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 ¨ntroduction

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 co..piled 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.

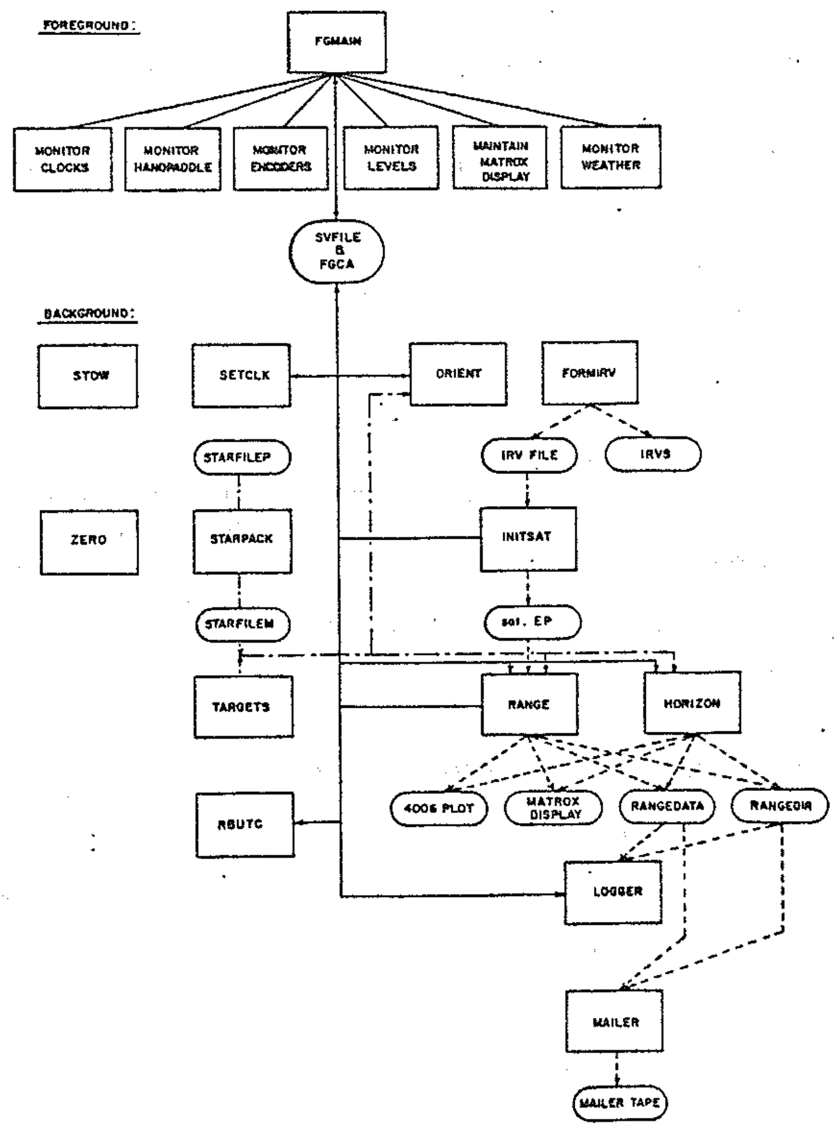

Figure 1: TLRS Software System Overview

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 software: 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
          [illegible]
          [illegible]
     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)=2HEO
      CHEBF(3)=2HO +IPARAM(1)*400B
      GO TO 3510
```

```
 3500 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=MINO(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,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
      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,TO,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,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
      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,*)XO,YO
      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
 2200 CONTINUE
      WRITE(1,5000)(LINEPQ(J),J=1,I)
      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,*)X0,Y0
      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-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)
   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)=X0+(180.0-X)+DELTAX
      XY(2)=Y0-Y
      CALL EXEC(29)
  262 XY(1)=X0-X-DELTAX
      XY(2)=Y0+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,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
      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,TO,SID,T1,T11,TT(4)
      READ(5,*) ID,TO
      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-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)
      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.

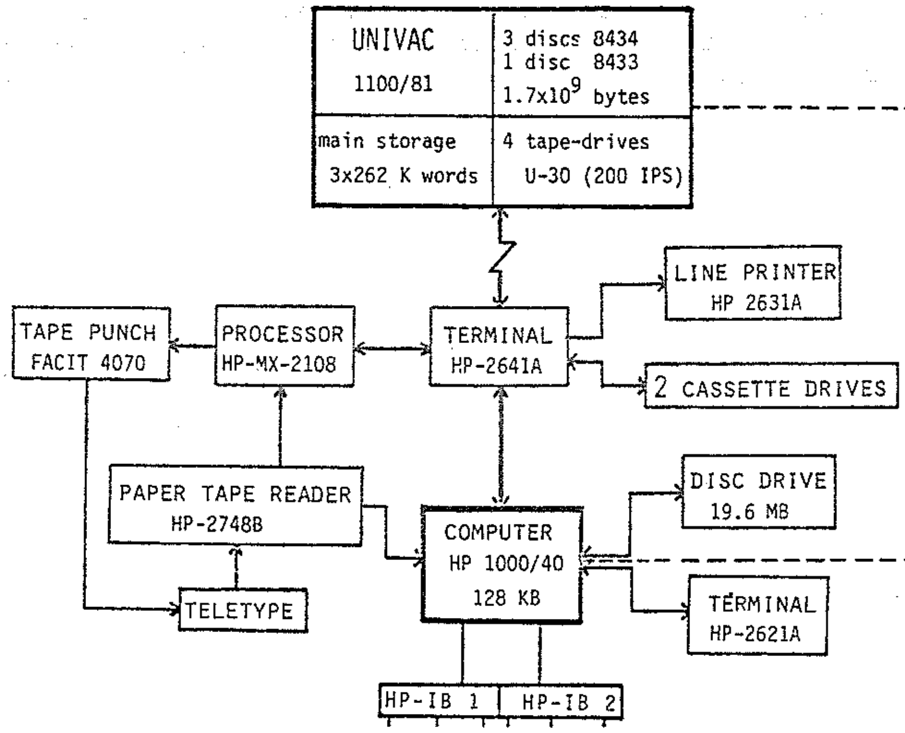

Figure 1:

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.

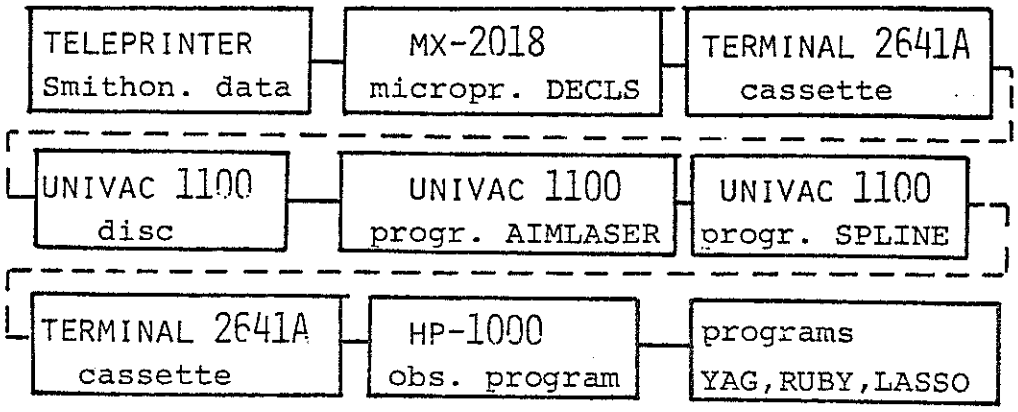

Figure 2:

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 $A_{i,j}^{-1}$ have been approximated analytically:

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

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

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

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

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

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

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.