OBSERVED SATELLITE RANGE VALUES (FROM INCOMING QUICK-LOOK RANGING DATA) TO CALCULATED (INTEGRATED) RANGE VALUES. RESIDUAL DIFFERENCES BETWEEN OBSERVED AND CALCULATED RANGES ARE PRODUCED.

THESE DRIVER PROGRAMS ARE INCLUDED TO DEMONSTRATE THE USE OF THE VARIOUS SUBROUTINES SO THAT MORE VERSATILE PACKAGES TAILORED TO EACH USERS SPECIFIC NEEDS CAN BE CONSTRUCTED.

AS THIS IS OUR FIRST ATTEMPT TO EXPORT THE TLRS/MLRS PREDICTION SOFTWARE, ANY FEEDBACK CONCERNING PROBLEMS YOU ENCOUNTER IN ADAPTING THE CODE TO USE ON OTHER SYSTEMS WILL BE APPRECIATED AND CAN BE TRANSMITTED TO THE ABOVE MENTIONED ADDRESS.
****

SOURCE CODE REQUIRED FOR PROGRAM COMPILATIONS:

IRVINT REQUIRES THE FOLLOWING TAPE FILES OF SOURCE CODE FOR COMPILATION:
IRVINT, KSG, GEOPINN, ROTLIB, AZELVR, DERIV, ASKIRV, SUNMON, HOLIB

RESGEN REQUIRES THE FOLLOWING TAPE FILES OF SOURCE CODE FOR COMPILATION:
RESGEN, KSG, GEOPINN, ROTLIB, AZELVR, DERIV, ASKIRV, SUNMON, HOLIB

****

COMPLETE LIST OF FILES ON THIS TAPE:

FILE NO.:	FILE NAME:	DESCRIPTION:
1	INTRO	TAPE CONTENT INFORMATION (THIS FILE).
2	IRVINT	PROGRAM IRVINT; ROUTINES SITES,ASKAZ.
3	RESGEN	PROGRAM RESGEN; ROUTINES SITES, READOB, FNDIRV.
4	AZELVR	ROUTINE AZELVR.
5	ASKIRV	ROUTINES ASKIRV, GETIRV, RAOGU.
6	GEOPINN	ROUTINES GEOPINN, GEOSET.
7	DERIV	ROUTINE DERIV.
8	KSG	ROUTINES INTEGR, KSG, KSGDR, SGSTRT, SGSTEP, SGNTRP, KSGCO, BCKDIF.
9	SUNMON	ROUTINES SUNMON, ECLEQ.
10	HOLIB	ROUTINES EQN, KEP, PMPTRB.
11	ROTLIB	ROUTINES ROTATE, RTNROT, MA3331.
12	GEM10N	GEOPHYSICAL DATA FILE; INPUT FOR IRVINT AND RESGEN.
13	SITE	LASER-RANGING SITE DATA FILE; INPUT FOR IRVINT AND RESGEN.
14	IRV	SAMPLE METRIC INTER-RANGE VECTOR DATA FILE; INPUT FOR IRVINT AND RESGEN.
15	OBS	SAMPLE QUICK-LOOK RANGE DATA FILE (SEASAT DECIMAL FORMAT); ONE PASS FROM HALEAKALA (7210) AND ONE PASS FROM WETTZELL (7834) ON 08 JULY 82; INPUT OBSERVATION FILE FOR RESGEN.
16	RES	SAMPLE RESGEN OUTPUT RESIDUAL FILE. RESIDUAL FILES FROM 3 EXECUTIONS OF RESGEN ARE APPENDED TOGETHER WITH COMMENTS.

****

END OF FILE INTRO.

****

Real-Time Data and Quick-Look for the CERGA LLR System

by

J. Kovalevsky and S. Lengelle
CERGA
Grasse, France

1 Object of the Programs

Two programs are used in CERGA in order to control, in real-time, the data acquisition of the lunar laser and to perform an almost real-time quick look treatment of the data obtained, so that the operator may know its approximate quality.

1.1 Program "VISTIR"

The program "VISTIR" (visulization des tirs = firing visualisation) performs the following tasks:

- Acquisition of the time of laser firings as registered by the event-timer.
- Control of the electronic gate after each firing so that it is opened around the return time as forecast by the ephemerides.
- Acquisition of the events registered by the event-timer and computation of the (o - c) by comparison of the ephemerides.
- Display of the (o - c) on the screen on a histogram divided in 50 channels corresponding each to 1/50 of the gate width.

At the end of a series of laser shots, the observer may:

- Request a print-out of the number of non-isolated (o - c) in 5 nanoseconds channels.
- Reduce at will the channel width of the histogram and recenter around any indicated old channel, producing a new histogram.
- Print the time, the measured delay and the corresponding (o - c) for all events in the channel of the maximum of the last histogram and in the four surrounding channels.

If this first quick-look shows for for one or several series of shots on the same retroreflector that there are probable returns above the noise, it is possible to analyze more accurately together these series of events using the next program.

1.2 Program "TIRESU"

The program "TIRESU" (tirs-resultats = firing results) performs the following task:

- Computation of the (o - c) of all events by comparison with ephemerides,

- Display of rejected events (|(o - c)| > 10 microseconds),
- Print-out of the number of all (o - c) in every 5 nanosecond channel,
- Display on the screen of the histogram of (o - c) (see VISTIR),
- Reduction of the channel width at request of the operator and display of a new histogram centered at any old channel indicated by the operator.

When the operator judges that no new histogram is needed, and on request of the operator, the program:

- Prints the time, the measured delay and the corresponding (o - c) for all events in the channel of the maximum of the last histogram and in the four surrounding channels.
- Computes by least squares method a regression straight line representing (o - c) as function of time. This is a good representation of the normal trend of (o - c) in function of time due to (UT1-UTC) difference if the duration of the observations does not exceed 20 minutes. The coefficients of the straight line are given and the residuals of the selected events with respect to this line are printed. The standard deviation of these residuals is also printed.
- The operator can examine these residuals and may order the program to ignore some of the events, for instance if their residuals are too large. Then, the whole procedure is started again.
- When the operator does not request more rejection, he enters in the computer the parameters of the observation (temperature, pressure,...) that are printed.
- Finally, on request of the operator, all events (time and delay) are printed.

2 Specifications of the Programs

Both programs are written in FORTRAN IV for the Data General Eclipse 5200 system composed of:

- CPU with 32 K byte
- Disk unit
- Alphanumeric Tektronix display
- 16 bits interface (digital input/output)

- Teletype
- RS232C interface (ALM 8).

The software support is the real-time disk operator system (RDOS), with the multitask package for FORTRAN.

However, one subroutine, (WPORTE) controlling the data flow to the electronic gate is written in assembly language.

Figure 1 shows the connections used in real-time data acquisition "VISTIR".

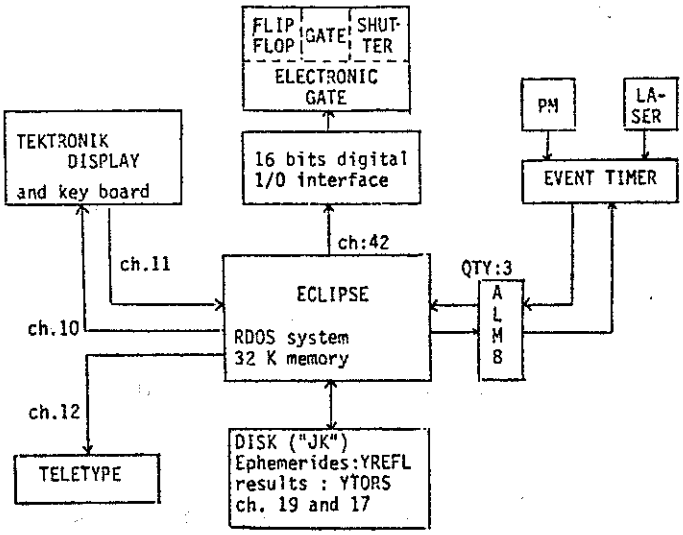


Figure 1: Connections with the "Eclipse" for VISTIR program.

They have the following characteristics:

- Connection with the event-timer through ALM 8: channel QTY: 3.
- Connection with the electronic gate through the 16 bit input/output interface: channel 42.

Other connections are common for both programs "VISTIR" and "TIRESU".

- The Tektronix display: channel WRITE = 10 and channel READ = 11.
- The teletype: channel WRITE = 12.

The program and files are in the directory "JK". It is also assumed that the ephemerides are prepared and recorded in an ephemeris file: "JK : YREFL", channel 19.

For the program "VISTIR" the observations are recorded in an observation file: "JK : YTORS", channel 17. All the observation files that are to be used in "TIRESU" will have to be previously appended in a file "JK : YTIRS".

Let us now study successively each of the two programs.

3 Real Time Data Acquisition and Visualization Program "VISTIR"

3.1 Structure of the program

The program has a multitask organization that permits to activate various parts of the computation in a given order defined by their priorities and the occurrence of exterior events.

The program starts with the initialization task that is "VISTIR" proper.

This task is the initialization of the overall program. It includes:

- Reading of the ephemerides into commons,
- Preparation of the gate treatment (request of its half-with),
- Drawing the histogram axes. This is done using the subroutine "WHIST" initialized by "HIS" reset to zero (subroutine WHIST is used to draw a histogram of 50 numbers written in the common "HIS").

At the end of the task, the three other task are activated. These are:

- ANALY : 1st priority
- SOLO 2 : 2nd priority
- SOLO 1 : 3rd priority

Before the initialization is killed, ANALY is held, and control given to SOLO 2.

The aim of SOLO 2 is to stop the treatment when the operator informs the computer that the firings are over. SOLO 2 is held until a "1" is typed on the Tektronix keyboard. Until this happens, the control is shared by the other two tasks, as shown on figure 2.

3.1.1 SOLO 1

The first task, SOLO 1, reads one by one the data present in the event-timer and recognizes whether it is a firing or a return event. The structure of the event-timer record is the following:

- 1st digit: tens of hours if it is an event; tens of hours plus 8 if it is a firing,

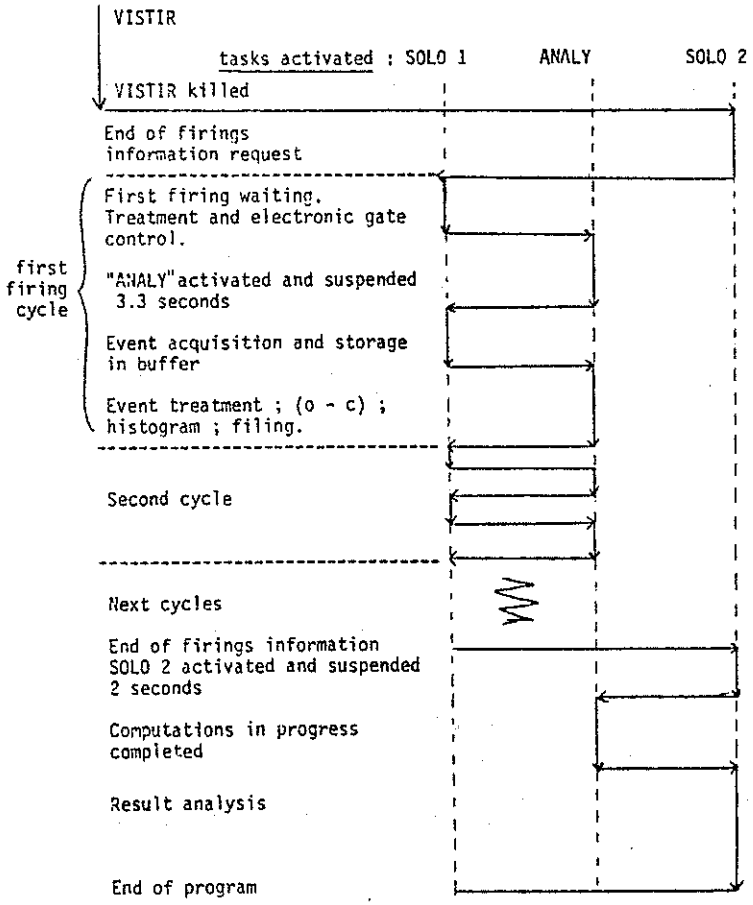


Figure 2: "VISTIR" task sequence.

- next digits: hours (1), minutes (2), seconds (2), hundreds of picoseconds (10).

If the first digit is "8", "9", or "A", the event is recognized as a firing. The expected delay is computed by interpolation of the ephemerides. The time is stored in the buffer. The delay "D" is also sent to the electronic gate using the subroutine "WPORTE". This subroutine is written in assembly language. It sends to the gate the eight BCD digits of (D - 2) seconds expressed in 10^{-8}s.

Finally, the control is given to ANALY that is, then suspended during 3.3 seconds, so as to leave time to SOLO 1 to treat the return events. If the first digit is "0", "1" or "2", the event is recognized as a return. It is stored in the buffer.

3.1.2 ANALY

The second task, ANALY, computes the (o - c) for each event present in the buffer and increments by 1 the corresponding channel of the histogram. The observations are filed in "JK : YTORS". If the (o - c) differs by less than 10 nanoseconds of a previous one or if it is at least the fourth in a

channel, the bell rings.

When all the events filed in the buffer are treated, the control is returned to SOLO 1. Warning is given to the operator when the observation files are 75% complete (150 events). If they are full, the computer turns SOLO 1 off. No more events can be treated and control is given to SOLO 2.

3.1.3 SOLO 2

The third task, SOLO 2, is called when the operator punches a "1" on the keyboard to indicate the end of the series of observations or when the observation files are full. Another two seconds are given to ANALY to finish the computation in progress before these tasks are killed. If there are less than four events, the program is not processed. If there are at least four events, SOLO 2 analyzes the (o - c) and displays the results as follows:

- A print-out on the teletype of the distribution of possible returns. The (o - c) range between $\pm$ the half width of the electronic gate. This interval is divided in 100 ns sections. Each section is divided in 20 channels of five nanosecond width. The numbers of (o - c) in all channels of a section are printed whenever:

  1. there is one (o - c) in the first or last two channels of the section.

  2. there are two (o - c) or more in at least one channel of the section.

  3. there are (o - c) in two consecutive channels or separated by one channel.

- The operator may request a change in the range of the histogram. The minimum and the maximum (o - c) of this range, the number of rejected (but not suppressed) events falling outside this new range, the new channel width of the histogram are displayed and, then, the new histogram itself (using subroutine "WHIST".) This request can be iterated as many times as desired. The channel width is usually diminished to observe more details in the distribution of the residuals, but it may also be widened without inconvenience.

- If the operator judges that at a certain stage there is a significant maximum in the histogram, the teletype prints out the events corresponding to the channel of the maximum in the last histogram displayed as well as to the four surrounding channels.

3.2 Operating Procedure

The assembly of the program is done with the following statement: "RLDR JK : <VISTIR WHIST SOLO1 WPORTE ANALY SOLO2> FMT.LB 10/C 5/K FORT.LB".

The program must be called when all the peripherals and hardware connections are ready, and the ephemeris file placed in "JK : YREFL". The procedure is then straight forward, since the computer displays in full words (in French) the actions that the operator should take. These are, in sequence, the following:

1. NUMERO DE SERIE ? FORMAT.., PUIS RETURN: request of serial number of the observation series, so as to recognize it in the appended file of observations.

2. QUELLE DEMI-LARGEUR DE PORTE ? FORMAT .... DIZAINES DE NANOSECONDE: request of the half width of the electronic gate in 10^{-8} second.

3. APPUYER SUR "RESET" PUIS SUR 0 ET "RETURN": operator is requested to clear the display unit. Minimum and maximum (o - c) are displayed as well as the axes of the histogram.

4. A LA FIN DES TIRS, TAPER 1 ET "RETURN": the operator is now authorized to start the laser operation. He has nothing to do with the keyboard until he judges that the series is over. He then should punch "1-RETURN" to end the data acquisition. During the firings, the histogram of (o - c) is built and the bell will ring to warn that there are at least four events in a channel or that there are two events within 10 nanoseconds.

When the analysis starts, a new conversation between the operator and the console takes place:

1. RESULTATS SAUVEGARDES DANS JK : YTORS: the computer informs the operator that the observations are saved and that the analysis will take place (if not, the program stops). While the teletype prints the distribution of possible returns, the console displays rejected events outside the histogram limits and the characteristics of the next histogram:

   - (o - c) minimum and maximum in the histogram,

   - width of the histogram,

   - number of suppressed events,

   - width of a channel of the histogram.

2. APPUYER SUR "RESET" PUIS SUR 0 ET "RETURN": the operator is requested to clear the display unit for the next histogram. Minimum and maximum (o - c) and the channel width are displayed, and then, the histogram itself.

3. FAUT-IL UN HISTOGRAMME PLUS FIN?/OUI: FAIRE 1; NON: FAIRE 0: the operator is requested to state whether he wants a more detailed histogram. If the answer is "YES":

4. NUMERO DU PAS DU MAXIMUM?: the operator must give the two digits of the channel number around which the new histogram should be displayed.

5. VALEUR DU PAS ? FORMAT ..., L'ANCIEN ETAIT DE: XXX NANOSECONDS: the operator is requested to give the channel width (in nanoseconds) he wishes for the new histogram. Then the new minimum and maximum values for (o - c) as well as the number of rejected events are displayed. The operation is resumed at phase 2).

6. If the answer to the question 3) is "NO", Y-A-T-IL UN RESULTAT PROBABLE ? OUI: FAIRE 1; NON: FAIRE 0: the operator should answer that there is a probable result if there is a well defined maximum of the histogram and if all probable results are within two channels of this maximum (if not, one should have chosen another channel width). If the answer is "YES", these observations are printed and the program stops. If it is "NO", the program stops.

3.3 Test Data

It is not possible to provide test data for "VISTIR", since it is a real-time program and requires the presence of the hardware (laser, event-timer, electronic gate). However, most of the functions of SOLO 2 (the only task that does not depend on the hardware) are present in TIRESU, for which we give test data.

4 Quick-look Treatment Program "TIRESU"

4.1 Structure of the Program

The program has a simple almost linearly constructed flow-chart. Only one subroutine is called: "WHIST" for drawing the histograms (as in "VISTIR").

Figure 3 that gives the flow-chart of the program, the main successive functions of the program being the following:

- Reading the ephemerides and the observation files and computing the (o - c) for each observation.

- Suppression and display of all observations that are outside the range $\pm$ 10 microseconds.

- Print out on the teletype of the distribution of all (o - c). All sections of 100 ns in the interval of (o - c) are divided in 20 channels of five nanosecond width. The numbers of (o - c) in all channels of a section are printed if there is at least one (o - c) in the corresponding interval.

- Display of the histogram of all the nonsuppressed (o - c).

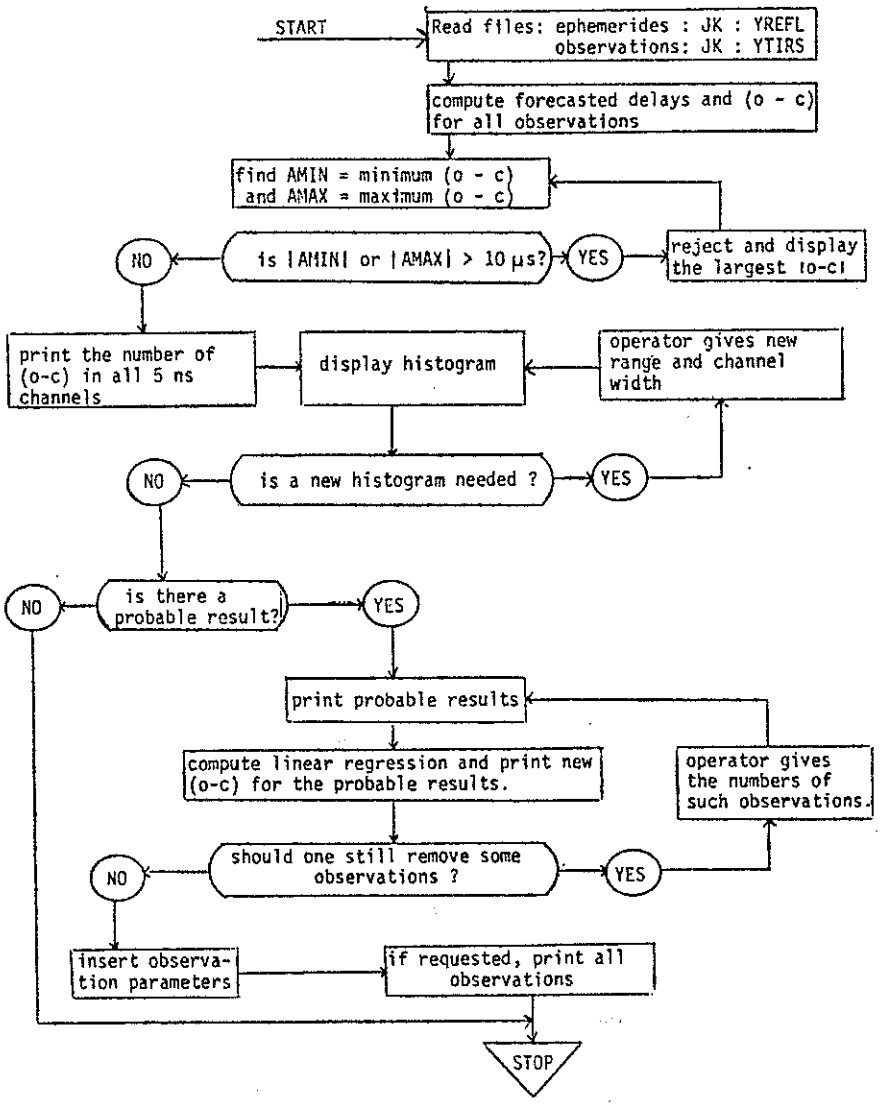


Figure 3: "TIRESU" flow-chart.

- The operator may request a change in the range of the histogram. The minimum and maximum (o - c) of the range and the number of rejected (but not suppressed) events falling outside the new range, the new channel width and finally, the new histogram itself are displayed.

This procedure can be iterated as many times as desired. The channel width is usually diminished to observe more details of the distribution of the residuals. It may also be widened, the "rejected" observations are then recovered.

- It the operator judges that, at a certain stage, there is a significant maximum in the histogram, the teletype prints out the events corresponding to the channel of the maximum in the last histogram displayed as well as to the four surrounding channels.

- A linear least squares fit of the (o - c) as function of the observation times is computed in the form {(o - c) = Y + X (T - TMM)} where T is the observation time and TMM is the mean observation time. X, Y, and TMM are printed, then the (o - c), the new residuals with respect to this regression straight line and their standard deviation.

- The operator may reject some of these observations if he judges that their residuals are not consistent with the others. Then the procedure starts again with the print out of the remaining events.

- When this iterative process is stopped by the operator, he must give the computer the observation parameters (crater used for tracking, pressure, temperature, calibration and mean noise). These are printed.

- The operator may also request a print out of all the observations.

4.2 Operating Procedures

The assembly of the program is done with the following statement: RLDR JK : <TIRESU WHIST> FORT.LB .

The ephemeris file "JK : YREFL" and the observation file "JK : YTIRS" must be prepared beforehand. Then, the procedure is dictated by the conversation with the display console. The actions to be taken by the operator start when the characteristics of the first histogram (minimum and maximum non-suppressed (o - c), extension and channel width of the histogram, number of suppressed events) are displayed and the teletype has started to print the number of (o - c) channel by channel.

1. APPUYER SUR "RESET" PUIS SUR 0 ET "RETURN": the operator is requested to clear the display unit for the histogram. Minimum and maximum (o - c) and the channel width are displayed and, then, the histogram itself.

2. FAUT-IL UN HISTOGRAMME PLUS FIN? OUI: FAIRE 1; NON: FAIRE 0: the operator is requested to state whether he wants a more detailed histogram.

3. If the answer is "YES"---> NUMERO DU PAS DU MAXIMUM?: the operator must give the two digits of the channel number around which the new histogram should be displayed.

4. VALEUR DU PAS ? FORMAT: ..., L'ANCIEN ETAIT DE: XXX NANOSECONDES: the operator is requested to give the channel width (in nanoseconds) he wishes for the new histogram. Then, the new minimum and maximum values for (o - c) as well as the number of rejected events are displayed. The operation is resumed at phase 1).

5. If the answer to the question 2) is "NO"---> Y-A-T-IL UN RESULTAT PROBABLE ? OUI: FAIRE 1; NON: FAIRE 0: If the answer is "NO", the program comes to its end. If it is "YES", all the observations corresponding to the maximum of the histogram and within two channels of this maximum are considered as probable. It is therefore necessary to optimize the last histogram range. After the list of events, the residuals with respect to the linear regression and the standard deviation are printed by the teletype, the operator must examine them and decide whether some of them appear to be abnormally large. These may be rejected in a new round.

6. FAUT-IL SUPPRIMER UN POINT? OUI: FAIRE 1; NON: FAIRE 0:

7. If the answer is "YES" ---> QUEL NO. ? FORMAT XXX: the operator gives the number of the observation to be rejected. The computer comes back to the question 6) as many times as it is necessary until the answer is "NO". Then it starts to print the remaining observations and computes the new linear regression. After this, a new iteration with question 6) is started.

8. If the answer to the question 6) is "NO" so that there is no rejection, the operator is asked a number of questions, the answers of which are necessary for the scientific use of the data.

9. NO ET NOM DU CRATERE SUIVI: XX AAAAAAAA. Number and name of the crater used for the tracking.

10. PRESSION : XXXX. Atmospheric pressure in millimeters of mercury.

11. TEMPERATURE : XXX. Temperature in degrees Celsius

12. CALIBRATION : XXX. Calibration in nanoseconds.

13. COMPTAGE DE BRUIT/SECONDE EN MILLIER D'EVT/SEC : XXX. Noise in kilohertz.

14. FAUT-IL IMPRIMER LES EVENEMENTS? OUI: FAIRE 1; NON :FAIRE 0. If the answer is "YES", all the observations are printed. If "NO" the program ends.

4.3 Test data

The following test data correspond to an actual observation that was made on July 7, 1981 by the CERGA lunar laser.

1. Ephemerides: The ephemeris data to be filed in "JK : YREFL" are given in table 1.

2. Observations: The observation data to be filed in "JK : YTIRS" are given in table 2.

```
 3
 7  7 31
13 30  0
 0.244479325000000  7
13 30  0.259923109327930  1  0.244479327033000  7  0.131915600  3  0.347990000  1
19  0  0.260196335666650  1  0.244479329167000  7  0.132033300  3  0.333490000  1
19 30  0.260511179133480  1  0.244479331250000  7  0.132259600  3  0.323960000  1
20  0  0.260861915954100  1  0.244479333333000  7  0.132440700  3  0.319400000  1
20 30  0.261244423999390  1  0.244479335417000  7  0.132631200  3  0.309939000  1
21  0  0.261653535654350  1  0.244479337500000  7  0.132830900  3  0.300240000  1
21 30  0.262089353631440  1  0.244479339583000  7  0.133040300  3  0.290650000  1
22  0  0.262523879913710  1  0.244479341667000  7  0.133260100  3  0.281030000  1
22 30  0.262998287543930  1  0.244479343750000  7  0.133490300  3  0.271520000  1
99  0  0.000000000000000  0  0.244479345000000  7  0.999999000  3  0.303990120-20
```

Table 1: Apollo 15 - 7 July 1981; File: "JK:YREFL"

```
.710418027361513D  5 .710444094443219D  5
.710418027361513D  5 .710444094447877D  5
.710663427100116D  5 .710689494675104D  5
.710724696681012D  5 .710750764381634D  5
.710756273755526D  5 .710812341576017D  5
.710970431653864D  5 .710996499803884D  5
.711215570180306D  5 .711241638793969D  5
.711215570180306D  5 .711241638850397D  5
.711276942359855D  5 .711303011134857D  5
.711276942359855D  5 .711303011137195D  5
.711338323174038D  5 .711364392071134D  5
.711461072770986D  5 .711487141961091D  5
.711522259919125D  5 .711548329175773D  5
.711645021103463D  5 .711671090596896D  5
.711767788546231D  5 .711793858283247D  5
.711767788546231D  5 .711793858285609D  5
.711828981145443D  5 .711855050950139D  5
.711890566598214D  5 .711916636575676D  5
.712013149731651D  5 .712039219910543D  5
.712136324498718D  5 .712162394974926D  5
.712136324498718D  5 .712162394957956D  5
.712197711393090D  5 .712223781917431D  5
.712259096258886D  5 .712285166889164D  5
.712320488971583D  5 .712346559792107D  5
.712320488971583D  5 .712346559736236D  5
.712504806496095D  5 .712530877662363D  5
.712504806496095D  5 .712530877708274D  5
.712565999587552D  5 .712592070090611D  5
.712565999587552D  5 .712592070842694D  5
.712627378376316D  5 .712653449747822D  5
.712627378376316D  5 .712653449772636D  5
.712688556869579D  5 .712714628413667D  5
.712688556869579D  5 .712714628405390D  5
.712872674636206D  5 .712898746542712D  5
.712872674636206D  5 .712898746547207D  5
.712872674636206D  5 .712898746572922D  5
.713056410283163D  5 .713082482551665D  5
.713178981103605D  5 .713205053613771D  5
.713362933074042D  5 .713389005947275D  5
.713455639028205D  5 .713511762121433D  5
.714037541951761D  5 .714063616108843D  5
.714344105483402D  5 .714370180298640D  5
.714650404302753D  5 .714676479726539D  5
.715385906153006D  5 .715411983041407D  5
.715385906153006D  5 .715411932993862D  5
.715447106269697D  5 .715473183286716D  5
.715630895439871D  5 .715656972791056D  5
.715692271832714D  5 .715718349348653D  5
.715692271832714D  5 .715718349332945D  5
.715876018480821D  5 .715902096325493D  5
.999900000000000D  6 .999900000000000D  6
```

Table 2: Observation data - 7 July 1981 (File: "JK:YTIRS"

3. Print-out of (o - c) channels: The matrix giving the number of events per 5 nanosecond channel as printed by the teletype is given in table 3. Sections with no event are not printed.

4. Histogram: We give, figures 4 to 7, the four histograms that are obtained if the following answers are successively given in the

CANAL, Ø-C NBRE D' EVENEMENTS PAR 5 NANØ

1	-5991	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
14	-4691	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
15	-4591	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
18	-4291	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0
20	-4091	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
22	-3891	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
23	-3791	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	
26	-3491	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
32	-2891	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
41	-1991	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
43	-1791	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
44	-1691	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
48	-1291	0	0	1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0
53	-791	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
54	-691	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
60	-91	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
62	109	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
64	309	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
67	609	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
68	709	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
70	909	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2	4	6	2	2
71	1009	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
73	1209	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
75	1409	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
77	1609	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
83	2209	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
86	2509	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
87	2609	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
100	3909	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
101	4009	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table 3: Number of events in each 5 nanosecond channel.

histogram conversion.

a. Fig. 4: histogram appearing in the first place.

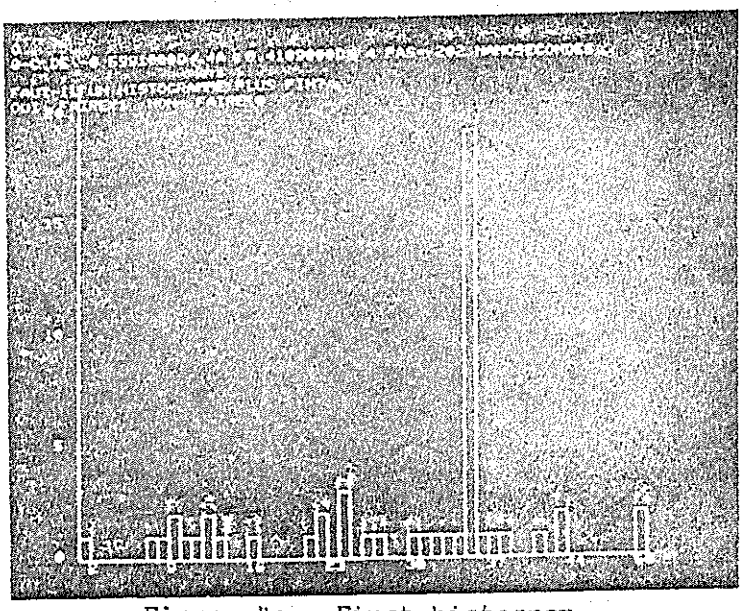


Figure 4: First histogram.

b. Request a new histogram. Central channel: 35; channel width: 150 ns. Figure 5 gives this second histogram.

c. Request a new histogram. Central channel: 24; channel width: 50 ns. Figure 6 gives this last histogram.

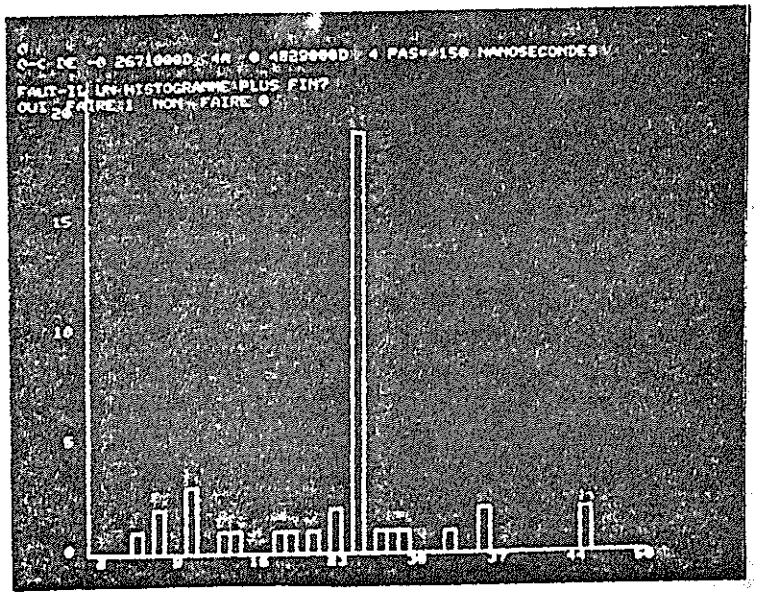


Figure 5: Second histogram.

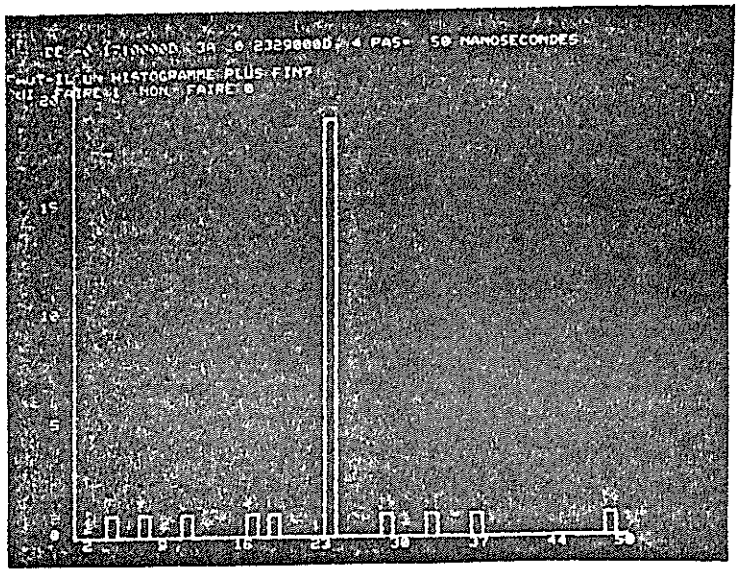


Figure 6: Third histogram

d. Request a new histogram. Central channel: 24; channel width: 15 ns. Figure 7 gives this last histogram.

After this, do not request a new histogram and indicate that there is a probable result. At this stage, the teletype prints 19 measurements, the coefficients of the linear regression, the residual and the mean quadratic error (table 4).

5. Answer "OUI" three times when rejection are requested. Reject the observations numbered 11, 24, and 44. Then answer that there are no more rejection. The teletype prints the 16 remaining measurements and the other data as given in 5). They are

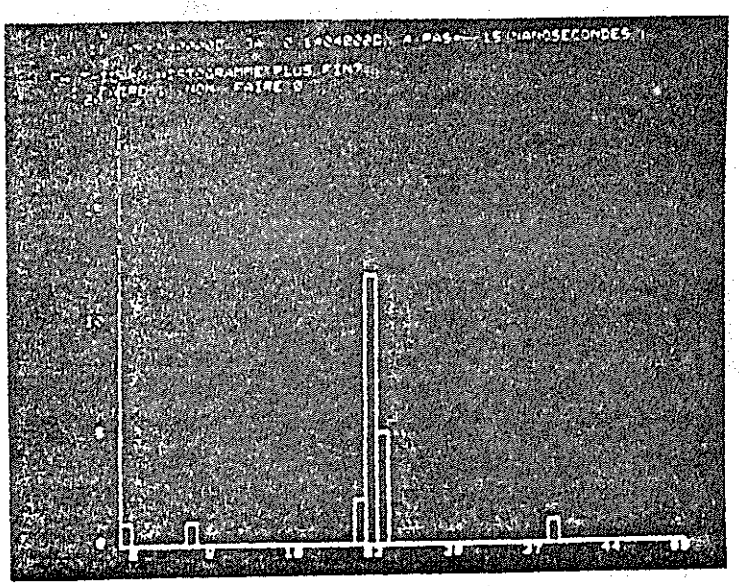


Figure 7: Last histogram.

RESULTATS DU 7 7 81 JJ A Ø H = 0.24447925D 7

NØ	J.JUL.		MESURE		J-C
4	0.2444793322596O3D	7	.260677006222D	1	0.98788D -6
5	0.2444793322669306	7	.260678204916	1	0.98550D -6
10	0.2444793323237206	7	.260687773406	1	0.99229D -6
11	0.2444793323308246	7	.260688970966	1	0.98105D -6
13	0.2444793323521136	7	.260692566486	1	0.99208D -6
15	0.2444793323805316	7	.260697370166	1	0.99323D -6
18	0.2444793323947416	7	.260699774626	1	0.99357D -6
21	0.2444793324231866	7	.260704592386	1	0.99563D -6
24	0.2444793324445016	7	.260708205246	1	0.98130D -6
28	0.2444793324729176	7	.260713030596	1	0.99730D -6
32	0.2444793324871016	7	.260715440886	1	0.99681D -6
34	0.2444793325084116	7	.260719065066	1	0.99867D -6
37	0.2444793325296776	7	.260722685026	1	0.99836D -6
38	0.2444793325438646	7	.260725101666	1	0.99612D -6
39	0.2444793325651546	7	.260728732336	1	0.10017D -5
42	0.2444793326787166	7	.260748154386	1	0.10058D -5
43	0.2444793327141676	7	.260754237866	1	0.10076D -5
44	0.2444793327992956	7	.260768884016	1	0.10004D -5
49	0.2444793328347546	7	.260775002316	1	0.10103D -5

X= 0.4570D-10 Y= 0.9932D -6 TM=0.712200D 5 OU=19 H47

NØ	Ø-C SEC	RESIDUS DTE
4	0.98788D -6	-0.14109D -8
5	0.98550D -6	0.12513D -8
10	0.99229D -6	-0.32995D -8
11	0.98105D -6	0.82193D -8
13	0.99208D -6	-0.19640D -8
15	0.99323D -6	-0.19983D -8
18	0.99357D -6	-0.17692D -8
21	0.99563D -6	-0.27111D -8
24	0.98130D -6	0.12464D -7
28	0.99730D -6	-0.24199D -8
32	0.99681D -6	-0.13646D -8
34	0.99867D -6	-0.23879D -8
37	0.99836D -6	-0.12377D -8
38	0.99612D -6	0.15652D -8
39	0.10017D -5	-0.31380D -8
42	0.10058D -5	-0.28207D -8
43	0.10076D -5	-0.31499D -8
44	0.10004D -5	0.73779D -8
49	0.10103D -5	-0.11561D -8

ERR. QUADR. MØYENNE = 0.4582D -8

Table 4: Print-out after 4th histogram; 19 events retained.

reproduced in table 5.

```
RESULTATS DU  7   7  81 JJ A 3 H =   0.2444792SD  7

NJ      J.JUL.                    MESURE                    O-C
  4 0.2444793322598603D  7    .260677006220  1          0.98788D -6
  5 0.2444793322669300D  7    .260678204910  1          0.98550D -6
 10 0.2444793323237200  7    .260687773400  1          0.99229D -6
 13 0.2444793323521130  7    .260692566480  1          0.99208D -6
 15 0.2444793323805310  7    .260697370160  1          0.99323D -6
 18 0.2444793323947410  7    .260699774620  1          0.99357D -6
 21 0.2444793324231860  7    .260704592380  1          0.99563D -6
 28 0.2444793324729170  7    .260713930590  1          0.99730D -6
 32 0.2444793324971010  7    .260715440880  1          0.99681D -6
 34 0.2444793325084110  7    .260719065060  1          0.99867D -6
 37 0.2444793325296770  7    .260722685020  1          0.99836D -6
 38 0.2444793325438640  7    .260725101660  1          0.99612D -6
 39 0.2444793325651540  7    .260728732330  1          0.10017D -5
 42 0.2444793326787160  7    .260748154380  1          0.10058D -5
 43 0.2444793327141670  7    .260754237860  1          0.10076D -5
 49 0.2444793328347540  7    .260775002310  1          0.10103D -5

X=  0.4745D-10 Y=  0.9949D -6 TH=0.712200D  5 OU=19 H47

NB   O-C SEC   RESIDUS DTE
  4   0.98788D -6   0.47492D-11
  5   0.98550D -6   0.26777D -8
 10   0.99229D -6  -0.17872D -8
 13   0.99208D -6  -0.40883D -9
 15   0.99323D -6  -0.40017D -9
 18   0.99357D -6  -0.14257D -9
 21   0.99563D -6  -0.10485D -8
 28   0.99730D -6  -0.68208D -9
 32   0.99681D -6   0.39471D -9
 34   0.99867D -6  -0.59633D -9
 37   0.99836D -6   0.58598D -9
 38   0.99612D -6   0.34103D -8
 39   0.10017D -5  -0.13106D -8
 42   0.10058D -5  -0.77171D -9
 43   0.10076D -5  -0.10472D -8
 49   0.10103D -5   0.11288D -8

ERR. QUADR. MOYENNE = 0.1450D -8
```

Table 5: Print-out after the rejection of obs. 11, 24, and 44.

6. The rest of the conversation is straightforward. Answer:

   - Reflector : 03
   - Crater : 09 POSIDO-A
   - Pressure : 0660
   - Temperature : 012
   - Calibration : 0198
   - Noise : 0050

   These numbers are printed as indicated in table 6.

7. If the print-out of events is requested, this is done as shown in table 7.

```
NØ REFLECTEU  :   3
CRATERE SUIVI NØ :  9    PØSID3-A
PRESSIØN :      660
TEMPERATURE :     12
CALIBRATIØN :     198
BRUIT :    50    KILOHERTZ
```

Table 6: Observation parameter print-out.

JJ	INSTANT DE TIR EN SEC.		A.R. EVENEMENT	
1	.7104150273615130	5	0.260670817060	1
2	.7104180273615130	5	0.260670864640	1
3	.7106634271001160	5	0.260675749880	1
4	.7107246966810120	5	0.260677006220	1
5	.7107862737555260	5	0.260678204910	1
6	.7109704316539640	5	0.260681500200	1
7	.7112155701800960	5	0.260686136630	1
8	.7112155701803060	5	0.260686700910	1
9	.7112769423598550	5	0.260687750020	1
10	.7112769423598550	5	0.260687773400	1
11	.7113383231740380	5	0.260688970760	1
12	.7114610727709360	5	0.260690901050	1
13	.7115222599191250	5	0.260692566480	1
14	.7116450211034630	5	0.260694934330	1
15	.7117677885462310	5	0.260697370160	1
16	.7117677885462310	5	0.260697393780	1
17	.7118239811454430	5	0.260698046960	1
18	.7118905665932140	5	0.260699774620	1
19	.7120131497316510	5	0.260701788920	1
20	.7121363244987180	5	0.260704762080	1
21	.7121363244987180	5	0.260704592380	1
22	.7121977113930900	5	0.260705243410	1
23	.7122590962539560	5	0.260706302780	1
24	.7123204889715830	5	0.260708205240	1
25	.7123204889715830	5	0.260707646530	1
26	.7125048064960250	5	0.260711662680	1
27	.7125048064960950	5	0.260712121790	1
28	.7125659995875520	5	0.260713030590	1
29	.7125659995875520	5	0.260712551420	1
30	.7126273783763160	5	0.260713715060	1
31	.7126273783763160	5	0.260713963200	1
32	.7126885568695790	5	0.260715440880	1
33	.7126885568695790	5	0.260715358110	1
34	.7128726746362060	5	0.260719065060	1
35	.7128726746362060	5	0.260719110010	1
36	.7128726746362060	5	0.260719367160	1
37	.7130564102831630	5	0.260722685020	1
38	.7131789811036050	5	0.260725101660	1
39	.7133629330740420	5	0.260728732330	1
40	.7134856890282050	5	0.260730932230	1
41	.7140375419517610	5	0.260741570820	1
42	.7143441054834020	5	0.260746154330	1
43	.7146504043027530	5	0.260754237960	1
44	.7153859061530060	5	0.260768884010	1
45	.7153859061530060	5	0.260768408560	1
46	.7154471062696970	5	0.260770170190	1
47	.7156308954398710	5	0.260773511850	1
48	.7156922718327140	5	0.260775159390	1
49	.7156922718327140	5	0.260775002310	1
50	.7158760184808210	5	0.260778446720	1

Table 7: Print-out of all events.

SAO Prediction and Data Review Algorithms

by

James H. Latimer
Smithsonian Astrophysical Observatory
Cambridge, MA 02138

```
C       CODE FOR GENERATION OF THE SAO 333 QUICK LOOK OBSERVATION MESSAGE
C       HAS THE FOLLOWING FORM:  FOR EACH PASS,
      CALL GENHED(ISTORE,PRECAL,ISTAT,DATE,SAT,ISKY,
     + IHUM,ITEMP,IPRESS,ICHECK,LEX,IENC)
C       FOR EACH OBSERVATION TO BE TRANSMITTED:
      CALL GENRAN(ISTORE,NTOTGOOD,NPUNCH,ICHECK,IENC,LEX,
     + RANGE,POSEC,JHR,JMIN,ISEC,ICONF)
C       AND AT THE END OF A PASS
        CALL NEWLINE(ISTORE,"<5><7>END<15><15><12><0>",1,1)
C       NOTE THAT SUBROUTINE NEWLINE, WHOSE DETAILS ARE NOT IMPORTANT,
C       IS THE DATA SINK ROUTINE
        COMPILER DOUBLE PRECISION
        SUBROUTINE GENHED(ISTORE,PRECAL,ISTAT,DATE,SAT,ISKY,
     + IHUM,ITEMP,IPRESS,ICHECK,LEX,IENC)
        DIMENSION LINE(30),ICHECK(2)
C
C GENERATE A HEADER TO BE PUT INTO THE STORAGE AREA
C FOR LATER PUNCHING
C
        ICHECK(1)=0
        ICHECK(2)=0
        CALL NEWLINE(ISTORE,"<1><15><15><15><12>..LASER<15><15><12>",1,1)
        CALL HEDENC(LEX,LINE,IENC,ISTAT,DATE,ICHECK)
        CALL NEWLINE(ISTORE,LINE,1,1)
        CALL SATENC(LEX,LINE,IENC,SAT,ISKY,IHUM,ITEMP,IPRESS,
     + PRECAL,ICHECK)
        CALL NEWLINE(ISTORE,LINE,1,1)
        END

        COMPILER DOUBLE PRECISION
        SUBROUTINE HEDENC(LEX,IAR,IENC,ISTAT,DATE,ICHECK)
        DIMENSION IAR(2)
C
C ENCODES THE FIRST HEADING LINE OF THE QUICK-LOOK MESSAGE
C
        CALL ENCODE(IAR,22)0029 WRITE(IENC,1) ISTAT,DATE
1       FORMAT("<2><24>33333 ",I4,F7.0"<15><15><12>")
C MOVE LOW 5 DIGITS OF DATE DOWN ONE SPACE, REMOVING "."
        I=19
10      IF(I.LE.14) GO TO 20
        CALL PUTC(IAR,I,IGETC(IAR,I-1))
        I=I-1
             1 OT OG
             20      CALL PUTC(IAR,14,40K)
             C ADD IN THE CHARACTERS TO THE CHECKSUM
                     CALL BCHECK(IAR(2),3,ICHECK)
        END

        COMPILER DOUBLE PRECISION
        SUBROUTINE SATENC(LEX,IAR,IENC,SAT,ISKY,IHUM,ITEMP,
     + IPRESS,CAL,ICHECK)
        DIMENSION IAR(2)
C
C ENCODES THE SATELLITE LINE DATA IN THE QUICK-LOOK FORMAT
```

```
C
        XCAL=CAL*10.
        CALL ENCODE(IAR,40)
        WRITE(IENC,1) SAT,ISKY,IHUM,ITEMP,IPRESS,XCAL
1       FORMAT(" <3><46>"F8.0,I1,I2,I5,I6,F7.0" 00000<15><15><12>")
C MOVE PIECES OF FLOATING PT#'S, TO REMOVE DECIMAL PTS.
        DO 20 J=10,31,21
        I=J
        K=8
        IF(I.EQ.31) K=26
10      IF(I.LE.K) GO TO 20
        CALL PUTC(IAR,I,IGETC(IAR,I-1))
        I=I-1
        GO TO 10
20      CONTINUE
C MAKE SURE LEADING ZEROES WRITTEN
        DO 110 I=3,37
110     IF(IGETC(IAR,I).EQ.40K) CALL PUTC(IAR,I,60K)
C PUT SPACING IN
        DO 120 I=8,32,6
120     CALL PUTC(IAR,I,40K)
        CALL BCHECK(IAR(2),6,ICHECK)
        END

        COMPILER DOUBLE PRECISION
        SUBROUTINE GENRAN(ISTORE,NTOTGOOD,NPUNCH,ICHECK,IENC,
     + LEX,RANGE,POSEC,JHR,JMIN,ISEC,ICONF)
C
C GENERATE THE RANGE LINE TO POSSIBLY BE INSERTED INTO THE
C STORAGE AREA FOR LATER PUNCHING
C
        DIMENSION LINE(30)
C
        CALL RANENC(LEX,LINE,IENC,JHR,JMIN,ISEC,POSEC,ICONF,RANGE,ICHECK)
        CALL NEWLINE(ISTORE,LINE,NTOTGOOD,NPUNCH)
        END

        COMPILER DOUBLE PRECISION
        SUBROUTINE RANENC(LEX,IAR,IENC,IHR,IMIN,ISEC,POSEC,
     + ICONF,RANGE,ICHECK)
        DIMENSION ICHECK(2),IAR(2)
C
C ENCODES AND PUNCHES THANGE LINE FOR THE QUICK-LOOK MESSAGE
C
        CALL ENCODE(IAR,34)
        JSEC=ISEC/10
        KSEC=MOD(ISEC,10)
        SEC=POSEC*1.E6
        RAN=RANGE*10.
        WRITE(IENC,1) IHR,IMIN,JSEC,KSEC,SEC,ICONF,RAN
1       FORMAT(" <4><40>",2I2,I1,I2,F7.0,I3,F12.0"<15><15><12>")
C MOVE PIECES OF THE FLOATING PT NUMBERS, REMOVING DECIMAL PTS.
        DO 20 J=16,31,15
        I=J
```

```
        K=14
        IF(I.EQ.31) K=26
10      IF(I.LE.K) GO TO 20
        CALL PUTC(IAR,I,IGETC(IAR,I-1))
        I=I-1
        GO TO 10
20      CONTINUE
C MAKE SURE LEADING ZEROES WRITTEN
        DO 110 I=3,32
110     IF(IGETC(IAR,I).EQ.40K) CALL PUTC(IAR,I,60K)
C PUT IN SPACING
        DO 120 I=8,31,6
120     CALL PUTC(IAR,I,40K)
C ADD IN FINAL CHECKSUM, INSERT INTO LINE TO PUNCH
        CALL BCHECK(IAR(2),5,ICHECK)
        CALL PUTC(IAR,17,ICHECK(1)+60K)
        CALL PUTC(IAR,18,ICHECK(2)+60K)
C RESET CHECKSUM TO START OVER
        ICHECK(1)=0
        ICHECK(2)=0
        END

        COMPILER DOUBLE PRECISION
        SUBROUTINE BCHECK(IAR,N,ICHECK)
        DIMENSION ICHECK(2)
C
C SUBROUTINE TO TAKE N GROUPS OF 5 CHARACTERS, SEPARATED WIT
HC A BLANK, AND BUILD A 2 DIGIT 10'S MODULUS CHECKSUM WITH EACH
C LINE. ANY CHARACTERS LESS THAN 60 OCTAL (CHARACTER ZERO)
C ARE CHECKSUMMED AS ZERO.
C
        NCHAR=N*6
        J=1
        DO 200 I=1,NCHAR
        IF(MOD(I,6).EQ.0) GO TO 200
        J=J+1
        L=MOD(J,2)+1
        ICHECK(L)=MOD(ICHECK(L)+MAX0(0,IGETC(IAR,I)-60K),10)
200     CONTINUE
        END

C       CODE TO READ THE SAO333 QL FORMAT AND THE NASA QL FORMAT
      SUBROUTINE OBSCARD
C
c*****************************************************************
c
c       Obscard initiates passes. It searches the DATA RECORDS
c       until a pass identifier record is found. Then it
c       calls the appropriate subroutine to process that TYPE OF
c       observation. PASSES MAY ALSO BE INITIATED WITHIN EACH DATA
C       TYPE (IN LASERA, NASA, OR BAKER NUNN). COMMENTS IN RERUN
C       FILES (ALWAYS PREFACED BY ....) ARE DELETED. LINE_FEEDS
C       IN FIRST CHARACTER POSITION ARE DELETED. SPECIAL LOGIC
C       ENSURES THAT "UNDETECTED" WETTZELL PASSES ARE REJECTED IN FULL
```

```
C       TO AVOID THE OBSERVATION TIME BIAS PROBLEM.

c
c*****************************************************************
c
      COMMON /PARAM/ FORMIN,NOBJS,OBJ1(100),OBJ2(100),LOBS(100),
     1NOBSV(100)
      COMMON /LASER/ SAT(25),NUMSAT,PASS(25),SUM(25)
      COMMON/LIMIT/ISTA(25),PRLIM(25,2),TMPLIM(25,2),CALLIM(25,2,2),
     1            RANLIM(25,2),EXPDATE(25)
        common/seq/passeq,saonum(25),nasnum(25),nnassat,nnassta,
     1          TRNFLG,WNDFLG,OBSONE,BADHFLG,PASSFLG,NSEQ
        common/nasac/nassat(25),saosat(25),nassta(25),saosta(25)
      common/lascoc/RCRD,pcar
        COMMON/WETZCOM/WCAL,ICORR,WETZFLG,WFLAG,RAN2WT
        DATA BLANK/'     '/
      character*80 RCRD,pcar*30,BLANK*5
        CHARACTER*4 nassat,saosat*7,nassta,saosta
      logical*1 WETZFLG(3),WFLAG,WNDFLG,PASSFLG
        double precision WCAL,RAN2WT
      INTEGER FORMIN,OBJ1,OBJ2,BUFF(8)
      INTEGER PASS,SUM,sat,SAONUM
        integer*2 passeq,OBSONE,BADHFLG
C
C-----INITIALIZE FOR PROCESSING OBSERVATIONS
C
      DO 6 J=1,NOBJS
      NOBSV(J)=0
 6    CONTINUE
      NGARB=0
        LPASS=0
        WFLAG=.FALSE.
c
c------parse observation RECORDS for an identifier and transfer to
c-----the appropriate pass processing subroutine.
c
 100  READ(2,101,err=200)RCRD
 101  FORMAT(A)
      NGARB=NGARB+1
        J=INDEX(RCRD,'....')    ! DELETE COMMENT RECORDS (RERUNS)
        IF (J.NE.0) GOTO 100
121     continue
        NDETFLG=0    ! FIND NASA OBS WITHOUT PROPER HEADER
        IF (RCRD(1:1).NE.CHAR(10)) GOTO 123    ! DELETE A LINE_FEED
        RCRD(1:80)=RCRD(2:80)//BLANK(1:1)
123     J=INDEX(RCRD,'33333')
        IF (J-1) 153,152,151
151     RCRD(1:80)=RCRD(J:80)//BLANK(1:J-1)    ! LEFT SHIFT RECORD
152     IF (LASTC(RCRD,80).LE.8) GOTO 103      ! WETTZELL PASS DETECTED
        IF (RCRD(J+6:J+9).NE.'7834') GOTO 106     ! SAO PASS DETECTED
        WRITE(11,902) NGARB-1    ! UNCORRECTABLE WETTZELL PASS
        NGARB=0
        WRITE(11,162) RCRD(1:20)
162     FORMAT(A20/'....WETTZELL PASS UNDETECTED OR NO CORRECTIONS')
```

```
        CALL WNCOREJ(RCRD,IFLG)
        NGARB=0
        IF (IFLG.EQ.1) GOTO 121
        GOTO 100
153     IF (INDEX(RCRD,'LASERQL')) 154,154,109   ! NASA PASS DETECTED
154     IF (INDEX(RCRD,'..LASERQL')) 155,155,109 ! NASA PASS DETECTED
155     IF (INDEX(RCRD,'LASER QL')) 156,156,109  ! NASA PASS DETECTED
156     if(lastc(RCRD,80).eq. 0) go to 100
        J=INDEX(RCRD,'END')
        IF (J.EQ.0.OR.J.GT.2) GOTO 157
        WRITE(11,101) RCRD   ! END OF AN UNLABELED PASS (REJECTED)
        WRITE(11,161) NGARB
161     FORMAT('....NUMBER OF GARBAGE RECORDS =',I3)
        NGARB=0
        GOTO 100
157     IF (RCRD(1:3).NE.'1BB') GOTO 111
        NDETFLG=1
        GOTO 109
111     WRITE(11,101) RCRD     ! GARBAGE RCRDS DUMPED
        IF (LPASS.EQ.PASSEQ) GOTO 100
        LPASS=PASSEQ   ! SUPPRESS SEQUENCES OF REJECT MESSAGES
        WRITE(11,102)
102     FORMAT('....REJECT/ GARBAGE RECORD')
        GO TO 100
C
C       CALL WETTZELL STATION
C
103     IF (NGARB.GT.5) WRITE(11,902) NGARB-1
902     FORMAT('....NUMBER OF GARBAGE RECORDS = ',I4)
        CALL WETZEL(RCRD,*100)   ! PROCESS CORRECTIONS FOR WETTZELL OBSERVATIONS
        GOTO 107
120     ngarb=0
        LPASS=0
        goto 121
C
c       call lasera
C
106     IF (NGARB.GT.5) WRITE(11,902) NGARB-1
107     CALL LASERA(*120)   ! PROCESS SAO LASER OBSERVATIONS
        NGARB=0
        LPASS=0
        WFLAG=.FALSE.
        GO TO 100
C
C     CALL NASA
C
109     IF (NGARB.GT.5) WRITE(11,902) NGARB-1
        CALL NASA   ! PROCESS NASA LASER OBSERVATIONS
        NGARB=0
        LPASS=0
        GO TO 100
C
C...PROCESSING TERMINATES NORMALLY ONLY WHEN FOR002.DAT IS EMPTY
C
```

```
200     CONTINUE
        RETURN
        END

        SUBROUTINE LASERA(*)
c
c*****************************************************************
c
c       lasera reads the observation records for a pass. each
c       record is parsed for type, i.e. date(header), satellite
c       (header),or range and related observations. the
c       appropriate processing subroutine is called for each
c       record. EACH DATE RECORD INITIATES A NEW PASS.
C       COMMENT RECORDS (....) ARE DELETED FROM RERUN FILES.
C       PART OF THE FIRST OBSERVATION AND THE NUMBER OF OBSERVATIONS
C       IN THE ORIGINAL DATA SET ARE CARRIED THROUGH MULTIPLE RERUNS
C       BY "___" CHARACTERS. RECORDS ARE SHIFTED LEFT IF NECESSARY.
C       A PASS IS REJECTED IF THERE IS ANY ERROR IN A HEADER RECORD.
C       AN OBSERVATION IS REJECTED IF THERE IS ANY ERROR IN THE
C       GIVEN OBSERVATION RECORD. WETTZELL OBSERVATIONS ARE TIME-
C       SHIFTED AND TIME UNBIASED.
c
c*****************************************************************
c
        common/seq/passeq,saonum(25),nasnum(25),nnassat,nnassta,
     1          TRNFLG,WNDFLG,OBSONE,BADHFLG,PASSFLG,NSEQ
       COMMON/LASER/SAT(25),NUMSAT,PASS(25),SUM(25)
       COMMON/LASCOM/CODE(4),DECADE,DT,C,Y,D,M,HR,MIN
     1  ,SEC,IDENT,ILLCODE,IFIRST,JSTA,NOGOOD,OBS,PR,RANGE,RELHUM,
     2     STAT,TEMP,PRESS,HUM,TCENT,JSAT,TM,CAL,OBSFLAG
     3  ,ACOEFF,BCOEFF,PRECAL,POSTCAL,ALOGSS,DIFF,NDIFF,PPCAL
     4    ,NCHKSM1,NCHKSM2
       common/lascoc/card,pcar
       COMMON/SOLVE/DATER(14),STORE(43),WORK(115),ARRAY(6),NRANGE
       COMMON/BLOCK/ IBLOCK(800)
       COMMON/SWITCH/GO,SW1,SW2,KILL1,KILL2
       COMMON /PNTS/ IPNTS,NUMPTS,NUMGPTS
         COMMON/WETZCOM/WCAL,ICORR,WETZFLG,WFLAG,RAN2WT
         DATA BLANK/'     '/
       INTEGER DELAY,SAONUM,FLAG,GO,HR,OBS
       INTEGER PASS,SAT,SEC,SUM,SW1,SW2,TEMP
       logical*1 WETZFLG(3),WFLAG,OBSFLAG(17),WNDFLG,PASSFLG
         integer*2 passeq,OBSONE,BADHFLG
       LOGICAL CHKCHR
       LOGICAL TMPCHK,PRCHK,CALCHK,RANCHK
       character*30 pcar
       character*80 card
         character*10 itest,BLANK*5
         double precision r,range,WCAL,RAN2WT
       INTEGER CODE
C
C-----INITIALIZE
C
       C=2.997925E8
```

```
        IRRFLG=0    ! RERUN FLAG (0,1,2)
      ITEST='          '
      IPNTS=0
      NUMPTS=0
      NUMGPTS=0
      KILL1=0
      KILL2=0
      SW1=0
      SW2=0
      GO=0
      SW2=-1
      do 500 j=1,17
500   obsflag(j)=.false.
      IFIRST=0
      NCHKSM1=0
      NCHKSM2=0
      PASSFLG=.FALSE.
      go to 110
C
C-----READ A RECORD AND DETERMINE ITS DISPOSITION
C
  100 CONTINUE
      obsflag(17)=.false.
      READ(2,3,err=250,END=800)card
    3 FORMAT(a)
      J=INDEX(CARD,'....')
      IF (J.NE.0) GOTO 100    ! DELETE COMMENT RECORDS
106   J=INDEX(CARD,'____')
      IF (J.EQ.0) GOTO 107    ! SAVE SPECIAL COMMENTS OVER MULTIPLE RERUNS
      CALL REJREC(CARD)
      IRRFLG=IRRFLG+1
      OBSONE=1
      GO TO 100
107   IF (CARD(1:1).NE.CHAR(10)) GOTO 108    ! DELETE LINE_FEED IN 1ST LOCATION
      J=LASTC(CARD,80)
      CARD(1:J-1)=CARD(2:J)
      CARD(J:J)=' '
      GOTO 106
108   J=0
      DO 102 I=1,5
        IF (CARD(I:I).NE.' ') GOTO 103
102     J=J+1
103     IF (J.GT.0) THEN    ! SHIFT LEFT IF NECESSARY
        CARD(1:80-J)=CARD(J+1:80)
        CARD(81-J:80)=BLANK(1:J)
        ENDIF
      IF(card(1:5).EQ.'     ') GOTO 100      ! DELETE BLANK RECORDS
      IF(card(1:3).EQ.'END') GOTO 250        ! OBS. PASS COMPLETED
        IF (CARD(1:5).NE.'33333') GOTO 110
        IW=0
        IF (LASTC(CARD,45).NE.5) GOTO 405
        IW=1
        GOTO 410
405     IF (LASTC(CARD,45).NE.17) GOTO 110
```

```
        IF (WFLAG) GOTO 110
410     IF (IRRFLG.GT.1) GOTO 422
        WRITE(11,411) NSEQ
411     FORMAT('___NUMBER OF OBSERVATIONS IN PASS =',I3/'END')
422     CALL INCPASS
        IF (IW.EQ.1) GOTO 300
c       squeeze out blanks, go from 35 to 30 characters in pcar
110     pcar=card(1:5)//card(7:11)//card(13:17)//card(19:23)//
        1card(25:29)//card(31:35)
C
C-----SETTER SPECIFIES RECORD TYPE (BASED ON THE NUMBER OF DIGITS)
C-----VIA THE PARAMETER JUMP
C-----JUMP   RECORD TYPE
C       1       DATE (AND STATION)
C       2       SATELLITE (AND ITS ENVIRONMENTAL PROPERTIES)
C       3       RANGE (AND OBSERVATION TIME AND RELATED VALUES)
C       4,5     GARBAGE RECORD
C
      CALL SETTER(JUMP,OBSFLAG,wflag)
C
      GO TO (130,130,130,300,200),JUMP
300     WRITE(11,253)
253     FORMAT('END')
        return 10

C-----SETSW SETS THE SWITCHES, AND GIVES THE PLACE TO JUMP AT THE END
C
130 CALL SETSW(JUMP)
      IF( JUMP.LT.3 .AND. IPNTS.NE.0 ) GOTO 260
      GO TO (160,170,180,200,200),JUMP
C
C-----PROCESS THE DATE RECORD
C
  160 CALL DATECRD
      GO TO 100
C
C-----PROCESS THE SATELLITE RECORD
C
  170 CALL SATCRD
      GO TO 100
C
C-----PROCESS THE RANGE RECORD
C
  180 CALL RANCRD
      GO TO 100
C
C-----REJECT GARBAGE RECORDS TO RERUN FILE
C
200     WRITE(11,3) card
        CALL REJERS(OBSFLAG,17,17)
      GO TO 100
C
C-----SAVE THE TOTAL NUMBER OF OBSERVATIONS IN ORIGINAL DATA SET
C
```

```
250     IF (WNDFLG) WNDFLG=.FALSE.
        IF (IRRFLG.GT.1) GOTO 252
        WRITE(11,259) NSEQ
259     FORMAT('___NUMBER OF OBSERVATIONS IN PASS =',I3)
252     WRITE(11,3) CARD
         CALL INCPASS
C       IF( IPNTS.EQ.0 ) RETURN
    JUMP=3
260     CONTINUE
      IPNTS=0
        NUMPTS=0
        NUMGPTS=0
      GOTO (160,170,999), JUMP
800     WRITE(11,259) NSEQ
999     RETURN
      END
       SUBROUTINE DATECRD
c
c*************************************************************
C
c       process date records. after checking for illegal characters
c       convert station, year, month, and day to integer
c       type, and CHECK that each is a legitimate value.
c       APPROPRIATE obsflag element is set .TRUE. if illegal station
C       or date COMPONENT is found.
c
c*************************************************************
c
      COMMON/SOLVE/DATER(14),STORE(43),WORK(115),ARRAY(6),NRANGE
        common/seq/passeq,saonum(25),nasnum(25),nnassat,nnassta,
        1          TRNFLG,WNDFLG,OBSONE,BADHFLG,PASSFLG,NSEQ
      COMMON/LASER/SAT(25),NUMSAT,PASS(25),SUM(25)
      COMMON/LASCOM/CODE(4),DECADE,DT,C,Y,D,M,HR,MIN
     1  ,SEC,IDENT,ILLCODE,IFIRST,JSTA,NOGOOD,OBS,PR,RANGE,RELHUM,
     2     STAT,TEMP,PRESS,HUM,TCENT,JSAT,TM,CAL,OBSFLAG
     3  ,ACOEFF,BCOEFF,PRECAL,POSTCAL,ALOGSS,DIFF,NDIFF,PPCAL
     4,NCHKSM1,NCHKSM2
        common/lascoc/card,pcar
      COMMON/BLOCK/IBLOCK(600)


      COMMON/SWITCH/GO,SW1,SW2,KILL1,KILL2


      COMMON/YEAR/IMO,IDY,IYEAR,IFYEAR


        COMMON/WETZCOM/WCAL,ICORR,WETZFLG,WFLAG,RAN2WT
        DATA DPMO/31,28,31,30,31,30,31,31,30,31,30,31/
      logical*1 WETZFLG(3),WFLAG,OBSFLAG(17),WNDFLG,PASSFLG
        character*80 card
        character*30 pcar
        double precision r,range,WCAL,RAN2WT
        INTEGER*2 PASSEQ,OBSONE,DPMO(12),BADHFLG
```

```
      INTEGER CODE,Y,D
      INTEGER SNUM,DECade,stat
      INTEGER PASS,SAT,SEC,SUM,SW1,SW2,TEMP
      INTEGER DELAY,SAONUM,FLAG,GO,HR,OBS
      LOGICAL CHKCHR
C
        DO 5 I=1,17
        OBSFLAG(I)=.FALSE.
5       CONTINUE
        BADHFLG=0
        OBSONE=0
        J=LASTC(CARD,40)
      WRITE(11,10) card(1:J)
   10 FORMAT(a)
C
C-----COMPUTE CHECKSUMS
C
30    DO 32 I=1,15,2
32    NCHKSM1=NCHKSM1 + ichar(pcar(i:i))-48
      DO 34 I=2,14,2
34    NCHKSM2=NCHKSM2 + ichar(pcar(i:i))-48
  100 CONTINUE
C
C-----CHECK STATION NUMBER
C
        IF(PCAR(6:6).EQ.' ') PCAR(6:6)='0'
      IF(CHKCHR(pcar,6,9)) GO TO 210
      JSTA=ICONV(pcar,6,9)
C      IF WETTZELL STATION, VERIFY IT WAS DETECTED IN OBSCARD, ELSE
C      THE CALIBRATION AND TIME SHIFT (AND CORRECTION) WOULD NOT BE MADE.
C     IF NOT DETECTED, ALL RAW WETTZELL DATA WILL BE AVAILABLE ON FORO11.DAT
C      FILE.
        IF (JSTA.NE.7834) GOTO 45
        IF (JSTA.EQ.7834.AND.WFLAG.EQ..TRUE.) GOTO 45
        WNDFLG=.TRUE.
        GOTO 1000
45      Y=0
        M=0
        D=0
      CALL TARGDIS(JSTA,DIS,Y,M,D)
C
C-----IF DIS=-99., THEN THE STATION WAS NOT FOUND
C
      IF(DIS.EQ.-99.) go to 210
        stat=jsta
 115 CONTINUE
C
C-----CHECK FOR ILLEGAL MONTH.
C
      IF(CHKCHR(pcar,12,13)) GO TO 118
      M=iconv(pcar,12,13)
      IF(M.le.12.and.M.ge.1) go to 120
  118 OBSFLAG(14)=.true.
      GO TO 120
```

```
C
C-----CHECK ILLEGAL YEAR DATE
C
 120 CONTINUE
      IF(CHKCHR(pcar,10,11)) GO TO 125
      DECADE=ichar(pcar(10:10))-48
      IF(decade.EQ.0.and.pcar(11:11).gt.'6' ) DECADE=7
      IF(decade.EQ.0 .AND.pcar(11:11).LE.'6') DECADE=8
        pcar(10:10)=char(decade+48)
      Y=iconv(pcar,10,11)
      CALL YRCHK(Y,M,OBSFLAG)
      IF(.not.OBSFLAG(15)) go to 130
C     year change problem causes no calc of sec, obsin rancrd
125    OBSFLAG(15)=.true.
  130 CONTINUE
C
C-----CHECK FOR ILLEGAL DAY.
C
      IF(CHKCHR(pcar,14,15)) GO TO 135
      D=iconv(pcar,14,15)
        K=0
        ITM=Y-76
        IF ((ITM/4)*4.EQ.ITM) K=1
        IF (OBSFLAG(14)) GOTO 999
        IF(M .EQ. IMO .AND. D .GT. IDY)GO TO 135
      IF(D.le.DPMO(M)+K.and.D.ge.1) go to 999
  135 OBSFLAG(13)=.true.
200   GO TO 999
210   OBSFLAG(16)=.true.
      GO TO 115
C
  999 CONTINUE
        do 500 i=13,17
        if(obsflag(i)) BADHFLG=1
500     CONTINUE
        IF (BADHFLG.EQ.1) GOTO 1000
        IF (WFLAG) CALL OBSTMSH
1000    CALL REJERS(OBSFLAG,13,17)
        RETURN
        END

      SUBROUTINE SATCRD
C
c*****************************************************************
c
C-----process a SATELLITE record (HEADER). convert ASCII DATA FIELDS TO
C-----VARIABLES (satellite identification,skycode, humidity,
c-----temperature, pressure and calibration).  IF ANY ERROR,
C-----ENTIRE PASS IS REJECTED.
C               OBSFLAG(I)      MEANING IF TRUE
C                       7       BAD CALIBRATION
C                       8       BAD PRESSURE
C                       9       BAD TEMPERATURE
C10     BAD HUMIDITY
```

```
C                       11       BAD SKYCODE
C                       12       BAD SATELLITE IDENTIFICATION

C
c*****************************************************************
c
        common/seq/passeq,saonum(25),nasnum(25),nnassat,nnassta,
     1          TRNFLG,WNDFLG,OBSONE,BADHFLG,PASSFLG,NSEQ
       COMMON/LASER/SAT(25),NUMSAT,PASS(25),SUM(25)
       COMMON/LASCOM/CODE(4),DECADE,DT,C,Y,D,M,HR,MIN
     1  ,SEC,IDENT,ILLCODE,IFIRST,JSTA,NOGOOD,OBS,PR,RANGE,RELHUM,
     2     STAT,TEMP,PRESS,HUM,TCENT,JSAT,TM,CAL,OBSFLAG
     3  ,ACOEFF,BCOEFF,PRECAL,POSTCAL,ALOGSS,DIFF,NDIFF,PPCAL
     4    ,NCHKSM1,NCHKSM2
        common/lascoc/card,pcar
       COMMON/SOLVE/DATER(14),STORE(43),WORK(115),ARRAY(6),NRANGE
       COMMON/BLOCK/IBLOCK(600)
       COMMON/SWITCH/GO,SW1,SW2,KILL1,KILL2
       COMMON/LIMIT/ISTA(25),PRLIM(25,2),TMPLIM(25,2),CALLIM(25,2,2),
     1            RANLIM(25,2),EXPDATE(25)
        COMMON/CONN/CON(25)
        CHARACTER*1 CON
        character*80 card
        character*30 pcar
       double precision r,range
        integer*2 passeq,OBSONE,BADHFLG
        INTEGER TMUNPTS,TMUNGPTS
       INTEGER ALPHA,DELAY,SAONUM,FLAG,GO,HR,OBS
       INTEGER PASS,SAT,SEC,SUM,SW1,SW2,TEMP,stat
       INTEGER CODE,Y,D
       logical*1 OBSFLAG(17),WNDFLG,PASSFLG
       LOGICAL TMPCHK,PRCHK,CALCHK,RANCHK,chkchr
       INTEGER PSURE
C
        J=LASTC(CARD,40)
       WRITE(11,20) card(:J)
   20 FORMAT(a)
       PRECAL=0.
        POSTCAL=0.
       DIFF=0.0
        do 500 i=1,17
500     obsflag(i)=.false.
        IF (WNDFLG) GOTO 400
C
C-----COMPUTE CHECKSUMS
C
   30 DO 32 I=1,29,2
   32 NCHKSM1=NCHKSM1 +ichar(pcar(i:i))-48
       DO 34 I=2,30,2
   34 NCHKSM2=NCHKSM2+ichar(pcar(i:i))-48
   36 CONTINUE
C
C-----CHECK FOR BAD SATELLITE IDENTIFICATION
C
```

```
      IF(.not.CHKCHR(pcar,1,7)) go to 100
      GO TO 118
  100 IDENT=ICONV(pcar,1,7)
      DO 115 i=1,NUMSAT
      jsat=i
      IF(SAT(i).EQ.IDENT) GO TO 120
      jsat=0
  115 CONTINUE
  118 OBSFLAG(12)=.true.
  120 CONTINUE
C
C-----GET TARGET DISTANCE IF POSSIBLE
C
      IF(OBSFLAG(16)) GO TO 300
      CALL TARGDIS(JSTA,decade,Y,M,D)
      IF(decade.EQ.-99.) GO TO 300
  125 CONTINUE
C
C-----CHECK FOR BAD SKYCODE
C
      ILLCODE=ichar(pcar(8:8))-48
      IF(ILLCODE.LE.2.and.ILLCODE.GE.0) GO TO 130
      OBSFLAG(11)=.true.
  130 CONTINUE
C
C----CHECK FOR BAD HUMIDITY CHARACTERS
C
      RELHUM=.50
        iHUM=50
      IF(CHKCHR(pcar,9,10)) GO TO 135
      iHUM=iconv(pcar,9,10)
        hum=floatj(ihum)
    IF(HUM.LT.0.0.OR.HUM.GT.100.0) GO TO 135
        relhum=hum/100.
      GO TO 140
  135 OBSFLAG(10)=.true.
  140 CONTINUE
C
C-----CHECK FOR BAD TEMPERATURE
C
      IF(CHKCHR(pcar,11,14)) GO TO 145
        IF (PCAR(11:11).GE. '2') GOTO 145
      TEMP=ICONV(pcar,11,14)
C
C-----pcar(11:11) IS 0 OR 1 DEPENDING ON + OR - TEMPERATURE, SO THESE STEPS
C-----CONVERT THIS TO THE CORRECT TEMPERATURE.
C
      K=1000-TEMP
      TCENT=FLOAT(TEMP)/10.
      IF(K.LE.0) TCENT=FLOAT(K)/10.
C
C-----PUT SIGN ON THE TEMPERATURE
C
      pcar(11:11)=' '
```

```
      IF(K.LT.0) pcar(11:11)='-'
      IF(OBSFLAG(16)) GO TO 150
        IF(.NOT. TMPCHK(TCENT,JSTA))GO TO 145
142     temp=tcent
      GO TO 150
  145 TCENT=25.
      OBSFLAG(9)=.true.
C
C-----CHECK FOR BAD PRESSURE CHARACTERS
C
150     CONTINUE
      IF(CHKCHR(pcar,16,19)) GO TO 155
        pr=floatj(iconv(pcar,16,19))
        IF(.NOT. PRCHK(PR,JSTA))GO TO 155
152     press=pr
        go to 160
  155 PR=1000.
      OBSFLAG(8)=.true.
        GOTO 152
C
C-----CHECK FOR BAD CALIBRATION CHARACTERS
C
160     CONTINUE
        IF(CHKCHR(pcar,20,25)) GO TO 164
        IF(CHKCHR(pcar,26,30)) GO TO 164
        DO 162 I=1,25
        IF (JSTA.EQ.ISTA(I)) GOTO 163
162     CONTINUE
        GOTO 164
163     K=I
        IF (CON(K).EQ.'X') K=K+1
        tm1=float(iconv(pcar,20,25))/10.
        tm2=float(iconv(pcar,26,30))/10.
        IF(JSTA .EQ. 7835) THEN
                        TM1 = TM1*2.0
                        TM2 = TM2*2.0
                ENDIF


        IF(OBSFLAG(16)) GOTO 164
C
        IF (TM1.EQ.0.) THEN
                IF (TM2.EQ.0.) GOTO 167 ! TM1=0.  TM2=0.
                GOTO 166                ! TM1=0.  TM2>0.
        ELSE
    IF (TM2.EQ.0.) GOTO 165   ! TM1>0.  TM2=0.
                GOTO 167                ! TM1>0.  TM2>0.
        ENDIF
C
  164   TM=1000.
        OBSFLAG(7)=.true.
        GOTO 170
165     IF (TM1.LT.CALLIM(K,2,1).OR.TM1.GT.CALLIM(K,1,1)) GOTO 164
        TM=TM1
```

```
        GOTO 170
166     IF (TM2.LT.CALLIM(K,2,1).OR.TM2.GT.CALLIM(K,1,1)) GOTO 164
        TM=TM2
        GOTO 170
167     TM2=float(ICONV(pcar,20,20))*10000.+TM2
        IF (TM1.LT.CALLIM(K,2,1).OR.TM1.GT.CALLIM(K,1,1)) GOTO 164
        IF (TM2.LT.CALLIM(K,2,1).OR.TM2.GT.CALLIM(K,1,1)) GOTO 164
        TM=(TM1+TM2)/2.
      PRECAL=TM1-TM
      POSTCAL=TM2-TM
      DIFF=PRECAL-POSTCAL
      DIFF=AMOD(DIFF,100.0)
      NDIFF=0
C
C-----COMPUTE CALIBRATION
C
170     CONTINUE
      CAL=FUNCAL(TM,TCENT,RELHUM,PR,decade)
C
C-----ADVANCE THE PASS NUMBER
C
175     CONTINUE
        do 508 i=7,12
508     if(obsflag(i))go to 180
      PASS(JSAT)=PASS(JSAT)+1
      IFIRST=1
C
C-----FOR THIS OBSERVATION PUT BAD DATA COMMENTS TO REJECT FILE
C
180     CONTINUE
        if(obsflag(16)) CALL REJERS(OBSFLAG,16,16)
        do 501 i=7,12
501     if(obsflag(i)) BADHFLG=1
400     CONTINUE
        CALL REJERS(OBSFLAG,7,12)
      RETURN
C
300     CONTINUE
      decade=100.
      OBSFLAG(16)=.true.
      GO TO 125
      end


        SUBROUTINE RANCRD
C
c*****************************************************************
c
C-----process A RANGE DATA record. COMPUTE THE ONE-WAY RANGE IN
c-----centimers. also process hours, minutes, seconds, confidence code,
c-----and calibration.
C-----THEN, via stdfrm, write THE REDUCED OBSERVATION DATA from  LASCOM
c-----to THE DATA STREAM file for001.dat.
C
```

```
C               OBSFLAG(I)      MEANING
C                       1       SKY CODE
C                       2       RANGE (DATA OR VALUE)
C                       3       SECONDS
C                       4       ILLEGAL MINUTES
C                       5       ILLEGAL HOURS
C                       6       ILLEGAL CHECKSUM
C
c*****************************************************************
c
        COMMON/SEq/passeq,saonum(25),nasnum(25),nnassat,nnassta,
     1          TRNFLG,WNDFLG,OBSONE,BADHFLG,PASSFLG,NSEQ
        COMMON/LASER/SAT(25),NUMSAT,PASS(25),SUM(25)
       COMMON/LASCOM/CODE(4),DECADE,DT,C,Y,D,M,HR,MIN
     1  ,SEC,IDENT,ILLCODE,IFIRST,JSTA,NOGOOD,OBS,PR,RANGE,RELHUM,
     1     STAT,TEMP,PRESS,HUM,TCENT,JSAT,TM,CAL,OBSFLAG
     2  ,ACOEFF,BCOEFF,PRECAL,POSTCAL,ALOGSS,DIFF,NDIFF,PPCAL
     3    ,NCHKSM1,NCHKSM2
        common/lascoc/card,pcar
       COMMON/SOLVE/DATER(7,2),STORE(43),WORK(115),ARRAY(6),NRANGE
       COMMON/BLOCK/IBLOCK(600)
       COMMON /PNTS/ IPNTS,NUMPTS,NUMGPTS
         integer*2 passeq,OBSONE,BADHFLG
       INTEGER CHKSUM
         logical*1 obsflag(17),WNDFLG
       INTEGER DELAY,SAONUM,FLAG,GO,HR,OBS
       INTEGER PASS,SAT,SEC,SUM,SW1,SW2,TEMP,stat
       INTEGER CODE,Y,D
       double precision r,range
         LOGICAL*1 PASSFLG
       LOGICAL TMPCHK,PRCHK,CALCHK,RANCHK
       LOGICAL CHKCHR
         character*80 card
         character*30 pcar
       DATA KPASS/0/
C
C        SAVE FIRST OBSERVATION OF A PASS ONLY OVER MULTIPLE RERUNS
C
         IF (OBSONE.EQ.1) GOTO 630
         WRITE(11,629) CARD(1:11)
629      FORMAT('____',A11)
         OBSONE=1
630 IF (WNDFLG) GOTO 500
         do 641 k=1,17
641      obsflag(k)=.false.
   50    CONTINUE
C
       IF( KPASS.NE.PASSEQ ) IUPDATE=0
       KPASS=PASSEQ
C
C-----CHECK IF ILLEGAL HOURS CHARACTERS
C
       IF(CHKCHR(pcar,1,2)) GO TO 110
       hr=iconv(pcar,1,2)
```

```
C
C-----CHECK IF HOURS LEGAL
C
      IF(HR.LE.23.AND.HR.GE.0) GO TO 116
110     CONTINUE
        obsflag(5)=.true.
      GO TO 122
  116 CONTINUE
C
C     ADVANCE DATE BY ONE DAY IF MIDNIGHT IS CROSSED
C
      IF( HR.EQ.23 ) IUPDATE=1
      IF( IUPDATE.EQ.0 .OR. HR.EQ.23 ) GO TO 122
      IF( HR.NE.0 ) GOTO 122
      CALL MIDNIT(Y,M,D)
      IUPDATE=0
C
C
C-----CHECK IF MIN,SEC LEGAL
C
122     CONTINUE
      IF(CHKCHR(pcar,3,4)) GO TO 118
        min=iconv(pcar,3,4)
      IF(MIN.LE.59.AND.MIN.GE.0)  GO TO 120
118     CONTINUE
        obsflag(4)=.true.
120     IF(CHKCHR(pcar,5,12)) GO TO 125
      IF(pcar(5:5).LE.'5') go to 130
125     CONTINUE
        obsflag(3)=.true.
130     CONTINUE
    sec=iconv(pcar,5,12)
C
C-----GET UNCORRECTED 2-WAY RANGE
C
     IF(.NOT.CHKCHR(pcar,16,25)) GO TO 140
     obsflag(2)=.true.
140     CONTINUE
      CODE(1)=ICONV(pcar,15,15)
      IF( pcar(15:15).LE.'2' .AND. pcar(15:15).GE.'0' ) GOTO 150
      obsflag(1)=.true.
150     CONTINUE
      NCHKSM1=0
        NCHKSM2=0
      CHKSUM=0
      IF( pcar(13:14).EQ.'00') GOTO 160
c  delete next line when stations give correct checksum for first
c  OBSERVATION record IN A PASS.
      if (nseq.eq.3) goto 160
      if (pcar(13:13).eq.'0'.or.pcar(14:14).eq.'0') goto 160
      DO 152 I=1,25,2
  152 nchksm1=nchksm1 + ichar(pcar(i:i))-ichar('0')
      nchksm1=nchksm1 - ichar(pcar(13:13))+ichar('0')
      CHKSUM = MOD(NCHKSM1,10)*10
```

```
      DO 153 I=2,24,2
  153 nchksm2=nchksm2 + ichar(pcar(I:i))-ichar('0')
      nchksm2=nchksm2 - ichar(pcar(14:14))+ichar('0')
      CHKSUM = MOD(NCHKSM2,10) + chksum
  158 NCHKSM1=0
        NCHKSM2=0
        if(chksum.EQ.iconv(pcar,13,14))go to160
        OBSFLAG(6)=.TRUE.
160     CONTINUE
      r=dflotj(ICONV(pcar,16,20))*1.d05 + dflotj(iconv(pcar,21,25))
      r=(r/1.d01)+dble(CAL)
      r=r*1.d-09
      RANGE =2.997925d08*r
C
C-----CONVERT TO 1-WAY RANGE IN CM.
C
      RANGE=RANGE*5.d01
      IF(OBSFLAG(12).OR.OBSFLAG(2)) GO TO 165
      IF(RANGE.GE.9900000000.) GO TO 300
      IF(RANCHK(RANGE,JSAT,OBSFLAG))go to 165
300       CONTINUE
      obsflag(2)=.true.
  165 CONTINUE
        if(ifirst.eq.0)go to 170
        do 888 j=4,17
888     if(obsflag(j))go to 170
      SYSCAL= TM - FUNCAL(0.0,TCENT,RELHUM,PR,RS)
C
C____PREPARE A TRANSIT DATA SORT MAP RECORD
C
        CALL TRANSMAP(lndx)
        TRNFLG=1   !IF 0, WILL BYPASS TRANSORT IN BOLO1
C
      IFIRST=0
170     CONTINUE
        IPNTS=1
        do 891 j=1,6
891     if(obsflag(j))go to 600
        goto 180
600     CALL REJREC(CARD)
        CALL REJERS(OBSFLAG,1,6)
        GOTO 999
180     CONTINUE
        IF (BADHFLG.EQ.1) GOTO 600
C
C-----KEEP COUNT OF GOOD AND BAD DATA POINTS
C
        OBS=SAONUM(JSAT)
        NUMPTS=NUMPTS+1
        IF (CODE(1).NE.3) NUMGPTS=NUMGPTS+1
        CALL TRNPTS(LNDX)
C
c-----build the standard binary output form.
C
```

```
      CALL stdfrm
C
C------ADVANCE OBSERVATION NUMBER.
C
      K=SAONUM(JSAT)+1
      IF(K.GE.30000)K=20001
      SAONUM(JSAT)=K
C
  999 CONTINUE
      RETURN
C
500   CONTINUE
        CALL REJREC(CARD)
        WRITE(11,502)
502     FORMAT(6X,'REJECT; WETTZELL OBS, NO CORRECTIONS')
        GOTO 999
      END

        SUBROUTINE NASA
C
C...REDUCE NASA QUICK LOOK LASER DATA TO  SAO FORMAT
C
C               FOR002.DAT      INPUT OBSERVATIONS DATA
C               FOR001.DAT      OUTPUT REDUCED OBSERVATIONS DATA STREAM
C               FOR011.DAT      OUTPUT REJECTED OBSERVATIONS, HEADER
C                               RECORDS AND COMMENTS. IF CORRECTED, CAN
C                               RERUN FOR011.DAT AS FOR002.DAT
C...THE OBSFLGN ARRAY FLAGS OBSERVATION DATA ERRORS. AN OBSERVATION
C...IS REJECTED IF ANY DATA ERROR OCCURS.
C               OBSFLGN(I) = .FALSE. IF THERE ARE NO DATA ERRORS
C                            .TRUE.  IF THERE ARE ANY DATA ERRORS
C               OBSFLGN(I)      MEANING
C       1       STATUS---PRIME LASER
C                        2      STATION
C                        3      SATELLITE
C                        4      YEAR
C                        5      DAYS PER YEAR
C                        6      DAY OF THE MONTH
C                        7      SECONDS PER DAY
C                        8      FRACTION OF A SECOND
C                        9      TWO WAY TIME RANGE
C                       10      RANGE
C                   11          BAD RECORD
C_____________________________________________________________________
C
        EXTERNAL ICONV,CHKCHR,RANCHK
        DOUBLE PRECISION FMJD,SECOND,FSEC,RANGE,RNGTIME,PRANGE
        CHARACTER*80 RCRD,PCRD*30
        CHARACTER*4 CSTA,NASSTA(25),SAOSTA(25),CND1*1,CYR*2
        CHARACTER*4 CSAT,NASSAT(25),SAOSAT(25)*7,CDAPYR*3,CSECPDA*5
        CHARACTER*6 CFSEC
        LOGICAL*1 LEAPYR,CHKCHR,FLAG(17),OBSFLGN(11),WNDFLG,PASSFLG
        LOGICAL *1 RANCHK
        REAL*4 NOISE
```

```
      INTEGER*2 JNASTA,STA,TDPY,YR,MONTH,DAY,HOUR,MIN,IDPMT(12,2)
      INTEGER*2 PASSEQ,TYPE,COL57,COL58,COL62,CNDX,DUM(6),OBSONE
      INTEGER*2 BADHFLG
      INTEGER*4 SAONUM,OBSNO,PRES
      COMMON/SEQ/PASSEQ,SAONUM(25),NASNUM(25),NNASSAT,NNASSTA,
     1         TRNFLG,WNDFLG,OBSONE,BADHFLG,PASSFLG,NSEQ
      COMMON/NASAC/NASSAT,SAOSAT,NASSTA,SAOSTA
      COMMON/LIMIT/ISTA(25),PRLIM(25,2),TMPLIM(25,2),CALLIM(25,2,2),
     1           RANLIM(25,2),EXPDATE(25)
      COMMON/LASCOC/RCRD,PCRD
      DATA IDPMT/31,59,90,120,151,181,212,243,273,304,334,365,
     1         31,60,91,121,152,182,213,244,274,305,335,366/
      DATA REFCOR,TIMPRE,CENMAS,HUMID,ATEMP,ELEVANG,
     1    AZIMUTH,NOISE,BIAS,PRES/9*0.0,0/
      DATA PRANGE,CNDX/0.D00,0/
C_____________________________________________________________
C...BEGIN A NASA PASS
      KOUNT=1
      OBSONE=0
      IRRFLG=0
C_____________________________________________________________
C...OUTPUT HEADER RECORD
80    IF (NDETFLG.EQ.0) GOTO 83
      WRITE(11,81)
81    FORMAT('LASERQL')
      NDETFLG=0
      GOTO 85
83    CALL REJREC(RCRD)
C_____________________________________________________________
C...GET THE NEXT OBSERVATION RECORD
100   READ(2,101,END=38) RCRD
101   FORMAT(A)
85    J=INDEX(RCRD,'....')   ! DELETE A COMMENT
      IF (J.NE.0) GOTO 100
      J=INDEX(RCRD,'____')   ! SAVE SPECIAL COMMENT OVER MULTIPLE RERUNS
      IF (J.EQ.0) GOTO 103
      IRRFLG=IRRFLG+1
      WRITE(11,101) RCRD
      OBSONE=1
      GOTO 100
103   DO 102 J=1,11              ! ERROR FLAGS
102   OBSFLGN(J)=.FALSE.
C...IS THIS RECORD AN OBSERVATION?
      J=LASTC(RCRD,80)
      IF (J.EQ.0) GOTO 100       ! DELETE EMPTY RECORDS
      J=INDEX(RCRD,'END') ! END OF PASS?
      IF (J) 104,104,38
104   J=INDEX(RCRD,'1BB') ! FIND BEGINNING OF DATA ON OBS RECORD
      IF (J-1) 107,106,105
105   RCRD(1:81-J)=RCRD(J:80)
106     IF(RCRD(1:3).EQ.'1BB'.AND.RCRD(70:72).EQ.'4FF') THEN
      IF (OBSONE.EQ.1) GOTO 122
      IF (IRRFLG.GT.0) GOTO 122
      WRITE(11,121) RCRD(1:14)
```

```
121     FORMAT('____',A14)
        OBSONE=1
        GOTO 122
        ELSE
107     IF (OBSONE.EQ.1) GOTO 120
        WRITE(11,121) RCRD(1:14)
        OBSONE=1
120     OBSFLGN(11)=.TRUE.
        GOTO 2000       ! REJECT RECORD AND CONTINUE
        ENDIF
C_______________________________________________________________
C...PROCESS THE RECORD; STATUS FIELD
122     CND1=RCRD(15:15)
        IF(CND1.EQ.'0'.OR.CND1.EQ.'4') GOTO 200
        OBSFLGN(1)=.TRUE.
        GOTO 2000
C_______________________________________________________________
C...PROCESS THE STATION
200     CONTINUE
        if (RCRD(10:10).eq.' ') RCRD(10:10)='0'
        IF (CHKCHR(RCRD,10,13))  GOTO 210
        CSTA=RCRD(10:13)
        DO 220 J=1,NNASSTA
        IF (CSTA.EQ.NASSTA(J)) GOTO 230
220     CONTINUE
210     CONTINUE
        OBSFLGN(2)=.TRUE.
        GOTO 2000
230     CSTA=SAOSTA(J)
        STA=ICONV(CSTA,1,4)
C_______________________________________________________________
C...PROCESS THE SATELLITE
300     CONTINUE
        IF (RCRD(4:4).EQ.' ') RCRD(4:4)='0'
        IF (CHKCHR(RCRD,4,7)) GOTO 310
        CSAT=RCRD(4:7)
        DO 320 L=1,NNASSAT
        IF (CSAT.EQ.NASSAT(L)) GOTO 330
320     CONTINUE
310     CONTINUE
        OBSFLGN(3)=.TRUE.
        GOTO 2000
330     CONTINUE
        LNASAT=L
        IDENT=ICONV(SAOSAT(L),1,7)
C_______________________________________________________________
C...PROCESS THE YEAR
400     CONTINUE
        IF (CHKCHR(RCRD,24,25)) GOTO 410
        CYR=RCRD(24:25)
        YR=ICONV(RCRD,24,25)
        IF (YR.GT.76) GOTO 430
410     CONTINUE
        OBSFLGN(4)=.TRUE.
```

```
        GOTO 2000
430     CONTINUE
        TDPY=365
        LEAPYR=.FALSE.
        ITM=YR-76
        IF ((ITM/4)*4.EQ.ITM) TDPY=366
        IF (TDPY.EQ.366) LEAPYR=.TRUE.
C_______________________________________________________________________
C...PROCESS DAYS PER YEAR INTO MONTH AND DAY OF MONTH
500     CONTINUE
        IF (CHKCHR(RCRD,26,28)) GOTO 510
        CDAPYR=RCRD(26:28)
        IDAPYR=ICONV(RCRD,26,28)
        IF (IDAPYR.GE.1.AND.IDAPYR.LE.TDPY)GOTO 530
510 CONTINUE
        OBSFLGN(5)=.TRUE.
        GOTO 2000
C...IS CURRENT YEAR A LEAP YEAR ?
530     CONTINUE
        J=1
        IF (LEAPYR) J=2
        DO 535 I=1,12
        IF (IDAPYR.LE.IDPMT(I,J)) GOTO 540
535     CONTINUE
        OBSFLGN(6)=.TRUE.
        GOTO 2000
540     DAY=IDAPYR
        MONTH=I
        IF (I.EQ.1) GOTO 600
        DAY=IDAPYR-IDPMT(I-1,J)
C_______________________________________________________________________
C...PROCESS SECONDS PER DAY INTO HOURS, MINUTES, AND SECONDS
600     CONTINUE
        IF (CHKCHR(RCRD,29,33)) GOTO 610
        CSECPDA=RCRD(29:33)
        ISECPDA=ICONV(RCRD,29,33)
        IF(ISECPDA.GE.0.AND.ISECPDA.LE.86400) GOTO 630
610     CONTINUE
        OBSFLGN(7)=.TRUE.
        GOTO 2000
630     INTM=ISECPDA
        HOUR=INTM/3600
        IHR=HOUR
        INTM=INTM-3600*IHR
        MIN=INTM/60
        MINT=MIN
        ISEC=INTM-60*MINT
C_______________________________________________________________________
700     CONTINUE
        CFSEC=RCRD(34:39)
        IF (.NOT.CHKCHR(RCRD,34,39)) GOTO 710
        OBSFLGN(8)=.TRUE.
        GOTO 2000
710     FSEC=DFLOTJ(ICONV(RCRD,34,39))/1.D06
```

```
        SECOND=DFLOTJ(ISEC)+FSEC
C_____________________________________________________________________
C...COMPUTE MEAN JULIAN DAY, AND FRACTION THEREOF
800     CONTINUE
        INTM=(YR-73)/4
        MJD=41682+INTM*366+(YR-73-INTM)*365+IDAPYR
        FMJD=(DFLOTJ(ISECPDA)+FSEC)/86400.D00
C_____________________________________________________________________
C...PROCESS RANGE (2-WAY TIME (NANOSEC) INTO 1-WAY DISTANCE (METERS)
900     CONTINUE
        IF (.NOT.CHKCHR(RCRD,55,66)) GOTO 910
        OBSFLGN(9)=.TRUE.
        GOTO 2000
910     RNGTIME=DFLOTJ(ICONV(RCRD,55,60))
        RNGTIME=RNGTIME*1.D06+DFLOTJ(ICONV(RCRD,61,66))
        RANGE=.299792458D00*RNGTIME/2.D00    ! METERS
        IF (RANCHK(RANGE,LNASAT,FLAG)) GOTO 950
        OBSFLGN(10)=.TRUE.
        GOTO 2000
950     RANGE=RANGE/1.D02    ! METERS
C_____________________________________________________________________
C...PREPARE INTERNAL DATA STREAM
1000    CONTINUE
        OBSNO=NASNUM(LNASAT)
        COL57=4
        COL58=8
        CALSTAB=0
        COL62=0
        TYPE=8
        RANPRE=2
        TIMPRE=0.0002
        CNDX=1
C_____________________________________________________________________
C...OUTPUT THE INTERNAL DATA STREAM, UNFORMATTED
        WRITE(1,ERR=2000) IDENT,OBSNO,STA,MJD,FMJD,TYPE,RANGE,
     1    REFCOR,TIMPRE,RANPRE,COL57,COL58,CALSTAB,COL62,CENMAS,
     2    PRES,HUMID,ATEMP,ELEVANG,PRANGE,AZIMUTH,CNDX,NOISE,
     3    BIAS,YR,MONTH,DAY,HOUR,MIN,SECOND,PASSEQ,DUM
C_____________________________________________________________________
C...INITIALIZE FOR NEXT OBSERVATION
        KOUNT=KOUNT+1
        K=NASNUM(LNASAT)+1
        IF (K.GE.40000) K=30001
        IF (K.LE.30000) K=30001
        NASNUM(LNASAT)=K
       GOTO 100
C_____________________________________________________________________
C...REJECT RECORD AND OUTPUT DATA ERROR COMMENT
2000    CONTINUE
        CALL REJREC(RCRD)
        CALL REJERN(OBSFLGN)
        KOUNT=KOUNT+1
        GOTO 100
C_____________________________________________________________________
```

```
C...PASS COMPLETED, RETURN TO OBSCARD
38      IF (IRRFLG.GT.0) GOTO 40
        WRITE(11,37) KOUNT-1
37      FORMAT('___NUMBER OF OBSERVATIONS IN PASS',I3)
40      CALL REJREC(RCRD)
        K=PASSEQ+1
        IF (K.GE.30000) K=1
        PASSEQ=K
        RETURN
        END

        subroutine stdfrm
c
c*****************************************************************
c
c       collect the necessary values, change a few DATA types,
c       and write the standard SAO OBSERVATION RECORD, unformated,
C       TO THE data stream IN for001.dat file.
c
c*****************************************************************
c
        external mjdf
        LOGICAL*1 PASSFLG
        integer*4 ident,obsno,mjd,stat,obs,code,y,d,sec,pass,
     1  sat,temp,chksum,SAONUM,hr,
     2  tenmm,ihum,ipres,col53,col5455,
     3  ccal
        integer*2 type,col57,col58,col62,cndx,year,month,day,sta,
     1 hour,minute,passeq,paseqn,OBSONE,BADHFLG
        integer*2 dy(52)
        integer*2 i2zero
        integer*4 i4zero
        real*4 r4zero
        double precision r8zero
        real refcor,timpre,ranpre,calstab,cenmas,humid,tcent,
     2  elevang,azimuth,noise,bias
        double precision range,second,mjdfr,prange,orange
        logical*1 obsflag(17),WNDFLG
        common/seq/passeq,saonum(25),nasnum(25),nnassat,nnassta,
     1          TRNFLG,WNDFLG,OBSONE,BADHFLG,PASSFLG,NSEQ
        COMMON /LASER/ SAT(25),NUMSAT,PASS(25),SUM(25)
        COMMON/LASCOM/CODE(4),DECADE,DT,C,Y,D,M,HR,MIN
     1  ,SEC,IDENT,ILLCODE,IFIRST,JSTA,NOGOOD,OBS,PR,RANGE,RELHUM,
     1     STAT,TEMP,PRESS,HUM,TCENT,JSAT,TM,CAL,OBSFLAG
     2  ,ACOEFF,BCOEFF,PRECAL,POSTCAL,ALOGSS,DIFF,NDIFF,PPCAL
     3    ,NCHKSM1,NCHKSM2
        equivalence (orange,dy(1))
        equivalence (refcor,dy(5))
        equivalence (timpre,dy(7))
        equivalence (ranpre,dy(9))
        equivalence (col57,dy(11))
        equivalence (col58,dy(12))
        equivalence (calstab,dy(13))
        equivalence (col62,dy(15))
```

```
      equivalence (cenmas,dy(16))
      equivalence (pres,dy(18))
      equivalence (humid,dy(20))
      equivalence (atemp,dy(22))
      equivalence (elevang,dy(24))
      equivalence (prange,dy(26))
      equivalence(azimuth,dy(30))
      equivalence (cndx,dy(32))
      equivalence (noise,dy(33))
      equivalence (bias,dy(35))
      equivalence (year,dy(37))
      equivalence (month,dy(38))
      equivalence (day,dy(39))
      equivalence (hour,dy(40))
      equivalence (minute,dy(41))
      equivalence (second,dy(42))
      equivalence (paseqn,dy(46))
      data i2zero,i4zero,r4zero,r8zero/4*0/
      data refcor,timpre,ranpre,ctrmas/4*0/
      data elevang,azimuth,prange/3*0/
      data noise,bias/2*0.0/
C
      PASSFLG=.TRUE.
c     if (code(1).ne.3) obsno=-obsno
      sta=stat
      col58=8
      col57=0
      calstab=0
      col62=1
      ndx=mjdf(y,m,d,ismd)
      mjd=ndx*ismd
      atemp=tcent
      pres=pr
      humid=hum
      orange=range*1.d-2
      year=y
      month=m
      day=d
      hour=hr
      minute=min
      second=dflotj(sec)*1.d-6
      mjdfr=((second/60.+min)/60.+hr)/24.
      type=8
      cndx=0
      timpre=.0002
      ranpre=2.0
      paseqn=PASSEQ
      write(1,err=100)ident,obs,sta,mjd,mjdfr,type,dy
      goto 999
C
C     I/O ERROR IN WRITE TO DATA STREAM RECORD. ANALYSIS PROGRAM WILL
C     DETECT SUCH ERRORS AND DELETE THE OBSERVATION RECORD.
C
100   CALL ERRSNS(IFNUM,IA,IB,IC,ID)
```

```
      WRITE(11,101) IFNUM,IA,IB
101   FORMAT('....WRITE ERROR IN DATA STREAM OBSERVATION  RECORD',
      1       ' TO FOR001.DAT'/'.......FORTRAN I/O ERROR # ',I2/
      2       '.......RMS I/O ERROR COMPLETION STATUS, (HEX) ',Z8/
      3       '.......RMS I/O ERROR STATUS VALUE, (HEX) ',Z8)
999   return
      end

            SUBROUTINE TCODE(NUMPASS)
C
C     ***************************************************************
C
C     PREPARES THE LASERA RECORD FROM THE PASS HEADER DATA STORED IN
C     COMMON LASHED AND FROM THE POINT OBS DATA STORED IN COMMON LASOBS
C
C     OUTPUT
C          BUFF = ARRAY CONTAINING THE LASERA VARIABLES AS EQUIVALENCED
C
C
C     ***************************************************************
C
C
      INTEGER*2  BUFF(64)
        INTEGER*2 STATION,STATIONX, SKYC, YEAR, DAY, MONTH,HOUR,MINUTE,STATUS
     1  ,TYPE,EPOCH_SYS,OBS_UNIT,CALIB_INDEX,SPEED_LIGHT
     1  ,YR,MO,DY,HR,MIN,L
        INTEGER*4 NSAT,NEXOBS,MJD,FILL(3),NSATX,NEXOBSX
        REAL*4 RFRCORR,TIME_PREC,RANGE_PREC,CALIB_STABIL,CEN_MASS
     1  ,PRESSURE,HUMIDITY,TEMP,ELEV_ANGLE,AZIMUTH,RANGE_NOISE,RANGE_BIAS
     2  ,HUM,PRESS,TMP,RFRCORRX,WT,COMCORR,ELEV,AZIM
        REAL*8 FRAC_DAY,RANGE,PREDIC_RANGE,SECOND,RANGEX,PRERANG,SEC
        LOGICAL RNGCM
        EQUIVALENCE (NSAT,BUFF(1))
        EQUIVALENCE (NEXOBS,BUFF(3))
        EQUIVALENCE (STATION,BUFF(5))
        EQUIVALENCE (MJD,BUFF(6))
        EQUIVALENCE (FRAC_DAY,BUFF(8))
        EQUIVALENCE (TYPE,BUFF(12))
        EQUIVALENCE (RANGE,BUFF(13))
        EQUIVALENCE (RFRCORR,BUFF(17))
        EQUIVALENCE (TIME_PREC,BUFF(19))
        EQUIVALENCE (RANGE_PREC,BUFF(21))
        EQUIVALENCE (EPOCH_SYS,BUFF(23))
        EQUIVALENCE (OBS_UNIT,BUFF(24))
        EQUIVALENCE (CALIB_STABIL,BUFF(25))
        EQUIVALENCE (CALIB_INDEX,BUFF(27))
        EQUIVALENCE (CEN_MASS,BUFF(28))
        EQUIVALENCE (PRESSURE,BUFF(30))
        EQUIVALENCE (HUMIDITY,BUFF(32))
        EQUIVALENCE (TEMP,BUFF(34))
        EQUIVALENCE (ELEV_ANGLE,BUFF(36))
        EQUIVALENCE (PREDIC_RANGE,BUFF(38))
        EQUIVALENCE (AZIMUTH,BUFF(42))
        EQUIVALENCE (SPEED_LIGHT,BUFF(44))
```

```
      EQUIVALENCE (RANGE_NOISE,BUFF(45))
      EQUIVALENCE (RANGE_BIAS,BUFF(47))
      EQUIVALENCE (YEAR,BUFF(49))
      EQUIVALENCE (MONTH,BUFF(50))
      EQUIVALENCE (DAY,BUFF(51))
      EQUIVALENCE (HOUR,BUFF(52))
      EQUIVALENCE (MINUTE,BUFF(53))
      EQUIVALENCE (SECOND,BUFF(54))
      EQUIVALENCE (STATUS,BUFF(58))
      EQUIVALENCE (FILL(1),BUFF(59))
     COMMON /LASHED/ STATIONX,DIST,YR,MO,DY,NSATX,SKYC,HUM,TMP,
     $                PRESS,PRECAL,POSTCAL,PPCAL,DIFF,NDIFF,ACOEFF,
     $ BCOEFF,REFAREA
     COMMON /LASOBS/ NEXOBSX,HR,MIN,SEC,AZIM,ELEV,PRERANG,RANGEX,WT,L,
     $                RFRCORRX,COMCORR
      COMMON /UNITS/ C,RNGCM
      FILL(1)=0
      FILL(2)=0
      FILL(3)=0
C
C
C
C
C  SET UP EQUIVALENCED VARIABLES--LASHED
      STATION=STATIONX
      YEAR=YR
      MONTH=MO
      DAY=DY
      HUMIDITY=HUM
      TEMP=TMP
      PRESSURE=PRESS
      NSAT=NSATX
C
C
C
C
C  SET UP EQUIVALENCED VARIABLES--LASOBS
      NEXOBS=NEXOBSX
      HOUR=HR
      MINUTE=MIN
      SECOND=SEC
      AZIMUTH=AZIM
      ELEV_ANGLE=ELEV
      PREDIC_RANGE=PRERANG
      RANGE=RANGEX/10.00
      RANGE_PREC=WT
      CALIB_INDEX=L
      RFRCORR=RFRCORRX
      CEN_MASS=COMCORR
      RANGE_NOISE=0.0
      RANGE_BIAS=0.0
      STATUS=NUMPASS
      FRAC_DAY=(DFLOAT(HOUR)+(DFLOAT(MINUTE)+SECOND/60.D00)/60.D00)/24.D00
      II=(MJDF(YEAR,MONTH,DAY,MJD))
```

```
C
C
C
      TIME_PREC=.0001
C
        TYPE=8
        EPOCH_SYS=4
      OBS_UNIT = 4
      IF( RNGCM ) OBS_UNIT = 7
      CALIB_STABIL=ABS(DIFF)
C
C
C
C
C
C
C
        WRITE(7) BUFF
C
C
      RETURN
      END

        PROGRAM NASABIN
C
C       THIS PROGRAM INVOKES SUBROUTINE GETREC WHICH READS SUCCESSIVE
C       RANGE RECORDS FROM TAPE IN NASA BINARY FORMAT; 9-TO-9 HAS BEEN
C       USED TO PRODUCE A VAX READABLE TAPE FROM THE ORIGINAL
C       IBM TAPE.  THE VARIOUS
C       QUANTITIES ARE UNSCRAMBLED AND CONVERTED FROM IBM TO DEC VAX
C       FORMAT, THEN WRITTEN TO TAPE IN SAO FORMAT WITH SUBROUTINE PUTREC.
C
C       INPUT: UNIT FOR001   OUTPUT: FOR002
C
        BYTE RECORD(68),B4(4),B5(4),B6(4)
        INTEGER*2 MDRTK, MDRAP, MDRRH, TIMESYS
        INTEGER*2 IVSHORT
        REAL*8 GMT, OBSVAL, AIBM, C_OLD
        EQUIVALENCE (IB4,B4(4)), (IB5,B5(4)),(IB6,B6(4))
        DATA C_OLD/2.997925D8/  !METERS/SEC. NASA USES OLD S.O.L.
C
        NREC = 0
        NBAD = 0
        DO 11 I=1,4
        B4(I)='00'X
        B5(I)='00'X
11      B6(I)='00'X
1       GO TO (100, 200, 300, 400) IGETREC(RECORD)
C
C       SUCCESSFUL TAPE READ:
100     NREC = NREC + 1
        IDSAT =  IVINT(RECORD(1))
        TIMESYS = IVSHORT(RECORD(7))
        NUMSTAT = IVINT(RECORD(9))
```

```
      MJD = IVINT(RECORD(17))
      GMT = AIBM(RECORD(21))
      OBSVAL = AIBM(RECORD(29))
C
C     NASA DATA GIVES EPOCH AS TIME AT SATELLITE, SAO AS
C     TRANSMISSION TIME.  CONVERT:
      GMT = GMT - (OBSVAL/C_OLD )/86400.D0
      IF (GMT.LT.0.D0) THEN
              GMT = GMT + 1.D0
              MJD = MJD - 1
      ENDIF
      TROPREF = SIBM(RECORD(49))
C
C     NASA SIGN CONVENTION ON CENTER OF MASS CORRECTION APPEARS
C     TO BE OPPOSITE SAO, SO WE CHANGE THE SIGN.
      CMASS = -SIBM(RECORD(65))
C
C     NOW WE UNSCRAMBLE THE METEOROLOGICAL DATA REPORT WHICH IS PACKED
C     IN BYTES 53-56. NOTE THAT DIVIDING (MULTIPLYING) BY 2**N SHIFTS
C     A BYTE RIGHT (LEFT) BY N BITS.
C     BITS 1-7 OF BYTE53 GIVE THE RELATIVE HUMIDITY; WE ARE TOLD
C     BIT 0 IS ALWAYS 0, AND WE SO ASSUME.
C
      MDRRH = RECORD(53)
C
C     MDRTK CONSISTS OF BYTE 54 PLUS FIRST HALF OF BYTE 55;
C     MDRAP IS SECOND HALF OF BYTE 55 PLUS BYTE 56.
C
C     PUT BYTES 54,55,56 OF RECORD IN BYTE ARRAYS EQUIVALENCED
C     TO IB4,IB5,IB6:
C
      B4(4)=RECORD(54)
      B5(4)=RECORD(55)
      B6(4)=RECORD(56)
C
C     IMAGINE THE 4 HALF-BYTES IN BYTES 54 AND 55 LABELLED A,B,
C     C,D IN ASCENDING ORDER. WE NOW ISOLATE THESE:
C
      IC = IB5/16
      IA = IB4/16
      IB = IB4 - 16*IA
      ID = IB5 - 16*IC
C
C     NOW CONSTRUCT THE PRESSURE AND TEMPERATURE:
C
      MDRTK = IC + 16*(IB+16*IA)
      MDRAP = IB6 + 256*ID
      CALL PUTREC(IDSAT, TIMESYS, NUMSTAT, MJD, GMT, OBSVAL,
     1   TROPREF, MDRRH, MDRTK, MDRAP, CMASS)
C
C     RECYCLE TO READ A NEW LOGICAL RECORD:
      GO TO 1
C
C     END OF TAPE SENSED; NORMAL TERMINATION:
```

```
C
 200   WRITE(6,201) NREC
 201   FORMAT('0   END OF TAPE SENSED. NORMAL TERMINATION AFTER '
     1 , 'READING', I6, ' LOGICAL RECORDS.')
       STOP
C
C      UNRECOGNIZABLE RECORD READ:
C
300    WRITE(6,301) NREC, (RECORD(I), I = 1, 68)
301    FORMAT('0  RECORD READ IS NEITHER LASER NOR ANGLE DATA:'/
     1  10X, 'NREC=',I7/ 10X, 9(1X,4Z2)/ 10X, 8(1X,4Z2))
       NBAD = NBAD+1
       IF (NBAD.LT.20) GOTO 1
       STOP
C
C      ABNORMAL STATUS RETURNED FROM TAPE READ. MESSAGE WRITTEN BY GETREC.
400    STOP
       END !NASABIN

       INTEGER FUNCTION IGETREC(RECORD)
C
C      IGETREC READS A TAPE CONTAINING DATA IN IBM/NASA BINARY FORMAT
C      (TRANSLATED TO VAX READABLE FORM BY 9-TO-9).  SUCCESSIVE CALLS
C      RETURN SUCCESSIVE 68-BYTE LOGICAL RECORDS OF LASER RANGING
C      DATA IN BYTE ARRAY "RECORD"; LOGICAL RECORDS OF ANGLE (POINTING)
C      DATA ARE IGNORED.  THE VALUE OF IGETREC ON RETURN INDICATES:
C        1     SUCCESSFUL RETURN
C        2     END OF TAPE
C        3     UNRECOGNIZABLE RECORD (NEITHER RANGING NOR ANGLE).
C              THE RECORD IS RETURNED IN "RECORD."
C        4     ABNORMAL STATUS ON RETURN FROM READ. MESSAGE WRITTEN.
C
       BYTE RECORD(68), BBUF(8196)
       LOGICAL EOF, FIRST
       INTEGER BUFCOUNT
       DATA FIRST/.TRUE./, LUN/1/
C
C      ON FIRST SUBROUTINE CALL WE MUST READ A RECORD
C
       IF (FIRST) THEN
               FIRST= .FALSE.
               NEOF = 0
100            CALL BREAD(LUN, ISTAT, NBYTES, BBUF, EOF) !READ A RECORD
               IF (EOF) THEN  !WE HAVE REACHED END OF TAPE
                       IGETREC = 2
                       RETURN
               ENDIF
               IF (ISTAT.NE.'1'X .AND. ISTAT.NE.'870'X .AND.
       1       ISTAT.NE.'878'X) THEN
                       WRITE(6,150) ISTAT     ! ABNORMAL ISTAT FROAD
150                    FORMAT('0*** ABNORMAL STATUS RETURNED FROM',
       1       ' TAPE READ ATTEMPT:', Z10, ' (HEX)'/)
                       IGETREC = 4
                       RETURN
```

```
                ENDIF
                IF (NBYTES.GE.76) THEN   !IBM FILE MARK LT 76 BYTES
                        BUFCOUNT = 5 !COUNTER TO FST BYTE OF LR1
                ELSE
                        NEOF = NEOF + 1  !WE HAVE READ FILE MARK
                        IF (NEOF.LT.2) THEN
                                GO TO 100 !ONE FM SO TRY AGAIN
                        ELSE
                                IGETREC = 2  !2 FILE MARKS = EOT
                                RETURN
                        ENDIF
        ENDIF
        ENDIF
C
C       TEST TO SEE IF WE HAVE USED UP LAST RECORD READ; IF SO, GET ANOTHER
C
200     IF (BUFCOUNT.GT.NBYTES) THEN
                NEOF = 0
300             CALL BREAD(LUN, ISTAT, NBYTES, BBUF, EOF) !READ A RECORD
                IF (EOF) THEN  !WE HAVE REACHED END OF TAPE
                        IGETREC = 2
                        RETURN
                ENDIF
                IF (ISTAT.NE.'1'X .AND. ISTAT.NE.'870'X .AND.
        1       ISTAT.NE.'878'X) THEN
                        WRITE(6,150) ISTAT     ! ABNORMAL ISTAT FROAD
                        IGETREC = 4
                        RETURN
                ENDIF
                IF (NBYTES.GE.76) THEN   !IBM FILE MARK LT 76 BYTES
                        BUFCOUNT = 5 !COUNTER TO FST BYTE OF LR1
                ELSE
                        NEOF = NEOF + 1  !WE HAVE READ FILE MARK
                        IF (NEOF.LT.2) THEN
                                GO TO 300 !ONE FM SO TRY AGAIN
                        ELSE
                       IGETREC = 2  !2 FILE MARKS = EOT
                           RETURN
                        ENDIF
                ENDIF
        ENDIF
C
C       WE NOW TEST TO SEE IF THE LOGICAL RECORD POINTED TO BY BUFCOUNT
C       IS A LASER RECORD, ANGLE RECORD OR UNRECOGNIZABLE
C
        IF (BBUF(BUFCOUNT+9).EQ.'46'X) THEN  !TEXT FOR ANGLE
                BUFCOUNT = BUFCOUNT + 72       !IF ANGLE, LOOK AT NEXT REC.
                GO TO 200
        ENDIF
C
C       IF WE HAVE A LASER RECORD,SET IGETREC = 1, OTHERWISE = 3;
C       IN EITHER CASE, WE WILL FILL THE OUTPUT ARRAY 'RECORD'
C
        IF (BBUF(BUFCOUNT+9).EQ.'14'X) THEN
```

```
              IGETREC = 1
        ELSE
              IGETREC = 3
        ENDIF
        DO 400 I = 1, 68    !FILL OUTPUT ARRAY; IGNORE FIRST 4 BYTES
400     RECORD(I) = BBUF(BUFCOUNT+3+I)
        BUFCOUNT = BUFCOUNT +72   !INSURE WE ARE AT RIGHT PLACE ON REENTRY
        RETURN
        END

        SUBROUTINE PUTREC(IDSAT, TIMESYS, NUMSTAT, MJD, GMT, OBSVAL,
     1 TROPREF, MDRRH, MDRTK, MDRAP, CMASS)
C
C       PUTREC ACCEPTS THE VARIOUS QUANTITIES ASSOCIATED WITH A
C       NASA LASER RANGING RECORD, CONVERTS THEM TO FORMATS
C       CONSISTENT WITH SAO BINARY FORMAT WHERE NECESSARY,
C       COMPUTES ASSOCIATED QUANTITIES AND PACKS THE RESULTS INTO
C       BYTE ARRAY "BY," WHICH IS THEN WRITTEN (UNFORMATTEDLY)
C       TO UNIT FOR002.
C
        INTEGER*2 MDRTK, MDRAP, MDRRH, TIMESYS
        REAL*8 GMT, OBSVAL
        BYTE BY(128)
        INTEGER*2 STATNO,YY, MM, DD, HH, MIN, CALIND, SOL
        INTEGER*2 I8, I5, I2
        REAL*8 GMTP, RANGE, PRERNG, SEC
        LOGICAL FIRST
        DATA FIRST/.TRUE./, LUOUT/2/
        EQUIVALENCE (IDSATP,BY(1)), (NSEQ, BY(5)), (STATNO,BY(9)),
       1 (MJDP,BY(11)), (GMTP,BY(15)), (I8,BY(23)), (RANGE,BY(25)),
       2 (TROPREFP,BY(33)), (TIMEPREC,BY(37)), (RANGPREC,BY(41)),
       3 (I5,BY(45)), (I2,BY(47)), (CALSTAB,BY(49)),
       4 (CALIND,BY(53)), (CMASSP,BY(55)), (ATPRES,BY(59)),
       5 (RELHUM,BY(63)), (ATTEMP,BY(67)), (PREEL,BY(71)),
       6 (PRERNG,BY(75)), (PREAZ,BY(83)), (SOL,BY(87)),
       7 (RNGNOIS,BY(89)), (RNGBIAS,BY(93)), (YY,BY(97)),
       8 (MM,BY(99)), (DD,BY(101)), (HH,BY(103)),
       9 (MIN,BY(105)), (SEC,BY(107))
C
C       ON FIRST ENTRY WE SET THE BYTES WHICH WILL NOT CHANGE.
C       ALSO, SET INITIAL SEQUENCE NUMBER.
C
        IF(FIRST) THEN
        FIRST = .FALSE.
        NSEQ = 0
        I8 = 8          !LASER OBSERVATION
        TIMEPREC = 1.E-6        !NASA FIGURE
        RANGPREC = 0.1  !NASA
        I5 = 5          !INDEX
        2 = 2           !INDEX
        CALSTAB = 0.0   !NO CALIBRATION INFORMATION
        CALIND = 0      !"
        PREEL = 0.0     !NO PREDICTIONS
        PRERNG = 0.D0   !"
```

```
      PREAZ = 0.0     !"
      SOL =00 !OLD SPEED OF LIGHT WAS USED
      RNGNOIS = .070710677    !SET TO GIVE NASA PRECISION OF
      RNGBIAS = .070710677    !0.1/SQRT(2)
      DO 20 I = 115,128       !PASS SEQUENCE NUMBER AND
20    BY(I) = '00'X           !SPARE BYTES SET TO 0
      ENDIF
C
C     NOW WE SET THE VARIABLE QUANTITIES
C
      IDSATP = IDSAT
      NSEQ = NSEQ + 1 !ARBITRARY OBSERVATION NUMBER
      IF (NSEQ.GT.69999) NSEQ = 1     !NSEQ TOO HIGH
      STATNO = NUMSTAT
      MJDP = MJD
      GMTP = GMT
      RANGE = OBSVAL
      TROPREFP= TROPREF       !TROPOSPHER REFRACTION CORR
      CMASSP = CMASS          !CENTER OF MASS CORR
      ATPRES = FLOAT(MDRAP)
      RELHUM = FLOAT(MDRRH)/100.   !PERCENT TO FRACTION
      ATTEMP = FLOAT(MDRTK)-273.2  !KELVIN TO CELSIUS
      CALL INVSAO(MJD, IY,IM,ID)   !FULL IN MONTH, DAY, YEAR
      YY = IY
      MM = IM
      DD = ID
      SGMT = 24*GMT           !HOURS, MINUTES, SECONDS.  CODE LOOKS
      HH = IIFIX(SGMT)        !ODD BECAUSE FIX ONLY TAKES REAL*4 ARGS.
      SGMT = 60*(24*GMT-HH)
      MIN = IIFIX(SGMT)0034   SEC = 60*(60*(24*GMT-HH)-MIN)
C
C     FINISHED!  WRITE TO TAPE AND RETURN
C
      WRITE(LUOUT) BY
      RETURN
      END

      REAL*8 FUNCTION AIBM(BY)

      IMPLICIT REAL*8 (A-H,O-Z)
      BYTE BY(8)
      INTEGER*4 IT(8)



      AIBM=0.D0
      DO 101 I=1,8
      IT(I)=BY(I)
      IF(IT(I).LT.0)IT(I)=IT(I)+256
  101 CONTINUE
      IF(IT(1).GT.'7F'X)THEN
              SIGN=-1.D0
              IT(1)=IT(1)-'80'X
      ELSE
```

```
              SIGN=+1.D0
        ENDIF
        DO 102 I=2,8
  102   AIBM=256.D0*AIBM+DFLOAT(IT(I))
C       FIXUP FOR CASE OF INPUT IBM F.P. ZERO'
        IF (AIBM.EQ.0.D0) RETURN
        AIBM=SIGN*AIBM*16.D0**(IT(1)-'4E'X)
        RETURN
        END     ! AIBM

        REAL*4 FUNCTION SIBM(BY)

        BYTE BY(4), BYT(8)
        REAL*8 AIBM
        DO 100 I =1, 4
        BYT(I) = BY(I)
100     BYT(I+4) = '00'X
        SIBM = AIBM(BYT)
        RETURN
        END  !SIBM

        subroutine invsao(isao,iy,mo,iday)
        IMPLICIT DOUBLE PRECISION (A-H,O-Z)
        dimension month(12)
        logical leap
        data month/0,31,59,90,120,151,181,212,243,273,304,334/
        idy(iy)=iy*365+iy/4-iy/100+iy/400
        is=isao+678576
        if(is .le. 0) return
        iy=ifix(float(is)/365.2424)+1           !in the form 19xx
        leap=mod(iy,4) .eq. 0 .and. (mod(iy,100).ne. 0 .or.
     1  mod(iy,400) .eq. 0)             !is this a leap yr?
        id=is-idy(iy-1)
        if(id .gt . 0)go to 4                   !valid day number.
        iy=iy-1                                 !bad day; fix it
        id=is-idy(iy-1)
        go to 6
4       if(id .le. 365 .or. (leap .and. (id .le. 366)))go to 6
        iy=iy+1
        id=is-idy(iy-1)
6       do 7 i=1,12
7       if(month(i) .LT. id)mo=i
        iday=id-month(mo)
        if(.not. leap .or. (leap .and. mo .le . 2))go to 50
        iday=iday-1
        if(iday .gt. 0) go to 50
        mo=mo-1                                 !back up one month.
        iday=id-1-month(mo)
        if(id .eq. 60)iday=29
50      iy=iy-1900
100     return
        end

C  LASERG PROGRAM FOR FINAL DATA.
```

```
C
C**** NOTE:      THIS IS NOT AN OPERATIONAL VERSION.  CODE NOT RELEVANT
C****            TO IN PUT/OUTPUT FORMATS AND DATA TRANSFORMATIONS HAS
C****            BEEN LARGELY OMITTED (INDICATED BY  . . . ).
C                       G. GULLAHORN  1 OCT 81
C
C  THIS PROGRAM TAKES THE INTERNAL FORMAT FILE CREATED BY Q,FLAC,LASERT,
C  etc. AND PRODUCES TWO OUTPUT FILES.  THE FIRST,CALLED SAODATA.DAT
C  IS IN THE 'STANDARD' SAO FORMAT; THE SECOND, CALLED GODDARD.DAT,
C  IS THE STANDARD NASA FORMAT.
C
C  G. GULLAHORN,  MAR 1980:
C  SAODATA.DAT IS NOW CALLED SO<sat mnem><mo><y>.DAT
C  GODDARD.DAT IS NOW CALLED GO<sat mnem><mo><yr>.DAT, E.G.
C  SOG31280.DAT FOR GEOS3 DATA DECEMBER 1980
C
 . . .
C
      DATA TIMING/0.,.003,.002,.005,.02,.05,.2,.5,2.0/
      DATA TCODE/1,2,3,4,5,6,7,8,9/
      INTEGER*4 SATID,OBSNO,NANORNG
      INTEGER*2 STAT,OBSTYPE,EPCHSYST,OBSUNITS,CALINDX,CINDX
      INTEGER*2 YR,MO,DAY,HR,MIN,TCODE(9),TIMCODE,TMICRO,SIGMA
      INTEGER*2 TSYSIND
      CHARACTER*17 INPUT
      REAL*4 REFCORR,TIMEPRE,RNGPRE,CALSTAB,CMCORR,PRESS,HUM
      REAL*4 TEMP,PELEV,PAZ,RNGNOISE,RNGBIAS,TIMING(9)
      REAL*8 OBSRNG,PRANGE,SEC,C,CONVRNG
      BYTE BARRAY(128)
C
      EQUIVALENCE(SATID,BARRAY(1)),(OBSNO,BARRAY(5)),(STAT,BARRAY(9))
     $ ,(OBSTYPE,BARRAY(23)),(OBSRNG,BARRAY(25)),(REFCORR,BARRAY(33)),
     $ (TIMEPRE,BARRAY(37)),(RNGPRE,BARRAY(41)),(EPCHSYST,BARRAY(45)),
     $ (OBSUNITS,BARRAY(47)),(CALSTAB,BARRAY(49)),(CALINDX,BARRAY(53)),
     $ (CMCORR,BARRAY(55)),(PRESS,BARRAY(59)),(HUM,BARRAY(63)),
     $ (TEMP,BARRAY(67)),(PELEV,BARRAY(71)),(PRANGE,BARRAY(75)),
     $ (PAZ,BARRAY(83)),(CINDX,BARRAY(87)),(RNGNOISE,BARRAY(89)),
     $ (RNGBIAS,BARRAY(93)),(YR,BARRAY(97)),(MO,BARRAY(99)),
     $ (DAY,BARRAY(101)),(HR,BARRAY(103)),(MIN,BARRAY(105)),
     $ (SEC,BARRAY(107))
C
 . . .
C
      TYPE *,' ENTER NAME OF INPUT FILE(USE QUOTES)'
      ACCEPT *,INPUT
      OPEN(UNIT=20,NAME=INPUT,TYPE='OLD',FORM='UNFORMATTED')
C
 . . .
C**** (FILE NAMES "SFILE" "GFILE" ARE GENERATED IN OMITTED CODE.)
 . . .
C  OPEN OUTPUT FILES,FORMERLY SAODATA.DAT, GODDARD.DAT
      OPEN(UNIT=21,NAME=SFILE,TYPE='NEW',RECORDSIZE=108)
      OPEN(UNIT=22,NAME=GFILE,TYPE='NEW',RECORDSIZE=90)
C
```

```
 . . .
C
C  SET UP CONSTANTS USED FOR GODDARD FORMAT.
        IZERO=0
        IONE=1
        C=299792.458D+00
        MEASTYPE=20
        TSYSIND=23
        ICOL=151
        CONVRNG=(C/(2.0D+08))*(1.0D+04)
C
C  MAJOR LOOP ::  READ AN OBSERVATION FROM INPUT FILE:::
C
10      READ(20,END=200)BARRAY
C
 . . .
C
        IREFCORR=NINT(REFCORR*100.)
        DO 50 I=1,8
        IF(TIMEPRE.GE.TIMING(I).AND.TIMEPRE.LE.TIMING(I+1))THEN
           TIMCODE=TCODE(I)
           GO TO 60
        END IF
50      CONTINUE
        TIMCODE=9
60      IRNGNOISE=NINT(25*(LOG10(RNGNOISE)+3))
        IRNGBIAS=NINT(25*(LOG10(RNGBIAS)+3))
        SIGMA=NINT(25*(LOG10(RNGPRE)+3))
        ISEC=NINT(SEC*10000000.)
        NANORNG=NINT(OBSRNG*10.)
        ICALSTAB=NINT(ABS(CALSTAB*10.))
        ICMCORR=NINT(CMCORR*100.)
        IPRESS=NINT(PRESS)
C       ONE-SHOT PROCESSING OF GENE & ALAN'S JAN/FEB 79 DATA REQUIRES
C       FOLLOWING PATCH TEMPORARILY:
        IF(HUM .LT. 1.0)HUM=HUM*100.
        IHUM=NINT(HUM)
        ITEMP=NINT(TEMP*10.)
        IPELEV=NINT(PELEV*1000.)
        IPRANGE=NINT(PRANGE*10.)
        IPAZ=NINT(PAZ*1000.)
C
C  WRITE A RECORD IN SAO FORMAT.
        WRITE(21,1000)SATID,OBSNO,STAT,YR,MO,DAY,HR,MIN,ISEC,
     $  NANORNG,IREFCORR,TIMCODE,SIGMA,OBSTYPE,EPCHSYST,OBSUNITS,
     $  ICALSTAB,CALINDX,ICMCORR,IPRESS,IHUM,ITEMP,IPELEV,IPRANGE,
     $  IPAZ,CINDX,IRNGNOISE,IRNGBIAS
1000    FORMAT(I7,2(I5),5(I2),I9.9,I10,2X,I4,I1,I2,3(I1),I3.3,I1,2(I4),
     $  I2,I4,I5,I11,1X,I7,1X,3(I2))
C
C  NOW SET UP FOR GODDARD OUTPUT FORMAT.
C  REFER TO THE GODDARD NOTES FOR A DESCRIPTION OF THE INFORMATION
C  IN EACH COLUMN.
        CALL FINDDAY(IYD,YR,MO,DAY)
```

```
        IGSEC=HR*3600.+MIN*60.+SEC
        J=SEC
        IFRSEC=NINT((SEC-J)*1000000.)
        IRNG=NINT(OBSRNG*CONVRNG)
        KTEMP=NINT(TEMP+273.15)
        MRNGBIAS=NINT(RNGBIAS*1000.)
        IGREFCORR=NINT(REFCORR*1000.)
        IGCMCORR=NINT(CMCORR*1000.)
        IGTIMPRE=LOG10((TIMEPRE+.0000005)*10**6)
C
C  WRITE A RECORD IN GODDARD FORMAT.
       WRITE(22,1001)SATID,MEASTYPE,TSYSIND,STAT,YR,IYD,IGSEC,
     $ IFRSEC,ICOL,IRNG,IZERO,IPRESS,KTEMP,IHUM,MRNGBIAS,
     $  IGREFCORR,IONE,IONE,IGCMCORR,IGTIMPRE
1001    FORMAT(I7,2(I2),I5,I2,I3,I5,I6.6,I3,5X,I10,4X,I1,1X,I4,2(I3)
     $  ,2X,I5,2X,I5,2(I1),I6,I2)
        GO TO 10
C
C       TERMINATE
C
 200    CLOSE(UNIT=20,DISP='SAVE')
        CLOSE(UNIT=21,DISP='SAVE')
        CLOSE(UNIT=22,DISP='SAVE')
 . . .
        END

        SUBROUTINE FINDDAY(GDDAY,YEAR,OMNTH,DAY)
C
C  THIS WAS ADAPTED FROM GENE CAMPBELL ROUTINE CALLED SMTHDAY
C  WHICH WAS USED BY HIS 6400 LASERG PROGRAM.
C  ALL I WANT TO DO WITH THIS ROUTINE IS RETURN THE DAY OF THE
C  YEAR(e.g. 93,281,365,etc.) FOR THE GODDARD FORMAT.
C
        INTEGER*2 YEAR,OMNTH,DAY,MO,J
        INTEGER GDDAY
        DIMENSION DZ(10)
        DATA DZ/334.,304.,273.,243.,212.,181.,151.,120.,90.,59./
C
        Y =YEAR
        M=OMNTH+.8
        D=DAY
        X=365.25*Y+.9
        J=X
        A=J
        MO=M-2
C
        IF(MO.GT.0)THEN
                IF((X-A-.8).LE.0.0)THEN
                    C=0.
0               ELSE
                C=1.
                ENDIF
                J=13-M
                E=DZ(J)+C
```

```
      ELSE IF(MO.EQ.0)THEN
      E=31.
      ELSE IF(MO.LT.0)THEN
      E=0.
      END IF
      GDDAY1=15018.+A+D+E
      GDDAY2=15018.+A
      GDDAY=GDDAY1-GDDAY2
      RETURN
      END
```

```
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
C
C LUNAR PERTURBATION OVERLAY
C INCORPORATED GRIPE LUNAR PERTURBATION INTO MINI  ON 23 MARCH 1981
C COMPLETED WORKING VERSION ON APRIL 1, 1981.  THIS VERSION
C USES THE LUNI-SOLAR PERTURBATION THEORY DEVELOPED BY
C Y. KOZAI AS REPORTED IN SMITHSONIAN SPECIAL REPORT 349.  ALL SOURCE
C CODES ARE ON FLPPS DISK ONE.  THE SUBROUTINES USED IN THIS OVERLAY
C ARE:
C
C RDZON---LOADS REGISTERS WITH ZONAL HARMONICS
C INST---CALCULATES INSTANTANEOUS ELEMENTS AT EPOCH OF OBSERVATION
C GETSMA---CALCULATES SEMI-MAJOR AXIS
C NFINC---CALCULATES INCLINATION FUNCTION
C FACCAL---COMPUTES FACTORIALS
C HANSEN---COMPUTES ECCENTRICITY FUNCTION
C LUNAR---CALCULATES LONG PERIOD AND SHORT PERIOD LUNI-SLAR PRTS
C SETUP---ASSIGN VARIABLES FOR INTEGRATION
C KIND---ROUTINE TO INTEGRATE TERMS FOR LUNAR PERTS
C SUNVECT---CALCULATES VECTOR TO SUN
C LUNVECT---CALCULATES VECTOR TO MOON
C PRECESS---CALCULATES TERMS DUE TO PRECESSION
C EVA---CO MPUTES SIN AND COS TERMS FOR ECC AND MA
C
C THE FUNCTIONS USED IN THIS OVERLAY ARE
C
C CONSOC---STORES THE CONSTANTS USED FOR OVERLAY
C ERIQA---SOLVES KEPLER'S EQUATION
C LOAD---TAKES LOWER ORDER 4 BYTES FROM A REAL*8 AND PUTS
C        THEM INTO AN INTEGER*4
C PUT---PUTS INTEGER*4 INTO BITS 0-3 OF REAL*8 WORD.
C ASIN---FINDS ARCSIN
C ATANG---FINDS ARCTAN
C
C THE LIBRARIES USED IN THIS OVERLAY ARE
C
C STDLUNLIB---CONTAINS BINARY FOR ALL OF ABOVE
C WFWLTUTIL---SYSTEM UTILITIES
C WFWRUN---FORTRAN UTILITIES
C WFWSOS---SYSTEM UTILITIES
C
      DIMENSION PM(8,12),TEMP(22),T(2),IP(6),TLUN(2)
      DIMENSION NAMST(10),ISTP(3),IANOM(23),ERLAT(5)
      COMMON ORB(300)
      COMMON/IOBLK/NXT,IN,IOUT
C
A     .EXTN;.DSI,.STTY,LTODC,LT1DC,.LEXD

      CALL READY;;LOAD POWER FAIL CODE
      CALL FOPEN(6,"$LEX")
      CALL FOPEN(10,"$TTO")
      WRITE(6,1)
    1 FORMAT(" COMPUTING COEFFICIENTS FOR LUNAR PERTURBATIONS")
```

```
      WRITE(6,11)
 11   FORMAT("S.D.V. VERSION 6.3B 9/9/81")
C
C INITIALIZE THE I/O FILES
C
      CALL RUBOUT("SCRATO")
      CALL FOPEN(1,"SCRAT1") ;INPUT FILE
      CALL FOPEN(0,"SCRATO,0") ;OUTPUT FILE
      IN=1
      IOUT=0
      JJ=1
      J=0
      ISTP(1)=2H99
      ISTP(2)=2H90
      CALL PUTC(ISTP(3),1,0)
      CALL PUTC(ISTP(3),2,0)
C
C  READ IN ID#'S OF SATELLITES AND # OF SECONDS FOR EARLY/LATE
C  CORRECTION
C
   23 IGO=IRDWX(35,IANOM(JJ))
      IF(ISTEQ(IANOM(JJ),ISTP)) GO TO 25
      JJ=JJ+4
      J=J+1
      IGO=IRDWX(36,ERLAT(J))
      GO TO 23
C
C READ IN PASS START TIME, # DAYS
C
   25 IGO=IRDWX(25,T)
   10 CONTINUE
C
C PREDICTION TIME ADJUSTED TO ACCOUNT FOR
C PASS START BEFORE TIME OF PREDICTION
C
      TLUN(1)=T(1)-.125
      TLUN(2)=T(2)
      IGO=IRDXL(51,NAMST,LEN,ITYPE)
      IF(IGO.EQ.0) GO TO 90
      CALL IWRX(51,NAMST,LEN,ITYPE)
      WRITE(6,2) (NAMST(I),I=1,LEN)
    2 FORMAT(1X,10A2)
      IGO=IRDWX(52,TEMP)
      IGO=IRDWX(53,IP)
      IGO=IRDX(54,TEMP(3))
      ISTEP=IP(2)+1
C
C  TEST EPOCH TO MAKE SURE THE CORRECT YEAR WAS TYPED IN.
C
       AHILM=TEMP(1)+TEMP(2)+30.
       ALOLM=TEMP(1)+TEMP(2)-30.
       IF(T(1).LE.AHILM.AND.T(1).GE.ALOLM) GO TO 28
       CALL PUTC(IRING,1,7)
       CALL PUTC(IRING,2,7)
```

```
      DO 27 I=1,10
  27  WRITE(6,1000) IRING
 1000 FORMAT(1X,A2)
      I=IOK(" YOUR INPUT TIME IS MORE THAN 30 DAYS DIFFERENT FROM
     1  THE EPOCH OF THE ELEMENTS. DO YOU WANT TO CONTINUE?(Y OR N)")
      IF(I.EQ.0) CALL CHAIN("GO")
C
C  IF MAKING A CORRECTION TO THE MEAN ANOMALY TERM, ADD CORRECTION
C  TO TEMP(IP(4)+3).
C
   28 IF(J.EQ.0) GO TO 35
      M=0
      DO 30 K=1,JJ,4
      M=M+1
      IF(ISTEQ(NAMST,IANOM(K))) GO TO 32
   30 CONTINUE
      GO TO 35
   32 CORR=-ERLAT(M)
      ISCND=IP(4)+4
      REV=TEMP(ISCND)
      SECRV=86400./REV
      AMNAN=CORR/SECRV
      IFRST=ISCND-10034
          TEMP(IFRST)=TEMP(IFRST)+AMNAN
C
   35 CONTINUE
      CALL IWRX(57,TEMP,76,4)
C  CHANGE NODE TO NODE OF 1950
C   TEMP(ISTEP)=TEMP(ISTEP)-(TEMP(1)-33281.)*3.508E -5
      IGO=IRDWX(16,MOON)
      IF(MOON.EQ.0) GO TO 10
C
C  STARTING GRIPE LUNAR CODE, 23 MARCH 1981
C
      DO 14 I=1,300
   14 ORB(I)=0.0
      NOD=12
      NUN=0
      IND=1
      ORB(265)=TEMP(1)+TEMP(2)
      ORB(266)=0.30
      ORB(267)=0.30
C
C INTERVAL OVER WHICH TABLE OF PERTURBATION VALUES ARE PREPARED
C FOR EPHEMIS T(2)+. 25.  T(2) IS AMOUTNT OF TIME OVER WHICH
C PREDICTIONS ARE WANTED, .25 IS 1/8TH DAY ALLOWANCES TO COMPLETE
C PASS IN EITHER DIRECTION
C
      DENT=(T(2)+.25)/11.
      ORB(1)=PUT(10)
      ORB(2)=PUT(21)
      ORB(3)=PUT(111)
      ORB(4)=PUT(121)
      ORB(5)=PUT(201)
```

```
      ORB(6)=PUT(250)
      ORB(7)=PUT(222)
      ORB(9)=PUT(260)
      III=LOAD(ORB(3))
      IAA=LOAD(ORB(4))
      IXX=LOAD(ORB(2))
      ORB(287)=PUT(0)
C
C  THE FOLLOWING ARE INTEGRAL AND FRACTIONAL EPOCH AT WHICH
C  ELEMENTS ARE DEFINED.  THEY WILL NOT CHANGE IN RUN
C
      ORB(10=TEMP(1)
      ORB(11)=TEMP(2)
      ORB(130)=TEMP(1)
      ORB(131)=TEMP(2)
C
C  DIFFERENCE IN TIME FROM EPOCH OF ELEMENTS AND EPOCH OF PREDICTION
C
      ORB(121)=T(1-(TEMP(1)+TEMP(2))
      DELT=ORB(121)
      DO 240 I=143,158
 240  ORB(I)=TEMP(I-140)
      CALL RDZON
      ORB(138)=IP(1)
      ORB(139)=(IP(2)-IP(1))
      ORB(40)=(IP(3)-IP(2))
      ORB(141)=IP(4)-IP(3)
      ORB(142)=IP(5)-IP(4)
      ORB(III+2)=0
      ORB(III+3)=PUT(1)
      TLUN(2)=0.0
      DO 250 J=1,NOD
      CALL INST
      DO 249 I=1,5
  249 ORB(IXX+I-1)=ORB(IAA+I-1)
      ORB(IXX+5)=ORB(IAA+6)
      ORB(IXX+10)=ORB(IAA+7)
      ORB(IXX+14)=ORB(IAA+5)
      ORB(IXX+11)=ORB(IAA+8)
      CALL LUNAR(TLUN)
C
C PLACE EPOCH AND PERTURBATIONS IN PM ARRAY FOR USE IN EPHEM
C
      DO 444 JW=2,6
      PM(1,IND)=TLUN(1)+TLUN(2)
      PM(JW,IND)=ORB(69+JW)
  444 CONTINUE
C CONVERT MA PERTS TO RADIANS
      PM(6,IND)=PM(6,IND)*CONSOC(2)
      PM(7,IND)=ORB(79)
      PM(8,IND)=ORB(80)
      IND=IND+1
      DELT=DELT+DENT
      TLUN(1)=TLUN(1)+DENT
```

```
      ORB(IAA)=DINT(DELT)
      ORB(IAA+1)=DELT-ORB(IAA)

  250 CONTINUE
C
C  END OF GRIPE LUNAR MODIFICATIONS
C
      CALL IWRX(17,PM,384,4)
      GO TO 10
   90 CALL IWRX(0,0,0,0)
      CALL FCLOS(0)
      CALL FCLOS(1)
      CALL FCLOS(10)
      CALL CHAIN("TESSER")
      END



      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE  RDZON
C
C EMGA GAPOSCHKIN
C
C       LOAD REGISTERS WITH ZONAL HARMONICS
        COMMON ORB(300)
        COMMON/LABE2/ZON(21)
        DATA  ZON/17.043570D0,  1.08262679D-03, -2.5356D-06,-1.6234D-06,
     1 -2.2759D-07,  5.4337D-07, -3.6066D-07, -2.0702D-07, -1.2002D-07,
     1-2.4111D-07,  2.3295D-07, -1.9312D-07, -2.2861D-07,  1.2378D-07, -
     17.9890D-09,  4.1823D-08, -9.9068D-08, -6.0868D-08, -1.2610D-09, -1
     1.5175D-07, -6.7624D-10  /
        I=LOAD(ORB(5))
        DO  1  J=1,21
        K=I+J-1
     1  ORB(K)=ZON(J)
C
        RETURN
        END


        COMPILER DOUBLE PRECISION
        COMPILER NOSTACK
        SUBROUTINE INST
C
        REAL II
        REAL DX(7)
        COMMON DUMMY(110),II(10),AA(80),ZON(21),PER(8,3),DUCO(55)
        COMMON/INSTL/TFP,TIP,TZIP,TZFP,IFF,PIP,PFP,A,B,L,XL,
       1QQQQ,SUM,ISAVE,ICF,XG
C * * * * * * * * * * * * * * * * * * * * * * * * ** * * * * * * * * * * * * *
C           THESE MODIFICATIONS BY E.M.G.
        II(5)=PUT(9)
        ISTAR=9
```

```
       RADIAN=CONSOC(7)
C
C      ICF POINTS TO COEFFICIENTS
       ICF=ISTAR+13
C  IPT POINTS TO POLYNOMIAL DEGREE
       IPT=ISTAR+8
C      STAR+8
C      SAVE T
       T1=AA(1)
       T2=AA(2)
       T=T1+T2
       I=ISTAR
C      SET UP TIME
       TIME=(AA(I+1)+T)+AA(I+2)
C      LOOP OVER 5 ELEMENTS (ICF IS CONTINUALLY UPDATED AS SUCCESSIVE
C      COEFFICIENS ARE REFERENCED.)
       DO 139 J=1,5
       IF(J-1)400,400,401
  400  AA(8)=AA(ICF+2)/RADIAN
C   STORE RATE OF PERIGEE (AT EPOCH)
       WDOT=AA(8)
  401  IF(J-2)403,402,403
  402  AA(9)=AA(ICF+2)/RADIAN
  403  CONTINUE
       SU= 6
       SUM=0.
       ISAVE=ICF
       IPT=IPT+10014
           K=AA(IPT)
       DO 160 M=1,K
       I=ICF+K+1-M
  160  SUM=SUM*T+AA(I)
       ICF=ICF+K
C      IF AP, AN, OR I, TAKE MOD 360 DEGREES, AND CONVERT TO RADIANS
       IF(J-3)166,166,167
  166  L=SUM/360.
        XL=L*360
       SUM=(SUM-XL)/RADIAN
       GO TO 205
C      IF MA, TAKE MOD. 1
  167  IF(J-5)205,204,205
  204  L=SUM
       XL=L
       SUM=SUM-XL
       IF(SUM)206,205,205
  206  SUM=1.+SUM
  205  I=J
  139  AA(I)=SUM
C      COMPUTE MEAN MOTION (DERIVATIVE OF POLYNOMIAL PART
C      OF MEAN ANOMALY).
       AA(6)=0.
       K=AA(IPT)-1
       DO 168 L=1,K
       IC=K+2-L
```

```
      I=ISAVE+IC
      XL=IC-1
  168 AA(6)=AA(6)*T+AA(I)*XL
C     COMPUTE SEMI-MAJOR AXIS
CMEAN MOTION IN REV/CTU
       XN=AA(6)*CONSOC(6)
C  RATE OF PERIGEE IN DEGREES PER CU
      XG=AA(8)*RADIAN*CONSOC(6)
      LEND=20
C
      CALL GETSMA(ZON,XN,XG,AA(4),AA(3),AA(7),LEND,II(2))
C
C  CONVERT TO MEGAMETERS
       AA(7)=AA(7)*CONSOC(5)
       RETURN
       END

      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE GETSMA(ZON,RMM,RMG,ECC,AI,SMA,LEND,IFLAG)
C
      REAL IFLAG
      REAL N,N1
      DIMENSION ZON(1)
      COMMON/SECT/SEC(2,20)
      COMMON/DIF/IFIRST
C
      DATA IFIRST/0/
C
      N=RMM*CONSOC(2)+RMG*CONSOC(1)/180.
      C=COS(AI)
      S=SIN(AI)
      C2=C*C
      C4=C2*C2
      E2=ECC*ECC
      ETA2=1.-E2
      ETA=SQRT(ETA2)
      A=(1./N**2)**(1./3.)
      IEND=LEND/2
C
      IF(IFLAG.EQ.1.AND.IFIRST.EQ.1) GO TO 11
      IFLAG=1
      IFIRST=1
C  COMPUTE SEC ONLY ON 1ST OBS/ITERATION OR FIRST TIME GETSMA CALLED
      DO 10 I=1,IEND
      L=I*2
      CALL NFINC(L,0,I,S,C,FXI,DFXI,DD)
      CALL HANSEN(-L-1,L-2*I,L-2*I,ECC,GE,DGE)
      SEC(1,I)=(-GE*DFLOAT(L+1)*2.+ETA2/ECC*DGE)*FXI*ZON(L)
      SEC(2,I)=(C/S/ETA*DFXI*GE-ETA/ECC*FXI*DGE)*ZON(L)
C
   10 CONTINUE
   11 CONTINUE
      T=10.*(1.-6.*C2+13.*C4)-5.*(5.-18.*C2+5.*C4)*E2+16.*ETA*(1.-3.
```

```
     1*C2)**2
      T=3./128.*T*ZON(2)**2/ETA**7
      W=35.-90.*C2-385.*C4+4.*ETA*(-6.+48.*C2-90.*C4)+ETA2*(-25.+12
     16.*C2-45.*C4)
      W=-W*3./128./ETA2**4*ZON(2)**2^^^^^^^^^^
      DO 20 J=1,10
      X=1.
      DO 21 I=1,IEND
      L=I*2
      AL=A**L
   21 X=X+(SEC(1,I)+SEC(2,I))/AL
      X=X+(T+W)/A**4
      N1=N/X
      A1=(1./N1**2)**(1./3.)
      IF(ABS(A-A1).LT.1.D-9) GO TO 30
      A=A1
   20 CONTINUE
   30 SMA=A10011
          RETURN
      END


     COMPILER DOUBLE PRECISION
     COMPILER NOSTACK
     SUBROUTINE HANSEN(N,M,I,E,HA,DHA)
C----------HA IS HANSEN COEFFICIENT.
C----------DHA IS DERIVATIVE OF HA WITH REPECT TO ECCENTRICITY.
C---------- REFERENCE  ,TISSERAND,VOL.1,P.260.
C---------H.KINOSHITA      10 APRIL 1975
C----------REVISED IN MAY 1976
      ETA=DSQRT(1.-E**2)
      BE=E/(1.+ETA)
      DBE=1./(1.+ETA)+(E/(1.+ETA))**2/ETA
      AN=ETA*DFLOAT(I)
      DX=E/ETA*DFLOAT(I)
      IM=IABS(I-M)
      ND=N-I
      NDD=N+I
      XD=-AN
      XDD=AN
      IF(I-M)5,20,20
    5 ND=N+I
      NDD=N-I
      XD=AN
      XDD=-AN
      DX=-DX
   20 K=0
      CALL PQ(ND,IM,XD,DX,P,DP)
      E0=P
      HA=P
      DHA=DP
      IF(E0.EQ.0.)GO TO 900
   10 K=K+1
      CALL PQ(ND,IM+K,XD,DX,P,DP)
```

```
      CALL PQ(NDD,K,XDD,-DX,Q,DQ)
      E1=P*Q*BE**(K*2)
      EPS=DABS(E1/E0)

      HA=HA+E1
      DHA=DHA+(DP*Q+P*DQ)*BE**(K*2)+P*Q*BE**(K*2-1)*DFLOAT(K*2)*DBE
      IF(EPS.GT.1.D-6)GO TO 10
      FAC=(1.-BE**2)**(N*2+3)/(1.+BE**2)**(N+1)*(-BE)**IM
      DFAC=BE**(IM-1)*(1.-BE**2)**(N*2+2)*(1.+BE**2)**(-N-2)*(DFLOAT
     1(IM)-DFLOAT(2*(3*N+4))*BE**2-DFLOAT(IM+2*(N+2))*BE**4)
      DFAC=DFAC*DBE*(-1.)**IM
      DHA=DFAC*HA+FAC*DHA
      HA=FAC*HA
  900 RETURN
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE NFINC(K,M,L,S,C,FXI,DFXI,DDFXI)
C----------H.KINOSHITA JAN 1973
C----------REVISED APRIL 1976
C----------WHEN I=0,THIS SUBROUTINE DOES NOTWORK.
      DIMENSION FAC(0:25),U(25)
      COMMON/IW1/IFACT
      COMMON/B/FAC,U
      IF(IFACT.EQ.11) GO TO 100
      CALL FACCAL(25,25)
      IFACT=11
  100 CONTINUE
      FAC(0)=1
      I=(K-M+1)/2+1
      I1=(K-L)*2+1
      I2=K-L+1
      N=K
      MD=K-L*2
      L1=L+1
      XNKD1=DFLOAT(K)
      C1=U(I)*FAC(I1)/FAC(L1)/FAC(I2)/(2.**XNKD1)
      SS=Q(K,M,MD,C)
      FXI=SS*C1
      W1=Q(K,M-1,MD,C)
      W2=Q(K,M-2,MD,C)
      NC1=(N+M)*(N-M+1)
      C2=DFLOAT(NC1)
      DFXI=(C2*W1+SS*(DFLOAT(MD)-DFLOAT(M)*C)/S)*C1
      C3=DFLOAT((N+M-1)*(N-+2))*C2
      C4=C2*(DFLOAT(MD*2)-DFLOAT(M*2-1)*C)/S
      C5=(DFLOAT(M)-DFLOAT(MD)*C+(DFLOAT(MD)-DFLOAT(M)*C)**2)/S**2
      DDFXI=(W2*C3+W1*C4+SS*C5)*C1
      RETURN
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE FACCAL(N,M)
      DIMENSION FAC(0:25),UP(25)
```

```
      COMMON/B/FAC,UP
C  THIS THE MAXIMUM NUMBER THAT CAN BE FACTORIALIZED CURRENTLY
C  WITHOUT OVERFLOW
      COMMON/MAXFA/MAXFAC
      DATA MAXFAC/25/
      NMAX=N
      MMAX=M
      IF(NMAX.LE.MAXFAC) GO TO 6
C  FACTORIAL OUT OF BOUNDS STOP
      STOP
    6 CONTINUE
C
      FAC(1)=1.
      DO 10 I=2,NMAX
   10 FAC(I)=FAC(I-1)*DFLOAT(I-1)
      IF(M.LE.MAXFAC) GO TO 160
C   FACTORIAL OUT OF BOUNDS STOP
      STOP
   16 CONTINUE
      UP(1)=1.
      DO 20 I=2,MMAX
   20 UP(I)=UP(I-1)*(-1.)
      RETURN
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE PQ(N,K,X,DX,P,DP)
      DIMENSION F(0:25),UP(25)
      COMMON/IW1/IFACT
      COMMON/B/F,UP
      IF(IFACT.EQ.11) GO TO 20
      CALL FACCAL(25,25)
      IFACT=11
   20 CONTINUE
      IF(K)40,30,40
   30 P=1.
      DP=0.
      RETURN
   40 CONTINUE
      IF(X.EQ.0.)GO TO 60
      P=0.
      P=0.
      K1=K+1
      DO 50 J1=1,K1
      J=J1-1
      KJ=K-J+1
      HP= FN(N+1+K,N+1+J)/F(J1)/F(KJ)*X**J
      P=P+HP
      DP=DP+HP*DFLOAT(J)/X
   50 CONTINUE
      DP=DP*DX
      RETURN
   60 CONTINUE
      P=FN(N+1+K,N+1)/F(K+1)
```

```
      DP=0.
      RETURN
      END
      COMPILER DOUBLE PRECISIION
      COMPILER NOSTACK
      FUNCTION Q(N,M,MD,C)
      DIMENSION FAC(0:25),U(25)
      COMMON/B/FAC,U
      IF(M)  10,11,11
   10 IF(MD)  20,21,21
   20 M1=-M
      MD1=-MD
      I=M1-MD1
      IF(I)  22,23,23
   22 I=-I+1
      GO TO 24
   23 I=I+1
   24 I1=N+MD1+1
      I2=N-MD1+ 1
      I3=N-M1+1
      I4=N+M1+1
      Q  =QQQP(N,M1,MD1,C)*U(I)*FAC(I1)/FAC(I2)*FAC(I3)/FAC(I4)
      RETURN
   21 M1=-M
      I=N-MD+1
      I1=N-M1+1
      I2=N+M1+1
      DC=-C
      Q=  QQQP(N,M1,MD,DC)*U(I)*FAC(I1)/FAC(I2)
      RETURN
   11 IF(MD)  30,31,31
   30 MD1=-MD
      DC=-C
      I=N-M+1
      I1=N+MD1+1
      I2=N-MD1+1
      Q=  QQQP(N,M,MD1,DC)*U(I)*FAC(I1)/FAC(I2)
      RETURN
   31 Q=  QQQP(N,M,MD,C)
      RETURN
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION QQQP(N,M,MD,C)
      DIMENSION FAC(0:25),U(25)
      COMMON/B/FA,U
      IF(M-MD)  10,11,11
   10 M1=MD
      MD1=M
      I=MD-M+1
      I1=N+M+1
      I2=N-M+1
      I3=N-MD+1
      I4=N+MD+1
```

```
      QQP=QP(N,M1,MD1,C)*U(I)*FAC(I1)/FAC(I2)*FAC(I3)/FAC(I4)
      RETURN
   11 QQQP=QP(N,M,MD,C)
      RETURN
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION QP(N,M,MD,C)
C    H.  KINOSHITA  JAN  U1973
      DIMENSION FAC(0:25),UP(25)
      COMMON/B/FAC,UP
      S=0.
      C1=1.D30
      A=1.-C
      B=1.+C
      IRE=N-M+1
      DO  10  I=1,IRE
      IR=I-1
      I1=M-MD+IR+1
      I2=N+MD-IR+1
      I3=N-M-IR+1
      IF(I3.EQ.0) I3=1
      S=S+C1*UP(I)/FAC(I1)/FAC(I2)/FAC(I)/FAC(I3)*FQP(A,IR)*FQP(B,I3-
     11)
  10  CONTINUE
      I1=N+M+1
      I2=N-MD+1
      C2=1./C1*FAC(I1)*FAC(I2)/2.**N
      C3=FQP(A,M-MD)*FQP(B,M+MD)
      C3=DSQRT(C3)
      QP=S*C2*C3
      RETURN
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION FQP(A,M)
      IF(M.EQ.0)GO TO 10
      FQP=A**M
      RETURN
   10 FQP=1.
      RETURN
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION FN(N,M)
      FN=1.
      ID=IABS(N-M)
      IF(ID)20,10,20
   10 RETURN
   20 DO 30 I=1,ID
      FN=FN*DFLOAT(N-I+1)
   30 CONTINUE
      RETURN
      END
```

```
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION CONSOC(I)
      REAL ITEM(10)
      COMMON/ITE/ITEM,GM,ROTE,SECDAY
      DATA ITEM/
     1  3.141592653589793D0,
     2  6.283185307179586D0,
     3  0.D0,0018     4  1.D20,
     5  6.378136D0,
     6  0. D0,
     7  57.29577951308232D0,
     8  0.D0,
     9  1.D20,
     A  1.D20/
C  GM  (MM**3/SEC**2)
      DATA GM/3.986005D-4/
C  ROTATION RATE OF EARTH (RAD/SEC)
      DATA ROTE/.7292115085D-4/
C  SEC/DAY
      DATA SECDAY/86400.D0/
C
C
  100 CONTINUE
      CONSOC=ITEM(I)
      IF(ITEM(I).NE.0) RETURN
C  CALCULATE CONSOC(3), CONSOC(6), CONSOC(8)
C    GE DAYS/CTU
      ITEM(6)=DSQRT(ITEM(5)**3/GM)/SECDAY
C  ROT EARTH (RAD/CTU)
      ITEM(8)=ROTE*ITEM(6)*SECDAY
C  GM (MM**3 (REV/DAY)**2 )
      ITEM(3)=GM*(SECDAY/ITEM(2))**2
      GO TO 100
      END


      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION ERIQA (AMA,ECC,SINE)
C     SOLVING KEPLER"S EQUATION
C     USING THEDOI SCHEME
C     RUDOLF LOESER, APRIL 1965
      KOUNT=0
      CONV=1.D-12
      MA=AMA
      YMA=MA
      XMA=(AMA-YMA)*CONSOC(2)
      EP=XMA
      DEP=0.
  100 SINE=DSIN(EP)
      KOUNT=KOUNT+1
```

```
      ERIQA=XMA+ECC*SINE
      DE=ERIQA-EP
      IF(ABS(DE)-CONV  )104,101,101
  101 IF(DEP)103,102,103
  102 DEP=DE
      EP=ERIQA
      GO TO 100
  103 EP=EP+DE*(DEP/(DEP-DE))
      DEP=0.
      GO TO 100
  104 CONTINUE
      RETURN
      END

      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK

      SUBROUTINE EVA (AMA,ECC,SINE,COSE,SINV,COSV,E,V)
C     RUDOLF LOESER, 1 MAR 66
C     USES THE VALUES OF AMA AND ECC TO COMPUTE THE VALUES OF
C     SIN(E), COS(E), SIN(V) AND COS(V).
      PI=CONSOC(1)
      TOPI=CONSOC(2)
      E=ERIQA(AMA,ECC,SINE)
      COSE=COS(E)
      C=1.-ECC*COSE
      SINV=SQRT(1.-ECC*ECC)*SINE/C
      COSV=(COSE-ECC)/C
      V=ASIN(SINV)
      IF(SINV)102,100,100
  100 IF(COSV)101,104,104
  101 V=PI-V
      GO TO 104
  102 IF(COSV)103,104,104
  103 V=-PI-V
  104 IF(V)105,106,106
  105 V=TOPI+V
  106 CONTINUE
      RETURN
      END

      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE KIND(TIME,ZZ,ROA)
      COMMON ORB(300)
      COMMON/VAR1/W,OM,INC,E,MA,N,A,SW,CW,S2W,C2W,S3W,C3W,
     X  E2,EFAC,SINI,COSI,TANI
      COMMON/VAR2/NMOON,AMOON,NSUN,ASUN,FACMOON,FACSUN,AE,
     X  J2,TWOPI,RACORR,DWE,DWI,DME,DMI,DOE,DOI
      COMMON/VAR3/B,,CJ,CT,DELT,POINT,DTMAX,K1,K2,K3,GMM
      COMMON/VAR5/ALF,AP,AOR,AOR2,AOR3,ARAT,ARAT2,ARAT3,AA,A2,
     X  AB2,APB,AMP,PB2,ACPBS,ACMBC
      COMMON/VAR6/DEC,SD,CD,BB,DAO,DBI,DBO,B2,SPEC1,SPEC2,SPEC3,
     X  X,Y,Z,V,TEMP
```

```
C
      DIMENSION B(5)C(5,3),XX(10,9),ZZ(3)
      DIMENSION BT(4,2,2),CJ(3,2),CT(16,3)
      EQUIVALENCE (BT,CT(1,3))
      EQUIVALENCE(ORB(21),XX)
      REAL INC,J2,MA,N,NMOON,NSUN
C
      INTEGER POINT
C
C
C         EVALUATE THE DERIVATIVES
C
      DO 90 KINDI=1,2
      IF(KINDI.EQ.2) GO TO 27
      CALL LUNVECT(TIME,ZZ,ROA)
      ALF=ATANG(ZZ(2),ZZ(1))
      DEC=ASIN(ZZ(3))
      AP=AMOON
      FAC=FACMOON
      GO TO 30
   27 CALL SUNVECT(TIME,ZZ,ROA)
      CALL PRECESS(ZZ,ZZ,TIME,1)
      ALF=ATANG(ZZ(2),ZZ(1))
      DEC =ASIN(ZZ(3)/ROA)
      AP=ASUN
      FAC=FACSUN
   30 AOR=1./ROA
      AOR2=AOR**2
      AOR3=AOR*AOR2
      FAC=FAC*AOR3*TWOPI
      ARAT=A/AP
      ARAT2=ARAT**2
      ARAT3=ARAT*ARAT2
      SD=SIN(DEC)
      CD=COS(DEC)
      IF(K1.EQ.1) ALF=ALF-RACORR
      X=SIN(OM-ALF)
      BB=-CD*COSI*X+SD*SINI
      DAO=-CD*X
      DBI=CD*SINI*X+SD*COSI
      X=COS(OM-ALF)
      AA=CD*X
      DBO=-CD*COSI*X
      A2=AA*AA
      B2=BB*BB
      AB2=2.*AA*BB
      APB=A2+B2
      AMB=A2-B2
      APB2=APB*APB
      ACPBS=AA*CW+BB*SW
      ASMBC=AA*SW-BB*CW
C
      X=AMB*S2W-AB2*C2W
      TEMP=15./4.*E*X
```

```
      X=(4.+3.*E2)*(5.*APB-4.)
      Y=(A2-3.*B2)*A
      Z=(3.*A2-B2)*BB
      V=3.*X*ASMBC+105.*E2*(Y*S3W-Z*C3W)
      TEMP=TEMP-5./64.*AOR*ARAT*V
      X=7.*APB-6.
      V=X*(AMB*S2W-AB2*C2W)
      TEMP=TEMP+105./32.*AOR2*ARAT2*E*V
      X=21.*APB2-28.*APB+8.
      V=X*ASMBC
      TEMP=TEMP-105./128.*AOR3*ARAT3*V
      B(4)=FAC*EFAC*TEMP
C
      SPEC1=AA*DAO+BB*DBO
      SPEC2=AA*DAO-BB*DBO
      SPEC3=AA*DBO+BB*DAO
      X=2.+3.*E2
      TEMP=-3./4.*SPEC1*X-15./4. *E2*(SPEC2*C2W+SPEC3*S2W)
      X=3.*(4.+3.*E2)
C
      Y=15.*A2+5.*B2-4.
      Y=Y*DAO+5.*AB2*DBO
      Z=5.*A2+15.*B2-4.
      Z=Z*DBO+5.*AB2*DAO
      V=X*(Y*CW+Z*SW)
      Y=AMB*DAO-AB2*DBO
      Z=AMB*DBO+AB2*DAO
      V=V+105.*E2*(Y*C3W+Z*S3W)
      TEMP=TEMP+5./64.*E*AOR*ARAT*V
      X=7.*APB-4.
      Y=1.+5.*E2
      V=2.*X*SPEC1*Y
      X=AA**3*DAO- BB**3*DBO
      X=7.*X-3.*SPEC2
      Y=3.*AA**2*BB*DAO+AA**3*DBO
      Y=Y+3.*AA*BB**2*DBO+BB**3*DAO
      Y=7.*Y-6.*SPEC3
      Z=2.*X*C2W+Y*S2W
      V=V+7.*E2*Z
      TEMP=TEMP-15./32.*AOR2*ARAT2*V
      X=28.*(3.*APB-2.)
      V=X*SPEC1*ACPBS
      Y=21.*APB2-28.*APB+8.
      Z=DAO*CW+DBO*SW
      V=V+Y*Z
      TEMP=TEMP+105./128.*E*AOR3*ARAT3*V
      B(3)=FAC/EFAC/SINI*TEMP
      X=E/TANI/(1.-E2)
      B(3)=B(3)-X*B(4)
C
      X=BB*(2.+3.*E2)
      Y=E2*(BB*C2W-AA*S2W)
      TEMP=3./4.*X-15./4.*Y
      X=4.+3.*E2
```

```
      Y=5.*A2+15.*B2-4.
      V=X*(5.*AB2*CW+Y*SW)
      X=AB2*C3W-AB*S3W
      V=V-35.*E2*X
      TEMP=TEMP-15./64.*E*AOR*ARAT*V
      X=7.*APB-4.
      Y=1.+5.*E2
      V=2.*BB*X*Y
      X=2.*BB*(7.*B2-3.)
      Y=7.*A2+21.*B2-6.
      V=V-7.*E2*(X*C2W-Y*AA*S2W)
      TEMP=TEMP+15./32.*AOR2*ARAT2*V
      X=14.*AB2*(3.*APB-2.)
      Y=21.*A2*A2+126.*A2*B2
      Y=Y+105.*B2*B2-28.*A2
      Y=Y-84.*B2+8.
      V=X*CW+Y*SW
      TEMP=TEMP-105./128.*E*AOR3*ARAT3*V
      B(2)=FAC/EFAC/SINI*DBI*TEMP
C
      X=3.*APB-2.
      TEMP=3./4.*X+15./4.*(AMB*C2W+AB2*S2W)
      X=4.+9.*E2
      Y=5.*APB-4.
      V=X*Y*ACPBS
      X=A2-3.*B2
      Y=3.*A2-B2
      Z=X*AA*C3W+Y*BB*S3W
      V=V+35.*E2*Z
      TEMP=TEMP-15./64.*AOR*ARAT/E*V
      V=35.*APB2-40.*APB+8.
      X=7.*APB-6.
      V=V+7.*X*(AMB*C2W+AB2*S2W)
      TEMP=TEMP+15./32.*AOR2*ARAT2*V
      X=21.*APB2-28.*APB+8.
      V=X*ACPBS
      TEMP=TEMP-105./128./E*AOR3*ARAT3*V
      B(1)=FAC*EFAC*TEMP
      B(1)=B(1)-COSI*B(2)
C
      X=7.+3.*E2
      V=-X/4.*(3.*APB-2.)
      X=1.+E2
      TEMP=V-15./4.*X*(AMB*C2W+AB2*S2W)
      X=4.+29.*E2
      Y=5.*APB-4.
      V=X*Y*ACPBS
      X=A2-3.*B2
      X=3.*A2-B2
      Z=X*AA*C3W+Y*BB*S3W
C  CHANGE CODE ACCORDING TO KOZAI 8/19/80
      X=1.+E2
      V=V+35.*E2*Z*X
      TEMP=TEMP+15./64.*AOR*ARAT/E*V
```

```
C  CHANGE CODE ACCORDING TO KOZAI 8/19/80
      X=9.+15.*E2
      Y=35.*APB2-40.*APB+8.
      V=X*Y
C  CHANGE CODE ACCORDING TO KOZAI 8/19/80
      X=1.+3.*E2
      Y=7.*APB-6.
      Z=AMB*C2W+AB2*S2W
      V=V+35.*X*Y*Z
      TEMP=TEMP-3 '32.*AOR2*ARAT2*V
      X=21.*APB2  .*APB+8.
      V=X*ACPB8
      TEMP=TEMP+105./128.*AOR3*ARAT3/E*V
      B(5)=FAC*TEMP/TWOPI
C
C         SAVE DERIVATIVES
C
      IF(KINDI.EQ.2) GO TO 42
      DO 40 J=1,5
      C(J,POINT)=B(J)
   40 CONTINUE
      GO TO 50
   42 DO 45 J=1,5
      C(J,POINT)=C(J,POINT)+B(J)
   45 CONTINUE
C
C         GET THE SHORT-PERIOD PERTURBATIONS
C
   50 IF(DELT.NE.0.)GO TO 60
      IF(POINT.NE.3) GO TO 60
      AM=MA*TWOPI
      X=AM+2.*W
      SINM=SIN(X)
      COSM=COS(X)
      X=X+AM
      SIN2M=SIN(X)
      COS2M=COS(X)
      X=X+AM
      SIN3M=SIN(X)
      COS3M=COS(X)
      X=X+AM
      COS4M=COS(X)
      SIN4M=SIN(X)
      XTEMP1=9.*COSM-COS3M
      XTEMP2=9.*SINM-SIN3M
      Y=1.-1.5D0*APB
C
C  GET IN IN RAD/AY FOR THIS SECTION (S.P. PERTS)
      N=N*TWOPI
C
C  INDENTED LINES ARE ADDITIONAL TERMS INVOLVING FIRST ORDER
C  ECCENTRICITY DEPENDENCE DEVELOPED BY KOZAI 8/13/79
C
      TEMP=AMB*COS2M+AB2*SIN2M
```

```
C
      DA=A*1.5D0*FAC*TEMP/N
C
      TEMP=Y*COS(AM)
      X=9.*COSM+COS3M
      TEMP=TEMP+.25D0*AMB*X
      X=9.*SINM+SIN3M
      TEMP=TEMP+AB2/4.*X
      DE=FAC*TEMP/N
C
      TEMP=-Y*SIN(AM)
      X=9.*SINM-SIN3M
      TEMP=TEMP+.25D0*AMB*X
      X=9.*COSM-COS3M
      TEMP=TEMP-AB2/4. *X
      DM=FAC*TEMP/N/E
C
      X=AMB*SIN2M-AB2*COS2M
      TEMP=-21./8.*X
      X=BB*SIN2M+AA*COS2M
      TEMP=TEMP+.75D0/TANI*DBI*X
      DW=FAC*TEMP/N
C
      TEMP=DBI*X/SINI
      DOMEGA=-.75D0*FAC*TEMP/N
C
      X=AMB*COS2M+AB2*SIN2M
      TEMP=COSI*X
      TEMP=TEMP-SPEC2*SIN2M
      TEMPTEMP+SPEC3*COS2M
      DI=.75D0*FAC/SINI*TEMP/N
C
C
C  CALCULATE S.P. EFFECTS DUE TO TIDES  (KOZAI 8/13/79)
C
      AAERAT=A/AE
      FACT=(FAC/N)/(AAERAT**5)*ORB(265+KINDI)
      Y=1.-1.5D0*APBO

      X=-3.D0*Y*E*COS(AM)
      X=X-.75D0*AMB*(E*COSM-2.*COS2M-7.*E*COS3M)
      X=X-.75D0*AB2*(E*SINM-2.*SIN2M-7.*E*SIN3M)
      DAT=A*FACT*X
C
      X=-.75*Y*(2.*COS(AM)+3.*E*COS(2.*AM))
      X=X+1./16.D0*AMB*(6.*COSM-6.*E*COS2M+14.*COS3M+51.*E*COS4M)
      X=X+1./16.D0*AB2*(6.*SINM-6. *E*SIN2M+14.*SIN3M+51.*E*SIN4M)
      DET=FACT*X
C
      X=.75*Y*(2.*SIN(AM)+3.*E*SIN(2*AM))
      X=X+1./16.D0*AMB*(6.*SINM+48.*E*SIN2M-14.*SIN3M-51.*E*SIN4M)
      X=X-1./16.D0*AB2*(6.*COSM+48.*E*COS2M-14.*COS3M-51.*E*COS4M)
      DMT=(FACT/E)*X
C
```

```
      XTEMP1=3.*E*COSM-3.*COS2M-7.*E*COS3M
      XTEMP2=3.*E*SIN M-3.*SIN2M-7.*E*SIN3M
      X=-21./4.D0*Y*E*SIN(AM)
      X=X-3./8.D0*AMB*XTEMP2
      X=X+3./8.D0*AB2*XTEMP1
      X=X-1./16.D0*E*AMB*(3.*SINM-7.*SIN3M)
      X=X+1./16.D0*E*AB2*(3.*COSM-7.*COS3M)
      TEMP=-9./2.D0*BB*E*SIN(AM)
      TEMP=TEMP.25D0*BB*XTEMP2
      TEMP=TEMP-.25D0*AA*XTEMP1
      X=X+TEMP*DBI/TANI
      DMWT=FACT*X
C
      X=9./2.D0*(AA*DAO+BB*DBO)*E*SIN(AM)
      X=X-.25*AMB*COSI*XTEMP1
      X=X-.25*AB2*COSI*XTEMP2
      X=X+.25*(AA*DAO-BB*DBO)*XTEMP2
      X=X-.25*(AA*DBO+BB*DAO)*XTEMP1
      DIT=FACT*X/SINI
C
      X=9./2.D0*BB*E*SIN(AM)
      X=X+.25*BB*XTEMP2
      X=X+.25*AA*XTEMP1
      DOMEGAT=FACT*X*DBI/SINI
C
C  ADD S.P. TIDAL PERTS TO EXISTING S.P. PERTS
C
      DA=DA+DAT
      DE=DE+DET
      DM=DM+DMT
      DMW=DMW+DMWT
      DI=DI+DIT
      DOMEGA=DOMEGA+DOMEGAT
C
C
C  ADD S.P. PERTS TO EXISTING PERTS FOR NODE,INC, U, AND R
C
      XX(2,6)=XX(2,6)+DOMEGA
      XX(3,6)=XX(3,6)+DI
C
      IF(KINDI.EQ.2) GO TO 55
      CALL EVA(MA,E,SE,CE,SV,CV,EA,VEE)
   55 ROA=1.-E*CE
      DW=DMW-DM
      X=1./ROA+1./EFAC/EFAC
      X=DW+X*SV*DE
      XX(9,6)=XX(9,6)+X+EFAC/ROA/ROA*DM
      X=ROA*DA-A*CV*DE
      XX(10,6)=XX(10,6)+X+A*E*SV/EFAC*DM
C
C
C
      N=N/TWOPI
C
```

```
C
C        ADD L.P. TERMS DUE TO THE BODY TIDES.a
C
   60 ALF=ALF+ORB(267+KINDI)*TWOPI/360.D0
      X=SIN(OM-ALF)
      BB=-CD*COSI*X+SD*SINI
      DAO=-CD*X
      DBI=CD*SINI*X+SD*COSI
      X=COS(OM-ALF)
      AA=CD*X
      DBO=-CD*COSI*X
      X=AA*AA+BB*BB
      SPEC=3.*(.75D0*X-.5D0)
      X=AE/A
      X=X**5
      Y=1.-E2
      Y=Y*Y
      FF=FAC*X*ORB(265+KINDI)/Y
C
      B(4)=0.
      X=AA*DAO+BB*DBO
      B(3)=-1.5D0*FF/SINI*X
      B(2)=1.5D0*FF/SINI*BB*DBI
      B(1)=-COSI*B(2)+F*SPEC
      B(5)=FF*SPEC*EFAC/TWOPI
      Y=ORB(265+KINDI)
      IF(Y.EQ.0) Y=1
      DO 63 J=1,3
      BT(J,1,KINDI)=B(J)/Y
   63 CONTINUE
      BT(4,1,KINDI)=B(5)/Y
C
      Y=AA**2-DAO**2+BB**2
      Y=Y-BB*SINI*SD-DBO**2
      BT(3,2,KINDI)=-1.5D0*FF/SINI*Y
      Y=AA*BB*SINI+DBO*DBI
      Y=-1.5D0*FF/SINI*Y
      B T(2,2,KINDI)=Y
      BT(1,2,KINDI)=-COSI*Y-4.5D0*FF*X
      BT(4,2,KINDI)=-4.5D0*FF*X*EFAC/TWOPI
C
C  ADD BODY TIDES
      DO 65 J=1,5
      C(J,POINT)=C(J,POINT)+B(J)
  65  CONTINUE
C
   90 CONTINUE
      RETURN
      END


      COMPILER NOSTACK
      FUNCTION LOAD(I)
C  FUNCTION LOAD TAKES THE LOWER ORDER 4 BYTES FROM A REAL*8 AND PUTS
```

```
C  THEM INTO AN INTEGER*4
      INTEGER I(2)
      LOAD=I(1)
      RETURN
      END

      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE LUNARK(TT)
C
C       SPECIAL VERSION INC. S.P. INT. TERMS FOR SUN AND MOON
C
C COMPUTE LUNI-SOLAR PERTURBATIONS BY THE NEW METHOD OF
C       KOZAI (SEE SPECIAL REPORT 349).
C
      COMMON ORB(300)
      COMMON/VAR1/W,OM,INC,E,MA,N,A,SW,CW,S2W,C2W,S3W,C3W,
     X  E2,EFAC,SINI,COSI,TANI
      COMMON/VAR2/NMOON,AMOON,NSUN,ASUN,FACMOON,FACSUN,AE,
     X  J2,TWOPI,RACORR,DWE,DWI,DME,DMI,DOE,DOI
      COMMON/VAR3/B,C,CJ,CT,DELT,POINT,DTMAX,K1,K2,K3,GMM
      COMMON/VAR4/INIT
C
      DIMENSION B(5),C(5,3),TT(2),XX(10,9),ZZ(3)
      DIMENSION BT(4,2,2),CJ(3,2),CT(16,3)
      EQUIVALENCE (BT,CT(1,3))
      EQUIVALENCE(ORB(21),XX)
      REAL INC,J2,MA,N,NMOON,NSUN
      INTEGER POINT
C
      DATA INIT/0/
      DATA GMM/76298.43D0
      DATA DTMAX,K1,K2,K3/.25D0,0,0,1/
C
      NO=10
      TWOPI=CONSOC(2)
C  SET INIT TO 1 FOR ALL SUBSEQUENT OBS
C
C  EPSILON FOR NUTATION ININC TERMS (SEC OF ARC)
      EPSILON=ORB(286)
      IF(EPSILON.LE.0.OR.EPSILON.GE.1.) EPSILON=.001D0
C
C         SET CONSTANTS
C
C  USE SIDEREAL PERIOD FOR MOON
      NMOON=1./27.321661D0
      AMOON=384.4D0
      NSUN=1./365.25964D0
      ASUN=1.4959884D5
      AE=CONSOC(5)
C                   SET ARRAY POINTERS
      ORB(112)=PUT(1)
      J2=ORB(202)
C                   INITIALIZE
```

```
      EP=ORB(265)
      X=ORB(10)+ORB(11)-33282.
      RACORR=(3.506D-5)*X/57.2957795D0
      TL=EP
      TIN=TT(1)+TT(2)
      DELT=TIN-TL
      SIGN1=1.
      IF(DELT.LT.0.) SIGN1=-SIGN1
      N=XX(5,2)
      FACMOON=.0121835D0*NMOON**2/N
      FACSUN=NSUN**2/N
      TIME=TL
      DO 14 J=1,6
      XX(J,6)=0.
   14 CONTINUE
      XX( 9,6)=0.
      XX(10,6)=0.
C
C          PREPARE DT FOR THE INTEGRATION STEP
C
C  INIT=0 ON FIRST PASS THROUGH LOOP
      POINT=2
      IF(INIT.EQ.0) GO TO 25
   20 X=ABS(DELT)
      DT=X
      IF(X.GT.DTMAX) DT=DTMAX
      DT=DT*SIGN1
      DT2=DT/2.
C  RETURN FOR 2ND HALF OF INT STEP
   22 IF(POINT.EQ.3) DELT=DELT-DT
      TIME=TIME+DT2
   25 CONTINUE
      ORB(121)=(TIME-ORB(10))-ORB(11)
      ORB(122)=0.
      ORB(111)=PUT(4)
      ISAVE=LOAD(ORB(113))
      ORB(113)=PUT(1)
      CALL INST
      ORB(113)=PUT(ISAVE)
      CALL SETUP
C
      CALL KIND(TIME,ZZ,ROA)
C
C          INTEGRATE
C
      IF(INIT.EQ.1) GO TO 100
C  INTERACTION WITH J2
      X=ORB(263)
      Y=ORB(262)
      CJ(1,2)=DWE*X+DWI*Y
      CJ(2,2)=DOE*X+DOI*Y
      CJ(3,2)=DME*X+DMI*Y
      INIT=1
      GO TO 110
```

```
  100 IF(POINT.EQ.3) GOTO 104
C                                  RECALL THE EQUIVALENCE OFBT AND CT(1,3)
      DO 102 J=1,16
      CT(J,2)=CT(J,3)
  102 CONTINUE
      POINT=3
      GO TO 22
C  INT DIR EFF USING SIMPSONS RULE
  104 DO 106 J=1,5
      X=(C(J,1)+4.*C(J,2)+C(J,3))
      ORB(260+J-1)=ORB(260+J-1)+X*DT2/3.
  106 CONTINUE
C  INT INTERACTION WITH J2 BY TRAPEZ
      X=ORB(263)
      Y=ORB(262)
      CJ(1,2)=DWE*X+DWI*Y
      CJ(2,2)=DOE*X+DOI*Y
      CJ(3,2)=DME*X+DMI*Y
      DO 107 J=1,3
      K=J
      IF(J.EQ.3) K=5
      X=(CJ(J,1)+CJ(J,2))/2.
      ORB(260+K-1)=ORB(260+K-1)+X*DT
  107 CONTINUE
C  INT TIDE BY SIMPSONS RULE
      DO 108 J=1,16
      X=(CT(J,1)+4.*CT(J,2)+CT(J,3))
      K=J+10
      ORB(260+K-1)=ORB(260+K-1)+X*DT2/3.
  108 CONTINUE
C
C  SHIFT DERIVS FOR DIRECT AND PERTS
  110 DO 113 J=1,5
      C(J,1)=C(J,POINT)
      IF(J.GT.3) GO TO 113
C  DERIVS FOR J2 INTERACTION
      CJ(J,1)=CJ(J,2)
  113 CONTINUE
C  DERIVS FOR TIDE
      DO 116 J=1,16
      CT(J,1)=CT(J,3)
  116 CONTINUE
      IF(DELT.EQ.0.) GO TO 120
      POINT=2
C  RETURN FOR NEXT TIME STEP
      GO TO 20
C
  120 TL=TIN
      DO 125 J=1,5
      XX(J,6)=XX(J,6)+ORB(260+J-1)
 125 CONTINUE
      ORB(265)=TL
C
C
```

```
C
      RETURN
C
      END



      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE PRECESS (XZ,X,T,JAY)
C
C             XZ IS VECTOR REFERRED TO EQUINOX OF 1950.
C             X  IS VECTOR REFERRED TO EQUINOX OF SMITHSONIAN DAY = T
C
C             JAY=1 MEANS XZ IS INPUT VECTOR AND X IS OUTPUT
C             JAY=2 MEANS X IS INPUT VECTOR AND XZ IS OUTPUT
C   JAY=3 MEANS REPEAT THE LAST CALL TO PRECESS USING NEW
C VALUES OF COMPONENTS IN THE INPUT VECTOR.
C
      DIMENSION V(3),W(3),X(3,XZ(3)
      IF(JAY-2)1,1,2
    1 KAY=JAY
      D=(T-33281.923D0)/36524.22D0
      D2=D*D
      D3=D2*D
      XX=1.-.00029696D0*D2-.00000013D0*D3
      YX=-.02234941D0*D-.00000676D0*D2+.00000221D0*D3
      ZX=-.00971690D0*D+.00000207D0*D2+.00000096D0*D3
      YY=1.-.00024975D0*D2-.00000015D0*D3
      ZY=-.00010858D0*D2
      ZZ=1.-.00004721D0*D2
    2 IF(KAY-1)3,3,6
C                               XZ IS INPUT
    3 DO 4 J=1,3
    4 W(J)=XZ(J)
      V(1)=ZX*W(3)+YX*W(2)+XX*W(1)
      V(2)=ZY*W(3)+YY*W(2)-YX*W(1)
      V(3)=ZZ*W(3)+ZY*W(2)-ZX*W(1)
      DO 5 J=1,3
    5 X(J)=V(J)
      RETURN
C X IS INPUT
    6 DO 7 J=1,3
    7 V(J)=X(J)
      W(1)=XX*V(1)-YX*V(2)-ZX*V(3)
      W(2)=YX*V(1)+YY*V(2)+ZY*V(3)
      W(3)=ZX*V(1)+ZY*V(2)+ZZ*V(3)
      DO 8 J=1,3
    8 XZ(J)=W(J)
      RETURN
C
      END
```

```
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION PUT(I)
      COMMON/EQUI/IY
      EQUIVALENCE (IY,Y)
C  C  FUNCTION THAT ON THE VAX PUTS INTEGER*4 INTO BITS 0-31 OF
C  REAL*8 WORD. (SEE FORTRAN USER'S GUIDE PGS A1-A3)

      Y=0
      IY=I
      PUT=Y
      RETURN
      END

      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION PUT(I)
      COMMON/EQUI/IY
      EQUIVALENCE (IY,Y)
C C  FUNCTION THAT ON THE VAX PUTS INTEGER*4 INTO BITS 0-31 OF
C  REAL*8 WORD. (SEE FORTRAN USER'S GUIDE PGS A1-A3)

      Y=0
      IY=I
      PUT=Y
      RETURN
      END


      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE SETUP
C
      COMMON ORB(300)
      COMMON/VAR1/W,OM,INC,E,MA,N,A,SW,CW,S2W,C2W,S3W,C3W,
     X  E2,EFAC,SINI,COSI,TANI
        COMMON/VAR2/NMOON,AMOON,NSUN,ASUN,FACMOON,FACSUN,AE,
     X  J2,TWOPI,RACORR,DWE,DWI,DME,DMI,DOE,DOI
C
      REAL INC,J2,MA,N,NMOON,NSUN
      W=ORB(121)
      OM =ORB(122)
      INC=ORB(123)
      E  =ORB(124)
      MA =ORB(125)
      N  =ORB(126)
      A  =ORB(127)
      SW =SIN(W)
```

```
      CW =COS(W)
      S2W=2.*SW*CW
      C2W=CW*CW-SW*SW
      S3W=SW*C2W+CW*S2W
      C3W=CW*C2W-SW*S2W
      E2=E*E
      EFAC=DSQRT(1.-E2)
      SINI=SIN(INC)
      COSI=COS(INC)
      TANI=SINI/COSI
C
C         GET THE CORRECTION TERMS INVOLVING J2
C
      X=AE/A
      Y=1.-E2
      X=J2*N*TWOPI*X*X
      X=X/Y/Y
      DOE=-6.*X*E*COSI/Y
      DOI=1.5D0*X*SINI
      X=1.-5.*COSI**2
      DWE=X*DOE/(2.*COSI)
      DWI=-5.*COSI*DOI
      X=1.-3.*COSI**2
      DME=3./8.*EFAC*X*DOE/COSI
      DMI=-3.*COSI*EFAC*DOI
      DME=DME/TWOPI
      DMI=DMI/TWOPI
      RETURN
      END



      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      FUNCTION ATANG (OVER,UNDER)
C
      DIMENSION C(3)
      C(1)=CONSOC(1)/2.
      C(2)=CONSOC(1)
      C(3)=CONSOC(2)
C
      A=OVER
      B=UNDER
      D=DSQRT(A*A+B*B)
      A=A/D
      B=B/D
      IF(ABS(B)-.00000001)1,4,4
    1 D=C(1)
      IF(A)2,3,3
    2 D=D+C(2)
    3 ATANG=D
      RETURN
    4 D=DATAN(A/B)
      IF(B)2,5,5
```

```
    5 IF(A)6,3,3
    6 D=D+C(3)
      GO TO 3
C
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE LUNVECT (TIME,VECTOR,ROVERA)
C
C         GETS DIRECTION COSINES OF THE MOON WITH RESPECT TO THE
C EQUINOX AND EQUATOR OF THE EPOCH TIME (EXPRESSED IN
C         SMITHSONIAN DAYS).
C                                          R.E.BRIGGS -- MAY 1973
C
C
C  REVISED MAR 77 TO COMPUTE POSITION FROM BROWNS THEORY
C
      DIMENSION VECTOR(3),XX(3)
      REAL LONGS,MS
      REAL LONG,M
      SPR=206264.806
C
      RADIAN = CONSOC(7)
      REVOL=360.
      T=(TIME-15019.5)/36525.
      T2=T*T
C
      X=1336.85523095*T
      J=X
      Y=J
      X=(X-Y)*REVOL
      Y=270.434164+X-.001133*T2
      LONG=Y/RADIAN
C
      X=11.30287231*T
      J=X
      Y=J
      X=(X-Y)*REVOL
      Y=334.329556+X-.010325*T2
      GAMMA=Y/RADIAN
C
       X=5.37261669*T
      J=X
      Y=J
      X=(X-Y)*REVOL
      Y=259.183275-X+.02078*T2
      OMEGA=Y/RADIAN
C
C                   COMPUTE POSITION FROM BROWNS THEORY.
C
      M=LONG-GAMMA
      X=129602768.13*T+1.089*T2
      X=279.69668+X/3600.
      LONGS=X/RADIAN
```

```
      X=129596579.10*T-0.54*T2
      X=358.47583+X/3600.
      MS=X/RADIAN
      FF=LONG-OMEGA
      DD=LONG-LONGS
C LONGITUDE FIRST
      DLONG=2369.912*SIN(2.*DD)+191.953*SIN(M+2.*DD)
      DLONG=DLONG+22639.500*SIN(M)-4586.465*SIN(M-2.*DD)
      DLONG=DLONG-38.428*SIN(M-4.*DD)-668.146*SIN(MS)
      DLONG=DLONG-165.145*SIN(MS-2.*DD)-125.154*SIN(DD)
      DLONG=DLONG+769.016*SIN(2.*M)-211.656*SIN(2.*M-2.*DD)
      DLONG=DLONG-30.773*SIN(2.*M-4.*DD)-109.673*SIN(M+MS)
      DLONG=DLONG-205.962*SIN(M+MS-2.*DD)+147.687*SIN(M-MS)
      DLONG=DLONG-411.608*SIN(2.*FF)-55.173*SIN(2.*FF-2.*DD)
      DLONG=DLONG+36.124*SIN(3.*M)-45.099*SIN(M+2.*FF)
      DLONG=DLONG+39.528*SIN(M-2.*FF)
      DLONG=DLONG-24.420*SIN(MS+2.*DD)+14.387*SIN(2.*M+2.*DD)
      DLONG=DLONG+14.577*SIN(M-MS+2.*DD)+28.475*SIN(M-MS-2.*DD)
      DLONG=DLONG+18.609*SIN(M-DD)+18.023*SIN(MS+DD)
      DLONG=DLONG-13.193*SIN(3.*M-2.*DD)
      DLONG=DLONG/SPR
      LONG=LONG+DLONG
C                              NOW THE LATITUDE
      XLAT=117.2608*SIN(FF+2.*DD)+18461.3493*SIN(FF)
      XLAT=XLAT-623.6553*SIN(F F-2.*DD)+1010.1724*SIN(M+FF)
      XLAT=XLAT-166.5729*SIN(M+FF-2.*DD)+199.4806*SIN(-M+FF+2.*DD)
      XLAT=XLAT-999.6848*SIN(-M+FF)-33.3628*SIN(-M+FF-2.*DD)
      XLAT=XLAT-29.6546*SIN(MS+FF-2.*DD)+61.9131*SIN(2.*M+FF)
      XLAT=XLAT-31.7627*SIN(-2.*M+FF)
      XLAT=XLAT+15.1194*SIN(M+FF+2.*DD)+12.1245*SIN(-MS+FF-2.*DD)
      XLAT=XLAT-15.5659*SIN(2.*M+FF-2.*DD)
      XLAT=XLAT/SPR
C                              THE PARALLAX
      DP=28.2333*COS(2.*DD)+3.0861*COS(M+2.*DD)
      DP=DP+186.5398*COS(M)+34.3117*COS(M-2.*DD)
      DP=DP+1.9178*COS(MS-2.*DD)+10.1657*COS(2.*M)
      DP=DP+1.4437*COS(M+MS-2.*DD)+1.1528*COS(M-MS)
      DP=DP-0.9781*COS(DD)-0.9490*COS(M+MS)
      DP=DP+0.6215*COS(3.*M)-0.7136*COS(M-2.*FF)
C
      X=COS(XLAT)
      XX(1)=X*COS(LONG)
      XX(2)=X*SIN(LONG)
      XX(3)=SIN(XLAT)
      X=1.+DP/3422.452
      ROVERA=1./X
C^^^^^^^^^^
C                  DIR. COSINES N THE EQUATORIAL SYSTEM
C
      EPS=(23.452294-.0130125*T)/RADIAN
      SE=SIN(EPS)
      CE=COS(EPS)
      VECTOR(1)=XX(1)
      VECTOR(2)=XX(2)*CE-XX(3)*SE
```

```
      VECTOR(3)=XX(3)*CE+XX(2)*SE
      RETURN
C
      END
      COMPILER DOUBLE PRECISION
      COMPILER NOSTACK
      SUBROUTINE SUNVECT (T,X,R)
C
C     T= TIME IN SMITHSONIAN DAYS
C              X=VECTOR TO SUN REFERRED TO EQUINOX AND EQUATOR OF 1950.0
C              R=MAGNITUDE OF X VECTOR    (LENGTHS ARE IN A.U.)
C
      DIMENSION X(3)
C
      A=6.2291402+.0172021242*(T-39125.)
      E=A
      DO 1 J=1,6
    1 E=A+.016728643*SIN(E)
      SE=SIN(E)
      CE=COS(E)
      R=1.0000002-.016728643*CE
      X(1)=.20947919*CE+.97767652*SE-.003504301
      X(2)=-.89708519*CE+.19215772*SE+.015007012
      X(3)=.43367832*X(2)
      RETURN
      END

      COMPILER DOUBLE PRECISION
      SUBROUTINE EPHLUN(T,ELE,PERT,RNU)
C
      DIMENSION T(2),ELE(6),PERT(6),RNU(2)
C
C     VERSION TO INTERPOLATE GRIPE LUNAR PERTURBATION TABLE
C     INCORPORATED IN MARCH 26, 1981 BY SAEQA DIL
C
      COMMON/MOON/PM(8,12)
C
      J1=1
C
      TEMP=T(1)+T(2)
C
      IF(TEMP.LT.PM(1,1)) STOP TIME LESS THAN TABLE
      IF(TEMP.GT.PM(1,12)) STOP TIME GREATER THAN TABLE
C
  100 CONTINUE
C
      IF(TEMP.GE.PM(1,J1).AND.TEMP.LE.PM(1,J1+1)) GO TO 200
      IF(TEMP.GT.PM(1,J1+1) J1=J1+1
C
      GO TO 100
C
  200 CONTINUE
C
      DO 300 I=1,5
```

```
      TEMP1=(PM(I+1,J1+1)-PM(I+1,J1))/(PM(1,J1+1)-PM(1,J1))
      TEMP2=TEMP1*(TEMP-PM(1,J1)
      PERT(I)=TEMP2+PM(I+1,J1)
  300 CONTINUE
      DO 400 I=1,2
      TEMP1=(PM(I+6,J1+1)-PM(I+6,J1))/(PM(1,J1+1)-PM(1,J1))
      TEMP2=TEMP1*(TEMP-PM(1,J1))
      RNU(I)=TEMP2+PM(I+6,J1)
  400 CONTINUE
      RETURN
      END
```

The M.I.T. Lunar and Planetary Ephemeris

by

P. J. Morgan and R. W. King
Department of Earth and Planetary Sciences
Massachusetts Institute of Technology
Cambridge, MA 02139 USA

1 Introduction

This document describes a software READ package for the export version of an M.I.T. planetary ephemeris tape. The tape itself contains as time-series the heliocentric orbits of the nine major planets, the geocentric orbit of the moon, the earth's nutation, and the moon's physical libration. Lunar and planetary coordinates are cartesian and are presently referenced to the 1950.0 mean equator and equinox. The tabular interval is one-half day for the moon, one day for Mercury, and two days for each of the remaining planets. Conventional nutation and libration angles are tabulated at half- day intervals.

The ephemeris tape is available at 800, 1600, or 6250 BPI on 9-track tape using the IBM EBCDIC character set. Records are of fixed length, 80 characters, and blocked using a factor of 100. There are two principal record groups on the tape. The first group of 61 records contains the 128 character header and 59 other records specifying the parameters used to calculate the ephemerides. The second group of record contains the positions and velocities of the bodies, and the nutation and libration angles, in block of 20 days.

Appendices A through L contain documentation and listings of the subroutines required to interpolate for any epoch the heliocentric coordinates (position, velocity, and acceleration) of the planets and the solar system barycenter, the geocentric coordinates of the moon, the two angles of the earth's nutation, and the three angles of the moon's physical libration. The subroutines are written in IBM FORTRAN, but are easily translatable for use on other machines. Appendix M lists an example of the first three data blocks from an ephemeris tape, and Appendix N gives sample output obtained using the READ package with this example.

2 APPENDIX A - SUBROUTINE PRTCRD

```
      SUBROUTINE PRTCRD (IFILE,JD,FRACT,LER,KISS,XPERT,XCNT,DPSI,
     1DEPS,LIBRAT)
C
C     MAIN ROUTINE TO COMPUTE POSITION,VELOCITY AND ACCELERATION OF
C     PLANETS AND SOLAR SYSTEM BARYCENTER RELATIVE TO CENTER
C     OF MASS OF SUN, MOON RELATIVE TO EARTH, THE EARTH'S NUTATION
C     AND THE MOON'S PHYSICAL LIBRATION.
C
C     LOGIC FOLLOWS THAT USED IN PEP AS WRITTEN BY M.ASH.
C     NO COMMON IS USED. THIS MAKES FOR EXTENSIVE CALLING SEQUENCES.
C
C     THE FIXED FORMAT MIT PLANETARY EPHEMERIS CONSISTS OF 3 TYPES OF
C     RECORD BLOCKS. BLOCK ONE HAS TWO RECORDS OF CHARACTERS
C     DESCRIBING THE EPHEMERIS IN ENGLISH OR AUSTRALIAN. BLOCK TWO
C     HAS 48 RECORDS DESCRIBING FIXED PARAMETERS & CONSTANTS ASSOCIATED
C     WITH THE EPHEMERIS. FOLLOWING BLOCKS 1 AND 2 THERE ARE N TYPE 3
C     BLOCKS EACH CONSISTING OF 261 RECORDS TABULATING FOR A 20-DAY
C     INTERVAL BODY POSITIONS AND BODY VELOCITIES, FOLLOWED BY
C     NUTATION AND LUNAR PHYSICAL LIBRATION ANGLES.  THE FORMAT OF
C     THE TAPE IS DESCRIBED IN MORE DETAIL IN THE DOCUMENTATION OF
C     THE NBODY SOURCE PROGRAM (NOT INCLUDED HERE).
C
C     NOTE (1) THIS ROUTINE SHOULD BE CALLED AS FOLLOWS:
C              THE FIRST CALL SHOULD BE WITH JD=0. THIS LOADS
C              FROM THE EPHEMERIS TAPE PARAMETERS AND
C              CONSTANTS FROM BLOCKS 1 & 2. IT ALSO LOADS SOME
C              ARRAYS WITH INITIAL VALUES AND THE LIMITS OF
C              APPLICABILITY OF THE DATA SET. ALL SUBSEQUENT CALLS
C              SHOULD BE MADE WITH JD .GT. 0.
C
C          (2) THIS CODE IS SET UP TO PERFORM NUMERICAL DIFFERENTIATION
C              TO OBTAIN FIRST AND SECOND DERIVATIVES WHEN REQUIRED.
C              CODE IS NOT IMPLEMENTED TO MAKE USE OF VELOCITY
C              INFORMATION ON THE NBODY TAPE. THESE DIFFERENCES ARE VERY
C              SMALL. THE FIRST DERIVATIVE FOR THE NUTATION TERMS IS
C              ALWAYS COMPUTED IF NUTATIONS ARE REQUESTED. THE FIRST
C              DERIVATIVE OF THE LIBRATIONS IS NEVER COMPUTED ALTHOUGH
C              THE ARRAYS AND LOGIC ARE CAPABLE OF SUPPORTING THIS
C              FEATURE.
C
C          (3) NOTE CAREFULLY THE DEFINITIONS OF JD AND FRACT, WHICH
C              SPECIFY THE EPOCH FOR WHICH EPHEMERIS INFORMATION
C              IS REQUIRED. FRACT IN PEP IS RECKONED FROM THE
C              MIDNIGHT PRECEDING THE BEGINNING (AT NOON) OF THE
C              JULIAN DAY JD.  THUS, PEP'S JD+FRACT IS ONE HALF DAY
C              GREATER THAN THE CONVENTIONAL JULIAN DATE. THE EPOCH
C              SHOULD BE SPECIFIED IN COORDINATE TIME (CT).
C
C          (4) UNITS: POSITIONS- AU
C                     VELOCITIES- AU/DAY
```

```
C                    ACCELERATIONS- AU/DAY**2
C                    ANGLES- RADIANS
C                    ANGULAR RATES- RADIANS/DAY
C
C
C     INPUT PARAMETERS ARE:
C
C  IFILE        :I*4  LOGICAL UNIT NUMBER OF EPHEMERIS TAPE/FILE
C  JD           :I*4  JULIAN DAY BEGINNING AT NOON OF DAY ON WHICH
C                     VECTOR IS REQUIRED.
C
C  FRACT        :R*8  FRACTION OF DAY FROM MIDNIGHT FOR WHICH
C                     VECTOR IS REQUIRED.
C  LER          :I*4  LER=0  POSITIONS ONLY ARE REQUIRED.
C                     LER=1  POSITIONS AND VELOCITIES ARE REQUIRED.
C                     LER=2  POSITIONS VELOCITIES AND ACCELERATIONS
C                     ARE REQUIRED.
C  KISS(13)     :I*4  KISS(I) .LT. 0  DO NOT COMPUTE VECTORS FOR BODY I.
C                     KISS(I) .GT. 0  COMPUTE VECTORS FOR THIS BODY.
C                     SPECIAL MEANING IS ATTACHED TO KISS(11) - KISS(13)
C                     WHICH OPERATE ANALOGOUSLY TO KISS(1) THROUGH KISS(10):
C                     KISS(11) IS FOR NUTATIONS.
C                     KISS(12) IS FOR LUNAR LIBRATIONS.
C                     KISS(13) IS FOR SOLAR SYSTEM BARYCENTER.
C
C     OUTPUT PARAMETERS ARE:
C
C  XPERT(9,10) :R*8  POSITION(1-3),VELOCITY(4-6),&  ACCELERATION(7-9) OF
C                     PLANETARY BODIES: MERCURY=1, VENUS=2, EARTH/MOON
C                     BARYCENTER=3, MARS=4, JUPITER=5, SATURN=6,
C                     URANUS=8, PLUTO=9, MOON=10.  ALL POSITIONS  ARE
C                     HELIOCENTRIC EXCEPT FOR THE MOON, WHICH IS
C                     GEOCENTRIC.
C  XCNT(9)      :R*8  SOLAR SYSTEM BARYCENTER RELATIVE TO SUN
C  DPSI(2)      :R*4  NUTATION IN LONGITUDE AND FIRST DERIVATIVE
C  DEPS(2)      :R*4  NUTATION IN OBLIQUITY AND FIRST DERIVATIVE
C  LIBRAT(2,3) :R*4  LUNAR PHYSICAL LIBRATIONS: TAU, THETA (=I+RHO),
C                     SIGMA, AND FIRST DERIVATIVE.
C
C     REQUIRED SUBROUTINES ARE:
C
C  READF  TO READ BLOCK ONE OF FIXED FORMAT NBODY TAPE
C  READG  TO READ BLOCK TWO OF FIXED FORMAT NBODY TAPE.
C  READH  TO READ FIRST RECORD OF A TYPE 3 BLOCK OF RECORDS
C         FROM FIXED FORMAT NBODY TAPE (EPHEMERIS TAPE)
C  READI  TO READ RECORDS 2 THROUGH 261 OF A TYPE 3 BLOCK OF RECORDS ON
C         FIXED FORMAT NBODY TAPE.
C  LVECTR COMPUTES DO LOOP LIMITS FOR SHIFTING STORAGE AND READING
C         EPHEMERIS DATA INTO ARRAYS FROM EPHEMERIS TAPE.
C  SKIPF  TO SKIP FORWARDS OVER A BLOCK OF TYPE 3 RECORDS.
C  SKIPB  TO SKIP BACKWARDS OVER A BLOCK OF TYPE 3 RECORDS.
C  YPRTCD COMPUTES EVERETT INTERPOLATION Y-VECTORS USING EIGHTH
C         DIFFERENCES.
C  YNUTLB COMPUTES EVERETT INTERPOLATION Y-VECTORS FOR NUTATION AND
```

```
C          LIBRATION USING FOURTH DIFFERENCES.
C  PRTERP PERFORMS EVERETT EIGHTH DIFFERENCE INTERPOLATION FOR THE
C          MOON AND PLANETS.
C  NUTERP PERFORMS EVERETT FOURTH DIFFERENCE INTERPOLATION FOR
C          NUTATIONS AND LIBRATIONS.
C
C      PETER MORGAN SEPT 81
C
C      DECLARATIONS:
C
       INTEGER *4 NBDY,NPL(10),NCP(10),INTB(10),INTBD,KBDY(40)
C
       INTEGER *4 IFILE,JD,LER,JDBD1,JDBD2,JDBD0(10),NMOON,JVLBD,ITRT,NPG
      1,LV(7),JDBD(3),IVEL(3),MVEL(3),ITEMP1,ITEMP2,ITEMP3,IVL,MVL,NTAB,
      2KISS(13)
C
       REAL *4 NAME(6,12),EPSBD,PSID(120),EPSD(120),LIBRT(120,3)
      1,DPSI(2),DEPS(2),LIBRAT(2,3)
C
       REAL *8 FRACT,XPERT(9,10),XCNT(9),
      1BETA(6,10),MASS1(10),FRCT(3),MERC(6,30),BODY(6,15,
      28),MON(6,120),DTEMP1,P(4),DMOON(2),DMERC(2),DBODY(2),
      3YMOON(5,9,2),YMERC(5,9,2),YBODY(5,9,2,8),YEPS(3,2,2),YPSI(3,2,2),
      4YLIB(3,2,2,3),FRACT1,DIR,MEQINC,MASCNT
C
C      END DECLARATIONS SECTION
C
       IF(JD.NE.0) GO TO 80
       REWIND IFILE
C      READ HEADER RECORDS (BLOCK 1) OF EPHEMERIS TAPE
       CALL READF (IFILE)
C      READ TYPE 2 DATA BLOCK. THIS INFO CONSTANT FOR NBODY TAPE.
       CALL READG (IFILE,NBDY,NPL,NCP,INTB,JDBD1,JDBD2,JDBD0,BETA,MEQINC,
      1NAME,NMOON,NBDY1,INTBD,JVLBD,MASS1)
C      TEST FOR TAPE TO BE AU UNITS FOR MOON
       IF(NMOON.EQ.0) GO TO 20
       WRITE(6,10) NMOON
   10  FORMAT(/,1X,'THIS ROUTINE SETUP FOR DISTANCES TO MOON IN AU ONLY:
      1VALUE ON NBODY TAPE IS',I3,/,' PROGRAM TERMINATING-FATAL ERROR')
       STOP
C
C      SUM UP SOLAR SYSTEM MASSES FOR COMPUTATION OF SOLAR SYSTEM
C      BARYCENTER. NOTE MASS OF SUN=1.D0 MASS UNITS. COMPUTATIONAL
C      ERRORS WILL RESULT IF ALL BODIES NOT PRESENT.
   20  MASCNT=1.D0
       ITRIG=1
       ITRIG2=0
       DO 30 I=1,10
   30  ITRIG2=ITRIG2+KISS(I)
       IF(ITRIG2 .NE. 10 .OR. NBDY .NE. 10) GO TO 50
       DO 40 I=1,NBDY
   40  MASCNT=MASCNT+MASS1(I)
       GO TO 60
   50  ITRIG=0
```

```
C
C     FILL STORAGE WITH FIRST THREE BLOCKS OF TYPE 3 RECORD BLOCKS.
C
   60 NN=1
      DO 70 L=NN,3
      CALL LVECTR(L,LV)
      CALL READH(IFILE,L,JDBD,FRCT,IVL,MVL)
      CALL READI(IFILE,LV,IVL,MVL,NBDY,MERC,BODY,MON,PSID,EPSD,LIBRT)
   70 CONTINUE
      IF(JD.EQ.0) RETURN
   80 CONTINUE
C
C     FIRST TEST THAT THE REQUESTED EPOCH IS ON THE NBODY TAPE.
      IF(JD.GE.JDBD1+20 .AND. JD.LT.JDBD2-20) GO TO 100
      WRITE(6,90) JD,JDBD1,JDBD2
   90 FORMAT(/,1X,'PROGRAM TERMINATING DUE TO OUT OF RANGE JULIAN DATES'
     1,/,5X,'REQUESTED EVALUATION JD IS',I10,/,5X,'STARTING JD OF TAPE
     2IS',I10,' END JD OF TAPE IS ',I10)
      STOP
C
C     ORDINARILY THE EPHEMERIS TAPE IS INCREASING IN TIME, THE FORWARD
C     DIRECTION. INTEGRATION AND HENCE TAPE CAN BE  BACKWARDS IN
C     TIME. PARAMETER IDIR=1 FOR FORWARD-NORMAL MODE OR -1 FOR
C     BACKWARD  MODE. THIS ROUTINE WORKS FOR BOTH DIRECTIONS DEPENDING
C     ON THE SETTING OF PARAMETER IDIR WHICH IS AUTOMATIC FROM
C     JDBD1 AND JDBD2.
C
  100 IDIR=ISIGN(1,JDBD2-JDBD1)
      DIR=DFLOAT(IDIR)
C
C     RESET ITRIG USED TO TEST SOLAR SYSTEM BARYCENTER COMPUTATION
C     SINCE THIS COMPUTATION DEPENDENT ON KISS ARRAY.
      ITRIG=0
C
C     A BODY'S STATE IS GIVEN BY A VECTOR OF COORDINATES; ONE VECTOR OF
C     3 COORDINATES FOR POSITION, VELOCITY, AND ACCELERATION. INPUT
C     PARAMETER LER SPECIFIES THIS STATE.
C     PARAMETERS LIMVEL,LIMNUT & LIMLIB CONTROL THE 1ST & 2ND
C     DERIVATIVES OF A BODY POSITION & 1ST DERIVATIVE OF NUTATION AND
C     LIBRATION.
      LIMVEL=3*(LER+1)
      LIMNUT=IABS(KISS(11))
      IF(KISS(11).GT.0) LIMNUT=LIMNUT+1
      LIMLIB=IABS(KISS(12))
      IF(KISS(12).GT.0)LIMLIB=LIMLIB+1
C
C     THEORETICALLY IT IS POSSIBLE TO GENERATE PEP TAPES WITH TABULATION
C     INTERVALS OTHER THAN MERCURY=2 DAYS, VENUS THROUGH PLUTO=4 DAYS AND
C     EARTH'S MOON =0.5 DAY. SUCH A TAPE WILL REQUIRE STORAGE ALLOCATION
C     TO BE MODIFIED AND/OR TAPE FORMATS. THIS SUITE OF ROUTINES ASSUMES
C     THE ABOVE STANDARD CONDITIONS. NEVERTHELESS FOR STRANGE AUSTRALIAN
C     AMERICAN REASONS IT IS NECESSARY TO DEFINE THESE VALUES
C     EXPLICITLY. THE PRINCIPAL REASON IS TO COPE WITH FORWARD AND
C     BACKWARD TAPES.
```

```
C
C     DMOON(1), DMERC(1) & DBODY(1) CONTAIN TABULAR VALUE WHILE
C     DMOON(2), DMERC(2) & DBODY(2) CONTAIN SQUARE OF TABULAR INTERVAL.
C     LATTER REQUIRED BY EVERETT INTERPOLATOR ROUTINE PRTERP.
C
      DMOON(1)=0.5D0*DIR
      DMOON(2)=DMOON(1)**2
      DMERC(1)=2.0D0*DIR
      DMERC(2)=DMERC(1)**2
      DBODY(1)=4.0D0*DIR
      DBODY(2)=DBODY(1)**2
      NN=IDIR*(JD-JDBD(2))
C
C     THREE WAY BRANCH: NN .LT. 0 CORRECT RECORDS ARE BEHIND ON TAPE.
C                       NN=0 CORRECTLY POSITIONED
C                       NN .GT. 0 CORRECT RECORDS ARE AHEAD ON TAPE.
C
      IF(NN)250,300,110
C
C     CORRECT RECORDS ARE AHEAD ON TAPE.
C     HOW MANY RECORDS AHEAD ON TAPE?
C
  110 N=NN/20
      IF(N.EQ.0) GO TO 300
      MM=N-3
      IF(MM)140,270,120
  120 DO 130 I=1,MM
      CALL SKIPF(IFILE,ITEMP1,DTEMP1,ITEMP2,ITEMP3)
  130 CONTINUE
      GO TO 270
C
C     CORRECT MIDDLE RECORD IS NO MORE THAN 2 RECORD BLOCKS AHEAD ON
C     TAPE
C
C     ALL STORAGE MUST BE SHIFTED
C
C     THE L VECTOR RESULTS FROM THE DIFFERENT TABULATION INTERVALS
C     FOR THE BODIES. L1 & L2 ARE FOR MERCURY WHICH HAS 10 COLUMNS
C     L3 & L4 ARE FOR VENUS THROUGH PLUTO WHICH HAVE 5 COLUMNS WHILE
C     L5 & L6 ARE FOR EARTH'S MOON WHICH HAS 40 COLUMNS.
  140 L1 = 1
      L2 =10
      L3 = 1
      L4 = 5
      L5 = 1
      L6 =40
      M1 =N*10
      M2 =N*5
      M3 =N*40
      NN =2
      JDBD(1) =JDBD(N+1)
      FRCT(1)=FRCT(N+1)
      IVEL(1)=IVEL(N+1)
      MVEL(1)=MVEL(N+1)
```

```
      IVL    =IVEL(1)
      MVL    =MVEL(1)
C     START SHIFT LOOP FOR MERCURY THROUGH PLUTO SINCE FIRST VARIABLE IS
C     COMMON
  150 DO 190 I=1,IVL
C     SHIFT STORAGE FOR MERCURY
      DO 160 J=L1,L2
      MM  =J+M1
      MERC(I,J)  =MERC(I,MM)
  160 CONTINUE
      DO 180 J=L3,L4
      MM  =J+M2
      DO 170 K=1,8
      BODY(I,J,K)=BODY(I,MM,K)
  170 CONTINUE
  180 CONTINUE
  190 CONTINUE
C     SHIFT STORAGE FOR EARTH'S MOON
      DO 210 J=L5,L6
      MM  =J+M3
      DO 200 I=1,MVL
      MON(I,J)  =MON(I,MM)
  200 CONTINUE
  210 CONTINUE
C     SHIFT STORAGE FOR NUTATION COEFFICIENTS
C     THIS IS ALSO THE OUTER LOOP FOR LIBRATION COEFFICIENTS
      DO 230 J=L5,L6
      MM  =J+M3
      PSID(J) =PSID(MM)
      EPSD(J) =EPSD(MM)
C     SHIFT STORAGE (INNER LOOP) FOR LIBRATION COEFFICIENTS.
      DO 220 I=1,3
      LIBRT (J,I) = LIBRT (MM,I)
  220 CONTINUE
  230 CONTINUE
      GO TO(240,280),N
C     SHIFT POINTERS TO NEW CENTER RECORD FOR SUCCESSFUL INTERPOLATION
C
  240 JDBD(2) =JDBD(3)
      FRCT(2)=FRCT(3)
      IVEL(2)=IVEL(3)
      MVEL(2)=MVEL(3)
      IVL    =IVEL(2)
      MVL    =MVEL(2)
      N  = 2
      NN = 3
      L1 =11
      L2 =20
      L3 = 6
      L4 =10
      L5 =41
      L6 =80
      GO TO 150
C
```

```
C           CORRECT RECORDS ARE BEHIND ON THE TAPE
  250 NN =4-(NN+1)/20
      DO 260 I=1,NN
      CALL SKIPB(IFILE,ITEMP1,DTEMP1,ITEMP2,ITEMP3)
  260 CONTINUE
C
C           READ PERTURBING PLANET TAPE
  270 NN=1
  280 DO 290 L=NN,3
      CALL LVECTR(L,LV)
      CALL READH(IFILE,L,JDBD,FRCT,IVL,MVL)
      CALL READI(IFILE,LV,IVL,MVL,NBDY,MERC,BODY,MON,PSID,EPSD,LIBRT)
      IVEL(L)=IVL
      MVEL(L)=MVL
  290 CONTINUE
      NTMOON=-99
      NTBODY=-99
      NTMERC=-99
C     SETUP FOR MOON,NUTATION & LIBRATION INTERPOLATION NOW
      NN=(JD-JDBD(2))*IDIR
  300 FRACT1=FRACT*DIR
C     REMEMBER KISS: KISS(I)=-1 DON'T COMPUTE INTERPOLATED POSITION.
C                    KISS(I)= 1 COMPUTE INTERPOLATED POSITION.
C                    KISS(11)   FOLLOWS SAME RULES FOR NUTATION.
C                    KISS(12)   FOLLOWS SAME RULES FOR LIBRATION.
C                    KISS(13)   FOLLOWS SAME RULES FOR SOLAR SYSTEM
C                               BARYCENTER.
C     HENCE KISS(10) .GT. 0 MEANS COMPUTE MOON VECTOR.
C           KISS(11) .GT. 0 MEANS COMPUTE NUTATION.
C           KISS(12) .LT. 0 DO NOT COMPUTE LIBRATION.
C           KISS(13) .GT. 0 MEANS COMPUTE SOLAR SYSTEM BARYCENTER
C                           NOTE KISS(1) THROUGH KISS(10) MUST BE
C                           GREATER THAN ZERO FOR THIS CONDITION.
C
      IF(KISS(10).GT.0) GO TO 310
      IF(KISS(11).GT.0) GO TO 310
      IF(KISS(12).LE.0) GO TO 530
  310 NTAB=2*NN+1
      P(1) =FRACT-0.5D0
      IF(P(1) .LT. 0.0D0) GO TO 320
      NTAB =NTAB+IDIR
      GO TO 330
  320 P(1) =FRACT
  330 P(1) =P(1)/DMOON(1)
      N    =3
C
C           COMPUTE P VECTOR AND TABULAR INDICES
  340 IF(P(1) .GE. 0.0D0) GO TO 350
      P(1) =P(1)+1.0D0
      NTAB =NTAB-1
  350 MTAB =NTAB+1
      P(3) =1.0D0-P(1)
      P(2) =P(1)**2
      P(4)  =P(3)**2
```

```
      GO TO (660,540,360),N
C
C           REST OF SETUP FOR MOON, NUTATION, LIBRATION INTERPOLATION
  360 N    =10
      IF (NTAB.EQ.NTMOON) GO TO 490
      IF (MTAB.EQ.NTMOON) GO TO 380
      IF (NTAB.EQ.(NTMOON+1)) GO TO 370
      K1 = NTAB + 36
      K2 = 2
      K3 = 1
      GO TO 450
  370 K1 = MTAB + 36
      K3 = 2
      GO TO 390
  380 K1 = NTAB  + 36
      K3 = 1
  390 K2 = 1
      K4 = 3 - K3
C
C           SHIFT Y-VECTORS FOR MOON, NUTATION LIBRATION INTERPOLATION
       IF(KISS(10).LT.0) GO TO 410
      LIMVL =LIMVEL
      DO 400 I=1,5
      DO 400 J=1,LIMVL
  400 YMOON(I,J,K4) = YMOON(I,J,K3)
  410 IF(KISS(11).LE.0) GO TO 430
      DO 420 I=1,3
      DO 420 J=1,LIMNUT
      YEPS(I,J,K4)=YEPS(I,J,K3)
      YPSI(I,J,K4)=YPSI(I,J,K3)
  420 CONTINUE
  430 IF(KISS(12) .LE.0) GO TO 450
      DO 440 I=1,3
      DO 440 J=1,LIMLIB
      DO 440 K=1,3
      YLIB(I,J,K4,K)=YLIB(I,J,K3,K)
  440 CONTINUE
C
C           CALCULATE Y-VECTORS FOR MOON NUTATION LIBRATION INTERP.
  450 NTMOON = NTAB
      IF(KISS(10).LT.0) GO TO 460
      CALL YPRTCD(LER,MON(1,K1),YMOON(1,1,K3),K2)
  460 IF(KISS(11) .LE. 0) GO TO 470
      CALL YNUTLB (PSID(K1),YPSI(1,1,K3),K2,KISS(11))
      CALL YNUTLB (EPSD(K1),YEPS(1,1,K3),K2,KISS(11))
  470 IF(KISS(12) .LE.0) GO TO 490
      DO 480 I=1,3
      CALL YNUTLB(LIBRT(K1,I),YLIB(1,1,K3,I),K2,KISS(12))
  480 CONTINUE
C
C           PERFORM MOON, NUTATION, LIBRATION INTERPOLATIONS
  490  IF(KISS(10) .LT. 0) GO TO 500
       CALL PRTERP(P,LER,N,XPERT,DMOON,YMOON)
  500 IF(KISS(11) .LE. 0) GO TO 510
```

```
      CALL NUTERP(P,YPSI,DPSI,DMOON,KISS(11))
      CALL NUTERP(P,YEPS,DEPS,DMOON,KISS(11))
  510 IF(KISS(12) .LE. 0) GO TO 530
      DO 520 I=1,3
      CALL NUTERP(P,YLIB(1,1,1,I),LIBRAT(1,I),DMOON,0)
  520 CONTINUE
C
C     TRANSFORM THE LIBRATION ANGLES FROM THE TAU, RHO, I*(TAU-SIGMA)
C     SYSTEM TO THE TAU, THETA, SIGMA SYSTEM.
C     NOTE THAT FIRST DERIVATIVES ARE NOT COMPUTED AND ARE THEREFORE
C     ZERO.
      LIBRAT(1,2)=LIBRAT(1,2)+MEQINC
      LIBRAT(1,3)=LIBRAT(1,1)-LIBRAT(1,3)/MEQINC
C
C           SET UP VENUS THROUGH PLUTO INTERPOLATION
  530 NTAB =NN/4
      N    =NN-4*NTAB
      P(1) =N
      P(1) =(P(1)+FRACT1)/DBODY(1)
      NTAB =NTAB+1
      N    =2
      GO TO 340
  540 NGO = 3
      IF (NTAB.EQ.NTBODY) GO TO 600
      IF (MTAB.EQ.NTBODY) GO TO 560
      IF (NTAB.EQ.(NTBODY+1)) GO TO 550
      NGO = 1
      K1 = NTAB + 1
      K2 = 2
      K3 = 1
      GO TO 580
  550 K1 = MTAB + 1
      K3 = 2
      GO TO 570
  560 K1 = NTAB + 1
      K3 = 1
  570 K2 = 1
      NGO = 2
  580 NTBODY = NTAB
      GO TO 600
C
C
C           PERFORM VENUS THROUGH PLUTO INTERPOLATIONS
  590 N    =N+1
  600 IF(KISS(N) .LT. 0) GO TO 650
      K5 = N-1
      GO TO (630,610,640),NGO
  610 K4 = 3 - K3
      LIMVL =LIMVEL
      DO 620 I=1,5
      DO 620 J=1,LIMVL
  620 YBODY(I,J,K4,K5) = YBODY(I,J,K3,K5)
  630 CALL YPRTCD(LER,BODY(1,K1,K5),YBODY(1,1,K3,K5),K2)
  640 CALL PRTERP (P,LER,N,XPERT,DBODY,YBODY(1,1,1,K5))
```

```
  650 IF(N .LT. 9) GO TO 590
C
C           SET UP MERCURY INTERPOLATION
      IF(KISS(1) .LT. 0) GO TO 730
      NTAB =NN/2
      N    =NN-2*NTAB
      P(1) =N
      P(1) =(P(1)+FRACT1)/DMERC(1)
      NTAB =NTAB+1
      N    =1
      GO TO 340
C
C           PERFORM MERCURY INTERPOLATION
  660 IF (NTAB.EQ.NTMERC) GO TO 720
      IF (MTAB.EQ.NTMERC) GO TO 680
      IF (NTAB.EQ.(NTMERC+1)) GO TO 670
      K1 = NTAB+ 6
      K2 = 2
      K3 = 1
      GO TO 710
  670 K1 = MTAB + 6
      K3 = 2
      GO TO 690
  680 K1 = NTAB + 6
      K3 = 1
  690 K2 = 1
      K4 = 3 - K3
      LIMVL =LIMVEL
      DO 700 I=1,5
      DO 700 J=1,LIMVL
  700 YMERC(I,J,K4) = YMERC(I,J,K3)
  710 CALL YPRTCD(LER,MERC(1,K1),YMERC(1,1,K3),K2)
      NTMERC = NTAB
  720 CALL PRTERP(P,LER,N,XPERT,DMERC,YMERC)
C
C     COMPUTE SOLAR SYSTEM BARYCENTER. STOP PROGRAM IF EITHER
C     SUM OF KISS ARRAY 1 THROUGH 10 .NE. 10 OR ALL BODIES NOT ON TAPE.
C
  730 IF(KISS(13) .NE.1) GO TO 800
      ITRIG2=0
      DO 740 I=1,10
  740 ITRIG2=ITRIG2+KISS(I)
      IF (ITRIG2 .EQ. 10) ITRIG=1
      IF(ITRIG .EQ. 1 ) GO TO 760
      WRITE(6,750) NBDY,ITRIG
  750 FORMAT(/,1X,'FATAL ERROR OCCURS WHEN SOLAR SYSTEM BARYCENTER IS',
     1' ABOUT TO BE COMPUTED.',/,1X,'# OF BODIES ON TAPE WAS',I3,
     2/,1X,'SUM OF KISS ARRAY THROUGH 10 WAS',I3)
      STOP
C
C     FIRST ZERO OUT XCNT ARRAY SO THAT ACCUMULATION CAN BE DONE.
  760 DO 770 I=1,LIMVEL
  770 XCNT(I)=0.0D0
C
```

```
C     OUTER LOOP IS FOR POSITION, VELOCITY AND ACCELERATION. INNER
C     LOOP IS THE INDIVIDUAL BODY CONTRIBUTION SUM.
      DO 790 I=1,LIMVEL
      DO 780 J=1,9
      XCNT(I)=XCNT(I)+XPERT(I,J)*MASS1(J)
  780 CONTINUE
      XCNT(I)=XCNT(I)/MASCNT
  790 CONTINUE
C
  800 RETURN
      END
```

3 APPENDIX B - SUBROUTINE LVECTR

```
      SUBROUTINE LVECTR(L,LV)
C
C     THIS SUBROUTINE COMPUTES FOR A GIVEN L A VECTOR OF DO LOOP LIMITS
C     USED IN READING BODY DATA INTO STORAGE. THE VECTOR IS LV AND IS OF
C     SIZE 7
C
C     PETER MORGAN SEPT 81
C
      INTEGER *4 LV(7)
      LV(1)=L
      LV(3)=L*10
      LV(2)=LV(3)-9
      LV(5)=L*5
      LV(4)=LV(5)-4
      LV(7)=L*40
      LV(6)=LV(7)-39
      RETURN
      END
```

)

4 APPENDIX C - SUBROUTINE SKIPF

```
      SUBROUTINE SKIPF(IFILE,JDBD,FRCT,IVL,MVL)
C
C     SUBROUTINE TO FORWARD SKIP BLOCKS OF RECORDS IN THE 80 COLUMN
C     EPHEMERIS TAPE. IFILE IS THE LOGICAL UNIT NUMBER OF THE TAPE/FILE.
C     A RECORD BLOCK IS 261 PHYSICAL TYPE3 3 RECORDS.
C     POSITION OF TAPE OR FILE IS GIVEN BY RETURN PARAMETERS,JDBD,FRCT,
C     IVL,AND MVL
C  IFILE       :I*4  LOGICAL UNIT NUMBER OF EPHEMERIS FILE
C  JDBD        :I*4  JULIAN DATE AT MIDNIGHT OF FIRST RECORD OF BLOCK
C                    OVER WHICH SKIPPING WILL OCCUR.
C  FRCT        :R*8  FRACTION OF DAY FROM MIDNIGHT OF FIRST RECORD OF
C                    BLOCK
C  IVL         :I*4  NUMBER OF COORDINATES FOR PLANETS (6 FOR POSITION
C                    & RATE)
C  MVL         :I*4  NUMBER OF COORDINATES FOR MOON (6 FOR POSITION
C                    & RATE)
C
C     PETER MORGAN SEPT 81
C
      REAL *8 FRCT
      DIMENSION JUNK(20)
      READ(IFILE,30) JDBD,FRCT,IVL,MVL
      DO 10 I=1,260
   10 READ(IFILE,20) JUNK
   20 FORMAT(20A4)
   30 FORMAT(I10,D25.18,2I5)
      RETURN
      END
```

5 APPENDIX D – SUBROUTINE SKIPB

```
      SUBROUTINE SKIPB(IFILE,JDBD,FRCT,IVL,MVL)
C
C     SUBROUTINE TO BACKWARD SKIP BLOCKS OF RECORDS IN THE 80 COLUMN
C     EPHEMERIS TAPE. IFILE IS THE LOGICAL UNIT NUMBER OF THE TAPE/FILE.
C     A RECORD BLOCK IS 261 PHYSICAL TYPE3 3 RECORDS.
C     POSITION OF TAPE OR FILE IS GIVEN BY RETURN PARAMETERS,JDBD,FRCT,
C     IVL,AND MVL
C  IFILE       :I*4  LOGICAL UNIT NUMBER OF EPHEMERIS FILE
C  JDBD        :I*4  JULIAN DATE AT MIDNIGHT OF FIRST RECORD OF BLOCK
C                    OVER WHICH SKIPPING HAS OCCURRED.
C  FRCT        :R*8  FRACTION OF DAY FROM MIDNIGHT OF FIRST RECORD OF
C                    BLOCK
C  IVL         :I*4  NUMBER OF COORDINATES FOR PLANETS (6 FOR POSITION
C                    & RATE)
C  MVL         :I*4  NUMBER OF COORDINATES FOR MOON (6 FOR POSITION
C                    & RATE)
C
C     PETER MORGAN SEPT 81
C
      REAL *8 FRCT
      DO 10 I=1,261
   10 BACKSPACE IFILE
      READ(IFILE,20) JDBD,FRCT,IVL,MVL
   20 FORMAT(I10,D25.18,2I5)
      BACKSPACE IFILE
      RETURN
      END
```

6 APPENDIX E - SUBROUTINE READF

```
      SUBROUTINE READF (IFILE)
C     SUBROUTINE TO REPOSITION EPHEMERIS TAPE TO BEGINNING OF TAPE AND
C     THEN READ TYPE 1 RECORD BLOCK.
C     TYPE ONE DATA RECORDS ARE TWO CHARACTER RECORDS DESCRIBING THE
C     EPHEMERIS TAPE IN ENGLISH OR AUSTRALIAN.
C     INPUT PARAMETERS ONLY IFILE THE LOGICAL UNIT NUMBER AT WHICH THE
C     EPHEMERIS TAPE IS MOUNTED. THERE ARE NO OUTPUT PARAMETERS.
C
C     PETER MORGAN SEPT 1981
C
      INTEGER *4 JUNK1(20),JUNK2(20)
      REWIND IFILE
      READ(IFILE,20,ERR=30) JUNK1
      READ(IFILE,20) JUNK2
      WRITE(6,10) JUNK1,(JUNK2(I),I=1,12)
   10 FORMAT(/,1X,'THE HEADER LABEL ON THE EPHEMERIS TAPE WAS:',
     1/,1X,20A4,/,1X,12A4)
      RETURN
   20 FORMAT(20A4)
   30  WRITE(6,40)
   40 FORMAT(/,1X,'AN ERROR OCCURRED ON OPENING THE EPHEMERIS TAPE'
     1,' PROGRAM ABORTING')
      STOP
      END
```

7 APPENDIX F - SUBROUTINE READG

```
      SUBROUTINE READG(IFILE,NBDY,NPL,NCP,INTB,JDBD1,JDBD2,JDBD0,BETA,
     1MEQINC,NAME,NMOON,NBDY1,INTBD,JVLBD,MASS1)
C     SUBROUTINE TO READ TYPE 2 BLOCKS OF RECORDS FROM FIXED FORMAT
C     EPHEMERIS TAPE.
C  IFILE          :I*4  LOGICAL UNIT NUMBER OF EPHEMERIS FILE.
C  NBDY           :I*4  NUMBER OF BODIES ON THE NBODY TAPE
C  NPL(NBDY)      :I*4  IDENTIFYING NUMBER FOR EACH BODY:NOTE VARIABLE
C                       DIMENSION.
C                       KEY: MERCURY=1, VENUS=2, EARTH-MOON BARYCENTER=3,
C                            MARS=4,JUPITER=5, SATURN=6, URANUS=7,
C                            NEPTUNE=8,PLUTO=9,MOON=10.
C  NCP(NBDY)      :I*4  IDENTIFYING NUMBER FOR CENTRAL BODY FOR EACH
C                       NPL(I): NOTE NORMALLY NCP(1) THROUGH NCP(9)=0 AND NCP
C                       NCP(10)=3.
C  INTB(NBDY)     :I*4  INTERVAL IN DAYS BETWEEN TABULAR POINTS FOR EACH
C                       BODY. NORMALLY MERCURY HAS INTB=2, VENUS THROUGH
C                       PLUTO HAVE INTB=4, MOON HAS INTB=-1
C                       (MEANING 2**(-1):0.5DAY).
C                       NOTE INTB .GT. 0 IS IN DAYS NOT POWERS OF 2.
C  JDBD1          :I*4  STARTING TIME OF TAPE.
C  JDBD2          :I*4  ENDING TIME OF TAPE.
C  JDBD0(NBDY)    :I*4  EPOCH OF INITIAL CONDITIONS OF INTEGRATION FOR
C                       BODY NPL(I).
C  BETA(6,NBDY)   :R*8  SIX INITIAL CONDITION FOR EACH BODY REFERENCED TO
C                       MEAN EQUATOR AND EQUINOX OF 1950.0.
C                       KEY: A(AU), E, INC(DEG), RA OF ASCENDING NODE(DEG)
C                            ARGUMENT OF PERIGEE(DEG), INITIAL MEAN
C                            ANOMALY(DEG).
C  MEQINC         :R*8  INCLINATION OF LUNAR EQUATOR TO ECLIPTIC
C                       FOR USE WITH LIBRATION ANGLES ON TAPE.
C  NAME(6,12  )   :R*4  24 CHARACTER TITLE FOR EACH BODY INTEGRATION
C                       PLUS THE NUTATION AND LIBRATION TABULATIONS. THE
C                       FORMAT IS FOR THE BODY TO BE CENTERED ON THE
C                       FIRST 8 CHARACTERS (++MARS++) FOLLOWED BY THE
C                       INTEGRATION RUN NUMBER AND DATE.
C  NMOON          :I*4  EQUALS 0 FOR PEP-GENERATED TAPES. MEANS LUNAR
C                       EPHEMERIS UNIT OF DISTANCE IS THE ASTRONOMICAL
C                       UNIT (AU).
C  NBDY1          :I*4  NUMBER OF BODIES (EXCLUDING MOON)
C  INTBD          :I*4  INTEGRATION STEP SIZE FOR NBDY INTEGRATION
C                       (DEFAULT IS 2 DAYS).
C  JVLBD          :I*4  SET EQUAL TO ZERO. INDICATES VELOCITY AS WELL AS
C                       POSITION ON TAPE AT EACH TABULAR POINT.
C  MASS1(NBDY)    :R*8  MASS OF BODY IN UNITS OF SOLAR MASS.
C
C      PETER MORGAN SEPT 81
C
      INTEGER *4 NPL(10),NCP(10),INTB(10)
      INTEGER *4 JDBD1,JDBD2,JDBD0(10)
      REAL *4 NAME(8,10)
```

```
      REAL *8 BETA(6,10),MASS1(10),MEQINC
      READ(IFILE,10) NBDY,(NPL(I),I=1,NBDY)
   10 FORMAT(I2,3X,10I3)
      READ(IFILE,20) (NCP(I),I=1,NBDY)
   20 FORMAT(10I3)
      READ(IFILE,20) (INTB(I),I=1,NBDY)
      READ(IFILE,30) JDBD1,JDBD2,(JDBD0(I),I=1,NBDY)
   30 FORMAT(2I10/10I8)
      READ(IFILE,40) ((BETA(I,J),I=1,6),J=1,NBDY)
   40 FORMAT(3D25.18)
      READ(IFILE,50) MEQINC
   50 FORMAT(D25.18)
      READ(IFILE,60) ((NAME(I,J),I=1,6),J=1,   12)
   60 FORMAT(6A4)
      READ(IFILE,70) NMOON,NBDY1,INTBD,JVLBD,(MASS1(I),I=1,NBDY)
   70 FORMAT(4I5,/,(D25.18))
      RETURN
      END
```

8 APPENDIX G - SUBROUTINE READH

```
      SUBROUTINE READH (IFILE,L,JDBD,FRCT,IVL,MVL)
C
C     SUBROUTINE TO READ HEADER RECORD FROM BLOCK OF TYPE 3 RECORDS.
C     A BLOCK OF TYPE 3 RECORDS CONSISTS OF A HEADER RECORD CONTAINING
C     PARAMETERS JDBD,FRCT,IVL AND MVL. THE SUBSEQUENT RECORDS CONTAIN
C     THE COORDINATES THEMSELVES.
C     NOTE: SUBROUTINE READI READS ALL THE SUBSEQUENT RECORDS OF THE
C     TYPE 3 BLOCK. THUS, TO ACCOMPLISH A SEQUENCE OF READS OF THE
C     EPHEMERIS TAPE YOU CALL READH - READI THEN READH - READI ETC.
C  IFILE       :I*4  LOGICAL UNIT NUMBER OF EPHEMERIS FILE
C  JDBD        :I*4  JULIAN DAY OF FIRST RECORD OF BLOCK.
C  FRCT        :R*8  FRACTION OF DAY FROM MIDNIGHT OF FIRST RECORD OF
C                    BLOCK.
C  IVL         :I*4  NUMBER OF COORDINATES FOR PLANETS (6 FOR POSITION
C                    & VELOCITY).
C  MVL         :I*4  NUMBER OF COORDINATES FOR MOON (6 FOR POSITION
C                    & VELOCITY).
C
C     PETER MORGAN SEPT 81
C
      REAL *8 FRCT(3)
      INTEGER *4 JDBD(3)
      READ(IFILE,10) JDBD(L),FRCT(L),IVL,MVL
   10 FORMAT(I10,D25.18,2I5)
      RETURN
      END
```

9 APPENDIX I - SUBROUTINE READI

```
      SUBROUTINE READI (IFILE,LV,IVL,MVL,NBDY,MERC,BODY,MON,PSID,EPSD,
     1LIBRT)
C
C     SUBROUTINE TO READ BODY (PLANET) DATA FROM 80 COLUMN TAPE/FILE.
C     THIS SUBROUTINE READS ONLY 260 TYPE 3 RECORDS OF 261 RECORD BLOCK
C     THE FIRST RECORD OF A TYPE 3 BLOCK IS A HEADER RECORD. IT IS
C     ASSUMED THAT THIS RECORD HAS BEEN READ BY SUBROUTINE READH
C     THE PARAMETERS IVL AND MVL ARE GATHERED FROM THIS HEADER RECORD.
C     THE PARAMETER NBDY GIVING THE NUMBER OF BODIES ON THE TAPE IS READ
C     FROM THE TYPE 2 RECORD BLOCK. FOR THE FIXED RECORD EPHEMERIS
C     TAPE THE NUMBER OF BODIES SHOULD BE 10; THAT IS, ALL PLANETS PLUS
C     THE MOON.
C
C     INPUT PARAMETERS:
C
C  IFILE       :I*4  LOGICAL UNIT NUMBER OF EPHEMERIS FILE
C  LV(7)       :I*4  VECTOR OF DO LOOP LIMITS INDICATING FIRST MIDDLE
C                    OR END RECORDS AND THEIR STORAGE POSITIONS.
C  IVL         :I*4  NUMBER OF COORDINATES FOR PLANETS (6 FOR POSITION
C                    AND RATE)
C  MVL         :I*4  NUMBER OF COORDINATES FOR MOON (6 FOR POSITION
C                    AND RATE)
C  NBDY        :I*4  NUMBER OF BODIES ON EPHEMERIS TAPE
C
C     OUTPUT PARAMETERS
C
C     THE OUTPUT ARRAYS ARE ALL DIMENSIONED TO HOLD 3 TYPE-3 RECORD
C     BLOCKS IN CORE. SINCE EACH BLOCK SPANS 20 DAYS, THE MAXIMUM
C     DIMENSION OF EACH ARRAY IS 3*(20/INTB), WHERE INTB IS THE
C     INTERVAL IN DAYS USED TO TABULATE THE BODY'S POSITION.
C
C  MERC(IVL,30) :R*8  COORDINATES OF MERCURY AT 2 DAY INTERVALS.
C
C  BODY(IVL,15,NBDY-2)
C               :R*8  COORDINATES OF PLANETS EXCEPTING MERCURY AND MOON
C                     TABULATED AT 4 DAY INTERVALS.
C  MON(MVL,120) :R*8  COORDINATES OF MOON TABULATED AT 0.5 DAY INTERVALS
C
C  PSID(120)    :R*4  NUTATION IN LONGITUDE- DELTA(PSI) TABULATED AT
C                     0.5 DAY INTERVALS.
C  EPSD(120)    :R*4  NUTATION IN OBLIQUITY- DELTA(EPSILON) TABULATED
C                     AT 0.5 DAY INTERVALS.
C  LIBRT(120,3) :R*4  LUNAR PHYSICAL LIBRATIONS: TAU, RHO, AND
C                     I*(TAU-SIGMA) TABULATED AT 0.5 DAY INTERVALS.
C                     NOTE REVERSAL OF ORDER OF ARRAY.
C
C      PETER MORGAN SEPT 1981
C
      INTEGER *4 NBDY,NBDY2
      INTEGER *4 LV(7)
```

```
      REAL *4 PSID(120),EPSD(120),LIBRT(120,3)
      REAL *8 MERC(6,30),BODY(6,15,8),MON(6,120)
      NBDY2=NBDY-2
      L=LV(1)
      L1=LV(2)
      L2=LV(3)
      L3=LV(4)
      L4=LV(5)
      L5=LV(6)
      L6=LV(7)
      DO 10 I=L1,L2
   10 READ(IFILE,20) (MERC(J,I),J=1,IVL)
   20 FORMAT(3D25.18)
      DO 30 I=1,NBDY2
   30 READ(IFILE,20) ((BODY(K,J,I),K=1,IVL),J=L3,L4)
      READ(IFILE,20) ((MON(I,J),I=1,MVL),J=L5,L6)
      READ(IFILE,40)(PSID(I),EPSD(I),I=L5,L6)
   40 FORMAT(2E15.8)
      READ(IFILE,50) ((LIBRT(J,I),I=1,3),J=L5,L6)
   50 FORMAT(3E15.8)
      RETURN
      END
```

10 APPENDIX I - SUBROUTINE YPRTCD

```
      SUBROUTINE YPRTCD(LER,X,Y,N)
C
C     SUBROUTINE TO COMPUTE EVERETT INTERPOLATION Y-VECTORS.
C
C     SUBROUTINE ADAPTED FROM YPRTCD WRITTEN BY M.ASH FOR PEP.
C
C     INPUT PARAMETERS ARE:
C
C  LER          :I*4  LER=0  POSITIONS ONLY ARE REQUIRED.
C                     LER=1  POSITIONS AND VELOCITIES ARE REQUIRED.
C                     LER=2  POSITIONS, VELOCITIES & ACCELERATIONS ARE
C                            REQUIRED
C  X(6,84)      :R*8  VECTOR TO BE INTERPOLATED.
C  N            :I*4
C
C     OUTPUT PARAMETERS ARE:
C
C  Y(5,9,41)    :R*8  INTERPOLATION Y-VECTORS.
C
C     PETER MORGAN SEPT 81
C
C     DECLARATIONS.
C
      REAL *8 X(6,84),Y(5,9,41),EVCF(25),FACT(9),F1,F2,F3,F4
      EQUIVALENCE (F1,FACT(1))
C           INTERPOLATION COEFFICIENTS
      DATA EVCF(1),EVCF(2),EVCF(3),EVCF(4),EVCF(5)
     1                        / 1.7873015873015873D 00,
     2-0.4960317460317460D 00, 0.1206349206349206D 00,
     3-0.1984126984126984D-01, 0.1587301587301587D-02/
      DATA EVCF(6),EVCF(7),EVCF(8),EVCF(9),EVCF(10)
     1                        /-0.9359567901234568D 00,
     2 0.6057098765432098D 00,-0.1632716049382716D 00,
     3 0.2779982363315696D-01,-0.2259700176366843D-02/
      DATA EVCF(11),EVCF(12),EVCF(13),EVCF(14),EVCF(15)
     1                        / 0.1582175925925926D 00,
     2-0.1171296296296296D 00, 0.4606481481481481D-01,
     3-0.8796296296296296D-02, 0.7523148148148148D-03/
      DATA EVCF(16),EVCF(17),EVCF(18),EVCF(19),EVCF(20)
     1                        /-0.9755291005291005D-02,
     2 0.7605820105820106D-02,-0.3505291005291005D-02,
     3 0.8597883597883598D-03,-0.8267195767195767D-04/
      DATA EVCF(21),EVCF(22),EVCF(23),EVCF(24),EVCF(25)
     1                        / 0.1929012345679012D-03,
     2-0.1543209876543210D-03, 0.7716049382716048D-04,
     3-0.2204585537918871D-04, 0.2755731922398589D-05/
      DATA FACT(2),FACT(3),FACT(4),FACT(5),FACT(6),FACT(7),FACT(8),
     1 FACT(9)            /3.0D0,5.0D0,7.0D0,9.0D0,2.0D0,4.0D0,6.0D0,
     28.0D0/
C
```

```
C            DETERMINATION OF POSITION Y-VECTORS
      DO 30 K=1,N
      NR =K+4
      DO 20 L=1,3
      F1 =X(L,NR+1)+X(L,NR-1)
      F2 =X(L,NR+2)+X(L,NR-2)
      F3 =X(L,NR+3)+X(L,NR-3)
      F4 =X(L,NR+4)+X(L,NR-4)
      DO 10 I=1,5
      J   =(I-1)*5+1
      Y(I,L,K) =EVCF(J)*X(L,NR)+(EVCF(J+1)*F1+(EVCF(J+2)*F2+(EVCF(J+3)*
     1       F3+EVCF(J+4)*F4)))
   10 CONTINUE
   20 CONTINUE
   30 CONTINUE
C
C            DETERMINATION OF VELOCITY,ACCELERATION Y-VECTORS (IF LER>0)
      IF(LER .LE. 0) GO TO 70
C            FUTURE OPTION TO CALCULATE VELOCITY Y-VECTORS FROM
C            COORDINATES IF THEY EXIST RATHER THAN BY NUMERICAL
C            DIFFERENTIATION
      DO 60 L=1,3
      J  =L+3
      M  =J+3
      DO 50 K=1,N
      Y(1,J,K) =Y(1,L,K)
      DO 40 I=2,5
      Y(I,J,K) =Y(I,L,K)*FACT(I)
      Y(I,M,K) =Y(I,J,K)*FACT(I+4)
   40 CONTINUE
   50 CONTINUE
   60 CONTINUE
   70 RETURN
      END
```

11 APPENDIX J - SUBROUTINE YNUTLB

```
      SUBROUTINE YNUTLB(NLB,YNLB,K2,KISS)
C
C     SUBROUTINE TO COMPUTE Y-VECTORS FOR NUTATION AND LIBRATION.
C
C     SUBROUTINE ADAPTED FROM YNUTLB WRITTEN BY M.ASH FOR PEP
C
C     INPUT PARAMETERS ARE:
C
C  NLB(120)     :R*4  VECTOR TO BE INTERPOLATED.
C  K2           :I*4
C  KISS         :I*4  A SWITCH PARAMETER TO DETERMINE IF
C                     FIRST DIFFERENTIALS ARE TO BE COMPUTED.
C
C     OUTPUT PARAMETERS ARE:
C
C  YNLB(3,2,2) :R*8  INTERPOLATION Y-VECTOR.
C
C     PETER MORGAN SEPT. 1981
C
C     DECLARATIONS.
      REAL *4 NLB(120)
      REAL *8 YNLB(3,2,2),F1,F2,FACT(3)
      REAL*8 EVCF(9) /1.533333333333333D0,-3.0D-1, 3.333333333333333D-2,
     1-5.833333333333333D-1, 3.333333333333333D-1,-4.166666666666667D-2,
     2 5.000000000000000D-2,-3.333333333333333D-2, 8.333333333333333D-3/
      DATA FACT /0.,3.,5./
C
      DO 20 K=1,K2
      NR=K+4
         F1 = DBLE(NLB(NR+1)) + DBLE(NLB(NR-1))
         F2 = DBLE(NLB(NR+2)) + DBLE(NLB(NR-2))
      DO 10 I=1,3
           J = 3*I - 2
      YNLB(I,1,K) = EVCF(J)*NLB(NR) + EVCF(J+1)*F1 + EVCF(J+2)*F2
   10 CONTINUE
   20 CONTINUE
C
      IF(KISS .LE. 0) GO TO 50
      DO 40 K=1,K2
      YNLB(1,2,K) =YNLB(1,1,K)
      DO 30 I=2,3
      YNLB(I,2,K) =YNLB(I,1,K)*FACT(I)
   30 CONTINUE
   40 CONTINUE
C
   50 RETURN
      END
```

12 APPENDIX K - SUBROUTINE PRTERP

```
      SUBROUTINE PRTERP(P,LER,N,XPERT,DIST,Y)
C
C     EVERETT EIGHTH DIFFERENCE INTERPOLATION FOR MOON AND PLANETS.
C
C     SUBROUTINE ADAPTED FROM PRTERP WRITTEN BY M.ASH FOR PEP.
C
C     INPUT PARAMETERS ARE:
C
C  P(4)          :R*8  VECTOR OF INTERPOLATION COEFFICIENTS.
C  LER           :I*4  A SWITCH FOR INTERPOLATION OF VECTOR AND FIRST AND
C                      SECOND DERIVATIVES.
C  N             :I*4  THE NUMBER OF THE BODY BEING INTERPOLATED.
C  DIST(2)       :R*8  TABULAR INTERVAL OF EPHEMERIS AND SQUARE.
C  Y(5,9,2)      :R*8  VECTOR TO BE INTERPOLATED.
C
C     OUTPUT PARAMETERS ARE:
C
C  XPERT(9,10)   :R*8  INTERPOLATED POSITION VECTOR FOR BODY N;
C       (I,N)          I=1,3 FOR POSITIONS; I=4,6 FOR VELOCITIES AND
C                      I=6,9 FOR ACCELERATIONS.
C
C     WRITTEN BY P. MORGAN SEPT 1981.
C
C     DECLARATIONS:
C
      REAL *8 P(4),Y(5,9,2),DIST(2),XPERT(9,10)
C
C           DETERMINE POSITION COORDINATES FOR BODY N
      DO 10 I=1,3
      XPERT(I,N) = P(1)*(Y(1,I,2)+P(2)*(Y(2,I,2)+P(2)*(Y(3,I,2)
     1            +P(2)*(Y(4,I,2)+P(2)* Y(5,I,2)))))
     2            +P(3)*(Y(1,I,1)+P(4)*(Y(2,I,1)+P(4)*(Y(3,I,1)
     3            +P(4)*(Y(4,I,1)+P(4)* Y(5,I,1)))))
   10 CONTINUE
C
C           DETERMINE VELOCITY,ACCELERATION FOR BODY N (IF THERE IS REL)
      IF(LER .LE. 0) GO TO 30
      DO 20 I=4,6
      XPERT(I,N) =(Y(1,I,2(2,I,2)+P(2)*(Y(3,I,2)+P(2)*
1        (Y(4,I,2)+P(2)* Y(5,I,2))))
        2            -Y(1,I,1)-P(4)*(Y(2,I,1)+P(4)*(Y(3,I,1)+P(4)*
        3            (Y(4,I,1)+P(4)* Y(5,I,1)))))/DIST(1)
         IF(LER.LE.1) GO TO 20
      J  =I+3
      XPERT(J,N) =(P(1)*(Y(2,J,2)+P(2)*(Y(3,J,2)+P(2)*(Y(4,J,2)
     1            +P(2)* Y(5,J,2))))
     2            +P(3)*(Y(2,J,1)+P(4)*(Y(3,J,1)+P(4)*(Y(4,J,1)
     3            +P(4)* Y(5,J,1)))))/DIST(2)
   20 CONTINUE
   30 CONTINUE
```

```
C
      RETURN
      END
```

13 APPENDIX L - SUBROUTINE NUTERP

```
      SUBROUTINE NUTERP(P,YNLB,NLB,DMOON,KISS)
C
C     FOURTH DIFFERENCE INTERPOLATOR FOR NUTATIONS AND LIBRATIONS.
C
C     SUBROUTINE ADAPTED FROM NUTERP WRITTEN BY M.ASH FOR PEP.
C
C     INPUT PARAMETERS ARE:
C
C  P(4)          :R*8  VECTOR OF INTERPOLATION COEFFICIENTS.
C  YNLB(3,2,2)   :R*8  VECTOR TO BE INTERPOLATED.
C      (I,J,K)
C  DMOON(2)      :R*8  TABULATION INTERVAL AND SQUARE.
C  KISS          :I*4  SWITCH ON WHICH TO TEST IF FIRST DERIVATIVES
C                      ARE REQUIRED.
C
C     OUTPUT PARAMETERS ARE:
C
C  NLB(2)        :R*4  THE INTERPOLATED VALUE & IF REQUESTED FIRST
C                      DERIVATIVE.
C
C     WRITTEN BY PETER MORGAN SEPT 1981
C
C     DECLARATIONS
C
      REAL *8 P(4),YNLB(3,2,2),DMOON(2)
      REAL *4 NLB(2)
      NLB(1)=P(1)*(YNLB(1,1,2)+P(2)*(YNLB(2,1,2)+P(2)*
     1           YNLB(3,1,2))) +
     2           P(3)*(YNLB(1,1,1) + P(4)*(YNLB(2,1,1) + P(4)*
     3           YNLB(3,1,1)))
       IF(KISS .LT. 1) GO TO 10
       NLB(2) = (YNLB(1,2,2) + P(2)*(YNLB(2,2,2) + P(2)*
     1          YNLB(3,2,2)) -YNLB(1,2,1) - P(4)*(YNLB(2,2,1)
     2           + P(4)*YNLB(3,2,1)))/DMOON(1)
C
   10 RETURN
      END
```

14 APPENDIX M - FIRST 3 DATA BLOCKS ON EPHEMERIS TAPE

```
 N-BODY RUN 311  9-BODY INTEGRATION 1961-1980 (305A I.C. INNER, 308 I.C. OUTER)
 8/16/69 PERTURBING PLNT TAPE FROM N-BODY INTEG.
10      1  2  3  4  5  6  7  8  9 10
  0  0  0  0  0  0  0  0  0  3
  2  4  4  4  4  4  4  4  4 -1
   2439121   2439521
 2440001 2440001 2440001 2440001 2440001 2440001 2440001 2440001 2440001 2440001
 0.38709882510136929 5D+00 0.20561435213449359 7D+00 0.28603371900040429 0D+02
 0.10859462530194820 0D+02 0.66932771939487519 6D+02 0.90839652202660797 5D+02
 0.72332807876331399 5D+00 0.67575406466986219 8D-02 0.24466854297897949 5D+02
 0.79781583135902709 4D+01 0.12378519103207569 8D+03 0.27440656335905708 8D+03
 0.99998276889251139 2D+00 0.16752718206768469 3D-01 0.23443312899451680 0D+02
 0.67546688060835967 0D-03 0.10203424253941180 0D+03 0.13936583362690339 9D+03
 0.15236045509404689 8D+01 0.93457824247582491 1D-01 0.24692967792928467 2D+02
 0.33466695743756669 2D+01 0.33210132079016239 0D+03 0.89899757820611206 7D+02
 0.52029462893508418 0D+01 0.48191780321928989 6D-01 0.23253022452121538 1D+02
 0.32610305963079639 9D+01 0.10373942982511720 0D+02 0.14118585856959059 7D+03
 0.95260463911804718 1D+01 0.54617208150069689 7D-01 0.22573218408700636 7D+02
 0.59687276499093249 2D+01 0.87593376735933809 1D+02 0.28974045665075476 6D+03
 0.19274453386370598 9D+02 0.51242037676612309 4D-01 0.23671477120432108 2D+02
 0.18473245138153708 8D+01 0.16834379820028920 0D+03 0.69909825906479949 9D+01
 0.30113698009424137 3D+02 0.69842230513267097 2D-02 0.22314827374776168 5D+02
 0.35199288698780639 7D+01 0.54075073580418347 8D+02 0.17759443490287989 7D+03
 0.39746941888417318 0D+02 0.25223738404518229 7D+00 0.23655569945193299 5D+02
 0.43770131195799159 2D+02 0.18196316840502969 8D+03 0.32984752745167719 5D+03
 0.25715143746425089 7D-02 0.55615447357618669 2D-01 0.28396854386341477 3D+02
 0.33129592098740428 0D+01 0.22627124722235860 0D+03 0.15488569568492079 8D+03
 0.0268587D0
MERCURY RUN 311  8/16/69
 VENUS  RUN 311  8/16/69
 EMBARY RUN 311  8/16/69
  MARS  RUN 311  8/16/69
JUPITER RUN 311  8/16/69
 SATURN RUN 311  8/16/69
 URANUS RUN 311  8/16/69
NEPTUNE RUN 311  8/16/69
 PLUTO  RUN 311  8/16/69
  MOON  RUN 440- 8/ 7/72
 MROTAT  KOZIEL   1967
 EROTAT   WHAR    1981
    0     9   2    0
 0.16578480204299926 5D-06
 0.24478485858778717 7D-05
 0.30404368986205841 5D-05
 0.32271597766805434 2D-06
 0.95475346928768138 6D-03
 0.28577960676726105 8D-03
 0.43610989969472304 8D-04
 0.51921079958463135 9D-04
 0.55187637969094921 6D-06
```

```
 C.0
   2439121 0.0                              6     6
-0.39585489928523098 8D+00-0.72743582011388072 1D-01 0.17150499962489173 0D-02
-0.11991836235052072 5D-02-0.23450265967780333 0D-01-0.12434992976694780 4D-01
-0.39475028837217725 2D+00-0.11886530318465402 0D+00-0.23097767220793506 7D-01
 0.22580157733589925 0D-02-0.22614983686539154 5D-01-0.12342878910573219 1D-01
-0.38700989497634574 7D+00-0.16299845969549349 4D+00-0.47527433596982066 2D-01
 0.54349799954682047 7D-02-0.21471748767741879 3D-01-0.12057056477672954 2D-01
-0.37320082044190239 2D+00-0.20458387191946547 6D+00-0.71216229121752128 2D-01
 0.83264800569761128 8D-02-0.20075547513423402 8D-01-0.11606434982201878 5D-01
-0.35389347033735402 1D+00-0.24316233093415429 2D+00-0.93859631118394445 9D-01
 0.10933853250298433 3D-01-0.18471589608214177 9D-01-0.11015362741378390 9D-01
-0.32965156377975621 3D+00-0.27835644964176521 6D+00-0.11519762153419234 3D+00
 0.13262041764005091 4D-01-0.16696730330230582 4D-01-0.10304084455395468 5D-01
-0.30102708676969178 2D+00-0.30985539624112609 6D+00-0.13500702984115130 2D+00
 0.15317559889463471 3D-01-0.14780900724137531 7D-01-0.94892982467615705 6D-02
-0.26855861266915699 6D+00-0.33740238576450908 0D+00-0.15309500700101634 5D+00
 0.17107120054581028 9D-01-0.12748423278770090 4D-01-0.85847190739584819 0D-02
-0.23277188893479229 5D+00-0.36078464917033249 7D+00-0.16929359338288592 4D+00
 0.18636711097929564 9D-01-0.10619171236560976 7D-01-0.76016048703010487 6D-02
-0.19418194469775772 2D+00-0.37982556299265629 6D+00-0.18345528745760174 3D+00
 0.19910978077082750 1D-01-0.84095721455907596 4D-02-0.65492302511953341 3D-02
 0.18684294813189508 7D+00 0.63882708423145689 9D+00 0.27606186539656435 4D+00
-0.19603677384450552 0D-01 0.42436086070682353 7D-02 0.31504412677177384 7D-02
 0.10741192233406640 6D+00 0.65172539748283038 5D+00 0.28689106959474074 1D+00
-0.20070047398646403 8D-01 0.21981181520574833 5D-02 0.22581732999944111 3D-02
 0.26620816189072695 3D-01 0.65637011453495865 1D+00 0.29408699897707460 4D+00
-0.20282842329725492 3D-01 0.12121143952686062 9D-03 0.13357316666429597 4D-02
-0.54507795051932608 8D-01 0.65268870724814863 3D+00 0.29755238003711581 6D+00
-0.20238484780478747 8D-01-0.19604739088526899 5D-02 0.39489349122438777 4D-03
-0.13494379921115962 1D+00 0.64071578037511724 1D+00 0.29723773798571485 9D+00
-0.19936794936906390 0D-01-0.40200714017552974 9D-02-0.55224636289183295 5D-03
-0.65013677324273488 9D-01 0.90030129832665735 4D+00 0.39041065068238356 4D+00
-0.17448190446116117 7D-01-0.10985557059217075 1D-02-0.47636221494385755 7D-03
-0.13458674646961892 0D+00 0.89367066952723348 0D+00 0.38753539631363292 9D+00
-0.17323902202653107 4D-01-0.22154541862526723 0D-02-0.96069924160669588 4D-03
-0.20348984774571729 5D+00 0.88259198764121923 9D+00 0.38273126811724478 3D+00
-0.17113345072987899 9D-01-0.33216124347164207 3D-02-0.14403785510886352 5D-02
-0.27137989749807149 6D+00 0.86711980636917267 2D+00 0.37602192442574326 7D+00
-0.16817597065752600 5D-01-0.44112465728889140 0D-02-0.19128918272726204 8D-02
-0.33791910331808201 8D+00 0.84733163961910919 2D+00 0.36744098084109079 5D+00
-0.16438224239237664 8D-01-0.54786667011619116 1D-02-0.23757715122422340 8D-02
 0.10840508461891798 0D+01-0.77171819925773455 0D+00-0.38333380576157402 6D+00
 0.92521384024253344 0D-02 0.11132139304157817 1D-01 0.48613653117508485 3D-02
 0.11200813300713226 0D+01-0.72651449944155083 2D+00-0.36355218447909402 9D+00
 0.87602256826807027 2D-02 0.11466482913325332 1D-01 0.50280410513129083 7D-02
 0.11541103328656394 7D+01-0.68001271597727784 6D+00-0.34312098684439908 1D+00
 0.82515759768805136 9D-02 0.11781043450222857 9D-01 0.51860855083021755 2D-02
 0.11860730851346343 9D+01-0.63229359339496747 6D+00-0.32207553653272870 4D+00
 0.77272886449679988 4D-02 0.12075024923134053 7D-01 0.53351037883020346 5D-02
 0.12159093327567305 2D+01-0.58344093562991435 1D+00-0.30045268278418690 9D+00
 0.71885211174418112 8D-02 0.12347694360681998 4D-01 0.54747283660730745 2D-02
 0.29696372142871845 0D+00 0.46998784378331732 7D+01 0.20089978045906546 9D+01
-0.76231986252540961 9D-02 0.66115178944009886 2D-03 0.46996364051329545 1D-03
```

```
 0.266465861511739441D+00 0.470244005369087104D+01 0.201084217935312815D+01
-0.762568602683512051D-02 0.619658053700624712D-03 0.452223457362566246D-03
 0.235958595433071489D+00 0.470483571859911676D+01 0.201261559042258842D+01
-0.762790176421336563D-02 0.578176570943504079D-03 0.434481904356384880D-03
 0.205443009147515121D+00 0.470706548465743579D+01 0.201431803365064233D+01
-0.762984618999959507D-02 0.536708887889387446D-03 0.416739648612341146D-03
 0.174920187172895605D+00 0.470912941018166387D+01 0.201594950757439606D+01
-0.763151967164380721D-02 0.495256564593383563D-03 0.398997365201801473D-03
 0.939865110408989457D+01-0.182394766896974292D+01-0.116039746488520668D+01
 0.922338565618071346D-03 0.503674030558125090D-02 0.204262136552797621D-02
 0.940231562228465534D+01-0.180379599548001734D+01-0.115222395921025056D+01
 0.909919112108714646D-03 0.503909216932957749D-02 0.204412977379959054D-02
 0.940593044567175474D+01-0.178363496578778657D+01-0.114404444029125085D+01
 0.897491172608719571D-03 0.504141840447532193D-02 0.204562798518269731D-02
 0.940949554043931591D+01-0.176346468241925813D+01-0.113585894893593364D+01
 0.885054813268694983D-03 0.504371900896107613D-02 0.204711599154567587D-02
 0.941301087303403961D+01-0.174328524786837047D+01-0.112766752596328090D+01
 0.872610095641111088D-03 0.504599399961164034D-02 0.204859379483846889D-02
-0.177729734370423706D+02 0.383844239318491676D+01 0.193297540158749026D+01
-0.960973886416227586D-03-0.367312096459336741D-02-0.159582824302039735D-02
-0.177768104537297980D+02 0.382374833119819679D+01 0.192659130077736562D+01
-0.957534172712796821D-03-0.367390953492950336D-02-0.159622194869125402D-02
-0.177806337081453627D+02 0.380905112086437958D+01 0.192020562770401915D+01
-0.954092749701735397D-03-0.367469513518589363D-02-0.159661437437978742D-02
-0.177844431934402891D+02 0.379435077411569965D+01 0.191381838749860989D+01
-0.950649611523560819D-03-0.367547773557936892D-02-0.159700551252197704D-02
-0.177882389027486632D+02 0.377964730304383667D+01 0.190742958534375950D+01
-0.947204757056103238D-03-0.367625728753700370D-02-0.159739534550454537D-02
-0.195707391535787991D+02-0.216014602604430266D+02-0.835608679104130636D+01
 0.237079710506702399D-02-0.183485280544284967D-02-0.811298629320883014D-03
-0.195612543135713466D+02-0.216087979291454566D+02-0.835933131528533968D+01
 0.237162298837771576D-02-0.183398147427178118D-02-0.810963462388057579D-03
-0.195517661689501523D+02-0.216161321116576417D+02-0.836257449849897938D+01
 0.237244941319505185D-02-0.183310970750692834D-02-0.810628113940992090D-03
-0.195422747175319600D+02-0.216234628061869998D+02-0.836581633994575680D+01
 0.237327638821765692D-02-0.183223747621903899D-02-0.810292576824218056D-03
-0.195327799571050278D+02-0.216307900107849171D+02-0.836905683883937157D+01
 0.237410391740781375D-02-0.183136473268549500D-02-0.809956833831454101D-03
-0.305812492297079075D+02 0.349348320443577243D+01 0.103722951500296636D+02
-0.750595021831047494D-04-0.312845374423225591D-02-0.967176440897862799D-03
-0.305815473634011425D+02 0.348096905463394890D+01 0.103684256867072004D+02
-0.740071480116030364D-04-0.312862106936532582D-02-0.967555203567567576D-03
-0.305818412853167025D+02 0.346845423658723018D+01 0.103645547085343408D+02
-0.729536120865895982D-04-0.312878786380731681D-02-0.967933865886072401D-03
-0.305821309907103291D+02 0.345593875246929638D+01 0.103606822159232244D+02
-0.718988856995990796D-04-0.312895409827438145D-02-0.968312420536132916D-03
-0.305824164748092251D+02 0.344342260461128169D+01 0.103568082093364704D+02
-0.708429648788333363D-04-0.312911972468944938D-02-0.968690850149689711D-03
 0.167449358457460822D-02-0.185226178945920250D-02-0.104416267147451879D-02
 0.443756046291836315D-03 0.323938380982394301D-03 0.115717376242538003D-03
 0.188642556933437349D-02-0.168027743857807049D-02-0.980564439860606393D-03
 0.403192221007736659D-03 0.363331290086340794D-03 0.138424235210601557D-03
 0.207694248750386624D-02-0.148962616879642971D-02-0.906006603531797346D-03
 0.358169050899078388D-03 0.398551348171793289D-03 0.159522602818547594D-03
```

```
 0.22439259162931881 6D-02-0.12824736477600553 0D-02-0.82134195463506387 9D-03
 0.30913414458526457 6D-03 0.42928702577088247D-03 0.17882072418784176 3D-03
 0.23854850031372622 2D-02-0.10611333079797844 4D-02-0.72751565219270633 8D-03
 0.25654993089622052 7D-03 0.45525832484799306 6D-03 0.19614061130681307 1D-03
 0.24999636482922780 5D-02-0.82805165037801750 5D-03-0.62555875618913086 1D-03
 0.20089302351634998 2D-03 0.47621239404397729 8D-03 0.21131637004630389 9D-03
 0.25859477783676891 0D-02-0.58579525668385430 5D-03-0.51658258740041110 3D-03
 0.14265521091342208 2D-03 0.49192113934232666 7D-03 0.22419260061163349 8D-03
 0.26422735037675443 3D-02-0.33703918904178891 9D-03-0.40177383003812700 7D-03
 0.82345980987328802 5D-04 0.50217850989427121 7D-03 0.23462305735804881 2D-03
 0.26680368895979064 5D-02-0.84556271301020543 6D-04-0.28239019970751011 7D-03
 0.20496416529661759 8D-04 0.50679943939703622 3D-03 0.24246976031212770 2D-03
 0.26626059677418385 0D-02 0.16879343689270968 4D-03-0.15975640139781967 9D-03
-0.42335783979993781 5D-04 0.50562027091468405 2D-03 0.24760275717159753 5D-03
 0.26256354715932208 8D-02 0.42007384352705460 4D-03-0.35260001361860580 1D-04
-0.10556050282416630 2D-03 0.49850110095659145D-03 0.24990074014030196 9D-03
 0.25570845747786030 7D-02 0.66628575108485497 7D-03 0.89653266136569173 2D-04
-0.16854907910079951 7D-03 0.48533038327305550 9D-03 0.24925272374760740 9D-03
 0.24572376483778678 6D-02 0.90438271358569055 3D-03 0.21348533796514636 1D-03
-0.23062847408867259 4D-03 0.46603208989309065 1D-03 0.24556098228916052 1D-03
 0.23267277036383967 9D-02 0.11312911603149420 2D-02 0.33469348187888971 2D-03
-0.29107651044999057 4D-03 0.44057564226964557 8D-03 0.23874542030935442 8D-03
 0.21665617470006704 7D-02 0.13439361953012695 7D-02 0.45170052626247931 6D-03
-0.34911916317574906 6D-03 0.40898867739587723 3D-03 0.22874949543001052 2D-03
 0.19781467370268605 1D-02 0.15392744595267708 4D-02 0.56290844688864348 0D-03
-0.40393106994981189 9D-03 0.37137247952135515 4D-03 0.21554771672831518 5D-03
 0.17633142094990256 4D-02 0.17143352779463237 7D-02 0.66671607412801251 7D-03
-0.45464057494239113 8D-03 0.32791956862008241 8D-03 0.19915459106332697 6D-03
 0.15243409633846555 3D-02 0.18662709508068017 9D-02 0.76154156949776743 1D-03
-0.50034066306007179 8D-03 0.27893248625462886 6D-03 0.17963467604924804 8D-03
 0.12639625469756780 6D-02 0.19924164365434552 5D-02 0.84585008352501068 0D-03
-0.54010700404748788 8D-03 0.22484227763217201 6D-03 0.15711312434208593 9D-03
 0.98537576158244308 3D-03 0.20903577788607802 2D-02 0.91818662177637824 9D-03
-0.57302392426667001 4D-03 0.16622459440766362 6D-03 0.13178579052069761 1D-03
 0.69222615948607104 7D-03 0.21580074621584383 4D-02 0.97721360330132545 2D-03
-0.59821839281636729 5D-03 0.10381084416241325 8D-03 0.10392766535191565 9D-03
 0.38857672442124533 6D-03 0.21936835120843314 1D-02 0.10217519164257413 3D-02
-0.61490103531845356 5D-03 0.38491544216049274 6D-04 0.73898176217270230 2D-04
 0.78854742271842499 2D-04 0.21961877511783664 7D-02 0.10508235238429353 3D-02
-0.62241185370357675 8D-03-0.28690822793919007 7D-04 0.42141839977094050 2D-04
-0.23222461739735618 9D-03 0.21648774174179146 3D-02 0.10636929524467788 7D-02
-0.62026693525785078 0D-03-0.96561456054021735 3D-04 0.91829662242073848 8D-05
-0.53975732678761848 0D-03 0.20997236538549252 9D-02 0.10599044711994545 3D-02
-0.60820129451006417 7D-03-0.16384608222033688 9D-03-0.24386361641423316 6D-04
-0.83876885943215298 4D-03 0.20013504654739961 5D-02 0.10393115698842891 8D-02
-0.58620247496441780 8D-03-0.22921588875058200 3D-03-0.57924974180012919 9D-04
-0.11243459844914076 8D-02 0.18710487953370800 4D-02 0.10020956253424812 4D-02
-0.55452994909988858 2D-03-0.29134143433534972 4D-03-0.90766916571358367 0D-04
-0.13917739355614014 2D-02 0.17107623840941072 2D-02 0.94877141907207890 5D-03
-0.51371680168476978 3D-03-0.34895068271216305 9D-03-0.12224933440163783 D-03
-0.16366696832619996 0D-02 0.15230447968915899 7D-02 0.88017837464164031 7D-03
-0.46455247981739102 6D-03-0.40088528847934189 0D-03-0.15174121903914444 7D-03
-0.18551021001289912 2D-02 0.13109899769811305 8D-02 0.79745782542307906 8D-03
-0.40804807804600547 0D-03-0.44614929305868123 1D-03-0.17867005876292587 3D-03
```

-0.20436911767016190D-02 0.10781413685651483D-02 0.70201804211484617D-03
-0.34538808132594258D-03-0.48394550769922185D-03-0.20254376338677597D-03
-0.21996808662168289D-02 0.82838655314194226D-03 0.59548988542469601D-03
-0.27787414824641574D-03-0.51369680046754788D-03-0.22296601487135061D-03
-0.23209831110860097D-02 0.56584516863063689D-03 0.47967661543183267D-03
-0.20686705926660412D-03-0.53505178062263011D-03-0.23964424893761626D-03
-0.24061935664367802D-02 0.29475759506965087D-03 0.35650151788544911D-03
-0.13373238568018269D-03-0.54787645716627655D-03-0.25239052872003565D-03
-0.24545819920046458D-02 0.19380700129132733D-04 0.22795665391841993D-03
-0.59794045308130325D-04-0.55223493511364518D-03-0.26111641465932498D-03
-0.24660619298619187D-02-0.25610480455768762D-03 0.96055346664257885D-04
 0.13701862139995482D-04-0.54836292301815463D-03-0.26582343709807317D-03
-0.24411450401258872D-02-0.52767275353996895D-03-0.37209835110470189D-04
 0.85612277082873914D-04-0.53663780244996419D-03-0.26659092265560034D-03
-0.23808854337016375D-02-0.79151477411245111D-03-0.16990268534073334D-03
 0.15491218052613596D-03-0.51754845053695551D-03-0.26356277980304051D-03
-0.22868187347843359D-02-0.10440888903168238D-02-0.30017237605194308D-03
 0.22070099304541051D-03-0.49166716180548296D-03-0.25693452204311631D-03
-0.21608996728105261D-02-0.12821549312389259D-02-0.42627623542952810D-03
 0.28220241009831501D-03-0.45962511235159360D-03-0.24694140862477836D-03
-0.71567731E-04 0.17581551E-04
-0.71410119E-04 0.17703380E-04
-0.71301634E-04 0.17829734E-04
-0.71242073E-04 0.17956394E-04
-0.71229442E-04 0.18079329E-04
-0.71260220E-04 0.18194492E-04
-0.71329210E-04 0.18298291E-04
-0.71429778E-04 0.18387393E-04
-0.71553965E-04 0.18458944E-04
-0.71692426E-04 0.18510807E-04
-0.71834904E-04 0.18541323E-04
-0.71970251E-04 0.18549748E-04
-0.72086841E-04 0.18536011E-04
-0.72172712E-04 0.18501130E-04
-0.72216179E-04 0.18447070E-04
-0.72206094E-04 0.18377104E-04
-0.72132665E-04 0.18295759E-04
-0.71988252E-04 0.18208753E-04
-0.71768212E-04 0.18123159E-04
-0.71472197E-04 0.18046674E-04
-0.71104645E-04 0.17987448E-04
-0.70675698E-04 0.17953207E-04
-0.70200855E-04 0.17950675E-04
-0.69700531E-04 0.17984552E-04
-0.69198373E-04 0.18057224E-04
-0.68719484E-04 0.18167979E-04
-0.68287889E-04 0.18313076E-04
-0.67924178E-04 0.18486026E-04
-0.67643530E-04 0.18677965E-04
-0.67454224E-04 0.18878665E-04
-0.67357192E-04 0.19077357E-04
-0.67346191E-04 0.19263854E-04
-0.67408997E-04 0.19429281E-04
-0.67528774E-04 0.19566971E-04

-0 67686196E-04 0.19672618E-04
-0.67861081E-04 0.19744461E-04
-0.68033973E-04 0.19783285E-04
-0.68187597E-04 0.19791783E-04
-0.68307447E-04 0.19774394E-04
-0.68382637E-04 0.19736515E-04
-0.29572984E-04 0.66689006E-03 0.25000260E-03
-0.20040854E-04 0.69389748E-03 0.16567609E-03
-0.10501899E-04 0.70999842E-03 0.78005905E-04
-0.10310823E-05 0.71473373E-03-0.11429607E-04
0.82940587E-05 0.70789666E-03-0.10098286E-03
0.17393744E-04 0.68954471E-03-0.18898462E-03
0.26187743E-04 0.65999944E-03-0.27379254E-03
0.34594283E-04 0.61983778E-03-0.35383669E-03
0.42532469E-04 0.56987535E-03-0.42766635E-03
0.49923154E-04 0.51113963E-03-0.49398746E-03
0.56690100E-04 0.44483598E-03-0.55169943E-03
0.62762832E-04 0.37231040E-03-0.59991935E-03
0.68076537E-04 0.29500504E-03-0.63800090E-03
0.72574665E-04 0.21441400E-03-0.66554314E-03
0.76210563E-04 0.13203872E-03-0.68238983E-03
0.78949917E-04 0.49343056E-04-0.68862061E-03
0.80770551E-04-0.32283933E-04-0.68453397E-03
0.81664824E-04-0.11156326E-03-0.67062164E-03
0.81640057E-04-0.18735419E-03-0.64753951E-03
0.80719503E-04-0.25867228E-03-0.61607338E-03
0.78942379E-04-0.32470445E-03-0.57710265E-03
0.76364144E-04-0.38481038E-03-0.53156400E-03
0.73054398E-04-0.43851952E-03-0.48041530E-03
0.69097863E-04-0.48552058E-03-0.42460416E-03
0.64591601E-04-0.52564265E-03-0.36504259E-03
0.59643236E-04-0.55883639E-03-0.30258019E-03
0.54369666E-04-0.58514485E-03-0.23799510E-03
0.48893940E-04-0.60468121E-03-0.17198172E-03
0.43342850E-04-0.61759888E-03-0.10515147E-03
0.37844424E-04-0.62407018E-03-0.38037688E-04
0.32525146E-04-0.62426412E-03 0.28893031E-04
0.27507049E-04-0.61832997E-03 0.95225419E-04
0.22905893E-04-0.60638785E-03 0.16057603E-03
0.18828272E-04-0.58852346E-03 0.22457245E-03
0.15370184E-04-0.56479033E-03 0.28683222E-03
0.12615109E-04-0.53521921E-03 0.34694397E-03
0.10632816E-04-0.49983081E-03 0.40445081E-03
0.94789611E-05-0.45865658E-03 0.45883795E-03
0.91942038E-05-0.41175820E-03 0.50952751E-03
0.98045675E-05-0.35925419E-03 0.55587874E-03

15 APPENDIX N - TEST OUTPUT LISTING FOR JD 243938.7500

```
NBODY TAPE WAS MOUNTED AT LOGICAL UNIT #  1
REQUIRED JD IS   2439381 FRACTION OF JD IS    0.7500000000
VALUE OF LER IS  2 KISS ARRAY IS   1   1   1   1   1   1   1   1   1   1   1   11

THE XPERT ARRAY: NBODY #  1
-0.38276393474125480 8D+00   0.24408056705293133 8D-02  0.40631688001812655 9D-01
-0.71851896314661231 3D-02  -0.23978564124537906 0D-01 -0.12102935929431979 7D-01
 0.19859801195595881 8D-02  -0.12661498830814697 5D-04 -0.21081744326955090 5D-03

THE XPERT ARRAY: NBODY #  2
-0.49262842840540190 0D+00   0.46440538737586900 5D+00  0.24038252784732842 8D+00
-0.14780780522747292 8D-01  -0.13096899557072907 3D-01 -0.49681355427815465 3D-02
 0.39312714428646911 7D-03  -0.37060215910896248 3D-03 -0.19182908047738040 5D-03

THE XPERT ARRAY: NBODY #  3
 0.98873574748416766 2D+00  -0.17153211263904596 9D+00 -0.74386495319394341 2D-01
 0.29160864743433022 9D-02   0.15453706900927370 2D-01  0.67013653828530590 8D-02
-0.28715443631526670 8D-03   0.49814516043815360 2D-04  0.21602393728852070 4D-04

THE XPERT ARRAY: NBODY #  4
-0.51669102121106434 7D+00   0.13844852916721368 5D+01  0.64936524364346215 3D+00
-0.12708087395044453 6D-01  -0.31249694174931121 4D-02 -0.10932393694710388 1D-02
 0.36352367512101704 3D-04  -0.97407285309669425 3D-04 -0.45687575966451651 2D-04

THE XPERT ARRAY: NBODY #  5
-0.16643599783335936 1D+01   0.45259668836477073 8D+01  0.19823289138748725 3D+01
-0.72406257904506742 0D-02  -0.19549855939925229 3D-02 -0.66171264430823128 9D-03
 0.34781877370332527 0D-05  -0.94593971308486147 6D-05 -0.41432087040849997 7D-05

THE XPERT ARRAY: NBODY #  6
 0.95321354505910995 7D+01  -0.49529298462754968 4D+00 -0.61682574966232610 3D+00
 0.96201677757979997 3D-04  -0.51354596094977710 3D-02  0.21191211127291823 9D-02
-0.32221515148318813 5D-05   0.15844616364741639 2D-06  0.20461368971159762 5D-06

THE XPERT ARRAY: NBODY #  7
-0.17994114417249495 1D+02   0.28745573487545277 3D+01  0.15137695664684023 3D+01
-0.73471210333601541 3D-03  -0.37179498041346919 0D-02 -0.16186432072530330 0D-02
 0.87500405997743230 0D-06  -0.14882342722751909 4D-06 -0.77601805043466901 6D-07

THE XPERT ARRAY: NBODY #  8
-0.18945537959642866 4D+02  -0.22072369257042904 4D+02 -0.85647312666213906 2D+01
 0.24243523609722371 9D-02  -0.17768146172018541 2D-02 -0.78891535708310072 3D-03
 0.20413186347388189 7D-06   0.22476038759650203 9D-06 -0.86657046625275275 8D-07

THE XPERT ARRAY: NBODY #  9
-0.30591764879249829 8D+02   0.26764585676428625 0D+01  0.10116923790801393 4D+02
-0.54014932762628720 6D-05  -0.31379084331283032 1D-02 -0.99149510206883689 6D-03
 0.27110527838709466 0D-06  -0.33183953336265287 5D-07 -0.93054205412203154 0D-07

THE XPERT ARRAY: NBODY # 10
```

-0.18408145887808599D-02 -0.13296406981113608D-02 0.80897765247084027D-03
-0.38611834274465191D-03 -0.45475875546605330D-03 -0.20191419620189688D-03
0.11696918386900432D-03 -0.85476252057609342D-04 -0.52059952194382008D-04

THE SOLAR SYSTEM BARYCENTRE ARRAY
-0.64869024184762509D-03 0.31615185282417701D-02 0.13438479206880702D-02
-0.68242968598239703D-05 -0.64510595891800355D-06 -0.13241545090633795D-06
0.28791123638421927D-08 -0.97702044439024437D-08 -0.43497870501656835D-08

NUTATION AND LIBRATION ARRAYS

1	-0.59444268E-04	0.31827120E-04	-0.27167588E-03	0.26300143E-01	0.52425005E-02
2	-0.97796601E-07	0.37475161E-06	0.0	0.0	0.0

A Real Time Display for Satellite Ranging

by

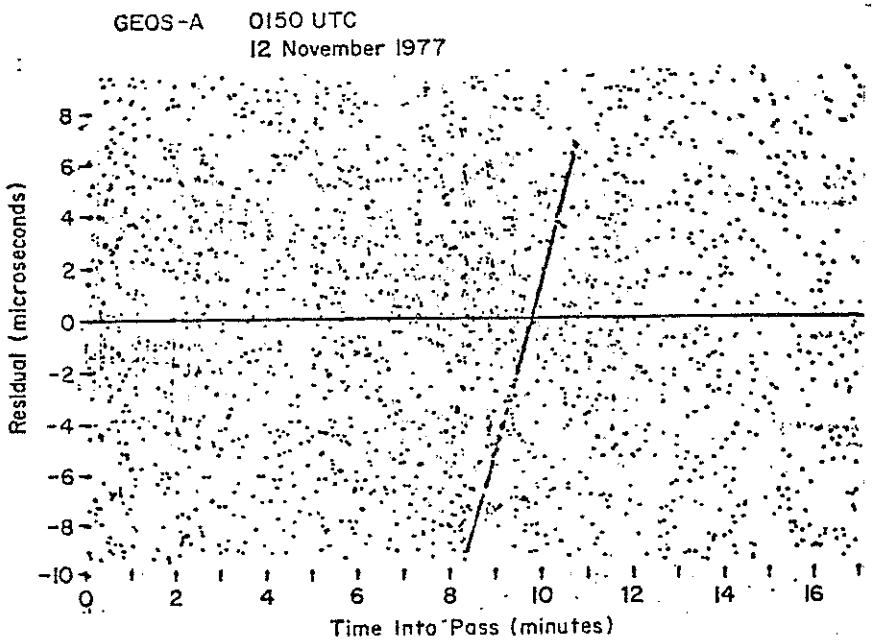
J. Rayner
Department of Physics and Astronomy
University of Maryland
College Park, Maryland

ABSTRACT

The low signal to noise ratio present in single photoelectron ranging prevents direct discrimination between satellite returns and noise. In order to provide a real time indication of success, the University of Maryland has developed a display program which plots residual range against time into the satellite pass. Since the satellite returns are correlated, they show up as a smooth curve which stands out from the random distribution of points due to noise. In practice this has provided very useful feedback for adjusting the pointing and time bias. A stand-alone version of the program is also available for viewing the data after a pass.

Working at the single photoelectron level implies a relatively poor signal to noise ratio. In general, there will be many noise events for each satellite return. This prevents the use of photomultiplier pulses as an indication of satellite returns. Fortunately, the relative stability of the real returns relative to the random distribution of noise events allows the real returns to be distinguished. The problem is then to display the data so that the real returns can be easily identified. We have taken several approaches to this problem in the various laser ranging timing systems that we have designed. The McDonald Lunar ranging system was limited to a teletype for output, but only fired every three seconds, allowing each residual range to be printed. All residual ranges lying within five nanoseconds of a previous residual were flagged and the teletype bell rung. Though simple, this system proved most useful and was quite well received. For the Maui ranging system, the computer video display was used to display a real time histogram of the residual ranges, allowing returns to be identified as a buildup in one or two bins. The parameters of the histogram could be changed in real time, allowing part of the histogram to be expanded once returns had been detected.

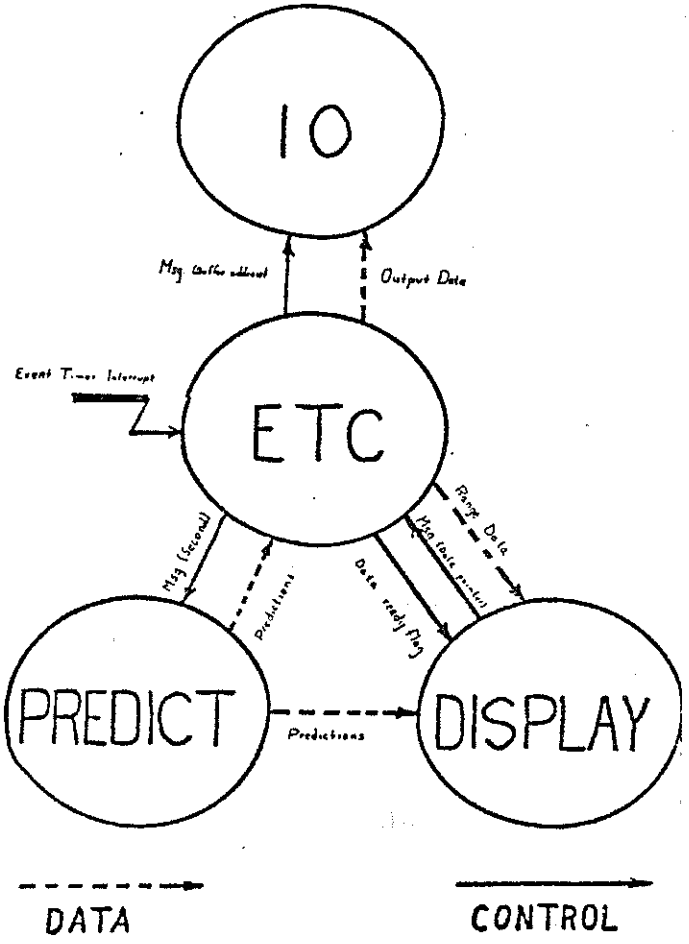
The high repetition rate and relative uncertainty of satellite predictions made a histogram less desirable for our present satellite system. A display of residual range versus time into the pass was chosen instead. As can be seen in the following illustration, satellite returns can be readily identified as a line against the scattered background points due to noise events.



The University of Maryland ranging system is distinguished by three features. We fire at the relatively high rate of 30 shots per second; we operate at the single photoelectron level; and we use an event timer rather than a time interval meter to time the returns. Our computer system consists of a NOVA 2/10 computer with 64 Kbytes of main memory, and 2.4 Mbytes of disk. It has a hardware multiply/divide unit but no hardware floating point. A

Tektronics 4013 graphics terminal is used for operator communication and graphic output, and a digi-data 9-track tape drive is used for bulk data transfer.

Although simple sequential processing is no longer possible, and must be replaced by multi-task processing, there are no significant problems handling 30 shots per second. The only critical point is assuring that the data output keeps up with data acquisition. The ranging program is structured as four cooperating tasks. In order of priority, they are: event timer control, data output, range prediction, and real time display. The various tasks are synchronized by the exchange of intertask messages and share data through common locations. With the exception of the event timer control and output tasks which are time critical, the program is written in FORTRAN.



The event timer control task is synchronized to the laser by an external interrupt from the event time. The start data from the event timer is then read. From this, the event timer control task calculates the expected return time using linear interpolation on predictions supplied by the range prediction task. There are two buffers for range prediction data. When the control task notes that the second has changed, it switches the buffer from which it takes its range predictions, and sends a message to the range prediction task. The prediction task then has a second to calculate the prediction for the following second and place it in the free buffer. The calculated return time is sent to the event timer, causing the range gate to be activated just before the expected return. Activation of the range gate causes another external interrupt that wakes up the control task which then reads the return data from the event timer and stores it in the output buffer. When the output buffer is full, the control task gets a new buffer and sends a message to the output task telling it to write out the old buffer. The output task then writes the buffer to disk and then returns it to the ready queue. Any time not used by the above tasks is available to the display task. When the display task is ready to process a new point it sets a flag which is tested by the control task. When it finishes processing a shot the control task checks the display flag; if set, it copies the data for the shot into a buffer accessible to the display task and sends a message to the display task. On receipt of this message, the display task picks up the data and calculates a new point for the real time display.

The ranging program described above is just one of a complex of programs which make up the Maryland satellite ranging system. These include a post processor which calculates ranges from the raw data and writes then to an IBM/360 compatible tape for later processing; a display program which can re-display the range data using various parameters and scales; and various clock setting and monitoring programs.

```
DISPLAY DGRID CD NXTR FIT4 DFLT2 RANGE PASS EVAL TRANS TDATA FILES ^
CHAR PBYTE PLOT.LB FORT.LB
C  THIS PROGRAM DISPLAYS THE RESIDUAL RANGES FOR A SATELLITE PASS
C    CALL LINE--DISPLAY <SATELLITE> [<STATION> [<DATA FILE>]]
C    IF <STATION> IS OMITTED 48INCH WILL BE USED
C    IF <DATA FILE> IS OMITTED RANGE$DATA WILL BE USED
C    GLOBAL
C      COEF--RANGE PREDICTION DATA (DESCRIBED IN EVAL)
C      PTATA--DISPLAY DATA (DESCRIBED IN DGRID)
C    LOCAL
C      DELTA--RANGE OFFSET(MICROSECONDS)
C      SCALE--VERTICAL SCALE FOR DISPLAY(MICROSEC/DIV)
C      TBIAS--TIME BIAS ON SATELLITE PREDICTION(SECONDS)
C      N--NUMBER OF ARGS ON CALL LINE
C      TJD--TIME(FRACTIONAL JULIAN DAY)
C      T--TIME(SECOND OF THE DAY)
C      IR1--RAW DATA(START TIME)
C      IR2--RAW DATA(STOP TIME)
C      TS--START TIME FOR PASS(SECOND OF THE DAY)
C      TE--END TIME FOR PASS(SECOND OF THE DAY)
C      T1--START TIME(FLOTING POINT)
C      T2--STOP TIME(FLOATING POINT)
```

```
C       MR--MEASURED RANGE(NANO SECONDS)
C       TLEAD--ONE WAY RANGE TO SATELLITE(SECONDS)
C       CR--CALCULATED RANGE(NONOSECONDS)
C       RR--RESIDUAL RANGE(NANOSECONDS)
        COMPILER DOUBLE PRECISION
        INTEGER IR1(7),IR2(11)
        REAL MR
        COMMON DUM1(4),DELTA
        COMMON /COEF/T0,N,COEF(10,3),TSCALE,TBIAS
        COMMON /PDATA/TS,TE


C     READ DELTA,SCALE,TBIAS
        ACCEPT 'DELTA(MICROSEC)=',DELTA
        ACCEPT 'SCALE(MICROSEC/DIV)=',SCALE
        ACCEPT 'TBIAS(SEC)=',TBIAS
C     GET DATA FROM CALL LINE AND ASSIGN FILES
        CALL FILES(N)
        IF(N.LE.0.OR.N.GT.3) STOP BAD CALL LINE
        IF(N.EQ.1) CALL FOPEN(2,'48INCH')
        IF(N.NE.3) CALL FOPEN(3,'RANGE$DATA')
C     GET FIRST POINT
        READ BINARY(3) IR1
        REWIND 3
C     CALCULATE FRACTIONAL JULIAN DAY AND SECOND OF THE DAY
        TJD=IR1(1)
        IF(TJD.LT.0.) TJD=TJD+65536.
        T=IR1(2)
        IF(T.LT.0.) T=T+65536.
        TJD=TJD/2.+T/86400.
        IF(MOD(IR1(1),2).EQ.0) T=T+43200.
C     INITALIZE PREDICTION ROUTINE
        CALL PASS(TJD)
        TS=86400.*DMOD(TS+.5,1.)
        TE=86400.*DMOD(TE+.5,1.)
C     DRAW AND LABEL AXIS FOR DISPLAY
        CALL DGRID(DELTA,SCALE,XS)
        SCALE=SCALE/.078
        TLEAD=.03
C     PLOT NEXT RESIDUAL RANGE
C       GET RAW DATA FOR NEXT POINT
3000    CALL NXTR(IR1,IR2,3,$4000)
C         CALCULATE TIME
          T=IR1(2)
          IF(IR1(2).LT.0) T=T+65536.
          IF(T.LT.T0) T=T+43200.
          IF(T.LT.TS) GO TO 3000
          IF(T.GT.TE) GO TO 4000
C         CALCULATE MEASURED RANGE
          T1=100.*DFLT2(IR1(3),IR1(4))-IR1(5)/20.
          T2=100.*DFLT2(IR2(3),IR2(4))-IR2(5)/20.
          T=T+T1/1D9-.6777215
          MR=T2-T1              ;MEASURED RANGE
          IF(MR.LT.0.) MR=MR+1D9
```

```
C          CALCULATE PREDICTED RANGE
           TLEAD=RANGE(T+TLEAD)
           CR=1000.*(2D6*TLEAD-DELTA)
C          CALCULATE RESIDUAL RANGE
           RR=MR-CR                ;RESIDUAL RANGE
C          PLOT RESIDUAL RANGE
           IX=(T-TS)/XS
           RR=RR/SCALE
           IF(RR.GT.0..AND.RR.LT.780.) CALL PNTABS(IX,IFIX(RR))
           GO TO 3000
C        END OF DATA FILE--CLOSE UP SHOP
4000       CONTINUE
           CALL MOVABS(0,0)
           CALL ANMODE
           TYPE
           END
C  DGRID DRAWS AND LABELS THE AXIS FOR THE REAL TIME DISPLAY
C    INPUT
C      DELTA--RANGE OFFSET(MICROSECONDS)
C      YS--VERTICAL SCALE
C    OUTPUT
C      XS--HORIZONTAL SCALE
C    GLOBAL
C      PDATA
C        TS--START TIME(MINIUTS)
C        TE--END TIME(MINUTS)
C    LOCAL
C      T--TIME OF DAY(SECONDS)
C      IT--INTEGER TIME OF DAY(SECONDS)
C
           COMPILER DOUBLE PRECISION
           SUBROUTINE DGRID(DELTA,YS,XS)
           COMMON /NFILE/NFILE(6,3)
           COMMON /PDATA/TS,TE
           INTEGER D,M,Y,DX

C        INITALIZE PLOT ROUTINES AND CLEAR SCREEN
           CALL INITT(100)
C        CALCULATE START OF PASS IN MINUTES AND SECONDS
           T=86400.*DMOD(TS+.5,1.)
           IH=T/3600.
           IM=DMOD(T,3600.)/60.
           IT=1440.*DMOD(TE+.5,1.)
           IT=IT-T/60.+1.
           IF(IT.LT.0) IT=IT+1440
C        CALCULATE HORIZONTIAL SCALE
           XS=60.*IT/1024.
           DX=60./XS
C        CALCULATE CIVIL DATE FROM JULIAN DAY
           CALL CD(D,M,Y,INT(TS-.5))
C        LABEL DISPLAY
           WRITE(10,101) NFILE(1,1),M,D,Y,IH,IM,YS
101          FORMAT(10X,S12,I2,2('/',I2),2X,I2,':',I2,
     1         '  SCALE=',F6.3,' MICROSEC/DIV')
```

```
        IY=78.*DELTA/YS
C     PLOT RESIDUAL RANGE EQUALS 0 LINE
        IF(IY.LT.0.OR.IY.GT.780) GO TO 300
          CALL MOVABS(0,IY)
          CALL DRWABS(1023,IY)
C       PLOT HORIZONTAL SCALE
300     DO 309 IX=0,1024,DX
          CALL MOVABS(IX,0)
          CALL DRWABS(IX,5)
309       CONTINUE
C       PLOT VERTICAL SCALE
        DO 319 IY=0,780,78
          CALL MOVABS(0,IY)
          CALL DRWABS(5,IY)
319       CONTINUE
        TS=T
        TE=TS+60*IT
        RETURN
        END
C  CD CALCULATES THE CIVIL DATE (DAY,MONTH,YEAR) FROM THE JULIAN DAY
C    INPUT
C      JD--MODIFIED JULIAN DAY
C    OUTPUT
C      DAY--DAY OF THE MONTH
C      MNTH--MONTH OF THE YEAR
C      YR--YEAR-1900
C
        SUBROUTINE CD(DAY,MNTH,YR,JD)
        INTEGER DAY,MNTH,YR,YR4,YR0,DAY0
        INTEGER DAYS
        COMMON /CDDATA/DAY0,YR0,DAYS(13)
        DATA DAYS/0,31,59,90,120,151,181,212,243,273,304,334,366/
        DATA DAY0/-144/,YR0/68/

        DAY=JD-DAY0
        YR4=DAY/1461
        DAY=MOD(DAY,1461)-1
        YR=DAY/365
        IF(DAY.EQ.-1) YR=0
        DAY=MOD(DAY,365)+1
        IF(YR.NE.0) GO TO 20
          IF(DAY.EQ.59) GO TO 40
          IF(DAY.LT.59) DAY=DAY+1
20      DO 29 MNTH=1,12
          IF(DAY-DAYS(MNTH+1).LE.0) GO TO 30
29      CONTINUE
        STOP
30      DAY=DAY-DAYS(MNTH)
        GO TO 45
40      MNTH=2
        DAY=29
45      YR=4*YR4+YR+YR0
        RETURN
        END
```

```
        SUBROUTINE NXTR(IR1,IR2,NFILE,EOF)
        INTEGER IR1(7),IR2(11),JUNK(4)
        COMMON /NXTRC/ICNT
        DATA ICNT/1/
        READ BINARY(NFILE,END=90)IR1,IR2
        ICNT=ICNT+1
        IF(ICNT.LE.14) RETURN
        ICNT=1
        READ BINARY(NFILE,END=90)JUNK
        RETURN
90      RETURN EOF
        END
;FUNCTION TO CONVERT TWO INTEGER ARGUMENTS TAKEN
;  AS A DOUBLE PRECISION WORD TO DOUBLE PRECISION
;  FLOATING POINT.
;    THE FIRST ARGUMENT MUST BE LESS THAN 256
        .TITL   DFLT2
        .ENT    DFLT2
        .EXTU
        .NREL
        3
DFLT2:  JSR     @.CPYL ;GET ARGUMENTS
        LDA     0,@.T+1,3 ;HIGH ORDER 8 BITS
        LDA     1,@.T+2,3 ;LOW 16 BITS
        LDA     2,EXP   ;EXPONENT=>2^24
        ADD     2,0     ;INSERT EXPONENT
        LDA     2,.T,3  ;LOC(RESULT)
        STA     0,0,2   ;RETURN RESULT
        STA     1,1,2
        SUB     1,1
        STA     1,2,2   ;FILL OUT DOUBLE PRECISION
        STA     1,3,2
        FRET            ;RETURN
EXP:    43000
.T=-167
        .END
C  RANGE RETURNS THE RANGE TO THE SATELLITE IN SECONDS
C
C    INPUTS
C      T--TIME OF DAY IN SECONDS
C    LOCAL
C      DELT--TIME INTO PASS IN SECONDS
C      P--POSITION VECTOR
         COMPILER DOUBLE PRECISION
         PARAMETER C=299792.5
         FUNCTION RANGE(T)
         COMMON /COEF/TO,N,COEF(10,3),TSCALE,TBIAS
         REAL P(3)

         DELT=T-TO
C     CALCULATE POSITION IN EARTH CENTERED INERTIAL FRAME
         CALL EVAL(P,DELT)
C     TRANSFORM TO STATION CENTERED ROTATING FRAME
         CALL TRANS(P,DELT)
```

```
C     CALCULATE RANGE
        RANGE=SQRT(P(1)*P(1)+P(2)*P(2)+P(3)*P(3))/C
C       AZ=57.2957795*ATAN2(P(1),P(2))
C       EL=57.2957795*ATAN2(P(3),SQRT(P(1)*P(1)+P(2)*P(2)))
        RETURN
        END
C  PASS INITALIZES THE VARIABLES USED IN THE SATALITE RANGE CALCULATION
    IT USES TEST DATA IF NO SATELLITE WAS SPECIFIED (N=0)
C
C  INPUT
C    JD--DATE IN FRACTIONAL JULIAN DAYS
C  GLOBAL
C    COEF--SATELLITE PASS DATA(DESCRIBED IN EVAL)
C    STATION--STATION POSITION INFORMATION(DESCRIBED IN TRANS)
C    PDATA--DISPLAY DATA(DESCRIBED IN DGRID)
C
        COMPILER DOUBLE PRECISION
        PARAMETER D1900=-24979.5
        SUBROUTINE PASS(T)
        COMMON T1
        COMMON/COEF/T0,N,COEF(10,3),TSCALE,TBIAS
        COMMON/STATION/SLON,SLAT,CLAT,DELV,DELN,LHA0
        COMMON /PDATA/TS,TE
        REAL LHA0
        INTEGER HR,MIN,SEC
        DATA TS/1310.8416666/,TE/1310.859027777/

        IF(N.EQ.0) GO TO 20
C         READ STATION LOCATION DATA
          READ BINARY(2) SLON,SLAT,CLAT,DELV,DELN
C         READ SATELLITE ORBIT DATA FOR NEXT PASS
10        IF(T.LT.TE) GO TO 20
            READ BINARY(1) TS,TE,T0,N,((COEF(I,J),I=1,N),J=1,3),TSCALE
            GO TO 10
C       MODIFY BEGINING AND ENDING TIMES FOR DISPLAY
20      ACCEPT 'TRIM(MIN) ',T1,T2
          TS=TS+T1/1440.
          TE=TE-T2/1440.
C       CALCULATE LOCAL HOUR ANGLE AT START OF THE PASSS
        ACCEPT 'DELPSI=',DELPSI
        HR=T0/3600.
        MIN=DMOD(T0/60.,60.)+.2
        SEC=DMOD(T0,60.)+.2
        WRITE(10,21) HR,MIN,SEC
21        FORMAT(' PASS STARTS AT ',I2,':',I2,':',I2)
        FCEN=(INT(TS-D1900)+.5)/36525.
        LHA0=DMOD(23925.836+FCEN*(8640184.542+.0929*FCEN)+DELPSI,86400.)
        LHA0=DMOD(LHA0+1.0027379093*T0-SLON,86400.)/13750.987083
        IF(T0.GE.86400.) T0=T0-86400.
        IF(N.NE.0) RETURN
          N=6
          T0=T1
          TYPE 'T0=',T0
          DT=TE-TS
```

```
          TS=T
          TE=TS+DT
        RETURN
        END
C  EVAL CALCULATES THE SATELLITE POSITION IN INERTIAL SPACE
C    FROM THREE N'TH ORDER POLYNOMIALS
C    INPUT
C      DELT--TIME INTO PASS IN SECONDS
C    OUTPUT
C      P--X Y Z POSITION IN EARTH CENTERED INERTIAL FRAME
C    GLOBAL
C      COEF
C        TO--START OF PASS(SECOND OF THE DAY)
C        N--ORDER OF PREDICTION POLYNOMIALS
C        COEF--COEFFICIENTS OF PREDICTION POLYNOMIALS
C        TSCALE--SCALING FACTOR FROM SECONDS FOR ARGUMENT
C          FOR POLYNOMIALS
C        TBIAS--ALONG TRACK BIAS IN SECONDS
C
        COMPILER DOUBLE PRECISION
        SUBROUTINE EVAL(P,DELT)
        COMMON /COEF/ TO,N,COEF(10,3),TSCALE,TBIAS
        REAL P(3)

        T=(DELT+TBIAS)/TSCALE
        DO 90 J=1,3
          POLY=COEF(1,J)
          DO 9 I=2,N
            POLY=POLY*T+COEF(I,J)
9           CONTINUE
          P(J)=POLY
90        CONTINUE
        RETURN
        END
C  TRANS TRANSLATES THE SATELLITE POSITION VECTOR
C    FROM A EARTH CENTERED INERTIAL FRAME
C    TO A STATION CENTERED ROTATING FRAME
C
C    INPUT
C      P--EARTH CENTERED XYZ POSITION VECTOR
C      DELT--TIME INTO PASS IN SECONDS
C    OUTPUT
C      P--STATION CENTERED ENV POSITION VECTOR
C    GLOBAL
C      STATION
C        SLON--STATION LONGITUDE
C        SLAT--SINE OF STATION LATITUDE
C        CLAT--COSINE OF STATION LATITUDE
C        DELV--VERTICAL DEVIATION
C        DELN--NORTH DEVIATION
C        LHAO--LOCAL HOUR ANGLE AT START OF PASS
C    LOCAL
C      LHA--LOCAL HOUR ANGEL
C      RPERS--EARTH ROTATION RATE(RADIANS/SEC)
```

```
C
        COMPILER DOUBLE PRECISION
        PARAMETER RPERS=7.292115854D-5
        SUBROUTINE TRANS(P,T)
        COMMON  /STATION/ SLON,SLAT,CLAT,DELV,DELN,LHAO
        REAL LHAO,LHA,P(3)

        LHA=RPERS*T+LHAO
        SLHA=SIN(LHA)
        CLHA=COS(LHA)
        XE1=P(1)*CLHA+P(2)*SLHA
        P(1)=-P(1)*SLHA+P(2)*CLHA
        P(2)=-XE1*SLAT+P(3)*CLAT-DELN
        P(3)=XE1*CLAT+P(3)*SLAT-DELV
        RETURN
        END
        COMPILER DOUBLE PRECISION
        BLOCK DATA
        COMMON /COEF/TO,N,COEF(10,3),TSCALE,TBIAS
        COMMON /STATION/SLON,SLAT,CLAT,DELV,DELN,LHAO
        REAL LHAO
        DATA N/0/COEF/
     1    -9.0192424D-6,-2.3252777D-4,6.4891695D-2,
     2    3.3866396D0,-1.7610474D2,-3.020634D3,4*0D0,
     3    -1.3092129D-6,1.5710605D-3,2.6791398D-2,
     4    -8.9414493D0,-1.0266486D2,8.0625833D3,4*0D0,
     5    1.525832D-5,-4.3730091D-4,-1.1845482D-1,
     6    -8.451897D-1,3.364123D2,7.0712201D2,4*0D0/
        DATA SLON,SLAT,CLAT/-67961.22933,.6295994624,.776919891/
        DATA DELV,DELN/6369.699554,-20.91386051/
        DATA TO/29520./
        DATA TBIAS,TSCALE/0.,60./
        END
C  FILES READS THE ARGUMENTS FROM THE PROGRAM CALL LINE
C    AND OPENS FILES 1...N TO THOSE FILENAMES
C    THE FILE NAMES ARE STORED IN NFILE
C    OUTPUT
C      N--THE NUMBER OF FILES
C    GLOBAL
C      NFILE
C        NFILE--THE FILE NAMES
C
        SUBROUTINE FILES(N)
        INTEGER CHAR
        COMMON /NFILE/NFILE(6,3)

        CALL FOPEN(0,'COM.CM')
        DO 99 I=1,12
          IF(CHAR(0,$300).EQ.0) GO TO 100
99        CONTINUE
100     DO 199 I=1,4
          JUNK=CHAR(0,$300)
199       CONTINUE
        N=0
```

```
        DO 299 J=1,3
          DO 209 I=1,12
            ICHAR=CHAR(0,$300)
            CALL PBYTE(ICHAR,NFILE(1,J),I)
            IF(ICHAR.EQ.0) GO TO 210
209         CONTINUE
210       N=N+1
          CALL FOPEN(N,NFILE(1,J))
          DO 219 I=1,4
            JUNK=CHAR(0,$300)
219         CONTINUE
299       CONTINUE
300     RETURN
        END
        .TITL   CHAR
        .ENT    CHAR
        .EXTU
        .NREL
.T=-167
.V=.T+200
        3
CHAR:   JSR     @.CPYL
        LDA     2,@.T+1,3 ;GET UNIT
        LDA     0,.IOCA ;GET CHANNEL TABLE
        MOVZL   2,2
        ADD     0,2
        LDA     2,0,2   ;GET CHANNEL
        LDA     0,BPTR
        SUBZL   1,1     ;GENERATE 1
        .SYSTM          ;READ CHARACTER
        .RDS    77
        JMP     ERR
        LDA     0,BUF
        MOVS    0,0
        STA     0,@.T,3         ;RETURN THE RESULT
        FRET
BPTR:   2*BUF
BUF:    0
ERR:    LDA     0,P6    ;HANDLE ABNORMAL RETURNS
        SUB     2,0,SZR ;EOF?
        JMP     ERR1    ;  NO
        JSR     @.AFRTN         ;  YES-TAKE EOF RETURN
        @.V+2
ERR1:   .SYSTM          ;  NO-ABORT PROGRAM
        .ERTN
        HALT            ;NEVER GET HERE
P6:     6
        .END

        .TITL   PBYTE
        .ENT    PBYTE
        .EXTU
        .NREL
.T=-167
```

```
        3
PBYTE:  JSR     @.CPYL
        LDA     1,@.T,3 ;GET DATA
        LDA     0,@.T+2,3 ;GET BYTE POSITION
        SUBZL   2,2     ;GENERATE ONE
        SUB     2,0     ;CHANGE TO 0 ORIGN
        LDA     2,MSK   ;=377
        AND     2,1     ;CLEAN UP INPUT BYTE
        MOVZR   0,0,SNC ;WHICH BYTE?
        MOVS    1,1,SKP ;  LEFT-FLIP INPUT
        MOVS    2,2     ;  RIGHT-FLIP MASK
        LDA     3,.T+1,3 ;GET BASE ADDRESS
        ADD     0,3     ;GET TARGET ADDRESS
        STA     3,TARG  ;SAVE IT
        LDA     3,0,3   ;GET TARGET WORD
        AND     2,3     ;CLEAR HALF
        ADD     1,3     ;INSERT NEW BYTE
        STA     3,@TARG         ;PUT IT BACK
        FRET
TARG:   0
MSK:    377
        .END
```

Orienting a Transportable Alt-Azimuth Telescope

by

Randall L. Ricklefs
Department of Astronomy
University of Texas
Austin, Tx 78712

ABSTRACT

The Transportable Laser Ranging System utilizes a telescope mount modeling system which allows tracking within 15 arcsec of a nominal satellite path from a remote, often poorly prepared site within a few hours of setup. The program allows the operator to select pointing targets, usually stars whose positions are available from a machine-readable star catalog, to point to these objects, and to record their predicted and observed positions. The same program can then perform a non-linear least-squares fit of the pointing data from up to 20 objects to obtain specified combinations of the 9 mount parameters. The mount model and its effectiveness are also discussed.

1 Introduction

The Transportable Laser Ranging System (TLRS) is designed to be driven to various sites and to be ready to range satellites within hours. This requires a quick method of determining the orientation of the telescope so that the satellite can be tracked accurately enough to obtain data.

2 Orientation Technique

The orientation process requires that one have a model for the behavior of the telescope under the influence of such effects as optical and mechanical misalignment, flexure, and encoder zero points as well as a method of acquiring pointing data and solving for the model parameters. Program ORIENT, diagramed in Figures 1-3, fills these needs.

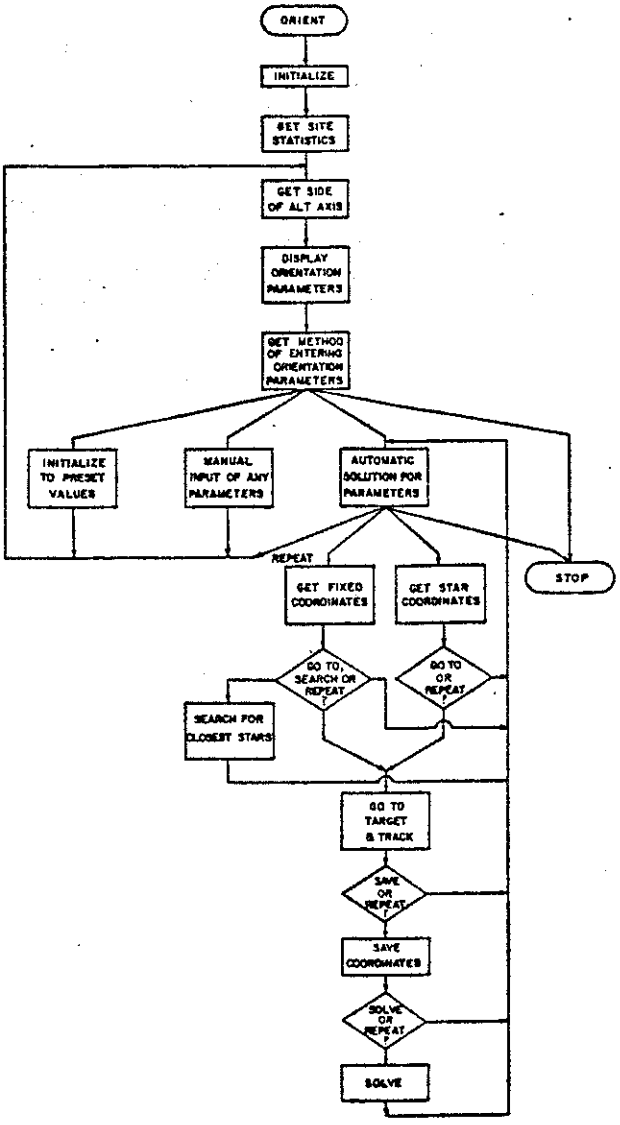


Figure 1: ORIENT Flow Diagram

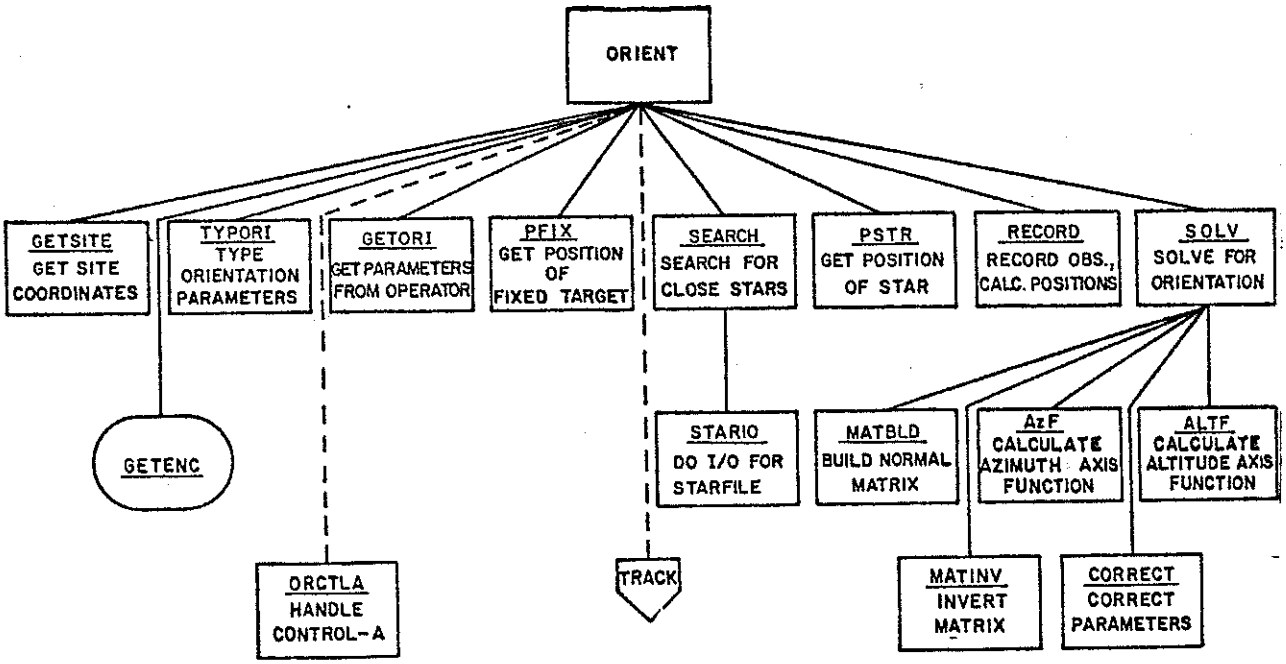


Figure 2: ORIENT Hierarchy Diagram

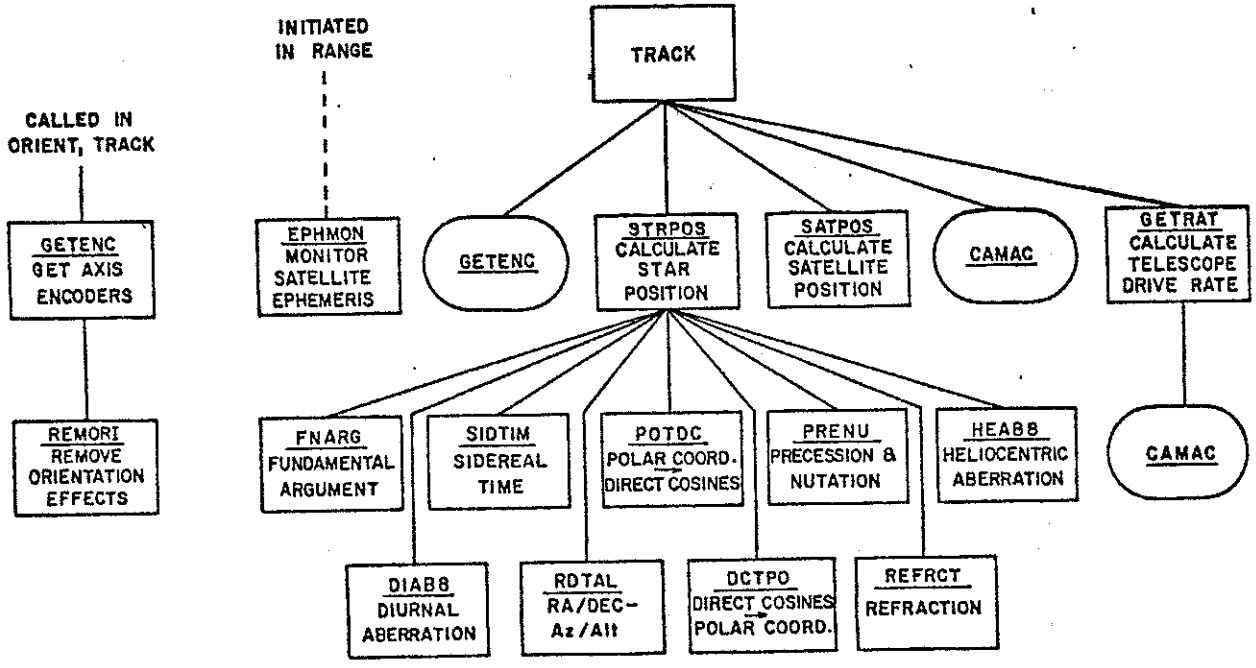


Figure 3: Tracking Subsystem Hierarchy Diagram

This interactive program queries the operator for site coordinates once at each site, for side of the altitude axis--experience has shown that one set of parameters will not perform adequately on both sides of the axis, and for method of model parameter input--manual entry or fit of observations. The operator can choose parts of the sky in which stars are to be observed i.e., everywhere for an initial global fit, and along the satellite path for fine

timing prior to a pass, and let the the computer scan its catalog to find stars near these positions.

The operator can command the computer to go to a particular star and wait until it arrives. The operator then centers the star on the fiducial on the CRT monitor displaying the telescope field, and automatically records the predicted and observed positions.

After recording the desired number of star positions, the operator can have the system solve for the desired mount parameters. During initial orientation at a site, the operator may spend quite a while finding the first star. When its position has been recorded, he solves for the axis encoder offsets, the 2 largest terms, and bootstrap his way to other stars, constantly solving for more parameters, until 10-20 stars have been observed and all the significant parameters have been fit.

3 Mount Model

The biggest difficulty in developing the system was determining the terms for the analytical mount model. The basic equations can be derived by following a beam of light down the telescope tube and mathematically rotating the beam through small errors in mirror and telescope misalignments as well as azimuth and altitude. If one cancels all cross terms as one goes, the problem becomes manageable and one arrives at the first-order effects of misalignment. One difficulty is that the cross terms may be significant for a very badly misaligned optical system. In fact, the whole advantage of having an analytical model is that misalignments can be spotted immediately from the results of a fit and corrected. For instance, if the tilt parameters are large (>.5 degree or so), the crew can adjust the tower to make that parameter reasonable.

It turns out that the theoretical model does not adequately describe our mount. The final TLRS mount model, shown in Figure 4, contains several empirical terms.
The sin(altitude) term is a major factor not determined analytically; its origin is still not known. The latch flexure is a small magnitude term which is due to the massive beam director deforming 3 small latches holding it to the tower. The encoder wobble terms are historical: a complaint on the poor altitude axis pointing arose after the alt-axis encoder backlash spring broke and wrapped itself around the shaft in such a way as to bend it. This does point out that a good model can account for almost any error!

To insure an adequate set of parameters, it has been necessary to fine-tune the fit prior to each pass by fitting stars along the satellite path. The values of the model parameters do change significantly from pass to pass. In fact, if one applies successive sets of parameters to the same predicted position, the total corrections vary by 3 to 4 arc minutes in both axes. This implies either that our model is missing some important terms, that there are significant variations in the tower and support surfaces due to thermal effects or settling, or that picking stars along a particular path biases the parameters quite significantly towards one part of the sky. Note that only

$$
\begin{aligned}
Az &= Az_I + X_1 \\
&+ (-X_3 \sin Az_I - X_4 \cos Az_I + \\
&\quad X_5 \cos 3Az_I + X_6) * \tan Alt_C \\
&+ X_7 \sec Alt_C
\end{aligned}
\qquad
\begin{aligned}
Alt &= Alt_I + X_2 \\
&- X_3 \cos Az_I + X_4 \sin Az_I \\
&+ X_8 \sin Alt_C \\
&+ X_9 \cos [32.8 (Alt_C - X_{10})]
\end{aligned}
$$

WHERE THE COEFFICIENTS REPRESENT:

Az, Alt	-	CORRECTED ANGLES,
Az_I, Alt_I	-	ANGLES READ FROM ENCODERS,
X_1, X_2	-	OFFSETS FROM ENCODER ZERO POINTS TO AZ AND ALT ZERO POINTS,
Alt_C	-	$Alt_I + X_2$, APPROXIMATE CORRECTED ALT,
X_3, X_4	-	TILTS OF TELESCOPE ALONG 2 PERPENDICULAR HORIZONTAL AXES,
X_5	-	FLEXURE IN 3 LATCHES ATTACHING BEAM DIRECTOR TO TOWER,
X_6	-	NON-ORTHOGONALITY OF AZ AND ALT AXES,
X_7	-	TRANSVERSE MISALIGNMENT IN ALT MIRROR,
X_8	-	EMPIRICAL TERM,
X_9	-	WOBBLE IN ALT ENCODER SHAFT, AND
X_{10}	-	PHASE OF THIS WOBBLE.

Figure 4: TLRS Mount Model

5-20 stars are used in any one of these fits.

4 MLRS Improvements

The McDonald Laser Ranging System (MLRS) contains hardware and software similar to that on TLRS. The software changes include use of the FK4 catalog (obtained on magnetic tape from USNO) for pointing instead of extracts from the American Ephemeris and Nautical Almanac's "Mean Places of Stars, 1978.0" (AENA, 1976). Also, the star position routines have been improved to include proper motion, more nutation terms, and a corrected diurnal aberration routine. The new star search algorithm allows picking the initial radius of search, initial faintest magnitude, and a preference for stars that are brightest or closest to the nominal position. The least squares routine has been generalized for all spherical coordinate systems and for a maximum of 20 model parameters and 200 observations.

An additional new feature is the "auto star" mode in which the program automatically and successively picks stars in a grid across the sky and points to them. When the computer-readable Reticon camera becomes operational, the operator will be able to instruct the program to "observe 100 stars" and leave hands off while the system does the rest.

5 Software Sets

The software sets accompanying this paper are the star position package and the non-linear least-squares package. The former was written mainly by Mark A. Powell and was, in part, installed, debugged, and enhanced by the author. Effects included are coordinate transformations, sidereal time (ASAENA,1961), fundamental arguments (APAENA VI, APAENA XV, Danby,1962), proper motion (Van de Kamp,1964), precession and nutation (ESAENA,1961), heliocentric and diurnal aberration (ESAENA,1961, Woolard and Clemence,1966), and refraction (Marini and Murray,1973). With the exception of the diurnal aberration, all effects are treated rigorously. The error in star positions due to approximations in this software is apparently less than 0.5 arcsec. STARDEM is the test driver which generates two corrected apparent star positions. Comparison coordinates are from Apparent Places of Fundamental Stars, 1980 (APFS ,1978).

The least squares package is tailored to a spherical coordinate system having longitudinal (RA,azimuth, or X) and latitudinal (Dec,azimuth, or Y) coordinates. It also allows use of more than one model to describe the mount. This is done with the MODEL variable which is usually recorded with each observation. This can be used in FUNC to select the proper model for the optical path the observation pertains to. One should refer to the paper by Jefferys (Jefferys,1980) for a thorough, general treatment of the method of least-squares. One can use any standard matrix inversion routine to handle the matrix of normal equations. The one we use is by Bevington (Bevington,1969). The TLRS mount model and a sample data set are used by LSQDEM for the test case.

6 Conclusion

The TLRS mount orientation program has been operating successfully in the field for almost two years. The goals for the program have been met: at several sites LAGEOS returns were obtained on the first pass attempted. On at least one occasion, this was within a few hours of arrival. There is more to be learned, however, about modeling the mount's performance. Time could be saved if a single global fit could be used for both front and back sides of the axis for periods of days or weeks. If the causes of the time-variation of the fits are thermal instability or settling, there may be little that can be done to improve the situation.

7 References

American Ephemeris and Nautical Almanac, 1978, Nautical Almanac Office, U.S. Naval Observatory, Washington, U.S. Government Printing Office, 1976.

Apparent Places of Fundamental Stars, 1980, Heidelberg, Astronomisches Rechen Institut, 1978.

Astronomical Papers of the American Ephemeris and Nautical Almanac VI, Nautical Almanac Office, U.S. Naval Observatory, Washington, U.S.

Government Printing Office.

Astronomical Papers of the American Ephemeris and Nautical Almanac XV, Nautical Almanac Office, U.S. Naval Observatory, Washington, U.S.Government Printing Office.

Bevington. P.R., Data Reduction and Error Analysis for the Physical Sciences, New York, McGraw-Hill, 1969.

Explanatory Supplement to the American Ephemeris and Nautical Almanac. London, Her Majesty's Stationery Office, 1961.

Danby, J.M., Fundamentals of Celestial Mechanics, New York, MacMillan, 1962.

Jefferys, W.H., Astron J. 85(2), February,1980.

Marini, J.W. and C.W. Murray, Correction of Laser Ranging Tracking Data for Atmospheric Refraction at Elevations Above 10 Degrees, NASA Technical Document X-591-73-3531 (1973).

Woolard, E.W., and G.M. Clemence, Spherical Astronomy, New York and London, Academic Press, 1966.

Van de Kamp, P., Elements of Astromechanics, San Francisco, W.H. Freeman, 1964.

8 APPENDIX A

```
        COMPILER DOUBLE PRECISION
C       ** STAR DEM **
C
C       MLRS/TLRS STAR PREDICTION SOFTWARE DEMONSTATION PROGRAM
C
C       R.L. RICKLEFS
C       UNIVERSITY OF TX, MCDONALD OBS.
C
C  REMEMBER TO LOAD BDMATH TO DEFINE COMMON /MATH/!
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$RLR 10/81
        INTEGER IC(8)
        REAL JD,LST
        COMMON /STCMN/ CATEP
        COMMON /SITE/ SITENO,MOVENO, LONG,GDLAT,LAT,HGT
        INTEGER SITENO
        REAL LONG, LAT, GDLAT, HGT
        COMMON /FUNAR/ ARG(12)
        COMMON /TARGET/ RA,DEC ,PMR,PMD,PMRD,PMDD,NSTAR
        COMMON /DIALOG/ IFILL0,IPTR,LINE(80),DW(2),DPNUM
        INTEGER IPTR,LINE,DW,SCAN
        COMMON /ENVPAR/ PRES, TEMP, FILL1(2)
        COMMON /MATH/ PIO2,PI,TWOPI,RDPDG,RDPHR,RDPAS,RDPTS
C
C  1950.0 IS DEFAULT CATALOG EPOCH
        DATA CATEP /2433282.423/
        DATA LONG,GDLAT,LAT,HGT  /4*0./
C*****************************************************************
C
C       INITIALIZATION
C
C*****************************************************************
C  OPEN OUTPUT FILE
        CALL OPEN(12,"STARSTAT",2,IERR)
        JD = 2444446.8
        INDX = 0
10      INDX = INDX+1
        GO TO (11,12,13), INDX
C  CANOPUS - FK4# 245
11      RA = 1.6704613902
        DEC = -.9192234655
        .290 * RDPTS
  PMD = 2.22 * RDPAS
  PMRD = .004 * RDPTS
  PMDD = -.04 * RDPAS
  IC(1) = 6
  IC(2) = 23
  IC(3) = 29
  IC(4) = 059
  IC(5) = -52
  IC(6) = 40
  IC(7) = 61
```

```
        IC(8) = 970
        GO TO 20
C  ETA UMA - FK4# 09
12      RA = 3.6022355123
        DEC = 0.8650245926
        PMR = -1.285 * RDPTS
        PMD =  -1.41 * RDPAS
        PMRD = .014 * RDPTS
        PMDD = -.08 * RDPAS
        IC(1) = 13
        IC(2) = 46
        IC(3) = 45
        IC(4) = 337
        IC(5) = 49
        IC(6) = 25
        IC(7) = 0
        IC(8) = 570
        GO TO 20
C  DONE
13      CALL FCLOS(12)
        STOP
C  SET UP FUNDAMENTAL ARGUMENTS FOR PROCESSING OF STAR POSITION
20      CALL FNARG(CATEP,JD)
C  WRITE THE RESULTS
        RLONG = LONG/RDPHR
        RLAT = LAT/RDPDG
        RGDLAT = GDLAT/RDPDG
        RRA = RA/RDPHR
        RDEC = DEC/RDPDG
        RPMR = PMR/RDPTS
        RPMD = PMD/RDPAS
        RPMRD = PMRD/RDPTS
        RPMDD = PMDD/RDPAS
        WRITE(12,1005) RLONG, RGDLAT, RLAT, TEMP,PRES, JD, ARG,
     X                 RRA,RDEC, RPMR,RPMRD, RPMD,RPMDD,CATEP
1005    FORMAT(" SITE:"/10X,"LONG:   ",F10.6 /
     X          10X,"GDLAT: ",F10.5 /10X,"LAT:   ",F10.5//
     X          " ENVIRONMENT:"/10X,"TEMP:  ",F5.0/10X,"PRES: ",F6.0//
     X          " TIME (JD): ",F15.6//
     X          " FUNDAMENTAL ARGUMENTS (RADIANS AND UNITLESS QUANTITIES):"/
     X          10X,"OBLIQUITY OF ECLIPTIC: ",10X,E20.10/
     X          10X,"PRECESS (ZETA): ",17X,E20.10/
     X          10X,"PRECESS (ZEE=RA OF NODE): ",7X,E20.10/
     X          10X,"PRECESS (THETA=INCL OF EQUATOR): ",E20.10/
     X          10X,"NUTATE (D PSI): ",17X,E20.10/
     X          10X,"NUTATE (D ETA): ",17X,E20.10/
     X          10X,"GEO X OF SUN: ",19X,E20.10/
     X          10X,"GEO Y OF SUN: ",19X,E20.10/
     X          10X,"GEO Z OF SUN: ",19X,E20.10/
     X          10X,"ECC OF SUN: ",21X,E20.10/
     X          10X,"TRUE ANOM OF SUN: ",15X,E20.10/
     X          10X,"GEOM LONG OF SUN: ",15X,E20.10//
     X          " STAR INFO:"/
     X          " (RA,DEC IN DEG,HOURS; CORRECTIONS IN SEC OF ARC,TIME) "/
```

```
     X          10X,"RA:  ",F11.7/10X,"DEC: ",F11.7/
     X          10X,"MU RA:  ",F9.3, 10X,"MU DOT RA:  ",F9.3/
     X          10X,"MU DEC: ",F9.3, 10X,"MU DOT DEC: ",F9.3//
     X          10X,"CATALOG EPOCH (JD): "F15.6)

C  DETERMINE SIDEREAL TIME
        CALL SIDTIM(JD, LONG, LST)
        RLST = LST/RDPHR
        WRITE(12,1010) RLST
1010    FORMAT(/" LOCAL SIDEREAL TIME (HOURS): ",F11.7)
C  CORRECT FOR PROPER MOTION
        CALL PRMTN(RA,DEC, PMR,PMD,PMRD,PMDD, (JD-CATEP)/36534.22,PRA,PDEC)
        IF (PRA.LT.0.) PRA = PRA+TWOPI
        RRA=PRA/RDPHR
        RDEC=PDEC/RDPDG
        DRA=(PRA-RA)/RDPTS
        DDEC=(PDEC-DEC)/RDPAS
        WRITE(12,1012) RRA,RDEC,DRA,DDEC
1012    FORMAT(/" RESULTS OF PROPER MOTION:"/
     X          5X,"NEW RA,DEC: ",F11.7,2X,F10.6/5X," DRA,DDEC:",F9.2,5X,F8.2)
C  CONVERT RA, DEC TO DIRECTION COSINES
        CALL POTDC(1, PRA,PDEC, X,Y,Z)
C  PRECESS & NUTATE
        CALL PRENU(X,Y,Z,XP,YP,ZP)
        CALL DCTPO(1,XP,YP,ZP,TRA,TDEC)
        IF (TRA.LT.0.) TRA = TRA+TWOPI
        RRA = TRA/RDPHR
        RDEC = TDEC/RDPDG
        DRA = (TRA-PRA)/RDPTS
        DDEC = (TDEC-PDEC)/RDPAS
        WRITE(12,1015) RRA,RDEC, DRA,DDEC
1015    FORMAT(/" RESULTS OF PRECESSION & NUTATION: "/
     X          5X,"NEW RA,DEC: ",F11.7,2X,F10.6/5X,"DRA,DDEC: ",F8.2,5X,F8.2)
C  APPLY HELIOCENTRIC ABERRATION
        CALL HEAB8(XP,YP,ZP, 1 ,X,Y,Z)
        CALL DCTPO(1,X,Y,Z,TPRA,TPDEC)
        IF (TPRA.LT.0.) TPRA = TPRA+TWOPI
        RRA = TPRA/RDPHR
        RDEC = TPDEC/RDPDG
        DRA = (TPRA-TRA)/RDPTS
        DDEC = (TPDEC-TDEC)/RDPAS
        WRITE(12,1020) RRA,RDEC, DRA,DDEC
1020    FORMAT(/" RESULTS OF HELIOCENTRIC ABERRATION:"/
     X          5X,"NEW RA,DEC: ",F11.7,2X,F10.6/5X,"DRA,DDEC: ",F8.2,5X,F8.2)
C  CORRECT FOR DIURNAL ABERRATION
        CALL DIABR(X,Y,Z, LST,LAT, XD,YD,ZD)
        CALL DCTPO(1, XD,YD,ZD, TRA,TDEC)
        IF (TRA.LT.0.) TRA = TRA+TWOPI
        RRA = TRA/RDPHR
        RDEC= TDEC/RDPDG
        DRA = (TRA-TPRA)/RDPTS
        DDEC = (TDEC-TPDEC)/RDPAS
        WRITE(12,1025) RRA,RDEC,DRA,DDEC
1025    FORMAT(/" RESULTS OF DIURNAL ABERRATION:"/
```

```
     X          5X,"NEW RA,DEC: ",F11.7,2X,F10.6/5X,"DRA,DDEC: ",F8.2,5X,F8.2)
C  CONVERT TO HH/DD MM SS
        CALL RDTHR(TRA,IH,IM,TSEC)
        CALL RDTDG(TDEC,ID,IMN,ASEC)
        WRITE(12,1027) IH,IM,TSEC,ID,IMN,ASEC
1027    FORMAT(5X,"CON RA,DEC: ",2(2I3,F7.3,5X))
        WRITE(12,1026) IC
1026    FORMAT(/" COMPARISON POSITION (W/O DIURNAL ABERRATION):"/
     X          17X,2(3I3,".",I3,5X))
C  CONVERT RA/DEC TO AZ/ALT
        CALL RDTAL(XD,YD,ZD, GDLAT, LST, XP,YP,ZP)
C  CONVERT AZ/ALT DIRECTIONS TO AZ/ALT
        CALL DCTPO(1, XP,YP,ZP, AZ,ALT)
C  CALCULATE REFRACTION (2ND & 4TH PARAMETERS ARE DUMMY RANGE)
        CALL REFRCT(ALT,0., FALT,DUMMY)
        DALT = (FALT-ALT)/RDPAS
        RAZ = AZ/RDPDG
        RALT = ALT/RDPDG
        RFALT = FALT/RDPDG
        WRITE(12,1030) RAZ,RALT,RFALT,DALT
1030    FORMAT(/" RESULTS OF REFRACTION:"/
     X          5X,"AZ,ALT: ",2F11.6/
     X          5X,"REFRACTED ALT: ",F11.6/5X,"DALT: ",F8.2//)
        GO TO 10
        END



        COMPILER DOUBLE PRECISION
        SUBROUTINE SIDTIM(JD, LONG, LST)
C  CALCULATE LOCAL APPARENT SIDEREAL TIME (LST)
C  AT MODIFIED JULIAN DATE (MJD)+UNIVERSAL TIME (UT)
C  AT THE SITE WITH LONGITUDE LONG.
C  THE METHOD IS DOCUMENTED ON PAGE 84 OF THE EXPLANATORY SUPPLEMENT
C  (ESAENA).
C
C       ** FNARG MUST BE CALLED BEFORE SIDTIM **
C
C               INPUT:  JD      - JULIAN DATE (DECIMAL DAYS)
C                       LONG    - EAST LONGITUDE OF SITE(RADIANS)
C
C               << /FUNAR/ >>
C                       DPSI    - DELTA PSI, NUTATION IN LONGITUDE AT
C                                       TIME JD.
C                       EPSLN   - OBLIQUITY OF ECLIPTIC AT TIME JD
C
C               << /SIDCOM/ >>
C                       A,B,C   - COEFFICIENTS FOR SIDEREAL TIME CALCS.
C                       STEPOK  - STARTING EPOCH OF SID. TIM SERIES
C                                       (JAN 0.5, 1900)
C
C               OUTPUT: LST     - LOCAL SIDEREAL TIME (RADIANS)
C
C               COMMONS: SIDCOM
```

```
C                        MATH   -- COMMONLY USED MATH CONSTANTS
C                                                                    RLR
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
      COMMON /FUNAR/ EPSLN,ZETA,ZEE,THETA,DPSI,DETA,
     1          XS,YS,ZS,ESUN,TSAN,GELS
C
      COMMON /SIDCOM/ A, B, C, STEPOK
C
      COMMON /MATH/ PIO2, PI, TWOPI, RDPDG, RDPHR, FILL(2)
      REAL JD, LONG, LST
C
C  COEFFICIENTS FOR SIDEREAL TIME (A IN HOURS, B&C IN SEC OF TIME)
      DATA A, B, C/6.646065556, 8640184.542, 0.0929/
C
C  START EPOCH OF SIDEREAL TIME SERIES (JAN 0.5, 1900)
      DATA STEPOK /2415020.0/
C
      TU = (JD - STEPOK)/36525.
C
C  EVALUATE SERIES FOR LMST AT GREENWICH
      GST = AMOD(A + (B*TU + C*TU**2.)/3600., 24.) * RDPHR
C
C  MODIFY RESULT FOR FRACTIONAL DAY, EAST LONGITUDE OF SITE,
C  AND EQUATION OF EQUINOXES. THE LATTER CONVERTS RESULT FROM
C  MEAN TO APPARENT SIDEREAL TIME.
      UTF = AMOD(JD-INT(JD)+.5, 1.)
      LST = UTF*TWOPI + GST + LONG + DPSI*COS(EPSLN)
      LST = AMOD(LST, TWOPI)
C
      RETURN
      END




      COMPILER DOUBLE PRECISION
      SUBROUTINE REFRCT(ALT,RANGE, ALTR,RANGER)
C
C  REFRACT SATELLITE ALTITUDE AND RANGE.
C  ALGORITHMS TAKEN FROM GODDARD LETTER OF 10/20/78,"LASER SOFTWARE NOTES"
C   FROM J. FRISKEY.
C  ORIGINALLY FROM "CORRECTION OF LASER RANGING TRACKING DATA FOR
C  ATMOSPHERIC REFRACTION AT ELEVATIONS ABOVE 10 DEGREES",
C  A NASA TECH DOCUMENT BY J.W. MARINI, &C.W. MURRAY,JR.
C
C      INPUT:  ALT     -- ALTITUDE IN RADIANS
C              RANGE   -- RANGE IN NSEC (TWO WAY)
C
C      << /ENVPAR/ >>
C              BARO    -- BAROMETRIC PRESSURE IN MILLIBARS
C              TEMP    -- ATMOSPHERIC TEMPERATURE IN DEGREES KELVIN
C              HUMID   -- ATMOSPHERIC HUMIDITY IN %. SMALL EFFECT ON
C                         REFRACTION, HENCE LEFT AT 50%.
```

```
C               LAMBDA  -- OBSERVING WAVELENGTH IN MICRONS (10000 ANG)
C
C       OUTPUT: ALTR    -- ALTITUDE + REFRCTION IN ALTITUDE
C               RANGER  -- RANGE + REFRACTION IN RANGE
C
C       COMMON: ENVPAR  -- ENVIRONMENTAL PARAMETERS
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
        REAL NS, Q, E, LAMBDA,LAMBD2
        COMMON /ENVPAR/ BARO, TEMP, HUMID, LAMBDA
        DATA BARO, TEMP, HUMID, LAMBDA /1000., 293., 50., .5320/
C
C  CALCULATE WAVELENGTH DEPENDENCE
        LAMBD2 = LAMBDA*LAMBDA
        F = 0.965 + (0.0164+0.000228/LAMBD2)/LAMBD2
CX      F = 1.
C
        SINALT = SIN(ALT)
        Q = 1./(SINALT + SQRT(SINALT*SINALT + 0.0025284))
C
C  PARTIAL PRESSURE OF WATER VAPOR
C
        DTEMP = TEMP - 273.15
        E = 0.0611*HUMID*10.**(7.5*DTEMP/(237.3 + DTEMP))
C
C  SURFACE REFRACTIVITY
        NS = 80.343*F*(BARO-0.140647*E)/TEMP
C
C  CORRECTION FOR ALTITUDE (RADIANS)
        DALT = Q*COS(ALT)*(6.01161E-4 + 1.92056E-6*(NS - 313.))
C
C  CORRECTION FOR RANGE (NSEC)
C    WHERE THE FIRST TERM CONTAINS THE NSEC/M CONVERSION (3.335640485)
C    ...BUT FIRST, BE SURE SINALT DOESNT CAUSE PROBLEMS AT ALT = 0DEG
        IF (SINALT.EQ.0) SINALT = 1.0E-6
        DRANGE = .0078621046*F*BARO/SINALT
C
X       WRITE(12,1111) ALT,DALT,RANGE,DRANGE
X1111   FORMAT(10X,4E20.10)
C  CALCULATE CORRECTED VALUES
C
        ALTR = ALT + DALT
        RANGER = RANGE + DRANGE
C
C
        RETURN
        END
        COMPILER DOUBLE PRECISION



        SUBROUTINE FNARG(FEPOK,EPOCH)

C  FNARG CALCULATES THE FUNDAMENTAL ARGUMENT FOR PRECESSION,
```

```
C  NUTATION, HELIOCENTRIC ABERRATION, AND HELIOCENTRIC
C  PARALLAX AND PLACES THEM IN A COMMON STORAGE BLOCK FINAR
C
C           INPUT: FEPOK - FIXED EPOCH IN MODIFIED JULIAN DATE
C                  EPOCH - EPOCH OF OBSERVATION
C
C           OUTPUT: NONE
C
C           COMMON BLOCKS: FUNAR
C
C           CALLS:  RVTRD
C                   MD2PI
C                   POTDC
C                   ECTRD
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

      COMMON /FUNAR/ EPSLN,ZETA,ZEE,THETA,DPSI,DETA,
     1          XS,YS,ZS,ESUN,TSAN,GELS
      COMMON /FNTEMP/ EPS(4),ZET(4),ZE,THET(4),LL(5),LLP(5),FF(5),DD(5),
     1          OM(5),PSI(24),ETA(8),AMN(5),PRTS(9),PRTC(9),SML(4),ESN(3),
     2          EQC(4),RDLS(8),SNLT,EQLN,GP1(4),GP3(4),GP4(3),
     3          EPO,CNTRY,EPTO,TCNTRY
      REAL LL,LLP,L,LP, ARGNU(12)
C     MEAN OBLIQUITY OF THE ECLIPTIC OF DATE-ES-AENA P.98
      DATA EPS / 0.4093197474, -2.271109689E-4,
     1          -2.862339973E-8, 8.779006138E-9 /
C  PRECESSION QUANTITIES - ES-AENA PP. 29-30
C  PI/2 - ZETA = RA OF ASCENDING NODE OF EQUATOR OF EPOCH T
C  RECKONED FROM EQUINOX OF TO
      DATA ZET / 1.117131925E-2, 6.767998988E-6,
     1          1.464137317E-6, 8.72664626E-8 /
C  PI/2 + ZEE = RA OF NODE RECKONED FROM EQUINOX OF T
      DATA ZE / 3.834876218E-6 /
C  THETA = INCLINATION OF EQUATOR OF T TO EQUATOR OF TO
      DATA THET / 9.718972599E-3, -4.1354607E-6,
     1          -2.065306282E-6, -2.036217461E-7 /
C  NUTATION QUANTITIES - ES-AENA P.44
C  L
      DATA LL /5.168000346, 1325., 3.470571655,
     X          1.6042E-4, 2.511E-7/
C  LP (L')
      DATA LLP /6.25658358, 99., 6.266600316,
     X          -2.62E-6,  5.818E-8/
C  F=MEAN ARGUMENT OF LATITUDE OF MOON AT EPOCH
      DATA FF / 1.963650549E-1, 1342., 1.431608937,
     1          -5.604446154E-5, -5.817764173E-9 /
C  D = MEAN ELONGATION OF MOON FROM SUN AT EPOCH
      DATA DD / 6.121523943, 1236., 5.360154261,
     1          -2.506486731E-5, 3.296733032E-8 /
C  OMEGA = LONGITUDE OF MEAN ASCENDING NODE OF MOON AT EPOCH
      DATA OM / 4.523601515, -5., -2.341219711,
     1          3.626406335E-5, 3.878509449E-8 /
C  DPSI = NUTATION IN LONGITUDE AT EPOCH
```

```
      DATA PSI / -8.354649E-5, -8.4212E-8,  1.0123E-6,   9.7E-11,
     X           -6.17119E-6,  -6.30E-10,   6.1135E-7,  -1.50E-9,
     X           -2.410E-7,     5.82E-10,   1.038E-7,   -2.4E-10,
     X            6.012E-8,     4.8E-11,   -9.8757E-7,  -9.7E-11,
     X            3.272E-7,     4.8E-11,   -1.658E-7,   -1.9E-10,
     X           -1.265E-7,     0.,        -7.224E-8,    0./
C  DETA = NUTATION IN OBLIQUITY AT EPOCH
      DATA ETA / 4.46513E-5, 4.41E-9, -4.383E-7,  1.9E-10,
     1           2.6771E-6, -1.41E-9,  4.286E-7, -2.4E-10/
C  DATA FOR COMPUTATION OF THE TRUE ANOMALY AND
C  TRUE GEOCENTRIC LONGITUDE AND POSITION OF THE SUN
C  AMAN = AVERAGE MEAN ANOMALY. P.60 APAENA XV
      DATA AMN / 6.25658358, 99., 6.266600316,
     1          -2.617993878E-6, -5.817764173E-8 /
C  PERTS = PERTURBATION ON SUN > 1ARCSEC. P. 64 APAENA XV SINES
      DATA PRTS / 2.5E-5, -2.E-5, 1.4E-5, -8.E-6,
     1          7.E-6, 7.883070455E-2, -1.3E-5, -1.3E-5, -7.E-6 /
C  PERTC = PERTURBATION ON SUN > 1 ARCSEC. P.64 APAENA XV COSINE
C  TERMS.
      DATA PRTC /1.1E-5, -2.3E-5, 9.E-6, 7.E-6,
     1  6.E-6, 1.8E-5, 1.E-5, -8.E-6, -3.5E-5/
C  SUNML = SUNS MEAN LONGITUDE . APAENA XV, P.59.
      DATA SML /4.881627934, 100., 1.342027295E-2,
     1  5.279620987E-6/
C  EQNCN = EQUATION OF CENTER FROM DANBY P.337
      DATA EQC /2., -.25, 1.25, 1.0833333333/
C  RADSN = RADIUS VECTOR OF THE SUN APAENA XV P 66.
      DATA RDLS /3.057E-5, -7.27412E-3, -9.138E-5,
     1  3.593E-6, 4.021E-6, 1.814E-5, 5.822E-6, -7.067E-6/
C  SNLAT = LATITUDE OF THE SUN APAENA XV P.65
      DATA SNLT /2.792526803E-6/
C  EQLUN = LUNAR INEQUALITY IN LONGITUDE. APAENA VI, P.18.
      DATA EQLN /3.128987498E-5/
C  GPP1 = MEAN ANOMALY OF VENUS. APAENA XV P.64.
      DATA GP1 /3.710626228, 162., 3.452328892,
     1  2.244687344E-5/
C  ESUN = ECCENTRICITY OF SUNS ORBIT APAENA VI, P.9.
      DATA ESN /1.675104E-2, -4.18E-5, -1.26E-7/
C  GPP3 = MEAN ANOMALY OF MARS. APAENA XV, P.64.
      DATA GP3 /5.576840378, 53., 1.04472791,
     1  3.156137064E-6/
C  GPP4 = MEAN ANONALY OF JUPITER APAENA XV, P.64.
      DATA GP4 /3.93288906, 8., 2.699885163/
C  INITIAL EPOCH: JAN 0.5 ET, 1900, JD2415020.0
      DATA EP0 /2415020.0/
C  INITIAL EPOCH FOR TROPICAL CENTURY=1900.0 = 2415020.313
      DATA EPT0 /2415020.313/
C
C  CENTURY = 36525 JULIAN(OR EPHEMERIS) DAYS.
      DATA CNTRY /36525./
C  TROPICAL CENTURY = 36524.22 DAYS
      DATA TCNTRY /36524.22/
C-----------------------------------------------------------------------
C  FIXED EPOCH IN CENTURIES SINCE EP0.
```

```
      TO = (FEPOK-EPTO)/TCNTRY
      TT = (EPOCH-EPTO)/TCNTRY
      TT2 = TT*TT
      TT3 = TT2*TT
C
C  TOTAL CENTURIES SINCE EPO.
      T = (EPOCH-EPO)/CNTRY
      T2 = T*T
      T3 = T2*T
C
C  TOTAL CENTURIES SINCE FEPOK (FOR USE IN PRECESSION CALCULATIONS)
      TP = (EPOCH-FEPOK)/TCNTRY
      TP2 = TP*TP
      TP3 = TP2*TP
C
C  COMPUTE OBLIQUITY OF ECLIPTIC
      EPSLN = EPS(1) + EPS(2)*TT + EPS(3)*TT2 + EPS(4)*TT3
C
C  COMPUTE ZETA
      ZETA = (ZET(1) + ZET(2)*TO)*TP + ZET(3)*TP2 + ZET(4)*TP3
C  COMPUTE ZEE
      ZEE = ZETA + ZE*TP2
C  COMPUTE THETA
      THETA = (THET(1) + THET(2)*TO)*TP + THET(3)*TP2 + THET(4)*TP3
C  COMPUTE F,D,OMEGA,AMAN,SUNML,GPP1,GPP3,GPP4 - ALL ANGULAR
C  VARIABLES.
      L = LL(1) + RVTRD(LL(2),T) + LL(3)*T + LL(4)*T2 + LL(5)*T3
      CALL MD2PI(L)
      LP= LLP(1) + RVTRD(LLP(2),T) + LLP(3)*T + LLP(4)*T2 + LLP(5)*T3
      CALL MD2PI(LP)
      F = FF(1) + RVTRD(FF(2),T) + FF(3)*T + FF(4)*T2 + FF(5)*T3
      CALL MD2PI(F)
      D = DD(1) + RVTRD(DD(2),T) + DD(3)*T + DD(4)*T2 + DD(5)*T3
      CALL MD2PI(D)
      OMEGA = OM(1) + RVTRD(OM(2),T) + OM(3)*T + OM(4)*T2 + OM(5)*T3
      CALL MD2PI(OMEGA)
      AMAN = AMN(1) + RVTRD(AMN(2),T) + AMN(3)*T + AMN(4)*T2 +
     1   AMN(5)*T3
      CALL MD2PI(AMAN)
      SUNML = SML(1) + RVTRD(SML(2),T) + SML(3)*T + SML(4)*T2
      CALL MD2PI(SUNML)
      GPP1 = GP1(1) + RVTRD(GP1(2),T) + GP1(3)*T + GP1(4)*T2
      CALL MD2PI(GPP1)
      GPP3 = GP3(1) + RVTRD(GP3(2),T) + GP3(3)*T + GP3(4)*T2
      CALL MD2PI(GPP3)
      GPP4 = GP4(1) + RVTRD(GP4(2),T) + GP4(3)*T
      CALL MD2PI(GPP4)
C  COMPUTE DPSI, DETA
      ARGNU(1)  = OMEGA
      ARGNU(2)  = 2.*OMEGA
      ARGNU(3)  = 2.*(F-D+OMEGA)
      ARGNU(4)  = LP
      ARGNU(5)  = LP + ARGNU(3)
      ARGNU(6)  =-LP + ARGNU(3)
```

```
      ARGNU(7)  = ARGNU(3) - OMEGA
      ARGNU(8)  = 2.*(F+OMEGA)
      ARGNU(9)  = L
      ARGNU(10) = 2.*F + OMEGA
      ARGNU(11) = L + ARGNU(8)
      ARGNU(12) = L - 2.*D
      DPSI = 0.
      DO 5 J=1,12
          K = (J-1)*2+1
          DPSI = DPSI + (PSI(K) + PSI(K+1)*T)*SIN(ARGNU(J))
5     CONTINUE
      DETA = (ETA(1) + ETA(2)*T)*COS(ARGNU(1))
     X       + (ETA(3) + ETA(4)*T)*COS(ARGNU(2))
     X       + (ETA(5) + ETA(6)*T)*COS(ARGNU(3))
     X       + (ETA(7) + ETA(8)*T)*COS(ARGNU(8))
C  CORRECT OBLIQUITY OF ECLIPTIC FOR NUTATION
      EPSLN = EPSLN+DETA
C  COMPUTE GEOMETRIC LONGITUDE OF SUN
C  1) COMPUTE SIN PERTURBATIONS TO LONGITUDE
      SINP = PRTS(1)*SIN(4.*AMAN - 8.*GPP3 + 3.*GPP4)
     1  + PRTS(2)*SIN(AMAN-GPP1) + PRTS(3)*SIN(2.*AMAN-2.*GPP1)
     2  + PRTS(4)*SIN(3.*AMAN-2.*GPP1) + PRTS(5)*SIN(13.*AMAN
     3  - 8.*GPP1+PRTS(6)*T) + PRTS(7)*SIN(GPP4)
     4  + PRTS(8)*SIN(2.*AMAN-2.*GPP4) + PRTS(9)*SIN(AMAN-GPP4)
C  2) COMPUTE COS PERTURBATIONS TO LONGITUDE
      COSP = PRTC(1)*COS(AMAN-GPP1) + PRTC(2)*COS(2.*AMAN-2.*GPP1)
     1  + PRTC(3)*COS(3.*AMAN-2.*GPP1) + PRTC(4)*COS(4.*AMAN-3.*GPP1)
     2  + PRTC(5)*COS(13.*AMAN-8.*GPP1+PRTS(6)*T)
     3  + PRTS(6)*COS(4.*AMAN-8.*GPP3+3.*GPP4)
     4  + PRTC(7)*COS(2.*AMAN-2.*GPP3) + PRTC(8)*COS(AMAN-2.*GPP3)
     5  + PRTC(9)*COS(AMAN-GPP4)
C  3) COMPUTE ECCENTRICITY OF SUN
      ESUN = ESN(1) + ESN(2)*T + ESN(3)*T2
C  4) COMPUTE LUNAR INEQUALITY IN LONGITUDE
      EQLUN = EQLN*SIN(D)
C  5) COMPUTE MEAN ANOMALY W/ PERTURBATION & NUTATION
      TAMAN = AMAN + EQLUN + DPSI
      CALL MD2PI(TAMAN)
C
C     COMPUTEUTE THE GEOMETRIC LONGITUDE W/ PERTURBATION
      GELS = SUNML + EQNCN + EQLUN + SINP + COSP + DPSI
      CALL MD2PI(GELS)
C
C  COMPUTE TRUE ANOMALY OF SUN
      TSAN = TAMAN + EQNCN
      CALL MD2PI(TSAN)
C  COMPUTE LATITUDE(GEOCENTRIC) OF SUN
      SNLAT = SNLT*SIN(F)
C  COMPUTE RADIUS VECTOR TO SUN
      RADSL = RDLS(1) + RDLS(2)*COS(AMAN) + RDLS(3)*COS(2.*AMAN)
     1  + RDLS(4)*COS(2.*AMAN-2.*GPP1) + RDLS(5)*COS(2.*AMAN-2.*GPP3)
     2  + RDLS(6)*COS(AMAN) + RDLS(7)*SIN(2.*AMAN-2.*GPP1)
     3  + RDLS(8)*SIN(AMAN-GPP4)
      RADSN = 10.**RADSL
```

```
C  COMPUTE GEOCENTRIC EQUATORIAL CARTESIAN COORDINATES IN A.U.
C  OF SUN
        IHAND = 1
        CALL POTDC(IHAND,GELS,SNLAT,X,Y,Z)
        CALL ECTRD(X,Y,Z,EPSLN,X,Y,Z)
        XS = X*RADSN
        YS = Y*RADSN
        ZS = Z*RADSN
        RETURN
        END



        COMPILER DOUBLE PRECISION
        SUBROUTINE DIABR(X,Y,Z,ST,LAT,XP,YP,ZP)

C  DIABR CORRECTS POSITION IN EQUATORIAL COORDIANTES FOR EFFECTS OF
C  DIURNAL ABERRATION.
C  NOTE: SIMPLE SERIES IS USED DUE TO SMALL SIZE OF CORRECTION
C
C  INPUT:
C       X,Y,Z - DIRECTION COSINES OF EQUATORIAL COORDS.
C       ST    - LOCAL SIDEREAL TIME IN RADIANS
C       LAT   - LATITUDE OF SITE IN RADIANS
C  OUTPUT:
C       X,Y,Z - CORRECTED EQUATORIAL DIRECTION COSINES.
C
C  REFERENCE: ESAENA, P.49.
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ RLR 1/81
        REAL LAT
        COMMON /MATH/ FILL(5),RDPAS,RDPTS
        CALL DCTPO(1,X,Y,Z,RA,DEC)
        HA = ST-RA
        COSLAT= COS(LAT)
        DRA = 0.0213*COSLAT*COS(HA)/COS(DEC)
        DDEC= 0.320 *COSLAT*SIN(HA)*SIN(DEC)
        RA = RA + DRA*RDPTS
        DEC= DEC+ DDEC*RDPAS
        CALL POTDC(1,RA,DEC,XP,YP,ZP)
        RETURN
        END



      COMPILER DOUBLE PRECISION
      SUBROUTINE HEAB8(X,Y,Z,ICTFG,XP,YP,ZP)
C CORRECTS THE DIRECTION COSINES OF A POSITION IN EQUATORIAL
C    COORDINATES FOR THE EFFECTS OF HELIOCENTRIC ABERRATION.
C X,Y,Z-POSITION IN EQUATORIAL COORDINATES.
C ICTFG-CATALOG FLAG-IF >0 POSITION FROM A CATALOG,I.E. NO E TERMS.
C XP,YP,ZP-CORRECTED POSITION IN EQUATORIAL COORDINATES.
C CALLS:SYSTEM:COS,SIN,ATAN2,SQRT;USER:RDTEC,ROTAZ,ROTAX,ECTRD
C USES COMMON BLOCK FUNAR.
```

```
C DOCUMENTATION:WOOLARD AND CLEMENCE.
C RADIUS IS CONSERVED.
C FNARG MUST BE CALLED PRIOR TO THE USE OF HEAB8 TO SET THE
C    FUNDAMENTAL ARGUMENTS TSUN,ESUN,EPSLN,GELS.
      COMMON/FUNAR/EPSLN,ZETA,ZEE,THETA,DPSI,DETA,XS,YS,ZS,ESUN,TSUN,
     1 GELS
C TSUN-TRUE ANOMALY OF SUN IN RADIANS.
C ESUN-ECCENTRICITY OF SUNS ORBIT
C EPSLN-OBLIQUITY OF THE ECLIPTIC IN RADIANS.
C GELS-GEOMETRIC LONGITUDE OF THE SUN IN RADIANS
C KAP IS THE CIRCULAR VELOCITY OF THE EARTH /SPEED OF LIGHT/SQRT(U-E**2)
      COMMON /MATH/ PIO2,PI,TWOPI,FILL(4)
      COMMON /H8COMM/ KAP
      REAL KAP, K
      DATA KAP /9.935346999E-5/
C COMPUTE THE LONGITUDE OF THE APEX.
      CSTA=COS(TSUN)
      APXL=PI+GELS+ATAN2((1.+ESUN*CSTA),(ESUN*SIN(TSUN)))
      K=KAP/SQRT(1.-ESUN*ESUN)
C CHECK  THE CATALOG FLAG.
      ETRM=0.
      IF(ICTFG.LE.0)ETRM=ESUN
C MOVE  X AXIS TO THE APEX OF THE MOTION.
      CALL RDTEC(X,Y,Z,EPSLN,XP,YP,ZP)
      CALL ROTAZ(XP,YP,ZP,APXL,XP,YP,ZP)
C GET THE POSITION AND THE X AXIS IN THE X-Y PLANE
      APLT=ATAN2(ZP,YP)
      CALL ROTAX(XP,YP,ZP,APLT,XP,YP,ZP)
C COMPUTE THE CORRECTION TO SECOND ORDER WITH ETERMS IF NEEDED.
      TETA=ATAN2(YP,XP)
      SNTH=SIN(TETA)
      VC2=(1.+ETRM*ETRM-2.*ETRM*COS(TSUN))*K*K
      DTET=SQRT(VC2)*SNTH-VC2*SNTH*COS(TETA)
C MAKE THE CORRECTION.
      CALL ROTAZ(XP,YP,ZP,DTET,XP,YP,ZP)
C GET BACK TO EQUATORIAL COORDINATES.
      APLTN=-APLT
      CALL ROTAX(XP,YP,ZP,APLTN,XP,YP,ZP)
      APXLN=-APXL
      CALL ROTAZ(XP,YP,ZP,APXLN,XP,YP,ZP)
      CALL ECTRD(XP,YP,ZP,EPSLN,XP,YP,ZP)
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE ECTRD(X,Y,Z,EPS,XP,YP,ZP)
C CONVERTS THE DIRECTION COSINES OF A POSITION IN THE ECLIPTIC
C    COORDINATES SYSTEM TO THE DITECTION COSINES OF THE POSITION IN THE
C    EQUATORIAL COORDINATE SYSTEM.
C X,Y,Z-POSITION IN ECLIPTIC COORDINATES.
C EPS-OBLIQUITY OF THE ECLIPTIC OF DATE IN RADIANS.
C XP,YP,ZP-POSITION IN EQUATORIAL COORDINATES.
```

```
C CALLS:USER:ROTAX.
C DOCUMENTATION:EICHHORN
C RADIUS IS CONSERVED.
      EPSN=-EPS
      CALL ROTAX(X,Y,Z,EPSN,XP,YP,ZP)
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE RDTEC(X,Y,Z,EPS,XP,YP,ZP)
C CONVERTS THE DIRECTION COSINES OF A POSITION IN THE EQUATORIAL
C    COOORDINATES SYSTEM TO THE DIRECTION COSINES OF THE POSITION
C    IN THE ECLIPTIC COORDINATE SYSTEM.
C X,Y,Z-POSITION IN EQUATORIAL COORDINATES.
C EPS-OBLIQUITY OF THE ECLIPTIC OF DATE IN RADIANS.
C XP,YP,ZP-POSITION IN ECLIPTIC COORDINTAES.
C CALLS:USER:ROTAX
C DOCUMENTATION: EICHHORN
C RADIUS IS CONSERVERED.
      CALL ROTAX(X,Y,Z,EPS,XP,YP,ZP)
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE PRENU(X,Y,Z,XP,YP,ZP)
C APPLIES THE CORRECTION FOR PRECESSION AND NUTATION TO THE DIRECTION
C    COSINES OF A POSITION IN EQUATORIAL COORDINATES.
C X,Y,Z-POSITION IN EQUATORIAL COORDINATES.
C CXP,YP,ZP-CORRECTED POSITION IN EQUATORIAL COORDINATES.
C CALLS:USER:ROTAZ,ROTAY,ROTAX.
C CONTAINS COMMON BLOCK FUNAR.
C DOCUMENTATION: ES-AENA.
C FNARG MUST BE CALLED PRIOR TO THE CALL TO PRENU TO SET THE
C    FUNDAMENTAL ARGUMENTS EPSLN,ZETA,ZEE,THETA,DPSI,DETA.
C RADIUS IS CONSERVED.
      COMMON /FUNAR/ EPSLN,ZETA,ZEE,THETA,DPSI,DETA,XS,YS,ZS,ESUN,TSAN,G
     CELS
C EPSLN-OBLIQUITY OF THE ELIPTIC OF DATE IN RADIANS.
C ZETA,ZEE,THETA-PRECESSIONAL QUATNTIES IN RADIANS
C DPSI-NUTATION IN LONGITUDE IN RADIANS.
C DETA-NUTATION IN OBLIQUITY IN RADIANS.
C CORRECT FOR PRECESSION.
      ZETAN=-ZETA
      ZAN=-ZEE
      CALL ROTAZ(X,Y,Z,ZETAN,XP,YP,ZP)
      CALL ROTAY(XP,YP,ZP,THETA,XP,YP,ZP)
      CALL ROTAZ(XP,YP,ZP,ZAN,XP,YP,ZP)
C CORRECT FOR NUTATION.
      DEPS=DETA-EPSLN
      CALL ROTAX(XP,YP,ZP,-DEPS,XP,YP,ZP)
```

```
      CALL ROTAZ(XP,YP,ZP,-DPSI,XP,YP,ZP)
      CALL ROTAX(XP,YP,ZP,-EPSLN,XP,YP,ZP)
      RETURN
      END



        COMPILER DOUBLE PRECISION
        SUBROUTINE PRMTN(RA,DEC,PMR,PMD,PMPR,PMPD,DT,RAP,DECP)

C  PROPER MOTION CALCULATIONS
C  INPUT:
C       RA,DEC  -- RADIANS
C       RMR,PMPR-- PM & DPM/DT IN RADIANS
C       PMD,PMPD-- PM & DPM/DT IN RADIANS
C       DT      -- FRACTION OF A TROPICAL CENTURY SINCE EPOCH OF CATALOG
C                        (TROPICAL CENTURY = 36524.22 DAY)
C  OUTPUT:
C       RAP,DECP-- CORRECTED POSITIONS IN RADIANS
C
C  REFERENCE: VAN DE KAMP, ELEMENTS OF ASTROMECHANICS
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ RLR 1/81

        DRA = -(PMR+PMPR*DT)*DT
        DDEC=  (PMD+PMPD*DT)*DT
        CALL OFSET(RA,DEC,DRA,DDEC,RAP,DECP)
        RETURN
        END



        COMPILER DOUBLE PRECISION
        SUBROUTINE OFSET (AZ,AL,DAZ,DAL,AZP,ALP)

C       OFSET CHANGES A POSITION AZ,AL BY ROTATING AMOUNTS DAZ,DAL
C       ABOUT THE AZIMUTH AND ITS SPECIFIC ALTITIUDE AXES RESPECTIVELY.
C       THE FINAL POSITION AZP,ALP IS ALSO POLAR ALT-AZIMUTH COORDINATES.
C
C       MARK A. POWELL - OCTOBER 28, 1976       COPY BY SJB 7/25/80


        IHAND=1
        CALL POTDC(IHAND,AZ,AL,X,Y,Z)
        CALL ROTAZ(X,Y,Z,AZ,X,Y,Z)
        CALL ROTAY(X,Y,Z,DAL,X,Y,Z)
        DDAZ=DAZ-AZ
        CALL ROTAZ(X,Y,Z,DDAZ,X,Y,Z)
        CALL DCTPO(IHAND,X,Y,Z,AZP,ALP)
        RETURN
        END
```

```
      COMPILER DOUBLE PRECISION
      SUBROUTINE HEPAR(X,Y,Z,PAR,XP,YP,ZP)
C
C     CORRECT A STAR'S POSITION FOR THE EFFECTS OF PARALLAX.
C
C           INPUT:  X,Y,Z   - POSITION IN DIRECTION COSINES OF
C                             EQUATORIAL COORDINATE SYSTEM.
C                   PAR     - TOTAL PARALLAX IN RADIANS.
C
C           OUTPUT: XP,YP,ZP- CORRECTED POSITIONS.
C
C           COMMON: FUNAR   -FUNDAMENTAL ARGUMENTS.
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
      COMMON /FUNAR/ EPSLN,ZETA,ZEE,THETA,DPSI,DETA,
     1         XS,YS,ZS,ESUN,TSAN,GELS
      DIS = COS(PAR)/SIN(PAR)
      XP = X*DIS + XS
      YP = Y*DIS + YS
      ZP = Z*DIS + ZS
      RAD = SQRT(XP*XP + YP*YP + ZP*ZP)
      XP = XP/RAD
      YP = YP/RAD
      ZP = ZP/RAD
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE ALTRD(X,Y,Z,LAT,LST,XP,YP,ZP)
C CONVERTS THE DIRECTION COSINES OF A POSITION IN THE ALTITUDE,AZIMUTH
C    COORDINATE SYSTEM TO THE DIRECTION COSINES OF THE POSITION IN
C    EQUATORIAL COORDINATES.
C X,Y,Z-POSITION IN ALT-AZIMUTH COORDINATES.
C LAT-GEOCENTRIC LATITUDE OF THE OBSERVER IN RADIANS.
C LST-LOCAL SIDEREAL TIME IN RADIANS
C XP,YP,ZP-POSITION IN EQUATORIAL COORDINATES.
C CALLS:USER:ALTHD,HDTRD
C DOCUMENTATION:EICHHORN
C RADIUS IS CONSEREVED
      REAL LAT,LST
      CALL ALTHD(X,Y,Z,LAT,XP,YP,ZP)
      CALL HDTRD(XP,YP,ZP,LST,XP,YP,ZP)
      RETURN
      END

      COMPILER DOUBLE PRECISION
      SUBROUTINE RDTAL (X,Y,Z,LAT,LST,XP,YP,ZP)
C CONVERTS THE DIRECTION COSINES OF A POSITION IN THE EQUATORIAL
C    COORDINATE SYSTEM TO THE DIRECTION COSINES IN ALTITUDE,AZIMUTH
C    COORDINATES.
C X,Y,Z-POSITION IN EQUATORIAL COORDINATES.
C LAT-OBSERVERS GEOCENTRIC LATITUDE IN RADIANS.
```

```
C LST-LCOAL SIDEREAL TIME IN RADIANS.
C XP,YP,ZP-POSITION IN ALTITUDE,AZIMUTH COORDINATES.
C CALLS:USER:RDTHD,HDTAL
C DOCUMENTATION:EICHHORN
C RADIUS IS CONSERVERED.
      REAL LAT,LST
      CALL RDTHD(X,Y,Z,LST,XP,YP,ZP)
      CALL HDTAL(XP,YP,ZP,LAT,XP,YP,ZP)
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE ALTHD(X,Y,Z,LAT,XP,YP,ZP)
C
C     CONVERTS THE DIRECTION COSINES OF A SET OF ALT-AZIMUTH COORDINATES
C       TO THE DIRECTION COSINES OF THE HOUR ANGLE - DECLINATION
C       COORDINATES.   MARK A. POWELL - OCTOBER 28, 1976
C
C       LAT IS THE LATITUDE (GEOCENTRIC) OF THE OBSERVER.
C
      REAL LAT
      COMMON /MATH/ PIO2,PI,TWOPI, FILL(4)
      COLAT=LAT-PIO2
      CALL ROTAZ(X,Y,Z,PI,XP,YP,ZP)
      CALL ROTAY(XP,YP,ZP,COLAT,XP,YP,ZP)
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE HDTAL(X,Y,Z,LAT,XP,YP,ZP)
C
C     CONVERTS THE DIRECTION COSINES OF A SET OF HOUR ANGLE - DECLINATION
C       COORDINATES TO THE DIRECTION COSINES OF THE ALT-AZIMUTH
C       COORDINATES.   MARK A. POWELL - OCTOBER 28, 1976
C
C       LAT IS THE LATITUDE (GEOCENTIRC) OF THE OBSERVER.
C
      REAL LAT
      COMMON /MATH/ PIO2,PI,TWOPI,FILL(4)
      COLAT=PIO2-LAT
      CALL ROTAY(X,Y,Z,COLAT,XP,YP,ZP)
      CALL ROTAZ(XP,YP,ZP,-PI,XP,YP,ZP)
      RETURN
      END



        COMPILER DOUBLE PRECISION
        SUBROUTINE HDTRD(X,Y,Z,LST,XP,YP,ZP)
C       CONVERT THE DIRECTION COSINES OF A POSITION IN THE HOUR ANGLE, DECLIN-
```

```
C       ATION COORDINATE SYSTEM TO TH DIRECTION COSINES OF THE POSITION IN
C       THE EQUATORIAL COORDINATE SYSTEM.
        REAL LST
        YT = -Y
        CALL ROTAZ(X,YT,Z,-LST,XP,YP,ZP)
        RETURN
        END



      COMPILER DOUBLE PRECISION
      SUBROUTINE RDTHD(X,Y,Z,LST,XP,YP,ZP)
C CONVERTS THE DIRECTION COSINES OF A POSITION IN THE EQUATORIAL COOR-
C    DINATES SYSTEM TO THE DIRECTION COSINES OF THE POSITION IN
C    H.A.,DECLINATION COORDINATES.
C X,Y,Z-POSITION IN EQUATORIAL COORDINATES
C LST-LOCAL SIDEREAL TIME IN RADIANS.
C XP,YP,ZP-POSITION IN H.A.,DEC. COORDINATES
C CALLS: USER: ROTAZ
C DOCUMENTATION: EICHHORN
C RADIUS IS CONSERVERED.
      REAL LST
      CALL ROTAZ(X,Y,Z,LST,XP,YP,ZP)
      YP=-YP
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE POTDC(IHAND,XL,XC,X,Y,Z)
C
C    CONVERTS POLAR COORDINATES TO DIRECTION COSINES. HANDEDNESS OF THE
C      COORDINATES SYSTEM IS CHANGED IF IHAND{0.
C    MARK A. POWELL - OCTOBER 28, 1976
C
      CSXC=COS(XC)
      X=CSXC*COS(XL)
      Y=CSXC*SIN(XL)
      Z=SIN(XC)
      IF (IHAND.LT.0) Y=-Y
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE DCTPO(IHAND,X,Y,Z,XL,XC)
C
C    CONVERTS DIRECTION COSINES TO POLAR COORDINATES. HANDEDNESS OF THE
C      POLAR COORDINATES IS CHANGED IF IHAND{0.
C    MARK A. POWELL - OCTOBER 28, 1976
C
      XL=ATAN2(Y,X)
```

```
      R=SQRT(X*X+Y*Y)
      XC=ATAN2(Z,R)
      IF (IHAND.LT.0) XL=-XL
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE RDTDG(X,ID,IM,XS)
C
C     CONVERT RADIANS TO DEGREES, ARCMINUTES AND ARCSECONDS.
C     MARK A. POWELL - OCTOBER 28, 1976
C
      COMMON /MATH/ FILL1(3), RDPDG, FILL2(3)
      XS=X/RDPDG
      ID=IFIX(XS)
      XS=ABS((XS-FLOAT(ID))*60.)
      IM=IFIX(XS)
      XS=(XS-FLOAT(IM))*60.
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE RDTHR(X,IH,IM,XS)
C
C     CONVERT RADIANS TO HOURS,MINUTES AND SECONDS.
C     MARK A. POWELL - OCTOBER 28, 1976
C
      COMMON /MATH/ FILL1(4),RDPHR,FILL2(2)
      XS=X/RDPHR
      IH=IFIX(XS)
      XS=ABS((XS-FLOAT(IH))*60.)
      IM=IFIX(XS)
      XS=(XS-FLOAT(IM))*60.
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE MD2PI(X)
C DOES MODULO 2 PI FOR X
C X-ANGLE IN RADIANS.
C CALLS:SYSTEM:ATAN2,ABS,SIGN.
      COMMON /MATH/ PIO2,PI,TWOPI,FILL(4)
    1 IF(ABS(X).LT.TWOPI) GO TO 2
      X=X-SIGN(TWOPI,X)
      GO TO 1
    2 RETURN
      END
```

```
      COMPILER DOUBLE PRECISION
      FUNCTION RVTRD(R,X)
C CONVERTS MANY REVOLUTIONS TO MODULO 2 PI.
C R-NUMBER OF REVOLUTIONS.
C X-MULTIPLIER OF REVOLUTINONS.
C CALLS:SYSTEM:ATAN2,AINT
C DOCUMENTATION:NONE
C DESIGNED FOR USE WITH FNARG.
      COMMON /MATH/ PIO2,PI,TWOPI,FILL(4)

      RV=R*X
      FR=RV-AINT(RV)
      RVTRD=FR*TWOPI
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE ROTAX(X,Y,Z,ANGLE,XP,YP,ZP)
C
C    ROTATE ANGLE AMOUNT ABOUT  X AXIS.
C    MARK A. POWELL - OCTOBER 28, 1976
C
      CSA=COS(ANGLE)
      SNA=SIN(ANGLE)
      YS=Y
      XP=X
      YP=Y*CSA+Z*SNA
      ZP=Z*CSA-YS*SNA
      RETURN
      END



      COMPILER DOUBLE PRECISION
      SUBROUTINE ROTAY(X,Y,Z,ANGLE,XP,YP,ZP)
C
C    ROTATE ANGLE AMOUNT ABOUT Y AXIS.
C    MARK A. POWELL - OCTOBER 28, 1976
C
      CSA=COS(ANGLE)
      SNA=SIN(ANGLE)
      ZS=Z
      YP=Y
      ZP=Z*CSA+X*SNA
      XP=X*CSA-ZS*SNA
      RETURN
      END
```

```
      COMPILER DOUBLE PRECISION
      SUBROUTINE ROTAZ(X,Y,Z,ANGLE,XP,YP,ZP)
C
C     ROTATE ANGLE AMOUNT ABOUT Z AXIS.
C     MARK A. POWELL - OCTOBER 28, 1976
C
      CSA=COS(ANGLE)
      SNA=SIN(ANGLE)
      XS=X
      ZP=Z
      XP=X*CSA+Y*SNA
      YP=Y*CSA-XS*SNA
      RETURN
      END
```

```
SITE:
         LONG:      0.000000
         GDLAT:     0.00000
         LAT:       0.00000

ENVIRONMENT:
         TEMP:    273.
         PRES:    800.

TIME (JD):  2444446.800000

FUNDAMENTAL ARGUMENTS (RADIANS AND UNITLESS QUANTITIES):
         OBLIQUITY OF ECLIPTIC:                 0.4091006810D  0
         PRECESS (ZETA):                        0.3415916548D -2
         PRECESS (ZEE=RA OF NODE):              0.3416274858D -2
         PRECESS (THETA=INCL OF EQUATOR):       0.2969972493D -2
         NUTATE (D PSI):                       -0.4819099457D -4
         NUTATE (D ETA):                       -0.3607581390D -4
         GEO X OF SUN:                         -0.5465765127D  0
         GEO Y OF SUN:                          0.7853086957D  0
         GEO Z OF SUN:                          0.3404819428D  0
         ECC OF SUN:                            0.1671728155D -1
         TRUE ANOM OF SUN:                      0.3505416968D  1
         GEOM LONG OF SUN:                      0.2139092112D  1

STAR INFO:
(RA,DEC IN DEG,HOURS; CORRECTIONS IN SEC OF ARC,TIME)
         RA:     6.3806925
         DEC: -52.6676250
         MU RA:       0.290          MU DOT RA:       0.004
         MU DEC:      2.220          MU DOT DEC:     -0.040

         CATALOG EPOCH (JD):  2433282.423000

LOCAL SIDEREAL TIME (HOURS):    3.4757140

RESULTS OF PROPER MOTION:
```

```
    NEW RA,DEC:    6.3807172  -52.667438
     DRA,DDEC:      0.09         0.67

RESULTS OF PRECESSION & NUTATION:
    NEW RA,DEC:    6.3919567  -52.686564
    DRA,DDEC:     40.46       -68.86

RESULTS OF HELIOCENTRIC ABERRATION:
    NEW RA,DEC:    6.3914018  -52.683962
    DRA,DDEC:     -2.00         9.37

RESULTS OF DIURNAL ABERRATION:
    NEW RA,DEC:    6.3914089  -52.683914
    DRA,DDEC:      0.03         0.18
    CON RA,DEC:    6 23 29.072     -52 41  2.089

COMPARISON POSITION (W/O DIURNAL ABERRATION):
                   6 23 29. 59     -52 40 61.970

RESULTS OF REFRACTION:
    AZ,ALT:  152.212627  25.977116
    REFRACTED ALT:   26.004286
    DALT:    97.81


SITE:
         LONG:      0.000000
         GDLAT:     0.00000
         LAT:       0.00000

ENVIRONMENT:
         TEMP:    273.
         PRES:    800.

TIME (JD):  2444446.800000

FUNDAMENTAL ARGUMENTS (RADIANS AND UNITLESS QUANTITIES):
         OBLIQUITY OF ECLIPTIC:                0.4091006810D  0
         PRECESS (ZETA):                       0.3415916548D -2
         PRECESS (ZEE=RA OF NODE):             0.3416274858D -2
         PRECESS (THETA=INCL OF EQUATOR):      0.2969972493D -2
         NUTATE (D PSI):                      -0.4819099457D -4
         NUTATE (D ETA):                      -0.3607581390D -4
         GEO X OF SUN:                        -0.5465765127D  0
         GEO Y OF SUN:                         0.7853086957D  0
         GEO Z OF SUN:                         0.3404819428D  0
         ECC OF SUN:                           0.1671728155D -1
         TRUE ANOM OF SUN:                     0.3505416968D  1
         GEOM LONG OF SUN:                     0.2139092112D  1

STAR INFO:
(RA,DEC IN DEG,HOURS; CORRECTIONS IN SEC OF ARC,TIME)
         RA:    13.7595261
         DEC:   49.5622583
```

```
        MU RA:      -1.285          MU DOT RA:      0.014
        MU DEC:     -1.410          MU DOT DEC:    -0.080

        CATALOG EPOCH (JD):  2433282.423000

LOCAL SIDEREAL TIME (HOURS):   3.4757140

RESULTS OF PROPER MOTION:
    NEW RA,DEC:  13.7594174   49.562137
     DRA,DDEC:    -0.39        -0.44

RESULTS OF PRECESSION & NUTATION:
    NEW RA,DEC:  13.7793077   49.411814
    DRA,DDEC:    71.61       -541.16

RESULTS OF HELIOCENTRIC ABERRATION:
    NEW RA,DEC:  13.7792679   49.416826
    DRA,DDEC:    -0.14        18.04

RESULTS OF DIURNAL ABERRATION:
    NEW RA,DEC:  13.7792596   49.416797
    DRA,DDEC:    -0.03        -0.10
    CON RA,DEC:  13 46 45.335      49 25  0.470

COMPARISON POSITION (W/O DIURNAL ABERRATION):
                 13 46 45.337      49 25  0.570

RESULTS OF REFRACTION:
    AZ,ALT:   20.206438 -35.975412
    REFRACTED ALT:  -25.968247
    DALT: 36025.80
```

9 APPENDIX B

```
        COMPILER DOUBLE PRECISION

C  THIS IS A TEST DRIVER FOR THE NON-LINEAR LEAST-SQUARES SUB-SYSTEM.
C  THE INPUT DATA IS IN AN ABBREVIATED FORMAT: MLRS DATA INCLUDES A
C  HEADER LINE, STAR NUMBER,OPTICAL PATH, TIME OF OBSERVATION,
C  AND PRE-FIT AND POST-FIT RESIDUALS.
C
C  INPUT (ALL ANGULAR VARIABLES ARE IN RADIANS)
C       MODEL   -- MODEL NUMBER TO BE USED INN "FUNC".
C                  DIFFERENT NUMBERS REPRESENT DIFFERENT LIGHT PATHS OR
C                  EXPERIMENNTAL MODELS.
C       OLON,OLAT- OBSERVED TARGET POSITION IN COORDINATE SYSTEM LATITUDINAL
C                  AND LONGITUDINAL AXES (E.G., AZ/ALT, X/Y). THESE ARE
C                  RAW ENCODER READINGS CONVERTED TO RADIANS.
C       CLON,CLAT- CALCULATED TARGETS POSITIONS.
C       NOBS    -- NUMBER OF POSITIONS OBSERVED.
C       SOLVEC--LIST OF PARAMETERS (BY NUMBER) TO BE IN SOLUTION (IN ANY ORDER)
C       NCOEF   -- NUMBER OF PARAMETERS IN SOLUTION.
C       ORIRAD  -- INITIAL ORIENTATION PARAMETERS IN RADIANS. THESE ARE
C                  USED AS "FIRST GUESS" BY FITTING ROUTINES.
C
C  OUTPUT
C       ORIRAD  -- FINAL ORIENTATION PARAMETER VALUES.
C       OCLONP, -- FINAL OBSERVED-CALCULATED RESIDUALS (POST-FIT)
C       OCLATP
C
C       STANDARD DEVIATION OF EACH ITERATION TO THE FIT IS WRITTEN TO
C       THE OPERATOR CONSOLE, AND A FINAL SUMMARY IS WRITTEN TO LINE PRINTER.
C
C  PERIPHERAL DEVICES ASSUMED:
C       UNIT 8  -- INPUT DATA FILE
C       UNIT 10 -- CONSOLE OUTPUT
C       UNIT 11 -- CONSOLE INPUT
C       UNIT 12 -- LINE PRINTER OUTPUT
C
C  LOAD THE FOLLOWING ROUTINES:
C       LSQDEM,BDMATH,BDPARM,LSOLN,LSINT,DNORMEQ,SOLCOR,FUNC
C       RECAP & MATRIX INVERSION ROUTINE
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ RLR 10/81
        COMMON /OBSVTN/ MODEL(200),OLON(200),OLAT(200), CLON(200),CLAT(200),
     X                  OCLON(200),OCLAT(200), OCLONP(200),OCLATP(200),NOBS
        COMMON /OBSINF/ HDRLIN(40), NSTAR(200),IHR(200),IMIN(200),ISEC(200)
        INTEGER HDRLIN,NSTAR,IHR,IMIN,ISEC
        INTEGER DATNAM(7)
        COMMON /HSCMN/ SOLVEC(20),NCOEF
        INTEGER SOLVEC,NCOEF
        COMMON /TELCOR/ ORFILL(20), ORIRAD(20), NP
        DATA SOLVEC /1,2,3,4,5,6,7,8,12*0/, NCOEF /8/
        DATA ORIRAD /20*0./
        DATA MODEL /20*1/
C-------------------------------------------------------------------
```

```
C  OPEN REPORT FILE FOR TEST
        CALL OPEN(12,"LSQOP",2,IERR)
C  GET THE DATA FILE NAME
5       WRITE(10,1000)
1000    FORMAT("  DATA FILE NAME?: ", Z)
        READ(11,1010) DATNAM
1010    FORMAT(7A2)
        IF (DATNAM(1).NE."  ") GO TO 25
C
9       WRITE(10,1001)
1001    FORMAT(8X, "INAPPROPRIATE RESPONSE")
        GO TO 5
C
C  OPEN THE DATA FILE
25      CALL OPEN(8,DATNAM,2,IERR)
          IF (IERR.NE.1) CALL ERROR(5240,IERR)
        NOBS = 1
        REWIND 8
C  READ THE POINTING DATA
50      READ(8,1020,END=100) OLON(NOBS),OLAT(NOBS),CLON(NOBS),CLAT(NOBS)
1020    FORMAT(4(E12.8,2X))
        NOBS= NOBS+ 1
        IF (NOBS.GT.200) GO TO 100
        GO TO 50
C
C  CLOSE OUT THE DATA FILE
100     NOBS= NOBS- 1
C  SOLVE FOR COEFFICIENTS WHOSE NUMBERS ARE IN SOLVEC
C  NOTE: SOLVEC IS NORMALLY OPERATOR SELECTED FOR FLEXIBILITY.
        CALL LSOLN(SOLVEC,NCOEF,NPASS)
C  PRINT THE RESULTS AND CLOSE IT ALL UP.
        CALL RECAP(-NPASS)
        CALL FCLOS(12)
        CALL FCLOS(8)
        STOP
        END

        COMPILER DOUBLE PRECISION
        BLOCK DATA
C  ** BDMATH **
C               BLOCK DATA FOR MATH COMMON
C
        COMMON /MATH/ PIO2,PI,TWOPI,RDPDG,RDPHR,RDPAS,RDPTS
C
C  ** MATH COMMON **
        DATA PIO2  /1.570796326794897/  ;PI/2
        DATA PI    /3.141592653589793/  ;PI
        DATA TWOPI /6.283185307179586/  ;2*PI
        DATA RDPDG /1.745329251994330E-2/;RADS/DEGREE
        DATA RDPHR /2.617993877991494E-1/;RADS/HOUR
        DATA RDPAS /4.848136811      E-6/;RADS/ARC SEC
        DATA RDPTS /7.272205217      E-5/;RADS/TIME SEC
C
        END
```

```
      BLOCK DATA

C  ** BDPARM **
C  BLOCK DATA FOR ORIENTATION PARAMETERS
      COMMON /LABELS/ ORILIS(20,20),MMM,NNN
      INTEGER ORILIS
      COMMON /ORILIM/ ORLO(20), ORHI(20)
      COMMON /TELCOR/ ORIENT(20), OPARMS(20), NPARM
C
      DATA MMM,NNN /20,20/
      DATA ORILIS
     1 /"1 AZIMUTH AXIS ENCODER OFFSET          "
     2 ,"2 ALTITUDE AXIS ENCODER OFFSET         "
     3 ,"3 X AXIS ROTATION COMPONENT            "
     4 ,"4 Y AXIS ROTATION COMPONENT            "
     6 ,"5 TRANSVERSE MISALIGNMENT              "
     8 ,"6 LATERAL MISALIGNMENT                 "
     7 ,"7 AZ & ALT AXIS NONORTHOGONALITY       "
     8 ,"8 LATCH FLEXURE                        "
     5 ,"9 ALTITUDE ENCODER WOBBLE              "
     9 ,"10 ALTITUDE ENCODER PHASE (ENTER ONLY) "
     1 ,"11 AZIMUTH STOW POSITION               "
     2 ,"12 ALTITUDE STOW POSITION              "
     X /
C
      DATA ORLO /-360., -105., 7*-2., -11., -360., -105./
C
      DATA ORHI /+360.,  -75., 7*+2., +11., +360.,  +90./
C
      DATA NPARM /10/
      END


      COMPILER DOUBLE PRECISION
      SUBROUTINE LSOLN(ISOLVEC,ICOEF,NPASS)
C
C     ROUTINE NSOLN CONTROLS THE NON-LINEAR LEAST
C  SQUARES SOLUTION FOR THE TELESCOPE ORIENTATION PARAMETERS USING
C  CALCULATED AND OBSERVED POSITIONS OF <= 200 STARS OR GROUND TARGETS
C  THE PROGRAM ITERATES UNTIL THE NEW CORRECTION TO THE
C  ORIENTATION PARAMETERS ARE "SUFFICIENTLY SMALL" -- 1.E-6 RADIANS, NOW.
C
C  SEE ANY TEXT DEALING WITH LEAST SQUARES FOR A DISCUSSION OF
C  THEORY OF THIS METHOD.
C
C  INPUT:
C       <<FROM /OBSVTN/ >>
C          OLON,OLAT -- OBSERVED POSITIONS  (AZ & ALT) UNEFFECTED
C                       BY ORIENTATION PARAMETERS.
C
C          CLON,CLAT -- CALCULATED POSITIONS IN RADIANS. (CALC.
C                       POSITIONS ARE NEVER EFFECTED BY ORIENTATION
C                       PARAMETERS.)
C
```

```
C          NOBS      -- NUMBER OF STARS RECORDED.
C
C  INTERMEDIATE VALUES:
C       <<FROM /LSTSQR/ >>
C          NMLMAT    -- THE NORMAL MATRIX AND INVERTED TO OBTAIN
C                       CORRECTIONS TO PARAMETERS
C
C          COEFO     -- ORIGINAL ORIENTATION PARAMETERS USED TO GET NEW FIT.
C
C          COCORR    -- THE SOLUTION OF THE LEAST SQUARES FIT. THESE ARE
C                       CORRECTIONS ADDED TO THE COEFO.
C
C          F         -- THE NUMERICAL DERIVATIVE OF THE CURRENT STAR
C                       POSITIONS W.R.T. EACH PARAMETER.
C
C          LHS       -- POST-FIT O-C (OBSERVED-CALCULATED) RESIDUALS FOR
C                       EACH OBSERVATION.
C
C          W         -- WEIGHT GIVEN EACH OBSERVATION.
C
C          SD        -- STANDARD DEVIATION (SQUARED) OF THE FIT.
C
C          COVMAT    -- COVARIANCE MATRIX OF THE FIT. (NOTE: ITH DIAGONAL
C                       ELEMENT IS STANDARD DEVIATION SQUARED OF
C                       ITH PARAMETER.)
C
C          NCOEF     -- NUMBER OF COEFFICIENTS BEING SOLVED FOR
C                       IN THE FIT.
C
C  OUTPUT:
C       <<FROM /TELCOR/ >>
C          ORIRAD    -- NEW ORIENTATION PARAMETERS IN RADIANS.
C                       SEE MNLORI FOR DESCRIPTION OF INDIVIDUAL PARAMETERS
C
C  CALLS:       SCAN
C               DNORMEQ
C               MATINV
C               SOLCOR
C               ERROR(532-,IERR)
C
C  COMMONS:     LSTSQR
C               TELCOR
C               MATH     -- COMMONLY USED MATH CONSTANTS
C
C                                                                       RLR
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

      COMMON /LSTSQR/ NMLMAT(20,20), COEFO(20), COCORR(20), COVMAT(20,20),
     X                LHS(20),SD, SOLVEC(20), NCOEF
      REAL NMLMAT,LHS
      INTEGER SOLVEC,NCOEF
      COMMON /TELCOR/ ORIENT(20), ORIRAD(20), NPARM
      COMMON /MATH/ FILLM1(3),RDPDG,FILLM2(3)
      INTEGER ISOLVEC(20),ICOEF
```

```
C-------------------------------------------------------------------------
        DO 10 II=1,20
10      SOLVEC(II) = ISOLVEC(II)
        NCOEF = ICOEF
        NPASS = 0
        WRITE(10,1000)
1000    FORMAT(/4X,"ITERATION",8X,"STANDARD DEVIATION"   /1X,47("-")/)
C
C  INITIALIZE VARIABLES
C
100     NPASS = NPASS + 1
        CALL LSINT(ORIRAD)
C  SET UP NORMAL EQNS
        CALL DNORMEQ
C  INVERT NORMAL MATRIX TO GET SOLUTION
        CALL MATINV(NMLMAT,NCOEF,DET)
        IF (DET.NE.0.) GO TO 150
            TYPE " ** DETERMINATE=0 - ACCEPT OLD SOLUTION OR TAKE MORE READINGS"
            NOBS = 0
            RETURN
C
C  CORRECT COEFFICIENTS
150     CALL SOLCOR
        SDD = SQRT(SD)/RDPDG
        WRITE(10,1010) NPASS, SDD
1010    FORMAT(8X,I2,15X,F10.4)
X       CALL RECAP(NPASS)
C  CHECK TO SEE THAT WE HAVE MET TOLERANCES
        DO 200 II=1,NCOEF
                IF (ABS(COCORR(II)).GT.1.E-6) GO TO 100
200     CONTINUE
C  FINISHED!
        RETURN
        END

        COMPILER DOUBLE PRECISION
        SUBROUTINE LSINT(BCOEF)
C  LSINT INITIALIZES ARRAYS FOR THE NON-LINEAR LEAST SQUARES SUBSYSTEM.
C  IT ZEROES:
C       NMLMAT
C       COCORR
C       LHS
C  AND SETS COEF0 TO BCOEF, THE INITIAL VALUES OF THE PARAMETERS IN THE
C  FIT.
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ RLR 10/80
        DIMENSION BCOEF(20)
        COMMON /LSTSQR/ NMLMAT(20,20), COEF0(20), COCORR(20), COVMAT(20,20),
     X                  LHS(20),SD, SOLVEC(20), NCOEF
        REAL NMLMAT, LHS
        INTEGER SOLVEC,NCOEF
C-------------------------------------------------------------------------
        SD = 0.
        DO 10 II=1,20
```

```
            DO 20 JJ=1,20
20          NMLMAT(II,JJ) = 0.
            COCORR(II) = 0.
            LHS(II)    = 0.
            COEF0(II)  = BCOEF(II)
10      CONTINUE
        RETURN
        END

        COMPILER DOUBLE PRECISION
        SUBROUTINE DNORMEQ
C
C  DOUBLE NORMAL EQUATIONS
C
C  THIS ROUTINE TAKES A SET OF GENERALIZED LONGITUDES AND LATITUDES
C  AS WELL AS THE NAME OF THE PARTIAL DERIVATIVE FUNCTION (USED FOR
C  BOTH COORDINATES) AND CREATES THE LEAST SQUARES NORMAL MATRIX.
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ RLR 10/80
        COMMON /OBSVTN/ MODEL(200), OLON(200),OLAT(200), CLON(200),CLAT(200),
     X                  OCLON(200),OCLAT(200), OCLONP(200),OCLATP(200), NOBS
        COMMON /LSTSQR/ NMLMAT(20,20), COEF0(20), COCORR(20), COVMAT(20,20),
     X                  LHS(20), SD, SOLVEC(20),NCOEF
        INTEGER SOLVEC,NCOEF
        REAL NMLMAT, LHS
        COMMON /MATCMN/ DLODCO(20),DLADCO(20), WLO,WLA
        COMMON /MATH/ PIO2,PI,TWOPI,RDPDG,RDPHR, FILLXX(2)
C-----------------------------------------------------------------
        DO 500 N=1,NOBS
C  FIND PREFIT OBSERVED-CALCULATED (O-C) RESIDUALS
        CALL FUNC(OLON(N),OLAT(N),MODEL(N),COEF0,0,0.,RLON0,RLAT0)
        OCLONP(N) = CLON(N)-RLON0
        IF (ABS(OCLONP(N)).GT.PI) OCLONP(N) = OCLONP(N)-SIGN(TWOPI,OCLONP(N))
        OCLATP(N) = CLAT(N)-RLAT0
        IF (ABS(OCLATP(N)).GT.PI) OCLATP(N) = OCLATP(N)-SIGN(TWOPI,OCLATP(N))
C  FIND PARTIALS W.R.T. THIS PAIR OF OBSERVATIONS
C  SOLVEC ALLOWS SOLVING FOR A SELECTED SET OF COEFFICIENTS.
        DO 100 II=1,NCOEF
            CALL FUNC(OLON(N),OLAT(N),MODEL(N),COEF0,SOLVEC(II),+1.E-5,RLON1,RLA
            CALL FUNC(OLON(N),OLAT(N),MODEL(N),COEF0,SOLVEC(II),-1.E-5,RLON2,RLA
            DLODCO(II) = (RLON1-RLON2)/2.E-5
            DLADCO(II) = (RLAT1-RLAT2)/2.E-5
100     CONTINUE
C  SET WEIGHT
C  LONG ACCURACY DECREASES WITH LATITUDE
        WLO = 1.*COS(RLAT0)
C  LAT ACCURACY IS CONSTANT
        WLA = 1.
C  UPDATE NORMAL MATRIX
        DO 200 II=1,NCOEF
            DO 150 JJ=1,NCOEF
                NMLMAT(II,JJ) = NMLMAT(II,JJ)
     X                        + DLODCO(II)*DLODCO(JJ)*WLO
     X                        + DLADCO(II)*DLADCO(JJ)*WLA
```

```
150     CONTINUE
C ... AND LEFT-HAND-SIDE OF NORMAL EQNS
        LHS(II) = LHS(II)
     X             + DLODCO(II)*OCLONP(N)*WLO
     X             + DLADCO(II)*OCLATP(N)*WLA
200     CONTINUE
C  UPDATE SIGMA
        SD = SD + OCLONP(N)*OCLONP(N)*WLO
     X          + OCLATP(N)*OCLATP(N)*WLA
500     CONTINUE
        RETURN
        END


        COMPILER DOUBLE PRECISION
        SUBROUTC          NSTARS -- NUMBER OF STARS RECORDED
C
C    <<FROM /LSTSQR/>>
C         NCOEF  -- NUMBER OF COEFFICIENTS IN THE SOLUTION.
C
C         COEF0  -- ORIGINAL COEFFICIENTS (IN RADIANS).
C
C         SD     -- STANDARD DEVIATION (SQUARED) IN RADIANS (SQUARED)
C
C         NMLMAT -- SOLUTION MATRIX
C
C  OUTPUT:
C
C    <<FROM /LSTSQR/>>
C         COCORR -- CORRECTIONS TO OLD PARAMETERS.
C
C         COVMAT -- COVARIANCE (ERROR) MATRIX
C
C    <<FROM /TELCOR/>>
C         ORIRAD -- NEW ORIENTATION PARAMETERS IN RADIANS.
C
C  COMMONS:     STARPS
C               LSTSQR
C               TELCOR
C
C                                                                    RLR
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
        COMMON /OBSVTN/ MODEL(200), OLON(200),OLAT(200), CLON(200),CLAT(200),
     X                  OCLON(200),OCLAT(200),OCLONP(200),OCLATP(200),NOBS
C
        COMMON /LSTSQR/ NMLMAT(20,20), COEF0(20), COCORR(20), COVMAT(20,20),
     X          LHS(20),SD,SOLVEC(20),NCOEF
        INTEGER SOLVEC,NCOEF
        REAL NMLMAT,LHS
C
        COMMON /TELCOR/ ORIENT(20), ORIRAD(20), NPARM
C
C  CALCULATE ESTIMATE FOR STANDARD DEVIATION WITH NOBS-NCOEF
```

```
C   DEGREES OF FREEDOM
C
        NDF = 2*NOBS-NCOEF
        IF (NDF.GT.0) SD = SD/FLOAT(NDF)
C
C  CREATE COVARIANCE MATRIX, CORRECTIONS TO COEFS
C
        DO 200 II=1,NCOEF
                DO 100 JJ=1,NCOEF
                        COVMAT(II,JJ) = SD*NMLMAT(II,JJ)
                        COCORR(II) = COCORR(II) + NMLMAT(JJ,II)*LHS(JJ)
100             CONTINUE
C
C     CALC NEW COEFN
        SUBROUTINE FUNC(AZ,ALT, MODEL, COEF0, L, DIFF, AZF,ALTF)
C  AZIMUTH FUNCTION USED IN NUMERICAL DIFFERENTATION
C  OF THE OBSERVATIONS
C
C               AZF = XLEVEL TERM + YLEVEL TERM
C                       + NONORTHOGONALITY TERM
C                       + TRANSVERSE TERM
C                       + AZIMUTH OFFSET + ORIGINAL AZ
C
C  INPUT:       AZ,ALT -- OBSERVED POSITIONS OF STARS IN RADIANS
C
C               MODEL  -- PARTICULAR MODEL OR LIGHT PATH. NOT USED HERE.
C
C               COEF0  -- CURRENT APPROX. OF THE ORIENTATION PARAMETERS
C
C               L      -- THE PARAMETER WHICH WE DIFFERENTIATE
C                         WITH RESPECT TO.
C
C               DIFF   -- THE AMOUNT BY WHICH THE PARAMETER IS BEING
C                         "WIGGLED" IN DIFFERENTIATION.
C
C  OUTPUT:      AZF    -- THE VALUE OF THIS AXIS
C                          WITH THE "COEF0" PARAMETERS AND "DIFF" OFFSET
C                          OF THE LTH PARAMETER FOR THE POSITIONS
C                          "AZ" & "ALT".
C
C  COMMONS:     PARTC
C
C                                                                    RLR
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
        REAL COEF0(20)
        COMMON /PARTC/ DELTA(20)
        DATA DELTA /20*0.0/
C
        IF (L.EQ.0) GO TO 10
            CHOLD = COEF0(L)
            COEF0(L) = COEF0(L)+DIFF
C
10      AZF = -((COEF0(4)+DELTA(4))*SIN(AZ)
```

```
     1        +(COEF0(3)+DELTA(3))*COS(AZ))
     2                           *TAN(ALT+COEF0(2)+DELTA(2))
     3        +(COEF0(7)+DELTA(7)
     3        +(COEF0(8)+DELTA(8))*COS(3.*AZ))
     3                           *TAN(ALT+COEF0(2)+DELTA(2))
     4        +(COEF0(5)+DELTA(5))/COS(ALT+COEF0(2)+DELTA(2))
     5        +(COEF0(1)+DELTA(1))
     6        +AZ
C
        ALTF = -(COEF0(4)+DELTA(4))*COS(AZ)
     1          +(COEF0(3)+DELTA(3))*SIN(AZ)
     2          +(COEF0(6)+DELTA(6))*SIN(ALT+COEF0(2)+DELTA(2))
     5          +(COEF0(9)+DELTA(9))
     6                   *COS(32.8*(ALT+COEF0(2)+DELTA(2)-COEF0(10)))
     3          +(COEF0(2)+DELTA(2))
     4          +ALT
C
C
        IF(COS(ALT+COEF0(2)+DELTA(2)) .LE. 0.) ALTF = ALTF -
     2           2.*(COEF0(6)+DELTA(6))*SIN(ALT+COEF0(2)+DELTA(2))
C
        IF (L.NE.0) COEF0(L) = CHOLD
C
        RETURN
        END



        COMPILER DOUBLE PRECISION
        SUBROUTINE RECAP(NPASS)

C  WRITES RECAP OF THIS ITERATION OF THE SOLUTION. SPECIFICALLY, IT PRINTS:
C       1) THE POSTFIT (O-C) RESIDUALS,
C       2) THE NEW CORRECTIONS TO THE COEFFICIENTS,
C       3) THE NEW COEFFICIENTS,
C       4) THE STANDARD DEVIATION
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ RLR 10/80
        COMMON // IFILLB(2),DATFIL(7)
        INTEGER DATFIL
        COMMON /MATH/ PIO2,PI,TWOPI,RDPDG,RDPHR, RDPAS,RDPTS
        COMMON /OBSVTN/ MODEL(200),OLON(200),OLAT(200),CLON(200),CLAT(200),
     X                  OCLON(200),OCLAT(200),OCLONP(200),OCLATP(200),NOBS
        COMMON /OBSINF/ HDRLIN(40),IFILL(800)
        INTEGER HDRLIN
        COMMON /TELCOR/ COFDEG(20),COFRAD(20),NPARM
        COMMON /LSTSQR/ NMLMAT(20,20),COEF0(20),COCORR(20),COVMAT(20,20),
     X                  LHS(20), SD, SOLVEC(20),NCOEF
        INTEGER SOLVEC
        REAL NMLMAT,LHS
        COMMON /LABELS/ LABLIS(20,20),M,N
        COMMON /MATCMN/ DLODCO(20),DLADCO(20),WLO,WLA
        REAL TEMP(200)
C-----------------------------------------------------------------------
```

```
      IF (NPASS.GT.1) GO TO 10
C         CALL PRTIME(12)
          WRITE(12,1007) HDRLIN
1007      FORMAT(" DATA FILE HEADER:"/ 1X,40A2)
          WRITE(12,1002)
1002      FORMAT(/31X,"OBSERVATIONS"/
     X         3X,"#",2X,"MODEL",10X,"OBSERVED",16X,"PREDICTED"/
     X         11X,2(3X,"LONGITUDE",4X,"LATITUDE")/1X,58("-")/)
          DO 5 II=1,NOBS
              OLO = OLON(II)/RDPDG
              OLA = OLAT(II)/RDPDG
              CLO = CLON(II)/RDPDG
              CLA = CLAT(II)/RDPDG
              WRITE(12,1001) II,MODEL(II),OLO,OLA,CLO,CLA
1001          FORMAT(1X,I3,3X,I3,1X,4(2X,F10.5))
5         CONTINUE
          WRITE(12,1000) (SOLVEC(II),II=1,NCOEF)
1000      FORMAT(/" PARAMETERS IN SOLUTION: ", 20(I2,1X))
C          WRITE(10,1003)
C1003      FORMAT(/4X,"ITERATION",8X,"STANDARD DEVIATION"/4X,35("-"))
C  CONVERT POSTFIT RESIDUALS TO ARCSEC AND PRINT
10      MPASS = IABS(NPASS)
        WRITE(12,1004) MPASS
1004    FORMAT(/" // PASS NUMBER ",I2/)
        DO 15 KK=1,NOBS
15      TEMP(KK) = OCLONP(KK)/RDPAS
        WRITE(12,1005) (TEMP(II),II=1,NOBS)
1005    FORMAT(/" LONGITUDINAL POSTFIT O-C RESIDUALS (ARCSEC):",
     X          40(/5(2X,E15.5)))
        DO 16 KK=1,NOBS
16      TEMP(KK) = OCLATP(KK)/RDPAS
        WRITE(12,1006) (TEMP(II),II=1,NOBS)
1006    FORMAT(/" LATITUDINAL POSTFIT O-C RESIDUALS (ARCSEC):",
     X          40(/5(2X,E15.5)))
C  CONVERT CORRECTIONS AND COEFFICIENTS TO DEGREES AND PRINT
        DO 17 KK=1,NPARM
        TEMP(KK) = COCORR(KK)/RDPDG
17      COFDEG(KK) = COFRAD(KK)/RDPDG
        WRITE(12,1010) ((TEMP(II),COFDEG(II),(LABLIS(JJ,II),JJ=1,M)),II=1,NPARM)
1010    FORMAT(/"   CORRECTION   NEW VALUE      COEFFICIENT"
     X          20(/1X,2(2X,F10.4),5X,20A2))
C  NOW FINISH OFF WITH THE STANDARD DEVIATION.
        SDD = SQRT(SD)/RDPDG
        WRITE(12,1020) SDD
1020    FORMAT(/" STANDARD DEVIATION OF NEW FIT: ", F10.4," DEGREE")
C        WRITE(10,1022) MPASS,SDD
C1022    FORMAT(8X,I2,15X,F10.4)
        RETURN
        END
```

```
 -0.94764035    2.05865310    0.85946807    0.41295492   -0.00008342   -0.000124
  1.79861926    2.26065368   -2.67738781    0.61500631   -0.00007932   -0.000039
```

1.98691368	2.39341470	-2.48897172	0.74776513	-0.00004787	-0.000085
1.51131678	2.87810115	-2.96234952	1.23246766	-0.00007308	-0.000158
1.47684547	2.33471098	-2.99907615	0.68917725	-0.00006555	0.000056
0.72826524	2.82681082	2.53686517	1.18132021	0.00017184	-0.000022
0.19022918	3.02674254	2.00080639	1.38136093	0.00001675	0.000038
-0.70493316	2.65921451	1.10277337	1.01384986	-0.00015939	0.000053
0.08570418	2.19720883	1.89284098	0.55166436	-0.00013322	0.000041
-0.97465996	1.97152179	0.83237986	0.32594292	-0.00011641	0.000023
-1.41203096	2.77474467	0.39606017	1.12942015	-0.00006360	0.000046
-1.36167928	2.57611555	0.44605433	0.93072816	0.00008396	0.000022
-1.79847559	2.25372876	0.00894293	0.60821791	0.00015262	-0.000018
-2.89200844	2.20135612	-1.08452317	0.55585962	0.00027847	0.000049
-2.27913927	2.59498427	-0.47141947	0.94960052	0.00012220	0.000027
-2.31506644	2.02266846	-0.50764951	0.37715784	0.00027477	0.000068
-1.66882743	2.77524272	0.13920039	1.12991441	-0.00008761	0.000039
-0.32192704	2.04730314	1.48507246	0.40167077	-0.00019113	-0.000021

DATA FILE HEADER:

		OBSERVATIONS			
#	MODEL	OBSERVED		PREDICTED	
		LONGITUDE	LATITUDE	LONGITUDE	LATITUDE
1	1	-54.29579	117.95213	49.24389	23.66057
2	1	103.05329	129.52591	-153.40302	35.23727
3	1	113.84177	137.13256	-142.60757	42.84379
4	1	86.59207	164.90305	-169.78742	70.61520
5	1	84.61701	133.76909	-171.83441	39.48695
6	1	41.72652	161.96433	145.35167	67.68466
7	1	10.89933	173.41957	114.63776	79.14615
8	1	-40.38969	152.36177	63.18426	58.08932
9	1	4.91049	125.89079	108.45180	31.60804
10	1	-55.84390	112.95988	47.69185	18.67515
11	1	-80.90341	158.98116	22.69258	64.71101
12	1	-78.01848	147.60055	25.55703	53.32680
13	1	-103.04506	129.12915	0.51239	34.84832
14	1	-165.69988	126.12841	-62.13860	31.84841
15	1	-130.58506	148.68165	-27.01035	54.40810
16	1	-132.64354	115.89037	-29.08617	21.60955
17	1	-95.61677	159.00969	7.97559	64.73933
18	1	-18.44506	117.30183	85.08838	23.01404

PARAMETERS IN SOLUTION: 1 2 3 4 5 6 7 8

// PASS NUMBER 4

LONGITUDINAL POSTFIT O-C RESIDUALS (ARCSEC):

-0.16690D 2 -0.15613D 2 -0.95963D 1 -0.10901D 2 -0.12517

0.38702D	2	0.90600D	1	-0.34610D	2	-0.27012D	2	-0.22973
-0.16932D	2	0.14618D	2	0.30109D	2	0.57058D	2	0.21600
0.56954D	2	-0.22672D	2	-0.38580D	2			

LATITUDINAL POSTFIT O-C RESIDUALS (ARCSEC):

-0.24625D	2	-0.94444D	1	-0.19171D	2	-0.33552D	2	0.10830
-0.41916D	1	0.90959D	1	0.12230D	2	0.96590D	1	0.59878
0.10122D	2	0.53132D	1	-0.38382D	1	0.84729D	1	0.47813
0.13108D	2	0.83359D	1	-0.31201D	1			

CORRECTION	NEW VALUE	COEFFICIENT
-0.0000	103.5014	1 AZIMUTH AXIS ENCODER OFFSET
0.0000	-94.2954	2 ALTITUDE AXIS ENCODER OFFSET
0.0000	-0.0033	3 X AXIS ROTATION COMPONENT
-0.0000	0.0010	4 Y AXIS ROTATION COMPONENT
0.0000	0.0358	5 TRANSVERSE MISALIGNMENT
-0.0000	0.0214	6 LATERAL MISALIGNMENT
-0.0000	0.0057	7 AZ & ALT AXIS NONORTHOGONALITY
-0.0000	-0.0002	8 LATCH FLEXURE
0.0000	0.0000	9 ALTITUDE ENCODER WOBBLE
0.0000	0.0000	10 ALTITUDE ENCODER PHASE (ENTER ONLY)

STANDARD DEVIATION OF NEW FIT: 0.0064 DEGREE

JPL Ephemeris Implemented on a DG NOVA Computer

by

Randall L. Ricklefs
Department of Astronomy
University of Texas
Austin, TX 78712

ABSTRACT

The standard JPL export ephemeris package has been implemented on a Data General NOVA minicomputer operating under RDOS. Particular problems in conversion include the incompatibility of 24000 character blocks with the local tape drive, the maximum size of integer that can be accommodated in a 16 bit word, the unavailability of DECODE/ENCODE statements in D.G.'s FORTRAN and differences in binary READ's and WRITE's.

1 Introduction

The standard JPL ephemeris software package provides a means of reformatting, accessing, and otherwise manipulating the JPL ephemerides. This package, written in FORTRAN, has been implemented on many large main-frames in the past, but our requirements for the McDonald Laser Ranging System (MLRS) necessitated its implementation on a minicomputer, specifically a Data General NOVA 4 running under RDOS. This paper outlines the conversion steps required. The reader should consult JPL's document on this package (Newhall,1976) and D.G.'s FORTRAN manual (Data General, 1978) for more information. The specific ephemeris used in our conversion was DE-111/LE-55, although any of JPL's export ephemerides should work as well.

In short, the ephemeris package contains the programs BCDEPH, XSHORT, TDUMP, as well as the READE subroutine set. BCDEPH converts the BCD ephemeris tape into a binary file; XSHORT creates a restricted time-span binary file for easy access; TDUMP prints operator-specified parts of the binary file; and TSTRDE exercises the READE package and prints the results. The READE package accesses and interpolates the ephemeris to obtain the position and velocity of a

2 Conversion Problems

The first problem encountered was that the unpacked ASCII ephemeris file was blocked by 24000 characters (300 80-character lines). Our Digi-Data tape drive cannot handle physical records larger than 8192 characters. Hence, the tape had to be reformatted in 8000-character records on the University of Texas' dual CDC Cyber 170/750 system.

Integers with magnitudes larger than 32767 cannot be handled in D.G. 16 bit integer words. Larger integers must be handled as double precision reals to retain accuracy. This problem appeared with the data "KEY1" header 999999 which was changed to 9999. The integer group with "KEY1"=1050 contains integers greater than the acceptable value. Hence, this was converted to a double precision group by our version of BCDEPH. Many large integer variables and constants in various of the routines were also changed to double precision real.

Since NOVA words are 2 characters long rather than 6 as on the JPL UNIVAC, the variable "C" which indicates the length of BCD records had to be tripled by BCDEPH for output to the binary ephemeris file.

NOVA FORTRAN IV does not support some of the options assumed by JPL's software. DECODE/ENCODE, NAMELIST, and BACKSPACE are not available. DECODE, used in BCDEPH, was replaced by a parser (SCAN) that pulls numbers one at a time from a line in A1 format. ENCODE, which is used only in XSHORT was replaced by a write to a temporary file which is subsequently rewound and read with a different format. The BACKSPACE used by XSTATE to back up the ephemeris a certain number of records was replaced by a call to LH to rewind the tape and search again for the appropriate ephemeris record.

Since the ephemeris manipulation programs such as XSHORT and TDUMP will be used in an interactive rather than batch mode on the NOVA, the NAMELIST variables are now accepted from the operator via the console.

The JPL software assumes that an unformatted read or write such as

```
READ(IUNIT) N
```

is a binary read. Under NOVA FORTRAN one must replace these with READ BINARY or WRITE BINARY to get the same results. Also, under binary I/O, one must read an entire record at a time, as the next read starts where the last read ended. To read an entire binary record from a JPL ephemeris one must replace the example above with

```
READ BINARY(IUNIT) N,(A(K),K=1,N)
```

where N is the record length.

The routine XSTATE contains local variables which are assumed to be zero initially and are assumed to retain their values between calls. Since that is definitely not the case on the NOVA (all variables are put onto a run-time stack for the sake of reentrancy) these variables were put in common storage and explicitly initialized.

The NOVA interprets a variable dimensioned X(N1,N2,1) to have a third dimension of "1". Hence, referring to X(1,1,2), for example, results in an "out of bounds" error. This caused problems in the READE routines.

Other changes necessary to implement this ephemeris package were calls to system routines to open and close disk and tape files being used. The UNIVAC and CYBER, for example, automatically open and close these files.

A page of output from the TSTRDE program has been included. The entire output and a dump of part of the binary ephemeris tape comes with the ephemeris tape from JPL.

3 Conclusion

In conclusion, the ephemeris package was not difficult to implement once the FORTRAN language incompatibilities were isolated and resolved. The entire conversion required a month or so of part-time effort. Although this conversion is specifically oriented toward a NOVA, many of the problems will probably be similar for other brands of minicomputers.

4 References

Newhall,X.X., JPL Export Ephemeris DE-96 Users Guide, JPL, Pasadena, Jan. 21, 1976.

NOVA-LINE FORTRAN IV User's Manual, 093-000053-09, Southboro, Data General Corporation, 1978.

5 Appendix A

```
C     THIS PROGRAM READS THE BCD TAPE CONTAINING THE EXPORT
C     JPL PLANETARY EPHEMERIS AND RECONSTRUCTS THE ABSOLUTE
C     FORTRAN-READABLE BINARY TAPE EXPECTED BY THE INTERPOLATION
C     ROUTINE.  THE FORMAT OF THE BCD TAPE EXPECTED BY THIS
C     PROGRAM IS A FOLLOWS --
C
C        CHARACTERS (300 CARD IMAGES OF 80 CHARACTERS EACH).
C        THE MAIN BODY OF THIS PROGRAM WILL PROCESS ONE CARD
C        IMAGE AT A TIME AND USE THE FORTRAN 'DECODE' STATEMENT
C        TO FORM THE BINARY REPRESENTATION OF THE DATA.
C
C                        ***************
C                        *             *
C                        *  IMPORTANT  *
C                        *             *
C                        ***************
C
C        THE ABILITY TO READ THE BLOCKED, 300-CARD RECORDS
C        IS MACHINE-DEPENDENT.  THE FORTRAN 'READ' STATEMENTS
C        CANNOT DO THE TASK.  THEREFORE, YOU, THE USER, MUST
C        SUPPLY A FORTRAN-CALLABLE SUBROUTINE CALLED RCI,
C        WHOSE JOB WILL BE TO READ THIS BLOCKED TAPE (ONE RECORD
C        AT A TIME) AND TRANSMIT ONE CARD IMAGE PER CALL.
C        CALLING SEQUENCE --
C
C           CALL RCI(IMAGE,EOF)
C
C             IMAGE  IS AN ARRAY LARGE ENOUGH TO HOLD
C                    ONE CRD IMAGE(80 CHRACTERS)
C                    RCI IS TO FILL THIS ARRAY WITH
C                    THE NEXT SEQUENTIAL CARD IMAGE
C                    ON THE FILE.
C
C               EOF  IS A LOGICAL FLAG TO BE SET BY
C                    RCI.  IT IS TO BE SET TO .TRUE. IF
C                    AN ATTEMPT TO FETCH A CARD IMAGE
C                    RESULTS IN AN END-OF-FILE BEING
C                    READ.  OTHERWISE, IT SHOULD BE
C                    SET TO .FALSE. .
C
C
C     THE BINARY FILE WHICH THIS PROGRAM WRITES WILL BE ON
C     UNIT 12.  IT IS SUGGESTED THAT RCI BE CODED TO READ
C     THE BCD TAPE ON UNIT 9.
C
C          (O/P EPH ON UNIT 2, I/P ON 1. RLR 3/81)
C
C
C     IF YOU HAVE TROUBLE OR QUESTIONS ABOUT THIS EPHEMERIS
C     PACKAGE, CALL OR WRITE
```

```
C                                SKIP NEWHALL
C                                JET PROPULSION LABORATORY
C                                4800 OAK GROVE DRIVE
C                                PASADENA, CALIFORNIA  91103
C
C                                PHONE --
C                                  COMMERICAL --
C                                     (213) 354-7000
C                                  FTS --
C                                     792-7000
C
C
C       MODIFIED FOR NOVA4 BY R. RICKLEFS       3/81
C       UNIVERSITY OF TX, MCDONALD OBS
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
C
        COMMON /DIALOG/ IDUM,IPTR,IMAGE(80),DW(2),DPNUM
        INTEGER IPTR,IMAGE,DW,SCAN
        DOUBLE PRECISION DPNUM

        COMMON /GARB/ A(6000)
        INTEGER A

      INTEGER EPHNAM(12)
      INTEGER C,H(5),H2
      INTEGER NDP1(3)
C
      DOUBLE PRECISION NDP2(3),NDP3(3)
      DOUBLE PRECISION PW2(201)
      DOUBLE PRECISION B(1500)
      EQUIVALENCE(A,B)
C
      LOGICAL EOF
C
C
C       SET UP DOUBLE PRECISION TABLE OF POWERS OF 2 FROM
C       2**(-100) THROUGH 2**(100).  THIS TABLE IS NEEDED
C       TO DECODE DP DATA ON THE FILE.
C
      PW2(100)=.5D0
      PW2(101)=1.D0
      PW2(102)=2.D0
      DO 50 K=1,99
      PW2(100-K)=PW2(101-K)*PW2(100)
      PW2(102+K)=PW2(101+K)*PW2(102)
   50 CONTINUE

C  NOVA-DEPENDENT CODE TO OPEN INPUT& OUTPUT FILES FOLLOWS
        CALL INIT("MT0",0,IERR)
        IF (IERR.EQ.1) GO TO 51
            TYPE "INIT MT0 ERROR",IERR
            STOP
51      CALL MTOPD(1,"MT0:0",0,IERR)
```

```
        IF (IERR.EQ.1) GO TO 52
            TYPE "OPEN MT0 ERROR",IERR
            STOP
52      WRITE(10,1000)
1000    FORMAT(" ENTER OUTPUT EPHEMERIS FILE NAME: ",Z)
        READ(11,1001) EPHNAM
1001    FORMAT(12A2)
        CALL OPEN(2,EPHNAM,2,IERR)
        IF (IERR.EQ.1) GO TO 53
            TYPE "OUTPUT EPHEMERIS OPEN ERROR",IERR
            STOP

53      REWIND 2
C
C       READ 5-WORD INTEGER HEADER RECORD
C
    1 CONTINUE
      CALL RCI(IMAGE,EOF)
      IF(EOF) GO TO 99
C
C       DECODE AND WRITE ONTO OUTPUT UNIT
C
C     DECODE(103,IMAGE)C,(H(K),K=1,5)
        C=5
        IPTR=1
        DO 55 K=1,5
            II=SCAN(IPTR,IMAGE,DW,DPNUM)
            IF (DPNUM.LE.36535.D0) GO TO 55
            WRITE(10,1010) IMAGE
1010        FORMAT(" INTEGER TOO LARGE"/80A1)
            DW(2)=9999
55          H(K)=DW(2)
C  IF THIS IS GROUP 1050, IT WILL BE D.P., NOT INTEGER, ON OUTPUT
        H2=H(2)
        IF (H(4).EQ.1050) H2=2
      WRITE BINARY(2) H(1),H2,H(3),H(4),H(5)
      WRITE(12,105)H
  105 FORMAT('0 HEADER:',6I12)
C
C       IF THIS IS THE LAST HEADER ON THE FILE, QUIT.
C
      IF(H(2).NE.5) GO TO 2
    3 ENDFILE 2
      REWIND 2
      WRITE(12,106)
  106 FORMAT('0 ALL DONE.')
        CALL FCLOS(1)
        CALL FCLOS(2)
      STOP
C
C       OTHERWISE, WORD 2 OF HEADER INDICATES TYPE OF
C       DATA THAT FOLLOWS IN THIS GROUP.  READ FIRST
C       CARD IMAGE
C
```

```
    2 CALL RCI(IMAGE,EOF)
      IF(EOF) GO TO 99
      IT=H(2)-1
      GO TO (11,21,31),IT
C
C       THIS BLOCK HANDLES DOUBLE PRECISION DATA.  EACH CARD IMAGE
C       HAS UP TO 3 DP NUMBERS, EACH NUMBER BEING ENCODED AS A
C       SERIES OF 3 INTEGERS.  THE INTERPRETATION OF THESE 3 INTEGERS
C       IS --    DP NO. = N2*(2**(N1-30))+N3*(2**(N1-60)).
C
C  11 DECODE(102,IMAGE)C,((NDP(K1,K2),K1=1,3),K2=1,3)
C 102 FORMAT(I5,3(I3,2I11))
11      IPTR=1
        II=SCAN(IPTR,IMAGE,DW,DPNUM)
        C=DW(2)
        CALL DECDP(NDP1,NDP2,NDP3)
      NWB=0
      DO 12 I=1,C,3
      IF(I.EQ.1) GO TO 13
      CALL RCI(IMAGE,EOF)
C     DECODE(102,IMAGE)NDUM,((NDP(K1,K2),K1=1,3),K2=1,3)
        IPTR=1
        CALL DECDP(NDP1,NDP2,NDP3)
   13 NW=MIN0(3,C-I+1)
      DO 14 K=1,NW
      NX=71+NDP1(K)
      NWB=NWB+1
      IF(NDP2(K).NE.0.D0) GO TO 16
      B(NWB)=0.D0
X       WRITE(12,1010) NWB,B(NWB)
      GO TO 14
   16 IF(NDP1(K).GT.-41) GO TO 15
      B(NWB)=NDP2(K)*(2.D0**(NDP1(K)-30))
     1      +NDP3(K)*(2.D0**(NDP1(K)-60))
X       WRITE(12,1010) NWB,B(NWB)
      GO TO 14
   15 B(NWB)=NDP2(K)*PW2(NX)+NDP3(K)*PW2(NX-30)
        NNN=NX-30
X       WRITE(12,1010) NWB,B(NWB),NX,NDP2(K),PW2(NX),NDP3(K),PW2(NNN)
X1010   FORMAT(5X,I5,E25.18/5X,I5,4E25.18)
   14 CONTINUE
   12 CONTINUE
C
C       WRITE RECORD AND CHECK FOR END-OF-GROUP TRAILER
C
      WRITE BINARY(2)C,(B(K),K=1,C)
      IF((C.EQ.1).AND.(B(1).EQ.0)) GO TO 1
      GO TO 2
C
C
C       THIS BLOCK HANDLES INTEGER DATA
C
C  NOTE: INTEGER DATA CAN BE >36535 ALLOWED IN 16 BIT D.G. WORD.
C        HENCE, THIS INTEGER DATA IS OUTPUT AS DOUBLE PRECISION REAL.
```

```
C        RLR 3/81
C
C  21 DECODE(103,IMAGE)C,(A(K),K=1,6)
C 103 FORMAT(I5,1X,6I12)
21       IPTR=1
         II=SCAN(IPTR,IMAGE,DW,DPNUM)
         C=DW(2)
         DO 210 K=1,6
             II=SCAN(IPTR,IMAGE,DW,B(K))
             IF (II.LT.0) GO TO 211
210      CONTINUE
211    DO 22 I=1,C,6
       IF(I.EQ.1) GO TO 23
       CALL RCI(IMAGE,EOF)
C      DECODE(103,IMAGE)NDUM,(A(I+K-1),K=1,6)
         IPTR=1
         DO 215 K=1,6
             II=SCAN(IPTR,IMAGE,DW,B(I+K-1))
             IF (II.LT.0) GO TO 23
215      CONTINUE
   23 CONTINUE
   22 CONTINUE
C
C        WRITE RECORD AND CHECK FOR TRAILER
C
X        WRITE(12,1010) (B(K),K=1,C)
X1010    FORMAT(5X,10F10.0)
       WRITE BINARY(2)C,(B(K),K=1,C)
       IF(C.EQ.1) GO TO 1
       GO TO 2
C
C
C        THIS BLOCK HANDLES BCD DATA
C
C  31 DECODE(104,IMAGE)C,(A(K),K=1,12)
C 104 FORMAT(I5,1X,12A6)
31       IPTR=1
         II=SCAN(IPTR,IMAGE,DW,DPNUM)
         C=DW(2)
         DO 310 K=1,36
         KK=2*(K-1)+7
310      A(K)=(IMAGE(KK).AND.177400K).OR.(IMAGE(KK+1)/400K)
       DO 32 I=1,C,12
       IF(I.EQ.1) GO TO 33
       CALL RCI(IMAGE,EOF)
C      DECODE(104,IMAGE)NDUM,(A(I+K-1),K=1,12)
         DO 315 K=1,36
         KK=2*(K-1)+7
315      A(3*I+K-3)=(IMAGE(KK).AND.177400K).OR.(IMAGE(KK+1)/400K)
   33 CONTINUE
   32 CONTINUE
C      GO TO 25
C
C  UNIVAC WORD IS 6 BYTES, D.G. WORD IS 2, HENCE WE MULTIPLY BY 3:
```

```
        C=3*C
        WRITE BINARY(2) C,(A(K),K=1,C)
        IF (C.EQ.3) GO TO 1
        GO TO 2
C
C       ERROR EXIT FOR END-OF-FILE READ
C
   99 WRITE(12,110)
  110 FORMAT('0 UNEXPECTED END OF FILE ENCOUNTERED.')
      GO TO 3
C
C
      END

        SUBROUTINE RCI(IMAGE,EOF)
C  RCI READ'S A CARD IMAGE FROM THE INPUT JPL BCD EPHEMERIS AND
C  RETURNS IT TO BCDEPH.
C
C  INPUT (FROM TAPE):
C       BLK   -- 4000  WORD (100 80 CHAR LINE) RECORDS (ONE AT A TIME)
C
C  OUTPUT:
C       IMAGE -- 80 CHARACTER LINE IMAGE IN A1 FORMAT,
C       EOF   -- LOGICAL END OF FILE FLAG. TRUE IF EOF OR READ ERROR ON BCD
C                EPHEMERIS FILE. THIS WILL NOT BE SET FOR A NORMAL EOF WHICH IS
C                DETECTED IN BCDEPH.
C
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$ RLR 4/81

        LOGICAL EOF
        INTEGER IMAGE(80)

        COMMON /RCICMN/ II,BLK(4096),NREAD
        INTEGER II,BLK,NREAD
        LOGICAL FRSTIM

        DATA II,NREAD /0,0/

        IF (II.NE.NREAD) GO TO 20

C  READ A NEW RECORD FROM TAPE
        K=0
        NREAD=0

C  USE NOVA MAG TAPE I/O (NEEDED TO READ 4000 WORD BLOCKS RATHER THAN 256)
        CALL MTDIO(1,K,BLK,ISTAT,IERR,NREAD)
        IF (IERR.EQ.1) GO TO 15
            WRITE(10,1020) IERR,ISTAT,NREAD
1020        FORMAT(" READ ERROR: ",I3,", STATUS WORD: ",OI8,
     X              ", WORDS READ: ",I8)
            EOF= .TRUE.
            RETURN

15      II= 0
```

```
C  TRANSFER A LINE IMAGE FROM BLK TO "IMAGE" FOR CALLER
C  (CONVERTS FROM A2 TO A1 FORMAT.)
20      DO 25 K=1,80,2
        II= II+ 1
        IMAGE(K)= BLK(II).AND.177400K.OR.40K
        IMAGE(K+1)= (BLK(II).AND.377K)*400K.OR.40K
25      CONTINUE
        EOF = .FALSE.

X       WRITE(10,1000) II, IMAGE, IMAGE
X1000   FORMAT(I4/80A1/8(10(1X,OI6)/))
        RETURN

        END



        SUBROUTINE DECDP(NDP1,NDP2,NDP3)
C
C  ROUTINE DECODES 3 SETS OF D.P. NUMBER COMPONENTS
C
        INTEGER NDP1(3)
        DOUBLE PRECISION NDP2(3),NDP3(3)

        COMMON /DIALOG/ IDUM,IPTR,IMAGE(80),DW(2),DPNUM
        INTEGER IPTR,IMAGE,DW,SCAN
        DOUBLE PRECISION DPNUM

        DO 10 K=1,3
        II=SCAN(IPTR,IMAGE,DW,DPNUM)
        IF (II.LT.0) RETURN
        NDP1(K)=DW(2)
        II=SCAN(IPTR,IMAGE,DW,NDP2(K))
        II=SCAN(IPTR,IMAGE,DW,NDP3(K))
10      CONTINUE

        RETURN
        END



        INTEGER FUNCTION SCAN(INDEX,CHARAY,DINUM,DPNUM)

C  THIS FUNCTION CONVERTS AN ASCII-CHARACTER ARRAY IN A1 FORMAT TO A
C  SIGNED 2-WORD INTEGER & A DOUBLE-PRECISION REAL.  THE USER PASSES
C  THE CHARACTER ARRAY OR IT CAN BE READ FORM THE CONSOLE DEVICE.  WITH
C  THE FORMER AN INDEX TO THE 1ST ARRAY MEMBER TO BE CONVERTED IS ALSO
C  PASSED.
C
C  ANY LEADING BLANKS ARE IGNORED.  THE NUMBER MAY BE PRECEDED BY AN
C  ASCII PLUS OR MINUS SIGN.  CONVERSION IS TERMINATED BY A BLANK OR
C  WHEN THE INDEX REACHES 80, WHICHEVER COMES 1ST.  IF THE CHARACTER
C  IMMEDIATELY FOLLOWING THE NUMBER IS NOT A BLANK NO CONVERSION IS
```

```
C  PERFORMED.  IF THE NUMBER HAS A FRACTIONAL PART (INDICATED BY A
C  DECIMAL POINT) THE 2-WORD INTEGER WILL CONTAIN ONLY THE INTEGER
C  PORTION, WHILE THE DOUBLE-PRECISION REAL WILL REFLECT BOTH PARTS.
C
C  ON INPUT:
C       CHARAY - (INTEGER ARRAY) CHARACTER STRING IN A1 FORMAT (CHARACTER
C                IN LEFT BYTE, BLANK IN RIGHT BYTE OF EACH WORD).  NUMBER
C                MUST BE FOLLOWED BY AT LEAST ONE NON-DIGIT.
C       INDEX  - (INTEGER) POSITION IN CHARAY TO BEGIN CONVERSION,
C                IF = -1, CHARAY WILL BE READ FROM $TTI
C
C  ON OUTPUT:
C       CHARAY - IF INDEX WAS = -1 WILL CONTAIN LINE READ FROM $TTI,
C                OTHERWISE UNCHANGED
C       SCAN   - (INTEGER) RETURN INFORMATION SWITCH
C                =-1 IF LINE CONTAINED ONLY BLANKS
C                = 0 IF NUMBER WAS CONVERTED
C                = INDEX OF NON-DIGIT CHARACTER OTHERWISE
C       INDEX  - POSITION IN CHARAY FOLLOWING LAST DIGIT CONVERTED
C       DINUM  - (2-WORD INTEGER) CONTAINS NUMBER IF CONVERSION
C                WAS PERFORMED
C       DPNUM  - (DOUBLE-PRECISION REAL) CONTAINS NUMBER IF
C                CONVERSION WAS PERFORMED
C
C  CALLS: DWMPY,DWADD,DWSUB,DWFL
C     NOTE: THE ABOVE ROUTINES ARE FOUND IN THE NOVA FORTRAN 4
C       COMMERCIAL SUBROUTINE PACKAGE. THEY PERFORM DOUBLE WORD INTEGER ARITH-
C       METIC. DWMPY=MULTIPLY,  DWADD=ADD,  DWSUB=SUBTRACT,
C       DWFL=DOUBLE WORD INTEGER TO DOUBLE PRECISION FLOATING POINT CONVERSION.
C                                                                      RWH
C$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$

      INTEGER CHARAY(1),NCHK,SIGNSW,DWDGT(2),DW10(2),INDEX,DINUM(2)
      DOUBLE PRECISION DPNUM,DPTMP

C  INITIALIZATIONS
      NCHK= 0          ;STATUS RETURNED FROM DOUBLE-INTEGER ROUTINES
      SIGNSW= 0        ;SET NON-ZERO IF MINUS SIGN FOUND
      DWDGT(1)= 0      ;UPPER HALF OF CONVERTED DIGIT
      DPNUM= 0         ;DOUBLE-PRECISION RUNNING TOTAL
      DINUM(1)= 0      ;DOUBLE-INTEGER RUNNING TOTAL
      DINUM(2)= 0
      DW10(1)= 0       ;DOUBLE-INTEGER 10 FOR MULTIPLICATION
      DW10(2)= 10

C  IF INDEX = -1, READ LINE FROM $TTI
      IF(INDEX.NE.-1)GO TO 2
 888  READ(11,777,ERR=888,END=888)(CHARAY(I),I=1,80)
 777  FORMAT(80A1)
      INDEX= 1

C  SKIP TO 1ST NON-BLANK CHARACTER
 2    IF(CHARAY(INDEX).NE."  ")GO TO 3
      INDEX= INDEX+1
```

```
      IF(INDEX.LE.80)GO TO 2

C  HERE LINE CONTAINED ONLY BLANKS
      SCAN= -1
      GO TO 12

C  IF 1ST NON-BLANK IS PLUS OR MINUS, SET SIGN FLAG & BUMP PTR
 3    IF((CHARAY(INDEX).NE."- ").AND.(CHARAY(INDEX).NE."+ "))GO TO 4
      IF(CHARAY(INDEX).EQ."- ")SIGNSW= 1
      INDEX= INDEX+1

C  CONVERT CHAR DIGITS OF INTEGER PART TO DOUBLE-WORD INTEGER
 4    IF((CHARAY(INDEX).LT."0 ").OR.(CHARAY(INDEX).GT."9 "))GO TO 5
      DWDGT(2)= ISHFT(CHARAY(INDEX)-"0 ",-8)
      CALL DWMPY(DW10,DINUM,NCHK)
      CALL DWADD(DWDGT,DINUM,NCHK)
      IF(NCHK.NE.0)GO TO 11
      INDEX= INDEX+1
      IF(INDEX.GT.80)GO TO 9
      GO TO 4

C  IF NEXT CHAR IS DECIMAL PT, CONVERT FRACTIONAL PART STORING IN DPNUM
 5    IF(CHARAY(INDEX).NE.". ")GO TO 8
      DPTMP= 10D0
 6    INDEX= INDEX+1
      IF(INDEX.GT.80)GO TO 9
      IF((CHARAY(INDEX).LT."0 ").OR.(CHARAY(INDEX).GT."9 "))GO TO 8
      IF(CHARAY(INDEX).EQ."0 ")GO TO 7
      DPNUM= DPNUM+DFLOAT(ISHFT(CHARAY(INDEX)-"0 ",-8))/DPTMP
 7    DPTMP= DPTMP*10D0
      GO TO 6

C  IF NEXT CHAR ISN'T BLANK ERROR HAS OCCURRED
 8    IF(CHARAY(INDEX).NE."  ".AND.CHARAY(INDEX).NE."- ")GO TO 11

C  HERE CONVERSION FINISHED, CONVERT TO NEGATIVE IF MINUS SIGN WAS FOUND
 9    SCAN= 0
      IF(SIGNSW.EQ.0)GO TO 10
      DWDGT(2)= 0
      CALL DWSUB(DINUM,DWDGT,NCHK)
      DINUM(1)= DWDGT(1)
      DINUM(2)= DWDGT(2)

C  ADD INTEGER PART TO DPNUM
 10   CALL DWFL(DINUM,DPTMP)
      IF(SIGNSW.EQ.0)DPNUM= DPNUM+DPTMP
      IF(SIGNSW.NE.0)DPNUM= DPTMP-DPNUM
      GO TO 12

C  HERE CHAR WHICH WASN'T PLUS, MINUS, DIGIT, OR BLANK WAS ENCOUNTERED
 11   SCAN= INDEX

 12   RETURN
```

```
      END



C     THIS PROGRAM READS A PLANETARY EPHEMERIS FILE AND CREATES A
C    SHORTENED VERSION.  THE INPUT FILE SHOULD BE ON FORTRAN LOGICAL
C     UNIT 9, AND THE OUTPUT FILE IS ON UNIT 2.
C
C     THE USER MUST INPUT START AND/OR STOP EPOCHS VIA THE CONSOLE.
C     IF EITHER EPOCH IS NOT INPUT, THE
C     EPOCH ON THE INPUT FILE IS USED AS A DEFAULT.
C
C     THE USER HAS A CHOICE OF TWO FORMATS TO EXPRESS EITHER EPOCH:
C
C       (1) VIGESIMAL, AN ENCODED FORM OF GREGORIAN CALENDAR DATE.
C           THE GENERAL FORM OF A VIGESIMAL DATE IS 2 INTEGERS:
C
C                          YYYYMM00DD,HHNNSSFFFF
C
C           WHERE YYYY = YEAR
C                   MM = MONTH NUMBER
C                   DD = DAY NUMBER IN MONTH
C                   HH = HOUR IN DAY
C                   NN = MINUTES
C                   SS = SECONDS
C                 FFFF = TENTHS OF MILLISECONDS
C
C            EXAMPLE:  THE EPOCH JULY 4, 1976, 13:21:17 WOULD BE
C            EXPRESSED AS  1976070004,1321170000
C
C
C       (2) JULIAN EPHEMERIS DATE
C
C
C     IF VIGESIMAL EPOCH IS INPUT, USE THE INPUT SYMBOL VEP.  THE START
C     EPOCH DESIRED ON THE OUTPUT FILE SHOULD BE IN VEP(1) AND VEP(2).
C     THE STOP EPOCH SHOULD BE IN VEP(3) AND VEP(4).  IF JULIAN DATE
C     FORMAT IS WANTED, USE THE DOUBLE PRECISION INPUT SYMBOL EP(1) FOR
C     START ND EP(2) FOR STOP.  EITHER EPOCH MAY BE IN EITHER FORMAT.
C
C       MODIFIED FOR D.G. NOVA BY R. RICKLEFS   3/81
C       UNIVERSITY OF TEXAS, MCDONALD OBS.
C
      DOUBLE PRECISION EP(2),VSEC
      DOUBLE PRECISION SS(6),D1
      DOUBLE PRECISION D(1000)
C
      INTEGER DW(2),ANS(80),SCAN
C     INTEGER VEP(4)
      INTEGER H(5),H4(8),EL(6),N1,EOF(5)
      INTEGER A(4000)
      INTEGER FILNAM(12)
```

```
      COMMON /SECTOR/ H, IFILL(2)
      COMMON /XSHCMN/ D,EOF,EL,N1,D1,H4

      EQUIVALENCE(A,D)
      DATA EOF/2,5,0,1,0/
      DATA EL/' EARLI   LAT'/,N1/1/,D1/1.D0/
      DATA H4/9999,1010,1020,1030,1040,1041,1050,1070/
C
C
      M=2
      J=9
      WRITE(10,110)
110   FORMAT(10X,"EPHEMERIS SHORTENING PROGRAM"/
     X          " ENTER INPUT (LONG) EPHEMERIS FILE NAME: ",Z)
      READ(11,105) FILNAM
      CALL OPEN(J,FILNAM,2,IERR)
        IF (IERR.NE.1) CALL ERROR(1,IERR)

      WRITE(10,115)
115   FORMAT(" ENTER OUTPUT (SHORT) EPHEMERIS FILE NAME: ",Z)
      READ(11,105) FILNAM
      CALL OPEN(M,FILNAM,2,IERR)
        IF (IERR.NE.1) CALL ERROR(2,IERR)
      CALL OPEN(3,"XTEMPX",2,IERR)
        IF (IERR.NE.1) CALL ERROR(3,IERR)
      REWIND J
C
C       SPACE INPUT FILE TO GROUP 1010 TO GET EXISTING LIMITS
C       FOR PRINTING AND THEN TO 1030 TO READ DP LIMITS
C
      CALL LH(J,1010)
      DO 1 I=1,3
      READ BINARY(J)N,(A(K),K=1,N)
      WRITE(12,101)(A(K),K=1,N)
  101 FORMAT(5X,42A2)
    1 CONTINUE
      READ BINARY(J)N,(A(K),K=1,N)
      CALL LH(J,-1030)
      READ BINARY(J)N,(SS(K),K=1,N)
      REWIND J
C
C       PRINT MESSAGE AND READ INPUTS
C
19    WRITE(10,102)
  102 FORMAT('0 ENTER START AND STOP EPOCH (JULIAN DATE OR YYYYMMOODD.HHMMSS).')
C     READ(5,IN)
      IPTR=-1
      IF (SCAN(IPTR,ANS,DW,EP(1))) 22,20,19
20    IF (SCAN(IPTR,ANS,DW,EP(2))) 22,22,19
22    DO 2 I=1,2
C     VEP(I*2)=0
C     IF(VEP(I*2-1).EQ.0) GO TO 3
C     CALL VIGSEC(VEP(I*2-1),EP(I))
C     EP(I)=EP(I)/86400.D0+2433282.5D0
```

```
      IF (EP(I).LT.2500000.D0) GO TO 3
          CALL VIGSEC(EP(I),VSEC)
          EP(I) = VSEC/86400.D0 + 2433282.5D0
    3 CONTINUE
      IF(EP(I).EQ.0.D0) EP(I)=SS(I)
    2 CONTINUE
X       TYPE "EPOCHS:",EP
C
C       BE SURE INPUT EPOCHS ARE WITHIN SPAN ON TAPE, AND SET LIMITS
C
      EP(1)=DMAX1(EP(1),SS(1))
      EP(2)=DMIN1(EP(2),SS(2))
      SS(1)=SS(1)+DINT((EP(1)-SS(1))/SS(4))*SS(4)
      SS(2)=SS(2)-DINT((SS(2)-EP(2))/SS(4))*SS(4)
C
C       NOW READ FROM INUT FILE TO OUTPUT FILE, MAKING
C       CHANGES AS WE GO
C
C       COPY HEADER RECORD
C
      REWIND M
    4 READ BINARY(J)H
      WRITE BINARY(M)H
      WRITE(12,103)H
  103 FORMAT('0 HEADER ',6I12)
C
C       WORD 4 OF HEADER IS KEY1 -- TELLS US WHAT DATA IS IN GROUP
C
      DO 5 I=1,8
      IP=I
      IF(H(4).EQ.H4(I)) GO TO 6
    5 CONTINUE
      IP=0
    6 GO TO (11,12,11,13,11,14,14,15),IP
C
C       THIS BLOCK COPIES A BCD OR INTEGER GROUP
C
   11 READ BINARY(J)N,(A(K),K=1,N)
      WRITE BINARY(M)N,(A(K),K=1,N)
      N=N/3
      IF(N.EQ.1) GO TO 4
      GO TO 11
C
C       THIS BLOCK ENCODES AND WRITES A NEW TITLE GROUP (1010)
C
   12 READ BINARY(J)N,(A(K),K=1,N)
      WRITE BINARY(M)N,(A(K),K=1,N)
      WRITE(12,106)(A(K),K=1,N)
  106 FORMAT('0 LABEL GROUP ON NEW EPHEMERIS:',/,5X,42A2)
      DO 21 I=1,2
      REWIND 3
      READ BINARY(J)N,(A(K),K=1,N)
      D(101)=(SS(I)-2433282.5D0)*86400.D0
      CALL SECCAL(D(101),A(51))
```

```
      A(53)=A(53)+1900
C
C  ENCODE IS REPLACED BY WRITE TO & READ FROM TEMPORARY FILE.
C     ENCODE(104,A(1))EL(I),SS(I),(A(50+K),K=1,3),D(101)
      IIL=3*(I-1)+1
      IIH=3*I
      WRITE(3,104) (EL(II),II=IIL,IIH),SS(I),(A(K),K=51,53),D(101)
  104 FORMAT(1X,3A2,'EST EPOCH: JED=',F10.1,',',I3,'/',I2,'/',I4,
     1F13.0,' SECS PAST 1950',15X)
      REWIND 3
      READ(3,105) (A(K),K=1,N)
  105 FORMAT(42A2)
      WRITE BINARY(M)N,(A(K),K=1,N)
      WRITE(12,101)(A(K),K=1,N)
   21 CONTINUE
      GO TO 11
C
C       HERE WHEN WE ARE READY TO WRITE LIMITS GROUP (1030)
C
   13 READ BINARY(J)N,(D(K),K=1,N)
      WRITE BINARY(M)N,(SS(K),K=1,N)
      READ BINARY(J)N,D(1)
      WRITE BINARY(M)N,D(1)
      GO TO 4
C
C       THIS BLOCK COPIES DP RECORDS
C
   14 READ BINARY(J)N,(D(K),K=1,N)
      WRITE BINARY(M)N,(D(K),K=1,N)
      IF(N.GT.1) GO TO 14
      GO TO 4
C
C       THIS BLOCK HANDLES THE DP EPHEMERIS DATA RECORDS
C
   15 READ BINARY(J)N,(D(K),K=1,N)
      IF(N.EQ.1) GO TO 16
      IF(SS(1).LE.D(1)) WRITE BINARY(M)N,(D(K),K=1,N)
      IF(SS(2).GT.D(2)) GO TO 15
   16 WRITE BINARY(M)N1,D1
      WRITE BINARY(M)EOF
      END FILE M
      REWIND M
      CALL FCLOS(J)
      CALL FCLOS(M)
      CALL DELETE("XTEMPX")
      WRITE(12,107)
  107 FORMAT('0 ALL DONE.')
      STOP
      END


C*******************************
C
```

```
      SUBROUTINE LH(NU,KEY1)

C
C  READ TO THE BEGINNING OF RECORD TYPE KEY1 ON I/O UNIT NU.
C       KEY1 > 0 - READ FORWARD,
C       KEY1 < 0 - REWIND AND READ
C
C******************************
C
      INTEGER IH(5),NSEC,NRS,IBUF(7200)
      DOUBLE PRECISION BUF
C
      COMMON/SECTOR/IH,NSEC,NRS
      COMMON/CETB3/ BUF(1800)

      EQUIVALENCE (BUF,IBUF)
C
      KY=IABS(KEY1)
      IF(KEY1.GT.0) REWIND NU
      NSEC=0
 1001 READ BINARY(NU)IH
      NSEC=NSEC+1
X       TYPE " LH:",NSEC,IH
      IF(IH(4).EQ.KY) RETURN
 1002 READ BINARY(NU)N
      IF(IH(2).EQ.2) GO TO 10
C  BCD
      READ BINARY(NU) (IBUF(K),K=1,N)
      N=N/3
      GO TO 20
C  DOUBLE PRECISION
10    READ BINARY(NU) (BUF(K),K=1,N)
20    NSEC=NSEC+1
C       TYPE " ...",NSEC,N,BUF(1,1),BUF(2,1)
      IF(IH(3).EQ.1) GO TO 1001
      IF(N.EQ.1) GO TO 1001
      GO TO 1002
C
      END



C*************************
C
      SUBROUTINE SECCAL(SECS,N)
C
C*************************
C
C     THIS SUBROUTINE CONVERTS DOUBLE PRECISION FLOATING-POINT SECONDS
C     PAST 0H, JAN. 1, 1950 TO CALENDAR DATE.
C
      DOUBLE PRECISION T,SECS
      DOUBLE PRECISION AL,ANC4,ANC,ANLY,ANY
      INTEGER N(7)
```

```
C
      T=DINT((SECS+61530883200.D0)*1.D4+.5D0)
      N(4)=IDINT(DMOD(T/36.D6,24.D0))
      N(5)=IDINT(DMOD(T/60.D4,60.D0))
      N(6)=IDINT(DMOD(T/1.D4,60.D0))
      N(7)=IDINT(DMOD(T,1.D4))
C
      AL=DINT(T/864.D6)
      ANC4=DINT(AL/146097.D0)
      ANC=DMIN1(DMOD(AL,146097.D0)/36524.D0,3.D0)
      ANLY=DMOD(DMOD(AL,146097.D0),36524.D0)/1461.D0
      ANY=DMIN1(DMOD(DMOD(DMOD(AL,146097.D0),36524.D0),1461.D0)/365.D0,3.D0)
      ANC=DINT(ANC)
      ANLY=DINT(ANLY)
      ANY=DINT(ANY)
C
      N(2)=IDINT(AL-(ANC4*146097.D0+ANC*36524.D0+ANLY*1461.D0+ANY*365.D0))+1
      DO 8 I=1,12
      IF(N(2)-1.LT.MOD(I+(I-1)/5,2)+30) GO TO 9
    8 N(2)=N(2)-MOD(I+(I-1)/5,2)-30
    9 N(1)=MOD(I+1,12)+1
      N(3)=DINT(ANC4*400.D0+ANC*100.D0+ANLY*4.D0+ANY)+I/11-1900
C
C     END OF CALENDAR DATE CONVERSION
C
      RETURN
      END



C*****************************
C
      SUBROUTINE VIGSEC(V,S)
C
C*****************************
C
C     THIS SUBROUTINE CONVERTS A VIGESIMAL DATE TO
C     D.P. SECONDS PAST 0H, JAN. 1, 1950
C
C     VIGESIMAL FORMAT:  D.P. REAL  ..  YYYYMMOODD.HHNNSSXXXX
C       WHERE  YYYY = CALENDAR YEAR (E.G. 1973)
C                MM = CALENDAR MONTH (E.G. 04 = APRIL)
C                DD = CALENDAR DAY (E.G. 15)
C                HH = HOUR OF DAY (0 .LE. HH .LE. 23)
C                NN = MINUTES (0 .LE. NN .LE. 59)
C                SS = SECONDS (0 .LE. SS .LE. 59)
C              XXXX = TENTHS OF MILLISECONDS (0 .LE. XXXX .LE. 9999)
C
C     CALLING SEQUENCE:     CALL VIGSEC(V,S)
C        V IS THE INPUT VIG. DATE
C        S IS THE OUTPUT D.P. EQUIVALENT OF V IN SECONDS PAST 0H, 1/1/50
C
C
C
```

```
      DOUBLE PRECISION S, V, V2, NYMD
C
C
      IF (V.LT.30001D0) RETURN
C
C
C
C       CALCULATE THE NUMBER OF FULL YEARS, MONTHS, AND DAYS ALREADY
C       ELAPSED SINCE 0H, MARCH 1, -1 (UP TO MOST RECENT MIDNIGHT).
C       (THE '91' IS COMPLEMENT ARITHMETIC FOR ADDING 9 TO 'MM'.)
C
      NYMD = DINT(V-910001D0)
C
C       CONVERT 2ND WORD OF VIGESIMAL DATE AND ELAPSED DAYS TO
C       D.P. SECONDS
C
      V2 = (V-DINT(V))*1.D10
      S = DMOD(DINT(V2/1.D4),1.D2)
     X  + DMOD(DINT(V2/1.D6),1.D2)*60D0
     X  + DINT(V2/1.D8)*3.6D3
     X  + DMOD(NYMD,1.D2)*8.64D4
C
C
C       CALCULATE THE NUMBER OF FULL MONTHS ELAPSED SINCE
C       THE CURRENT OR MOST RECENT MARCH
C
      MTH = MOD(IDINT(DMOD(DINT(NYMD/1.D4),1.D2)),12)
C
C
C       CALCULATE NUMBER OF FULL 12-MONTH YEARS SINCE 3/1/-1
C
      IY = IDINT(NYMD/1.D6) + IDINT(DMOD(DINT(NYMD/1.D4),1.D2))/12
C
C
C       STARTING WITH MARCH AS MONTH 0 AND ENDING WITH THE FOLLOWING
C       FEBRUARY AS MONTH 11, THE CALCULATION OF THE NUMBER OF DAYS
C       PER MONTH REDUCES TO A SIMPLE FORMULA. THE FOLLOWING STATEMENT
C       DETERMINES THE NUMBER OF WHOLE DAYS ELAPSED SINCE 3/1/-1 AND
C       THEN SUBTRACTS THE 712163 DAYS BETWEEN THEN AND 1/1/50.  IT CONVERTS
C       THE RESULT TO SECONDS AND ADDS THE ACCUMULATED SECONDS ABOVE.
C
      S = DINT(DMOD(V2,1.D4))*1.D-4
     X  + S
     X  +(DFLOAT(IY)*365D0
     X  + DFLOAT((MTH+1+MTH/6+MTH/11)/2 * 31
     X  +        (MTH-MTH/6-MTH/11)/2 * 30
     X  +        IY/4
     X  -        IY/100
     X  +        IY/400)
     X  - 712163D0) * 8.64D4
C
      RETURN
      END
```

```
C     THIS PROGRAM IS A TEST CASE FOR READE.  IT CALLS READE
C     AT 3 SEPARATE POINTS 32 DAYS APART.
C
C       MODIFIED FOR D.G. NOVA 4 BY R. RICKLEFS          3/81
C       UNIVERSITY OF TX, MCDONALD OBS.
C
      DOUBLE PRECISION AU,RE,TPD,EMRAT,TABOUT,NUT
      DOUBLE PRECISION JED,TSEC
      DOUBLE PRECISION XAU(2),D,E
      INTEGER CAL(7)
C
      COMMON/CETB1/AU,RE,TPD,EMRAT
      COMMON/CETB2/ICW,ICENT,IREQ(13)
      COMMON/CETB4/TABOUT(6,12),NUT(4)
      COMMON/TSTCMN/XAU
C
      DATA XAU(1),XAU(2) /0.D0,1.D0/
      DATA IREQ /13*2/
C
      CALL OPEN(2,"XJPLEPH",2,IERR)
      IF (IERR.NE.1) STOP 1
      CALL OPEN(12,"TROP",2,IERR)
      IF (IERR.NE.1) STOP 2

      ICW=1
      TSEC=86400.D0*.25D0

C  DO FOR AU & KM
      DO 1 I=1,2
          JED=2441001.5D0
          AU=XAU(I)
          TPD=XAU(I)

C  DO FOR 3 JULIAN DATES 32 DAYS APART
          DO 2 J=1,3
              D=JED+TSEC/86400.D0
              E=(D-2433282.5D0)*86400.D0
              CALL SECCAL(E,CAL)
                  CAL(3)=CAL(3)+1900
              WRITE(12,101)D,E,(CAL(K),K=1,6),AU
  101         FORMAT('1 JD=',F12.3,F14.0,' SECS PAST 1950',I4,'/',I2,'/',I4,
     1        I3,2(':',I2),3X,'AU FAC =',F3.0)

C  DO FOR EACH BODY AS CENTER OF SYSTEM
              DO 3 K=1,11
                  ICENT=K
                  CALL READE(JED,TSEC,IERR)
                  IF(IERR.NE.0) GO TO 99

C  WRITE HEADER
                  WRITE(12,102)ICENT
```

```
 102              FORMAT('0',40X,'ICENT =',I3,/)
                  DO 4 K1=1,12
                      DO 5 K2=1,2
                          K3=3*K2-3
                          L1=K3+1
                          L2=K3+3

C  WRITE POSITION COMPONENTS ON ONE LINE & VELOCITY ON NEXT
                          WRITE(12,103) K1,K2,(TABOUT(KX,K1),KX=L1,L2)
 103                      FORMAT(2I3,3(1PD25.17))
   5                  CONTINUE
   4              CONTINUE

C  WRITE NUTATION AND ITS DERIVATIVES
                  WRITE(12,104)(NUT(IK),IK=1,4)
 104              FORMAT('0 NUTATIONS:',2(/,2X,2(1PD25.17)))
   3          CONTINUE
              JED=JED+32.D0
   2      CONTINUE
   1 CONTINUE

      CALL FCLOS(2)
      CALL FCLOS(12)
      STOP

  99 WRITE(12,105)IERR
 105 FORMAT('0 ERROR RETURN FROM READE. ERROR FLAG=',I4)
      STOP
      END



      OVERLAY RDEPKG
      SUBROUTINE READE(JED,TSEC,IERR)
C
      DOUBLE PRECISION AU,RE,TPD,EMRAT,TABOUT,NUT,JED,TSEC
      DOUBLE PRECISION JD(2),BIVECT,TP,EM,EMR,PF,VF,BUF,PVSUN(6)
C
      INTEGER LIST(11)
C
      LOGICAL KM,BARY
C
      COMMON/CETB1/AU,RE,TPD,EMRAT
      COMMON/CETB2/ICW,ICENT,IREQ(13)
      COMMON/CETB4/TABOUT(6,12),NUT(4)
      COMMON/STCOMM/KM,BARY,PVSUN
      COMMON/CETB3/BUF(900,2)
      COMMON/CETB5/BIVECT(6,13)
C
C
C
C  KM OR AU?
      KM=AU.LE.0.D0
```

```
C  SET KM TO AU CONVERSIONS
      PF=1.D0
      TP=TPD
      IF(TP.EQ.0.D0) TP=86400.D0
      IF(.NOT.KM) PF=AU
      VF=PF/TP
      IF(KM) VF=VF*86400.D0

C  SET EARTH-MOON MASS RATIO TO INTERNAL CONSTANT IF NOT SET BY USER
      EM=EMRAT
      IF(EM.EQ.0.D0) EM=81.3007D0

C  IF EPHEMERIS TO BE RE-READ FROM BEGINNING, DO APPROPRIATE INITIALIZATION
      IF(ICW.EQ.2) GO TO 1
      ICW=2
      BUF(1,1)=0.D0
      BUF(2,1)=0.D0
      BUF(1,2)=0.D0
      BUF(2,2)=0.D0
    1 CONTINUE
      EMR=1.D0/(1.D0+EM)

C  VERIFY THAT CENTRAL-BODY NUMBER IS VALID
      IF(ICENT.GE.1.AND.ICENT.LE.11) GO TO 7
      IERR=4
      RETURN
    7 CONTINUE

C  CHECK TYPE OF INTERP REQUESTED (0=NONE,1=POSITION,2=POSITION&VELOCITY)
      DO 2 I=1,9
          IF(IREQ(I).GE.0.AND.IREQ(I).LE.2) GO TO 2
          IERR=3
          RETURN
    2     LIST(I)=IREQ(I)

C  MOON:
      LIST(10)=IREQ(11)

      IF(ICENT.EQ.10) GO TO 8
      LST=0

C  IF ANY BODY REQUIRES POSITION OR VELOCITY, CENTRAL BODY MUST ALSO
      DO 3 I=1,10
    3     LST=MAXO(LST,LIST(I))
      IC=ICENT-ICENT/11
      LIST(IC)=MAXO(LST,IREQ(10))
    8 CONTINUE

C  EARTH:
      LIST(3)=MAXO(LIST(3),LIST(10))

C  EARTH-MOON BARYCENTER:
      LIST(3)=MAXO(LIST(3),IREQ(12))
```

```
C  SUN:
      LIST(10)=LIST(3)

C  NUTATION:
      LIST(11)=IREQ(13)
C

C  SET UP JULIAN DATES & INTERPOLATE
      JD(1)=DINT(JED-.5D0)+.5D0
      JD(2)=(JED-JD(1))+TSEC/86400.D0
      CALL XSTATE(JD,LIST,BIVECT,NUT,IERR)
      IF(IERR.NE.0) RETURN
      DO 4 I=1,6
C  MOON:
         BIVECT(I,11)=BIVECT(I,3)+EM*EMR*BIVECT(I,10)
C  NUTATION:
         BIVECT(I,12)=BIVECT(I,3)
C  EARTH:
         BIVECT(I,3)=BIVECT(I,3)-EMR*BIVECT(I,10)
C  SUN:
         BIVECT(I,10)=0.D0
    4 CONTINUE
C
C  DO FOR EACH BODY & NUTATION
      DO 5 I=1,12
C  DO FOR POSITION/VELOCITY ELEMENT
      DO 5 J=1,3
C  SUBTRACT CENTRAL-BODY CONTRIBUTION
         IF(IREQ(I).EQ.0) GO TO 5
         TABOUT(J,I)=(BIVECT(J,I)-BIVECT(J,ICENT))*PF
         IF(IREQ(I).EQ.1) GO TO 5
         TABOUT(J+3,I)=(BIVECT(J+3,I)-BIVECT(J+3,ICENT))*VF
    5 CONTINUE
C
      RETURN
C
      END



      SUBROUTINE XSTATE(JD,LIST,PV,NUT,IFL)
      DOUBLE PRECISION JD(2),PV(3,2,10),NUT(4),SS(6),BUF(900,2),T(2)
      DOUBLE PRECISION TFAC,BF
      DOUBLE PRECISION AUFAC,PVSUN(3,2)
      INTEGER LIST(11),LOC(3,12),NRL
      LOGICAL FLAG,KM,BARY
C
      COMMON/STCOMM/KM,BARY,PVSUN
      COMMON/SECTOR/IH(5),NSEC,NRS
      COMMON/CETB3/BUF
      COMMON/XSTCMN/TFAC,AUFAC,FLAG,SS,BF,LOC,NRL
C
C  DATA FOR LOCAL VARIABLES. FLAG MUST BE INITIALIZED TO .FALSE.
```

```
C  IN THE CALLING PROGRAM TO AVOID ITS RESETTING TO .FALSE.
C  WHEN READE IS IN AN OVERLAY.
      DATA TFAC/1.D0/,AUFAC/1.D0/
      DATA KM/.FALSE./,BARY/.FALSE./
      DATA NRL/0/
C
      IF(FLAG) GO TO 1
      CALL LH(2,1030)
      READ BINARY(2)N,(BUF(K,1),K=1,N)
      DO 180 I=1,6
180   SS(I)=BUF(I,1)
      READ BINARY(2)N,(BUF(K,1),K=1,N)
      CALL LH(2,-1041)
      DO 8 I=1,4
    8 READ BINARY(2)N,(BUF(K,1),K=1,N)
      READ BINARY(2)N,(BUF(K,1),K=1,N)
      BF=BUF(1,1)
      READ BINARY(2)N,(BUF(K,1),K=1,N)
      CALL LH(2,-1050)
      READ BINARY(2)N,(BUF(K,1),K=1,N)
      READ BINARY(2)N,(BUF(K,1),K=1,N)
      DO 190 II=1,12
      KK=3*(II-1)
      DO 190 JJ=1,3
190   LOC(JJ,II)=IDINT(BUF(KK+JJ))
      READ BINARY(2)N,(BUF(K,1),K=1,N)
      CALL LH(2,-1070)
      FLAG=.TRUE.

    1 CONTINUE
      IFL=0
      T(2)=SS(3)
      AUFAC=1.D0
      TFAC=1.D0
      IF(KM) T(2)=SS(3)*86400.D0
      IF(KM) TFAC=86400.D0
      IF(.NOT.KM) AUFAC=1.D0/BF
      IF(JD(1).LT.SS(1)) GO TO 99
      IF(JD(1)+JD(2).GT.SS(2)) GO TO 98
   13 IF((JD(1).GE.BUF(1,1)).AND.(JD(1)+JD(2).LE.BUF(2,1))) GO TO 11
      IF((JD(1).GE.BUF(1,2)).AND.(JD(1)+JD(2).LE.BUF(2,2))) GO TO 14
      NR=IDINT((JD(1)-SS(1))/SS(4))+1
C     ****************************************************
      IF(JD(1)+JD(2).GE.SS(2)-SS(3)) NR=NR-1
      IF(JD(1)+JD(2).EQ.SS(2)) NR=NR-1
      NSK=NR-NRL-1
      IF(NSK)7,5,12
   12 DO 9 I=1,NSK
    9 READ BINARY(2,END=98)N,(BUF(K,1),K=1,N)
      GO TO 5
    7 CALL LH(2,1070)
      NSK=NR-1
      GO TO 12
C   7 NSK=-NSK
```

```
C      DO 10 I=1,NSK
C   10 BACKSPACE 2
C      ******************************************************
C      NS=NSEC+NR
C      CALL FSEEK(2,NS)
C        TYPE " CALLING FSEEK ",NR,NSEC
     5 READ BINARY(2,END=98)N,(BUF(I,1),I=1,N)
       READ BINARY(2,END=15)N,(BUF(I,2),I=1,N)
X        TYPE "READ: ",JD, BUF(1,1), BUF(1,2), BUF(2,1), BUF(2,2)
15     NRL=NR+1
       GO TO 13
    11 NB=1
       GO TO 2
    14 NB=2
     2 T(1)=((JD(1)-BUF(1,NB))+JD(2))*TFAC/T(2)
       LL=LOC(1,11)
       CALL XINTRP(BUF(LL,NB),T,LOC(2,11),3,LOC(3,11),2,PVSUN)
     6 DO 3 I=1,10
       IF(LIST(I).EQ.0) GO TO 3
       LL=LOC(1,I)
       CALL XINTRP(BUF(LL,NB),T,LOC(2,I),3,LOC(3,I),LIST(I),PV(1,1,I))
       NM=LIST(I)*3
       DO 4 J=1,NM
       IF(I.LE.9.AND.(.NOT.BARY))PV(J,1,I)=(PV(J,1,I)-PVSUN(J,1))*AUFAC
       IF(I.LE.9.AND.BARY)PV(J,1,I)=PV(J,1,I)*AUFAC
       IF(I.EQ.10) PV(J,1,I)=PV(J,1,I)*AUFAC
     4 CONTINUE
     3 CONTINUE
       IF(LIST(11).EQ.0) RETURN
       LL=LOC(1,12)
       CALL XINTRP(BUF(LL,NB),T,LOC(2,12),2,LOC(3,12),LIST(11),NUT)
       RETURN
C
    99 CONTINUE
       IFL=1
       RETURN
    98 IFL=2
       RETURN
C
       END


C*************************************************
C
      SUBROUTINE XINTRP(BUF,T,NCF,NCM,NA,FL,PVA)
C
C*************************************************
C
C
C     THIS SUBROUTINE DIFFERENTIATES AND INTERPOLATES A
C     SET OF CHEBYSHEV COEFFICIENTS TO GIVE POSITION, VELOCITY,
C     AND ACCELERATION.
C
```

```
C     CALLING SEQUENCE PARAMETERS --
C
C       INPUT --
C
C         BUF   1ST LOCATION OF ARRAY OF DP CHEBYSHEV COEFFICIENTS OF POSITION
C
C           T   T(1) IS DP FRACTIONAL TIME IN INTERVAL COVERED BY
C               COEFFICIENTS AT WHICH INTERPOLATION IS WANTED
C               (0 .LE. T(1) .LE. 1).  T(2) IS DP LENGTH OF WHOLE
C               INTERVAL IN INPUT TIME UNITS.
C
C         NCF   NO. OF COEFFICIENTS PER COMPONENT
C
C         NCM   NO. OF COMPONENTS PER SET OF COEFFICIENTS
C
C          NA   NO. OF SETS OF COEFFICIENTS IN FULL ARRAY
C               (I.E., NO. OF SUB-INTERVALS IN FULL INTERVAL)
C
C          FL   INTEGER FLAG  =1 FOR POSITIONS ONLY
C                             =2 FOR P,V ONLY
C                             =3 FOR P,V,A
C
C
C       OUTPUT --
C
C         PVA   INTERPOLATED QUANTITIES REQUESTED.  DIMENSION
C               EXPECTED IS PVA(NCM,FL), DP.
C
C
      DOUBLE PRECISION BUF(NCF,NCM,15)
      DOUBLE PRECISION T(2)
      DOUBLE PRECISION PVA(NCM,FL)
      DOUBLE PRECISION BMA,BMA2
      DOUBLE PRECISION TEMP
      DOUBLE PRECISION TC
      DOUBLE PRECISION TCL
      DOUBLE PRECISION PC(18),VC(18),AC(18)
      DOUBLE PRECISION TWOT

      INTEGER FL

      COMMON /XINCMN/ PC,VC,AC

      EQUIVALENCE (TCL,PC(2))

      DATA PC(1),PC(2)/1.D0,2.D0/
      DATA VC(2)/1.D0/
      DATA AC(3)/4.D0/
C
C       CALCULATE LOCATION OF CORRECT COMPONENT ARRAYS
C       AND SCALED CHEBYSHEV TIME IN THAT INTERVAL
C
X       TYPE " ENTERING XINTRP"
      TEMP=T(1)*DFLOAT(NA)
```

```
      L=IDINT(TEMP-DINT(T(1)))+1
      TC=2.D0*((TEMP-DINT(TEMP))+DINT(T(1)))-1.D0
C
C       CHECK TO SEE WHETHER CHEBYSHEV TIME HAS CHANGED,
C       AND COMPUTE NEW POLYNOMIAL VALUES IF IT HAS
C
      IF(TC.EQ.TCL) GO TO 1
      NP=2
      NV=3
      NAC=4
      TCL=TC
      TWOT=TC+TC
    1 CONTINUE
      IF(NP.GE.NCF) GO TO 2
      M=NP+1
      NP=NCF
      DO 3 I=M,NCF
    3 PC(I)=TWOT*PC(I-1)-PC(I-2)
    2 CONTINUE
C
C       INTERPOLATE TO GET POSITION FOR EACH COMPONENT
C
X       TYPE "INTERP",NCM,NCF,FL
      DO 4 I=1,NCM
      PVA(I,1)=0.D0
      DO 4 J=1,NCF
      JJ=NCF-J+1
      PVA(I,1)=PVA(I,1)+PC(JJ)*BUF(JJ,I,L)
    4 CONTINUE
      IF(FL.LE.1) RETURN
C
C       CHECK VELOCITY POLYNOMIAL VALUES, AND GENERATE
C       NEW ONES IF REQUIRED
C
C
X       TYPE "CHECK VELOCITY"
      BMA=2.D0*DFLOAT(NA)/T(2)
      VC(3)=TWOT+TWOT
      IF(NV.GE.NCF) GO TO 5
      M=NV+1
      NV=NCF
      DO 6 I=M,NCF
    6 VC(I)=TWOT*VC(I-1)+PC(I-1)+PC(I-1)-VC(I-2)
C
C       EVALUATE VELOCITY FOR EACH COMPONENT
C
    5 CONTINUE
X       TYPE "EVAL VELOC (LABEL 5)"
      DO 7 I=1,NCM
      PVA(I,2)=0.D0
      DO 11 J=2,NCF
      JJ=NCF-J+2
X       TYPE I,J,JJ,BMA,NCM,NCF
      PVA(I,2)=PVA(I,2)+VC(JJ)*BUF(JJ,I,L)
```

```
   11 CONTINUE
      PVA(I,2)=PVA(I,2)*BMA
    7 CONTINUE
      IF(FL.EQ.2) RETURN
C
C       CHECK ACCELERATION POLYNOMIAL VALUES, AND
C       RE-DO IF NECESSARY
C
X       TYPE "CHECK ACCEL"
      BMA2=BMA*BMA
      AC(4)=24.D0*PC(2)
      IF(NAC.GE.NCF) GO TO 8
      M=NAC+1
      NAC=NCF
      DO 9 I=M,NCF
    9 AC(I)=TWOT*AC(I-1)+4.D0*VC(I-1)-AC(I-2)
    8 CONTINUE
C
C       GET ACCELERATION FOR EACH COMPONENT
C
X       TYPE "GET ACCEL"
      DO 10 I=1,NCM
      PVA(I,3)=0.D0
      DO 12 J=3,NCF
      JJ=NCF-J+3
      PVA(I,3)=PVA(I,3)+AC(JJ)*BUF(JJ,I,L)
   12 CONTINUE
      PVA(I,3)=PVA(I,3)*BMA2
   10 CONTINUE
C
X       TYPE " XINTRP RETURN"
      RETURN
C
      END



C**************************************
C
      SUBROUTINE XCONST(NAM,VAL,SS,N)
C
C**************************************
C
C     THIS SUBROUTINE READS THE EXPORT JPL PLANETARY EPHEMERIS
C     AND OBTAINS THE NAMES AND VALUES OF THE ASTRONOMICAL CONSTANTS
C     THAT WERE USED TO GENERATE THE EPHEMERIS.
C
C     CALLING SEQUENCE PARAMETERS:
C
C         NAM   INTEGER ARRAY THE N 6-CHARACTER NAMES OF THE CONSTANTS
C
C         VAL   DOUBLE PRECISION ARRAY THAT WILL CONTAIN THE N VALUES
C
```

```
C          SS   6-WORD DOUBLE-PRECISION ARRAY THAT WILL BE FILLED
C               WITH THE CONTENTS OF GROUP 1030 (START AND STOP EPOCHS, ETC.)
C
C           N   INTEGER NUMBER OF VALUES READ FROM FILE
C
C
C     *** NOTE ***   BECAUSE OF TAPE POSITIONING, THIS ROUTINE, IF IT
C                    IS USED AT ALL, MUST NOT BE CALLED AFTER READE
C                    OR STATE HAVE BEEN REFERENCED IN THE SAME PROGRAM.
C
C
      INTEGER NAM(600)
      INTEGER H(5)
C
      DOUBLE PRECISION VAL(200),SS(6)
C
      COMMON/SECTOR/H
C
C
C
C       IF OTHER EPH ROUTINES HAVE BEEN CALLED, DON'T DO ANYTHING
C
      IF(H(4).NE.0) RETURN
C
C       OTHERWISE, FIND LIMITS GROUP AND COPY INTO SS
C
      CALL LH(2,1030)
      READ BINARY(2)NDUM,(SS(K),K=1,NDUM)
C
C       FIND RECORD 4 IN GROUP 1040 (NAMES OF CONSTANTS)
C
      DO 1 I=1,5
      READ BINARY(2)NDUM,(NAM(K),K=1,NDUM)
    1 CONTINUE
      READ BINARY(2)NDUM,(NAM(K),K=1,NDUM)
C
C       FIND AND READ VALUES OF CONSTANTS
C
      DO 2 I=1,6
      READ BINARY(2)NDUM,(VAL(K),K=1,NDUM)
    2 CONTINUE
      READ BINARY(2)N,(VAL(K),K=1,N)
C
      REWIND 2
      RETURN
      END



C   THIS PROGRAM PRINTS THE CONTENTS OF A TYPE-66 TAPE.
C
C     INPUTS
C         TAPE - ONE TYPE 66 EPHEMERIS TAPE - ON UNIT -NUNIT-
C         CARDS - CIN NAMELIST
```

```
C           CONTENTS
C              IOPT     1-5        OPTION REQUESTED
C              KEY                  KEY1 OF SPECIAL GROUP ON TAPE
C              NREC                 COUNT OF RECORDS TO BE DUMPED FOR
C                                   OPTIONS 2 AND 4
C              NUNT                FORTRAN UNIT NUMBER OF INPUT TAPE
C              JD                   STARTING JD IOPT=4 OPTION
C
C
C     OUTPUT
C         PRINTED- A COMPLETE LIST OF HEADERS AND TRAILERS FROM THE TAPE
C                  AS WELL AS LIST OF ALL DATA WITH THE DELETIONS AS
C                  SPECIFIED BY THE IOPT FLAG.
C
C
C       MODIFIED FOR D.G. NOVA BY R.L. RICKLEFS          3/81
C       UNIVERSITY OF TX, MCDONALD OBS.
C
      DOUBLE PRECISION JD(10),TT
      DOUBLE PRECISION DBUF(2000)
C
      INTEGER NW(4)
      INTEGER IBUF(6000)
      INTEGER NPOS,NWD(2),IOPT
      INTEGER STAR
      INTEGER T
      INTEGER ATYPE(5)
      INTEGER FILNAM(12)
C
      LOGICAL EOG
C
      INTEGER SBUF(6000)
C
      COMMON ISNGL,NWORDS,KTYPE
      COMMON /DDDCMN/STAR,NW,ATYPE,DBUF, KEY,NPOS,NWD

      EQUIVALENCE (DBUF(1),IBUF(1)), (DBUF(1),SBUF(1))
C
      DATA STAR /"**"/
      DATA NW /7,4,9,54/
      DATA ATYPE /1HR,1HD,1HI,1HB,1HE/
      DATA KEY /1070/
      DATA NPOS /1/
      DATA NWD /1,10000/
C
 2000 FORMAT (1H1,8X,27HGENERAL EPHEMERIS TAPE DUMP,13X,15A2)
 2010 FORMAT (1H0,8X,30HDUMPING ALL OF ALL KEY1 GROUPS)
 2020 FORMAT (1H0,8X,49HDUMPING ALL OF ALL KEY1 GROUPS BUT ONLY THE FIRST,
     *       I3,9H AND LAST,I3,8H OF KEY1,I3,6H GROUP)
 2030 FORMAT (1H0,8X,47HDUMPING ALL OF ALL KEY1 GROUPS EXCEPT ONE LINE ,
     1       20HONLY FROM EACH KEY1 ,I4, 6H BLOCK)
 2040 FORMAT (28HOT TREC GR GREC  NWDS  DATA ,49A2,5H DATA/)
 2050 FORMAT (1H ,A1,I5,I3,I5,I6,2X,9I10)
 2060 FORMAT (1H ,A1,I5,I3,I5,I6,2X,54A2)
```

```
 2070 FORMAT (1H ,A1,I5,I3,I5,I6,2X,1P7E15.7)
 2080 FORMAT (1H ,A1,I5,I3,I5,I6,2X,1P4D25.17)
 2090 FORMAT (1H0,65A2////50X,17HNORMAL END OF JOB////,1H0,65A2)
 2100 FORMAT (1H0,8X, "DUMPING ALL OF ALL KEY1 GROUPS EXCEPT FOR GROUP "
     *"KEY1 OF",I2," WHERE WE ARE DUMPING",I4/" RECORDS FROM FIRST RECORD "
     *"WITH JD.GE.           ",7P7D24.7)
 2110 FORMAT ('0',8X,'DUMPING FIRST GROUP WITH KEY1=',I10)
C
C
C
C     ********************** BEGIN EXECUTION HERE **********************
C
C
      NUNIT=2
      WRITE(10,1000)
1000  FORMAT(" ENTER EPHEMERIS FILE NAME: ",Z)
      READ(11,1005) FILNAM
1005  FORMAT(12A2)
      CALL OPEN(NUNIT,FILNAM,2,IERR)
      IF (IERR.EQ.1) GO TO 3
        TYPE "OPEN ERROR: ",IERR
        STOP
   3  NPOS=1
      ACCEPT "OPTION: ",IOPT
      IF (IOPT.NE.2.AND.IOPT.NE.4) GO TO 5
      ACCEPT "KEY1: ",KEY
      ACCEPT "NUMBER OF RECORDS TO BE DUMPED: ",NREC
      IF (IOPT.EQ.4) ACCEPT "STARTING JULIAN DATES (10 OF THEM): ",JD
5     ACCEPT "POSITION OF TIME TAG: (1) ",NPOS
      ACCEPT "START DUMP W/ WHICH WORD: (1) ",NWD(1)
      ACCEPT "END DUMP W/ WHICH WORD: (10000) ",NWD(2)
      REWIND NUNIT
C
C
      IF (IOPT.EQ.1) WRITE (12,2010)
C
      IF (IOPT.EQ.2) WRITE (12,2020) NREC,NREC,KEY
C
      IF (IOPT.EQ.3) WRITE (12,2030) KEY
C
      IF (IOPT.EQ.4) WRITE (12,2100) KEY,NREC,JD
C
      IF (IOPT.EQ.5) WRITE (12,2110) KEY
C
C         PRINT OUT THE STARS AND INITIALIZE COUNTS
C
      WRITE (12,2040) (STAR,I=1,49)
      ITREC=0
      IGR=0
C
C     READ A HEADER
C
   10 READ BINARY(NUNIT) NMAX,KTYPE,ISNGL,KEY1,KEY2
C
```

```
C     STEP COUNTS AND LIST THE HEADER
C
      IFRONT=0
      ITREC=ITREC+1
      IGR=IGR+1
      IGREC=0
      NWORDS=0
      T=ATYPE(KTYPE)
      WRITE (12,2050) T,ITREC,IGR,IGREC,NWORDS,NMAX,KTYPE,ISNGL,KEY1,KEY2
C
C     IS THIS AN END OF FILE HEADER (Q)
C
      IF (KTYPE.EQ.5) GO TO 90
C
C     NO, STEP RECORD COUNTS
C
C
C         IF OPTION IS TWO, ARE WE AT RIGHT KEY1 GROUP, IF SO DO WE
C              NEED TO PRINT THIS RECORD OR START SPINNING DOWN TAPE.
C
   20 IF (IOPT.EQ.5.AND.KEY.NE.KEY1) GO TO 30
      IF (IOPT.NE.2) GO TO 50
C
      IF (KEY1.NE.KEY) GO TO 50
C
      IF (IGREC.LE.NREC) GO TO 50
C
C     READ ON THRU THE END OF THIS KEY1 GROUP.
C
   30 IF(KTYPE.EQ.2) READ BINARY(NUNIT)NWORDS,DBUF(1)
      IF(KTYPE.NE.2) READ BINARY(NUNIT)NWORDS,IBUF(1)
      IGREC=IGREC+1
      ITREC=ITREC+1
      IF(.NOT.EOG(1)) GO TO 30
C
C     BACK UP TO ALLOW DUMP OF NREC RECORDS FROM END OF GROUP
C
      GO TO 10
      IF (IOPT.GE.4) GO TO 10
      L= IGREC-NREC+1
      M= NREC+1
      N=MINO(L,M)
      IGREC=IGREC-N
      ITREC=ITREC-N
      DO 40 I=1,N
   40 BACKSPACE NUNIT
      IOPT=1
C
C     READ A DATA RECORD
C
   50 IF (KTYPE.EQ.2) READ BINARY(NUNIT) NWORDS,(DBUF(I),I=1,NWORDS)
      IF (KTYPE.NE.2) READ BINARY(NUNIT) NWORDS,(SBUF(I),I=1,NWORDS)
      ITREC=ITREC+1
      IGREC=IGREC+1
```

```
      TT=DBUF(NPOS)
      IF(KTYPE.NE.2)TT=DBLE(FLOAT(SBUF(NPOS)))
C
C     DECIDE WHETHER TO PRINT JUST ONE LINE FROM THIS RECORD.
C
   60 CONTINUE
      IF(NWD(1).NE.0) GO TO 61
      N1=1
      N2=NWORDS
      GO TO 62
   61 IF(NWD(1).GT.0) GO TO 63
      N2=NWORDS
      N1=MAXO(NWORDS+NWD(1)+1,1)
      GO TO 62
   63 IF(NWD(2).NE.0) GO TO 64
      N2=MINO(NWD(1),NWORDS)
      N1=1
      GO TO 62
   64 N1=MINO(NWD(1),NWORDS)
      N2=MINO(NWD(2),NWORDS)
   62 NWRITE=MINO(NW(KTYPE),N2-N1+1)
      IF (KEY1.NE.KEY) GO TO 70
      IF (IOPT.EQ.4) GO TO 80
      IF (IOPT.EQ.2.AND.IGREC.GT.NREC) GO TO 30
      IF(IOPT.EQ.5.AND .IGREC.GT.NREC.AND.NREC.GT.0) GO TO 90
C
C     LIST A DATA RECORD.
C
   70 L1=N1
      L2=N1+NWRITE-1
      IF (KTYPE.EQ.1) WRITE (12,2070) T,ITREC,IGR,IGREC,NWORDS,
     *                                (SBUF(I),I=L1,L2)
      IF (KTYPE.EQ.2) WRITE (12,2080) T,ITREC,IGR,IGREC,NWORDS,
     *                                (DBUF(I),I=L1,L2)
      IF (KTYPE.EQ.3) WRITE (12,2050) T,ITREC,IGR,IGREC,NWORDS,
     *                                (SBUF(I),I=L1,L2)
      IF (KTYPE.EQ.4) WRITE (12,2060) T,ITREC,IGR,IGREC,NWORDS,
     *                                (SBUF(I),I=L1,L2)
C
      N1=N1+NWRITE
      IF(IOPT.EQ.3.OR.N1.GT.N2) GO TO 75
      DO 76 K=N1,N2,NWRITE
      KW=MINO(NWRITE-1,N2-K)
      L1=K
      L2=K+KW
      IF(KTYPE.EQ.1) WRITE(12,101)K,(SBUF(I),I=L1,L2)
  101 FORMAT(I21,'.',1X,7(1PE15.7))
      IF(KTYPE.EQ.2) WRITE(12,102)K,(DBUF(I),I=L1,L2)
  102 FORMAT(I21,'.',1X,4(1PD25.17))
      IF(KTYPE.EQ.3) WRITE(12,103)K,(SBUF(I),I=L1,L2)
  103 FORMAT(I21,'.',1X,9I10)
      IF(KTYPE.EQ.4) WRITE(12,104)K,(SBUF(I),I=L1,L2)
  104 FORMAT(I21,'.',1X,54A2)
   76 CONTINUE
```

```
C
C     IS THIS THE LAST DATA RECORD OF THE GROUP
C
   75 IF(EOG(1).AND.IOPT.EQ.5) GO TO 90
      IF(EOG(1)) GO TO 10
C
C     NO, GO READ ANOTHER DATA RECORD
C
      GO TO 50
C
C **** HERE  ON  IOPT=4 ****
C
80    IF(JD(2).NE.0.D0)GO  TO  81
      IF(TT.LT.JD(1))GO  TO  75
      IFRONT=IFRONT+1
      IF(IFRONT.LE.NREC)GO  TO  70
      GO  TO  90  ;FINISHED  WITH  TAPE
81    DO  811  K=1,5
      IF(JD(2*K-1).EQ.0.D0)GO  TO  75
      X=(TT-JD(2*K-1))*(JD(2*K)-TT)
      IF(X.GE.0.)GO  TO  70
811   CONTINUE
      GO  TO  75
C
C     WRITE END OF JOB MESSAGE.
C
   90 WRITE (12,2090) (STAR,I=1,130)
C
      STOP
      END
```

JD= 2441001.750 666943200. SECS PAST 1950 2/19/1971 6: 0: 0 AU FAC = 0.

ICENT = 1

```
 1  1  0.00000000000000000D  0  0.00000000000000000D  0  0.00000000000000000D  0
 1  2  0.00000000000000000D  0  0.00000000000000000D  0  0.00000000000000000D  0
 2  1 -1.15841653845160410D  8  2.51377345681265740D  6  1.39764531574209220D  7
 2  2 -1.85668327698334960D  1 -4.55985271532626940D  1 -1.95042326863801700D  1
 3  1 -1.50351954250855510D  8  1.23070576998692220D  8  6.12583073787269440D  7
 3  2 -5.16081084967129120D  1 -4.19954671049660640D  1 -1.63736275160668360D  1
 4  1 -2.03594504214149160D  8 -8.42087254995767880D  7 -2.72925205412061880D  7
 4  2 -1.96060970246275160D  1 -3.32123296179223430D  1 -1.33824975755540900D  1
 5  1 -4.95873472528967680D  8 -5.49328607710719700D  8 -2.15954755152517630D  8
 5  2 -2.56183242574628980D  1 -2.47209031502363490D  1 -9.10801053288185040D  0
 6  1  8.08383574218654630D  8  1.06698285095959220D  9  4.14228285811124860D  8
 6  2 -4.42288037171842530D  1 -1.30240479853370660D  1 -3.56365473242831430D  0
 7  1 -2.71841365066665730D  9 -4.25641995722163200D  8 -1.40764168373901400D  8
 7  2 -3.48402332240523730D  1 -2.47379578391935060D  1 -8.93624404410176700D  0
 8  1 -2.23124686181931040D  9 -3.63107322287867630D  9 -1.42265057028855480D  9
 8  2 -3.13354169479585960D  1 -2.07013666992783460D  1 -7.20128599288886820D  0
 9  1 -4.54510677664667700D  9 -3.08297081141257940D  8  1.28731420955798450D  9
```

```
 9  2 -3.52644116433816490D  1 -2.37583961112890840D  1 -8.06300569155179360D  0
10  1 -2.27388009771042430D  7  5.[illegible]5068932979396770D  7  3.15262937042456120D  7
10  2 -3.60564055822395880D  1 -1.83188932560449050D  1 -6.10643026840661900D  0
11  1 -1.50494265534759560D  8  1.22755219458875790D  8  6.10839193135507890D  7
11  2 -5.06471875940273400D  1 -4.22926238600450010D  1 -1.64696505141363940D  1
12  1 -1.50353683413352360D  8  1.23066745226425200D  8  6.12561884652161710D  7
12  2 -5.15964327648821310D  1 -4.19990777274161640D  1 -1.63747942497406630D  1

 NUTATIONS:
   5.37071582249820450D -5  3.66039299624601530D -5
   4.02632824244921960D-12 -9.15520080485547150D-13

                                        ICENT =  2

 1  1  1.15841653845160410D  8 -2.51377345681265740D  6 -1.39764531574209220D  7
 1  2  1.85668327698334960D  1  4.55985271532626940D  1  1.95042326863801700D  1
 2  1  0.00000000000000000D  0  0.00000000000000000D  0  0.00000000000000000D  0
 2  2  0.00000000000000000D  0  0.00000000000000000D  0  0.00000000000000000D  0
 3  1 -3.45103004056951030D  7  1.20556803541879560D  8  4.72818542213060220D  7
 3  2 -3.30412757268794120D  1  3.60306004829662970D  0  3.13060517031333400D  0
 4  1 -8.77528503689887520D  7 -8.67224989563894460D  7 -4.12689736986271110D  7
 4  2 -1.03926425479401960D  0  1.23861975353403490D  1  6.11998292882476540D  0
 5  1 -3.80031818683807310D  8 -5.51842381167532380D  8 -2.29931208309938550D  8
 5  2 -7.05149148762940520D  0  2.08776240030263410D  1  1.03962221534983240D  1
 6  1  9.24225228063814990D  8  1.06446907750277950D  9  4.00251832653703930D  8
 6  2 -2.56619709473507530D  1  3.25744791679256270D  1  1.59405779539518600D  1
 7  1 -2.60257199682149690D  9 -4.28155769178975880D  8 -1.54740621531322330D  8
 7  2 -1.62734004542188730D  1  2.08605693140691810D  1  1.05679886422784070D  1
 8  1 -2.11540520797415010D  9 -3.63358699633548900D  9 -1.43662702344597570D  9
 8  2 -1.27685841781250990D  1  2.48971604539843410D  1  1.23029466934913060D  1
 9  1 -4.42926512280151650D  9 -3.10810854598070620D  8  1.27333775640056360D  9
 9  2 -1.66975788735481490D  1  2.18401310419736030D  1  1.14412269948283810D  1
10  1  9.31028528680561630D  7  5.19931198411270190D  7  1.75498405468246900D  7
10  2 -1.74895728124060880D  1  2.72796338972177850D  1  1.33978024179735550D  1
11  1 -3.46526116895991530D  7  1.20241446002063130D  8  4.71074661561298670D  7
11  2 -3.20803548241938400D  1  3.30590329321769240D  0  3.03458217224377470D  0
12  1 -3.45120295681919490D  7  1.20552971769612540D  8  4.72797353077952490D  7
12  2 -3.30295999950486310D  1  3.59944942584652930D  0  3.12943843663950650D  0

 NUTATIONS:
   5.37071582249820450D -5  3.66039299624601530D -5
   4.02632824244921960D-12 -9.15520080485547150D-13
```

On-Site Integration of Starlette in a Taylored Field

by

E. Vermaat
Working Group for Satellite Geodesy
Kootwijk
Delft University of Technology
Delft, The Netherlands.

ABSTRACT

The periodical transmission of updated orbital parameters by a prediction center to mobile laser ranging stations, can be limited to very small data sets if there is an on-site capacity to reproduce an approximating orbit, not deviating from the original orbit beyond the prediction accuracy. To meet this requirement the STARLETTE orbit is numerically integrated in a tailored force field, especially designed to allow for integration of the equations of motion in a small computer. Deviations from the reference orbit can be kept within typically 50 to 75 m for 7-day arcs with a gravity field comprising about 20 selected terms, together with auxiliary expressions for the effects of lunar gravity and air drag.

1 Introduction

The reproduction of an orbit of STARLETTE from a given set of osculating elements (state vector) requires numerical integration of the equations of motion, to be derived from a dynamical model featuring the significant forces acting on the satellite. On-site this would require an unrealistically large computer capacity. For several SLR stations it is therefore common practice to truncate the dynamical model to acceptable sizes, thus obtaining a reproduced orbit which is tangential to the originally predicted orbit at the initial epoch. The reproduced orbit will diverge from the original one. The divergence occurring, being a function of the degree of simplification, can be controlled by limiting the arc length.

An alternative solution, which might allow for a considerably greater length of arc, would be to re-estimate the initial state vector, fitting the reproducing arc to the originally predicted orbit over a certain arc length, in an adjustment, utilizing a truncated dynamical model. The actual reproduction then proceeds as described above, but now starting from the re-estimated state vector, which could be considered to be a 'mean' state vector typical to the truncated model utilized. This model will have to be designed carefully, optimizing permissible arc length and prediction accuracy versus simplicity. It can be expected that the model will have to be tailored specifically to the characteristics of the orbit of each satellite involved.

This approach has been investigated to some extent, mainly for the orbit of STARLETTE and the results indicate that with relatively simple models, arc lengths can be covered of the order of the valid length of the predicted orbit itself.

2 Earth Gravity

As a first step it was attempted to locate those spherical harmonics coefficients of the earth's gravity field which significantly would contribute to the accuracy of the reproduced orbit in the approximation to the original orbit. Therefore a reference orbit of STARLETTE was generated, exclusively utilizing the complete GEM9 gravity field. From a 3-day arc (the 3-day arc length was more or less hand-picked, since by that time there was no clear insight into the permissible arc length to which these investigations would converge) out of this orbit a gravity field model up to degree and order 14 was estimated. To test the significance of the coefficients solved for, each estimated value was divided by its formal standard deviation. These 'significance parameters' as far as exceeding the value of 1.0 are displayed in figure 1.
This resulted in 34 individual C- or S-coefficients, pertaining to 29 different terms. With this tailored gravity model comprising the estimated values of the 'significant' terms, a 6-day arc out of the reference orbit was approximated, solving only for the initial state vector. From this re-estimated state vector a new 6-day arc was reproduced utilizing the same tailored model. The deviations of the reproduced orbit with respect to the reference orbit are displayed in figure 2 in along-track,radial and across-track components.

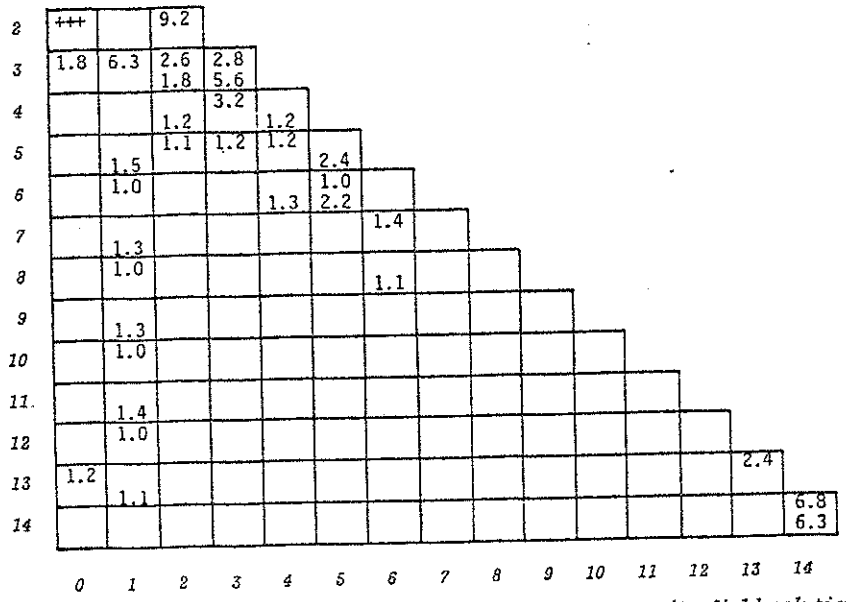
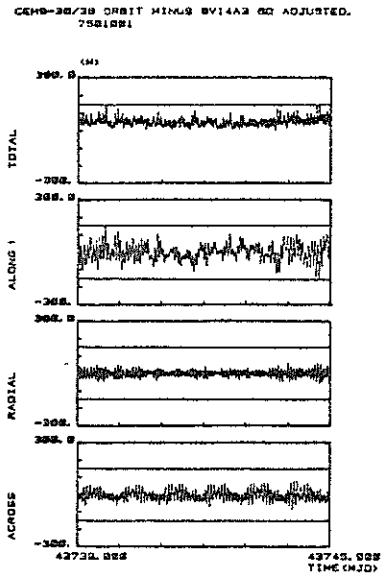


Figure 1. *Values of significance parameters from a gravity field solution up to degree and order 14 utilising a 3-day STARLETTE arc. The significance parameter is the estimated value devided by its standard error. For each term the upper value pertains to the C-coefficient and the lower value to the S-coefficient. Only the values greater than 1.0 are displayed.*

A dominant feature is the short-periodic oscillation with a typical wave length of one revolution. Further attempts have been mainly focussed on limiting the amplitude of these oscillations. After various attempts it was concluded that the 'significance parameters' were not very informative, probably mainly due to the neglected correlation. Therefore a more systematic approach was followed starting from all coefficients up to degree and order 4. Highlights of these attempts are depicted in the figures 3, 4 and 5.
The solution in figure 5 was concluded to be very satisfactory, yielding a maximum along-track error of 50 m.

3 Disturbing Force Fields

The next step was to investigate till what extent the influence of other force fields peculiar to the STARLETTE orbit, could be absorbed by the gravity coefficients deployed in the solution of figure 5. Therefore a new reference orbit was generated utilizing the GEM9 earth gravity field as well as the additional force models, outlined in table 1.
The result of an approximation of a 7-day arc out of this reference orbit, utilizing the tailored earth gravity model only, is displayed in figure 6.
It is remarkable that only the across-track component suffers notably from the addition of the disturbing forces and the amplitude is quite acceptable still. Further study revealed that this result does not always reproduce as can be seen from figure 7.
There a 7-day reference orbit was generated for a completely different period. This figure shows the result of an approximation to a GEM9 reference orbit and an orbit generated in a realistic force field discussed above, respectively. The solution for the realistic orbit now suffers from large effects up to 300 m in the along-track component. Figure 8 indicates which of the disturbing



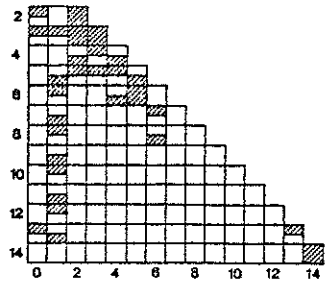
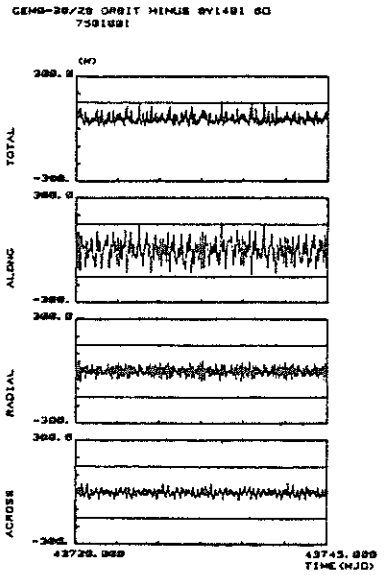
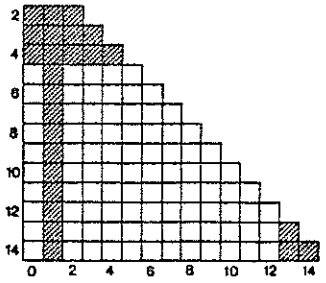


fig. 2

fig. 3





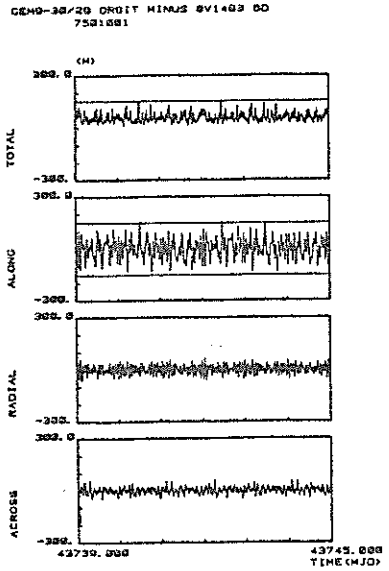
Figures 2 and 3.

Deviations of the approximated arc from the reference orbit. The reference orbit is created from the GEM9 gravity field and no other forces were accounted for. The selected gravity coefficients included in the taylored model are indicated in the right hand figures.

force fields are primarily responsible.
Both the effects of lunar gravity and of air drag cannot be sufficiently absorbed by the gravity coefficients in this case. The influence of all other individual effects was negligible. It is anticipated that simplified analytical expressions to accommodate the effect of lunar gravity and of air drag would in practice be sufficient to limit the approximation error in a 7-day orbit to 50-75 m.

4 LAGEOS

To have an impression on the applicability of this technique to the LAGEOS orbit, a reference orbit was generated for this satellite utilizing the



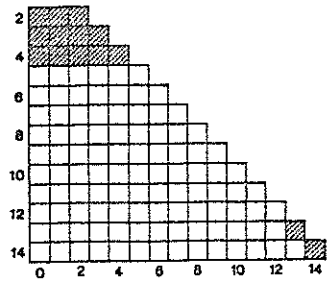
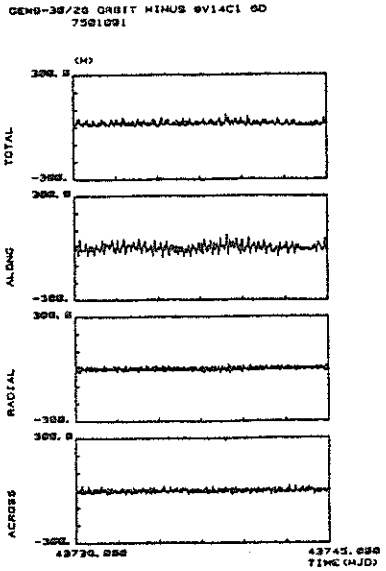
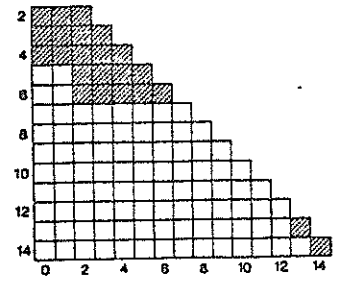


fig. 4

fig. 5





Figures 4 and 5.

Similar results as in figures 2 and 3. The taylored model indicated in figure 5 has been selected for the subsequent investigations.

models as outlined in table 1. Results of the approximation of a 30-day arc to this orbit are illustrated in figure 9.
At first the tailored model consisted of earth gravity coefficients up to degree and order 4 only. The quite fatal long-periodic effects illustrated, appeared to be mainly due to lunar and solar gravity, as is illustrated by the lower graph.

5 Conclusions

Although further investigations will be required, the results obtained so far, clearly indicate the feasibility of the approximation technique to represent a 7-day STARLETTE arc by one single state vector, going with a

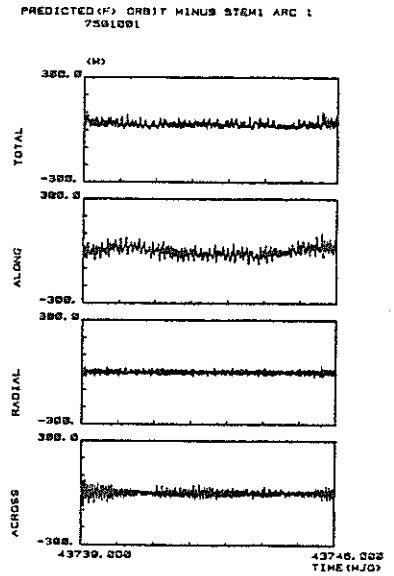


Figure 6. *Deviations from the reference orbit described in table 1. The taylored model used in the approximation is the one indicated in figure 5.*

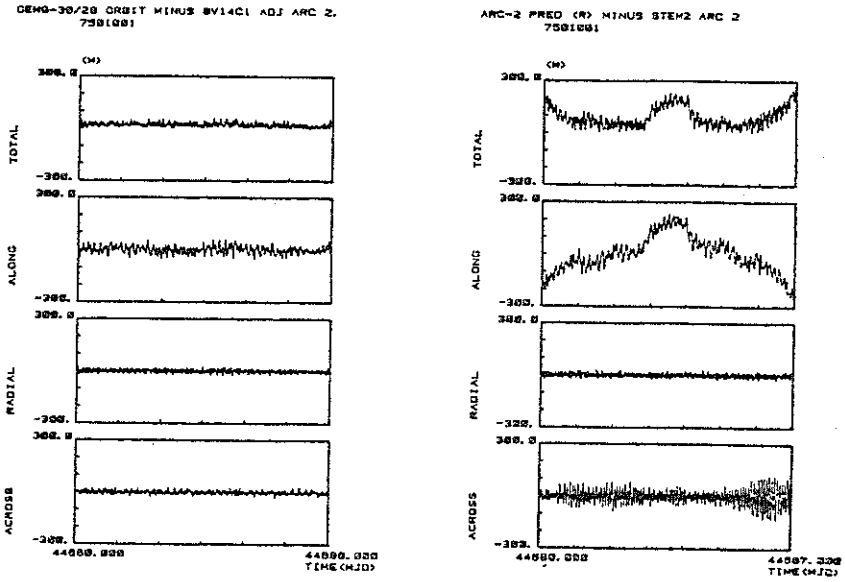


Figure 7. *Another STARLETTE arc. In the left hand figure only GEM9 was incorporated in the reference orbit. The right hand figure illustrates the result of an approximation to the reference orbit described in table 1. In this case a significant along-track error occurs (compare figure 6).*

strongly simplified tailored force field from which the equations of motion of

	STARLETTE	LAGEOS
Earth gravity	GEM9 up to (30,28)	GEM10B up to (13,13)
Lunar gravity	mass ratio 1.229997E-2	mass ratio 1.229997E-2
Solar gravity	mass ratio 3.329456E+5	mass ratio 3.329456E+5
Earth tides	K2= 0.290 K3= 0.0 Phase angle= 2.5 degr.	K2= 0.350 K3= 0.0 Phase angle= 0.0 degr.
Air drag	C_d= 3.5 Jacchia 1965 static density model	-
Solar radiation	4.500E-6 N/M^2 C_r= 1.5 Area= 4.522 E-2 M^2 Mass= 4.700E+1 KG	4.500E-6 N/M^2 C_r= 1.158 Area= 2.827E-1 M^2 Mass= 4.110E+2 KG
Auxiliary	-	Along-track acceleration -3.850 pM/S^2

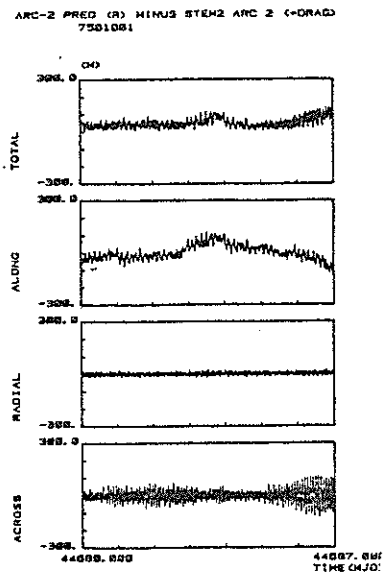
Table 1. Description of the force models defining the reference orbits of STARLETTE and LAGEOS used in the analysis.

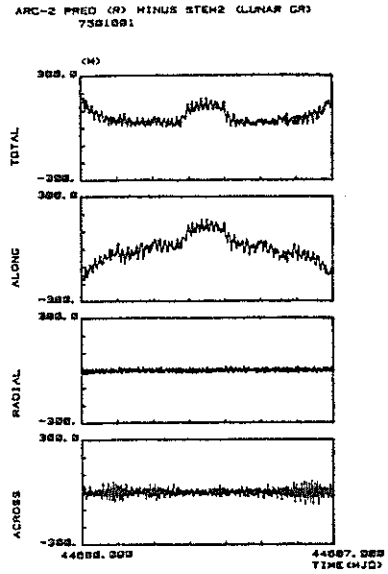
the satellite are to be derived. Maximum errors of 50 to 75 m can be expected. The mathematics involved could hardly impose serious problems in developing software even for very small computer systems.

In case of LAGEOS it might be sufficient to have only one single state vector to predict the orbit during an entire observation campaign period of several months time.

6 Acknowledgment

Discussions with B.A.C. Ambrosius and B.H.W. van Gelder substantially incited these investigations.





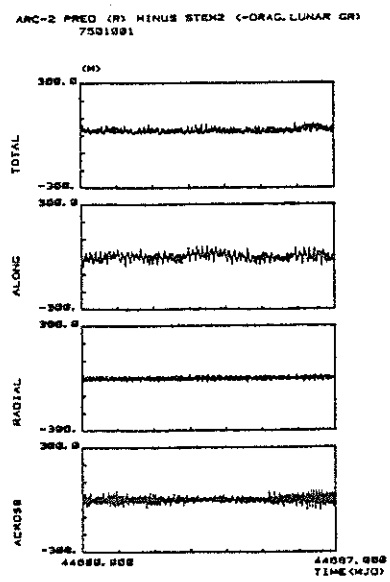


Figure 8.

The solution to the problem of figure 7. The effects of air drag (top left) and lunar gravity (top right) were individually accounted for. If both the effects of air drag and lunar gravity are included in the taylored model, the result of the approximation for this arc is quite acceptable (bottom).

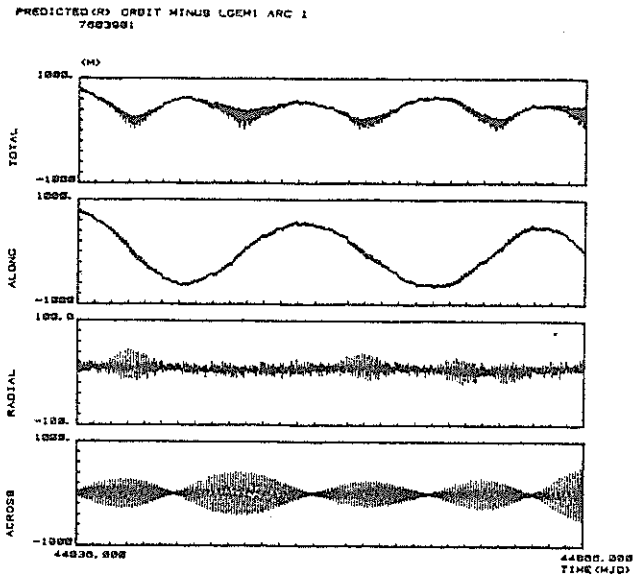


Figure 9. *The approximation technique applied to a 30-day arc of LAGEOS. The taylored model comprises earth gravity coefficients up to degree and order 4 exclusively (top) and together with models for solar and lunar gravity (bottom).*