UNCLASSIFIED

AD NUMBER
ADB134931
LIMITATION CHANGES
TO: **Approved for public release; distribution is unlimited.**
FROM: **Distribution authorized to U.S. Gov't. agencies and their contractors; Critical Technology; AUG 1988. Other requests shall be referred to Air Force Geophysics Laboratory, Attn: PHG, Hanscom AFB, MA 01731-5000. This document contains export-controlled technical data.**
AUTHORITY
AFGL ltr dtd 26 Dec 1989

THIS PAGE IS UNCLASSIFIED

AFGL-TR-88-0163

The CRRES Langmuir Probe and Fluxgate Magnetometer Instrument

Peter R. Harvey

University of California
Space Sciences Laboratory
Berkeley, CA 94720

9 May 1989

Final Report
21 March 1983 - 31 December 1987

Air Force Geophysics Laboratory
Air Force Systems Command
United States Air Force
Hanscom Air Force Base, MA 01731-5000

" This technical report has been reviewed and is approved for publication"

MICHAEL SMIDDY
Contract Manager

NELSON C. MAYNARD
Branch Chief

FOR THE COMMANDER

RITA C. SAGALYN
Division Director

Qualified requestors may obtain additional copies from the Defense Technical Information Center.

If your address has changed, or if you wish to be removed from the mailing list, or if the addressee is no longer employed by your organization, please notify AFGL/DAA, Hanscom AFB, MA 01731. This will assist us in maintaining a current mailing list.

Do not return copies of this report unless contractual obligations or notices on a specific document requires that it be returned.

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER AFGL-TR-88-0163	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) THE CRRES LANGMUIR PROBE AND FLUXGATE MAGNETOMETER INSTRUMENT		5. TYPE OF REPORT & PERIOD COVERED Final 3/21/83 - 12/31/87
		6. PERFORMING ORG. REPORT NUMBER N/A
7. AUTHOR(s) Peter R. Harvey		8. CONTRACT OR GRANT NUMBER(s) F19628-83-K-0029
9. PERFORMING ORGANIZATION NAME AND ADDRESS Space Sciences Laboratory University of California Berkeley, CA 94720		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS 62101F 760112AM
11. CONTROLLING OFFICE NAME AND ADDRESS Air Force Geophysics Laboratory Hanscom AFB, MA 01731 Contract Manager: Michael Smiddy/PHG		12. REPORT DATE 5/9/89
		13. NUMBER OF PAGES 400
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office) Office of Naval Research Resident Representative University of California at Berkeley Office Richmond Field Station Richmond, CA 94804-0001		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A

16. DISTRIBUTION STATEMENT (of this Report)

Distribution limited to US Government agencies and their contractors; Critical Technology; August 1988. Other requests for this document shall be referred to AFGL/PHG, Hanscom AFB, MA 01731-5000.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

CRRES, Langmuir Probe Instrument, Electric Field Instrument, Fluxgate Magnetometer, Hardware, Software, (jhd)

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

A description of the CRRES Langmuir Probe Instrument is given in three parts. The first part is a general description and contains sensor information, telemetry and command formats, and top-level capabilities of the instrument. The second and third sections describe the hardware and software designs. A software listing is included in the appendix. Keywords:

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-LF-014-6601

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Contents

Contents

Illustrations

Tables

1. General Information

The AFGL-701-14 Langmuir Probe Instrument on CRRES has two separate functions, namely to act as 1) a combined Langmuir-Probe/Electric-Field instrument and 2) as the control element for the Fluxgate Magnetometer (AFGL-701-13). This chapter describes the general capabilities of the instrument while later chapters describe the software and hardware in more detail.

The electric-field/langmuir probe part of the instrument connects to two orthogonal double probes, each of which is a pair of separated conductors whose potential difference is measured. One pair of separated conductors are spheres that are located in the spin plane on the ends of wire booms and separated by 100 meters. The other pair of conductors are cylindrical wire boom elements also in the spin plane that are separated by an effective distance of 90 meters.

The fluxgate magnetometer part of the instrument consists of a 3-axis magnetometer sensor which is attached to the end of a 20' rigid boom in the spin plane. The sensor has a plus or minus 45000 nT range and sensitive to less than 1/2 nT.

Both parts of the instrument are controlled by a central computer which telemeters to the ground approximately 96 Langmuir Probe/Electric Field samples and 48 Magnetometer samples per second.

In addition, the instrument coordinates with two other instruments on the spacecraft. The electric field analog signals are repeated to both an AC electric field instrument and a plasma analyser, and magnetic field data is sent in reduced digital form to the plasma analyser.

1.1 Sensor Descriptions

Wire Booms with Spherical Sensors
(AFGL-701-14D and 14E)

The spherical sensors consist of opposing wire booms that are positioned by centrifugal force. Each boom has four major components: the spherical sensor, "stub" and "guard" segments, the cable and the deployment hardware.

The sensor consists of an aluminum sphere which is coated with a conductive material. Inside this shell is a small circuit board containing circuits for both voltage and current measurements.

The cable consists of 8 conductors surrounding a coaxial conductor. These wires are surrounded by a stainless steel braid which acts as an electrically conducting outer shield. The conductors feed voltages and accept signals from the preamplifiers located in the spheres. The mechanical member which supports the centrifugal force load of the probes is stranded Kevlar that is located between the wires and the braid. The wires and outer shield connect electrically at the sensor sphere.

The cable's outer shield is broken into three sections. The section closest to the sphere, which is called the STUB section, is electically connected to a voltage equal to the preamplifier output plus or minus a small DC offset so as to force its potential to be near that of the sphere, and, thus, to minimize the perturbing effect of the cable on the plasma. To guard against the possibility of oscillations being set up by this arrangement due to resonances in the plasma, it is possible to insert a low pass RC filter with 100 Hz rolloff between the preamplifier and the stub section by actuating a latching relay in the main electronic box.

To prevent a positively charged spacecraft from attracting electrons away from the sensors, a small section (10cm), called the GUARD, is placed between the shield and the stub sections. The guard section is adjusted to be more negative than the sphere and stub sections and its potential with respect to the sphere is controlled by the microprocessor via ground command.

Stubs and guards exists symmetrically on both sides of each sphere, with the outer stub and guard being restrained at launch via a "tophat" mechanism. Basically, the outward cable segment is wrapped inside the "hat" while the "brim" of the hat holds onto the sphere. When the centifugal force on the tophat exceeds its ability to hold on, the tophat releases and the cable unravels as the tophat floats away.

The deployment units incorporate two methods of measuring boom length as well as microswitches to reveal both sphere-release and end-of-wire conditions. One method of length measurement involves a simple potentiometer tied to the cable spool, while the second method uses a microswitch tied to cam on the cable feed roller.

Boom deployment is accomplished under microprocessor control that monitors the boom lengths and temporarily stops the deployment of any unit whose length differs from that of its mate by more than a few inches. As a backup in case of microprocessor failure in launch, the boom deployment can be accomplished directly by the spacecraft control system.

Wire Booms with Cylindrical Sensors (AFGL-701-15D and 15E)

Each cylindrical sensor unit is driven by a 28-volt brushless DC motor which powers both a storage reel and a drive roller assembly. The driver roller incorporates a slip clutch which allows the roller to be driven at a slightly faster rate than the storage spool. This feature ensures that a positive tension is maintained on the wire while being dispensed from the mechanism. A potentiometer is driven by the moving wire to give a continuous indication of deployed length. Microswitches are used to signal the full extension point. In addition, at full extension, a lever trips another microswitch which cuts power to the motor to provide automatic shut-off of the mechanism. This function is backed up by a positive mechanical stop to prevent the possibility of backwrapping the wire through switch failure.

The deployment of cylindrical boom units is under direct spacecraft control, not instrument control.

Electric Field measurements using the cylindrical units involve attaching small preamplifier circuits near the base of each antenna unit. Each preamplifier box is then connected to the main electronics box.

DC Preamplifiers for Cylindrical Booms (AFGL-701-14B and 14C)

The DC measurement of cylindrical sensors is accomplished by means of preamplifier circuits located close to the base of the cylinder deployment mechanisms. Bias current to the sensors is supplied by means of a relay inside the DC preamplifier unit. This relay connects a bias voltage to the sensor through a large valued resistor.

Shonstedt Fluxgate Magnetometer (AFGL-701-13-1A, 1B, 2)

The magnetometer measurement is made using a standard fluxgate instrument of which there are two units: a main electronics box and sensor. The sensor is mounted on a rigid 20' long boom in order to get it far enough away from the spacecraft body that the total spacecraft generated magnetic field will have a strength less than 2 nT along the spin axis and 4 nT in the spin plane at the sensor location. The sensor is oriented such that the outputs called BX, BY and BZ are in the spacecraft coordinate system -X, -Z, and -Y. The sensor is also tipped slightly in order to give a spin frequency waveform in the other measurements. This allows for the spacecraft z-axis offset to be calculated.

Table 1. Summary of Physical Attributes

BOX DESCRIPTION	DIMENSIONS (in)	WEIGHT (lbs)	AVG.POWER (W)
13-1a:Fluxgate Elect.	5.6 x 6.5 x 2.1	1.5	.500
13-1b:Sensor	4.7 x 2.8 x 2.8	.7	.050
14A :Electronics	10.5 x 10.5 x 5.0	16.0	8.250
14B :DC Preamp	4.0 x 2.4 x 1.3	0.6	.125
14C :DC Preamp	4.0 x 2.4 x 1.3	0.6	.125
14D :Spherical Boom	14.8 x 6.5 x 7.5	8.2	.500
14E :Spherical Boom	14.8 x 6.5 x 7.5	8.2	.500
15D :Cylindrical Boom	7.5 x 12.2 x 7.3	5.6	----
15E :Cylindrical Boom	7.5 x 12.2 x 7.3	5.6	----

Notes:

1. 14D&E weights include 50 meters of wire and spherical sensor.

2. 15D&E weights include 50 meters of wire.

3. Peak power of 14A is 10.5 Watts (Burst collecting).

1.2 Instrument Interfaces

The 701-14 instrument has a number of interfaces which share signals and data with other experiments. These are described below:

IOWA SOUNDER INTERFACE. The IOWA Sounder is an AC electric field instrument capable of frequencies up to several hundred thousand Hertz. Two analog signals are sent from the -14A box to the IOWA instrument (701-15A). In the voltage mode, these are the Voltages on spheres 1 and 2, while in the current mode these are the current on sphere 1 and ground.

The IOWA instrument incorporates a search coil magnetometer, whose signal is buffered to the -14A box. This signal is available to the Burst computer system only.

LEPA INTERFACE. The Low Energy Plasma Analyser (LEPA) interface consists of a few open collector digital lines which are used to communicate reduced magnetometer information from the Langmuir Probe to the LEPA instrument. This data points out the loss cone to the LEPA instrument, so that it can take high resolution samples in this region.

SPACECRAFT INTERFACE. The spacecraft interface consists of both digital and analog wires. Timing signals provided include telemetry shift clocks at 16 KHz, a 2 KHz clock (8-bit telemetry word timing), and a major frame spike every 4.096 seconds. Data is shifted out in 16-bit packages. The 2 KHz clock is used to maintain timing in the MAIN and BURST computers.

Commands are shifted into the instrument in 16-bit packages using a shift clock and envelope signals. The envelope is wired into a MAIN computer interrupt to provide command ready information.

A sun pulse is also provided by the spacecraft electronics. Using this signal and the 2 KHz clock, the MAIN computer continuously calculates the spacecraft sun angle for use by internal functions such as spin-fitting and bias sweeping.

Analog and bi-level monitors of the instrument health are provided through the main spacecraft interface.

1.3 Analog Electronics

The following is an overview of the analog electronics in the Langmuir Probe instrument (701-14A). For more details, refer to the hardware description in chapter 3.

E-field Sensor Interfacing

The instrument has four main electric field sensors, namely the two spheres and the two cylinders. The spheres are capable of being operated in two modes, one which measures electric fields and the other which measures the current.

The electronics which measure the sphere voltages have a bandwidth of about 1 MHz and the differential signal dynamic range of +/- 200 Volts. This dynamic range is achieved by operating +/- 12 volt preamplifiers from floating power supplies. The output voltage of the sensor is used to driver a unity gain preamplifier which operates from a +/- 100 Volt supply that is referenced to the experiment ground. The output of this supply drives the common terminals of its floating preamp supplies. This circuit allows the sensor potentials to swing plus or minus 100 volts with respect to the spacecraft. To reduce power consumption, the large signal bandwidth is limited to 1000 Hz.

The cylinder measurement electronics differ significantly from the above by operating from fixed +/- 35 Volt supplies rather than from floating power supplies. This limits the dynamic range of the cylinder measurement to about +/- 33 Volts.

These four sensor measurements are called V1 thru V4 and are available to be digitized by both the MAIN and the BURST computer systems (see the digital electronics below).

Sensor Differencing

The main measurement of the electric field is the difference in voltage between the two spheres and the two cylinders. By convention, the voltage difference between V1 and V2 is called V12 and the difference between V3 and V4 is V34.

Difference Amplification

While a "times 50" amplifier is available to any qunatity on the MAIN multiplexor, backup "times 50" amplifiers are provided for V12 and V34. These signals are called V12H and V34H on the MAIN multiplexor.

Differencing Trims

The difference measurements V12 and V34 are trimmable by the operation of a pair of DAC's so that offsets which occur as the result of radiation damage may be adjusted so the measurement stays in the range of the high gain amplifier (see the A/D section below). To adjust these trims use the following commands:

.VTRIM n xx

where n is 1 for V12 trim, and 2 for the V34 trim,
and xx is a 2's complement 8-bit value (+127 to -128).

Band Filters

The V12 signal is fed into a bank of 3 bandpass filters with center frequencies at 32, 256 and 2048 Hz. The filters are called F1, F2 and F3 and are connected to the main multiplexor only.

Each of these bank filters consists of a 2-pole bandpass function followed by a logarithmic amplifier, a full wave rectifier and an integrator. The overall response of the filter bank constant in the frequency range from 32 to 2048 Hz; i.e. the filter band widths are made sufficiently large that there is no loss of signal with frequencies between the center frequencies of the filters. The result of this is to provide for each of the frequency ranges a voltage which is proportional to the logarithm of the power in that frequency range.

Magnetic Field Measurements

The fluxgate magnetometer interface converts the +/- 10 Volt signal from the Shonstedt unit into +/- 5 Volts for the A/D circuitry of the Langmuir Probe. At the same time, these signals are rolled off at 6 Hz for the MAIN telemetry sampling system.

The additional capability is that of amplifying the BY signal from the magnetometer by a factor of 6. BY is in the spacecraft Z axis and may at times be very small. To select this amplification, use the command

.BMODE x

where x = 1 turns ON the amplification
and x = 0 turns OFF the amplifier.
The instrument defaults to OFF (BY not amplified).

Filtering Electronics

There are two basic types of anti-aliasing filters used by the Langmuir Probe instrument, fixed and variable. The fixed filters are used on nearly all quantities which are fed into the MAIN analog multiplexor since the sample frequencies are pretty much dictated by the telemetry capabilities.

The variable filters are used for quantities which are fed into the BURST computer system. Since the Burst system can vary its sample frequency from 10 Hz to 62500 Hz, the completely general anti-aliasing filter should optimally be tunable from 5 to 31250 Hz. Using knowledge about what was reasonable to expect for sampling frequencies for each quantity, the filters were set as shown in the table below.

To set the value of a given filter, use the command as follows:

.FILTER n xx

where n is the filter number (1 thru 7) and "xx" is a value between 1 and 255. The value 1 provides the maximum rolloff while 255 opens the low pass filter to its maximum value. Note: Do not use a filter value of 0! The filters do not work when programmed with 0. The output simply saturates.

Table 2. Programmable Filter Response Characteristics

FILTER	SIGNAL	BURST NAME	MAX(255)	MIN(1)
1.	V12/RI1 AC	BV12AC	11.5 KHZ	45 HZ
2.	VS2/RI2	BV2	11.5	45
3.	V12/RI1	BV12	25.5	98
4.	DIRECT AC	BDIRECT	11.5	45
5.	VS1/-SC	BV1SC	15.0	59
6.	V34	BV34	18.0	70
7.	V34 AC	BV34AC	12.0	47

Filters corresponding to Burst multiplexor quantities and the names on the block diagram.

Multiplexing

In order to maximize capabilities of the instrument using only 7 filters, a number of multiplexors were added to provide options as to which value should be filtered and then sampled. These multiplexors are responsible for some quantity names to have a "/" in them. For example, the quantity "V1/SC" is either V1 or SC (Search Coil) depending upon the setting of multiplexor 3. To set these multiplexors, use the command as follows:

```
.MUX n x
```

where n is the multiplexor (0 thru 3) and x is the value.

Table 3. Multiplexor Settings

MUX	QTY	x=0	x=1	x=2	x=3
0	V12/RI1	RI1	V12		
1	V2/RI2	RI2	V2		
2	KAGC	V2/RI2	V12/RI1	SC	V34
3	V1/SC	SC	V1		

Note:
1. The multiplexor setting is available in the DSC data.
2. MUX 0 and MUX 1 are operated when changing from the Voltage to the Current mode and vice versa. (See Mode Switching.)

Relay Control

A number of the options of the sensor measurement and control electronics are implemented using relays as the switching elements. These switches are shown on the instrument block diagram. To set or reset relay number "n", use the command

.SET n   or   .RESET n

It is important to note:

1. Relays 0 and 1 are used to "steer" current for setting and resetting other relays, so their state may change when operating other relays.

2. Relays 0, 1, 7, 8, 9, 16, 18 and 19 are changed by the switching from the Voltage to the Current mode or vice versa. See Mode Switching.

Sensor Bias Electronics

An important capability of the instrument is that of applying bias currents to the sensors. The impedance between the sensor and the plasma is a non-linear function of the current flowing between them and it exhibits a minimum at a value of bias current which depends upon the plasma conditions. Thus, the accuracy of the electric field instrument can be maximized by applying the optimum value of bias current to the sensor.

The analog circuitry which accomplishes sensor, stub and guard biasing consists of eight 8-bit DACs which produce a bipolar effect upon the eight biasing circuits (2 spheres, 2 guards, 2 stubs and 2 cylinders). The value of the sensor bias currents are set either by ground command or by on-board algorithms.

The sensor bias voltages are connected to the sensors through 100 MegaOhm resistors. Ground commanded relays can be operated to remove the biasing capability for the cylinders only. The spheres are always biased when in the voltage mode.

To set any of these DAC's, use the appropriate command:

.BIAS n xx

.STUB m xx

.GUARD m xx

where *n* is the sensor number (1 thru 4),

m is the sphere number (1 or 2),

and xx is a 2's complement 8-bit value (+127 to -128).

The voltages output are shown in Table 4.

Table 4. Bias, Stub and Guard Characteristics

DAC	-128	+127	GAIN	OFFSET
BIAS1	-35.31	35.50	.2777	.2345
BIAS2	-35.24	35.55	.2776	.2960
BIAS3	-35.23	35.45	.2772	.2515
BIAS4	-35.19	35.58	.2776	.3360
STUB1	- 1.21	1.25	.0096	.0210
STUB2	- 1.23	1.22	.0096	-.0030
GUARD1	-35.20	35.41	.2769	.2415
GUARD2	-35.33	35.47	.2776	.2075

Analog to Digital Conversion

Selected potential difference measurements as well as analog outputs of filter banks, the potentials of each sensor, and other analog quantities such as boom lengths, motor currents, etc. are fed through two multiplexors, one for telemetry sampling by the MAIN computer and one for high rate sampling by the BURST computer.

Each multiplexor is followed by a pair of op amp circuits, one with unity gain and the other with a gain of about 50. The outputs of these circuits are then fed into a final multiplexor with which the processor can select one signal for digitization. The gain decision for the MAIN computer is performed in software by actually digitizing the low gain value and then re-digitizing either the high or low gain value. On the BURST computer system, the gain decision is made automatically by comparators whose outputs drive the last multiplexor.

The output of these final multiplexors go to fast 12-bit A/D converters, one for each computer system.

The computed gain of the "times 50" amplifiers is 51.12 and -49.75 for the MAIN and BURST systems, respectively.

1.4 Digital Electronics

General

The instrument digital section consists of two microprocessor systems arranged in a master-slave relationship. The master processor, called the MAIN processor, is responsible for most of the mission operations, namely telemetry formatting, command reception and execution, sensing burst conditions, boom deployment, current sweeps and other control items. The slave microprocessor, called the BURST, is responsible for high frequency data sampling and storage.

The MAIN system consists of a SANDIA 3000 microprocessor which is a radiation tolerant version of the Intel 8085 only in CMOS. On the system buss is 8K bytes of ROM and 4K of RAM, plus a host of input and output ports, of course.

The BURST system also uses a SANDIA 3000 processor but has only half the amounts of ROM and RAM in which to store programs; i.e. 4K bytes of ROM and 2K of RAM. In addition to the normal RAM, the BURST system includes a memory unit of 192K bytes used for storing bursts of digitizations which it later plays back to the MAIN system on request.

Mode Control

The Main computer controls the mode of the spherical sensors in two ways, either automatically or manually. The mode of these sensors appears in the Fast Digital Monitor data.

The executive will automatically switch modes back and forth at programmable time intervals (measured in the number of spacecraft rotations). These intervals are selected using the command:

.EMODE n m

This instructs the instrument to operate for 2**(n-1) spins of the current mode and 2**(m-1) spins of the voltage mode. If either n or m is zero, the corresponding mode is not used. If n = m = 0, mode switching is disabled. For example, to select voltage mode only use ".EMODE 0 1" . The default is ".EMODE 7 7" which means the instrument will flip modes every 64 spins (32 minutes).

The executive changes the modes only at the beginning of a spin period, phased with the spin-fitting software so that no data is lost. The first mode change occurs at the first spin period boundary following the EMODE command entry.

Two vestigial commands worth noting are as follows:

.VMODE   selects the Voltage mode and
.IMODE   selects the Current mode.

As soon as these commands are entered, the instrument will flip all the necessary relays and multiplexors in order to configure itself for that mode. However, these commands control neither the current mode sawtooth nor the spin-fit calculations (both of these ARE controlled in the automatic mode). Hence, if you ask for the IMODE, you should also enable the sawtooth using SAW commands. And if you ask for the VMODE, you probably will want to enable the spin-ftting.

The MAIN computer can configure the instrument such that the health of nearly all of the sensor electronics can be determined. This automatic test sequence can be invoked using either the ".TEST" or ".CALIBRATE" commands. The sequencer begins at the start of a telemetry major cycle (every 32 seconds) and lasts about a minute. Simultaneous measurements of V1 thru V4, their voltage differences, the BIASing circuitry and so forth are recorded by both the MAIN and the BURST circuitry. The TEST sequence requires about 1 minute, including the BURST playback time.

Telemetry Processing

The MAIN computer processes telemetry for both the Langmuir Probe/Electric Field part of the instrument and the Fluxgate Magnetometer part. Hence, there are two separate and independent telemetry formats, one for each function, which share the total telemetry allotment from the spacecraft. These formats are detailed in Figures 2 through 6 in the software chapter and are further described below.

L-Probe/E-Field (LPEF) Formatting. The LPEF part of the telemetry is a table driven (and therefor programmable) format. Two tables, called HX and LX, define the High Rate and Low Rate sampling profile, respectively. The HX table defines 16 channels through which any quantity available to the MAIN processor can be sampled at 4 Hz. Similarly, the LX table defines 32 channels through which any quantity can be sampled at 1 Hz.

The MAIN computer manages 16 16-byte formatting tables in memory, 10 in ROM and 6 in RAM. The ten ROM formats are numbered 0 thru 9, while the six RAM formats start at 10. (Of the 10 ROM formats designed into the software, only 3 were actually filled, namely 0, 1 and 2.) To select which format tables to use, there are three commands:

```
.FORMAT n m
.VFMT   n m
.IFMT   n m
```

where n is the table to use for the HX list and m is the table for the LX list. FORMAT says that the list is to be used in both Voltage and Current modes. VFMT says the list is to be used ONLY in the Voltage mode, and IFMT says the list is to be used ONLY in the Current mode. These provide the capacity to have different sampling formats for different modes.

The six RAM lists are located in a continuous section of memory which is 64 bytes long. To load a format one uses the "INDEX n" command which selects location n (0 < n < 64) and "QTY q" which describes what to sample. For example, to load format 11 with alternating V1 and V2F samples:

```
.INDEX 16
.QTY V1
.QTY V2F
.QTY V1
.QTY V2F
.QTY V1
.QTY V2F
     .
     .
     .
.QTY V1
.QTY V2F
```

Each channel descriptor "q" can be either a MAIN multiplexor quantity or what we call a RAM quantity. To select a multiplexor value, simply name the quantity, such as "V1" or "V2F" above.

RAM quantities are simply values taken from the memory of the MAIN computer system. These are of interest mainly for diagnostic purposes when it is important to have a high bandwidth of information regarding some variables in the computer memory. Since there is a lot of RAM and only 6-bits of possible indexation, the RAM quantities use a programmable 16-bit base address called RAMBASE. The 6-bits are added to RAMBASE to produce an effective address from which 13-bits are retrieved (low byte first then 5-bits of the high byte).

```
              MAIN MEMORY
             .---------.
RAMBASE-->!          ! 0
          !          ! 1
          !          ! 2
          !          ! .
          !          ! .
          !---------!
          !          ! m
          !--+------!
          !  !       ! m+1
          !  '------!
          !          !
          !          ! .
          !          ! .
          !          ! 62
          !          ! 63
          '---------'
```

To select a RAM quantity 0 thru 63, use the for ".QTY RAM+n". To set the value of RAMBASE, use the command

.RAMBASE n

(This allows 11-bit values to be loaded into RAMBASE. For other values, use the .LOAD facility).

Finally, quantity descriptors include one bit which, if enabled, allows playbacks to preempt them. To indicate that playbacks may preempt the channel, add "PE" to the command. For example,

```
.QTY V1 PE
```

Sawtooth Generation. In the Current mode, the MAIN computer's SAWTOOTH module is enabled by the executive program to generate linear sweeps on the sphere BIAS voltages.

Playbacks. There are two basic types of playbacks, those from the MAIN and those from the BURST. MAIN transmissions always take priority over BURST playbacks since the latter take much longer. One bit, called the "MAIN/BURST XMIT" bit, is used to distinguish between these two in the telemetry stream.

While decoding playback telemetry, one must watch for transitions between the states of this "MAIN/BURST XMIT" bit. For example, if the BURST is playing back, one must switch to a MAIN playback if the MAIN/BURST XMIT bit goes to a 1 (MAIN).

If the "MAIN/BURST XMIT" bit starts out as 1, the lower priority BURST transmission will not override it and will follow immediately after the MAIN playback finishes.

Command Reception

The chief command capabilities of the instrument are implemented via the serial digital commands. Each command is a 16-bit value which is shifted into the instrument using a standard CLOCK, DATA, and STROBE protocol (see the CRRES-225 document). Each command interrupts the MAIN processor which either executes it or passes it to the BURST processor for interpretation.

Hex Digit Commands. The command capabilities while in orbit are incredibly limited compared to those available through the Ground Support Equipment.

Commands are uplinked to the spacecraft in what is called a "command pass", which ranges from a few minutes to hours in duration. Prior to a command pass is a command planning meeting in which all command sets are determined. These command sets are relayed to the specific ground station (worldwide) which will be in contact with the spacecraft. Voice communication is used between the Sunnyvale operation and the ground station in order to invoke specific command sets as needed.

The primary limitation with the system is that one cannot send an arbitrary command to an instrument in real-time. (One could concieve a plan in which one sent all permutations of 16-bits to the ground station, but there is a 1000 command limit per instrument.) To get around this problem, seventeen "hex-digit" commands were incorporated into the instrument. These commands are simply the digits plus an "enter" command. To send any 16-bit command, one must convert the command bits into a series of hex digit commands. The on-board microprocessor will act on a string of digit commands just as if it had received the standard serial input command.

For example, to enter the 16-bit command '5678' one would send the following sequence of hex digit commands:

```
DIGIT 5
DIGIT 6
DIGIT 7
DIGIT 8
ENTER
```

This method is obviously slow (as many as five commands will have to be sent instead of one), and error prone (five times the error rate of one), but it is the only way to do real-time commanding given the design of the ground stations we must use.

Burst Sampling Formats

Just as there are 16 sampling formats available to the MAIN processor, 16 have been implemented in the BURST system. As in the MAIN, 10 formats (0 thru 9) are located in ROM and therefor cannot change. The remaining six sample formats (A thru F) are programmable on the fly. But unlike the MAIN, the BURST system provides sampling formats of varying length. They can have as few as zero and as many as 64 quantities. (Note: The total number of quantities in the 6 RAM lists is limited to 64.)

The procedure for requesting a sampling list involves two separate operations, that of 1) list selection and 2) list definition. To select a format to be sampled, one uses the command "BFMT n", where n is 0 thru 9 for the ROM formats and 10 thru 15 for the RAM formats.

To define a RAM list, first select the list you want to define and then enter the sample series Q1..Qn using the "BQTY q" command as follows:

```
BFMT f
BQTY q1
BQTY q2
BQTY q3
.
.
BQTY qn
```

The format number "f" must be in the range 10 thru 15 decimal in order for anything to change. (You can't change ROM, of course).

Once defined, sample formats may be selected at will by using the BFMT command; i.e. you don't have to re-define the RAM list each time you use it. You can switch around between the different ROM and RAM formats as conditions may warant.

Upon reset, the BURST cpu defines the RAM formats by copying an area of its ROM over into the RAM lists. This provides for the immediate use of all 16 formats by the user from the start. These formats are given in table 5.

Table 5. Burst Default Sampling Formats

FORMAT	QTY LIST	DESIRED FREQUENCY
0.	V12/RI1	60 KHZ
1.	V12/RI1	30 KHZ
	V1/SC	
2.	V12/RI1	30 KHZ
	V34	
3.	V12/RI1	20 KHZ
	V34	
	V1/SC	
4.	V12/RI1	120 HZ
	V34	
	BX	
	BY	
	BZ	
5.	BZ	
	BX	
	BY	
	V3	
	V4	
	V34	
	V34 AC	
	V1SC	
	V12/RI1 AC	
	V2	
	V1	
	V12/RI1	
	DIRECT	
	AGCU	
	GUARD	
	STUB	
6-F.	[EMPTY]	

Burst Sample Frequency Control

The BURST computer is capable of sampling a list of quantities at a number of frequencies up to 60 KHz. These frequencies are listed in the Table below. It is important to observe that the frequency is for the whole list, not individual quantities, and thus the size of the sample list defines the maximum frequency at which it can be sampled.

The command which set the BURST sample frequency is

BFREQ f

where f is a frequency code from 0 to 15. (See the table below for the equivalent frequency.) If a frequency is requested which is greater than the maximum for a given list, the BURST processor will sample the list at the highest frequency possible for that list. Thus "BFREQ 15" always guarantees that a list will be sampled at the very highest sample frequency possible.

Burst playbacks always contain the ACTUAL frequency code of the playback data, not what was commanded. Thus, there can be no confusion over the sample frequency of playback data.

The frequency code is independent of the sampling lists so that one does not need to re-command the frequency when one changes lists. Also, the frequency code is not modified by the BURST cpu even when it describes a frequency which is impossible for a given list. Thus, one can set the frequency either before or after one defines or selects the sample list without fear that these other commands may effect the frequency.

Table 6. Burst Frequencies

CODE	FREQUENCY (Hz)	QTY LIMIT
15	62,500	1
14	62,500	1
13	31,250	2
12	20,833	3
11	15,625	4
10	10,417	6
9	6,250	10
8	3,125	20
7	2,000	30
6	1,000	60
5	500	64
4	200	64
3	100	64
2	50	64
1	20	64
0	10	64

2. Software

This chapter describes the software for both the Langmuir Probe Instrument (AFGL-701-14) and the Fluxgate Magnetometer (AFGL-701-13). This is such a large task that it requires its own chapter, one that is separate from the scientific and hardware oriented descriptions of the instrument. At the same time it would be unwise to clutter these other descriptions with the particulars of the instrument's inner mechanisms.

Both the MAIN computer and the BURST computer programs are modularized according to the best description of their function. For example, the electric field module (ELE.A) handles all E-Field/Langmuir telemetry formatting and commands dealing with that part of the instrument. The magnetometer data formatting and commands are handled by the MAG.A module, etc. Table 7 provides a list of the modules in the system. The balance of the chapter describes the detail for each module.

Table 7. MAIN and BURST software modules

MAIN	Major Function
EXEC.A	Executive Program
IO.A	Input/Output Module
BKG.A	Background Processing Manager
ELE.A	Electric Field/Langmuir Probe Manager
MAG.A	Magnetic Field Manager
UTIL.A	Small Utilities
DEP.A	Deployment Manager
PLA.A	Plasma (Low Energy) Instrument Manager
LD.A	Program Load Manager
BUR.A	BURST Triggering Manager
SWP.A	Bias Sweep Manager
FIT.A	Spin Fitting Manager
SPIN.A	Spin Fit Calculator
MATRIX.A	Matrix Solver
TRIG.A	Trigonometric Functions
FFP.A	Fast Floating Point Utility

BURST	Major Function
BEXEC.A	Executive Program
BIO.A	Input/Output Module
BCMP.A	Burst Microcode Compiler
BSMP.A	Sampling Control Module
BFMT.A	Format Manager
BLD.A	Program Loading Manager

2.1 Main Executive Module

The executive module is responsible for coordinating the activities of the Main computer system and its slave, the Burst system. It forms the "foreground" part of the instrument, that part in which non-real-time calculations such as spin-fits can be done. While instrument "autonomic" functions like sampling and telemetry formatting proceed in the "background" under interrupts, the foreground is free for data analysis, mode switching decisions, ground loaded programs, etc.

The EXEC module is responsible for the following functions:

1) Defining the instrument initial state;
2) Controlling the instrument mode switching (Langmuir Probe versus E-Field versus CALIBRATE);
3) Coordinating the mode switching with on-orbit spin fitting and bias sweeps;
4) Running ground loaded programs.

The MAIN module has the following entry points :

EXEINIT Jumped to when the processor is reset.

EXEANG Called when the background has changed the sun angle. On Entry: [A] = new sun angle.

EXEDSC Called when the Digital Subcom wants a status word from the EXEC module. On Entry: [A] is the index into the status word requested. On exit, [A] contains the status byte.

INITIALIZATION / EXECUTIVE LOOP. The module 1) clears the RAM, 2) initializes all of the modules it controls, 3) sends in a command sequence to define the initial instrument state, 4) resets the "EXEVECT" to null and 5) begins the MAIN executive loop. In this loop, it checks to see if the executive vector has been armed, and executes the vector is so. This allows ground loaded programs to gain control of the foreground (otherwise programs run 1 interrupt deep). The executive loop mainly just calls three routines to share the CPU between the spin fitting (FITEXEC), the bias sweeping (SWPEXEC) and the mode decision (DECMODE).

A technique is used in this executive loop to lower the power of the system by 1) halting the processor when it is not being used and 2) by stopping in RAM not ROM. This latter part works because the ROMS are turned ON only when they are addressed, so to turn them OFF, one simply has to stop the CPU in the RAM.

MODE DETERMINATION. The executive mode switching is controlled by two 16 bit counters called VTIME and ITIME and a 16 bit register called MODTIM. VTIME and ITIME are the number of spins in which the instrument should be in either the Voltage or the Current mode. MODTIM is the count remaining in the present mode.

The EMODE command simply loads VTIME and ITIME with the 2**N-1 calculations (as described in the LP.DOC). It also sets MODTIM to 1 so that the mode will switch in the next spin period. If one chooses to operate the instrument with other timing, one merely has to load values for VTIME and ITIME using the loader commands.

Mode switching is performed by a combination of the EXEANG and DECMODE routines. The EXEANG routine compares the current sun angle reported by the background to the angle at which the mode can be switched (CHGANG). This mode-change-angle is initially set to the beginning of V12 spin fits less 11 degrees, or 1/2 the period between fit samples. When the sun angle equals the selected angle, MODTIM is decremented. DECMODE simply checks for MODTIM == 0 to decide to change the mode. (Note: EXEANG runs in the background and DECMODE in the foreground. Since spin fits and bias sweeps are occurring in the foreground, one cannot be sure to "see" every spin angle from the foreground.)

CALIBRATE SEQUENCING. The TEST/CALIBRATE command is implemented by sending in a series of commands just like the initializing sequence. The CALSEQ is the list of commands which implement the test mode. Delay timing is implemented using a modified CALIBRATE command (91xx instead of 90xx). The delay routine "SYNCWT" implements this command which simply waits until the minor frame matches the data portion of the command. By inserting these "SYNC" commands in the CALSEQ, one can set the timing of the events.

2.2 Background Management

The Main Computer system cpu time is subdivided into the standard foreground/background processing profile. Management of the background (interrupts) is accomplished in a single module which organizes and further subdivides cpu time between the several modules which operate in the background.

The major functions of background management are in telemetry formatting, command reception, time and sun angle determination. Other functions include cycling the Kelley AutoGain circuit, and sampling instrument temperatures for the digital subcom. The background manager calls each of its modules when its time for them to sample some data or to telemeter some data.

The timing requirements implemented by the module are in Table 8. The sampling schedule is defined in Figure 1, and the real-time telemetry formats are given in Figures 2 through 4.

Table 8. Module Sampling Frequencies

Frequency	Action
16 Hz	: Sampling of BX,BY,BZ
32 Hz	: Sampling of LEPA
32 Hz	: Sampling for burst Triggering
64 Hz	: High Rate Electric Field (HX)
32 Hz	: Low Rate Electric Field (LX)
64 Hz	: Sawtooth Generator for Langmuir Mode
64 Hz	: Command Execution
64 Hz	: Sun Angle Phase Lock Loop
8 Hz	: Deployment

	0 1	2 3	4 5	6 7	8 9	10 11	12 13	14 15	16 17	18 19	20 21	22 23	24 25	26 27	28 29	30 31
0+	SYNC	SWP	T/M	SWP	T/M	SWP	T/M	SWP	HX	SWP	BUR	SWP	SUN	SWP	CD/SW	SWP
32+	PLA	SWP	T/M	SWP	T/M	SWP	T/M	SWP	HX	SWP	LX	SWP	SUN	SWP	CD/SW	SWP
64+	MG	SWP	TM/MS	SWP	T/M	SWP	TM/ME	SWP	HX	SWP	BUR	SWP	SUN	SWP	CD/SW	SWP
96+	PLA	SWP	T/M	SWP	T/M	SWP	T/M	SWP	HX	SWP	LX	SWP	SUN	SWP	CD/SW	SWP
128+	AUTO	SWP	T/M	SWP		SWP		SWP	HX	SWP	BUR	SWP	SUN	SWP	CD/SW	SWP
160+	PLA	SWP	T/M	SWP		SWP		SWP	HX	SWP	LX	SWP	SUN	SWP	CD/SW	SWP
192+	MG	SWP	TM/MS	SWP		SWP	ME	SWP	HX	SWP	BUR	SWP	SUN	SWP	CD/SW	SWP
224+	PLA	SWP	T/M	SWP		SWP		SWP	HX	SWP	LX	SWP	SUN	SWP	CD/SW	SWP

MAIN TELEMETRY SAMPLING FORMAT

NOTES:
- SYNC = MINOR FRAME SYNC CALL MADE TO TELEMETRY MODULES
- T/M = TIME TO LOAD THE OUTPUT SHIFT REGISTERS IN PREPARATION FOR TELEMETRY CLOCKS
- HX = ELECTRIC FIELD HIGH FREQUENCY LIST SAMPLING
- LX = ELECTRIC FIELD LOW FREQUENCY LIST SAMPLING
- SUN = SUN ANGLE UPDATES
- CD/SW = COMMAND EXECUTION IF COMMAND READY / SAWTOOTH GENERATION
- BUR = BURST TRIGGER ALGORITHM
- AUTO = KELLEY AUTO GAIN CIRCUIT UPDATE / DEPLOYMENT CHECKING / TEMPERATURE MONITORS
- PLA = LOW ENERGY PLASMA SAMPLING/CALCULATION
- MG = MAGNETOMETER GAIN SAMPLES TAKEN AND EVALUATED
- MS - MAGNETOMETER SAMPLES TAKEN
- ME = MAGNETOMETER TELEMETRY ENCODING PERFORMED
- SWP = SWEEP SAMPLING PERFORMED (BACKGROUND PROGRAMMABLE VECTOR)

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY

PROJECT: AFGL-201-14
TITLE : LANGMUIR PROBE INSTRUMENT
SECTION: SYNCHRONOUS SAMPLING FORMAT
FILE : SAMP.DWG

DESIGNER: PETER HARVEY
DATE : 27 JUL 1983 REV3: 18 SEP 1985
REV1 : 21 DEC 1984 REV4: 8 APR 1986
REV2 : 23 APR 1985

Figure 1. Synchronous Sampling Format

MINOR FRAME 0

WORD	6	7	10	11	14	15
0+	HX0	:	HX1	DSC0	MAG0	MAG1
32+	HX2	:	HX3	MAG2	MAG3	MAG4
64+	HX4	:	HX5	MAG5	MAG6	MAG7
96+	HX6	:	HX7	MAG8	MAG9	MAG10
128+	GAINS	FDM				
160+	LX0	:				
192+	LX1	LX2				
224+	: LX3	: *				

* = GAINS FOR LX0 TO LX3

MINOR FRAME 1

WORD	6	7	10	11	14	15
0+	HX8	:	HX9	EXP	MAG11	MAG12
32+	HX10	:	HX11	MAG13	MAG14	MAG15
64+	HX12	:	HX13	MAG16	MAG17	MAG18
96+	HX14	:	HX15	MAG19	MAG20	MAG21
128+	GAINS	SFR				
160+	LX4	:				
192+	LX5	LX6				
224+	: LX7	: *				

* = GAINS FOR LX4 TO LX7

MINOR FRAME 2

WORD	6	7	10	11	14	15
0+	HX0	:	HX1	DSC1	MAG22	MAG23
32+	HX2	:	HX3	MAG24	MAG25	MAG26
64+	HX4	:	HX5	MAG27	MAG28	MAG29
96+	HX6	:	HX7	MAG30	MAG31	MAG32
128+	GAINS	FDM				
160+	LX8	:				
192+	LX9	LX10				
224+	: LX11	: *				

* = GAINS FOR LX8 TO LX11

.
.
.

MINOR FRAME 7

WORD	6	7	10	11	14	15
0+	HX8	:	HX9	EXP	MAG33	MAG34
32+	HX10	:	HX11	MAG35	MAG36	MAG37
64+	HX12	:	HX13	MAG38	MAG39	MAG40
96+	HX14	:	HX15	MAG41	MAG42	MAG43
128+	GAINS	SFR				
160+	LX28	:				
192+	LX29	LX30				
224+	: LX31	: *				

* = GAINS FOR LX28 TO LX31

NOTES:

1) 12-BIT QUANTITY PACKAGING

X0	:	X1

LEAST SIG. BITS OF X0 — MOST SIG. BITS OF X1

2) DIGITAL SUBCOM (DSC) DATA ALTERNATES WITH FLUXGATE MAGNETOMETER DATA

3) FAST DIGITAL MONITOR (FDM) BITS ALTERNATE WITH SPIN FIT RESULT (SFR)

4) GAIN BYTES ARE ALL CODED LEFT TO RIGHT FOR THE PRECEDING N QUANTITIES. E.G. THE FIRST GAIN BYTE IS AS FOLLOWS:

G0:G1:G2:G3:G4:G5:G6:G7

5) HXn AND LXn REFER TO HIGH AND LOW FREQUENCY TELEMETRY CHANNELS. THERE ARE 16 HIGH FREQUENCY CHANNELS AND 32 LOW FREQUENCY CHANNELS. THE QUANTITY WHICH IS SAMPLED IS IS DETERMINED BY THE VALUE IN CORRESPONDING TABLE ENTRIES IN HCHAN AND LCHAN (SEE DIGITAL SUBCOM).

6) FDM DEFINED AS "PTCCXSEM" WHERE
"P" = PLAY BACK STATUS (1=IN PROGRESS)
"T" = TEST/CALIBRATE (1 = IN PROGRESS)
"CC" = BURST CONDITION CODE
00 = OFF
01 = SEARCHING
10 = COLLECTING
"X" = MAIN/BURST TRANSMISSION (1=MAIN)
"S" = BIAS SWEEP IN PROGRESS
"E" = COMMAND COUNT ERROR (1 = ERROR)
"M" = VOLTAGE/CURRENT MODE (1=CURRENT)

7) EXP PROVIDES TELEMETRY FOR ON-BOARD EXPERIMENTS. FORMAT TBD.

8) THE FORMAT OF FLUXGATE MAGNETOMETER DATA IS DETAILED IN A SEPARATE DWG.

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY

PROJECT: AFGL-701-14
TITLE : LANGMUIR PROBE INSTRUMENT
SECTION: TELEMETRY FORMAT SPEC.
FILE : FMT.DWG

DESIGNER: PETER HARVEY
DATE : 24 JUN 1983 REV4: 15 AUG 85
REV1 : 27 JUL 1983 REV5: 18 SEP 85
REV2 : 1 AUG 1983 REV6: 8 APR 86
REV3 : 21 DEC 1984

Figure 2. General Telemetry Format

FLUXGATE MAGNETOMETER FORMAT
LOGICAL ARRANGEMENT (1/2 SECOND)

BX0L	BY0L	BZ0L	MD
BX0	BY0	BZ0	
BX1	BY1	BZ1	
BX2	BY2	BZ2	
BX3	BY3	BZ3	
BX4	BY4	BZ4	
BX5	BY5	BZ5	
BX6	BY6	BZ6	
BX7	BY7	BZ7	
GX			
GY			
GZ			

FLUXGATE MAGNETOMETER FORMAT
PHYSICAL ARRANGEMENT (1/2 SECOND)

0	BX0L	5	BX0	23	BX4	41	GX
1	:	6	:	24	:	42	GY
2	BY0L	7	BY0	25	BY4	43	GZ
3	BZ0L	8	BZ0	26	BZ4		
4	: MD	9	:	27	:		
		10	BX1	28	BX5		
		11	BY1	29	BY5		
		12	:	30	:		
		13	BZ1	31	BZ5		
		14	BX2	32	BX6		
		15	:	33	:		
		16	BY2	34	BY6		
		17	BZ2	35	BZ6		
		18	:	36	:		
		19	BX3	37	BX7		
		20	BY3	38	BY7		
		21	:	39	:		
		22	BZ3	40	BZ7		

NOTES:

1) BX0L, BY0L, BZ0L : LOW GAIN VALUES ONLY (12-BIT)
: WHICH ARE TAKEN AT THE
: SAME TIME AS THE BX0-BZ0 SET.

2) BX0-BX7
BY0-BY7
BZ0-BZ7
: AUTO-GAIN VALUES (12-BIT)
: (APPROPRIATE GAIN IS DECIDED WHEN EACH
: SAMPLE IS TAKEN)

3) GX, GY, GZ : GAIN BITS FOR BX, BY, BZ (1=HIGH GAIN)
: CODED LEFT TO RIGHT AS:

: G0:G1:G2:G3:G4:G5:G6:G7

4) MD : 4-BIT MODE INFORMATION [...Y]
: WHERE Y=1 MEANS BY IS AMPLIFIED -- TIMES

5) THE FORMAT TOTALS 9*3 12-BIT VALUES, 8*3 1-BIT VALUES,
PLUS 4 MODE BITS.
THIS REQUIRES 352 BITS OR 44 BYTES.

6) EACH 12-BIT VALUE IS IN 2'S COMPLEMENT FORM HAVING A RANGE
THEREFORE OF -2048 TO +2047. THE LOW GAIN 12-BIT VALUES
CORRESPOND TO A +/- 45000 nT RANGE AND THE HIGH GAIN VALUES
CORRESPOND TO A +/- 900 nT RANGE.

UNIVERSITY OF CALIFORNIA, BERKELEY SPACE SCIENCES LABORATORY
PROJECT: AFGL-701-14 TITLE : LANGMUIR PROBE INSTRUMENT SECTION: FLUXGATE MAGNETOMETER FORMAT VERSION 2
NAME : PETER R HARVEY DATE : 21 DEC 1984 REV1 : 6 FEB 1985

Figure 3. Fluxgate Magnetometer Telemetry Format

MINOR FRAME	DSC	HEX	QUANTITY
0	0	0	MAJOR FRAME COUNT (MOD 8=0)
2	1	1	HCHAN(0)
4	2	2	HCHAN(1)
6	3	3	HCHAN(2)
8	4	4	HCHAN(3)
10	5	5	HCHAN(4)
12	6	6	HCHAN(5)
14	7	7	HCHAN(6)
16	8	8	HCHAN(7)
18	9	9	HCHAN(8)
20	10	A	HCHAN(9)
22	11	B	HCHAN(10)
24	12	C	HCHAN(11)
26	13	D	HCHAN(12)
28	14	E	HCHAN(13)
30	15	F	HCHAN(14)
0	16	10	MAJOR FRAME COUNT (MOD 8=1)
2	17	11	HCHAN(15)
4	18	12	LCHAN(0)
6	19	13	LCHAN(1)
8	20	14	LCHAN(2)
10	21	15	LCHAN(3)
12	22	16	LCHAN(4)
14	23	17	LCHAN(5)
16	24	18	LCHAN(6)
18	25	19	LCHAN(7)
20	26	1A	LCHAN(8)
22	27	1B	LCHAN(9)
24	28	1C	LCHAN(10)
26	29	1D	LCHAN(11)
28	30	1E	LCHAN(12)
30	31	1F	LCHAN(13)
0	32	20	MAJOR FRAME COUNT (MOD 8=2)
2	33	21	LCHAN(14)
4	34	22	LCHAN(15)
6	35	23	LCHAN(16)
8	36	24	LCHAN(17)
10	37	25	LCHAN(18)
12	38	26	LCHAN(19)
14	39	27	LCHAN(20)
16	40	28	LCHAN(21)
18	41	29	LCHAN(22)
20	42	2A	LCHAN(23)
22	43	2B	LCHAN(24)
24	44	2C	LCHAN(25)
26	45	2D	LCHAN(26)
28	46	2E	LCHAN(27)
30	47	2F	LCHAN(28)
0	48	30	MAJOR FRAME COUNT (MOD 8=3)
2	49	31	LCHAN(29)
4	50	32	LCHAN(30)
6	51	33	LCHAN(31)
8	52	34	RAMBASE POINTER *
10	53	35	
12	54	36	EXECUTIVE VERSION
14	55	37	MODE TIMING
16	56	38	MODE CHANGE ANGLE
18	57	39	EXECUTIVE SPARE 0
20	58	3A	EXECUTIVE SPARE 1
22	59	3B	EXECUTIVE SPARE 2
24	60	3C	EXECUTIVE SPARE 3
26	61	3D	EXECUTIVE SPARE 4
28	62	3E	VTRIM 12
30	63	3F	VTRIM 34

MINOR FRAME	DSC	HEX	QUANTITY
0	64	40	MAJOR FRAME COUNT (MOD 8=4)
2	65	41	BIAS 1
4	66	42	BIAS 2
6	67	43	BIAS 3
8	68	44	BIAS 4
10	69	45	STUB 1
12	70	46	STUB 2
14	71	47	GUARD 1
16	72	48	GUARD 2
18	73	49	FILTER 1
20	74	4A	FILTER 2
22	75	4B	FILTER 3
24	76	4C	FILTER 4
26	77	4D	FILTER 5
28	78	4E	FILTER 6
30	79	4F	FILTER 7
0	80	50	MAJOR FRAME COUNT (MOD 8=5)
2	81	51	FILTER MUX
4	82	52	RELAY STATUS 0
6	83	53	RELAY STATUS 1
8	84	54	RELAY STATUS 2
10	85	55	SPHERE SWEEP OPTIONS
12	86	56	SPHERE SWEEP ANGLE
14	87	57	SPHERE SWEEP ALGORITHM
16	88	58	SPHERE SWEEP ALTERNATE BIAS
18	89	59	SPHERE SWEEP RESULT
20	90	5A	SPHERE SWEEP M AVERAGE
22	91	5B	CYLINDER SWEEP OPTIONS
24	92	5C	CYLINDER SWEEP ANGLE
26	93	5D	CYLINDER SWEEP ALGORITHM
28	94	5E	CYLINDER SWEEP ALTERNATE BIAS
30	95	5F	CYLINDER SWEEP RESULT
0	96	60	MAJOR FRAME COUNT (MOD 8=6)
2	97	61	CYLINDER SWEEP M AVERAGE
4	98	62	SAWTOOTH OFFSET
6	99	63	SAWTOOTH DELTA
8	100	64	SAWTOOTH PERIOD
10	101	65	SAWTOOTH DIVIDER
12	102	66	BURST MODE/FREQ STATUS
14	103	67	BURST TRIGGER MODE
16	104	68	BURST DELAY TIME 0
18	105	69	BURST DELAT TIME 1
20	106	6A	DEPLOYMENT STATUS
22	107	6B	BOOM MICROSWITCHES
24	108	6C	TURNS COUNT BOOM 1
26	109	6D	TURNS COUNT BOOM 2
28	110	6E	TURNS COUNT LIMIT
30	111	6F	PLASMA MODE REGISTER
0	112	70	MAJOR FRAME COUNT (MOD 8=7)
2	113	71	32.7 SEC CLOCK
4	114	72	2.25 HOUR CLOCK
6	115	73	24 DAY CLOCK
8	116	74	GOOD COMMAND COUNT
10	117	75	BAD COMMAND COUNT
12	118	76	SUN ANGLE *
14	119	77	
16	120	78	SUN PERIOD *
18	121	79	
20	122	7A	BOOM LENGTH 1
22	123	7B	BOOM LENGTH 2
24	124	7C	TEMPERATURE 1
26	125	7D	TEMPERATURE 2
28	126	7E	TEMPERATURE 3
30	127	7F	TEMPERATURE 4

NOTE: (*) INDICATES A 2-BYTE VALUE WHICH IS GIVEN LOW BYTE FIRST.

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY

PROJECT: AFGL-701-14
TITLE : LANGMUIR PROBE INSTRUMENT
SECTION: TELEMETRY FORMAT SPEC.
DIGITAL SUBCOMMUTATOR FORMAT

DESIGNER: PETER HARVEY
DATE : 24 JUN 1983 REV3: 8 APR 86
REV1 : 31 JAN 1985
REV2 : 24 APR 1985

Figure 4. Digital SubCommutator Telemetry Format

Theory of Operation

The background manager handles hardware interrupts from instrument commands, major frames, word timing and the watchdog circuit as described below:

WORD RATE INTERRUPTS. Every other word clock from the spacecraft provides a Word Rate interrupt to the background manager. This is used to update the instrument word count within the minor frame. If it is an even interrupt, one of 64 routines will be chosen depending upon the value of the count. If armed, the background manager vectors to the address contained in BKGVECT on all odd interrupts.

The BKG module has been implemented in such a way as to keep the telemetry formatting modules ignorant of the specific telemetry format. Sampling calls and calls for data bytes are done separately so that each sub-module, like ELE.A or MAG.A, doesn't know where in the minor frame the data is going. ELE.A and MAG.A only know how much data to produce per sample call. In this way, changes to the telemetry format (for the next instrument) should be simple.

The Digital SubCommutator (DSC) output is formatted in the BKG module by the "DSC" routine. Using the list DSCTAB, this routine calls in turn each of the many modules which report status in the DSC data. Each module is handed an index in [A] and returns a byte in [A] which goes into the telemetry. Each module is therefore ignorant of the format of the DSC other than that of its own data. No module knows where in the DSC its data appears.

Along with telemetry formatting and data sampling calls, the word interrupts provide the basic timing for the sun angle phase locked loop, command execution, sawtooth generation, calls to the low energy plasma calculator, sampling of boom lengths and temperature monitors, and Kelley gain circuit pulsing.

MAJOR FRAME INTERRUPTS. Every four seconds, the major frame interrupt occurs. These interrupts used to synchronize the telemetry word count with the spacecraft as well as to internally synchronize the data sampling packages such as ELE and MAG. The 40-bit instrument clock is updated during the major frame interrupts.

COMMAND INTERRUPTS. Command interrupts occur as soon as the spacecraft command begins to shift into the instrument. The software merely notes that a new command will be arriving soon. The actual command processor is a routine which runs under the word rate clock interrupt.

WATCH DOG INTERRUPT. The "watchdog" is a simple circuit which counts major frames and is reset by a bit on one of the output ports. If two major frames occur while the watchdog has not been reset, the watchdog fires a TRAP type interrupt. Normal processing of the telemetry includes periodic pulsing to the reset line on the watchdog. A loss of this reset pulse means that the software has crashed for some reason, so the background manager simply resets the processor when this occurs.

There are several entry points for the background manager, most of which perform some utility function of the background. These as described below:

CMDGO. This entry point, called by software restart 6, executes a command in the [HL] registers as if the command just came from the ground. This is used for internal module to module controls.

BKGFN(1). This call (software restart 4 with [A] = 1) is a batch command processor which executes a list of commands terminated with a 0FFFFH command. On entry, [DE] address a list of commands.

BKGFN(2). Function 2 is a request of the command count status. If the command count matches what it is supposed to match then a zero is returned in the [A] register. If the command count is incorrect, [A] is returned with 1.

BKGFN(3). Restart 4 with [A]=3 is a call to the STVECT routine. If [HL] contain a non-zero address, the background vector BKGVECT is set to [HL]. If [HL] is zero, BKGVECT is reset and disarmed. As described above, the BKGVECT routine is called every odd word interrupt (which is every 2 milliseconds).

BKGFN(4). The fourth function on restart 4 is the STEXP function. On entry, [HL] points to a data block of [DE] bytes. The STEXP routine starts playing bytes from this block into the EXP telemetry slot. (EXP is the undefined, EXPerimental output which resulted from a decrease in the magnetometer telemetry allocation.)

INITIALIZATION. The BKGINIT entry enables the interrupts and initializes variables internal to the module so that it works correctly. Modules which are subordinate to the background module are initialized so they too are guaranteed to work when called upon.

The BKG module waits for the first major frame spike to enable word interrupts so that the telemetry comes "up" in synchronization. Depending upon when the instrument is turned ON, this may take up to one major frame time to start processing (4.096 seconds). During this time, commands will not be recognized.

COMMAND EXECUTION. Commands are executed by vectoring through a 32 element table called CMDTAB using the upper 5 bits of the command. Each command routine is executed with [HL] holding the command bits as well as [A] redundantly holding the low 8 bits. Command routines return carry if the command was BAD and no-carry if the command was GOOD. The BKG module keeps 8-bit counters GOODCNT and BADCNT of the commands which were so noted. If the command table has no vector there (the address is 0), the BADCNT is incremented.

BACKGROUND COMMANDS. The module implements two types of commands, namely, the "digit" and the "command count" commands. Digit commands emulate a hexadecimal shift register having the digits 0 thru F and an ENTER key. Digit commands shift a hex register left one digit and add the new digit. The ENTER command sends the contents of the register into the instrument as a new command. This is designed to be used with an old command capability of the Air Force facilities which could not readily send arbitrary bit patterns to our instrument. Digit commands would be used to create arbitrary bit patterns inside the instrument.

Command-count commands are used to check for lost commands within a long command series from the ground. The satellite control facilities cannot check each command if they are grouped in series. The "CMDS" command enters a count of the commands which will be arriving in a block. This data is put into a register called CMDCNT while GOODCNT and BADCNT are zeroed. When the block of the expected length is finally entered, the Fast Digital Monitor will report the result of BKGFN(2), which compares the expected count to the good count. If a command is missed (satellite uplink problems), the good count will be too small and the Fast Digital status will report the error.

Some of the BKG variables are described below:

ON-BOARD CLOCK. The BKG module maintains a 40 bit clock which counts 0.5 milliseconds (2 KHz). This clock is defined as follows:

```
 "DAY24"  "HR225"  "CYCLE"  "FRAME"  "WORD"
.--------.--------.--------.--------.--------.
!dddddddd!hhhhhhhh!cccccccc!jjjmmmmm!wwwwwwww!
'--------'--------'--------'--------'--------'
```

where w is the telemetry word within the minor frame,

m is the minor frame within the major,

j is the major frame count (internal),

c is the count of digital subcom cycles since reset,

h is the count of 2.25 hours since reset,

d is the count of 24 days since reset.

The clock is zeroed when the instrument is turned on and overflows after approximately 17 years.

SPIN PERIOD. The module keeps a 16-bit value of the spin period by counting the number of 16 millisecond pulses between sun pulses. This yields a value which can describe spin periods of up to 1024 seconds.

SUN-ANGLE. The BKG module keeps a 16-bit value of the angle between the sun sensor and the sun. The low eight bits subdivide a spin in 256 equal parts of 1.41 degrees each. The upper byte can be used as an overflow byte which shows that the sun sensor is not working (this occurs in shadow). The algorithm uses the spin period value calculated above divided by 256 (with remaindering) as a counter to decide when to step the sun angle. Below 4.096 second spin periods, the division underflows and causes the sun angle to stay at zero.

For a 30 second spin, the sun angle is accurate to 1 part in 2000 or .05% . During shadow periods, however, this error will accumulate since the sun pulse will not reset the angle. One hour of shadow (120 spins) times an accumulated .05% error is about .06 x 360 or 21.6 degrees (worst case). A typical error value would be half of that, or 10.8 degrees.

2.3 Electric Field Management

The ELE package manages the Electric Field/Langmuir Probe part of instrument operations. The functions include sampling, telemetry formatting and commanding the related sections of the instrument. For this module there are the following entry points:

ELEINIT : This entry requires no parameters and initializes the module for subsequent calls. The only thing this entails is zeroing the electric field RAM area.

ELEFRAME: This entry tells the module the minor frame time. On entry, [A] contains the 8-bit frame count. The module uses this information to decide whether to output SFR or FDM data, to synchronize the HX and LX list pointers, etc.

ELESAMP : The sample entry tells the module when to take one of its analog samples. The ELE package requires 16 sample calls per frame call. In addition, both the even calls and the odd calls must be periodic; i.e. even-to-even and odd-to-odd is a constant length of time.

ELETELEM: The telemetry call requests the ELE package to give some of its formatted data. On entry, the [A] register contains 0 if 1 byte is requested, 1 if 2 bytes are requested. On exit, the [L] register contains the first byte and [H] contains the optional second byte of data. The ELE package produces 20 bytes of data per 16 samples.

ELEDSC : Slow status of the package can be obtained by calling the DSC entry point. On entry, [A] is an index into the lists and variables which the ELE package maintains. ELEDSC returns its data in the [A] register.

When [A] is 0 to 15, the HX list is returned.

When [A] is 16 to 47, the LX list is returned.

When [A] is 48 or more, the ELE variables are returned. Note that the HX and LX lists for E-field and Langmuir modes are output in alternating fashion. The LP lists are output on odd major cycles.

ELEXMIT : Accesses the MAIN playback feature of the ELE package. On entry : [HL]-> data block of 13-bit data, [DE] is the number of samples to play back. On the first even minor frame, the playback will begin, overriding Burst playbacks if need be.

ELESTAT : Returns the Fast Digital Monitor in [A].

Theory of Operation

TELEMETRY SAMPLING/FORMATTING. The telemetry formatting uses a double-buffered scheme with each buffer representing one minor frame worth of data. Each minor frame toggles which buffer is to be used as the input buffer and which to be used as the output buffer.

The two types of samples, namely High Rate (HX) and Low Rate (LX), are dumped into the input buffer using two pointers called

HBPTR and LBPTR. Data is read out of the buffer using a single pointer called TMPTR. This results in a one minor frame delay in the data stream as samples which are taken in even frames are played in odd frames and vice versa.

```
                      SAMPLES        TELEMETRY
                    .---------.     .---------.
HBPTR-------->!         !     !         !<-------- TMPTR
                    !         !     !         !
                    !         !     !         !
                    !         !     !         !
                    !         !     !         !
                    !---------!     !---------!
                    !   FDM   !     !   SFR   !
                    !---------!     !---------!
LBPTR-------->!         !     !         !
                    !         !     !         !
                    '---------'     '---------'
------------------------------------------------------------------
```

Figure 5. Electric Field Telemetry Buffering

Corresponding with each high or low rate sample is a quantity descriptor which details what value to sample at that time. The quantity descriptors are in lists which are read using HQPTR and LQPTR as the samples are taken. These pointers are reset to the top of their lists as required and incremented along with each use.

```
                 Hx List       Lx List
               .---------.    .---------.
     HQPTR---->!         !  0 !         !<----LQPTR
               !         !  1 !         !
               !         !  . !         !
               !         !  . !         !
               !         !    !         !
               !         !    !         !
               !         !    !         !
               !         ! 15 !         !
               '---------' 16 !         !
                              !         !
                              !         !
                            . !         !
                            . !         !
                           30 !         !
                           31 !         !
                              '---------'
```

Figure 6. Electric Field Quantity Lists

Each channel descriptor has the following format:

```
.---------.
!brmmmmmmm!
'---------'
```

where b = 1 enables Main or Burst Playbacks to replace this channel when needed.

r = 1 indicates that RAM quantity #mmmmmm is to be sampled (see explanation of RAM quantities below).

r = 0 indicates that MAIN Multiplexor quantity #mmmmmm is to be sampled.

RAM quantities are simply values taken from the memory of the MAIN computer system. These are of interest mainly for diagnostic purposes when it is important to have a high bandwidth of information regarding some variables in the computer memory. Since there is a lot of RAM and only 6-bits of possible indexation, the RAM quantities use a programmable 16-bit base address called RAMBASE. The 6-bits are added to RAMBASE to produce an effective address from which 13-bits are retrieved (low byte first then 5-bits of the high byte).

```
                  MAIN MEMORY
                 .----------.
      RAMBASE-->!          ! 0
                !          ! 1
                !          ! 2
                !          ! .
                !          ! .
                !----------!
                !          ! m
                !--+-------!
                !  !       ! m+1
                !  '-------!
                !          !
                !          ! .
                !          ! .
                !          ! 62
                !          ! 63
                '----------'
```

Figure 7. Ram Quantities

When the Burst playback is indicated by the Fast Digital Monitor, sampling of channels with their playback bit "b" enabled is replaced with a BURPLAY call to the BUR package. Depending upon the state of the playback, the BUR module may offer some of its Burst Header or may request data from the Burst computer. In either case, thirteen bits are stored in the sample buffer as described above.

When a Main playback is indicated, the ELEXMIT facility is invoked. Samples are played from the buffer pointed at by the XMTPTR variable until the count held in XMTCNT goes to zero. When XMTCNT is zero, zero fill data is returned as a sample and the next FDM calculation clears the MAIN transmit state.

HQPTR/LQPTR. The quantity list pointers are reset when necessary to the value contained in VHXPTR and VLXPTR when in the E-field mode or to IHXPTR and ILXPTR when in the Langmuir Probe mode. Thus, if the instrument switches modes, the sampling list is changed along with it.

As all of these pointers are 16-bit values, any buffer in memory can be used for a telemetry format. Six buffers of 16 bytes each have been allocated for this purpose in the ELE module RAM area. This is enough for complete redefinitions of both the V and the I mode telemetry formats. Originally, the package was defined with the idea of having multiple ROM loaded formats from which to choose. This would facilitate various tests, sampling modes, etc. Ten format buffers of 16 bytes each were planned for the ROM, but only 3 (0,1,and 2) could be accomodated. The "format" command was devised as an address independent method to access these buffers. See the "format" command in the general instrument description.

FAST DIGITAL MONITOR CALCULATION. Once every other minor frame the Fast Digital Monitor is calculated and stored into the samples buffer. The FDM is a conglomeration of a number of status bits which indicate the instrument mode (E-field or Langmuir Probe), playback mode (Main, Burst or Real-time), etc. The format of the FDM is given in Figure ELE-1. These status bits are collected by calling the various modules responsible for the bits; i.e. the BUR, SWP, and BKG modules.

COMMANDS. Commands are vectored through the command vector table to routines contained in the ELE package. Most of these commands are simple calls to the IO package to perform some function such as setting the bias voltage. The software in the ELE package simply makes up the differences between the external command specifications and the internal IO call specifications. Where no differences occur, the ELE package directly references the IO call.

The VMODE and IMODE commands are interesting since they are actually command macros. When a VMODE command is executed, a list of commands are sent into the command-list processing facility of the BKG module. Since command lists are executed only when real commands are not ready, the execution of a VMODE command may occur after the series of real commands which follow the VMODE. One should probably not use relay commands in series right after either a VMODE or IMODE command (i.e. within 25 milliseconds). This could cause contention for the relay hardware.

2.4 Magnetic Field Management

The MAG package manages the Fluxgate Magnetometer part of instrument operations. The functions include sampling, telemetry formatting and commanding the related sections of the instrument. For this module there are the following entry points:

MAGINIT : This entry requires no parameters and initializes the module for subsequent calls. It also sets the mode of the BY amplifier to 0 (BY*1 operation).

MAGFRAME: This entry tells the module the minor frame time. On entry, [A] contains the frame count. On minor frames 2, 6, 10,... the sampling pointers are resychronized while on frames 0, 4, 8,... the output pointers are synchronized.

MAGGAIN : This entry tells the package to determine the gains for the three axes. Each axis is sampled in low gain and compared to 1/51th of full scale. If less than that, high gain will be selected for the SAMP call. If more, the low gain sample will be selected.

MAGSAMP : The sample entry tells the module when to take the three axis measurement using the gains determined in the GAIN call.

MAGENCD : This entry tells the module to buffer the data sampled by the last SAMP call.

MAGTELEM: The telemetry call requests the MAG package to give some of its formatted data. On entry, the [A] register contains 0 if 1 byte is requested, 1 if 2 bytes are requested. On exit, the [L] register contains the first byte and [H] contains the optional second byte of data.

The GAIN/SAMP/ENCD calls are actually three parts of 1 process. They are split in order to meet the system requirement to stay under 1 millisecond for background processes. The package requires two SAMP calls per FRAME call and produces 11 bytes of data per FRAME. (See Figures 1 through 3).

Theory of Operation

The magnetometer format involves buffering 8 triplets of autogain (BX,BY,BZ), eight triplets of gain bits, as well as 1 triplet of (BX,BY,BZ) taken in low gain. Four bits of mode information fill out the format to 44 bytes.

The package uses a single buffer scheme in formatting its data. Three pointers are required as well as temporary storage for gain registers. One pointer, called OTPTR, is used to read out of the buffer. It is reset to the beginning of the buffer at every frame divisible by 4 (1/2 second) and is stepped every time a byte of magnetometer data is required.

Two pointers, LGPTR and AGPTR, are used to store data into the buffer. Both pointers are reset at every frame mod 4 equal to 2. This causes the input buffering to be filling a different half of the buffer that is being read from by the OTPTR. The LGPTR is used to index low gain storage while the AGPTR indexes the auto gain storage. Both pointers actually count the number of nibbles from the beginning of the buffer since this facilitates storing 12 bit quantities.

The number of samples in the autogain buffer is counted by SMPCNT, which is reset when the storage pointers are. On the first sample the low gain values (which are taken for the auto-gain decision) are stored using LGPTR. Otherwise, all sample triplets are stored using AGPTR.

Since the sampling of autogain quantities occurs simultaneous to the output of previous gain decisions, the gain bits for a given buffer are stored in temporary locations TMPX thru TMPZ. On the last sample of the buffer (all quantities finished) these values are copied into the output buffer.

Mode information (4-bits) is buffered at the end of the first sample, following on the heals of the low gain storage. Mode bits are set by the BMODE command to the package.

MAG COMMANDS. The single command which the MAG module knows about is the BMODE command. The least significant bit of this command determines whether or not the BY amplifier is engaged on the Filter board. The software in the MAG module simply calls the IO module SETMUX facility is response to the BMODE command.

2.5 Plasma Instrument Data Management

The PLA package manages the Low Energy Plasma part of the instrument operations. The science requirements of this package are as follows:

1) Whenever the ABS(SCBY) is less than 1/8th ABS(SCBX), send the LEPA instrument a compressed calculation of SCBZ/SCBX plus current BURST mode information.

2) The software should produce results at least every 64 milli-seconds.

3) Use appropriate gain (high or low) in producing the calculations.

Note that SCBX, SCBY, and SCBZ listed above refer to the mag field in the spacecraft coordinate system, not the fluxgate outputs. The fluxgate sensor is turned such that its axis outputs correspond to spacecraft coordinates as follows:

SCBX = - FLUXGATE BX

SCBY = - FLUXGATE BZ - .044 FLUXGATE BY

SCBZ = - FLUXGATE BY

The LEPA aperture lies in the spacecraft XZ plane with Z along the spin axis.

The system requirements for the package are as follows:

1) The sample/calculation time cannot exceed 1 millisecond.

2) Since the spin-axis measurement has a mode of amplification, adjust it back to unity gain whenever the

amplifier is used.

Specifically, the format for the information transferred to the LEPA instrument is as follows:

```
16-bit LEPA shift register:        .--------.--------.
                                   !10qqffff!vsnnnnnn!
                                   '--------'--------'
```

where qq is a status code which indicates the condition of the BURST sampling.

qq= 00 = BURST OFF

01 = BURST ON, SEARCHING AT FREQUENCY F

10 = BURST ON, COLLECTING AT FREQUENCY F

f is a frequency code at which the burst is operating. (See General Description, "Burst Sample Frequency Control")

v is a range error bit if ABS(BZ/BX) >= 2.

s is a sign bit of BZ/BX (1=negative)

n is 32*ABS(BZ/BX)

For example, if BZ=BX, n= 100000; if BZ=1/2 BX, n=010000.

For this module there are the following entry points:

PLAINIT : This entry requires no parameters and initializes the module for subsequent calls.

PLASAMP : The sample entry tells the module when to take one of its analog samples.

PLADSC : Status of the package can be obtained by calling the DSC entry point. The PLA package has 8 bits of status returned in the [A] register as "esqqffff" where

e=0 indicates the package is enabled

s=1 indicates the package is currently sending.

q and f are as described above.

PLA PACKAGE COMMANDS. The PLA package has a single command, called LMODE, whose routine "PLACMD" is referenced in the command vector table CMDTAB. The format of that command is as follows:

.LMODE d where d= "eeqqffff"

will set the q and f fields of the mode register. Field "ee" enables/disables the entire package in case a ram-loaded algorithm is being used.

If ee = 10 the package is enabled.

= 11 the package is disabled.

= 0x (no change).

The operation of the "ee" field seems a little strange. Why go to the trouble of an arming bit for another bit? Well, the problem is simply that the BUR module communicates its status via the LMODE command. It of course uses the "no change" form of the command in case the PLA module has been disarmed earlier.

Theory of Operation

With the exception of PLASAMP, the PLA entry points are already described. PLASAMP is a routine which performs the following (note that BX, BY and BZ are fluxgate measurements and SCBX, SCBY and SCBZ are spacecraft coordinates):

1) For each axis, the PLA package uses the MAG package samples to determine the field values to 16-bit accuracy (2nT per bit).

   a) If low gain sample, it adds a low gain offset. If a high gain sample it adds a high gain offset.

   b) If a low gain sample, it then multiplies by 51 (18-bit result).

   c) For either gain, it divides by 4 (16-bit result).

2) For the BY measurement, if the BY amplifier is ON, the BY field value is scaled by 39/256 (1/6.2) and inverted.

3) Rotation to spacecraft coordinates is approximated by

   a) ABS(SCBX) = BX

   b) ABS(SCBY) = BZ + (11/256)*BY

   c) ABS(SCBZ) = BY

4) Compares ABS(SCBX) with 8*ABS(SCBY) and performs the SCBZ/SCBX calculation if the former is greater.

   4a) Calculates 32*ABS(SCBZ/SCBX) by performing an 8-bit divide of ABS(SCBZ) / 2*ABS(SCBX). The 8-bit result is then rounded and shifted to produce a 7-bit result which exceeds 64 if ABS(BZ) > 2*ABS(BX).

4b) Valid 6-bit results and masked 7-bit results are put into the 6-bit mantissa field along with an appropriate overflow bit (0 or 1).

4c) The sign of SCBZ/SCBX is calculated using an exclusive-or of the signs of their field values.

2.6 Burst Processing Management

The BUR package coordinates the Burst Sampling and Playback part of the instrument. The functions include sampling of conditions to determine when to take a burst of data, communicating with the burst computer system to receive stored data, and accepting commands for both the module and the BURST computer.

The science requirements for this function are as follows:

1) Sampling of conditions must operate at a fixed rate with a period no greater than 32 milliseconds, and must continue even while playing back data to the telemetry system.

2) Ram algorithms must be loadable from the ground and there should be a selection of ROM algorithms to choose from. One algorithm must be a simple clock which will be used for the timely Bursting during the chemical releases.

The system requirements for the module are:

1) The module must be provided some capability for starting and stopping burst playback transmissions.

The module has the following entry points:

BURINIT : This entry requires no parameters and initializes the module for subsequent calls. Playback requests are cleared and the default burst duration set to 4 seconds.

BURSAMP : The sample entry tells the module when to take one of its analog samples with which it will determine whether or not to take a burst collection, start the playback, etc.

BURPLAY : The telemetry call requests the BUR package to give some of its stored data. On exit, [HL] contain a 13-bit value for the telemetry stream if playback has been requested. Otherwise, [HL] will contain zero.

BURDSC : Status of the package can be obtained by calling the DSC entry point. On entry, [A] is an index into the variables which the package maintains. Data is returned in the [A] register.

Of special interest is the most significant nibble of DSC(0) which contains the internal mode of the package. These 4 bits are intended to be used in the fast digital monitor as an indicator of burst conditions.

Theory of Operation

The burst module operates completely in the background (under interrupts) trying to decide when to take bursts of data. As the controller of the Burst computer, all commands to the Burst computer go through the BUR module in order to keep the module aware of what is going on with the Burst.

The module has a state variable called MODFREQ (mode and frequency) which is used to remember from one SAMP call to another what the state of the Burst is. Three of the bits of

MODFREQ end up in the Fast Digital Monitor since they describe the internal mode of the BUR module. Eight states are defined for the module as follows:

0	OFF
1	SEARCH
2	COLLECT
3	WAIT
4	R1--SENDING "BGO" COMMAND TO BURST
5	R2--RECEIVING "REAL FREQUENCY" DATA
6	R3--RECEIVING "DURATION" DATA
7	R0--DELAYING BEFORE ENTERING R1 STATE

When any algorithm is selected using the control command (see below), the state of the module is set to R1. The module goes thru states R1, R2, R3 and ends up in SEARCH.

In SEARCH mode, each SAMP call simply vectors to the selected algorithm to determine whether searching is over or not. When the decision is made to end the SEARCH mode, the algorithm calls the "TRIGGER" routine which copies the burst duration parameter into the delay timer DTIME, saves the "EVENT TIME" and sets the state to COLLECT.

In COLLECT mode, each SAMP call simply decrements the delay timer. When zero, the collection phase is over, the "BSTOP" command is issued to the Burst computer and the "END TIME" is recorded.

Finally, the state is set to "WAIT" for the Burst computer to close its Burst memory file and get ready to play back. When this occurs, the state is set to "OFF", the Burst is commanded to play its data back and the playback bit is turned on in the MODFREQ status byte.

While the playback bit is set, the PLAY entry point will be called each time there is a channel armed for playback. The first thing played back into the telemetry is the header which shows the algorithm information and so forth. After its been played out, the BUR module requests data from the Burst computer system. This is the data portion of the Burst playback. See Figure 8.

When the Burst no longer has any data to play back, the BURPLAY routine decides whether to go into the OFF state or back into the R1 state. This depends upon the AUTOSEARCH bit in the trigger mode word (set by the BTRIG command). If AUTOSEARCH is armed, then the BUR module uses the R0 state to delay 1/2 second before going into R1. This makes sure that the playback bit in the Fast Digital Monitor returns to zero in between burst playbacks---a fact required by the telemetry decoding in order to distinguish between burst playbacks.

COMMANDS. BUR commands fall into three categories: (1) Burst CPU commands, (2) BUR algorithm commands and (3) BUR control commands. The first category of commands is simply those commands which the Burst CPU knows how to perform, such as BFMT and BANKS, etc. To each Burst computer command, the BUR module has a response in terms of its own state. That is to say, if the user commands the Burst computer to play back some data, the BUR module goes into the playback state as well.

There are four BUR module control parameters: (0) trigger mode, (1) timer delay Low, (2) timer delay High and (3) spare. Setting the trigger mode turns ON and OFF the BUR module, etc. See the general information for more detail on what the commands do.

The algorithm commands simply set the four variables which are available for use by the algorithms. These are mostly undefined and are intended for use by the RAM algorithms.

Whenever any of the control or algorithm parameters are going to be changed, the burst trigger is first set to "OFF". This keeps algorithms from being executed using partial parameters and so forth. The impact to the user is that the last command should be the setting of the burst trigger (BTRIG command).

ROM TRIGGERS. The triggers available in the flight ROM are as follows:

0	OFF	
1	IMMEDIATE	Immediately begins collecting.
2	VALCHECK	Begins collecting when magnitude of SAMPLE(PARAM0) exceeds PARAM1.
3	MAGCHECK	Begins collecting when the PLA module detects the loss cone (it is sending). The status of the PLA package is requested through its DSC function.
4-7	RAM	RAM loaded algorithm.

BURST HEADER INFORMATION

0	10110001	FORMAT CODE
0	ALG	TRIGGER ALGORITHM CODE
0	ST[0]	START TIME OF DATA (*)
0	ST[1]	
0	ST[2]	
0	ST[3]	
0	ST[4]	
0	VT[0]	EVENT TIME (*)
0	VT[1]	
0	VT[2]	
0	VT[3]	
0	VT[4]	
0	ET[0]	END TIME OF DATA (*)
0	ET[1]	
0	ET[2]	
0	ET[3]	
0	ET[4]	
0	PARAM 0	PARAMETERS FOR THE TRIGGERING ALGORITHM.
0	PARAM 1	
0	PARAM 2	
0	PARAM 3	
0	10110010	BURST CHECK CODE
0	10100001	BURST FORMAT CODE
0	BFREQ	BURST FREQUENCY CODE
START ADDR0		BURST START ADDRESS LOW 12 BITS
START ADDR1		BURST START ADDRESS HIGH 12 BITS
END ADDR0		BURST END ADDRESS LOW 12 BITS
END ADDR1		BURST END ADDRESS HIGH 12 BITS
0	#QTYS	# QUANTITIES PER RECORD OF DATA
0	BQTY[0]	LIST OF QUANTITY DESCRIPTORS
0	BQTY[1]	
0	BQTY[2]	
. . .	. . .	
0	BQTY[N]	

BURST DATA

SAMP(BQTY[0])	RECORD 0
SAMP(BQTY[1])	
SAMP(BQTY[2])	
. . . . .	
SAMP(BQTY[N])	
SAMP(BQTY[0])	RECORD 1
SAMP(BQTY[1])	
SAMP(BQTY[2])	
. . . . .	
SAMP(BQTY[N])	
SAMP(BQTY[0])	RECORD 2
SAMP(BQTY[1])	
SAMP(BQTY[2])	
. . .	
SAMP(BQTY[0])	LAST RECORD
SAMP(BQTY[1])	
SAMP(BQTY[2])	
. . . . .	
SAMP(BQTY[N])	
0	
0	ZERO FILL TRAILER
0	
0	

(*) : ALL CLOCK VALUES ARE BYTES ENCODED INTO THE 12-BIT CHANNEL.
THESE TIMES REFER TO THE 40-BIT INSTRUMENT CLOCK, NOT THE SPACECRAFT CLOCK.

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY

PROJECT: AFGL-701-14
TITLE: BURST PLAYBACK TM FORMAT
FILE : BPLAY.DWG

DESIGNER: PETER R HARVEY
DATE : 24 APR 1985
REV1 : 8 APR 1986

Figure 8. Burst Playback Telemetry Format

2.7 Spin-Fitting Management

The FIT package controls the spin-fitting of electric field data. The functions include sampling electric field data at particular sun angles, calculating the sine wave least squares fit, and telemetering this information. Unlike other modules, these operations are synchronized with the spin period of the spacecraft rather than with telemetry timing.

Specifications for the module are as follows:

1) 32 points taken at even angles with respect to the sun.
2) Full period capability only.
3) No bias changes should be made during the sampling.
4) Perform fits in voltage mode only
5) Sine wave fit the boom systems at 180 degrees out of phase with each other. This will give better temporal resolution (1/2 spin) of the electric field if the sphere and cylinder measurements are comparable. If they are not comparable, it doesn't hurt.
6) Each fit has the following results:
   - Sine and Cosine components (floating pt)
   - High and Low gain offsets to the fit (floating pt)
   - Standard deviation (floating pt)
   - A code to distinguish sphere and cylinder fits.
   - The number of points used in the fit.

For this module there are the following entry points:

FITINIT : This entry requires no parameters and initializes the module for subsequent calls by copying initial values from ROM into the parameter blocks for both the V12 and V34 sensor pairs. For both blocks, it NULLs all old AHI and ALO values so that the SPIN package starts off with no history of AHI or ALO to contend with.

FITSMP : The sample tells the package of a particular sun angle at which the module may decide to take a sample. On entry, [A] is the 8-bit sun angle. This function samples and stores the data from both boom systems when the angle is a 360/32 boundary.

FITTELEM: The telemetry call requests the FIT package to give some of its formatted data. On exit, the [A] register contains the byte of data. The FIT package produces an asynchronous format of data (zero filled) with up to 34 bytes of non-fill data every spin. See Figure 9.

FITEXEC : Performs the spin fits if ready to do so. Returns minus if its not ready. This entry point should be called from the foreground since it may take up to 1/2 second to complete the pair of fits.

```
SFR DATA
----------

 .---------.
 !1111000X!    FIT TYPE  (X = 0 FOR SPHERES, X = 1 FOR CYLINDERS)
 +---------+
 ! AHI[0] !    AHI OFFSET (FLOATING POINT)
 +---------+
 ! AHI[1] !
 +---------+
 ! AHI[2] !
 +---------+
 ! ALO[0] !    ALO OFFSET (FLOATING POINT)
 +---------+
 ! ALO[1] !
 +---------+
 ! ALO[2] !
 +---------+
 !   B[0] !    B COMPONENT (FLOATING POINT)
 +---------+
 !   B[1] !
 +---------+
 !   B[2] !
 +---------+
 !   C[0] !    C COMPONENT (FLOATING POINT)
 +---------+
 !   C[1] !
 +---------+
 !   C[2] !
 +---------+
 !SIGMA[0]!    STANDARD DEVIATION (FLOATING POINT)
 +---------+
 !SIGMA[1]!
 +---------+
 !SIGMA[2]!
 +---------+
 !   N    !    NUMBER OF POINTS REMAINING IN FIT
 '---------'


NOTE: THE SFR TELEMETRY DATA IS ASYNCHRONOUS WITH THE NORMAL TELEMETRY DATA.
      BLOCKS OF SPIN FIT DATA ARE SENT SYNCHRONOUS TO THE SPIN PERIOD. TWO
      BLOCKS SENT EACH ROTATION (ONE FOR SPHERES, ONE FOR CYLINDERS).

      THE SFR BLOCKS ARE SEPARATED FROM OTHER SFR BLOCKS BY ZERO FILL.
```

```
+----------------------------------------+
! UNIVERSITY OF CALIFORNIA, BERKELEY     !
! SPACE SCIENCES LABORATORY              !
+----------------------------------------+
! PROJECT: AFGL-701-14                   !
! TITLE:   SPIN FIT RESULT TELEMETRY     !
! FILE :   FIT.DWG                       !
+----------------------------------------+
! DESIGNER: PETER R HARVEY               !
! DATE :   8 APR 1986                    !
+----------------------------------------+
```

Figure 9. Spin-Fit Result Telemetry Format

BUFFERING. At each selected angle which is a multiple of 360/32 the FIT module samples the V12 and V34 measurements (unless disabled of course). These data are stored in two double buffers, one pair for each type of data, using pointers V12IN and V34IN, which are incremented with each use.

For each boom system there is an angle at which fit sampling is started and ended. When this angle is met for a given boom pair, the buffer pointer is checked to see if the previous buffer has been completed. This is a reasonable question since changes in the spin period may cause short cycling of the sun angle phase locked loop (i.e. skips from 280 degrees to 360 degrees for example). If the buffer has been filled with the correct number of points, a variable called V12OUT (or V34OUT) is pointed at it.

These pointers are checked by the foreground process FITEXEC. If they point at a buffer, then the SPIN module is called with the appropriate parameter and result pointers. When the spin fit is finished, the V12OUT or V34OUT pointer is set to a "READY" value to indicate that new spin fit results are ready to be transmitted.

TELEMETRY. Sending fit results into the telemetry stream is accomplished by the FITTELEM routine. It is called by the ELE module each time the SFR TM slot comes up (every other minor frame). The FITTELEM routine hands out bytes one by one from memory pointed at by PTR, using a count contained in COUNT. When COUNT is zero, the PTR is reloaded with any results marked

"READY" by their output pointer (V12OUT or V34OUT). Once the pointer PTR and COUNT are loaded, the "OUT" pointer is set to "DONE", so that the results will be transmitted only once.

FIT COMMAND. Options in the package operations are selectable using the single FITMODE command (see the general description). One option completely disables the FIT package so that a RAM loaded procedure can be substituted.

A second control bit determines whether V12 fits should be performed. This is used by the EXEC mode switcher when going into the Langmuir Probe Mode since it is nonsense to spin fit V12 at that time.

A third option is a diagnostic dump of the data points used in the spin fit. This feature uses the ELEXMIT routine to form a MAIN transmission of the 32 samples. Unfortunately, there is no header protocol to distinguish it from bias sweep results except for the lack of the sweep header.

INITIAL PARAMETERS. The parameter block for each boom system is copied from the ROM into the RAM at initialization so that adjustments may be made as desired. The initial values for both boom systems are given below:

GAINFACTOR	1/50.9
ALPHA	1.40
BETA	0.40

2.8 Spin-Fitting Computations

The SPIN.A module is devoted to only one function, namely producing sine wave least squares fits of some measured data. This function is by far the most complicated of those in the instrument (although bias sweep analysis is a close second). It is called solely by the FIT.A module which handles the sampling of the data for the two boom systems. All CRRES dependent stuff (if there is any) is contained in FIT.

The SPIN module takes 32 points of evenly sampled data which represents one spin period. It produces the sine, cosine and offset parameters which approximate the waveform. Also produced is the standard deviation for the points.

Assume E(ti) is the ti'th measured value of the electric field between the two sensors (either V12 or V34). The approximating waveform's expression in general would be

E(ti) = A + B*cos(wti) + C*sin(wti)

where w is 2*pi*v and v is the spacecraft spin frequency. However, since the instrument samples data in two gain states, the true approximating formulas are

E(ti) = Ahi + B*cos(wti) + C*sin(wti)

for high gain points and

E(ti) = Alo + B*cos(wti) + C*sin(wti)

for the low gain points.

What we want to do is form a difference function between the approximate expression and the sampled data. The best fit occurs when we minimize this function. Figure 10a thru 10e show the difference function and its associated differentials. Equations (b) thru (e) give four equations in four unknowns which can, of course, be solved by standard matrix methods. The matrix elements are shown in Figures 11a and 11b.

The procedure of SPIN is actually more complicated than simply solving the function once. Error points should be removed in the process of determining the best fit of the DC field. These "error" points are real data, of course, but reveal AC activity which we want to remove from a DC measurement. The procedure followed by the SPIN module is as follows:

1. Least squares fit the input data to find Ahi, Alo, B, C and the standard deviation Sigma.
2. Discard all points more than Alpha*Sigma from the least squares curve, where Alpha is an input constant.
3. Repeat the least squares fit using the remaining points. For the jth fit, throw away points more than (Alpha+Beta)*Sigma from the curve. (Beta is also an input constant.)
4. Stop the above procedure when no more points have been removed.
5. Average Ahi with the previous AVPTS values of Ahi to produce Ahi'. Do the same for Alo.
6. Use Ahi' and Alo' as input values (not to be determined by the fit), remove these offsets from the remaining data points and perform a least squares fit to obtain B and C only.

7. Proceed as above until no more points are removed.

8. Transmit Ahi', Alo', B, C, Sigma and N, the number of points remaining in the fit.

It is important to note:

1. The algorithm does not use Ahi or Alo when fewer than 3 points occur in that gain. To do so would cause the fit to errantly report the value of the offset. For example, if 1 point of high gain were in the buffer, the high gain offset (Ahi) would be set to that value. This is wrong, of course, since the value of the point is simply a small electric field value at the sampled time, not the offset of the high gain amplifier.

2. When rejecting points from the curve, the algorithm always removes their effects from the matrix sums by subtraction. It does not simply recalculate the sums for the remaining points. While being a simpler procedure, this would take at least fifteen times longer to perform.

3. Relatively large offset values (AHI and ALO) produce large sigmas and poor fits. It is easy to see why. In each summation of the points, ALO will be summed 32 times while each point only once. If ALO is large relative to the sum of the points, the floating point value will more or less reflect only the ALO value and lose track of the point sum.

The procedure is invoked with three parameters. On entry, [HL] address the sampled data block, [DE] points to the input parameter block, and [BC] point to where SPIN should put the results (see Figure 12). The SPIN calculation takes 500 milliseconds or less when running the cpu at 5 MHz (crystal).

EQ(a):

$$F = \sum_{i=1}^{M} [E(t_i) - (A_{HI} + B\cos(wt_i) + C\sin(wt_i))]^2$$

$$+ \sum_{i=M+1}^{N} [E(t_i) - (A_{LO} + B\cos(wt_i) + C\sin(wt_i))]^2$$

EQ(b):

$$\frac{\Delta F}{\Delta A_{HI}} = \sum_{i=1}^{M} 2[E(t_i) - (A_{HI} + B\cos(wt_i) + C\sin(wt_i))]$$

EQ(c):

$$\frac{\Delta F}{\Delta A_{LO}} = \sum_{i=M+1}^{N} 2[E(t_i) - (A_{LO} + B\cos(wt_i) + C\sin(wt_i))]$$

EQ(d):

$$\frac{\Delta F}{\Delta B} = \sum_{i=1}^{M} [E(t_i) - (A_{HI} + B\cos(wt_i) + C\sin(wt_i))]\cos(wt_i)$$

$$+ \sum_{i=M+1}^{N} [E(t_i) - (A_{LO} + B\cos(wt_i) + C\sin(wt_i))]\cos(wt_i)$$

EQ(e):

$$\frac{\Delta F}{\Delta C} = \sum_{i=1}^{M} [E(t_i) - (A_{HI} + B\cos(wt_i) + C\sin(wt_i))]\sin(wt_i)$$

$$+ \sum_{i=M+1}^{N} [E(t_i) - (A_{LO} + B\cos(wt_i) + C\sin(wt_i))]\sin(wt_i)$$

EQ(f):

$$\text{Sigma} = \sqrt{\frac{F}{N-1}}$$

Spin Fit Difference Function F of M high gain points and (N-M) low gain points.

UNIVERSITY OF CALIFORNIA, BERKELEY SPACE SCIENCES LABORATORY	
PROJECT	: AFGL-701-14
TITLE	: LANGMUIR PROBE INST.
SECTION	: SPIN FIT DIFFERENCE FUNCTION
FILE	: SPIN1.CAD
DESIGNER:	PETER R HARVEY
DATE	: 7 MAY 86

Figure 10. Spin Fit Difference Function

	A_{HI}	A_{LO}	B	C	
$\frac{\Delta F}{\Delta C}$	$\sum^{M} \cos$	$\sum_{M+1}^{N} \cos$	$\sum^{N} \cos^2$	$\sum^{N} \sin\cos$	$\sum^{N} E(t_i)\cos$
$\frac{\Delta F}{\Delta B}$	$\sum^{M} \sin$	$\sum_{M+1}^{N} \sin$	$\sum^{N} \sin\cos$	$\sum^{N} \sin^2$	$\sum^{N} E(t_i)\sin$
$\frac{\Delta F}{\Delta A_{LO}}$	0	N-M	$\sum_{M+1}^{N} \cos$	$\sum_{M+1}^{N} \sin$	$\sum_{M+1}^{N} E(t_i)$
$\frac{\Delta F}{\Delta A_{HI}}$	M	0	$\sum^{M} \cos$	$\sum^{M} \sin$	$\sum^{M} E(t_i)$

Spin Fitting Matrix including offset parameters

	B	C	
$\frac{\Delta F}{\Delta C}$	$\sum^{N} \cos^2$	$\sum^{N} \sin\cos$	$\sum^{N} E(t_i)\cos - A_{HI} \sum^{M} \cos - A_{LO} \sum_{M+1}^{N} \cos$
$\frac{\Delta F}{\Delta B}$	$\sum^{N} \sin\cos$	$\sum^{N} \sin^2$	$\sum^{N} E(t_i)\sin - A_{HI} \sum^{M} \sin - A_{LO} \sum_{M+1}^{N} \sin$

Spin Fitting Matrix with offsets removed

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY
PROJECT : AFGL-701-14
TITLE : LANGMUIR PROBE INST.
SECTION : SPIN FITTING MATRICES
FILE : SPIN2.CAD
DESIGNER: PETER R HARVEY
DATE : 7 MAY 86

Figure 11. Spin-Fitting Matrices

```
          .--------.--------.
[HL]->    !LLLLLLLL!...GHHHH!  0    The Sampled Data Block
          +--------+--------+
          !LLLLLLLL!...GHHHH!  1
          +--------+--------+
               .
               .
          +--------+--------+
          !LLLLLLLL!...GHHHH! 31
          '--------'--------'

                                          The Parameter Block
          .--------.--------.--------.
[DE]->    !SEEEEEEE!HHHHHHHH!LLLLLLLL!    1/GAIN FACTOR
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    ALPHA
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    BETA
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    AHI[0]
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    ALO[0]
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    AHI[1]
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    ALO[1]
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    AHI[2]
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    ALO[2]
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    AHI[3]
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    ALO[3]
          '--------'--------'--------'

                                          The Result Block
          .--------.--------.--------.
[BC]->    !SEEEEEEE!HHHHHHHH!LLLLLLLL!    Ahi
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    Alo
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    B
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    C
          +--------+--------+--------+
          !SEEEEEEE!HHHHHHHH!LLLLLLLL!    Sigma
          +--------+--------'--------'
          !NNNNNNNN!                       N (# pts left)
          '--------'
```

Figure 12. Input/Output Data Blocks for Spin Fitting

2.9 Sawtooth Generator

The SAW module is used to generate sawtooth waveforms on the sphere bias lines. This is nominally useful in the Langmuir Probe mode but for test purposes, can produce diagnostic waveforms on the bias DACs in the electric field mode as well. The module parameters are defined to generate either sawtooth waveforms or square waves. The module has the following entry points:

SAWINIT : The initialization entry requires no parameters and simply copies its default parameters from ROM into RAM where the user commands can change them. Its initial state is OFF (no waveform).

SAWSTEP : The step input does everything. It synchonizes itself word 224 of every minor frame 30. It does the calculation of the bias value and the setting of that value on both boom bias circuits.

SAWDSC : The Digital SubCom function returns the slow status of the package (what has been commanded). On entry, the index into this data is in the [A] register. On exit, the [A] register contains one byte of data.

GENERATION. The Sawtooth generation is a simple process complicated by the many things going on in the rest of the system. The process itself involves just the four values:

1.	SAWOFF	The bias offset DAC value
2.	SAWDEL	The size of each change to the bias
3.	SAWPER	The number of steps up and down
4.	SAWDIV	The divider of the input frequency

Each of these has a temporary value as follows:

1.	BIASREG	The present bias value
2.	DELREG	The delta for this series
3.	PERCNT	The period counter
4.	DIVCNT	The divider counter

The basic input calls to SAWSTEP are divided by the count in SAWDIV using the temporary variable DIVCNT. Each time DIVCNT is reloaded from the commanded value in SAWDIV, the present bias value is put on the DACs and a new step of the bias value is made.

Before each step the period counter is first decremented and zero checked. If zero, the DELREG is flipped in sign to make the waveform go in the opposite direction, and the period counter PERCNT is reloaded from SAWPER. If not zero, the procedure simply adds the DELREG to the BIASREG. When flipping directions, the BIASREG is not changed.

This causes the BIASREG to repeat the values at the top and the bottom of the waveform. This is a nice feature since the waveform has exactly the number of steps asked for rather than N+1 or N-1, etc. If you ask for 128 steps, you get 128 steps up and 128 steps down. Simple.

SYNCHRONIZATION. A very important part of the package is its synchronization code since, for data analysis purposes, one must know what values are on the BIAS DACs at all times. The synchronization accomplishes this by watching the internal word and frame clocks in the BKG module and waiting for minor frame 30 and word 224. This is just before the ELE package sampling begins for the next major frame. Thus, data within major frames always reflect multiples of the sawtooth starting at the beginning of the waveform.

The synchronization operation is performed by simply copying the four commanded values SAWOFF thru SAWDIV into the temporaries BIASREG thru DIVCNT. Commands changing these values are not guaranteed of affecting the waveform until the first sync time (every four seconds).

COMMANDS. The commands for the package simply set one of the six parameters SAWOFF thru SAWDIV, OPTION and SENSOR. The first four of these have already been described.

"OPTION" has two enable bits, one for stepping and the other for biasing. This flexibility allows the package to run the bias stepping algorithm without actually setting the bias DACs. This is required for the 2 seconds when the bias sweeps occur. We want the bias sawtooth to maintain its place in the waveform regardless of the fact that it shouldn't set the biases for a couple of seconds. The OPTION parameter is used internally by the EXEC mode switcher and by the SWP module. To completely disarm the package, one must set the SAWDEL to zero.

"SENSOR" simply holds the bias DAC pair to set. Of course, since only the spheres have a current mode, this value is set to 1. Bias DACs 1 and 2 get the sawtooth. Just for fun the user can set this value to 3, sending the sawtooth out to the cylinders.

2.10 Bias Sweeps

The SWP module manages the timing and analysis of Bias sweeps on the spheres and cylinders. Sweeps are used to calculate the proper sensor biasing while in the electric field mode and to calculate the temperature and density information in both E-field and Langmuir modes.

The requirements for the voltage mode sweeps are

1. Perform a current sweep in the voltage mode at a programmable time interval, initially 120 seconds.
2. Sweep both sensors (spheres or cylinders) when that boom system is perpendicular to the sun-spacecraft line. Do not sweep both spheres and cylinders at the same time. See Figure 13.
3. Sweep from -360 to +360 nanoamperes in 128 steps that take 500 milliseconds. At each step, measure and store both sensor outputs in LOW GAIN (V1 and V2 or V3 and V4). Note that V2 must be inverted to be consistent with the rest. See Figure 14A.
4. Allow maximum settling time between setting the current level and measuring the value.
5. Analyze both V1S and V2S (or V3S and V4S) curves as defined below.
6. Average the BIAS results from the two curves and set both BIAS DAC's to that value.
7. Transmit the sweep curves, the parameters and results of the analysis if enabled for playback.

Screening out noise from the curves is required and specified as follows:

1. Do not analyse regions of the curve where the measurement is within 10 percent of full scale, positive or negative.
2. Let V(i) be the ith measurement on a scale of -2048 to +2047. If V(i+1) - V(i) < -N then if V(i+2)-V(i+1) > V1(i)-V(i-1), replace V(i+1) by (V(i)+V(i+2))/2. Otherwise, replace V(i) by (V(i+1)+V(i-1)/2. See Figure 16.

Analysis of the curves follows the noise step:

1. Form a new function whose ith value is defined as shown in Figure 16B. Find the minimum value of this function, since the bias changes make the smallest effects on the measured field.
2. If there are two or more minima with the same delta-V, select the one having the smallest bias value (most negative bias current).
3. If the algorithm fails, set the bias current to an alternate value (IBALT). This can only occur if the signal is within 10 percent of full scale the entire time.

Requirements for the current mode voltage sweeps are

1. Perform current mode voltage sweeps at programmable time intervals, initially 120 seconds.
2. Wait until the sensors are perpendicular to the sun-spacecraft line.

3. Sweep both sensors (spheres 1 and 2 only) through 128 voltage steps from -35 to +35 volts. Measure RI1 and RI2 at these 128 points in LOW GAIN only. See Figure 14B.

4. Transmit the curves.

Note: Analysis of the voltage sweeps is left TBD by ground loaded analysis programs. Though the initial specifications for this analysis was finished prior to instrument completion, there was no longer enough memory available in the ROM to do the calculations required.

There are a number of system requirements placed on the operation of this module. Amoung them are:

1. Sweeps must inform the Fast Digital Monitor bit when they begin. As illustrated by Figure 14, the software finds the correct sun angle, and then waits until the Fast Digital Monitor is calculated. This keeps the sweep action synchronous with the telemetry while not causing too much error in sun angle.

2. The SWP module must coordinate with the sawtooth generator so that the SAW module doesn't continue to set the BIAS DAC's while the sweep is occurring.

3. The SWP module must not place BIAS results on the sensors until the beginning/ending of that sensor pair's spin fit. Otherwise, part of the first fit will be done at one bias value while part of the second will be done at the new bias

value.

The SWP module has the following entry points

SWPINIT This entry has no parameters and simply sets the initial values for the module.

SWPANG This entry is called to report a new sun angle. On entry: [A] is the new sun angle. This routine checks for sun angle related events such as the start of a sweep or the setting of the bias result.

SWPSTAT This entry is used in the Fast Digital Monitor calculation since the SWP module provides one bit of the FDM word. On exit: [A] = 1 for SWEEP-IN-PROGESS.

SWPEXEC This is a foreground entry point for performing sweeps. If it is the proper time, a sweep will be performed. If not, the code will simply return.

SWPDSC This routine returns the status of the module to the Digital SubCom processor. On entry: [A] is the index into the status block. On exit : [A] holds the status byte.

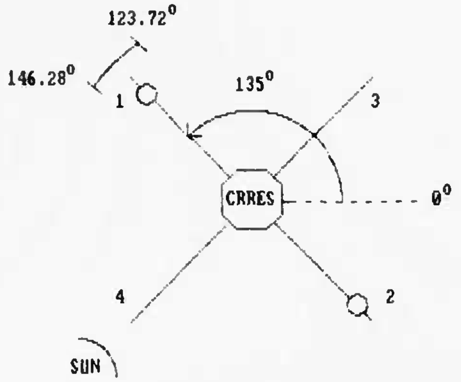


Fig A. V12 I and V Bias Sweeps are taken when the spheres are perpendicular to the sun. Since they take 1-2 seconds to complete, the begin 11.28 degrees prior to perpendicular.

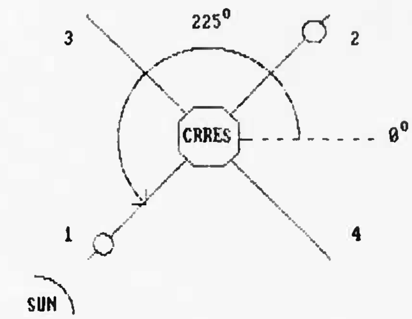


Fig B. Bias results are applied 1.41° prior to when sphere 1 points at the sun (223.59). Spin fits take their first data point at 225 degrees.

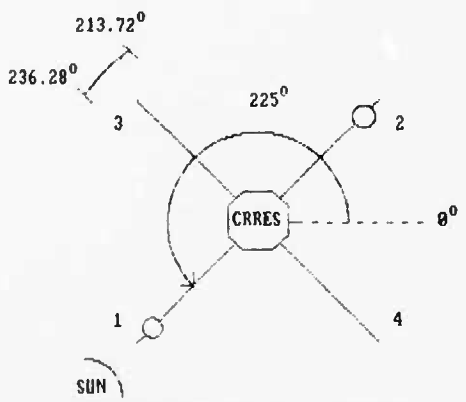


Fig C. Cylinder Bias Sweeps are taken when they are perpendicular to the sun.

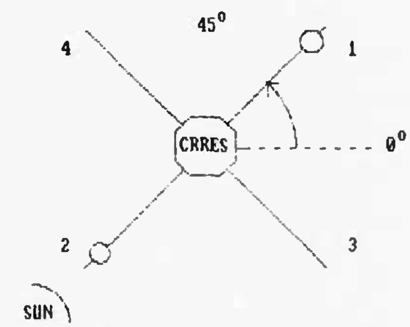


Fig D. Cylinder Bias results are output 1.41 degrees prior to the start of cylinder spin fits.

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY

PROJECT : AFGL-701-14
TITLE : LANGMUIR PROBE INSTR.
SECTION : ORIENTATION OF BOOMS DURING BIAS SWEEPS
FILE : SWEEP1.CAD

DESIGNER: PETER R HARVEY
DATE : 9 MAY 1986

Figure 13. Orientation of Booms During Sweeps

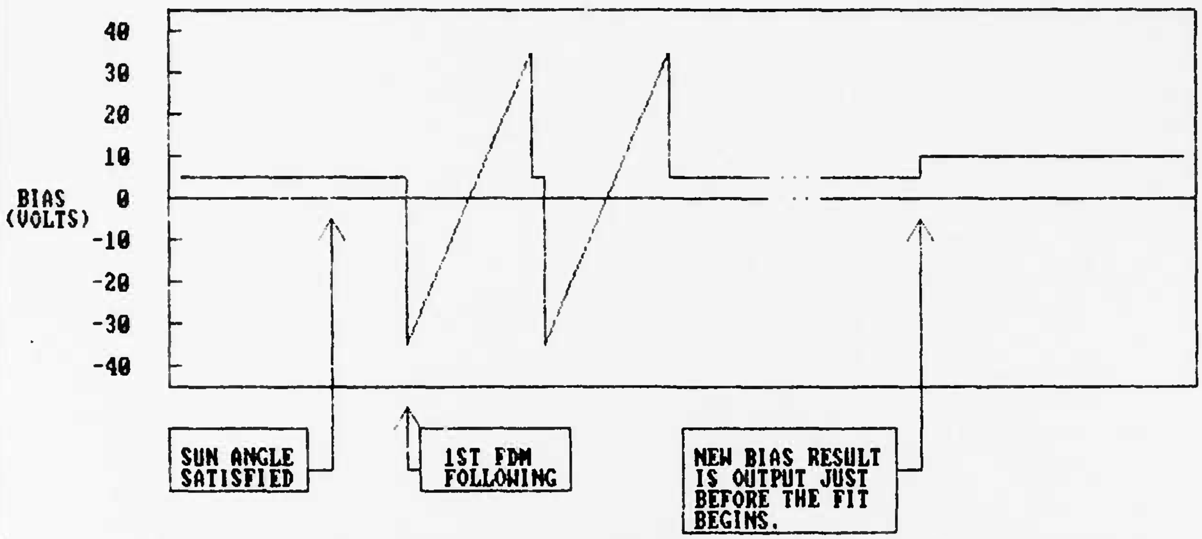


Fig A. Voltage mode bias output on the spheres when controlled by the sweep module.
Note: Delay between sun angle and 1st FDM varies between 0 and 256 msec.
Each of the two sweeps takes 512 msec for 128 points.

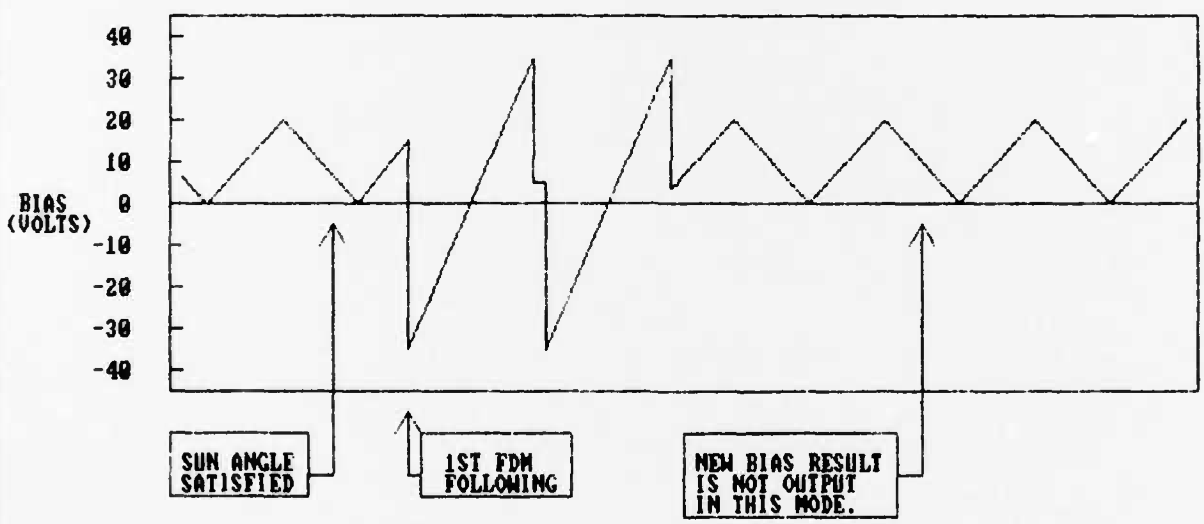


Fig B. Current mode bias output on the spheres when controlled by the sweep module.
Note: Delay between sun angle and 1st FDM varies between 0 and 256 msec.
Each of the two sweeps takes 512 msec for 128 points.

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY

PROJECT : AFGL-701-14
TITLE : LANGMUIR PROBE INSTR.
SECTION : BIAS OUTPUT TIMING

FILE : SWEEP2.CAD

DESIGNER: PETER R HARVEY
DATE : 9 MAY 1986

Figure 14. Bias Output Timing for Spheres

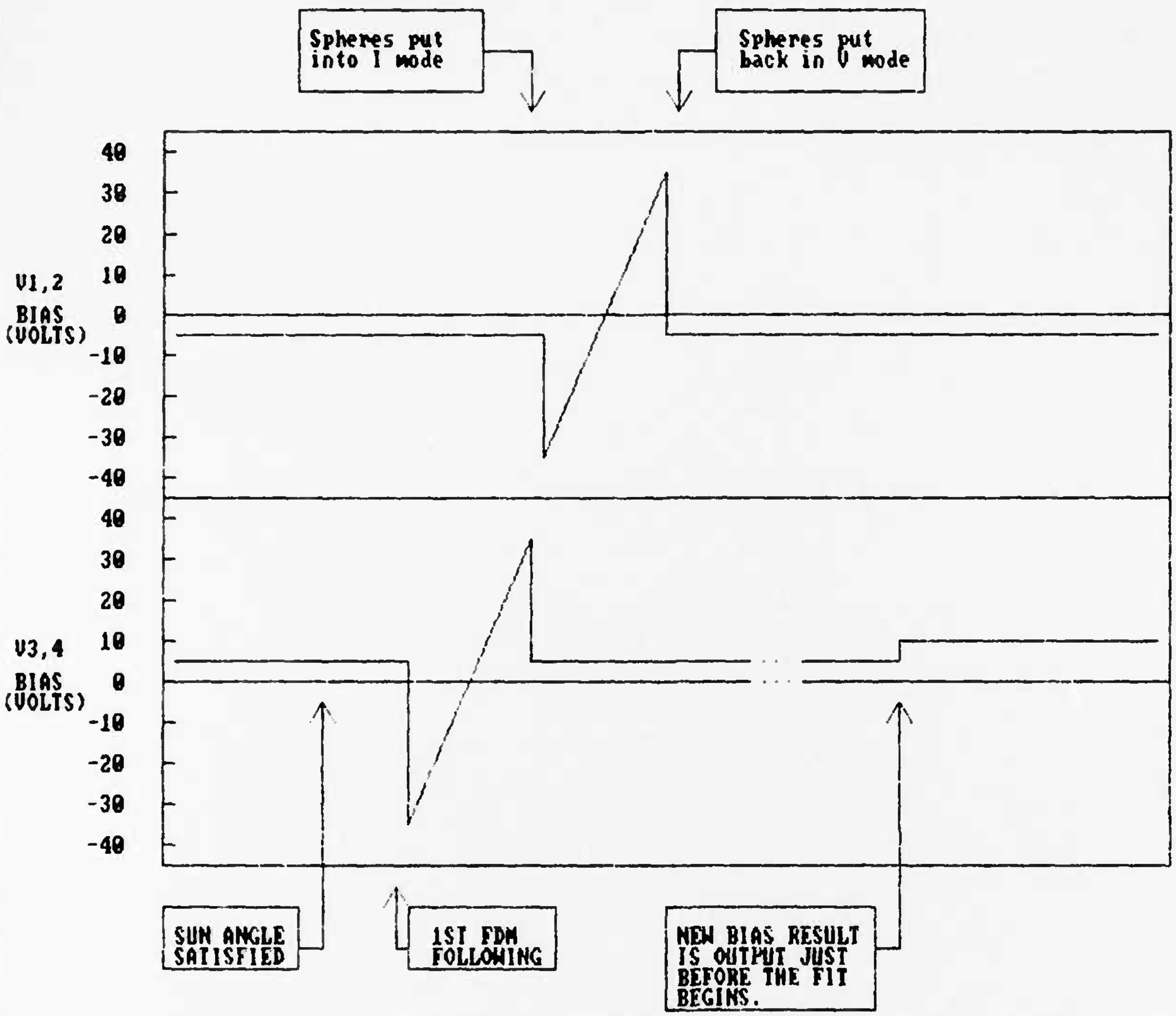


Fig A. Voltage mode bias output on the spheres and cylinders in a cylinder sweep.

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY

PROJECT : AFGL-701-14
TITLE : LANGMUIR PROBE INSTR.
SECTION : BIAS OUTPUT TIMING CYLINDER SWEEPS
FILE : SWEEP3.CAD

DESIGNER: PETER R HARVEY
DATE : 15 MAY 86

Figure 15. Bias Output Timing for Cylinders

$$\Delta V(i) = \sum_{j=i}^{M} V1S(i+j) - V1S(i-j) \qquad \text{for } (M+1) < i < (128-M)$$

Fig A. Sweep differential function with M point averaging.
For sensors V2S, V3S and V4S, substitute for V1S above.

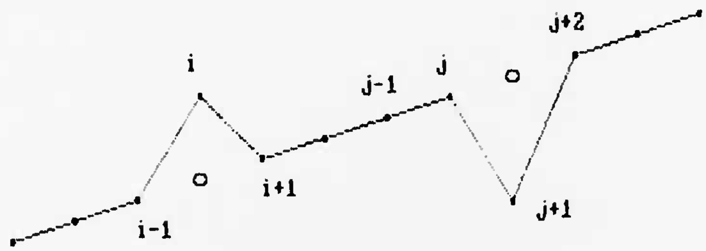


Fig B. Noise points on a bias sweep curve. Circles show replacement point.

UNIVERSITY OF CALIFORNIA, BERKELEY
SPACE SCIENCES LABORATORY

PROJECT : AFGL-701-14
TITLE : LANGMUIR PROBE INSTR.
SECTION : BIAS SWEEP ANALYSIS

FILE : SWEEP4.CAD

DESIGNER: PETER R HARVEY
DATE : 15 MAY 86

Figure 16. Bias Sweep Analysis

Theory of Operation

GENERAL. The SWP module is a complicated module controlling two boom systems in two modes, while conforming to a number of system timing constraints, etc. A large number of options are provided for the sake of flexibility that probably won't be used. If things happen the way they did in the ISEE instrument, the algorithm will be eventually replaced by a commanded program, and that will be that. Nevertheless, the RAM loaded programs may find parts of this code useful and thus these should be described.

COMMANDS. Commands are vectored to the module through the CMDTAB vector table as are all other instrument commands (see BKG). To handle the large number of options and values for the two boom systems' bias sweeps, the SWP module used a simple "index and value" commanding format like the ELE INDEX and QTY commands. One command sets the INDEX variable to point to one of the 32 RAM variables which can be changed. The second command sets the value of the variable and then increments the INDEX.

BOOM SWITCHING. The module works on one boom system at a time. The variable BOOM is either 1 or 3 to indicate spheres and cylinders, respectively. At the end of a sweep cycle, the boom system is switched by the routine SWITCH.

SWEEP TIMING. Sweeps are performed at multiples of the spacecraft spin period. The commanded variable SPINMAX holds the number of spins to wait between sweeps. SPINCNT is loaded from SPINMAX when BOOM is set and is decremented each time the SUN

ANGLE is 0.

ONE SWEEP CYCLE. When SPINCNT is zero (the last spin) and the SUN ANGLE matches the SWPANG value for that boom system, the STATE is set to 1. This indicates to the foreground routine SWPEXEC that a sweep is ready to be done, and this will occur as soon as SWPEXEC is called.

SWPEXEC performs the sweep management which is pretty well described in the software listing. It is worth noting :

1. The SAW module status is saved by SWPEXEC before doing the sweeps, and later re-commanded after the sweeps.
2. The SWPPTR variable is set at the beginning of the sweeps and is incremented by the action of the SAMSTO routine. Since sweeps can be disabled, the number of data points is variable. Hence, the TRANSMIT routine is smart enough to telemeter from the start of the buffer to wherever SWPPTR ends up.
3. When the sweep analysis is done, STATE is set to 2. This allows the BIAS setting to occur.

BIAS SETTING. If the STATE is 2 (analysis done), the SWPANG code waits for the sun angle to match the spin fit angle for that boom system. When these are equal, either the RESULT or the ALTERNATE values are sent to the boom system DAC's, depending upon one of the option bits. Note that the code also checks if the mode is correct for setting bias DAC's. It will not set the sphere DAC's in the Langmuir (current) mode.

SWEEP DETAILS. Here are some notes the sweeps:

1. Sweeps are synch'd with the telemetry system by SWPREQ and SWPOK variables. The FDM routine copies REQuest into the OK variable, so to start a sweep the code sets REQ=1 and waits for OK=1.

2. Relays are flipped only if necessary. For example, if performing a voltage sweep and the instrument were already in the voltage mode, the code will switch to the current mode. It is smart enough (because of table lookup) to know that a current sweep on the cylinders can be performed while the spheres are in the current mode.

3. The first BIAS step is output 4*4 or 16 milliseconds before the first measurement is taken. Thereafter, all measurements are taken at 4.00 millisecond intervals (at words 2, 10, 18 etc).

4. Buffered data is sign extended from 12 to 16 bits. This is needed to simplify the analysis phase and doesn't effect the transmit phase except negative values show high gain.

SWEEP ANALYSIS. The sweep analysis routine (for the voltage mode current sweeps) is contained in the SWP module. Users may override this function by using the ANAVECT vector. When ANAVECT is armed (=0AAH), then the ANA routine used is the one pointed to by ANAVECT+1.

Note that there are 4 undefined variables which are available in each of the two sweep parameter blocks. These are zeroed at SWPINIT and are transmitted as spares in the sweep transmissions. If the user reprograms the bias sweep, these variables should be utilized to indicate that this has been done, as well as possibly to show the parameters of the new function.

MODE BITS. The large number of option bits are handled by two routines CHKENA and CHKWTD ("check enable" and "check-what-to-do"), the two distinguished by the first referencing RAM options and the second ROM options. There is one byte of RAM and one byte of ROM enable/disable bits per boom system. These routines form a one bit mask depending upon the instrument mode and the value of [C]. This mask is ANDed with the option bits and NZ if indicated if a 1 is found.

For example, to determine if transmissions of V1,2 sweeps are enabled, a call is made to CHKENA with [C]=40H. In the voltage mode, the mask will be 40H while in the current mode the mask will be shifted to 80H. These are the bit positions for transmission enables in those modes.

2.11 MAIN Program Loader

One of the most important, yet simplest, pieces of the flight code is its program loader in the LD.A module. Together with various vectors tucked away in critical places, much of the MAIN CPU operations can be changed or increased, and any location in the MAIN memory can be modified. (To load bytes into the Burst computer, one invokes the Burst Program Loader BLD.)

For this module there is only one entry point:

LDINIT : This entry initializes by setting the load address to the beginning of the user area defined for programs.

The available commands, as described in the general description, are

ADRL	Set Low byte of memory address
ADRH	Set High byte of memory address
LOAD	Load byte into memory
EXEC	Execute program

It is important to note that programs must begin with a code "AA" in hexadecimal. This is to prevent an errant command from crashing the system by executing a program not ready to be run. (Had the command count error bit been invented prior to this module, it would have been a better check of correct program loading.) This byte is zeroed before the program is executed, so one cannot re-execute a program by sending a second EXEC command.

MAIN CPU programs must always begin at "USER" (address 2930H). If you want to run something elsewhere, you must load a jump at

USER.

MAIN programs are executed as part of the command execution. They are therefore part of the background and have suspended the foreground until they execute a RET instruction. To get to the foreground, one can change the foreground vector in the EXEC module and then RET.

Without using any leftover data areas in the other modules, programs of about 1.6 to 1.7 KBytes can be loaded. The stack is going to operate in the first 30 to 50 bytes from 2FFFH, but one should never get too close if one can help it.

2.12 Boom Deployment

The deployment of spherical booms is performed by the Deployment module. This software is intented to monitor the lengths of the boom systems as measured by their "turns counter" microswitches and to deploy the boom systems as commanded.

Requirements for this module are outlined as follows:

1) Never start a boom motor if its cover is on.
2) Turn off a boom unit immediately if any end-of-wire indicator is tripped (not just it's end-of-wire).
3) When deploying both booms at the same time, if one boom gets too far ahead of the other, turn it off until they are the same length again.
4) An override command can disable points 1 and 2.

For this module there are the following entry points:

DEPINIT : This entry initializes the boom deployment relays and the module by executing a deploy stop command.

DEPSAMP : This entry is used for monitoring the boom status information and making decisions to turn on or off the boom unit motors.

DEPDSC : The DSC entry returns the status of the boom deployment module. On entry: A is an index into the variables used by the deploy module. On return: A contains the value at that location.

Theory of Operation

DEPLOY STATE. The deploy module handles the two booms as separate devices using two nibbles of a state variable DEPSTAT.

Each nibble has four possible states:

OFF	The motor has been commanded OFF and is OFF.
PAUSED	The motor is currently OFF, but is commanded ON.
STOPPING	The motor is currently ON, but is commanded OFF.
RUNNING	The motor is currently ON and commanded ON.

SWITCHING. On each call to DEPSAMP, only one boom system is checked for what to do. This is controlled by the LSB in DEPCNT which is incremented on each call. DEPSAMP first samples the microswitches in order to determine the boom lengths. It then looks at the state of the one boom system. If OFF, it does nothing. If STOP, it calls the IO system to turn OFF that boom motor relay. If PAUSED, it compares the boom lengths and restarts the current boom when the other boom is longer. And if already RUNNING, it checks both the microswitches as well as compares the length of this boom versus the other.

LIMIT CHECKING. The microswitch limits which show the covers ON or the End-of-Wire status is done by the routine LIMCHK. This code returns no-carry is there is a problem, such as a cover which is ON. Limit checking can be disabled if there is a sticky switch by a command which sets the OVERRIDE bits. These are OR'd into the boom status bits in the limit checking process and prevent a zero condition (error) in any of these bits.

COMMANDS. The deploy command simply sets the value of the deloy limit and then uses the two bits from the deploy command (0, 1 , 2 or 3) to set the DEPSTAT variable with the appropriate states of each boom. To turn ON a boom system, one sets the boom in the PAUSED state from which it will try to turn ON. To turn OFF a boom, one sets its state to STOPPING from which it will turn OFF its motor and enter the OFF state.

2.13 General Utilities

The smallest of the modules has to be the UTIL.A module. Many of the modules in the MAIN computer needed the same very low level functions to be performed. These were grouped into this file since it made no sense to put them elsewhere.

The functions are as follows:

ZERO	Zeroes C bytes of memory from [HL] on.
COPY	Copies C bytes of memory from [DE] to [HL].
REF	Adds [A] to [HL] in sixteen bit fashion. This is useful for array referencing.
UNARY	Calculates 2**[A] and returns the value in [HL]
NEG16	Negates [HL] in 2's complement form.
MARK	This routine outputs [HL] to the diagnostic LEDS of the development system

ZERO, COPY and REF have been assigned software restarts 1, 2 and 3 respectively. This conserves on memory since the restart instrument is one byte whereas subroutine calls are three.

2.14 Main Input/Output Utilities

The Input/Output module for the Main computer system is contained in a file called "IO.A". The entry points for it are described below:

IOINIT	Initializes the input/output variables so that the module functions correctly.
GETMASK	Reads the interrupt mask of the processor. On exit: A is the interrupt mask.
SETMASK	Sets the interrupt mask of the processor. On entry: A is the interrupt mask to set.
CMDIN	Reads the command input shift register. On exit: [HL] is the current shifted value.
TMOUT	Sets the telemetry shift register. On entry: [HL] is the value to send to the telemetry shift register.
SUNSTAT	Returns the status of the sun pulses. On exit: Status is non zero if a sun pulse has occurred since the last call.
BOOMSTAT	Reads the microswitches in the boom deployment units. On exit: A is E2.R2.T2.L2.E1.R1.T1.L1 where E is the Endwire indicator, R is the right cover, L is the left cover and T is the turns counter. The endwire and covers are active low.
SAMPLE	Sample an analog quantity on the MAIN multiplexor. On entry: A is the multiplexor address On exit: [HL] contains the 12-bit value (0XXX). If Kelley qty, [HL] = 0XXG, where G is the gain.

AGC	Sample an analog quantity after selecting its gain.	
	On entry: A is a multiplexor address.	
	On exit : D is the proper address to digitize.	
SETBIAS	Sets a bias D/A converter.	
	On entry: H is the boom number (0 thru 3).	
	L is the 8-bit dac value (-128 to 127).	
SETGUARDS	Sets a guard D/A converter.	
	On entry: H is the boom number (0 thru 3).	
	L is the 8-bit dac value (-128 to 127).	
SETSTUBS	Sets a stub D/A converter.	
	On entry: H is the boom number (0 thru 3).	
	L is the 8-bit dac value (-128 to 127).	
SETVTRIM	Sets a Vtrim D/A converter.	
	On entry: H is the boom pair (0 or 1).	
	L is the 8-bit dac value (-128 to 127).	
SETRELAY	Sets or resets a relay in the analog/filtering.	
	On entry: A is the relay number (0 thru 17).	
	Carry is 1 to set, 0 to reset.	
SETFILTER	Sets a Filter register.	
	On entry: H is the filter number (0 thru 6).	
	L is the filter value (0 to 255).	

SETMUX Sets the multiplexor bits which steer the filtering.

On entry: H holds a 3-bit code for which mux to set.

L holds the value (1 to 5 bits)

```
Code      Mux effected
-----------------------------------------------
000       V12/RI1
001       V2/RI2
010       6-POLE SELECT (2 BITS)
               00 : V2/RI2
               01 : V12/RI1 RAW
               10 : SC
               11 : V34 RAW

011       V1/SC
100       COMBINES CODES 0 AND 1 (L IS 2 BIT)
101       COMBINES CODES 0 THRU 2 (L IS 4 BIT)
110       COMBINES CODES 0 THRU 3 (L IS 5 BIT)
-----------------------------------------------
```

TSTMUX Requests a copy of the multiplexor bits which steer the filtering. On entry: a holds a 3-bit code for which mux to request. On exit: a = the bits not right justified, condition code set.

SETMOTOR Turns on/off the boom motors.

On entry: A is the motor number (0 or 1).

Carry is 1 for on, 0 for off.

SETKLY Cycle the Kelley AutoGain Circuit.

On entry: no parameters.

SETPLA Load the LEPA shift register.

On entry: [HL] is the value to send.

SEND Send data to the Burst computer system.

On entry: [HL] is the value to send.

Note: if the burst is already trying to send to the main system, this information will be lost. See the LPHW.DOC for timing information on SEND and RECEIVE.

RECEIVE Receive data from the Burst computer system.

On exit: If zero returned, the Burst is not ready.

If non-zero, [HL] contain the data.

RWATCH Resets the watchdog timer.

On entry : no parameters.

IODSC Requests the digital status of relays, and dacs.

On entry: [A] is the index

On exit:

A =	0-1 :	VTRIM1-VTRIM2
	2-5 :	BIAS1-BIAS4
	6-7 :	STUB1-STUB2
	8-9 :	GUARD1-GUARD2
	10-16:	FILTER1-FILTER7
	17 :	FILTER MULTIPLEXOR (SEE LPHW.DOC)
	18-20:	RELAYS (K0 IS LSB OF 1ST BYTE)

2.15 Fast Floating Point Utilities

The FFP.A module fulfills the needs of the instrument in performing on-orbit data analysis of the DC electric field. Specifically, the Sine Wave least squares fit subroutine requires the range and precision of floating point. Originally designed to produce least squares fits in 500 milliseconds on an 8085 running at 2.5 MHz, this package is roughly 30 times as fast as it needs to be for CRRES. The one drawback to the package is the fact that it uses only a two byte mantissa (instead of three) and therefore has less precision than full implementations. Nevertheless, the package is ideal for scientific applications of this sort.

The format of the data is SIGN (S), 7-bit EXPONENT (E), and 16 bits of MANTISSA (HL) as follows:

```
.--------.--------.--------.
!SEEEEEEE!HHHHHHHH!LLLLLLLL!
'--------'--------'--------'
MSB                      LSB
```

The registers are organized with the current value held in [CDE] and the second parameter pointed at by [HL]. When floating values are stored in memory, they are stored with the exponent byte first and low mantissa last.

The functions available in the package are as follows:

LODFP Loads [CDE] from memory at [HL]

STOFP Stores the result in [CDE] in memory at [HL]

FMUL Multiplies [CDE] by value at [HL], leaving the result in [CDE].

FDIV Divides [CDE] by value at [HL]; leaves result in [CDE].

FADD Adds value at [HL] to [CDE].

FSUB Subtracts value at [HL] from value in [CDE].

FCMP Compares values in [HL] and [CDE] using subtraction. Returns carry and zero flags as appropriate.

FNEG Negates the value in [CDE].

FLT32 Floats a signed 32-bit value in [DEHL] leaving the result in [CDE].

FIX32 Fixes a floating value in [CDE] leaving the signed result in [DEHL].

FSQUA Squares the value in [CDE].

FSQRT Takes the square root of [CDE].

MU21 Fast fixed point 8 bit by 16 bit unsigned multiply. On entry: [A] is the 8 bit value and [DE] hold the 16 bit value. On exit: [AHL] hold the 24 bit result. This is a useful utility though it isn't a floating point call.

Underflow and overflow conditions are treated by returning zero and maximum values respectively.

Useful timing information has been collected under 8085 simulations of the package. These are listed below with respect to their minimum, average and worst cases. (Multiplying by zero would be a minimum case for example).

FUNCTION	MINIMUM (cycles)	AVERAGE (cycles)	WORST (cycles)
FADD	76 cyc	300	465
FSUB	97	400	716
FMUL	48	600	1003
FDIV	48	1600	2030
MU21	197	250	298

Table 9. Fast Floating Point Execution Times

These data are useful for estimating the amount of time it will take for the CPU to calculate the floating point result. To convert these cycle times into microseconds, multiply by 0.4. For example, a worst case FMUL will take 1003 X .4 or 400 microseconds.

2.16 Matrix Utilities

The MATRIX.A package is really just one routine which solves up to 4 X 5 matrices in floating point. It has two entry points, IMATX, for defining where the matrix is, its size, and where the solution should go. The second entry point, SOLVE, calculates the solution to the matrix defined earlier and stores it where it was told.

The procedure the standard one which first diagnalizes the matrix to produce the result for one unknown. That value is then substituted back into the equations to remove the unknown value. This gives the result for the second unknown and so forth.

The matrix solver operates on either 4X5, 3X4 or 2X3 matrices. To save time in the spin-fitting solver and to keep the indexing in this module simple, matrices smaller than 4X5 are still stored in the same amount of space as a 4X5 matrix would take. In other words, the 1st element of the 2nd row is always stored in the 6th memory position as if 5 elements were in the top row.

Finally, one other note about the module is its definition of "0.0" when trying to find non-zero elements in the diagnalization process. A common problem with floating point is that small errors get significant when large multipliers are used. (The same problem exists in a smaller way with integers, but that's another story.) As a result, the diagnalization process must stay away from choosing very small non-zero numbers as the radix. This package uses 1/2**10 as the limit defining a practical zero.

2.17 Trigonometric Functions

The TRIG.A package provides floating point subroutines for the common trigonometric functions. These are used by the spin-fitting subroutine to calculate sin and cosine terms in the matrices. Since data is sampled at fixed intervals in the spin package, the trigonometric functions did not need to be complete and, in fact, only work for 32 discrete angles. Their only real requirement is to be very fast. Hence, the trig functions merely play small games with the angle parameter and then reference a floating point table. Note: the angle ranges between 0 and 31*3, in steps of 3.

The functions available are:

SIN	Returns [HL] -> Sin(A)
COS	Returns [HL] -> Cosine(A)
SINSQ	Returns [HL] -> Sin(A)**2
COSSQ	Returns [HL] -> Cosine(A)**2
SNCS	Returns [HL] -> Sin(A) * Cosine(A)

2.18 Burst Executive Module

The BEXEC.A module contains the Burst computer executive logic. This software is responsible for initializing all the modules and then distributing any commands which come into the system. Except for some interrupt-timed sampling modes, the Burst computer operates almost completely in the foreground.

The executive has only 1 entry point, namely the cpu reset. All other modules are called by BEXEC. None call it.

Commands from the MAIN cpu enter the system by an interrupt which causes the transfer of data from the MAIN to the BURST IO system. The executive polls the IO system to see if there is a command using the RECEIVE function. When this returns the NZ flag, [HL] contain the command bits.

With a valid command in hand, the executive calls the other Burst modules to see whose command it is. Each module returns a carry when the command is not theirs. If it is theirs, they execute it, of course. Command errors are recorded in the BEXEC RAM area but NO status is reported to the MAIN computer. This could be a place for future improvement.

In order to cut down the power of the Burst system, the executive uses the same trick as the MAIN executive when it has nothing to do. If no command is ready, the BEXEC module puts a HLT (halt) instruction followed by a RET (return) in RAM memory, and then executes it. This causes the CPU to shut down, until the command interrupt from the MAIN system, at which time the Return instruction is executed. The Burst then recogizes the command-ready and executes it.

Two vestigial diagnostic routines remain in the Burst executive module. Since the Burst memory was no cramped, these were left in for the sake of future problem solving. One is a memory test program which tests the buss memories, not the Burst Memory bank. The other is a diagnostic output routine which displays [HL] to the diagnostic LEDS of the development system.

2.19 Burst Input/Output Module

The Input/Output routines for the Burst computer system are contained in file "BIO.A." The entry points for it are described below:

BIOINIT : Initializes the input/output variables so that the module functions correctly.

GETMASK : Reads the interrupt mask of the processor.
On exit: A is the interrupt mask.

SETMASK : Sets the interrupt mask of the processor.
On entry: A is the interrupt mask to set.

SETVECT : Sets/Resets the 2KHz interrupt vector.
On entry: If [HL] is zero, the 2KHz int is disabled.
Else the vector is set to [HL] and the int is enabled.

RECSTAT : Returns not zero if RECEIVE data is ready.

RECEIVE : Receives data from the MAIN processor over the interprocessor communication lines. If no data is ready, the zero flag is set on return.
On exit : If zero, no data ready.
If not zero, [HL]= 16-bit data.

SEND : Sends data to the MAIN processor.
On entry: [HL] contain the data to be sent.

ADPWR : Controls power to the A/D converter circuitry.
On entry: Carry = 1 to turn ON the A/D.
Carry = 0 to turn it OFF.

SAMPLE : Sample an analog quantity on the BURST multiplexor.

On entry: [A] is the multiplexor address.

On exit : [HL] contains the 13-bit value in the format

(...G xxxx xxxx xxxx)

MEMPWR : Controls power to the memory banks.

On entry: [A] = the bank number to turn ON or OFF.

carry = 1 for ON, 0 for OFF.

MARSET : Sets the Memory Address Register.

On entry: [BHL] = 18-bit address to set

BANKSET : Sets the start and end banks to use.

On entry: [B] = the start bank to use (0..5)

[C] = the end bank to use (0..5)

MODESET : Turns on/off the memory autowrite mode.

On entry: [A] = 1 for autowrite, 0 for normal memory.

SECOND : Delays 1 second.

D5MS : Delays 5 milliseconds.

READ : Performs a memory read of the burst memory and returns the value in [HL].

WRITE : Writes [HL] to the burst memory.

REWIND : Resets the burst memory address register to the start bank.

MARGET : Returns the value of the memory address register in [AHL]. Note: since the hardware counter itself cannot actually be read back, the software simulates the action of the MAR whenever READ, WRITE, MARSET and REWIND are used.

SETIO : This function sets the "IOMODE" to parameter [A]. In IOMODE 1, the carry flag will be set when a command interrupt occurs. This is used by the BURST sampling procedure so it need only check carry to decide when to stop.

2.20 Burst Format Control

The BFMT.A module controls the sampling format lists. As described elsewhere, the Burst can remember 16 formats, 10 of which are in ROM while 6 are in RAM. This allows for sophisticated programming which might take different Bursts depending upon conditions seen by the MAIN cpu decision maker. The ability to hold multiple lists lowers the time needed to switch between lists since one need only refer to the list, rather than define it each time.

The module has the following functions:

INIFMT	Initializes the module and defines the default RAM formats 10 thru 15.
SETFMT	Sets which format to use. On entry: [A]=format number.
ADDFMT	Add a quantity to the current format. On entry: [A] holds the quantity's multiplexor address.
ADRFMT	Returns the address of the current format in [HL].
LNGFMT	Returns the length of the current format in [A].
ENDFMT	Returns the end address of current format in [HL].

Theory of Operation

The sampling list are contained in two separate areas, one for ROM and one for RAM. Each format is simply a list of bytes ended by an EOL (End-of-List) marker. The 10 ROM lists and 6 RAM lists are placed contiguously in memory so they occupy the minimum amount of space. The code provides only 64+6 bytes of total space for the 6 RAM lists, so this limits the RAM quantities to a maximum of 64.

Finding the start and end addresses as well as the length of any list is done by linear search. Adding to a given list in RAM simply moves all RAMLIST bytes down one. There isn't much to this, so I won't labor the description. Just look at the listing for more details.

2.21 Burst Program Loader

Just like the MAIN system, programs can be loaded into the memory of the BURST computer system and then executed. Even more so than in the MAIN system, BURST programs can completely change the way the BURST operates, since almost all of the BURST is run in the foreground.

For this module there are two entry points:

BLDINIT Initializes the module and resets the load address register to point to available memory.

BLDCMD Accepts loading type commands:

BADRH	Set high byte of address register
BADRL	Set low byte of address register
BLOAD	Load a byte into memory, increment register
BEXEC	Execute the program.

It is important to note that programs must begin with a code "AA" in hexadecimal. This is to prevent errant commands from crashing the system. As in the MAIN system, this byte is zeroed before the program is executed.

BURST programs load at address 1202H. That is where the code "AA" must appear. The first executable opcode must be loaded at 1203H. The user program area extends from 1203H to nearly 17FFH less 20 bytes or so for the stack. This amounts to approximately 1.5 KBytes.

2.22 Burst Sampling Module

Burst sampling functions are contained in the BSMP.A module. These include controlling the frequency of Bursts, the memory banks used, starting, stopping and playing back data to the MAIN system. This module uses the BFMT module for its format control.

The BSMP module has two entry points:

BSMPINIT This entry initializes the module, sets initial default values such as the frequency and so forth.

BSMPCMD This entry executes Burst commands which are in [HL] registers.

Theory of Operation

INITIALIZATION. The initialization of the sampling module sets up the defaults as follows: (1) use the V12 only format (2) frequency to maximum, (3) burst A/D turned OFF to save power, (4) memory banks 4 and 5 turned ON while banks 0 thru 3 are turned OFF. This allows for small bursts of V12 data to be run with just a BGO command.

BANK SELECTION. If response to Bank commands, the software sets the memory address control hardware so that the sampling uses all of the memory banks which are powered. Routine BSELECT extracts the STBANK and ENBANK (start and end banks) while routine MEMPC turns those memory banks ON (and the others OFF).

A more sophisticated sampling program may wish to utilize only part of the memory at a time. For example, one could take 6 separate Bursts and then play them all out at the same time, or perhaps choose which to play back and which to junk. The data returned would be HELL to analyze of course.

BURSTING. The process of taking a Burst is as follows:

1. Make sure the A/D converter is turned ON.
2. Compile the current format into a program.
3. Send the MAIN computer information about this burst:
   a) the real frequency of the burst
   b) the number of milliseconds to fill the burst
4. Rewind the memory and write a "START" marker
5. Check that the memory is working by reading back the "START" marker
6. Calculate the amount of delay required for this frequency
7. Mark the memory as an "OPEN" file.
8. Execute the compiled program.

STOPPING. The Bursting stops when the MAIN sends anything to the Burst. The BSTOP command turns OFF the A/D converter and "CLOSES" the Burst memory by obtaining the Start and End addresses of the data in the memory. Since closing a file may take the Burst processor several seconds, the MAIN computer waits for the BSTOP command to signal the end of the file closing process.

PLAYING BACK. The PLAY command starts the playback of header and data to the MAIN system. The playback begins with the header which is played by routine PHEAD. Once the header is finished, the data section begins.

The data section is played from the start address (STADR) to the end address (ENADR) which were determined at the time of file closing. Since memory errors due to cosmic ray upsets, the code does not simply look for the End Mark which was written into the memory.

Playing back may be interrupted by any command from the MAIN processor. Bursts may be played back over and over again, but this is more of a testing level feature than one useful in orbit.

DURATION CALCULATION. The calculation of the time it takes to fill the memory at a given frequency is errant by a small fraction as a result of the late changes to the burst frequency. The calculation tends to overestimate the time by about 5 percent.

This calculation was originally going to be used by the Burst controller module in the MAIN cpu as a way of deciding how long to wait for a Burst. This turned out to be very hard to use. Nevertheless, it is still available for more sophisticated Burst triggering algorithms.

FILE CLOSING. Closing a memory file of data means to find the start and the end addresses of the data in the memory. This is obviously necessary in order to play back the data starting from the first data point and ending with the last. Since the memory has a "wrap-around" capability, the first data point to be played out may occur anywhere in the memory and it takes this routine to figure out where it is.

The first step is to find the End address (where in the memory the last data point is). This is done by writing an END marker into the memory, rewinding and searching for the END marker.

Step two is to determine if the Burst was short or long. Short bursts occupy less than the amount of available memory while long bursts last long enough to overfill the memory. Short bursts are the easy case, detected by the fact that their "START" marks are still in the memory (at the rewind point). To close the file on a short burst, the routine simply records where it found the START and the END markers.

Long bursts are much harder to close. They have no "START" marker since the wrap-around feature of the memory wiped it out. Long burst have 1 END marker and an unknown number of PAUSE markers. We know that the very next memory location following the END marker is some data, but we don't know which of the quantities in the sample list it corresponds to. If there are three quantities in the sample record, it could be any one of them. In other words, we must skip over the partial record of data following the END marker (if there IS a partial record).

One way to compute the partial record is to divide the available memory by the length of each record. The remainder is the amount to skip. The available memory would be calculated from the memory size less the END marker and PAUSE markers.

A simpler method was to simply search for the first END or PAUSE marker following the END marker while keeping track of the record elements.

2.23 Burst Compiler

The Burst sampling program compiler is contained in the "BCMP.A" module. It has 1 entry point as follows:

BCMP : Compiles a high-speed sampling program from a sample list.

On entry: [HL] points to an available code area;
[DE] points to a sample list;
[C] = size of the sample list.
[B] = 2 for interrupt type timing
1 for software delay timing
0 for no delay at all.

On exit: [HL] points to the next available memory loc.

Theory of Operation

Naturally, one might ask "Why is there a compiler in the BURST computer system?" The reason is that it is too difficult to write sampling programs which take into account all the right gate delays and parallelism provided by the hardware. The compiled code runs extremely fast because its takes full advantage of these characteristics.

Much of the details on the burst sampling are discussed in the hardware chapter and I won't repeat these. Rather, I list below only a few examples of the code produced by sampling lists of various sizes (the compiler produces binary of course):

SINGLE QTY SAMPLING ROUTINE (NO DELAY).

```
          MVI  B,f             ;INIT( FREQ= f )
          RST  3               ;
     LOOP LHLD MEM+ADC+QTY1    ;SAMPLE/STORE 1 POINT
          JNC  LOOP            ;WAIT FOR COMMAND READY
          JMP  ENDBURST
```

The execution time of the loop is 40 cycles (16 microseconds) which matches the hardware capabilities.

SINGLE QTY SAMPLING WITH A DELAY OF C SAMPLE PERIODS

```
          MVI  B,f             ;INIT( FREQ= f )
          RST  3               ;
     LOOP LHLD MEM+ADC+QTY1    ;SAMPLE/STORE 1 POINT
          MOV  L,C             ;DELAY C SAMPLE PERIODS
          CALL SOFDLA
          JNC  LOOP
          JMP  ENDBURST
```

The sampling frequency is approximately 62.5 KHz divided by (C+1). See the section below for more details.

MULTIPLE QTY (4) SAMPLING PROGRAM WITH INTERRUPT TIMING

```
          MVI  B,f             ;INIT( FREQ= f )
          RST  3               ;
     LOOP LHLD MEM+ADC+QTY3    ;MEM=QTY1,DIGITIZE QTY2, MUX=QTY3
          IN   7FH             ;(DELAY 10 CYCLES)

          LHLD MEM+ADC+QTY4    ;MEM=QTY2,DIGITIZE QTY3, MUX=QTY4
          IN   7FH             ;DELAY 10 CYCLES

          LHLD MEM+ADC+QTY1    ;MEM=QTY3,DIGITIZE QTY4, MUX=QTY1
          IN   7FH             ;(DELAY 10 CYCLES)

          LHLD MEM+ADC+QTY2    ;MEM=QTY4,DIGITIZE QTY1, MUX=QTY2

          MOV  L,C             ;DELAY C INTERRUPTS AT 2KHZ
          CALL INTDLA

          JNC  LOOP            ;LOOP UNTIL CARRY SET (CMD RDY)
          JMP  ENDBURST        ;THEN QUIT
```

The sampling frequency for the loop is 2/C KHz, where C is the value contained in the C register.

BURST FREQUENCY UPDATE. A late modification to the Burst frequency changed it from 59.5 to 62.5 KHz and the software was not able to be updated. This causes the SOFT-DELAY routine to delay more than the equivalent sample by 5 percent. If C=1 in the above case , this would cause the overall frequency to be 2.5 percent below 31.25 KHz or 30.49 KHz. A correct calculation of Burst frequencies corresponding to the parameter RFREQ and the size of the record NREC is given below:

1. Find NMAX in the table corresponding to RFREQ.

RFREQ	14	13	12	11	10	9	8
NMAX	1	2	3	4	6	10	20

2. Calculate the number of delays used:

   NDLA = NMAX - NREC

3. Calculate the Frequency as follows:

   Frequency = 1000000/ (NREC*16 + NDLA*16.8)

Only high frequency bursts (over 3 KHz) use software delays to regulate the frequency. Interrupt regulated frequencies (2 KHz and under) are not affected.

2.23 Burst Floating Point Utilities

The Burst computer system comes complete with its own floating point package for the simple reason that there was plenty of room in the ROM, that the DURATION calculation was facilitated and third, some ground loaded program may be able to use it. For the sake of sanity in file-keeping and in order to simplify the assembly process, it is contained in the BFFP.A file, not in the FFP.A file.

The entry points are exactly the same as those in the MAIN floating point package so please refer to that section for further information.

3. Hardware

The function of this chapter is to describe the circuitry and physical characteristics of the CRRES Langmuir Probe Instrument (AFGL-701-14A) and its spherical sensors. It is assumed that the reader understands related parts of the general instrument description.

This chapter describes each of the seven boards of electronics in 701-14A. The functions of each board are summarized below:

Analog Board :	Sphere and Cylinder Sensor Interfaces AC Instrument Interface
Filter Board :	Filtering of sensor signals Fluxgate and Search Coil Interfaces
IO Board :	D/A conversion for bias control Relay controls for sensors and Analog Filter board control
MAIN CPU Board :	Telemetry Formatting, Command Reception MAIN A/D Conversion
BURST CPU Board:	High Speed Data Sampling Burst Memory control circuitry
Memory Boards :	3 Banks of 32 Kbytes on each board Individually powered

3.1 The Analog Board

The "Analog" board is so called because it handles the primary analog inputs and outputs of the instrument. The primary input signals are the sensor inputs from the two spheres and two cylinders. The outputs are the sphere voltages which are repeated to the IOWA Sounder instrument.

SPHERE BIASING. Each spherical sensor has three control signals called BIAS, STUB and GUARD. These are generated by "offset" circuits which add the +/- 2.5 Volt signals (supplied by the IO board) to the voltage output from the sphere preamplifier. As a result, these voltages track the sphere potential up or down. The range of the STUB and GUARD voltages is approximately +/- 37 Volts, while the STUB voltage ranges between +/- 1.25 Volts. For example, if the IO board supplies a +2.5 Volt signal to BIAS 1, and the sphere 1 preamp is 1 Volt, then the BIAS 1 output will be 37+1 or 38 Volts. The maximum range of these outputs is +/- 100 Volts.

SPHERE MEASUREMENTS. Each sphere produces two outputs called V/I and I/V. The V/I output represents the voltage measured on the sphere when the sphere is in the voltage mode, and the current collected when in the current mode. The I/V output is just the reverse of these.

The "voltage" output from the sphere 1, for example, is current amplified by the opamp U5 and fed four places: 1) the GUARD offset circuit, 2) the STUB offset circuit, 3) the floating power supply driver and 4) the signal divider. The floating power supply driver (sphere 1) is comprised of U5, Q13, Q14, etc.

and simply keeps the ground for the sphere components near the preamp's output voltage. In this way, the sphere preamps can float between -100 and +100 Volts with respect to the instrument ground, while all the sphere op amps nead only operate from +/- 12 Volts. The +/-100 Volt signal from U5 is divided down to +/-5 Volts by the resistor divider R79/R80 and is then buffered by U1 for exit to the filtering board.

The "current" output from sphere 1 is really the voltage from the DAC plus the collected current times the resistor in the sphere. The op amp U4 is used to remove the DAC voltage from this measurement while amplifying the signal a factor of ten. The resulting voltage, RI1, is fed to the multiplexor U3 and optionally to the Sounder interface circuit.

SPHERE RELAY CONTROL. Each sphere contains two relays which determine whether it's in the voltage or the current mode. For lack of wires both relays are controlled using the same wire. Relays K2 and K3 control whether power is sent down the mode control line. Relay K1 determines whether +35V or -35V will be sent. Each sphere relay circuit uses diodes to determine which coils should get the power based upon the polarity of the signal. Note that the sphere relays are 12 Volt relays, but since the 50 meter wire is so resistive, 35 Volts had to be used on this end.

CYLINDERS. Control of the bias current to the cylinders and receiving the measurements proceeds in much the same way as the spheres. However, no floating power supplies are used as the cylinder amplifiers work from fixed +/- 35 Volt supplies.

Relays K4 and K5 pulse the relays inside the cylinder preamplifiers to either put biases on the cylinders or not. Relay K6 holds closed the calibrate relay inside the cylinder preamp.

SENSOR DIFFERENCING. The main measurement of the instrument is the difference in voltage between the two spheres and between the two cylinders. This difference is done by opamp U4 (for V1 - V2) in a way that allows for the introduction of a trimming potential from the IO board. This trim potential will allow for the near-zeroing of any offsets which arise because of radiation effects to the opamps.

DIFFERENCING AMPLIFICATION. As a precaution again possible noise problems in the system, the Analog board provided an extra amplifier for getting a high gain version of the difference measurement. These signals are called V12X50 and V34X50, and proceed directly to the inputs of the MAIN analog multiplexor.

DERIVATIVES. Both difference measurements are also differentiated to make what are called "AC" signals V12/RI1 AC and V34 AC.

RELAYS. There are two types of relays on the Analog board, latching and non-latching. The former require current to be applied to their coils for the entire time they are flipped. They flip within 2 milliseconds after current is supplied.

The latching type require only a 2 millisecond pulse directed down one of its two coils, one to set and one to reset the relay. In order to cut down on the number of drive transistors required for all these coils, a separate relay (either K1 or K0) is used to determine whether current will flow down the set or the reset coil of a relay. This is illustrated below:

```
------------------------------------------------------------------

                                            .---------------
SET 7 to 17 :              K1      _____!

                                                 .----.
                           Kn      __________!    !_____

                                       -----.
RESET 7 to 17 :            K1               !_______________

                                                 .----.
                           Kn      __________!    !_____

------------------------------------------------------------------
```

Figure 17. Relay Set and Reset Logic

The relays which are in the spheres and cylinders require extra time to flip since they are only controlled by relays on the analog board. For example, to flip relay K18 (sphere 1), relay K2 must go into the set position. We must wait for K2 to settle, then wait more for K18 to settle. Extended relays like these are flipped as seen below:

```
                                          .--------------------
SET (18,19) :               K1     _____!
                                               .----------.
                          K(2,3)   __________!            !_____


                                   -----.
RESET (18,19) :             K1          !____________________

                                               .----------.
                          K(2,3)   __________!            !_____


                                          .--------------------
SET (20, 21) :              K0     _____!

                                               .----------.
                          K(4,5)   __________!            !_____

                                   -----.
RESET (20, 21) :            K0          !____________________

                                               .----------.
                          K(4,5)   __________!            !_____
```

Figure 18. Extended Relay Set and Reset

3.2 The Filter Board

The filter board has three major functions: 1) to interface with the Fluxgate Magnetometer, 2) to interface with the Search Coil Magnetometer and 3) to filter the above signals as well as the signals from the Analog board.

FLUXGATE INTERFACE. The Fluxgate signals (X,Y, and Z) come into the board in +/- 10 Volt form (see page 2 of the Filter drawings). The BY input can be multiplied by a factor of 6 using the amp U27 when selected by multiplexor U31. The three signals are rolled off to 60 Hz and converted to +/- 5 Volts by one stage of opamps U32 and U34. These three signals are called BXFAST, BYFAST and BZFAST, and are sent to the IO board for multiplexing into the Burst computer system.

A second stage of filtering, again using U32 and U34, rolls off these signals to 6 Hz for input to the MAIN multiplexor U29.

SEARCH COIL INTERFACE. The Search Coil Magnetometer measurement comes to the Filter board as a differential signal. Amp U22 converts this to a single ended signal, ready to be used by multiplexor U23.

VARIABLE FILTERS. On the board are seven programmable filters which are used for signals heading for the Burst computer system. Since burst collections can be taken at frequencies from 10 to 60 KHz, the programmable filters are used to stop aliasing while sampling at the lower rates. The heart of each filter is the D/A converter located in the feedback loop. The resistance in the DAC changes the rolloff characteristics. (See the filter rolloff figure below).

FIXED FILTERS. For the sake of telemetry sampling, signals input to the MAIN multiplexor are rolled off at 1/2 their approximate sampling frequencies.

BAND FILTERS. As a gauge of AC activity on V12/RI1 the Filter board has three comb filters called F1, F2 and F3, whose notch frequencies are 32, 256 and 2048 Hz, respectively. These are drawn on page 2 of the filter diagrams.

KELLEY GAIN CIRCUIT. Finally, the analog or clean part of the KELLEY automatic gain circuit is found on the Filter board. The digital section on the IO board controls the operations of this circuit. See below for the description.

Table 10. Filter Rolloffs

FILTER	BURST QTY	FILTER MAX FREQUENCY
1.	V12/RI1 AC	11.5 KHZ
2.	V2/RI2	11.5 KHZ
3.	V12/RI1	25.5 KHZ
4.	DIRECT	11.5 KHZ
5.	V1/SC	15.0 KHZ
6.	V34	18.0 KHZ
7.	V34 AC	12.0 KHZ

3.3 The Input/Output (IO) Board

The primary functions of the IO board is to interface the central processor boards (MAIN and BURST) with both the ANALOG and FILTER boards. The fundamental reason for a separate board comes from the requirement that all analog signals must be kept as clean as possible of digital noise.

There are six subcircuits to the IO board:

1) Bias control
2) Relay control
3) Boom status information
4) Filter control
5) Kelley Automatic Gain Control (Digital part)
6) BURST multiplexing and Gain control

SERIAL/PARALLEL CONVERSION. The IO board is controlled by a total of 6 lines over which information is passed serially. The historical reasoning behind this was to minimize wires travelling between the digital and analog sections of the instrument and, by such an arrangement, optimize the ability to shield fewer wires, etc. As it stands, this vestigial protocol could be removed with little overall effects to cleanliness, while vastly improving the bandwidth of processor control to the board.

BIAS CONTROL. The bias currents which go to both the spherical and cylindrical boom systems are controlled by eight 8-bit DAC's (packaged in 4 AD7528 dual-DAC's).

RELAY CONTROL. The relay control circuit is responsible for providing power to the relay coils on the analog board. The circuit can simultaneously provide coil power to a total of three relays: K0, K1 and 1 of K2 thru K17. (Note: there are no relays K12 and K15). This is accomplished using 7 bits from the IO shift register: 2 bits directly control the K0 and K1 coils through coil-driver circuits; 5 bits control relays K2 thru K17 using one disable line and 4 address lines to select 1 relay coil using 2 3-to-8 decoders.

Each coil driver consists of a 4050 gate, a base current limiting resistor and a PNP transistor. The normal "OFF" condition is a high level of the 4050 which causes no current to be drawn thru the transistor. A low level on the 4050 turns "ON" the transistor and pulls roughly 20 times that current through the relay coil which is attached between the transistor and the -7 Volt supply.

The minimum output sink current of the 4050B is roughly 2ma when the 4050 is commanded low and its output held at 0.4 Volts. The minimum current delivered thru the base is limited by the 2.2K resistor which at 4 Volts is around 2ma. With a gain of 20 through the transistor, deliverable current to the relay coil is approximately 40ma. Smaller resistors (3.3K) are used for the latching type relay drivers and limit deliverable current to 25 ma each. Since the relay coil resistances are 390 and 500 Ohms for the non-latching and latching types, respectively, each coil will see 12 volts across it when commanded. This retains a 25% margin over the minimum guaranteed switching voltage for these relays.

The power dissipated is each transistor itself is .25V x

40ma = 10 milliwatts which is safe for continuous operation.

BIAS CONTROL. The eight biases which can be applied to the spheres, the stubs, the guards and the cylinders, are controlled by circuitry on the IO board. Each bias voltage generated is a bipolar signal which gets added to other signals on the analog board.

Each bias voltage is made using 1/2 of a 7528 dual D/A converter whose reference is tied to +5 Volts. The current output is converted to a voltage by an inverting amplifier whose output is in the 0 to -5 Volt range. This signal is then subtracted from a -2.5 Volt reference by a second op amp whose output is bipolar and centered on zero Volts.

Each 7528 is programmed by its 8 parallel input lines and chip select which are attached to the serial shift register mentioned earlier. The write and DAC select (A/B) are directly controlled by MAIN computer output pins.

Note that the sharing of 3 lines between the bias circuitry and relay control has the limitation that one cannot hold relay coil K2 thru K17 while changing the bias D/A values.

BOOM STATUS. The IO board is also responsible for making motor currents, length potentiometers and thermistors into voltages for the MAIN A/D multiplexor. Converting motor current into voltages involves simple op amps. Measuring the lengths and temperatures merely requires pull-up resistors.

FILTER CONTROL. The IO board controls the low pass filters and multiplexors of the Filter board with 8 strobe signals, 8 data lines, and 5 multiplexor bits. These are shifted into a 24-bit shift register composed of 3 4094 shift registers. This 24-bit register re-uses the same clock and data lines as the 16-bit register for biasing and relay control, but has a separate strobe line. Thus, one can program only the 24-bit or the 16-bit register at a time.

KELLEY GAIN CONTROL. The digital control section for the automatic gain circuit consists of an up-down counter (54193), a latch (4042) and some random logic. On a rising edge of "GAIN CTL MAIN", the 4042 latches in the OVER and UNDER signals which are determined by comparators on the Filter board. These latched values are combined with the strobe signal and limit checking logic to provide either a "count up" or "count down" signal to the 54193. Limit check is performed to keep the gain in the range of 0 to 13.

BURST MULTIPLEXING AND AUTOGAIN CONTROL. The BURST multiplexor circuit is placed on the IO board instead of the Filter board because of the need to switch its mux address at high speed. Between the Filter board and the IO board are isolation ("feedthru") filters which keep the Filter board free from digital noise.

The multiplexor has 16 input channels which are addressed by BURST MUX A0 thru A3. The output goes to both "times 50" and "times 1" amplifiers as well as a bipolar comparator operation. The comparator is formed by 2 CMP04 comparator stages, one for positive and one for negative signals. Their outputs are open collector and thus pull down when they are active. The limit used to decide whether the signal is "high gain-able" is roughly 1 65th of full scale.

The "times 50" amplifier has one additional feature which speeds up the process of preparing a quantity for digitization. The feedback loop is clamped so that the op amp cannot saturate. Once in saturation, this type of amplifier can take many microseconds to work properly again. Thus, no matter the prior state of the multiplexor, the high gain amp will be ready for the new quantity.

Figure 19. Filter Board Control Register Definition

```
.--------.--------.--------.
!K MMMMMM!DDDDDDDD!FFFFFFFF!
`--------'--------'--------'
MSB                      LSB
```

---

where F represents the Filter Strobes as follows:

```
(LSB) 0:   VTRIM
      1:   V12/RI1AC
      2:   V2/RI2
      3:   V12/RI1
      4:   DIRECT AC
      5:   V1/SC
      6:   V34
(MSB) 7:   V34AC
```

and D is 8-bit data for the filters as follows:

```
      1:   FILTER AT LOWEST ROLLOFF VALUE
    255:   FILTER AT HIGHEST ROLLOFF VALUE
      0:   FILTER UNDEFINED (DO NOT USE)
```

and M represents the Filter Multiplexor bits as follows:

```
(LSB) 0:   V12/RI1 CONTROL--- 0 = RI1    1 = V12

    2,1:   6-POLE SELECT  --- 00 = V2/RI2
                              01 = V12/RI1 RAW
                              10 = SC
                              11 = V34 RAW

      3:   V1/SC CONTROL  --- 0 = SC     1 = V1

      4:   BY/BY6 CONTROL --- 0 = BY     1 = BY*(-6)
      5:   V2/RI2 CONTROL --- 0 = RI2    1 = V2
```

and K represents the Kelley Gain Control circuit reset

```
      1:   Holds Kelley Gain Control circuit reset
      0:   Allows gain circuit to operate
```

Figure 20. Analog Board Control Register Definition

```
.--------.--------.
!DDDDDDDD!AAAAAAAA!
'--------'--------'
```

WHERE D is the data to the DAC's or the relay number

A represents the Analog select bits as follows:

0:	Bias 1/2 DAC Select	(Active low)
1:	Stub DAC Select	(Active low)
2:	Guard DAC Select	(Active low)
3:	Bias 3/4 DAC Select	(Active low)
4:	Disables Relays K2-K18	(1=disabled)
5:	Relay Set Select	(0 for K2-9)
6:	K0 coil	(Active low)
7:	K1 coil	(Active low)

3.4 The Main Processor Board

The main processor board is the heart of the instrument. Information flows to and from the outside world to all parts of the instrument through this board. It's basic purpose is in digitizing data, formatting telemetry, and receiving commands from the ground. Associated with those services, this processor controls the analog and filter boards, calculates magnetic field data for the LEPA instrument, and controls the Burst computer. The following paragraphs describe the subcircuits of this board.

THE PROCESSOR. Central to the design is the processor itself (U2), a Sandia SA3000 which is really just the radiation tolerant version of the Intel 8085 (although in CMOS). The input crystal frequency used is 5 MHz which produces an internal cycle time of 400 nanoseconds. This is a lower frequency than is possible with the part (post-radiation) and was chosen since it would both increase the reliability of the part as well as lower its power consumption.

Connected to the processor buss for compatibility with the OKI processor are pullup resistors (U29) which give the buss some direction whenever the OKI 8085 part is tristated. This occurs at reset and whenever the cpu executes a HLT instruction. The board thus supports both the Sandia and OKI manufactured parts. However, if the Sandia part is to be used, the resistor package should not be installed since the Sandia has weak output latches which might fight with the weak resistors.

One item to note about the Sandia cpu, however, is that it tends to not start up using the standard crystal interface (2

10pF capacitors on pins 1 and 2). Instead, Sandia uses the less common interface as shown on the drawing (1 10 Mohm between pins 1 and 2, 1 20 pF capacitor on pin 1 only).

The 8085 multiplexes its low 8 address bits with its data bits and this causes problems for ROM and RAM devices which require the low address bits to be stable for the entire READ cycle. For this reason a buss latch (U3) and pullups (U47) are included. The latch holds the low 8 address lines which are strobed out by the processor while ALE is high.

INTERRUPTS. The processor can be interrupted by any of five types of events. In order of priority from top to bottom they are 1) WatchDog Timer Overflow, 2) 1 KHz clock, 3) Major Frame pulse, 4) Command Envelope, and 5) RC Int. Interrupts 3 and 4 are simply the filtered signals from the spacecraft interface. The 1 KHz clock is the 2 KHz telemetry synchronous clock divided by 2 by the flip-flop U9A. The WatchDog Timer Interrupt is the very highest priority interrupt and is connected to the Non-Maskable-Interrupt of the 8085 processor (see below).

The RC Interrupt is used by the software to shut off the cpu for a time while the A/D converter is operating. The software simply outputs a "1" to the RC using U14 pin 1 and goes into HALT. The interrupt wakes up the processor after approximately 1 RC time of 33 microseconds so it can get the results of the conversion. (the conversion completes in 11.2 microseconds.)

WATCHDOG TIMER. The Watchdog timer is a small circuit (U5) which is intended to "wake-up" software which has crashed or gotten lost as a result of a Cosmic Ray upset, other radiation damage or errant commanding. In actuality, it is a counter which is incremented by the Major Frame pulse (every 4 seconds) and reset by software. If the software does not reset this timer/counter in any major frame, the WatchDog will stop the software by issuing a Non-Maskable-Interrupt.

WAIT STATE GENERATOR. Since some of the design uses 4000 series CMOS devices which are not fast enough (over temperature) to drive the 8085 bus at 5 MHz, a wait state generator is included to provide a small delay when these devices are addressed. More specifically, the wait state is applied whenever an address of 8000H or greater is used. All CMOS devices which need a wait state are given an address in this range.

The wait state works as follows. The falling edge of ALE clocks valid A15's into the flip-flop U10B. If A15 is a 1 (address above 8000H) then the inverted Q goes low which declares a NOT-READY condition. This will cause the 8085 to delay its READ or WRITE cycle until RDY is 1 when it begins a clock cycle. The wait state generator releases control of the 8085 on the first rising edge of CLK since U10A takes A15(=1) into its D input, causing its -Q output to go low, thus resetting U10B. This action causes the -Q of U10B to go high, which re-enables the RDY input.

ADDRESS DECODING. Devices are distinguished from one another by the 3-to-8 line decoders U22 and U25. Both are Sandia 2995's which are emulations of the standard 74LS138 device. U22 handles addresses over 8000H while U25 handles addresses below

8000H. A full address map is given below.

READ ONLY MEMORY (ROM). The program memory for the computer is contained in a pair of Raytheon 29673 SMB ROMS, each holding 4096 8-bit bytes of information. To lower the power consumption of these devices, 5 Volts is provided only while they are selected. This is accomplished using the transistor circuits Q1/Q2 and Q3/Q4. U23C and U23D provide logic level 1 whenever the respective ROM is being selected by the decoder. This high level causes current to flow through the base resistor (R55 or R52) turning ON the npn transistor (Q1 or Q3). The collector is pulled near ground which causes the base of the pnp transistor (Q2 or Q4) to turn ON. This causes the ROM power pin to be pulled up near 5 Volts. For full TTL-CMOS compatibility, bus pullups (U30) are used on the outputs of these ROMS.

RANDOM ACCESS MEMORY. The RAM for the main computer system is comprised of 8 5114's, each of which organized as 4 bits by 1024. These are arranged in pairs, one taking the high nibble and one taking the low nibble. The addressing of these pairs is done with another SA2995 decoder (U26), coupled with a READ-or-WRITE signal from gate U23A. This strange configuration is brought about by the fact that the 5114 memories are unlike most microprocessor compatible memories; i.e. they have no READ input. To READ a 5114, one merely has to select it and not WRITE to it. To WRITE to a 5114, one must signal WRITE to it before one selects it. Hence, the select signals for these chips must arrive during the processor READ cycle or WRITE cycle.

As a final note, both the NAND gate and the decoder must be fast in order for this circuit to work. The response time of the circuit is the sum of the NAND, the decoder, and the RAM itself. Since the processor takes 3/2 cycle (400 ns each) the response of this circuit must be under 600 nanoseconds. The 5114 responds in 400 nanoseconds and so the NAND and decoder must total less than 200 nanoseconds (worst case).

SERIAL CONTROL REGISTER. The main processor controls the operations of the Analog, Filter and IO boards using a serial protocol in order to minimize the number of interboard connections needed. U4 is a 4034 register which is employed to latch data from the processor for the other boards. Software actually serializes the data and presents it to the port.

BOOM STATUS INPUTS. Microswitch closures from the boom units are input to the microprocessor via U18 which is another 4034 register. These inputs are not filtered in hardware (which saves some components) and are expected to be software de-bounced.

BOOM MOTOR CONTROL. The two boom motor relays are contained in the power supply unit of the instrument (for height reasons). The control lines for those relays consist of four lines, one wire for each coil; i.e. an ON COIL and OFF COIL for two relays. Each control wire uses an inverter (for current), a base resistor and a transistor. A logic level 1 from the port to any of these four wires energizes the respective relay coil. At reset, the 4034 is tri-stated so that the 100K resistors will guarantee the motor coils will be off.

COMMAND SHIFT REGISTER. U24 and U27 are a pair of 4094 shift registers each having 8 bits. The resulting 16-bit register is clocked into the register by the falling edge of "COMMAND CLOCK" (after it has been filtered etc). The COMMAND ENVELOPE signals an interrupt to the processor all the time the command is being shifted into the register. Software can acknowledge the interrupt when shifting begins but must wait until the shifting ends to receive the data.

TELEMETRY SHIFT REGISTER. U31 and U32 form a 16-bit telemetry register using a pair of 4021 parallel-to-serial converters. Each is clocked out by the rising edge of GATED SHIFT CLOCK. The very first bit is therefore valid only until that first rising edge.

LEPA SHIFT REGISTER. The main processor sends reduced magnetometer data to the LEPA instrument via a 16-bit register (U19 and U20) using exactly the same protocol as the telemetry shift register. The LEPA instrument provides the pulses when needed to shift data out. Status of the register is provided by the main processor using the MAG BUSY signal (U34 and Q6).

SUN PULSE COUNTER. Once per spin while the spacecraft is in sunlight, a sun pulse will occur. Each pulse toggles a flip-flop (U9B) which can be read in the most significant bit of U12. The software defines a sun pulse as a change in the polarity of the flip-flop.

BURST CONTROL. The Burst computer can be reset using the "BRESET" output of U34. This is useful for stopping whatever the Burst computer was told to do, errant commanding, crashed programs, or whatever.

THE MAIN MULTIPLEXOR. The multiplexor circuit starts with a register to hold the multiplexor quantity (U13). Some of these bits go to another part of the multiplexor which is on the IO board (in a clean area). Two eight line multiplexors (Harris 508A's) are on the CPU board (U8 and U21). U21 is used for selecting analog monitors while U8 is used to select between the offboard multiplexor outputs. The output signal is held by a .01 microfarad capacitor (C8) while U8 is disabled (holding pin 2 low).

The signal is amplified by op-amp U16B since the multiplexors have a large impedance over temperature which would contribute to signal noise. Also, the input to the AM6112 is a very low impedance to the internal DAC. Pretty hefty spikes come out of the AM6112 input pin and the opamp takes care of these.

Another amplifier is needed to buffer the analog housekeeping values which pass through U21. Something that isn't apparent when one first looks at the U8 to C8 connection is that it has a very low impedance (maybe 200 to 1000 Ohms). Any input to U8 must be buffered by an amplifier. This is true of any science quantity already, but the analog monitors are simply resistors and such. U16c buffers the signals from those monitors before going into U8. (It is amazing to me that the ISEE instrument had no such buffer.)

ANALOG TO DIGITAL CONVERSION. The analog to digital conversion circuit is comprised of three parts, namely, the A/D itself, a clock, and a 5 Volt reference. The A/D converter is a microprocessor buss compatible device made by Advanced Micro Devices (the AM6112). It can digitize a 12-bit quantity in as little as 4 microseconds (according to the company promises) but is in this circuit set to convert in 10.0 microseconds. The flip-flop (U46A) divides the processor clock by 2 in order to clock the 6112. Each clock period is 800 nanoseconds and a conversion requires 12.5 cycles.

Initially, the A/D timing was a simple delay loop in the conversion software. After the prototype was completely assembled and closed up against outside noise, it was found that the A/D was missing codes. Testing proved that the 6112 was sensitive to the operation of the CPU, even though the manufacturer claims this isn't true. With the addition of the RC interrupt circuit, conversions were greatly improved.

Finally, the reference circuit for the 6112 converter is composed of a Precision Monolithics REF-02 (U7) followed by an op amp (U16A). One may notice if one reads the information on the 6112 that it has an internal reference circuit. While this is true, it has been shown to be rather rad-soft; i.e. the reference voltage drifts rather badly after radiation. The REF-02 doesn't have such a problem in radiation. Variable resistor R70 is provided to trim the reference circuit so that U6 pin 20 is

2.5000 Volts.

SUMMARY OF MAIN COMPUTER I/O. Below is a short summary of the input and output ports decoded on the MAIN processor buss.

Analog to Digital Converter Control (STA 5001H)

```
 .--------.
 !.....MMB!
 '--------'
       !!`-----   1 = 2'S COMPLEMENT OUTPUT, 0 = OFFSET BINARY
       !`-----    MODE = 00 : WRITE STARTS CONVERIONS
       `------           01 : READ STARTS CONVERSIONS
                         10 : READ STARTS CONVERSIONS
                         11 : WRITE STARTS CONVERSIONS
```

Analog to Digital Converter Read (LHLD 5000H)

```
 .--------.
 !LLLLLLLL! : 5000H --- LOW BYTE OF CONVERSION
 +--------+
 !....HHHH! : 5001H --- HIGH 4 BITS
 '--------'
```

Multiplexor Control (OUT 0E0H)

```
 .--------.
 !TAAAAAAA!
 '--------'
  !!!!!!!`----- A is the MAIN multiplexor address
  !`----------- described by Table 1-1
  !
  `------------ T is the TRACK/HOLD signal (0 = TRACK)
```

Command Input Register (LHLD 0AFFFH)

```
 .--------.
 !LLLLLLLL! : AFFFH --- LOW BYTE OF THE COMMAND
 +--------+
 !HHHHHHHH! : B000H --- HIGH BYTE OF THE COMMAND
 '--------'
```

Telemetry Output Register (SHLD 0AFFFH)

```
----------------------------------------
 .--------.
 !LLLLLLLL! : AFFFH --- LOW BYTE OF THE TELEMETRY WORD
 +--------+
 !HHHHHHHH! : B000H --- HIGH BYTE OF THE TELEMETRY WORD
 '--------'
```

Deployment Unit Microswitches (IN 90H)

```
---------------------------------------
 .--------.
 !22221111!
 '--------'
  !!!!!!!`-----      BOOM 1 LEFT COVER        (0 = COVER ON)
  !!!!!!`-----       BOOM 1 TURNS COUNTER
  !!!!!`-----        BOOM 1 RIGHT COVER       (0 = COVER ON)
  !!!!`-----         BOOM 1 END-OF-WIRE       (0 = END-OF-WIRE)
  !!!`-----          BOOM 2 LEFT COVER        (0 = COVER ON)
  !!`-----           BOOM 2 TURNS COUNTER
  !`-----            BOOM 2 RIGHT COVER       (0 = COVER ON)
  `-----             BOOM 2 END-OF-WIRE       (0 = END-OF-WIRE)
```

General Status Input (IN 80H)

```
------------------------------
 .--------.
 !S...KKKK!
 '--------'
  !   !!!`-----      KELLEY GAIN BIT (LSB)
  !   !!`-----       KELLEY GAIN BIT
  !   !`-----        KELLEY GAIN BIT
  !   `-----         KELLEY GAIN BIT (MSB)
  `-----             SUN PULSE INDICATOR
```

General Control Output (OUT C0H)

```
--------------------------------
 .--------.
 !KLBWMMMM!
 '--------'
  !!!!!!!`-----      MOTOR 1 RELAY OFF COIL
  !!!!!!`-----       MOTOR 1 RELAY ON  COIL
  !!!!!`-----        MOTOR 2 RELAY OFF COIL
  !!!!`-----         MOTOR 2 RELAY ON COIL
  !!!`-----          WATCHDOG RESET CIRCUIT (1=RESET)
  !!`-----           BURST COMPUTER RESET (0 = RESET, 1=RUN)
  !`-----            LEPA REGISTER LOAD STATUS (1 = LOADING)
  `-----             KELLEY GAIN CHANGE STOBE (RISING EDGE)
```

Serial Control Output (OUT 0F3H)

```
 .--------.
 !R.ZWFACD!
 '--------'
  !!!!!!!`----- D is the data (positive polarity)
  !!!!!!`-----  C is the clock (rising edge)
  !!!!!`-----   A is the Analog board strobe (active high)
  !!!!`-----    F is the Filter board strobe (active high)
  !!!`-----     W is the -WR signal to the 7528 DACs (active low)
  !!`-----      Z is the A/-B signal to the 7528 DACs (1=A side DAC)
  !`-----
  `-----         R is the RC interrupt circuit control (1 = interrupt)
```

LEPA Instrument Communication (OUT 90H, OUT D0H)

```
 .--------.
 !LLLLLLLL! : 090H --- LEPA DATA REGISTER
 '--------'

 .--------.
 !HHHHHHHH! : 0D0H --- LEPA MODE REGISTER
 '--------'
```

3.5 The Burst Processor Board

The Burst computer system is a slave of the MAIN processor and is dedicated to the singular task of high frequency sampling and storage of field data. Equipped with its own multiplexor, analog to digital converter and a very large memory, this processor non-deterministically records data in commanded formats at frequencies of up to 60 KHz. The following paragraphs describe the subcircuits of the Burst system.

THE PROCESSOR. The Burst processor is the same computer used by the MAIN system, namely a Sandia SA3000, running at 5 MHz input frequency just like the MAIN system. And similar to that system, the low address pins are latched by an 8-bit latch (U19) on the high level of ALE (see the MAIN PROCESSOR description).

INTERRUPTS. The BURST system has two types of interrupts: 1) the BCMD line from the MAIN system and 2) the word rate clock interrupts (2 KHz) from the telemetry system. The BCMD line is the higher priority of the two and is used to signal the beginning if some communication from the MAIN processor. The 2 KHz interrupt is used to sample data synchronous with the telemetry system whenever frequencies of the word rate or less are requested.

ADDRESS DECODING. The BURST system devices are distinguished from each other by U25, a 3-to-8 line decoder the SA2995. The eight decoded outputs operate as 0, 1000H, 2000H,

system operates on ANY address between 8000H and FFFFH so that it can be activated at the same time as any other device (see the memory circuit below).

ROM/RAM. The Read Only Memory for the BURST computer is contained in a single 4096x8 chip, a Raytheon 29673SMB (U26). A power controller circuit similar to those on the MAIN system is used here (U24A, Q12 and Q13).

The Random Access Memory for the BURST system consists of 4 5114's (U27 thru U30) which together comprise a total of 2048 bytes of memory and are addressed as 1000H to 17FFH. The RAM is selected in a manner similar to that used in the MAIN system.

KELLEY GAIN CONTROL. The Kelley automatic gain control circuit has 4-bits of digital information which describe its current gain state. These bits are input to the BURST system by the 8-bit register U36 (four bits unused).

THE BURST MULTIPLEXOR. The BURST system multiplexor address is controlled by an 8-bit register (U37) which provides 4 bits to the multiplexor located elsewhere and 3 bits of local control (1 bit is not used). See the BURST MUX address table.

The way in which data is written to this latch dramatically differs from normal buss operations. Instead of responding to a write signal, this latch grabs bits from the low address buss when the A/D low byte is read. The facilitates high speed sampling which is described a little bit later.

ANALOG TO DIGITAL CONVERSION. The BURST A/D circuitry is quite a bit more complicated than the equivalent circuit in the

MAIN system, basically resulting from the very high sampling rate requirement. There are 3 major features added to this circuit as compared to the MAIN: 1) a high speed sample and hold is used, 2) automatic gain control is accomplished in hardware, and 3) most of this circuit can be powered-down.

At the heart of all this is, of course, the A/D itself --- an AM6112 (U21) which is the same device used in the MAIN system. On its digital side are two devices namely a buss tranceiver (U20) and a 4050 (U31) which isolate the AM6112 from the processor buss when the A/D is turned off. This is a nessesary step since the 6112 diode clamps its input pins when it has no power. The 6112 is clocked by BADCLK, a signal which comes from the MAIN system and is the Burst CPU clock out divided by two.

BURST AUTO-GAIN CIRCUIT. The analog signal to digitize comes into the board both in low and high gain forms, just as in the MAIN system. In the BURST system, however, a third line called the "GAIN DECISION" is provided. This is a digital signal which indicates whether high gain is in range of the A/D or not.

This gain decision is latched by the flip-flop U17B when clocked by the timing circuit (described later). The output of this latch determines which of the two input signals to choose for digitization by controlling the multiplexor U33.

The output of this high/low gain multiplexor is connected to a fast (5 microsecond) sample and hold, the Precision Monolithics SMP-11. The sample/hold timing is provided once again by the timing circuit described later. The value of the "hold" capacitor, 5000 pF, was chosen for the fact that the SMP11 is internally trimmed for that value. Hence, no offsets will result in the output signal.

To indicate which gain state was used, the gain bit is jammed onto the processor buss when the high order bits of the 6112 are read. This is accomplished with U43B which is high when the A/D high byte is read. This gates the gain bit, driven by U39A, onto the processor buss by multiplexor U66, replacing the 4th most significant bit if the byte. This results in a 12-bit sample plus gain packed neatly in 13 contiguous bits.

Finally, the timing for all of this is accomplished using a single 4015 (shift register U38), configured as a unary counter. Reset by the reading of the A/D low byte, it is incremented with every A/D clock cycle. On the 6th thru 14th clocks, the CPU is stopped by the nand gate U2D. This keeps digital noise from the A/D converter while converting. On the 10th clock, this timer strobes the new gain decision into U17B (see above) and the old gain decision into U17A. Gain bits require this double buffering in order to track the pipeline correctly. On the 14th clock (A/D is done), the sample and hold is instructed to sample again.

A/D POWER DOWN. The BURST analog to digital conversion circuitry described above involves a number of high power devices, specifically, the 6112 A/D, the 4602 quad opamp, the SMP11 sample and hold, and the REF02 reference. Since during the long playback periods these parts will not be needed, the power to these devices has been routed through a pair of relays K7 and

K8. Relay K7 handles the +/- 5 Volts to the 6112 and the 4050 isolation buffer. Relay K8 switches OFF the +/- 12 Volts to the circuit.

THE MEMORY UNIT. The memory circuit for the BURST system is a rather autonomous unit which functions more like a digital tape recorder than a typical memory. Complete with its own memory address register (MAR) and bank wrap-around logic, this memory unit features total-dose radiation tolerance, protection against cosmic ray induced power surges, individual bank power-down, and a special "autowrite" mode which effectively doubles the memory transfer rate.

The unit acts like a tape recorder for two reasons. First, whenever reading or writing, the memory unit uses its own memory address register (U3, U6, U9, U11 and U13) rather than one supplied by the processor address buss. This allows the central processor to spend its time sampling rather than incrementing an address which is larger than the processor is designed for. The MAR keeps track of the address for the cpu and increments itself whenever either a read or a write to the memory is done.

Second, when the MAR reaches a specified end address, the next read or write causes the start address to be jammed into the MAR. This is accomplished by the upper address comparison (U7 and U8A) combined with the detection of an all 1's condition in the rest of the address (U8B, U15D and U2C). Thus, the "tape" automatically rewinds to the beginning and starts recording again.

Setting a particular address into the MAR is achieved by writing directly to the lower 12 address bits (the counters are on the processor buss as output ports). The upper six bits have to be written to the start address latch (U1) and then strobed into the counters U3 and U6 by toggling bit 6 (2nd MSB) of the start address port while doing one memory access. Note: the 40163 counters accept input data only when clocked while their load pins are down.

The decoder U16 is used to select which of the six memory banks the MAR is addressing. It is delected whenever the MAR is written into by virtue of a high level on pin 5 (2nd MSB of the start address port). This keeps accesses to the MAR from writing anything into the memory banks.

One peculiar feature of the memory unit is referred to as its "autowrite" mode. Initiated by a high level on the MSB of the start address port, this mode causes the memory unit to perform a WRITE operation when the processor buss says to READ. This seemingly useless configuration actually has a tremendous advantage in high speed data recording of data from another buss device like an a/d converter. The important thing to notice is that the selection of the memory unit can be overlapped with any other device on the processor buss. In the autowrite mode, if one addresses both the memory and another device and one reads from the device, the memory unit will catch whatever is read from that device! With an extremely fast a/d, this can effectively double the memory transfer capability of the system.

The swapping of the read and write lines is accomplished using a multiplexor and 4049 drivers (U14, U15A, U15B). Note that it is not a complete swap since writing to the memory in the autowrite mode still works --- the memory won't try to read.

The memory banks are protected from SCR latchups (caused by cosmic radiation) by a current foldback circuit on their +5V power. Consisting of Q10, Q11, and some discrete components, this circuit removes power from the entire bank of memory as soon as the SCR power surge is detected and long before the RAM device burns out. Measurements of this response show it to go into current limiting when the load exceeds 800 mW (20 Ohms at 4 Volts). Radiation tests with the RAM devices show them to take 4 times this amount of power in the latched condition. The entire memory bank requires only one tenth this amount of power when operating normally (pre-radiation).

A total dose radiation effect which is expected to occur with the type of memories used is a growth in power consumption. Radiation tests performed on these devices have shown a factor of 100 growth in power consumption before the devices actually fail to work. As a counter measure, the banks can be individually turned OFF in case one or all become excessively power hungry. If all banks are extremely hungry, the decision might be to turn all of them OFF and use the processor's 1K memory.

Bank power is controlled by relays K1 thru K6 in a Set/Reset scheme using K9. This is similar to the arrangement on the analog board. Basically, to power OFF a bank, e.g. bank 1, one must apply current though the right coil or relay K1. To do this, a digital value is applied to the base resistor R1 which turns on Q1. This runs +5 to -7 Volts through the K1 reset coil. Similarly, to turn ON a bank, one simply flips Relay K9 before powering relay K1. The current then flows through the set coil instead of the reset coil.

BURST SAMPLE/HOLD/CONVERT PIPELINE. Unlike the MAIN computer system which is constrained to sample data only as fast as it can telemeter it, the BURST computer is designed for high speed data collection and playback. The BURST analog to digital conversion circuitry meets the high speed requirement by having its own multiplexor, a/d converter, AND a fast sample and hold (the SMP 11).

While the circuit design is capable of sampling rates in excess of 150 Khz, both the substitution of radiation tolerant devices and the slowing of the converter clock pulses has resulted in the current 60 KHz conversion frequency maximum. For example, the specifications for the AMD6112 converter say it should be capable of converting in 3 microseconds but the manufacturer will deliver only parts spec'd at 8 microseconds. The closest clock frequency which we have available is 1.25 MHz and this results in a convert time of 10.0 microseconds.

The procedure for sampling a qty in general is 1)address the quantity on the multiplexor, 2) set the sample-hold to SAMPLE for the required time, 3) set the sample-hold to HOLD, 4) start the a/d converter and wait until the it's ready, 5) read out the converted data, and 6) store the result.

In the BURST processor, it takes a single instruction per sample to accomplish this series of events. In pipeline fashion, a new value can be addressed on the multiplexor and have its gain decision prepared while the prior value is being converted. To do this, the BURST software configures the a/d to start a new conversion as soon as the last converted value is read out (see the specifications on the AMD6112). BURST software also configures the MEMORY circuit to AUTOWRITE, which means it can take data directly off the buss when the a/d converter gives it to the cpu.

The process of converting a value is shown in figure AD-1. Any read of the a/d converter causes four things to happen. First, the sample/hold is placed in the HOLD condition. At the same time, the a/d converter puts its last result on the buss for the cpu and begins converting the quantity which the sample/hold is now holding. Lastly, the multiplexor register will take a new value described by some of the address bits used to read the a/d converter.

At the end of the 12.5th cycle after the a/d read, the conversion will complete. At the 14th cycle, an automatic timer will force the sample/hold back into the SAMPLE mode where it will track the value (we just set) on the multiplexor. This returns the circuit to its original state, ready to convert another quantity.

The net effect of the timing circuit is that the gain decision is latched after 8 microseconds from the start and the sample-hold is released to track the new value after 11.2 microseconds. If the sampling loop is 16.0 usec, then the hold time is 4.8 usec.

It is important to note that it actually takes three READs of the a/d converter to get the converted value into memory. After the first read, the sample/hold will acquire it. The second read will cause it to be converted but it will only be inside the converter. The third READ actually obtains the value and stores it.

INSTR.	CYC	MUX	S/H	A/D	MEMORY
LHLD Q1	0	--	HOLD --	CONVERT --	STORE --
	1	Q1	HOLD --	CONVERT --	
	14	Q1	SAMPLE --	FINISHED	
LHLD Q2	0	Q1	HOLD Q1	CONVERT Q1	STORE --
	1	Q2	HOLD Q1	CONVERT Q1	
	14	Q2	SAMPLE Q1	FINISHED	
LHLD Q3	0	Q2	HOLD Q2	CONVERT Q2	STORE Q1
	1	Q3	HOLD Q2	CONVERT Q2	
	14	Q3	SAMPLE Q2	FINISHED	

Figure 21. Burst A/D Pipeline Operation

SUMMARY OF BURST COMPUTER I/O. Described below are the input and output ports available to the Burst computer system:

Analog to Digital Converter Control (STA 3001H)

```
 .--------.
 !.....MMB!
 '--------'
      !!`-----   1 = 2'S COMPLEMENT OUTPUT, 0 = OFFSET BINARY
      !`-----    MODE = 00 : WRITE STARTS CONVERIONS
      `------           01 : READ STARTS CONVERSIONS
                        10 : READ STARTS CONVERSIONS
                        11 : WRITE STARTS CONVERSIONS
```

Analog to Digital Converter Read (LHLD 30xxH)

```
 .--------.
 !LLLLLLLL! : 3000H --- LOW BYTE OF CONVERSION
 +--------+
 !...GHHHH! : 3001H --- HIGH 4 BITS PLUS GAIN BIT
 '--------'
```

Multiplexor Control (LHLD 30xxH)

```
 .--------.
 !AAAAAAA.!
 '--------'
  !!!!!!`------ A is the 7-bit BURST multiplexor address
  `------------ described by Table 1-2
```

High Memory Address Register (OUT 70H)

```
 .--------.
 !ALMMMMMM!
 '--------'
  !!!!!!!`-----      Upper Memory Address bit A12
  !!`-----           Upper Memory Address bit A17
  !`-----            Load upper MAR (1=load)
  `-----             Autowrite control (1=autowrite)
```

Low Memory Address Registers

```
 .--------.
 !LLLLLLLL!    :    0CFFFH
 +--------+
 !....HHHH!    :    0D000H
 '--------'
```

Relay Control Bits (OUT 20H)

```
 .--------.
 !AABBBBBB!
 '--------'
  !!!!!!!`-----      Bank 0 Relay
  !!!!!!`-----       Bank 1 Relay
  !!!!!`-----        Bank 2 Relay
  !!!!`-----         Bank 3 Relay
  !!!`-----          Bank 4 Relay
  !!`-----           Bank 5 Relay
  !`-----            A/D +5V Relay
  `-----             A/D +12V Relay
```

General Control Bits (OUT 63H)

```
 .--------.
 !..R..EEE!
 '--------'
    !  !!`-----      End Memory address (A15)
    !  !`-----       End Memory address (A16)
    !  `-----        End Memory address (A17)
    `-----           Set/Reset Relay Control for above relays
```

3.6 The Memory Boards

The Burst memory itself (not the control circuitry) is located on two memory cards, each of which contains 3 banks of 32 KBytes each. Originally there had been 4 banks on each board, but we couldn't fit in that many chips. Hence, the total memory capacity of the Burst memory is 6 X 32 K or 192 KBytes.

As was mentioned earlier is the Burst CPU description, each memory bank can be individually powered. Both transient and total-dose radiation effects are expected to be encountered, particularly in view of the fact that the memory chips used are not "radiation tolerant" in the full definition of the term. While several rad-hard memories exist, they are either outrageously expensive or did not have sufficient density to be useful. These memories, made by Integrated Device Technologies, have survived more than 10000 rads which is the expected "box-level" total dose for the instrument. Assuming the shielding which the box provides, we shouldn't have too much radiation problems with them.

Nevertheless, each bank of memory is separated by a buss tranceiver and can be turned off completely by the Burst computer system. The trick to doing this is in isolating the memory address and control signals from the chips while powered down. (One must do this with most CMOS since the chips will take power from their input pins if their input pins have higher voltage than their power pins.)

Each memory bank's address and control lines are buffered through

CD4049 and CD4050 packages which are powered by the bank's power. When the bank power goes down, the 4049 and 4050 devices cannot drive the private bank buss. Unlike most CMOS devices, CD4049 and 4050 packages are made to accept input signals higher than their power pins.

Within a memory bank, one of the 16 memory chips is selected by a pair of 3-to-8 line decoders (SA2995). These are only enabled if the Bank Enable line is low (this signal comes from the Burst CPU board).

Six jumpers on each memory board allow for the address definition of the banks on that board. For example, the first bank may be configured as either bank 0 or 3, the second bank as either 1 or 4, and the third bank as either 2 or 5. For each bank, one must make two jumpers: 1) for power and 2) for the digital bank enable line. The standard way to configure the two boards is Banks 0, 1 and 2 together, and banks 3, 4 and 5 together.

3.7 The Power Converter

High level commands are pulsed commands which have 28 Volts on them (and considerable power capability). These are used in the instrument for switching on and off the power converter relays. Four commands (1 wire each) are used to control the MAIN and BACKUP power converters. The converters can both be on at the same time without damage to the converter.

1. MAIN CONVERTER POWER ON (CONNECTOR J8 PIN 18)
2. MAIN CONVERTER POWER OFF (CONNECTOR J8 PIN 17)
3. BACKUP CONVERTER POWER ON (CONNECTOR J8 PIN 5)
4. BACKUP CONVERTER POWER OFF (CONNECTOR J8 PIN 6)

The immediate effect of these four commands are as follows:

1. POWER ON/OFF -LP (CONNECTOR J8 PIN 4) IS SET TO BILEVEL 1
2. POWER ON/OFF -LP (CONNECTOR J8 PIN 4) IS SET TO BILEVEL 0
3. POWER ON/OFF -MAG (CONNECTOR J8 PIN 10) IS SET TO BILEVEL 1
4. POWER ON/OFF -MAG (CONNECTOR J8 PIN 10) IS SET TO BILEVEL 0

3.8 Inter-Processor Communications

The two computer systems communicate over two wires labelled BCMD and BRDY. BCMD is controlled by the MAIN processor and BRDY is controlled by the BURST. Both commands and data are transferred between the systems in 16-bit serial strings using BCMD and BRDY.

The protocol employed to send data from the MAIN to the BURST is the same as that used to send data the other way, so one need only describe one transmission to describe both. Initially both the BCMD and BRDY lines are low. From the MAIN cpu point of view, these names mean "BURST COMMAND" and "BURST READY". To send data from the MAIN processor to the BURST, the MAIN cpu sets BCMD to 1 which requests the BURST's attention. When the BURST is ready to accept the data, it sets BRDY to 1. The falling edge of BCMD is used as the starting time after which the MAIN cpu presents data bits on the BCMD line every 45 cycles. The BURST samples the BCMD line every 45 cycles and stores the bits in its software shift register. Finally, the MAIN processor restores the BCMD line low as does the BURST processor its status line BRDY.

These transmissions have a maximum operating frequency of approximately 56 Kbps with both cpu's using 5.0 MHz crystals. (Each transmission takes around 290 microseconds.)

Contention between the two processors for use of these lines is possible since both processors may want to send information to the other at the same time. For example, the BURST may be playing data back to the MAIN processor at the same time as the MAIN gets a command from the ground with the BURST cpu as the destination. The situation in which both cpu's raise their request lines at the same time is handled by the BURST relenting and receiving data instead of sending it. (The BURST will immediately try again to send its data at the conclusion of the MAIN-to-BURST transmission unless explicitly stopped by command.)

Commands are distinguished from data only by context; i.e. after RESET to the Burst processor it enters the command state which expects to get command data. If the command has data following it, then the Burst will interpret the next transmissions as data. When the data portion is finished, it will again interpret incoming data as commands. This context sensitive protocol has one minor problem, namely that all command-data sequences from the MAIN cpu to the BURST must supply all of the data required. If the MAIN tries to stop the last command and start a new one before all of the data is sent, the BURST will confuse the new commands as if they were data. To stop a command-data sequence one must RESET the BURST processor.


```
0119 6F        0200    MOV   L,A
```

```
--------------------------------------------------------------------

                             .-----------------.    .--------.--------.-
BURST BRDY          ______!                  !____X________X________X_

                                   .------------------------------------
BURST BCMD          _______________!

                                   !          !    !    !   !
                                   !----A-----!-B--!    !   !
                                   !          !----C----!   !
                                              !------D------!

------------------------------------------------------------------
```

Figure 22. BURST to MAIN communication timing

	MIN	MAX
A : BURST RECOGNIZES MAIN READY	45 cyc	69
B : BURST FIRST BIT TRANSITION	31	31
C : TRANSITION TO TRANSITION	45	45
D : MAIN BIT SAMPLE FROM START	32	60

Table 11. Main Multiplexor Quantities

ADDR	0x LOW GAIN	1x HI GAIN	2x	3x	4x
x0	BZ	BZ	MTR1	V12X50	V34X50
x1	BY	BY	MTR2	V12X50	V34X50
x2	BX	BX	LEN1	V12X50	V34X50
x3	V3	V3	LEN2	V12X50	V34X50
x4	V2/RI2	V2/RI2	TMP1	V12X50	V34X50
x5	V1/SC	V1/SC	TMP2	V12X50	V34X50
x6	V1	V1	TMP3	V12X50	V34X50
x7	AGC	AGC**	TMP4	V12X50	V34X50
x8	V12/RI1	V12/RI1	MTR1	V12X50	V34X50
x9	F3	F3	MTR2	V12X50	V34X50
xA	F2	F2	LEN1	V12X50	V34X50
xB	F1	F1	LEN2	V12X50	V34X50
xC	V4	V4	TMP1	V12X50	V34X50
xD	AGC*	AGC*	TMP2	V12X50	V34X50
xE	V12/RI1*	V12/RI1*	TMP3	V12X50	V34X50
xF	V34	V34	TMP4	V12X50	V34X50

* : UNFILTERED QUANTITIES
**: NON-SENSE QUANTITY

Table 12. Burst Multiplexor Quantities

ADDR	BQTY
x0	BZFAST
x1	BXFAST
x2	BYFAST
x3	V3
x4	V4
x5	V34
x6	V34 AC
x7	V1/SC
x8	V12 AC
x9	V2/RI2
xA	V1
xB	V12
xC	DIRECT
xD	AGC*
xE	GUARD 1
xF	STUB 1

* : UNFILTERED QUANTITIES

x : 0 -- AUTOMATIC GAIN
1 -- FORCED HIGH GAIN
2 -- FORCED LOW GAIN
3,4,7 -- UNDEFINED
5 -- GROUND
6 -- BURST MEM+5V MONITOR

Table 13. Analog to Digital Conversion Times

	MAIN	BURST	UNITS
A/D CLOCK FREQUENCY :	1.25	1.25	MHz
A/D CONVERT TIME : (12.5 clock cycles)	10.0	10.0	microsec
SAMPLE TIME REQD :			
(10V step to .02%)	153(1)	4.8	microsec
(10V step to .10%)	---	3.5	
MINIMUM A/D TIME :	163.0	15.0	microsec
ACTUAL A/D TIME :	165	16.0	microsec

(1) The impedance of a Harris 508A (1.8K Ohms max) and a .01 microfarad capacitor have an RC of 18 microseconds. Twelve bit accuracy translates into 8.5 RC periods or 153 microseconds.

4. Contributing Engineers and Scientists

The CRRES Langmuir Probe and Fluxgate Magnetometer required the efforts of a number of engineers and scientists at the University of California, Regis College in Massachusetts, the Air Force Geophysics Lab in Massachusetts and Analytyx Corporation in New Hampshire. The following is a list of those people and their functions:

Name	Org	Functions
Peter R Harvey	UCB	Project Management at UCB Flight Hardware and Software GSE Hardware and Software
Dr. John Wygant	UCB	Project Scientist at UCB Flight Sphere Design Analog Section Design
Dr. Forrest Mozer	UCB	Co-Investigator for 701-14A
Dr. David Pankow	UCB	Boom Deployment Systems Engineering
Dr. Roy Torbert	UCSD	Flight Filtering section
Peter Anderson	Regis	Parts Engineering Test Management
Bill Sullivan	AFGL	Project Management at AFGL
Dr. Howard Singer	AFGL	Scientist for Fluxgate Magnetometer
Dr. Michael Smiddy	AFGL	Scientist for Langmuir Probe UCB Contract Monitor
Dr. Nelson Maynard	AFGL	Principle Investigator for 701-14A
Ken Fredholm	Analytyx	Project Management at Analytyx Systems Engineering
Bob Hayes	Analytyx	Power Supply Engineering
Paul Murray	Analytyx	Main Box Mechanical Engineering

Appendix A.

CRRES FLIGHT SOFTWARE

RELEASE 2.1, 12-6-88

MAIN SOFTWARE : 12-6-88
BURST SOFTWARE : 2-1-85

PETER R. HARVEY

```
0000      *
0000      * CRRES FLIGHT PROGRAM---CONFIGURATION
0000      * WRITTEN BY PETER R HARVEY
0000      * FILE CONFIG.A
0000      *
0000      * 8085 SPECIFIC INFORMATION
0000      *
0000      PSW     EQU    6
0000      SP      EQU    6
0000      *
0000      * ROM CONFIGURATION OF PACKAGES
0000      *
0000              ORG    40H
0040      CMDTAB  DS     40H
0080      NEXT    DS     0CD0H
0D50      MAG     DS     148H
0E98      PLA     DS     170H
1008      BUR     DS     1E0H
11E8      FIT     DS     148H
1330      SAW     DS     90H
13C0      ELE     DS     2C0H
1680      LD      DS     44H
16C4      DEP     DS     120H
17E4      SWP     DS     380H
1B64      EXEC    DS     17CH
1CE0      BKG     DS     320H
2000      *
2000      * RAM CONFIGURATION
2000      *
2000      RAM     EQU    2000H
2000      RAMSIZE EQU    1000H
2000      *
2000              ORG    RAM
2000      IORAM   DS     1CH
201C      BKGRAM  DS     28H
2044      DEPRAM  DS     0CH
2050      ELERAM  DS     0A8H
20F8      *
20F8              ORG    */256+1*256
2100      MAGRAM  DS     50H    MUST BE ALIGNED ON 256
2150      PLARAM  DS     20H
2170      BURRAM  DS     30H
21A0      SAWRAM  DS     10H
21B0      FITRAM  DS     170H
2320      SPINRAM DS     170H
2490      MATRAM  DS     10H
24A0      EXERAM  DS     20H
24C0      SWPRAM  DS     460H
2920      LDRAM   EQU    $
```

```
2920    STACK  EQU    RAM+RAMSIZE-1
2920    *
2920           COM    CMDTAB
2920           COM    NEXT
2920           COM    BKG
2920           COM    ELE
2920           COM    MAG
2920           COM    PLA
2920           COM    DEP
2920           COM    LD
2920           COM    EXEC
2920           COM    BUR
2920           COM    SAW
2920           COM    SWP
2920           COM    FIT
2920    *
2920           COM    IORAM
2920           COM    BKGRAM
2920           COM    ELERAM
2920           COM    MAGRAM
2920           COM    PLARAM
2920           COM    DEPRAM
2920           COM    BURRAM
2920           COM    SAWRAM
2920           COM    SWPRAM
2920           COM    FITRAM
2920           COM    SPINRAM
2920           COM    MATRAM
2920           COM    LDRAM
2920           COM    EXERAM
```

```
0000                 *
0000                 * CRRES FLIGHT SOFTWARE --- UTILITIES
0000                 * WRITTEN BY PETER R HARVEY
0000                 *
0000                         ORG    1*8
0008 36 00           ZERO    MVI    M,0
000A 23                      INX    H
000B 0D                      DCR    C
000C C2 08 00                JNZ    ZERO
000F                 *
000F                         ORG    2*8-1
000F C8              CP1     RZ     .
0010 1A              COPY    LDAX   D
0011 77                      MOV    M,A
0012 13                      INX    D
0013 23                      INX    H
0014 0D                      DCR    C
0015 C3 0F 00                JMP    CP1
0018                 *
0018                         ORG    3*8
0018 85              REF     ADD    L
0019 6F                      MOV    L,A
001A D0                      RNC    .
001B 24                      INR    H
001C B7                      ORA    A
001D C9                      RET    .
001E                 *
001E                 UTIL    EQU    NEXT
001E                         ORG    UTIL
0080                         COM    ZERO
0080                         COM    COPY
0080                         COM    REF
0080                         COM    UNARY
0080                         COM    NEG16
0080                         COM    MARK
0080                 *
0080                 * CONVERT [HL] INTO 2**A-1
0080                 *
0080 21 01 00        UNARY   LXI    H,1
0083 E6 0F                   ANI    15
0085 C8              UNA1    RZ
0086 29                      DAD    H
0087 3D                      DCR    A
0088 C3 85 00                JMP    UNA1
008B                 *
008B                 * NEGATE [HL]
008B                 *
008B 7C              NEG16   MOV    A,H
008C 2F                      CMA
```

```
008D 67              MOV   H,A
008E 7D              MOV   A,L
008F 2F              CMA
0090 6F              MOV   L,A
0091 23              INX   H
0092 C9              RET
0093            *
0093            * OUTPUT (HL) TO THE DIAGNOSTIC LEDS
0093            *
0093 7C         MARK MOV   A,H
0094 D3 01           OUT   1
0096 7D              MOV   A,L
0097 D3 00           OUT   0
0099 C9              RET
009A            NEXT EQU   $
009A                 COM   NEXT
```

```
0000  *
0000  * CRRES FLIGHT SOFTWARE---INPUT/OUTPUT DRIVER SECTION
0000  * WRITTEN BY PETER R HARVEY
0000  *
0000  * FILE IO.A
0000  *
0000  * IO SYSTEM DESCRIPTION
0000  *
0000  SERIAL  EQU   0F3H    SERIAL CONTROL OUTPUT
0000  SERCLK  EQU   2       SERIAL CLOCK
0000  ANASTB  EQU   4       ANALOG SHIFT REGISTER STROBE
0000  FILSTB  EQU   8       FILTER SHIFT REGISTER STROBE
0000  NWR     EQU   16      7528 WRITE (INVERTED)
0000  AB      EQU   32      7528  A/B CONTROL
0000  INT     EQU   128     INTERRUPT CIRCUIT
0000  *
0000  ADC     EQU   5000H   A/D CONVERTER
0000  ADCTL   EQU   ADC+1   A/D CONTROL
0000  *
0000  MUX     EQU   0E0H    MUX, TRACK/HOLD, AND POWER
0000  ADDR    EQU   07FH    MUX ADDRESS BITS
0000  HIGAIN  EQU   10H     HIGH GAIN SELECT
0000  TRACK   EQU   080H    TRACK BIT
0000  PRVAL   EQU   079H    PREFERRED RESET VALUE FOR MUX
0000  AGCF    EQU   7       KELLEY QUANTITIES ON TM MUX
0000  AGCU    EQU   13
0000  V12H    EQU   2EH     "V12 X50" QTY, V34H = V12H+1
0000  V12     EQU   0EH     V12 LOW GAIN USED IF X50 SAT.
0000  V12X50  EQU   30H
0000  V34     EQU   0FH     V34 LOW USED IF X50 SAT.
0000  V34X50  EQU   40H
0000  *
0000  COMMAND EQU   0AFFFH  COMMAND REGISTER (MEMORY MAPPED)
0000  TELEM   EQU   0AFFFH  TELEMETRY BUFFER OUTPUT
0000  *
0000  MICROS  EQU   90H     BOOM MICROSWITCHES
0000  MOTORS  EQU   0C0H    MOTOR CONTROL
0000  KLYGC   EQU   080H    KELLEY GAIN CONTROL
0000  MAGLOAD EQU   040H    MAG LOAD STATUS
0000  BRUN    EQU   20H     BURST PROCESSOR RESET/RUN
0000  WDRES   EQU   10H     WATCHDOG RESET (1=RESET)
0000  *
0000  KLYIN   EQU   080H    KELLEY GAIN BITS
0000  KLYGN   EQU   0FH
0000  SUNBIT  EQU   80H     SUN PULSE FLOP STATUS
0000  FLIGHT  EQU   10H     PROTOFLIGHT IF 1
0000  *
0000  LEPAH   EQU   0D0H    LEPA SHIFT REGISTER OUTPUT
0000  LEPAL   EQU   090H
```

```
0000            *
0000            PSW    EQU   6      8085 INFORMATION
0000            SP     EQU   6
0000            RIM    EQU   20H
0000            SIM    EQU   30H
0000            SLDE   EQU   18H
0000            SDG    EQU   80H
0000            SODEN  EQU   40H
0000            RES75  EQU   10H
0000            MSE    EQU   8
0000            DIS75  EQU   4
0000            DIS65  EQU   2
0000            DIS55  EQU   1
0000            *
0000            * RESTART 7 (CONVERSION DONE INTERRUPT)
0000            *
0000                   ORG   7*8
0038 C9                RET
0039            *
0039            * IO CONTROL ENTRY POINTS
0039            *
0039            IO     EQU   NEXT
0039                   ORG   IO
009A                   COM   IOINIT
009A                   COM   GETMASK
009A                   COM   SETMASK
009A                   COM   CMDIN
009A                   COM   TMOUT
009A                   COM   SUNSTAT
009A                   COM   BOOMSTAT
009A            *
009A                   COM   SAMPLE
009A                   COM   AGC
009A                   COM   SETBIAS
009A                   COM   SETGUARDS
009A                   COM   SETSTUBS
009A                   COM   SETVTRIM
009A                   COM   SETRELAY
009A                   COM   SETFILTER
009A                   COM   SETMUX
009A            *
009A                   COM   SETMOTOR
009A                   COM   SETKLY
009A                   COM   SETPLA
009A            *
009A                   COM   SEND
009A                   COM   RECEIVE
009A                   COM   TSTMUX
009A                   COM   RWATCH
009A                   COM   IODSC
```

```
009A                 COM    RBURST
009A           *
009A           * INITIALIZE THE INPUT/OUTPUT SO THE MODULE
009A           *  FUNCTIONS CORRECTLY.
009A           *
009A 3E 20     IOINIT MVI   A,BRUN   CLEAR MOTORS
009C 32 00 20         STA   MKBCOPY  KELLEY,RUN BURST
009F CD B5 03         CALL  RWATCH  RESET THE WATCHDOG
00A2 3E 40            MVI   A,SDOEN  RESET BCMD LINE
00A4 30               DB    SIM
00A5 00               NOP
00A6 00               NOP
00A7 00               NOP
00A8           *
00A8 97               SUB   A       RESET MOTOR 0
00A9 CD 18 03         CALL  SETMOTOR
00AC 3E 01            MVI   A,1     AND MOTOR 1
00AE CD 18 03         CALL  SETMOTOR
00B1           *
00B1 3E 3F            MVI   A,3FH  RESET THE SERIAL CTL
00B3 D3 F3            OUT   SERIAL
00B5 32 14 20         STA   MUXCOPY  CLEAR RESET TO KELLEY
00B8 3E 01            MVI   A,1     SET A/D FOR 2'S COMPLEMENT
00BA 32 01 50         STA   ADCTL
00BD           *
00BD CD FA 02         CALL  FILCLEAR
00C0 97               SUB   A       CLEAR THE RELAY BITS
00C1 32 15 20         STA   RS
00C4 32 16 20         STA   RS+1
00C7 32 17 20         STA   RS+2
00CA C3 08 02         JMP   RLYOFF
00CD           *
00CD           * SYSTEM INTERRUPT STATUS CALLS.
00CD           *
00CD 20        GETMASK DB   RIM    READ THE INTERRUPT MASK
00CE C9               RET   .
00CF 30        SETMASK DB   SIM    SET THE INTERRUPT MASK
00D0 C9               RET   .
00D1           *
00D1           * COMMAND INPUT.
00D1           *  ON EXIT: [HL]= COMMAND
00D1           *
00D1 2A FF AF  CMDIN  LHLD  COMMAND  PICK THE COMMAND REGISTER
00D4 C9               RET   .
00D5           *
00D5           * TELEMETRY OUTPUT.
00D5           *  ON ENTRY: [HL]=TELEMETRY OUTPUT
00D5           *
00D5 22 FF AF  TMOUT  SHLD  TELEM  SET THE SHIFT REGISTER
00D8 C9               RET   .
```

```
00D9                *
00D9                * SUN AND BOOM STATUS CALLS
00D9                *
00D9 DB 80          SUNSTAT IN    SUNBIT  THE SUN STATUS IS
00DB E6 80                  ANI   80H    IN THE MSB
00DD 21 02 20               LXI   H,SUNCOPY  IF SAME AS WE'VE SEEN
00E0 BE                     CMP   M      THEN JUST RETURN ZERO
00E1 77                     MOV   M,A    ELSE RECORD NEW STATUS
00E2 C9                     RET
00E3                *
00E3 DB 90          BOOMSTAT IN   MICROS  READ THE BOOM
00E5 C9                     RET   .      MICROSWITCHES.
00E6                *
00E6                * SAMPLE THE A/D CONVERTER
00E6                *  ON ENTRY: [A]=QTY TO ADDRESS ON MULTIPLEXOR
00E6                *  ON EXIT : [HL]=THE 12-BIT VALUE
00E6                *
00E6 6F             SAMPLE MOV    L,A    SAVE MUX #
00E7 3E 0D                  MVI   A,MSE+DIS75+DIS55  TURN OFF INTS (NOT SYNCS)
00E9 30                     DB    SIM
00EA                *
00EA 7D                     MOV   A,L
00EB F6 80                  ORI   TRACK  START SAMPLING
00ED D3 E0                  OUT   MUX
00EF                *
00EF 3E 19                  MVI   A,153*5/2-21/14  DELAY 153 MICROSECS
00F1 3D             SDLA    DCR   A      (14 CYCLE LOOP)
00F2 C2 F1 00               JNZ   SDLA   (21 OTHER CYCLES)
00F5 7D                     MOV   A,L    REMOVE TRACK
00F6 D3 E0                  OUT   MUX
00F8                *
00F8 3E 90                  MVI   A,NWR+INT  REQUEST INTERRUPT
00FA D3 F3                  OUT   SERIAL
00FC 3E FF                  MVI   A,0FFH  PUT OUT RESTART 7 ON BUSS
00FE D3 50                  OUT   ADC/256  WRITE TO START CONVERSION
0100 76                     HLT
0101 3E 10                  MVI   A,NWR
0103 D3 F3                  OUT   SERIAL
0105 3E 08                  MVI   A,MSE  ENABLE WORD CLOCKS
0107 30                     DB    SIM
0108                *
0108 7D                     MOV   A,L    GET MUX ADDRESS
0109 2A 00 50               LHLD  ADC    GET THE 12 BITS
010C FE 07                  CPI   AGCF   IF KELLEY, GO TO IT
010E CA 16 01               JZ    GETKLY
0111 FE 0D                  CPI   AGCU
0113 C2 20 01               JNZ   MASK
0116                *
0116 7D             GETKLY MOV    A,L    MASK THE LOW BITS
0117 E6 F0                  ANI   -1-KLYGN
```

```
0119 6F            MOV    L,A
011A DB 80         IN     KLYIN
011C E6 0F         ANI    KLYGN
011E B5            ORA    L
011F 6F            MOV    L,A
0120               *
0120 7C      MASK  MOV    A,H     MASK UPPERS
0121 E6 0F         ANI    0FH
0123 67            MOV    H,A
0124 3E F9         MVI    A,TRACK+PRVAL  RETURN TO ZERO
0126 D3 E0         OUT    MUX
0128 FB            EI
0129 C9            RET    .
012A               *
012A               * AUTOMATIC GAIN CONTROL LOGIC
012A               *  ON ENTRY: A=LOW GAIN QTY ADDRESS
012A               *  ON EXIT:  [D]=PROPER QTY TO DIGITIZE
012A               *            ZERO FLAG SET IF LOW GAIN, NZ FOR HI
012A               *
012A 57      AGC   MOV    D,A     ASSUME LOW GAIN
012B FE 07         CPI    AGCF    IF EITHER KELLEY QTY, TAKE A
012D CA 5E 01      JZ     AGC2    DUMMY SAMPLE.
0130 FE 0D         CPI    AGCU
0132 CA 5E 01      JZ     AGC2
0135 FE 10         CPI    HIGAIN  IF AUTOGAIN QTY, GO
0137 DA 4C 01      JC     AUTO
013A FE 2E         CPI    V12H    IF FORCED LOW GAIN AND NOT X50
013C DA 5B 01      JC     FLG     THEN GO
013F 11 30 0E      LXI    D,V12*256+V12X50  ELSE DO THE X50
0142 CA 48 01      JZ     X50GO
0145 11 40 0F      LXI    D,V34*256+V34X50
0148 7B      X50GO MOV    A,E
0149 C3 4F 01      JMP    AGC1
014C               *
014C F6 10   AUTO  ORI    HIGAIN  SAMPLE THE HI GAIN
014E 5F            MOV    E,A
014F CD E6 00 AGC1 CALL   SAMPLE  [HL]=SAMPLE
0152 7C            MOV    A,H     IF SAMPLE=7XX THEN IT'S
0153 FE 07         CPI    7       VERY HIGH POSITIVE.
0155 C8            RZ     .       SO USE LOW GAIN
0156 FE 08         CPI    8       IF SAMPLE=8XX THEN IT'S
0158 C8            RZ     .       VERY NEGATIVE. USE LOW
0159 53            MOV    D,E     ELSE HIGH IS OK
015A C9            RET    .
015B               *
015B E6 EF   FLG   ANI    -1-HIGAIN  REMOVE HIGAIN FROM QTY
015D 57            MOV    D,A
015E CD E6 00 AGC2 CALL   SAMPLE  AND TAKE DUMMY SAMPLE
0161 97            SUB    A       AND RETURN(0) FOR LOW GAIN
0162 C9            RET
```

```
0163               *
0163               * BIAS, STUB AND GUARD CONTROL LOGIC.
0163               *  ON ENTRY: [L]=8-BIT DAC VALUE
0163               *            [H]=BOOM NUMBER 1-4
0163               *
0163               SPHAD  EQU    0EH    BIT 1 SELECT
0163               CYLAD  EQU    07H    BIT 4 SELECT
0163               GRDAD  EQU    0BH    BIT 2 SELECT
0163               STBAD  EQU    0DH    BIT 3 SELECT
0163               NILAD  EQU    0FH    ALL OFF
0163               RLYDBL EQU    010H   RELAY DISABLE
0163               *
0163 1E 05         SETBIAS MVI   E,SBIAS*256/256  RECORD IT
0165 3E 03                MVI    A,3    ONLY 4 BIASES MAX
0167 CD AF 01             CALL   RECORD  RECORD
016A FE 02                CPI    2      IF SPHERE BOOM, USE
016C 2E 0E                MVI    L,SPHAD THE 1ST SELECT BIT
016E DA 87 01             JC     ANADAC  SHIFT INTO ANALOG BOARD
0171 2E 07                MVI    L,CYLAD  CYLINDERS ARE ANOTHER BIT
0173 C3 87 01             JMP    ANADAC
0176               *
0176 1E 0B         SETGUARD MVI  E,SGUARD*256/256
0178 CD AD 01             CALL   RECD1
017B 2E 0B                MVI    L,GRDAD  GUARDS ARE AT
017D C3 87 01             JMP    ANADAC  2ND SELECT BIT
0180               *
0180 1E 09         SETSTUB MVI   E,SSTUB*256/256
0182 CD AD 01             CALL   RECD1
0185 2E 0D                MVI    L,STBAD  STUBS ARE 3RD.
0187 F5            ANADAC PUSH   PSW
0188 CD 3D 02             CALL   NLOFF  GET SELECT FOR RELAYS OFF
018B E6 F0                ANI    0F0H
018D B5                   ORA    L      PUT IN SELECT BIT
018E 6F                   MOV    L,A
018F 7C                   MOV    A,H    CONVERT TO 0 TO FF
0190 EE 7F                XRI    7FH    FROM +/- 127
0192 67                   MOV    H,A
0193 F1                   POP    PSW
0194 CD 81 02             CALL   SHF16  SHIFT HL TO I/O REGS
0197 2E 14                MVI    L,NWR+ANASTB  STROBE ANALOG 4094'S
0199 0F            STROBE RRC    .      SELECT EITHER A OR B SIDE
019A 7D                   MOV    A,L
019B D2 A0 01             JNC    AS1
019E F6 20                ORI    AB     (B SIDE)
01A0 F3            AS1    DI     DON'T LET SAMPLING INTERRUPT
01A1 D3 F3                OUT    SERIAL
01A3 E6 EF         LATCH  ANI    -NWR-1  REMOVE WRITE BAR
01A5 D3 F3                OUT    SERIAL
01A7 F6 10                ORI    NWR    AND REPLACE IT
01A9 D3 F3                OUT    SERIAL
```

```
01AB FB          0300          EI
01AC C9          0301          RET    .
01AD             0302 *
01AD 3E 01       0303 RECD1    MVI    A,1     MASK TO 0 OR 1
01AF 25          0304 RECORD   DCR    H       DECREMENT THE BOOM TO 0-N
01B0 A4          0305          ANA    H       MASK(4) BITS
01B1 F5          0306          PUSH   PSW
01B2 83          0307          ADD    E       ADD OFFSET TO E
01B3 5F          0308          MOV    E,A
01B4 16 20       0309          MVI    D,IOSTAT/256
01B6 7D          0310          MOV    A,L     STORE THE VALUE
01B7 12          0311          STAX   D
01B8 F1          0312          POP    PSW
01B9 65          0313          MOV    H,L     PREPARE TO SHIFT
01BA C9          0314          RET
01BB             0315 *
01BB             0316 * RELAY CONTROL LOGIC.
01BB             0317 *  ON ENTRY: A=RELAY NUMBER 0 TO 21
01BB             0318 *           CRY=1 TO SET, 0 TO RESET
01BB             0319 *
01BB F5          0320 SETRELAY PUSH PSW
01BC CD 4E 02    0321          CALL   RECRLY  RECORD RELAY
01BF F1          0322          POP    PSW
01C0 17          0323          RAL
01C1 FE 04       0324          CPI    2*2     IF K0 OR K1, DIRECTIONAL
01C3 DA 02 02    0325          JC     RLYON
01C6 FE 0E       0326          CPI    2*7     IF K2-K6, NONLATCHING TYPE
01C8 DA FD 01    0327          JC     NONLAT
01CB FE 24       0328          CPI    2*18    IF K7-K17, LATCHING
01CD DA EF 01    0329          JC     LAT
01D0 2E 01       0330          MVI    L,1     IF K18,K19 THEN EXTERNAL(K1)
01D2 FE 28       0331          CPI    2*20
01D4 DA D8 01    0332          JC     EXT
01D7 2D          0333          DCR    L       ELSE K20-K21, THEN EXTERNAL(K0)
01D8             0334 *
01D8 F5          0335 EXT      PUSH   PSW     SAVE THE REAL RELAY#
01D9 1F          0336          RAR    .       SETRELAY(KL,ONOFF)
01DA 7D          0337          MOV    A,L
01DB CD BB 01    0338          CALL   SETRELAY
01DE F1          0339          POP    PSW
01DF             0340 *
01DF D6 20       0341          SUI    16*2    SETRELAY(K-16,ON)
01E1 1F          0342          RAR    .
01E2 37          0343          STC    .
01E3 F5          0344          PUSH   PSW
01E4 CD BB 01    0345          CALL   SETRELAY
01E7 CD A7 03    0346          CALL   DLA5    DELAY EXTRA
01EA F1          0347          POP    PSW
01EB B7          0348          ORA    A       SETRELAY(K-16,OFF)
01EC C3 BB 01    0349          JMP    SETRELAY
```

```
01EF            *
01EF F5         LAT    PUSH   PSW    SETRELAY(K1,ONOFF)
01F0 1F                RAR
01F1 3E 01             MVI    A,1
01F3 CD BB 01          CALL   SETRELAY
01F6 F1                POP    PSW
01F7 CD 02 02          CALL   RLYON  PULSE(K) ON/OFF
01FA C3 08 02          JMP    RLYOFF
01FD            *
01FD 0F         NONLAT RRC    .      FOR NONLATCHING, SIMPLY
01FE 07                RLC    .      TEST WHETHER TO TURN THE CURRENT
01FF D2 08 02          JNC    RLYOFF  ON OR OFF
0202            *
0202 CD 19 02   RLYON  CALL   SELECT  A=SELECT BITS FOR POWER
0205 C3 0B 02          JMP    SRLY
0208 CD 3D 02   RLYOFF CALL   NLOFF  A=TURN OFF BITS
020B 67         SRLY   MOV    H,A    PUT DECODER BITS IN MSBYTE
020C F6 0F             ORI    NILAD  DESELECT DACS FOR LOW POWER
020E 6F                MOV    L,A
020F CD 81 02          CALL   SHF16
0212 3E 14             MVI    A,NWR+ANASTB
0214 D3 F3             OUT    SERIAL
0216 C3 A7 03          JMP    DLA5
0219            *
0219            * RELAY SELECT ROUTINE
0219            * ON ENTRY: [A]=RELAY # TIMES 2
0219            * ON EXIT: [A]=DECODER SELECT BITS PLUS K0,K1
0219            *
0219 B7         SELECT ORA    A      A=RELAY NUMBER AGAIN
021A 1F                RAR    .
021B D6 02             SUI    2      IF K0 OR K1, USE OLD SELECT BITS
021D DA 29 02          JC     SEL01
0220 E6 0F             ANI    0FH    ELSE CONVERT A INTO DECODER #
0222 C6 18             ADI    18H
0224 E6 27             ANI    27H    MASK AND LOWER RLYDBL
0226 C3 2C 02          JMP    SETSEL
0229 3A 01 20   SEL01  LDA    OLDSEL  GET PRIOR SELECTION
022C            *
022C E6 3F      SETSEL ANI    03FH   MASK THE SELECTION BITS
022E 32 01 20          STA    OLDSEL
0231 5F                MOV    E,A
0232 3A 15 20          LDA    RS     PUT IN K0 AND K1
0235 E6 03             ANI    3
0237 0F                RRC
0238 0F                RRC
0239 EE C0             XRI    0C0H   INVERT FOR PNP TRANSISTORS
023B B3                ORA    E
023C C9                RET
023D            *
023D            * TURN OFF POWER TO RELAYS FROM K2 THRU K21
```

```
023D               *
023D CD 45 02      NLOFF  CALL   CLRNL  CLEAR NONLATCHING STATUS
0240 3E 10                MVI    A,RLYDBL  DISABLE RELAYS
0242 C3 2C 02             JMP    SETSEL
0245               *
0245 3A 15 20      CLRNL  LDA    RS     SHOW NON-LATCHING OFF
0248 E6 83                ANI    083H
024A 32 15 20             STA    RS
024D C9                   RET
024E               *
024E               * RECORD RELAY ON/OFF
024E               *
024E F5            RECRLY PUSH   PSW
024F FE 02                CPI    2      ANY RELAYS BUT KO/KI
0251 D4 45 02             CNC    CLRNL  CAUSE NONLATCHING TO RESET
0254 F1                   POP    PSW
0255 11 15 20             LXI    D,RS
0258               *
0258               *  SBIT: [DE]-> ARRAY, [A]=#, CARRY=BIT
0258               *
0258 F5            SBIT   PUSH   PSW    SAVE CARRY
0259 FE 08         SBI    CPI    8      SET HL->BYTE
025B DA 64 02             JC     SB2
025E 13                   INX    D
025F D6 08                SUI    8
0261 C3 59 02             JMP    SBI
0264 CD 80 00      SB2    CALL   UNARY  L=BIT MASK
0267 EB                   XCHG   .
0268 F1                   POP    PSW    GET CARRY
0269 7B                   MOV    A,E
026A DA 71 02             JC     SBSET
026D 2F                   CMA    .      CLEARING A BIT
026E A6                   ANA    M
026F 77                   MOV    M,A
0270 C9                   RET    .
0271 B6            SBSET  ORA    M      SETTING A BIT
0272 77                   MOV    M,A
0273 C9                   RET
0274               *
0274 21 03 20      IODSC  LXI    H,IOSTAT  REFERENCE IO DIGITAL STAT
0277 85                   ADD    L
0278 6F                   MOV    L,A
0279 7E                   MOV    A,M
027A C9                   RET
027B               *
027B               * SEND BITS TO THE ANALOG I/O REGISTER
027B               *  ON ENTRY: [HL]=DATA TO SHIFT
027B               *
027B C5            SHFB   PUSH   B
027C 0E 08                MVI    C,8    COUNT OF BITS
```

```
027E C3 84 02              JMP    ASHF
0281 C5          SHF16     PUSH   B
0282 0E 10                 MVI    C,16
0284             ;
0284 F5          ASHF      PUSH   PSW     SAVE ACCUM FOR RETURN
0285 29          ASH1      DAD    H       SHIFT ONE BIT INTO CARRY
0286 3E 08                 MVI    A,NWR/2  THEN INTO THE LSB OF A
0288 17                    RAL    .
0289 F3                    DI     .       DON'T ALLOW SAMPLING TO INTERRUPT
028A D3 F3                 OUT    SERIAL  PUT DATA OUT WITH CLOCK=0
028C F6 02                 ORI    SERCLK  THEN WITH RISING EDGE
028E D3 F3                 OUT    SERIAL
0290 FB                    EI
0291 0D                    DCR    C       UNTIL --COUNT=0
0292 C2 85 02              JNZ    ASH1
0295 F1                    POP    PSW
0296 C1                    POP    B
0297 C9                    RET
0298             ;
0298             ; VTRIM CONTROL LOGIC.
0298             ;  ON ENTRY: H=1 FOR VTRIM12, 2 FOR VTRIM34
0298             ;            [L]=DAC VALUE (8-BITS)
0298             ;
0298 1E 03       SETVTRIM MVI E,SVTRIM*256/256
029A CD AD 01              CALL   RECD1
029D F5                    PUSH   PSW     SAVE WHICH VTRIM
029E EB                    XCHG   .
029F 7A                    MOV    A,D     CONVERT TO 0 TO FF
02A0 EE 7F                 XRI    7FH     FROM +/- 127
02A2 57                    MOV    D,A
02A3 2E 01                 MVI    L,1     SELECT VTRIM PAIR DAC
02A5 CD 00 03              CALL   FSEND
02A8 F1                    POP    PSW     NOW IF VTRIM12,USE SIDE A
02A9 2E 18                 MVI    L,NWR+FILSTB
02AB C3 99 01              JMP    STROBE
02AE             ;
02AE             ; FILTER CONTROL LOGIC.
02AE             ;  ON ENTRY: H=FILTER NUMBER 1-7
02AE             ;            [L]=FILTER VALUE
02AE             ;
02AE 1E 0D       SETFILTER MVI E,SFILTER*256/256
02B0 3E 07                 MVI    A,7     SET MASK
02B2 CD AF 01              CALL   RECORD
02B5 54                    MOV    D,H     PUT DATA IN D REG
02B6 3C                    INR    A       TURN ON BIT 1-7
02B7 E6 07                 ANI    7
02B9 CD 80 00              CALL   UNARY  L=BIT SELECT
02BC CD 00 03              CALL   FSEND
02BF 3E 18                 MVI    A,NWR+FILSTB  CHOOSE A SIDE
02C1 CD A0 01              CALL   AS1
```

```
02C4 3E 38              MVI   A,NWR+FILSTB+AB  THEN B SIDE
02C6 CD A0 01           CALL  AS1
02C9 C3 FA 02           JMP   FILCLEAR
02CC              *
02CC              * SET MULTIPLEXOR BITS WHICH STEER THE FILTERING
02CC              *  ON ENTRY: A=MULTIPLEXOR NUMBER ON FILTER BOARD
02CC              *           L=VALUE TO ADDRESS ON THAT MUX
02CC              *
02CC C5           SETMUX PUSH B      CALL THE BIT REPLACEMENT
02CD 5D                 MOV   E,L    ROUTINE TO REPLACE ONLY
02CE 21 F2 02           LXI   H,MUXFIELD  THE FIELD FOR THAT REG
02D1 E6 07              ANI   7      MASK THE REGISTER NUMBER
02D3 4F                 MOV   C,A
02D4 06 00              MVI   B,0
02D6 09                 DAD   B
02D7              *
02D7 3A 14 20           LDA   MUXCOPY  GET THE MUX REGISTER
02DA CD C1 03           CALL  REPFIELD  REPLACE THE ONE FIELD
02DD 32 14 20           STA   MUXCOPY  AND THEN SEND IT OUT
02E0 C1                 POP   B
02E1 C3 FA 02           JMP   FILCLEAR
02E4              *
02E4              * TEST MULTIPLEXOR BITS
02E4              *  ON ENTRY: [A]=FIELD NUMBER
02E4              *  ON EXIT: ZERO FLAG IF FIELD=0
02E4              *
02E4 E6 07        TSTMUX ANI  7      [HL]->MUXFIELD(A)
02E6 5F                 MOV   E,A
02E7 16 00              MVI   D,0
02E9 21 F2 02           LXI   H,MUXFIELD
02EC 19                 DAD   D
02ED 3A 14 20           LDA   MUXCOPY  PICK UP ALL BITS
02F0 A6                 ANA   M      MASK TO SET STATUS
02F1 C9                 RET
02F2              *
02F2 20           MUXFIELD DB  32     V2/RI2 MUX
02F3 01                 DB    1      V12/RI1
02F4 06                 DB    6      6-POLE SELECT
02F5 08                 DB    8      V1/SC
02F6 21                 DB    32+1   V2+V12 TOGETHER
02F7 27                 DB    32+1+6  V2+V12+6POLE
02F8 2F                 DB    32+1+6+8  V2+V12+6POLE+V1
02F9 10                 DB    16 B AMPLIFIER
02FA              *
02FA              * FILTER BOARD CONTROL ROUTINES
02FA              *
02FA 21 00 00     FILCLEAR LXI H,0   CLEAR THE SELECTS
02FD 11 00 00           LXI   D,0    CLEAR THE DATA REGS
0300              *
0300              * FILTER SEND.  ON ENTRY [DE]=DATA AND [L]=SELECT BITS
```

```
0300             ;
0300 E5          FSEND   PUSH   H        SEND MUX BITS DOWN
0301 3A 14 20            LDA    MUXCOPY  WITHOUT THE LATCHING
0304 67                  MOV    H,A
0305 CD 7B 02            CALL   SHF8
0308 E1                  POP    H
0309 EB                  XCHG   .        NOW SEND BITS FROM [DE]
030A CD 7B 02            CALL   SHF8
030D             ;
030D 7B                  MOV    A,E      INVERT SELECT BITS
030E 2F                  CMA    .
030F 67                  MOV    H,A
0310 CD 7B 02            CALL   SHF8
0313             ;
0313 3E 12               MVI    A.NWR+FILSTB  STROBE THE FILTER
0315 D3 F3               OUT    SERIAL   BOARD
0317 C9                  RET    .
0318             ;
0318             ; MOTOR CONTROL SECTION
0318             ;  ON ENTRY: A=MOTOR#, CARRY=1 FOR ON, 0 FOR OFF
0318             ;
0318             ;
0318 8F          SETMOTOR ADC   A        CONVERT # AND ON/OFF
0319 CD 80 00            CALL   UNARY    INTO BIT NUMBER
031C CD 49 03            CALL   FLIP     INVERT THAT BIT
031F 3E 0A               MVI    A,10     DELAY 50 MILLISECONDS
0321 CD A7 03    SMDLA   CALL   DLA5
0324 3D                  DCR    A
0325 C2 21 03            JNZ    SMDLA
0328 C3 49 03            JMP    FLIP     AND RESET
032B             ;
032B             ; CYCLE THE KELLEY GAIN CHANGE.
032B             ;
032B 2E 80       SETKLY  MVI    L,KLYGC
032D C3 46 03            JMP    CYCLE
0330             ;
0330             ; OUTPUT THE MAGNETOMETER DATA TO THE PLASMA EXP.
0330             ;  ON ENTRY: [HL]=16 BIT VALUE TO SEND
0330             ;
0330 E5          SETPLA  PUSH   H
0331 2E 40               MVI    L,MAGLOAD  INDICATE REGISTER LOADING
0333 CD 49 03            CALL   FLIP
0336 E1                  POP    H
0337 7C                  MOV    A,H      SET HIGH BYTE
0338 2F                  CMA    .        INVERT FOR THE OPEN COLLECTOR
0339 D3 D0               OUT    LEPAH
033B 7D                  MOV    A,L      AND LOW BYTE
033C 2F                  CMA    .
033D D3 90               OUT    LEPAL
033F 2E 40               MVI    L,MAGLOAD  RE-ENABLE SHIFTING
```

```
0341 C3 49 03          JMP    FLIP
0344                *
0344 2E 20      RBURST MVI    L,BRUN  RESET BURST
0346 CD 49 03   CYCLE  CALL   FLIP   CYCLE=FLIP TWICE
0349 3A 00 20   FLIP   LDA    MKBCOPY  PICK UP COPY OF PORT
034C AD                XRA    L      INVERT THE BIT IN L
034D D3 C0             OUT    MOTORS
034F 32 00 20          STA    MKBCOPY
0352 C9                RET    .
0353                *
0353                * SEND [HL] TO THE BURST PROCESSOR
0353                * ON EXIT: CARRY SET IF FAILURE TO COMMUNICATE
0353                *
0353 20         SEND   DB     RIM    IF BURST RDY NOW
0354 07                RLC    .      RECOVER BY RECEIVING
0355 DC 7E 03          CC     RECOVER
0358                *
0358 E5                PUSH   H      SAVE HL AND BC
0359 C5                PUSH   B
035A 01 10 00          LXI    B,16   C=#BITS TO SEND
035D 3E C0             MVI    A,SDO+SDOEN  SDO=1 (REQUEST ATTN)
035F 30                DB     SIM
0360 05         WTBRS  DCR    B      WAIT MAX OF 3.9 MSEC
0361 37                STC    .
0362 CA 77 03          JZ     SENDX
0365 20                DB     RIM    WAITING FOR BURST TO RESPOND
0366 07                RLC    .      (READY IF 1)
0367 D2 60 03          JNC    WTBRS
036A 00                NOP    .      DISABLE WORD AND CMD INTERRUPTS ONLY
036B 3E 4D             MVI    A,SDOEN+MSE+DIS75+DIS55  SDO=0 TO START
036D 30                DB     SIM
036E 3E 80      SEND1  MVI    A,SDOEN*2  GET THE NEXT BIT
0370 29                DAD    H      BY SHIFTING THRU MSB OF HL
0371 1F                RAR    .      THEN INTO MSB OF ACCUM
0372 30                DB     SIM
0373 0D                DCR    C      COUNT # BITS
0374 C2 6E 03          JNZ    SEND1
0377 3E 48      SENDX  MVI    A,SDOEN+MSE  LEAVE SDO=0, INTS ON
0379 30                DB     SIM
037A FB                EI     .      RESTART INTERRUPTS
037B C1                POP    B
037C E1                POP    H
037D C9                RET
037E                *
037E E5         RECOVER PUSH  H      GRAB THE INFO
037F CD 84 03          CALL   RECEIVE  FROM THE BURST
0382 E1                POP    H      AND THROW AWAY
0383 C9                RET
0384                *
0384                * RECEIVE DATA FROM THE BURST
```

```
0384               * ON EXIT: IF NOT ZERO, [HL]=16 BITS FROM BURST
0384               *          ELSE NOTHING READY
0384               *
03B4 20            RECEIVE DB   RIM    IF REQUEST FROM BURST
03B5 E6 80                 ANI  SDO    IS NOT PRESENT, RETURN NOW.
03B7 C8                    RZ   .
03B8               *
03B8 C5                    PUSH B      SAVE REGISTERS
03B9 EB                    XCHG .
03BA               *
03BA 01 10 00              LXI  B,16   SHIFT * BITS
03BD 00                    NOP  .      DISABLE ONLY WORD AND CMD INTERRUPTS
03BE 3E CD                 MVI  A,SDO+SDOEN+MSE+DIS75+DIS55  ANSWER WITH SDO=1
0390 30                    DB   SIM
0391 20            RECWT   DB   RIM    WAIT FOR START EDGE
0392 04                    INR  B      BUT DON'T WAIT FOREVER
0393 A8                    XRA  B      IF B=128, QUIT
0394 FA 91 03              JM   RECWT
0397               *
0397 0A            RECBIT  LDAX B      7 CYCLE DELAY
0398 20                    DB   RIM    GET THE BIT
0399 07                    RLC
039A 18                    DB   SLDE   SHIFT INTO DE
039B 0D                    DCR  C
039C C2 97 03              JNZ  RECBIT
039F EB                    XCHG .      PUT RESULT INTO HL
03A0 3E 48                 MVI  A,SDOEN+MSE  SDO=0, INTS ON
03A2 30                    DB   SIM
03A3 B7                    ORA  A      RETURN NOT-ZERO
03A4 C1                    POP  B
03A5 FB                    EI   .      RE-ENABLE INTERRUPTS
03A6 C9                    RET
03A7               *
03A7               * DELAY ROUTINE
03A7               *
03A7 F5            DLA5    PUSH PSW    WAIT 5 MILLISECONDS
03A8 D5                    PUSH D
03A9 11 FE 01              LXI  D,510   2.5 MHZ/23 CYCLES
03AC 1B            DL1     DCX  D      BY COUNTING
03AD 7B                    MOV  A,E
03AE B2                    ORA  D
03AF C2 AC 03              JNZ  DL1
03B2 D1                    POP  D
03B3 F1                    POP  PSW
03B4 C9                    RET  .
03B5               *
03B5               * RESET WATCHDOG CIRCUIT
03B5               *
03B5 3A 00 20      RWATCH  LDA  MKBCOPY  PULSE THE RESET
03B8 F6 10                 ORI  WDRES  LINE TO THE 4015
```

```
03BA D3 C0       0700          OUT   MOTORS  COUNTER
03BC EE 10       0701          XRI   WDRES
03BE D3 C0       0702          OUT   MOTORS
03C0 C9          0703          RET   .
03C1             0704 *
03C1             0705 * FIELD REPLACEMENT ROUTINE
03C1             0706 * ON ENTRY: [A] = ORIGINAL 8-BIT VALUE
03C1             0707 *           [HL]->MASK OF THE FIELD
03C1             0708 *           [E] = NEW VALUE RIGHT ADJUSTED
03C1             0709 *
03C1 4F          0710 REPFIELD MOV  C,A    SAVE ORIGINAL
03C2 7E          0711          MOV   A,M    REMOVE THE MASKED FIELD
03C3 2F          0712          CMA
03C4 A1          0713          ANA   C
03C5 4F          0714          MOV   C,A
03C6             0715 *
03C6 EB          0716          XCHG  .      [DE]->MASK, L=VALUE
03C7 1A          0717          LDAX  D      SHIFT THE VALUE INTO POSITION
03C8 0F          0718 RF1      RRC   .      BY SHIFTING THE MASK
03C9 DA D0 03    0719          JC    RF2    THE OTHER WAY UNTIL A BIT
03CC 29          0720          DAD   H
03CD C3 C8 03    0721          JMP   RF1
03D0 1A          0722 RF2      LDAX  D      MASK THE SHIFTED VALUE
03D1 A5          0723          ANA   L
03D2 B1          0724          ORA   C      AND APPLY TO MODIFIED ORIGINAL
03D3 C9          0725          RET
03D4             0726 NEXT     EQU   $
03D4             0727          COM   NEXT
03D4             0728 *
03D4             0729 * RAM SECTION
03D4             0730 *
03D4             0731          ORG   IORAM
2000             0732 MKBCOPY DS     1      COPY OF MOTOR PORT
2001             0733 OLDSEL  DS     1      COPY OF RELAY SELECT BITS
2002             0734 SUNCOPY DS     1      COPY OF SUN BIT
2003             0735 IOSTAT  EQU    $      DIGITAL STATUS RETURNED FROM IO
2003             0736 SVTRIM  DS     2      STATUS OF VTRIMS,ETC
2005             0737 SBIAS   DS     4
2009             0738 SSTUB   DS     2
200B             0739 SGUARD  DS     2
200D             0740 SFILTER DS     7
2014             0741 MUXCOPY DS     1      COPY OF FILTER MUX
2015             0742 RS      DS     3      RELAY BITS 0-23
```

```
0000          *
0000          * CRRES FLIGHT SOFTWARE---FAST FLOATING POINT
0000          *  WRITTEN BY PETER HARVEY
0000          *  FILE : FFP.A
0000          *
0000          *  F.P. REGISTER IS CDE.
0000          *  FORMAT IS SIGN(1)+EXP(7)+MANTISSA(16)
0000          *  NO HIDDEN BIT
0000          *
0000          PSW    EQU    6
0000          SP     EQU    6
0000          *
0000          FFP    EQU    NEXT
0000                 ORG    FFP
03D4                 COM    LODFP
03D4                 COM    STOFP
03D4                 COM    FMUL
03D4                 COM    FDIV
03D4                 COM    FADD
03D4                 COM    FSUB
03D4                 COM    FCMP
03D4                 COM    FNEG
03D4                 COM    FLT32
03D4                 COM    FIX32
03D4                 COM    FSQUA
03D4                 COM    FSQRT
03D4                 COM    MU21
03D4          *
03D4 4E       LODFP  MOV    C,M
03D5 23              INX    H
03D6 56              MOV    D,M
03D7 23              INX    H
03D8 5E              MOV    E,M
03D9 C9              RET    .
03DA          *
03DA 71       STOFP  MOV    M,C
03DB 23              INX    H
03DC 72              MOV    M,D
03DD 23              INX    H
03DE 73              MOV    M,E
03DF C9              RET    .
03E0          *
03E0          * F.P. MULTIPLY ROUTINE
03E0          *
03E0 7A       FMUL   MOV    A,D     IF Y=0, QUIT NOW
03E1 B7              ORA    A
03E2 C8              RZ     .
03E3 46              MOV    B,M     LOAD PARAM FROM MEM
03E4 23              INX    H       INTO BHL FORMAT
```

```
03E5 7E                 MOV   A,M
03E6 B7                 ORA   A      IF ZERO THEN SET TO 0
03E7 CA 70 05           JZ    RETO
03EA 23                 INX   H      ELSE LOAD THE REST
03EB 6E                 MOV   L,M
03EC 67                 MOV   H,A
03ED               ;
03ED 78            FMS33 MOV   A,B    IF SAME SIGN, GO
03EE A9                 XRA   C
03EF F2 FD 03           JP    FMU33
03F2 CD 81 05           CALL  STRIP  REMOVE SIGNS FROM B&C
03F5 CD FD 03           CALL  FMU33  MULTIPLY THEN NEGATE
03F8               ;
03F8               FNEG  EQU   $
03F8 79            NEGFP MOV   A,C    AND NEGATE F.P
03F9 EE 80              XRI   80H
03FB 4F                 MOV   C,A
03FC C9                 RET   .
03FD               ;
03FD               ; F.P. MULTIPLY POSITIVES ONLY
03FD               ;
03FD 78            FMU33 MOV   A,B    ADD EXPONENTS
03FE 81                 ADD   C
03FF D6 40              SUI   40H    ADJUST BACK TO EXCESS 64
0401 FA 76 05           JM    ERCHK  IF MINUS, CHECK THE ERROR
0404 4F                 MOV   C,A
0405 CD F0 05           CALL  MU22F  [AHL.]= DE X HL
0408 C3 5C 05           JMP   NCHK   SHIFT UNTIL AHL NORMED,ROUND OFF
040B               ;
040B               ; F.P. DIVIDE
040B               ;
040B 7A            FDIV  MOV   A,D    IF ZERO DIVIDEND, QUIT
040C B7                 ORA   A
040D C8                 RZ    .
040E 46                 MOV   B,M    PICK UP DIVISOR
040F 23                 INX   H
0410 7E                 MOV   A,M
0411 B7                 ORA   A      IF DIVISOR 0, OVERFLOW
0412 CA 7B 05           JZ    OVERFLOW
0415 23                 INX   H
0416 6E                 MOV   L,M
0417 67                 MOV   H,A
0418               ;
0418 78                 MOV   A,B    IF SAME SIGN, DO
0419 A9                 XRA   C      SAME SIGNED VERSION
041A F2 26 04           JP    FDU33
041D CD 81 05           CALL  STRIP  REMOVE SIGNS
0420 CD 26 04           CALL  FDU33  DIVIDE OUT
0423 C3 F8 03           JMP   NEGFP  AND NEGATE
0426               ;
```

```
0426 79          0099 FDU33  MOV    A,C     EXP=C-B+40H
0427 90          0100        SUB    B
0428 C6 40       0101        ADI    40H
042A FA 76 05    0102        JM     ERCHK
042D 4F          0103        MOV    C,A
042E C5          0104        PUSH   B       SAVE EXPONENT
042F             0105 *
042F 7C          0106        MOV    A,H     BC=-DIVISOR
0430 2F          0107        CMA
0431 47          0108        MOV    B,A
0432 7D          0109        MOV    A,L
0433 2F          0110        CMA
0434 4F          0111        MOV    C,A
0435 03          0112        INX    B
0436             0113 *
0436             0114 * IF THE REMAINDER STARTS AS LARGE AS
0436             0115 * THE DIVISOR, THE FIRST BIT IS 1
0436             0116 *
0436 62          0117        MOV    H,D     HL=REMAINDER
0437 6B          0118        MOV    L,E
0438 09          0119        DAD    B       HL=REMAINDER-DIVISOR
0439 DA 56 04    0120        JC     FBIT1
043C             0121 *
043C             0122 * IF REMAINDER LESS THAN DIVISOR, THE FIRST
043C             0123 * BIT (INTEGER PART) IS ZERO.  DIVIDE FOR
043C             0124 * FRACTIONAL PART WHICH WILL BE AUTOMATICALLY
043C             0125 * NORMALIZED.
043C             0126 *
043C EB          0127        XCHG   .       HL=REMAINDER AGAIN
043D 3E 10       0128        MVI    A,16
043F CD 6F 04    0129        CALL   FDSHF   [DE]=[HL]*2/[BC]
0442 29          0130        DAD    H       IF REMAINDER>8000H
0443 DA 4C 04    0131        JC     DVROND   THEN ROUND UP
0446 09          0132        DAD    B       IF NEXT BIT WOULD BE 1
0447 DA 4C 04    0133        JC     DVROND   THEN ROUND UP
044A C1          0134        POP    B       RESTORE EXPONENT
044B C9          0135        RET    .       NO NORMALIZATION REQD
044C             0136 *
044C C1          0137 DVROND POP    B       C=EXPONENT
044D 1C          0138 ROND   INR    E ROUND OFF DE
044E C0          0139        RNZ    .       BUT DON'T PRODUCE
044F 14          0140        INR    D       A ZERO
0450 C0          0141        RNZ    .
0451 11 00 80    0142        LXI    D,8000H  IF ZERO, THEN
0454 0C          0143        INR    C       UP THE EXPONENT
0455 C9          0144        RET    .
0456             0145 *
0456             0146 * FIRST BIT=1. DIVIDE OUT 16 MORE BITS
0456             0147 * USING WHAT'S LEFT OF THE REMAINDER IN HL
0456             0148 *
```

```
0456 3E 10        FBIT1  MVI   A,16
0458 11 FF FF            LXI   D,-1
045B CD 6F 04            CALL  FDSHF  [DE]=[HL]/[BC]
045E                     *
045E C1                  POP   B      RESTORE THE EXPONENT
045F 0C                  INR   C      ADJUST SINCE 1ST BIT=1
0460 37                  STC   .      RIGHT SHIFT A 1 INTO DE
0461 7A                  MOV   A,D
0462 1F                  RAR
0463 57                  MOV   D,A
0464 7B                  MOV   A,E
0465 1F                  RAR
0466 5F                  MOV   E,A
0467 D0                  RNC   .      IF 17TH BIT WAS 0, STOP
0468 C3 4D 04            JMP   ROND   ELSE ROUND OFF
046B                     *
046B                     * DIVIDE NORMALIZED INTEGERS FOR F.P.
046B                     *
046B 33           FDSTK  INX   SP     REMOVE PARTIAL REMAINDER
046C 33                  INX   SP     FROM STACK
046D 3D           FDTST  DCR   A      DECR BIT COUNTER
046E C8                  RZ    .
046F 29           FDSHF  DAD   H      BRING DOWN A BIT INTO REM
0470 DA 81 04            JC    SUBIT  IF >=10000, THEN SUBTRACT
0473 EB                  XCHG  .
0474 29                  DAD   H      AND SHIFT RESULT REG
0475 EB                  XCHG
0476                     *
0476 1C                  INR   E      ASSUME RESULT=1
0477 E5           FDV22  PUSH  H      SAVE REMAINDER ON STK
0478 09                  DAD   B      IF REM<DIVISOR,LEAVE REM ALONE
0479 DA 6B 04            JC    FDSTK
047C E1                  POP   H      ELSE RESTORE REMAINDER
047D 1D                  DCR   E      SET RESULT BIT=0
047E C3 6D 04            JMP   FDTST
0481                     *
0481 EB           SUBIT  XCHG  .      FINISH THE SHIFT
0482 29                  DAD   H
0483 EB                  XCHG  .
0484 09                  DAD   B      SUBTRACT DIVISOR
0485 1C                  INR   E      SET RESULT BIT
0486 C3 6D 04            JMP   FDTST
0489                     *
0489                     * F.P. COMPARE
0489                     * ON EXIT: ZERO SET IF EQUAL, CARRY IF LESS THAN
0489                     *          CDE UNTOUCHED
0489                     *
0489 C5           FCMP   PUSH  B      SAVE CDE
048A D5                  PUSH  D
048B CD 98 04            CALL  FSUB   SUBTRACT THE TWO
```

```
048E 7A                  MOV   A,D    IF RESULT=0, RET
048F B7                  ORA   A
0490 CA 95 04            JZ    FCMPX
0493 79                  MOV   A,C    IF NEGATIVE, THEN
0494 07                  RLC   .      SET CARRY, ELSE NO CARRY
0495 D1           FCMPX  POP   D      RESTORE CDE
0496 C1                  POP   B
0497 C9                  RET   .
0498              ;
0498              ; F.P. SUB
0498              ;
0498 7E           FSUB   MOV   A,M    INVERT SIGN OF 2ND
0499 EE 80               XRI   80H    PARAMETER
049B 47                  MOV   B,A
049C C3 A0 04            JMP   FAD1
049F              ;
049F              ; F.P. ADD
049F              ;
049F 46           FADD   MOV   B,M    LOAD UP
04A0 23           FAD1   INX   H
04A1 7E                  MOV   A,M
04A2 23                  INX   H
04A3 6E                  MOV   L,M
04A4 67                  MOV   H,A
04A5 97                  SUB   A
04A6 BC                  CMP   H      IF BHL=0, QUIT
04A7 C8                  RZ    .
04A8 BA                  CMP   D      IF CDE=0, QUIT
04A9 CA 0C 05            JZ    SWITCH
04AC              ;
04AC 79                  MOV   A,C    COMPUTE EXP DIFFERENCE
04AD 90                  SUB   B
04AE 87                  ADD   A
04AF F2 B8 04            JP    POSDX
04B2 78                  MOV   A,B    SWAP CDE FOR BHL
04B3 41                  MOV   B,C
04B4 4F                  MOV   C,A
04B5 EB                  XCHG  .
04B6 90                  SUB   B      COMPUTE EXP DIFFERENCE
04B7 87                  ADD   A      AGAIN
04B8 CA C2 04     POSDX  JZ    ADSUB
04BB 0F                  RRC   .      DIV BY 2
04BC FE 10               CPI   16     IF CDE>>BHL, QUIT
04BE D0                  RNC
04BF CD F4 04            CALL  SHFHL  REDUCE HL A TIMES
04C2              ;
04C2 78           ADSUB  MOV   A,B    IF SIGNS DIFFER, GO
04C3 A9                  XRA   C
04C4 FA D2 04            JM    DIFFER
04C7 19                  DAD   D      ADD DE TO HL
```

```
04C8 EB                      XCHG  .      IF NO CARRY,
04C9 D0                      RNC   .      RETURN CDE
04CA                   *
04CA 7A                RITE1 MOV   A,D    ELSE SHIFT RIGHT ONE
04CB 1F                      RAR   .      INCLUDING THE CARRY
04CC 57                      MOV   D,A
04CD 7B                      MOV   A,E
04CE 1F                      RAR
04CF 5F                      MOV   E,A
04D0 0C                      INR   C      ADJUST EXPONENT
04D1 C9                      RET   .
04D2                   *
04D2 7B                DIFFER MOV  A,E    IF DE<HL,
04D3 95                      SUB   L      THEN NORM(B:HL-DE)
04D4 7A                      MOV   A,D
04D5 9C                      SBB   H
04D6 DA E7 04                JC    SUBD
04D9                   *
04D9 57                      MOV   D,A    ELSE NORM(C:DE-HL)
04DA 7B                      MOV   A,E
04DB 95                      SUB   L
04DC 5F                      MOV   E,A
04DD 21 00 00                LXI   H,0
04E0 B2                      ORA   D      IF DE<>0,
04E1 C2 21 05                JNZ   NORM   NORMALIZE WITH C
04E4 0E 00                   MVI   C,0    IF HL-DE=0, RETURN
04E6 C9                      RET   .
04E7                   *
04E7 7D                SUBD  MOV   A,L    DE = DE - HL
04E8 93                      SUB   E
04E9 5F                      MOV   E,A
04EA 7C                      MOV   A,H
04EB 9A                      SBB   D
04EC 57                      MOV   D,A
04ED 48                      MOV   C,B    USE BHL'S EXPONENT
04EE 21 00 00                LXI   H,0    AND SHIFT IN ZEROES
04F1 C3 21 05                JMP   NORM   NORMALIZE
04F4                   *
04F4                   * SHIFT HL RIGHT A TIMES
04F4                   *
04F4 D6 08             SHFHL SUI   8      IF LT 8, GO NOW
04F6 DA FF 04                JC    LT8
04F9 6C                      MOV   L,H    SHIFT 8
04FA 26 00                   MVI   H,0
04FC C8                      RZ    .      IF EXACTLY 8, RETURN
04FD D6 08                   SUI   8      ELSE DO SECOND 8
04FF C5                LT8   PUSH  B      SAVE EXPONENTS
0500 47                      MOV   B,A    SAVE INVERTED COUNTER
0501 97                      SUB   A      CLEAR ACCUM
0502 29                SHF1  DAD   H      SHIFT LEFT
```

```
0503 8F                     ADC   A      INTO A FROM HL
0504 04                     INR   B      COUNT UP TO 0
0505 C2 02 05               JNZ   SHF1
0508 C1                     POP   B      RESTORE EXPS
0509 6C                     MOV   L,H
050A 67                     MOV   H,A
050B C9                     RET   .
050C                  *
050C EB               SWITCH XCHG  .      CDE= BHL
050D 48                     MOV   C,B
050E C9                     RET   .
050F                  *
050F                  * CONVERT 32 BIT DATA TO F.P. FORMAT
050F                  *
050F 7A               FLT32 MOV   A,D    IF POSITIVE, JUST NORM
0510 B7                     ORA   A
0511 0E 60                  MVI   C,64+32  WITH LSB=2**0 TO BEGIN
0513 F2 21 05               JP    NORM
0516 CD 69 06               CALL  NEG32  NEGATE DEHL
0519 CD 21 05               CALL  NORM   NOW NORMALIZE
051C 79                     MOV   A,C    AND NEGATE FP
051D F6 80                  ORI   80H
051F 4F                     MOV   C,A
0520 C9                     RET   .
0521                  *
0521                  * NORMALIZE C:DEHL TO F.P. NORMAL FORM
0521                  *
0521 79               NORM  MOV   A,C    IF C NEGATIVE, TRAP IT
0522 B7                     ORA   A
0523 F2 2F 05               JP    NORMP
0526 E6 7F                  ANI   7FH
0528 4F                     MOV   C,A
0529 CD 2F 05               CALL  NORMP
052C C3 F8 03               JMP   NEGFP  AND NEG LATER
052F                  *
052F 7A               NORMP MOV   A,D    IF WITHIN 8 BITS, GO NOW
0530 B7                     ORA   A
0531 C2 59 05               JNZ   NORM1
0534 B3                     ORA   E      IF WITHIN 16, USE EHL
0535 C2 4E 05               JNZ   NRMEHL
0538 B4                     ORA   H      IF WITHIN 24, USE HL
0539 C2 48 05               JNZ   NRMHL
053C B5                     ORA   L      IF JUST L, USE IT
053D C2 42 05               JNZ   NRML
0540 4A                     MOV   C,D    ELSE CDE=0
0541 C9                     RET   .
0542                  *
0542 55               NRML  MOV   D,L    LOO FOR 3 BYTES
0543 06 18                  MVI   B,24   ADJUST EXP BY 24 BITS
0545 C3 53 05               JMP   AJEXP
```

```
0548 EB          NRMHL  XCHG   .      HLO FOR 3BYTES
0549 06 10              MVI    B,16   ADJUST EXP 16
054B C3 53 05           JMP    AJEXP
054E 53          NRMEHL MOV    D,E    SHIFT EHL TO DEH
054F 5C                 MOV    E,H
0550 65                 MOV    H,L
0551 06 08              MVI    B,8    ADJUST 8 BITS
0553 79          AJEXP  MOV    A,C    EXP=EXP-B
0554 90                 SUB    B
0555 4F                 MOV    C,A    IF PROBLEM, THEN UNDER
0556 DA 70 05           JC     UNDERFLOW
0559             *
0559             * BIT BY BIT NORMALIZATION
0559             *
0559 7A          NORM1  MOV    A,D    AHL=DEH
055A 6C                 MOV    L,H
055B 63                 MOV    H,E
055C B7          NCHK   ORA    A      SHIFT AHL TILL NORMED
055D FA 66 05           JM     NRMFIN
0560 0D          NCHK1  DCR    C      EXP<-EXP-1
0561 29                 DAD    H
0562 8F                 ADC    A
0563 F2 60 05           JP     NCHK1
0566 57          NRMFIN MOV    D,A    DE=AH
0567 5C                 MOV    E,H
0568 7D                 MOV    A,L    IF MSB(L)=1, ROUND OFF DE
0569 07                 RLC    .
056A DC 4D 04           CC     ROND
056D 79                 MOV    A,C    IF EXP POSITIVE, OK
056E B7                 ORA    A
056F F0                 RP     .
0570             *
0570             * ERRORS : UNDERFLOW AND OVERFLOW
0570             *
0570             UNDERFLOW EQU $
0570 0E 00       RETO   MVI    C,0    RETURN CDE=0
0572 11 00 00           LXI    D,0
0575 C9                 RET    .
0576             *
0576 FE C0       ERCHK  CPI    0C0H   IF BETWEEN 080H AND 0BFH
0578 D2 70 05           JNC    UNDERFLOW  THEN UNDERFLOW, ELSE OVER
057B             *
057B 0E 7F       OVERFLOW MVI  C,7FH  RETURN CDE=MAXIMUM
057D 11 FF FF           LXI    D,-1
0580 C9                 RET    .
0581             *
0581 78          STRIP  MOV    A,B    REMOVE SIGNS FROM B
0582 E6 7F              ANI    7FH
0584 47                 MOV    B,A
0585 79                 MOV    A,C
```

```
05B6 E6 7F           ANI   7FH
05B8 4F              MOV   C,A
05B9 C9              RET   .
05BA           *
05BA           * FIX32:  FLT TO FIX CONVERSION
05BA           *
05BA 79        FIX32 MOV   A,C    IF NEGATIVE, INVERT
05BB EE 80           XRI   80H    RESULTS
05BD FA 97 05        JM    FIXPOS
0590 4F              MOV   C,A
0591 CD 97 05        CALL  FIXPOS
0594 C3 69 06        JMP   NEG32
0597           *
0597 E6 7F     FIXPOS ANI  7FH    IF CDE<1, RETURN(0)
0599 FE 41           CPI   41H
059B DA C5 05        JC    ZERDH
059E FE 60           CPI   60H    IF >2**31, MAX IT
05A0 D2 CC 05        JNC   MAXDH
05A3           *
05A3 21 00 00        LXI   H,0    ELSE SHIFT MANTISSA
05A6 D6 50           SUI   40H+16  IF 2**16, QUIT
05A8 C8              RZ    .
05A9 EB              XCHG  .      DEHL=00XX, READY TO SHIFT
05AA D2 BC 05        JNC   SHDH   IF EXP WAS 51 TO 5F, 60
05AD C6 10           ADI   16     ELSE 41-4F, SHIFT THEN
05AF CD BC 05        CALL  SHDH   DIVIDE BY 2**16
05B2 EB              XCHG
05B3 11 00 00        LXI   D,0
05B6 C9              RET   .
05B7           *
05B7 29        SHCAR DAD   H      SHIFT DE PART
05B8 2C              INR   L      AND PUT IN CARRY
05B9 EB        DECRA XCHG  .      SWAP BACK HL
05BA 3D              DCR   A      IF COUNT=0, QUIT
05BB C8              RZ    .
05BC 29        SHDH  DAD   H      SHIFT HL ONE BIT
05BD EB              XCHG  .      IF CARRY, THEN
05BE DA B7 05        JC    SHCAR  UPDATE DE WITH CARRY
05C1 29              DAD   H      ELSE WITHOUT CARRY
05C2 C3 B9 05        JMP   DECRA
05C5           *
05C5 11 00 00  ZERDH LXI   D,0    DEHL=0
05C8 21 00 00        LXI   H,0
05CB C9              RET   .
05CC 11 FF 7F  MAXDH LXI   D,7FFFH  DEHL=MAXIMUM
05CF 21 FF FF        LXI   H,-1
05D2 C9              RET   .
05D3           *
05D3           * SQUARE [CDE]
05D3           *
```

```
05D3 7A          FSQUA  MOV   A,D    CHECK FOR 0
05D4 B7                 ORA   A
05D5 C8                 RZ    .
05D6 41                 MOV   B,C    BHL=CDE
05D7 62                 MOV   H,D
05D8 6B                 MOV   L,E
05D9 C3 ED 03           JMP   FMS33
05DC             *
05DC 7A          FSQRT  MOV   A,D    IF ZERO, QUIT
05DD B7                 ORA   A
05DE C8                 RZ    .
05DF 79                 MOV   A,C    IF ODD EXPONENT, SHIFT
05E0 E6 01              ANI   1
05E2 C4 CA 04           CNZ   RITE1
05E5 C5                 PUSH  B      SAVE EXPONENT
05E6 CD 44 06           CALL  SQR2   DE=DE**1/2
05E9 C1                 POP   B
05EA 79                 MOV   A,C    DIVIDE EXP BY 2
05EB 0F                 RRC
05EC C6 20              ADI   20H    IN EXCESS 64
05EE 4F                 MOV   C,A
05EF C9                 RET   .
05F0             *
05F0             * 16 X 16 MULTIPLY UNSIGNED.  OPTIMIZED FOR F.P.
05F0             *  [AHL] = [HL] * [DE]  TOP 3 BYTES
05F0             *
05F0 97          MU22F  SUB   A      IF E=0, DO SHORT MULT
05F1 BB                 CMP   E
05F2 CA 08 06           JZ    SHORD
05F5 B5                 ORA   L      IF L=0, DO SHORT WITH H
05F6 CA 09 06           JZ    SHORH
05F9 E5                 PUSH  H      AHL= L*DE
05FA CD 0A 06           CALL  MU21
05FD 6C                 MOV   L,H    THROW AWAY LS BYTE
05FE 67                 MOV   H,A    SAVE UPPER BYTES
05FF E3                 XTHL  .      SAVE EM, GET MS BYTE OF 1ST
0600             *
0600 7C                 MOV   A,H    AHL=MSB*DE
0601 CD 0A 06           CALL  MU21
0604 D1                 POP   D      GRAB THE TWO STORED
0605 19                 DAD   D      ADD PARTIAL RESULTS
0606 88                 ADC   B      FOR THREE BYTES (AHL)
0607 C9                 RET   .
0608             *
0608 EB          SHORD  XCHG  .      SHORT MULT
0609 7C          SHORH  MOV   A,H    JUST MULT H*DE
060A             *
060A             * 16 X 8 MULTIPLY UNSIGNED
060A             *  [AHL] <- A * [DE]
060A             *  TAKES 198 TO 297 CYCLES
```

```
060A              *
060A 21 00 00     MU21   LXI   H,0   ZERO RESULT REG
060D 44                  MOV   B,H   B<-0
060E              *
060E 87           MULTX  ADD   A     SHIFT MSB TO CARRY
060F D2 14 06            JNC   X2
0612 19                  DAD   D     IF C=1, THEN ADD [DE]
0613 88                  ADC   B     IF OVERFLOW,BUMP MSBYTE
0614 29           X2     DAD   H     SHIFT FOR NEXT TEST
0615              *
0615 8F                  ADC   A     AND SO ON
0616 D2 1B 06            JNC   X4
0619 19                  DAD   D
061A 88                  ADC   B
061B 29           X4     DAD   H
061C              *
061C 8F                  ADC   A
061D D2 22 06            JNC   X8
0620 19                  DAD   D
0621 88                  ADC   B
0622 29           X8     DAD   H
0623              *
0623 8F                  ADC   A
0624 D2 29 06            JNC   X10
0627 19                  DAD   D
0628 88                  ADC   B
0629 29           X10    DAD   H
062A              *
062A 8F                  ADC   A
062B D2 30 06            JNC   X20
062E 19                  DAD   D
062F 88                  ADC   B
0630 29           X20    DAD   H
0631              *
0631 8F                  ADC   A
0632 D2 37 06            JNC   X40
0635 19                  DAD   D
0636 88                  ADC   B
0637 29           X40    DAD   H
0638              *
0638 8F                  ADC   A
0639 D2 3E 06            JNC   X80
063C 19                  DAD   D
063D 88                  ADC   B
063E 29           X80    DAD   H
063F              *
063F 8F                  ADC   A
0640 D0                  RNC
0641 19                  DAD   D
0642 88                  ADC   B
```

```
0643 C9                RET
0644              *
0644              * INTEGER SQUARE ROOT OF DE
0644              *    [\]
0644 01 00 80     SQR2  LXI   B,8000H  GUESS=80, ROOT0=0
0647 CD 54 06     SQRA1 CALL  APPX     CHECK APPROXIMATION
064A 78                 MOV   A,B      AND SHIFT APPX BIT
064B 0F                 RRC
064C 47                 MOV   B,A
064D D2 47 06           JNC   SQRA1
0650 51                 MOV   D,C      DE=RESULT
0651 1E 00              MVI   E,0
0653 C9                 RET   .
0654              *
0654 D5           APPX  PUSH  D        SAVE X
0655 78                 MOV   A,B      TRY NEW TEST BIT
0656 81                 ADD   C
0657 5F                 MOV   E,A
0658 16 00              MVI   D,0
065A C5                 PUSH  B        SAVE BC
065B CD 0A 06           CALL  MU21     AHL=A*DE
065E C1                 POP   B
065F D1                 POP   D        COMPARE TO X
0660 7B                 MOV   A,E      IF X <HL THEN TOO BIG
0661 95                 SUB   L
0662 7A                 MOV   A,D
0663 9C                 SBB   H
0664 D8                 RC    .
0665 78                 MOV   A,B      ELSE ADD TEST BIT TO C
0666 81                 ADD   C
0667 4F                 MOV   C,A
0668 C9                 RET   .
0669              *
0669 CD 77 06     NEG32 CALL  INV16    INVERT DEHL
066C EB                 XCHG
066D CD 77 06           CALL  INV16
0670 EB                 XCHG  .
0671 23                 INX   H        AND ADD 1
0672              *
0672 7C                 MOV   A,H      IF HL=0, INCR DE
0673 B5                 ORA   L
0674 C0                 RNZ   .
0675 13                 INX   D
0676 C9                 RET   .
0677              *
0677 7C           INV16 MOV   A,H      INVERT HL
0678 2F                 CMA
0679 67                 MOV   H,A
067A 7D                 MOV   A,L
067B 2F                 CMA
```

```
067C 6F          0599        MOV   L,A
067D C9          0600        RET   .
067E             0601 NEXT   EQU   $
067E                         COM   NEXT
```

```
0000                 *
0000                 * CRRES FLIGHT SOFTWARE---MATRIX SOLVER
0000                 * WRITTEN BY PETER R HARVEY
0000                 * FILE : MATRIX.A
0000                 *
0000                 MATR1  EQU   NEXT
0000                        ORG   MATR1
067E                        COM   IMATX  INIT MATRIX ADDR/SIZE
067E                        COM   SOLVE  SOLVE MATRIX
067E                 *
067E                 * INITIALIZE MATRIX SIZE AND ADDRESS
067E                 *
067E 22 92 24        IMATX  SHLD  MATRX  SAVE ADDRESS OF MATRIX
0681 EB                     XCHG
0682 22 94 24               SHLD  RESULT  SAVE ADDRESS OF RESULT
0685 32 90 24               STA   N      SAVE SIZE
0688 3C                     INR   A      SAVE SIZE+1
0689 32 91 24               STA   NP1
068C C9                     RET   .
068D                 *
068D                 * MATRIX SOLVER
068D                 * RETURNS CARRY SET IF MATRIX ERROR
068D                 *
068D 3E 01           SOLVE  MVI   A,1    FOR J:=1 TO N-1
068F 32 97 24               STA   J
0692 CD FF 06        SOL1   CALL  LOCNZ  LOCATE NON-ZERO IN COL
0695 D8                     RC    .      IF NOT POSS, ERROR
0696 CD 5D 07               CALL  SUBROWJ  SUBTRACT JTH FROM OTHERS
0699 21 97 24               LXI   H,J    J=J+1
069C 34                     INR   M
069D 3A 90 24               LDA   N      IF N=J, QUIT
06A0 BE                     CMP   M
06A1 C2 92 06               JNZ   SOL1
06A4                 *
06A4                 * CALCULATE SOLUTION FROM A DIAGONAL MATRIX
06A4                 *
06A4 3A 90 24               LDA   N      FOR M2:=N DOWNTO 1
06A7 32 9A 24        CAL1   STA   M2
06AA 67                     MOV   H,A    LOAD ELEMENT (M2,N+1)
06AB 3A 91 24               LDA   NP1
06AE 6F                     MOV   L,A
06AF CD D7 07               CALL  LODZ
06B2                 *
06B2 3A 9A 24               LDA   M2     DIVIDE BY ELEMENT (M2,M2)
06B5 6F                     MOV   L,A
06B6 67                     MOV   H,A
06B7 CD E3 07               CALL  DIVZ
06BA                 *
06BA 2A 94 24               LHLD  RESULT  STORE IN X(M2)
```

```
06BD CD BC 07          CALL   REFXM2
06C0 CD DA 03          CALL   STOFP
06C3             *
06C3             * IF TOP ROW, THEN WE'RE FINISHED.
06C3             * ELSE UPDATE THE CONSTANTS ON THE RIGHT SIDE
06C3             *
06C3 3A 9A 24          LDA    M2      IF M2=1, QUIT
06C6 3D                DCR    A
06C7 C8                RZ     .
06C8 32 96 24    UPCON STA    Q       FOR Q=M2-1 DOWNTO 1
06CB 67                MOV    H,A     Z(Q,N+1)=Z(Q,N+1)-X(M2)*Z(Q,M2)
06CC 3A 9A 24          LDA    M2
06CF 6F                MOV    L,A
06D0 CD D7 07          CALL   LODZ
06D3             *
06D3 2A 94 24          LHLD   RESULT  MULTIPLY BY X(M2)
06D6 CD BC 07          CALL   REFXM2
06D9 CD E0 03          CALL   FMUL
06DC             *
06DC 79                MOV    A,C     CHANGE SIGN OF RESULT
06DD EE 80             XRI    80H
06DF 4F                MOV    C,A
06E0             *
06E0 3A 96 24          LDA    Q       SUBTRACT FROM Z(Q,N+1)
06E3 67                MOV    H,A
06E4 3A 91 24          LDA    NP1
06E7 6F                MOV    L,A
06E8 E5                PUSH   H
06E9 CD E9 07          CALL   ADZ
06EC E1                POP    H       STORE AT Z(Q,N+1)
06ED CD DD 07          CALL   STRZ
06F0             *
06F0 3A 96 24          LDA    Q       Q=Q-1 UNTIL 0
06F3 3D                DCR    A
06F4 C2 C8 06          JNZ    UPCON
06F7             *
06F7 3A 9A 24          LDA    M2      M2=M2-1
06FA 3D                DCR    A       UNTIL 0
06FB C2 A7 06          JNZ    CAL1
06FE C9                RET    .
06FF             *
06FF             * LOCATE A NON-ZERO IN COLUMN J
06FF             *
06FF 3A 97 24    LOCNZ LDA    J       FOR Q:=J TO N
0702 32 96 24    LOC1  STA    Q
0705 67                MOV    H,A     IS-ZERO Z(Q,J)?
0706 3A 97 24          LDA    J
0709 6F                MOV    L,A
070A CD EF 07          CALL   ISZRO
070D C2 1D 07          JNZ    SWAPROW  NO. 60
```

```
0710 3A 96 24             LDA   Q       YES. Q=Q+1
0713 3C                   INR   A
0714 21 91 24             LXI   H,NP1   UNTIL Q=N+1
0717 BE                   CMP   M
0718 C2 02 07             JNZ   LOC1
071B 37                   STC   .       ELSE RETURN CARRY
071C C9                   RET   .
071D                *
071D                * SWAP QTH ROW WITH JTH ROW
071D                *
071D 3A 96 24       SWAPROW LDA Q       IF Q=J, WE'RE DONE
0720 21 97 24             LXI   H,J
0723 BE                   CMP   M
0724 C8                   RZ    .
0725 7E                   MOV   A,M     FOR KLM:=J TO NP1
0726 32 9B 24       SWP1  STA   KLM
0729 6F                   MOV   L,A     SWAP(Z(Q,KLM),Z(J,KLM))
072A 3A 96 24             LDA   Q
072D 67                   MOV   H,A
072E CD CB 07             CALL  REFZ
0731 EB                   XCHG
0732 3A 97 24             LDA   J
0735 67                   MOV   H,A
0736 3A 9B 24             LDA   KLM
0739 6F                   MOV   L,A
073A CD CB 07             CALL  REFZ
073D CD 4F 07             CALL  SWAP3
0740                *
0740 3A 9B 24             LDA   KLM
0743 3C                   INR   A
0744 21 91 24             LXI   H,NP1
0747 BE                   CMP   M
0748 DA 26 07             JC    SWP1
074B CA 26 07             JZ    SWP1
074E C9                   RET   .
074F                *
074F 0E 03          SWAP3 MVI   C,3     SWAP F.P. QTYS
0751 1A             SWPLP LDAX  D       GRAB DATA
0752 46                   MOV   B,M     FROM BOTH
0753 EB                   XCHG
0754 12                   STAX  D       STORE FROM BOTH
0755 70                   MOV   M,B
0756 23                   INX   H
0757 13                   INX   D
0758 0D                   DCR   C
0759 C2 51 07             JNZ   SWPLP
075C C9                   RET   .
075D                *
075D                * SUBTRACT JTH ROW FROM SUBSEQUENT ROWS
075D                *
```

```
0750 3A 97 24   SUBROWJ LDA   J       FOR K=J+1 TO N
0760 3C                 INR   A
0761 32 98 24   SUBK1   STA   K
0764 67                 MOV   H,A     LOAD Z(K,J)
0765 3A 97 24           LDA   J
0768 6F                 MOV   L,A
0769 CD 07 07           CALL  LODZ
076C 3A 97 24           LDA   J       DIVIDE BY Z(J,J)
076F 67                 MOV   H,A
0770 6F                 MOV   L,A
0771 CD E3 07           CALL  DIVZ
0774 21 9C 24           LXI   H,RATIO  STORE IN RATIO
0777 CD 0A 03           CALL  STOFP
077A                    *
077A 3A 97 24           LDA   J       FOR P=J TO N+1
077D 32 99 24   SUBL1   STA   P
0780 6F                 MOV   L,A     LOAD Z(J,P)
0781 3A 97 24           LDA   J
0784 67                 MOV   H,A
0785 CD 07 07           CALL  LODZ
0788 21 9C 24           LXI   H,RATIO  MULT BY RATIO
078B CD E0 03           CALL  FMUL
078E 79                 MOV   A,C     CHANGE SIGN
078F EE 80              XRI   80H
0791 4F                 MOV   C,A
0792                    *
0792 3A 98 24           LDA   K       ADD TO Z(K,P)
0795 67                 MOV   H,A
0796 3A 99 24           LDA   P
0799 6F                 MOV   L,A
079A E5                 PUSH  H
079B CD E9 07           CALL  ADZ
079E E1                 POP   H       REPLACE Z(K,P)
079F CD DD 07           CALL  STRZ
07A2                    *
07A2 3A 99 24           LDA   P       P=P+1
07A5 3C                 INR   A
07A6 21 91 24           LXI   H,NP1   IF P<N+1, OK
07A9 BE                 CMP   M
07AA DA 70 07           JC    SUBL1
07AD CA 70 07           JZ    SUBL1
07B0                    *
07B0 3A 98 24           LDA   K       K=K+1
07B3 3C                 INR   A
07B4 21 91 24           LXI   H,NP1   UNTIL K=N+1
07B7 BE                 CMP   M
07B8 C2 61 07           JNZ   SUBK1
07BB C9                 RET   ,
07BC                    *
07BC 2A 94 24   REFXM2  LHLD  RESULT
```

```
07BF 3A 9A 24             LDA   M2
07C2 3D                   DCR   A
07C3 47                   MOV   B,A
07C4 87                   ADD   A
07C5 80                   ADD   B
07C6 DF                   RST   REF/8
07C7 C9                   RET
07C8                 *
07C8                 * MATRIX UTILITIES
07C8                 * H=ROW, L=COL
07C8                 * IN A 4X5 ARRAY
07C8                 *
07C8 25              REFZ DCR   H
07C9 2D                   DCR   L
07CA 7C                   MOV   A,H    ROW*5 ACROSS
07CB 87                   ADD   A
07CC 87                   ADD   A
07CD 84                   ADD   H
07CE 85                   ADD   L      +COL NUMBER (0 THRU 4)
07CF 6F                   MOV   L,A    TIMES FLT (3)
07D0 87                   ADD   A
07D1 85                   ADD   L
07D2 2A 92 24             LHLD  MATRX  GET ADDRESS OF ARRAY Z
07D5 DF                   RST   REF/8
07D6 C9                   RET   .
07D7                 *
07D7 CD C8 07        LODZ CALL  REFZ
07DA C3 D4 03             JMP   LODFP
07DD CD C8 07        STRZ CALL  REFZ
07E0 C3 DA 03             JMP   STOFP
07E3 CD C8 07        DIVZ CALL  REFZ
07E6 C3 0B 04             JMP   FDIV
07E9 CD C8 07        ADZ  CALL  REFZ
07EC C3 9F 04             JMP   FADD
07EF CD D7 07        ISZRO CALL LODZ   RETURN(Z) IF REALLY ZERO
07F2 7A                   MOV   A,D
07F3 B7                   ORA   A
07F4 C8                   RZ    .
07F5 79                   MOV   A,C    OR IF UNDER 1/2**10
07F6 E6 7F                ANI   7FH
07F8 FE 37                CPI   41H-10
07FA D0                   RNC   .
07FB 97                   SUB   A
07FC C9                   RET
07FD                 NEXT EQU   $
07FD                      COM   NEXT
07FD                 *
07FD                 * VARIABLES
07FD                 *
07FD                      ORG   MATRAM
```

```
2490          0249 N      DS     1     SIZE OF ARRAY TO SOLVE
2491          0250 NP1    DS     1     SIZE+1
2492          0251 MATRX  DS     2     MATRIX STARTING ADDRESS
2494          0252 RESULT DS     2     RESULT ADDRESS
2496          0253 Q      DS     1     TEMPORARIES
2497          0254 J      DS     1
2498          0255 K      DS     1
2499          0256 P      DS     1
249A          0257 M2     DS     1
249B          0258 KLM    DS     1
249C          0259 RATIO  DS     4
```

```
0000                 *
0000                 * CRRES FLIGHT SOFTWARE---TRIG FUNCTION
0000                 * FILE : TRIG.A
0000                 *
0000                 NPOINTS EQU   32
0000                 FLT     EQU   3
0000                 *
0000                 TRIG    EQU   NEXT
0000                         ORG   TRIG
07FD                         COM   SIN
07FD                         COM   COS
07FD                         COM   SINSQ
07FD                         COM   COSSQ
07FD                         COM   SNCS
07FD                 *
07FD 21 91 08        SNCS    LXI   H,SNCST  REFER TO SIN*COS
0800 FE 33                   CPI   SNCSX-SNCST  IF IN TABLE OK
0802 DA 07 08                JC    REFOK
0805 D6 30                   SUI   NPOINTS/2*FLT  ELSE BEGIN AT
0807 DF              REFOK   RST   REF/8
0808 C9                      RET
0809                 *
0809                 * SIN SQUARED/COSINE SQUARED
0809                 *
0809 CD 16 08        SINSQ   CALL  SIN    REFER TO SINE
080C C3 12 08                JMP   ADOF
080F CD 1D 08        COSSQ   CALL  COS    REFER TO COSINE
0812 3E 33           ADOF    MVI   A,COS2T-COSTAB  ADD OFFSET
0814 DF                      RST   REF/8
0815 C9                      RET   .
0816                 *
0816                 * SIN AND COS ROUTINES.
0816                 * ON ENTRY: A=0 THRU 31 TIMES FLOAT
0816                 * ON EXIT : HL-> SIN OR COSINE F.P. VALUE
0816                 *
0816 D6 18           SIN     SUI   NPOINTS/4*FLT  REMOVE 1/4 PHASE
0818 D2 1D 08                JNC   COS    AND USE COSINE ROUTINE
081B C6 60                   ADI   NPOINTS*FLT  MOVE TO END OF PERIOD
081D                 *
081D 21 2B 08        COS     LXI   H,COSTAB  REFERENCE COSINE TABLE
0820 FE 33                   CPI   COS2T-COSTX  IF IN TABLE, OK
0822 DA 29 08                JC    REFCOS
0825 2F                      CMA   ELSE NPOINTS-INDEX
0826 3C                      INR   A
0827 C6 60                   ADI   NPOINTS*FLT
0829 DF              REFCOS  RST   REF/8
082A C9                      RET   .
082B                 *
082B                 * SIN/COSINE TABLES
```

```
082B         *
082B         COSTAB EQU   $
082B 41      COSTX  DB    041H   COSINE
082C 80             DB    080H
082D 00             DB    000H
082E 40             DB    040H
082F FB             DB    0FBH
0830 15             DB    015H
0831 40             DB    040H
0832 EC             DB    0ECH
0833 83             DB    083H
0834 40             DB    040H
0835 D4             DB    0D4H
0836 DB             DB    0DBH
0837 40             DB    040H
0838 B5             DB    0B5H
0839 04             DB    004H
083A 40             DB    040H
083B BE             DB    0BEH
083C 3A             DB    03AH
083D 3F             DB    03FH
083E C3             DB    0C3H
083F EE             DB    0EEH
0840 3E             DB    03EH
0841 C7             DB    0C7H
0842 C5             DB    0C5H
0843 00             DB    000H
0844 00             DB    000H
0845 00             DB    000H
0846 BE             DB    0BEH
0847 C7             DB    0C7H
0848 C5             DB    0C5H
0849 BF             DB    0BFH
084A C3             DB    0C3H
084B EE             DB    0EEH
084C C0             DB    0C0H
084D BE             DB    0BEH
084E 3A             DB    03AH
084F C0             DB    0C0H
0850 B5             DB    0B5H
0851 04             DB    004H
0852 C0             DB    0C0H
0853 D4             DB    0D4H
0854 DB             DB    0DBH
0855 C0             DB    0C0H
0856 EC             DB    0ECH
0857 83             DB    083H
0858 C0             DB    0C0H
0859 FB             DB    0FBH
085A 15             DB    015H
```

```
0B5B C1        0100         DB    0C1H
0B5C B0        0101         DB    0B0H
0B5D 00        0102         DB    000H
0B5E 41        0103 COS2T   DB    041H   COS SQUARED
0B5F 80        0104         DB    080H
0B60 00        0105         DB    000H
0B61 40        0106         DB    040H
0B62 F6        0107         DB    0F6H
0B63 43        0108         DB    043H
0B64 40        0109         DB    040H
0B65 DA        0110         DB    0DAH
0B66 83        0111         DB    083H
0B67 40        0112         DB    040H
0B68 B0        0113         DB    0B0H
0B69 FC        0114         DB    0FCH
0B6A 40        0115         DB    040H
0B6B B0        0116         DB    0B0H
0B6C 00        0117         DB    000H
0B6D 3F        0118         DB    03FH
0B6E 9E        0119         DB    09EH
0B6F 0B        0120         DB    00BH
0B70 3E        0121         DB    03EH
0B71 95        0122         DB    095H
0B72 F6        0123         DB    0F6H
0B73 3C        0124         DB    03CH
0B74 9B        0125         DB    09BH
0B75 E5        0126         DB    0E5H
0B76 00        0127         DB    000H
0B77 00        0128         DB    000H
0B78 00        0129         DB    000H
0B79 3C        0130         DB    03CH
0B7A 9B        0131         DB    09BH
0B7B E5        0132         DB    0E5H
0B7C 3E        0133         DB    03EH
0B7D 95        0134         DB    095H
0B7E F6        0135         DB    0F6H
0B7F 3F        0136         DB    03FH
0B30 9E        0137         DB    09EH
0BB1 0B        0138         DB    00BH
0B82 40        0139         DB    040H
0BB3 B0        0140         DB    0B0H
0BB4 00        0141         DB    000H
0BB5 40        0142         DB    040H
0886 B0        0143         DB    0B0H
0B87 FC        0144         DB    0FCH
0B88 40        0145         DB    040H
0B89 DA        0146         DB    0DAH
08BA B3        0147         DB    0B3H
08BB 40        0148         DB    040H
0B8C F6        0149         DB    0F6H
```

```
088D 43        0150       DB    043H
088E 41        0151       DB    041H
088F 80        0152       DB    080H
0890 00        0153       DB    000H
0891 00        0154 SNCST DB    000H    SIN*COS
0892 00        0155       DB    000H
0893 00        0156       DB    000H
0894 3E        0157       DB    03EH
0895 C3        0158       DB    0C3H
0896 EF        0159       DB    0EFH
0897 3F        0160       DB    03FH
0898 B5        0161       DB    0B5H
0899 04        0162       DB    004H
089A 3F        0163       DB    03FH
089B EC        0164       DB    0ECH
089C 83        0165       DB    083H
089D 40        0166       DB    040H
089E 80        0167       DB    080H
089F 00        0168       DB    000H
08A0 3F        0169       DB    03FH
08A1 EC        0170       DB    0ECH
08A2 83        0171       DB    083H
08A3 3F        0172       DB    03FH
08A4 B5        0173       DB    0B5H
08A5 04        0174       DB    004H
08A6 3E        0175       DB    03EH
08A7 C3        0176       DB    0C3H
08A8 EF        0177       DB    0EFH
08A9 00        0178       DB    000H
08AA 00        0179       DB    000H
08AB 00        0180       DB    000H
08AC BE        0181       DB    0BEH
08AD C3        0182       DB    0C3H
08AE EF        0183       DB    0EFH
08AF BF        0184       DB    0BFH
08B0 B5        0185       DB    0B5H
08B1 04        0186       DB    004H
08B2 BF        0187       DB    0BFH
08B3 EC        0188       DB    0ECH
08B4 83        0189       DB    083H
08B5 C0        0190       DB    0C0H
08B6 80        0191       DB    080H
08B7 00        0192       DB    000H
08B8 BF        0193       DB    0BFH
08B9 EC        0194       DB    0ECH
08BA 83        0195       DB    083H
08BB BF        0196       DB    0BFH
08BC B5        0197       DB    0B5H
08BD 04        0198       DB    004H
08BE BE        0199       DB    0BEH
```

```
08BF C3            0200          DB     0C3H
08C0 EF            0201          DB     0EFH
08C1 00            0202          DB     000H
08C2 00            0203          DB     000H
08C3 00            0204          DB     000H
08C4               0205 SNCSX    EQU    $
08C4               0206 NEXT     EQU    $
08C4               0207          COM    NEXT
```

```
0000             *
0000             * CRRES FLIGHT SOFTWARE--- SPIN FITTING ALGORITHM
0000             * FILE : SPIN.A
0000             *
0000             FLT     EQU   3        FLOATING POINT LENGTH
0000             NULL    EQU   40H      NULL FLOAT INDICATOR
0000             NPOINTS EQU   32       NUMBER OF POINTS IN FIT
0000             MINN    EQU   3        MINIMUM N OR MHI
0000             AVPTS   EQU   4        # POINTS TO RETAIN AHI/ALO
0000             HIGAIN  EQU   10H      HIGH GAIN SAMPLE
0000             REJBIT  EQU   20H      REJECTED BIT
0000             *
0000             GAINF   EQU   0*FLT    INPUT PARAMETER BLOCK
0000             ALPHA   EQU   1*FLT
0000             BETA    EQU   2*FLT
0000             ATABL   EQU   3*FLT  AHI/ALO TABLE
0000             ATEND   EQU   AVPTS*2*FLT+ATABL
0000             PSW     EQU   6
0000             *
0000             * SPIN FIT SUBROUTINE.
0000             * ON ENTRY: [HL]-> SAMPLED DATA BLOCK
0000             *           [DE]-> PARAM BLOCK
0000             *           [BC]-> WHERE TO PUT RESULTS
0000             *
0000             SPIN    EQU   NEXT
0000                     COM   SPIN
0000                     ORG   SPIN
08C4 22 20 23            SHLD  SAMPTR  SAVE DATA IN ADDR
08C7 EB                  XCHG  .
08C8 22 22 23            SHLD  PRMPTR  SAVE PARAMS ADDR
08CB C5                  PUSH  B       STACK RESULTS ADDR
08CC 3E C3               MVI   A,0C3H  PUT JUMP INTO FUNCTION
08CE 32 28 23            STA   FN
08D1             *
08D1 CD E3 08            CALL  PHAS1   DO 4X4 SOLUTIONS
08D4 CD 0E 0C            CALL  AVERAGE  AVERAGE AHI AND ALO
08D7 CD FB 08            CALL  PHAS2   THEN DO FITS W/ B AND C
08DA             *
08DA D1          EXIT    POP   D       MOVE LOCAL RESULTS INTO
08DB 21 35 23            LXI   H,AHI   THE DESTINATION AREA
08DE 0E 10               MVI   C,FLT*5+1  (AHI,LO,B,C,SIGMA,N)
08E0 C3 B8 0C            JMP   COPY
08E3             *
08E3             * 1ST AND 2ND PHASES.
08E3             *
08E3 3E 04       PHAS1   MVI   A,4     INIT FOR 4X4 MATRIX
08E5 11 35 23            LXI   D,AHI   RESULTS TO AHI THRU C
08E8 CD 13 09            CALL  IPHASE
08EB CD 24 09            CALL  GEN44   GENERATE SUMS FOR 4X4
```

```
08EE CD AE 09     PH1    CALL   SOL44  SOLVE THE 4X4
08F1 CD 85 0B            CALL   CALCSD  CALCULATE STD DEVIATION
08F4 CD E1 09            CALL   REJ44  REJECT ALL POINTS OFF CURVE
08F7 C2 EE 08            JNZ    PH1    REPEAT UNTIL NO GARBAGE
08FA C9                  RET    .
08FB              *
08FB 3E 02        PHAS2  MVI    A,2    INIT FOR SMALLER MATRIX
08FD 11 3B 23            LXI    D,BCOMP  RESULTS TO B AND C
0900 CD 13 09            CALL   IPHASE
0903 CD 62 09            CALL   GEN22
0906 CD D3 09     PH2    CALL   SOL22  SOLVE SMALLER MATRIX
0909 CD 85 0B            CALL   CALCSD  COMPUTE SIGMA
090C CD ED 09            CALL   REJ22  REJECT POINTS
090F C2 06 09            JNZ    PH2
0912 C9                  RET    .
0913              *
0913              * INIT A PHASE.
0913              *
0913 21 48 24     IPHASE LXI    H,SCRATCH  TELL SOLVER WHERE
0916 CD 7E 06            CALL   IMATX  TO FIND MATRIX COPY
0919 3E 03               MVI    A,ALPHA  APJB <- ALPHA
091B CD DB 0C            CALL   REFP
091E 11 46 23            LXI    D,APJB
0921 C3 86 0C            JMP    FMOV
0924              *
0924              * GENERATE THE MATRIX TO BEGIN
0924              *
0924 CD 6D 0A     GEN44  CALL   CLRMAX  ZERO THE MATRIX (A=0)
0927 32 44 23            STA    N      # POINTS=0
092A 32 45 23            STA    NHI    # HI GAIN POINTS=0
092D 32 2D 23            STA    ADDSB  SET ADD MODE FOR SUMS
0930              *
0930 21 39 09            LXI    H,GENM1  REPEAT GENM1 FOR EACH POINT
0933 CD 10 0D            CALL   DOLOOP
0936 C3 0A 08            JMP    Q34    THEN PRODUCE COPIES
0939              *
0939 CD C1 0C     GENM1  CALL   REFSAM  [HL]->SAMDTA(I)
093C 5E                  MOV    E,M    [DE]=SAMPLE
093D 23                  INX    H
093E 56                  MOV    D,M
093F E5                  PUSH   H
0940 CD 9F 0C            CALL   FLT12  FLOAT 12 BIT FORMAT
0943 E1                  POP    H
0944 7E                  MOV    A,M    CHECK INDICATOR
0945 E6 10               ANI    HIGAIN  IF HIGH GAIN, DO IT
0947 C2 53 09            JNZ    MULGN
094A              *
094A CD D0 0C            CALL   REFET  ET(I)=CDE
094D CD DA 03            CALL   STOFP
0950 C3 E9 0A            JMP    Q2LO1  DO QUAD 2 LO AND Q1
```

```
0953               *
0953 2A 22 23      MULGN  LHLD   PRMPTR  MULTIPLY BY GAIN FACTOR
0956 CD E0 03             CALL   FMUL    TO ADJUST FOR GAIN
0959 CD D0 0C             CALL   REFET   STORE IN ET(1)
095C C3 DA 03             CALL   STOFP
095F C3 C2 0A             JMP    Q2H11   DO QUAD2 HI AND Q1
0962               *
0962               * GENERATE THE 2X2 MATRIX
0962               *
0962 3E 80         GEN22  MVI    A,80H   SUBTRACT MODE
0964 32 2D 23             STA    ADDSB
0967 01 18 24             LXI    B,ETCOS  ETCOS=ETCOS-AHI*ECSH1
096A 11 3F 24             LXI    D,ECSH1
096D 21 35 23             LXI    H,AH1
0970 CD 94 09             CALL   REMOVE
0973 01 18 24             LXI    B,ETCOS  - ALO*ECSLO
0976 11 30 24             LXI    D,ECSLO
0979 21 38 23             LXI    H,ALO
097C CD 94 09             CALL   REMOVE
097F               *
097F 01 27 24             LXI    B,ETSIN  ETSIN=ETSIN-AH1*ESNH1
0982 11 42 24             LXI    D,ESNH1
0985 21 35 23             LXI    H,AH1
0988 CD 94 09             CALL   REMOVE
098B 01 27 24             LXI    B,ETSIN  -ALO*ESNLO
098E 11 33 24             LXI    D,ESNLO
0991 21 38 23             LXI    H,ALO
0994               *
0994 C5            REMOVE PUSH   B       SAVE DEST
0995 D5                   PUSH   D       SAVE MULTIPLIER
0996 CD D4 03             CALL   LODFP
0999 7A                   MOV    A,D     IF MULTIPLIER NULL
099A B7                   ORA    A       OR ZERO, QUIT
099B CA AB 09             JZ     UNSTK
099E FE 40                CPI    NULL
09A0 CA AB 09             JZ     UNSTK
09A3 E1                   POP    H
09A4 CD E0 03             CALL   FMUL    MULT
09A7 E1                   POP    H
09A8 C3 7E 0C             JMP    SUM     AND SUBTRACT
09AB E1            UNSTK  POP    H
09AC E1                   POP    H
09AD C9                   RET    .
09AE               *
09AE               * SOLVE 4X4 AND 2X2 MATRICES.
09AE               *
09AE 11 48 24      SOL44  LXI    D,SCRATCH  SOLVE 4X4
09B1 21 0C 24             LXI    H,MAT
09B4 0E 3C                MVI    C,FLT*4*5
09B6 CD DB 09             CALL   SLVNN
```

```
09B9                 *
09B9 3A 45 23              LDA   MHI    IF NO HI GAIN POINTS
09BC B7                    ORA   A
09BD C2 C5 09              JNZ   CHKLO
09C0 3E 40                 MVI   A,NULL  THEN NULL AHI
09C2 32 36 23              STA   AHI+1
09C5                 *
09C5 3A 44 23        CHKLO LDA   N      IF NO LO GAIN POINTS
09C8 21 45 23              LXI   H,MHI  (N-MHI=0)
09CB 96                    SUB   M
09CC C0                    RNZ   .
09CD 3E 40                 MVI   A,NULL  THEN NULL ALO
09CF 32 39 23              STA   ALO+1
09D2 C9                    RET   .
09D3                 *
09D3 11 48 24        SOL22 LXI   D,SCRATCH  COPY SMALLER PART
09D6 21 12 24              LXI   H,ECOS2  OF THE MATRIX
09D9 0E 1B                 MVI   C,ETSIN-ECOS2+FLT
09DB CD BB 0C        SLVNN CALL  COPY
09DE C3 B0 06              JMP   SOLVE
09E1                 *
09E1                 * REJECT ALL POINTS OVER DISC AWAY FROM THE CURVE
09E1                 *
09E1 21 1D 0A        REJ44 LXI   H,RJ4  USE RJ4 REJECT FUNCTION
09E4 CD 0F 0A              CALL  REJN
09E7 CD 0A 0B              CALL  Q34    UPDATE THE REST FOR 4X4
09EA C3 F3 09              JMP   RJFIN  INCREASE APJB, RETURN POINTS
09ED                 *
09ED 21 29 0A        REJ22 LXI   H,RJ2  REJECT POINT FOR 2X2
09F0 CD 0F 0A              CALL  REJN
09F3 21 46 23        RJFIN LXI   H,APJB  APJB = APJB+BETA. (INCREASE
09F6 CD D4 03              CALL  LODFP  THE DIFFICULTY FACTOR)
09F9 3E 06                 MVI   A,BETA
09FB CD DB 0C              CALL  REFP
09FE CD 9F 04              CALL  FADD
0A01 21 46 23              LXI   H,APJB
0A04 CD DA 03              CALL  STOFP
0A07                 *
0A07 3A 2E 23              LDA   NP     RETURN # POINTS REJECTED
0A0A 21 44 23              LXI   H,N
0A0D 96                    SUB   M
0A0E C9                    RET   .
0A0F                 *
0A0F 3A 44 23        REJN  LDA   N      RECORD # POINTS NOW
0A12 32 2E 23              STA   NP
0A15 3E B0                 MVI   A,B0H  SET SUBTRACT MODE
0A17 32 2D 23              STA   ADDSB
0A1A C3 10 00              JMP   DOLOOP  EXECUTE REJECT FUNCTION
0A1D                 *
0A1D CD 33 0A        RJ4   CALL  RJTEST  TEST FOR NEW REJECTION
```

```
0A20 D0          0200        RNC    .       IF OLD OR NO REJECT, QUIT
0A21 E6 10       0201        ANI    HIGAIN  IF NEW REJ, UPDATE SUMS IN Q2 AND Q1
0A23 CA E9 0A    0202        JZ     Q2LO1
0A26 C3 C2 0A    0203        JMP    Q2HI1
0A29             0204 *
0A29 CD 33 0A    0205 RJ2    CALL   RJTEST  TEST FOR REJECTION
0A2C D0          0206        RNC    .       OLD OR NO REJECT
0A2D CD 4D 0B    0207        CALL   Q1X22   UPDATE SUMS IN Q1 FOR 2X2
0A30 C3 7D 0A    0208        JMP    Q1      AND FINISH WITH NORMAL Q1
0A33             0209 *
0A33             0210 * RETURNS NO CARRY IF NO CHANGE
0A33             0211 * CARRY IF NEWLY REJECTED
0A33             0212 *
0A33 CD C1 0C    0213 RJTEST CALL   REFSAM  IF REJECTED ALREADY,
0A36 23          0214        INX    H
0A37 7E          0215        MOV    A,M     THEN RETURN NO CHANGE
0A38 FE 20       0216        CPI    REJBIT
0A3A D0          0217        RNC    .
0A3B             0218 *
0A3B E6 10       0219        ANI    HIGAIN  REJECT IF TOO FEW POINTS
0A3D 21 45 23    0220        LXI    H,MHI
0A40 7E          0221        MOV    A,M     IN THIS GAIN.
0A41 C2 48 0A    0222        JNZ    CM1
0A44 3A 44 23    0223        LDA    N       (N-MHI)<MINN
0A47 96          0224        SUB    M
0A48 FE 03       0225 CM1    CPI    MINN
0A4A DA 63 0A    0226        JC     MRKREJ
0A4D             0227 *
0A4D CD CA 0C    0228        CALL   REFDIF  IF ABS(DIFF(I))<DISC
0A50 CD D4 03    0229        CALL   LODFP   THEN RETURN(NO CHANGE)
0A53 7A          0230        MOV    A,D     (IF DIFF=0, RETURN(NC))
0A54 B7          0231        ORA    A
0A55 C8          0232        RZ     .
0A56 79          0233        MOV    A,C     TAKE ABSOLUTE VALUE
0A57 E6 7F       0234        ANI    7FH
0A59 4F          0235        MOV    C,A
0A5A 21 49 23    0236        LXI    H,DISC  COMPARE EXPONENTS
0A5D BE          0237        CMP    M       IF EQUAL EXPS, THEN CALL
0A5E CC 89 04    0238        CZ     FCMP    COMPARE TO SET FLAGS
0A61 3F          0239        CMC    .       RETURN(NC) IF DIFF < DISC
0A62 D0          0240        RNC    .
0A63             0241 *
0A63 CD C1 0C    0242 MRKREJ CALL   REFSAM  MARK POINT REJECTED
0A66 23          0243        INX    H
0A67 7E          0244        MOV    A,M
0A68 F6 20       0245        ORI    REJBIT
0A6A 77          0246        MOV    M,A
0A6B 37          0247        STC    .       RETURN (CHANGED)
0A6C C9          0248        RET    .
0A6D             0249 *
```

```
0A6D 21 0D 24     CLRMAX LXI    H,MAT+1  ZERO BYTE 2
0A70 0E 14               MVI    C,4*5  OF EACH IN MATRIX
0A72 11 03 00            LXI    D,FLT
0A75 97                  SUB    A
0A76 77           CLR1   MOV    M,A
0A77 19                  DAD    D
0A78 0D                  DCR    C
0A79 C2 76 0A            JNZ    CLR1
0A7C C9                  RET    .
0A7D              *
0A7D              * QUAD 1
0A7D              *
0A7D 21 44 23     Q1     LXI    H,N    N = N (+/-) 1
0A80 CD 88 0C            CALL   COUNT
0A83 CD F5 0C            CALL   COSISQ  SUM COS**2
0A86 21 12 24            LXI    H,ECOS2
0A89 CD 7E 0C            CALL   SUM
0A8C CD 07 0D            CALL   SINCOS  SUM SINCOS
0A8F 21 15 24            LXI    H,ESNCS
0A92 CD 7E 0C            CALL   SUM
0A95 21 21 24            LXI    H,ECSSN  SUM COSSIN
0A98 CD DA 03            CALL   STOFP
0A9B CD FE 0C            CALL   SINESQ  SUM SIN**2
0A9E 21 24 24            LXI    H,ESIN2
0AA1 CD 7E 0C            CALL   SUM
0AA4              *
0AA4 CD E3 0C            CALL   COSINE  SUM COS*ET(I)
0AA7 CD D0 0C            CALL   REFET
0AAA CD E0 03            CALL   FMUL
0AAD 21 18 24            LXI    H,ETCOS
0AB0 CD 7E 0C            CALL   SUM
0AB3              *
0AB3 CD EC 0C            CALL   SINE   SUM SIN*ET(I)
0AB6 CD D0 0C            CALL   REFET
0AB9 CD E0 03            CALL   FMUL
0ABC 21 27 24            LXI    H,ETSIN
0ABF C3 7E 0C            JMP    SUM
0AC2              *
0AC2              * QUAD 2. SUM OF COSINES AND SINES
0AC2              *
0AC2 21 45 23     Q2HI1  LXI    H,MHI  KEEP TRACK OF HIGH POINTS
0AC5 CD 88 0C            CALL   COUNT  M=M+1 OR -1
0AC8 CD E3 0C            CALL   COSINE  ECSHI=ECSHI(+/-)COS
0ACB 21 3F 24            LXI    H,ECSHI
0ACE CD 7E 0C            CALL   SUM
0AD1 CD EC 0C            CALL   SINE   ESNHI=ESNHI(+/-)SIN
0AD4 21 42 24            LXI    H,ESNHI
0AD7 CD 7E 0C            CALL   SUM
0ADA CD D0 0C            CALL   REFET  EHI=EHI(+/-)ET(I)
0ADD CD D4 03            CALL   LODFP
```

```
0AE0 21 45 24     0300        LXI    H,EHI
0AE3 CD 7E 0C     0301        CALL   SUM
0AE6 C3 7D 0A     0302        JMP    Q1
0AE9              0303 !
0AE9 CD E3 0C     0304 Q2L01  CALL   COSINE  SAME AS ABOVE FOR LO GAIN
0AEC 21 30 24     0305        LXI    H,ECSLO
0AEF CD 7E 0C     0306        CALL   SUM
0AF2 CD EC 0C     0307        CALL   SINE
0AF5 21 33 24     0308        LXI    H,ESNLO
0AF8 CD 7E 0C     0309        CALL   SUM
0AFB              0310 !
0AFB CD D0 0C     0311        CALL   REFET
0AFE CD D4 03     0312        CALL   LODFP
0B01 21 36 24     0313        LXI    H,ELO
0B04 CD 7E 0C     0314        CALL   SUM
0B07 C3 7D 0A     0315        JMP    Q1
0B0A              0316 !
0B0A              0317 ! QUADS 3 AND 4. COPY VALUES FROM QUAD 2
0B0A              0318 !
0B0A 21 3F 24     0319 Q34    LXI    H,ECSHI  ECH=ECSHI
0B0D 11 0C 24     0320        LXI    D,ECH
0B10 CD B6 0C     0321        CALL   FMOV
0B13 11 1B 24     0322        LXI    D,ESH  ESH=ESNHI
0B16 CD B6 0C     0323        CALL   FMOV
0B19              0324 !
0B19 21 30 24     0325        LXI    H,ECSLO  ECL=ECSLO
0B1C 11 0F 24     0326        LXI    D,ECL
0B1F CD B6 0C     0327        CALL   FMOV
0B22 11 1E 24     0328        LXI    D,ESL
0B25 CD B6 0C     0329        CALL   FMOV
0B28              0330 !
0B28              0331 ! QUAD 4.
0B2B              0332 !
0B2B 3A 45 23     0333        LDA    MHI    FMHI=FLOAT(M)
0B2B B7           0334        ORA    A      IF M=0 THEN SET MHI=1
0B2C C2 30 0B     0335        JNZ    Q4A
0B2F 3C           0336        INR    A
0B30 CD 96 0C     0337 Q4A    CALL   FLT8
0B33 21 39 24     0338        LXI    H,FMHI
0B36 CD DA 03     0339        CALL   STOFP
0B39              0340 !
0B39 3A 44 23     0341        LDA    N      NLM=FLOAT(N-M)
0B3C 21 45 23     0342        LXI    H,MHI
0B3F 96           0343        SUB    M
0B40 C2 44 0B     0344        JNZ    Q4B
0B43 3C           0345        INR    A
0B44 CD 96 0C     0346 Q4B    CALL   FLT8
0B47 21 2D 24     0347        LXI    H,NLM
0B4A C3 DA 03     0348        JMP    STOFP
0B4D              0349 !
```

```
0B4D                 * REMOVE SUMS FOR 2X2 FROM QI ETSIN AND ETCOS
0B4D                 *
0B4D 2I 36 23        QIX22  LXI   H,AHI+I  IF HIGH GAIN
0B50 E6 I0                  ANI   HIGAIN
0B52 C2 5B 0B               JNZ   QIXI
0B55 2I 39 23               LXI   H,ALO+1  IF LOW GAIN
0B5B                 *
0B5B 7E              QIXI   MOV   A,M    CHECK IF AX NULL
0B59 B7                     ORA   A      IF NULL OR ZERO, QUIT
0B5A CB                     RZ    .
0B5B FE 40                  CPI   NULL
0B5D C8                     RZ    .
0B5E 2B                     DCX   H
0B5F                 *
0B5F E5                     PUSH  H
0B60 CD D4 03               CALL  LODFP  ETCOS=ETCOS-A(GAIN)*COS(I)
0B63 3A 2B 23               LDA   INDEX  (HL->COS(INDEX))
0B66 CD ID 0B               CALL  COS
0B69 CD E0 03               CALL  FMUL
0B6C 2I I8 24               LXI   H,ETCOS
0B6F CD 7E 0C               CALL  SUM
0B72                 *
0B72 EI                     POP   H
0B73 CD D4 03               CALL  LODFP  ETSIN=ETSIN-A(GAIN)*SIN(I)
0B76 3A 2B 23               LDA   INDEX
0B79 CD 16 0B               CALL  SIN
0B7C CD E0 03               CALL  FMUL
0B7F 21 27 24               LXI   H,ETSIN
0B82 C3 7E 0C               JMP   SUM
0B85                 *
0B85                 * CALCULATE THE STANDARD DEVIATION AND THE
0B85                 *   DISCRIMINATION FACTOR
0B85                 *
0B85 97              CALCSD SUB   A      INIT SIGMA SUM = 0
0B86 32 42 23               STA   SIGMA+I
0B89 3A 44 23               LDA   N      FLTNI = FLOAT(N-I)
0B8C 3D                     DCR   A
0B8D CD 96 0C               CALL  FLT8
0B90 2I 2F 23               LXI   H,FLTNI
0B93 CD DA 03               CALL  STOFP
0B96 2I BD 0B               LXI   H,CDIFF  CALCULATE DIFFERENCES
0B99 CD I0 0D               CALL  DOLOOP  AND SUM SQUARES
0B9C 2I 4I 23               LXI   H,SIGMA  SIGMA = SIGMA/(N-I)
0B9F CD D4 03               CALL  LODFP
0BA2 2I 2F 23               LXI   H,FLTNI
0BA5 CD 0B 04               CALL  FDIV
0BA8 CD DC 05               CALL  FSQRT  TAKE ROOT
0BAB 2I 4I 23               LXI   H,SIGMA  THEN STORE
0BAE CD DA 03               CALL  STOFP
0BBI                 *
```

```
0BB1 21 46 23            LXI    H,APJB  DISC=SIGMA*(ALPHA+J*BETA)
0BB4 CD E0 03            CALL   FMUL
0BB7 21 49 23            LXI    H,DISC
0BBA C3 DA 03            JMP    STOFP
0BBD                *
0BBD                * CALCULATE THE DIFFERENCES ARRAY
0BBD                *  FOR POINTS WHICH HAVE NOT BEEN REJECTED
0BBD                *  DIFF(I) = A(GAIN)+B*COS(I)+C*SIN(I) - ET(I)
0BBD                *  ALSO SUM SIGMA AT THE SAME TIME
0BBD                *
0BBD CD C1 0C       CDIFF  CALL   REFSAM  IF POINT(I)=REJECTED,QUIT
0BC0 23                  INX    H
0BC1 7E                  MOV    A,M
0BC2 FE 20               CPI    REJBIT
0BC4 D0                  RNC    .
0BC5 F5                  PUSH   PSW    ELSE SAVE GAIN INFO
0BC6 CD E3 0C            CALL   COSINE  B*COSINE(I)
0BC9 21 3B 23            LXI    H,BCOMP
0BCC CD E0 03            CALL   FMUL
0BCF 21 32 23            LXI    H,FTEMP
0BD2 CD DA 03            CALL   STOFP
0BD5 CD EC 0C            CALL   SINE    + C*SIN(I)
0BD8 21 3E 23            LXI    H,CCOMP
0BDB CD E0 03            CALL   FMUL
0BDE 21 32 23            LXI    H,FTEMP
0BE1 CD 9F 04            CALL   FADD
0BE4                *
0BE4 F1                  POP    PSW    IF LO GAIN, USE ALO
0BE5 E6 10               ANI    HIGAIN
0BE7 21 38 23            LXI    H,ALO
0BEA CA F0 0B            JZ     ADOFF
0BED 21 35 23            LXI    H,AHI  ELSE USE AHI
0BF0 CD 9F 04       ADOFF  CALL   FADD   ADD OFFSET
0BF3 CD D0 0C            CALL   REFET  SUBTRACT ET(I)
0BF6 CD 98 04            CALL   FSUB
0BF9 CD CA 0C            CALL   REFDIF  STORE IN DIFF(I)
0BFC CD DA 03            CALL   STOFP
0BFF CD D3 05            CALL   FSQUARE  SQUARE DIFF(I)
0C02 21 41 23            LXI    H,SIGMA  SIGMA=SIGMA+DIFF(I)**2
0C05 CD 9F 04            CALL   FADD
0C08 21 41 23            LXI    H,SIGMA
0C0B C3 DA 03            JMP    STOFP
0C0E                *
0C0E                * MAINTAIN AVERAGES OF AHI AND ALO
0C0E                *
0C0E 3E 09          AVERAGE MVI   A,ATABL   FORGET THE OLDEST
0C10 CD D8 0C            CALL   REFP    AHI/ALO PAIR
0C13 EB                  XCHG   .       [DE]->ATABL[0]
0C14 21 06 00            LXI    H,FLT*2  [HL]->ATABL[2]
0C17 19                  DAD    D
```

```
0C18 0E 12            MVI   C,AVPTS-1*2*FLT
0C1A CD B8 0C         CALL  COPY
0C1D             *
0C1D 21 35 23         LXI   H,AHI  COPY AHI/ALO INTO TABLE BOTTOM
0C20 0E 06            MVI   C,2*FLT
0C22 CD B8 0C         CALL  COPY
0C25             *
0C25 3E 09            MVI   A,ATABL  ADD ALL THE AHI'S
0C27 21 35 23         LXI   H,AHI  STORE INTO AHI RESULT
0C2A CD 32 0C         CALL  AVGI
0C2D 3E 0C            MVI   A,ATABL+FLT  ADD ALL THE ALO'S
0C2F 21 38 23         LXI   H,ALO
0C32             *
0C32 E5          AVGI PUSH  H      SAVE ADDRESS OF AHI/ALO
0C33 0E 00            MVI   C,0    SUM=0.0
0C35 11 00 00         LXI   D,0
0C38 21 2C 23         LXI   H,PTCNT
0C3B 36 00            MVI   M,0
0C3D             *
0C3D F5          SUMLP PUSH PSW    REFERENCE PARAM[A]
0C3E CD D8 0C         CALL  REFP
0C41 CD 70 0C         CALL  CHKADD  ADD IT
0C44 F1               POP   PSW    STEP 2 FLTS DOWN IN TABLE
0C45 C6 06            ADI   FLT*2
0C47 FE 21            CPI   ATEND  IF MORE IN TABLE, LOOP
0C49 DA 3D 0C         JC    SUMLP
0C4C             *
0C4C CD 53 0C         CALL  AVDIV
0C4F E1               POP   H
0C50 C3 DA 03         JMP   STOFP
0C53             *
0C53 3A 2C 23    AVDIV LDA  PTCNT  IF ZERO PTS, RETURN NULL
0C56 B7               ORA   A
0C57 CA 6D 0C         JZ    AVGNULL
0C5A C5               PUSH  B      SAVE SUM
0C5B D5               PUSH  D
0C5C CD 96 0C         CALL  FLTB
0C5F 21 32 23         LXI   H,FTEMP
0C62 CD DA 03         CALL  STOFP
0C65 D1               POP   D      DIVIDE SUM BY COUNT
0C66 C1               POP   B
0C67 21 32 23         LXI   H,FTEMP
0C6A C3 0B 04         JMP   FDIV
0C6D 16 40       AVGNULL MVI D,NULL
0C6F C9               RET
0C70             *
0C70 23          CHKADD INX H      CHECK IF (HL)-> GOOD FLT
0C71 7E               MOV   A,M
0C72 FE 40            CPI   NULL
0C74 C8               RZ    .
```

```
0C75 2B                    DCX   H        IF GOOD, ADD TO SUM
0C76 CD 9F 04              CALL  FADD
0C79 21 2C 23              LXI   H,PTCNT  PTCNT++
0C7C 34                    INR   M
0C7D C9                    RET
0C7E                 !
0C7E                 ! << UTILITY SECTION >>
0C7E                 ! SUM CDE INTO VALUE AT HL.
0C7E                 !
0C7E E5              SUM   PUSH  H        SAVE ADDRESS
0C7F 3A 2D 23              LDA   ADDSB    PUT ADD/SUB MARK
0C82 A9                    XRA   C        INTO CDE (INVERT SIGN FOR SUB)
0C83 4F                    MOV   C,A
0C84 CD 9F 04              CALL  FADD     ADD 'EM
0C87 E1                    POP   H        AND STORE
0C88 C3 DA 03              JMP   STOFP
0C8B                 !
0C8B 3A 2D 23        COUNT LDA   ADDSB    IF ADD MODE,
0C8E B7                    ORA   A        THEN
0C8F FA 94 0C              JM    SUBNG
0C92 34                    INR   M        INCREMENT M
0C93 C9                    RET   .
0C94 35              SUBNG DCR   M        ELSE DECR M
0C95 C9                    RET   .
0C96                 !
0C96 6F              FLT8  MOV   L,A      FLOAT ACCUM
0C97 26 00                 MVI   H,0
0C99 11 00 00              LXI   D,0
0C9C C3 0F 05              JMP   FLT32
0C9F                 !
0C9F                 ! FLOAT 12-BIT 2'S COMPLEMENT IN [DE]
0C9F                 ! VALUE RETURNED IS -1 TO 1
0C9F                 !
0C9F 21 00 00        FLT12 LXI   H,0      LOW 16 BITS ARE 0
0CA2 7A                    MOV   A,D      STRIP TO 12 BITS
0CA3 E6 0F                 ANI   0FH
0CA5 57                    MOV   D,A
0CA6 FE 08                 CPI   8        IF POS THEN FLOAT NOW
0CA8 DA AE 0C              JC    FLTIT
0CAB F6 F0                 ORI   0F0H     ELSE EXTEND SIGN
0CAD 57                    MOV   D,A      AND FLOAT IT
0CAE CD 0F 05        FLTIT CALL  FLT32    CDE=FLOAT(DEHL)
0CB1 79                    MOV   A,C      REMOVE EXPONENT
0CB2 D6 1B                 SUI   27       BIAS WE IMPOSED
0CB4 4F                    MOV   C,A      TO YIELD VALUE 0 TO 1
0CB5 C9                    RET   .
0CB6                 !
0CB6 0E 03           FMOV  MVI   C,FLT    MOVE 1 FLT VALUE
0CB8 7E              COPY  MOV   A,M      FROM [HL] TO [DE]
0CB9 12                    STAX  D
```

```
0CBA 23                    INX    H
0CBB 13                    INX    D
0CBC 0D                    DCR    C
0CBD C2 B8 0C              JNZ    COPY
0CC0 C9                    RET    .
0CC1                ;
0CC1                ; REFERENCE FUNCTIONS FOR ARRAYS
0CC1                ;
0CC1 2A 20 23       REFSAM LHLD   SAMPTR  ADDRESS SAMPLE(I2)
0CC4 3A 2C 23              LDA    INDX2
0CC7 C3 D6 0C              JMP    REF2
0CCA 21 AC 23       REFDIF LXI    H,DIFF  ADDRESS DIFF(I)
0CCD C3 D3 0C              JMP    REF
0CD0 21 4C 23       REFET  LXI    H,ET    ADDRESS ET(I)
0CD3 3A 2B 23       REF    LDA    INDEX   GET THE INDEX
0CD6 85             REF2   ADD    L
0CD7 6F                    MOV    L,A
0CD8 D0                    RNC
0CD9 24                    INR    H
0CDA C9                    RET    .
0CDB                ;
0CDB 2A 22 23       REFP   LHLD   PRMPTR  (HL)->PARAMETER BLOCK
0CDE 85                    ADD    L       REFER TO PARAM(A)
0CDF 6F                    MOV    L,A
0CE0 D0                    RNC
0CE1 24                    INR    H
0CE2 C9                    RET    .
0CE3                ;
0CE3                ; LOADING TRIGS
0CE3                ;
0CE3 3A 2B 23       COSINE LDA    INDEX
0CE6 CD 1D 08              CALL   COS
0CE9 C3 D4 03              JMP    LODFP
0CEC 3A 2B 23       SINE   LDA    INDEX
0CEF CD 16 08              CALL   SIN
0CF2 C3 D4 03              JMP    LODFP
0CF5 3A 2B 23       COSISQ LDA    INDEX
0CF8 CD 0F 08              CALL   COSSQ
0CFB C3 D4 03              JMP    LODFP
0CFE 3A 2B 23       SINESQ LDA    INDEX
0D01 CD 09 08              CALL   SINSQ
0D04 C3 D4 03              JMP    LODFP
0D07 3A 2B 23       SINCOS LDA    INDEX
0D0A CD FD 07              CALL   SNCS
0D0D C3 D4 03              JMP    LODFP
0D10                ;
0D10                ; DO-LOOP EXECUTOR.
0D10                ;
0D10 22 29 23       DOLOOP SHLD   FN+1    SET FUNCTION ADDRESS
0D13 97                    SUB    A
```

```
0D14 32 2C 23          STA    INDX2
0D17 32 2B 23     DOST STA    INDEX
0D1A CD 28 23          CALL   FN
0D1D 3A 2C 23          LDA    INDX2
0D20 C6 20             ADI    2*16
0D22 32 2C 23          STA    INDX2
0D25 3A 2B 23          LDA    INDEX  DO THE OTHER SIDE
0D28 C6 30             ADI    3*16   OF THE PERIOD NOW
0D2A 32 2B 23          STA    INDEX
0D2D CD 28 23          CALL   FN
0D30 3A 2C 23          LDA    INDX2
0D33 D6 1E             SUI    2*16-2
0D35 32 2C 23          STA    INDX2
0D38 3A 2B 23          LDA    INDEX  SUBTRACT BACK TO
0D3B D6 2D             SUI    3*16-3  FRONT SIDE, ADD 3
0D3D FE 30             CPI    NPOINTS*FLT/2  ONLY DO HALF
0D3F DA 17 0D          JC     DOST
0D42 C9                RET    .
0D43              NEXT EQU    $      END OF SPIN.A
0D43                   COM    NEXT
0D43              *
0D43              * SPIN FIT VARIABLES
0D43              *
0D43                   ORG    SPINRAM
2320              SAMPTR DS   2      SAMPLE POINTER
2322              PRMPTR DS   2      PARAM BLOCK POINTER
2324              AXPTR  DS   2      AHI/LO POINTER TEMP
2326              TBPTR  DS   2      TABLE POINTER TEMP
2328              FN     DS   3      FUNCTION FOR DOLOOP
232B              INDEX  DS   1      DOLOOP INDEX FOR FLT
232C              INDX2  DS   1      DOLOOP INDEX FOR SAMPLES
232D              ADOSB  DS   1      ADD OR SUB MODE
232E              NP     DS   1      TEMP FOR N
232F              FLTN1  DS   FLT    FLOAT VALUE OF N-1
2332              FTEMP  DS   FLT    TEMPORARY
2335              PTCNT  EQU  INDX2  TEMP COUNT FOR AVERAGE
2335              *
2335              AHI    DS   FLT
2338              ALO    DS   FLT
233B              BCOMP  DS   FLT    B COMPONENT
233E              CCOMP  DS   FLT    C COMPONENT
2341              SIGMA  DS   FLT
2344              N      DS   1      #POINTS ACTIVE
2345              MHI    DS   1      #HIGH GAIN ACTIVE
2346              *
2346              APJB   DS   FLT
2349              DISC   DS   FLT
234C              *
234C              ET     DS   NPOINTS*FLT  POINTS IN F.P.
23AC              DIFF   DS   NPOINTS*FLT  FIT(I) - ET(I)
```

```
240C       *
240C       * THE MATRIX FOR SOLVING
240C       *
240C       MAT     EQU    $
240C       ECH     DS     FLT     TOP ROW
240F       ECL     DS     FLT
2412       ECOS2   DS     FLT
2415       ESNCS   DS     FLT
2418       ETCOS   DS     FLT
241B       *
241B       ESH     DS     FLT     2ND ROW
241E       ESL     DS     FLT
2421       ECSSN   DS     FLT
2424       ESIN2   DS     FLT
2427       ETSIN   DS     FLT
242A       *
242A       Z2      DS     FLT     3RD ROW
242D       NLM     DS     FLT
2430       ECSLO   DS     FLT
2433       ESNLO   DS     FLT
2436       ELO     DS     FLT
2439       *
2439       FMHI    DS     FLT     LAST ROW
243C       Z1      DS     FLT
243F       ECSHI   DS     FLT
2442       ESNHI   DS     FLT
2445       EHI     DS     FLT
2448       *
2448       SCRATCH DS     FLT*4*5  SCRATCH AREA FOR SOLVING
```

```
0000                *
0000                * CRRES FLIGHT PROGRAM---MAGNETIC FIELD MANAGEMENT
0000                * WRITTEN BY PETER R HARVEY
0000                *
0000                * FILE : MAG.A VERSION 3
0000                *
0000                PSW     EQU     6       8085 SPECIFIC INFORMATION
0000                SP      EQU     6
0000                *
0000                BX      EQU     2       MULTIPLEXOR QTY ADDRESSES
0000                BY      EQU     1
0000                BZ      EQU     0
0000                HIGAIN  EQU     10H     HIGH GAIN BIT IN QTY
0000                BAMP    EQU     7       CODE FOR B AMPLIFIER
0000                BMODE   EQU     0FEH    MAG MODE COMMAND PREFIX
0000                *
0000                POSLIM  EQU     41      POSITIVE GAIN LIMIT
0000                NEGLIM  EQU     38      NEGATIVE GAIN LIMIT
0000                *
0000                        ORG     MAG
0D50 C3 62 0D               JMP     MAGINIT  INITIALIZATION
0D53 C3 96 0D               JMP     MAGFRAME  MINOR FRAME SYNC
0D56 C3 A7 0D               JMP     MAGGAIN  GAIN DECISIONS
0D59 C3 EA 0D               JMP     MAGSAMP  SAMPLE TIME
0D5C C3 F7 0D               JMP     MAGENCD  BUFFERING TIME
0D5F C3 85 0D               JMP     MAGTELEM  TELEMETRY TIME
0D62                *
0D62                * INITIALIZE THE B-FIELD PACKAGE
0D62                *
0D62 21 00 FE       MAGINIT LXI     H,BMODE*256  BMODE(0)
0D65 CD 7D 0D               CALL    MAGCMD
0D68 21 26 29               LXI     H,POSLIM*256+NEGLIM
0D6B 22 A6 24               SHLD    LIMITS
0D6E                *
0D6E 97             MAGSYNC SUB     A       RESET THE
0D6F 32 2F 21               STA     SMPCNT  SAMPLE STATE COUNTER
0D72 3E FD                  MVI     A,LGBUF*256/128-3  SET NIBBLE ADDRESS
0D74 32 2C 21               STA     LGPTR  LESS 3 NIBBLES
0D77 3E 07                  MVI     A,AGBUF*256/128-3  FOR BOTH LOW AND
0D79 32 2D 21               STA     AGPTR  AUTO GAIN POINTERS
0D7C C9                     RET     .
0D7D                *
0D7D                * PERFORM THE BMODE COMMAND
0D7D                *  ON ENTRY: [HL] = 16-BIT COMMAND. (A==L)
0D7D                *
0D7D 32 33 21       MAGCMD  STA     MODE   REMEMBER THE MODE
0D80 3E 07                  MVI     A,BAMP  SET THE B AMPLIFIER ON/OFF
0D82 C3 CC 02               JMP     SETMUX  BY A I/O IN THE LSB OF L
0D85                *
```

```
0D85             * B-FIELD TELEMETRY OUTPUT ROUTINES.
0D85             *  ON ENTRY: A= 0 FOR BYTE, 1 FOR WORD
0D85             *  ON EXIT : [L]=BYTE VALUE, [HL]=WORD VALUE
0D85             *
0D85 B7          MAGTEL ORA    A      IF WORD REQ'D
0D86 C4 8F 0D           CNZ    READ   THEN GET 2 ELSE 1
0D89 5D                 MOV    D,E
0D8A CD 8F 0D           CALL   READ
0D8D EB                 XCHG   .
0D8E C9                 RET    .
0D8F             *
0D8F 21 2E 21    READ   LXI    H,OTPTR  RETURN E=MEM[++OTPTR]
0D92 34                 INR    M
0D93 6E                 MOV    L,M
0D94 5E                 MOV    E,M
0D95 C9                 RET
0D96             *
0D96             * MINOR FRAME SYNC.
0D96             *  ON ENTRY: A=FRAME NUMBER
0D96             *
0D96 E6 03       MAGFRAME ANI  3      IF FRAME 0 MOD 4
0D98 CA A1 0D           JZ     RESOUT
0D9B FE 02              CPI    2
0D9D CA 6E 0D           JZ     MAGSYNC
0DA0 C9                 RET
0DA1             *
0DA1 3E FF       RESOUT MVI    A,LGBUF*256/256-1  RESET THE
0DA3 32 2E 21           STA    OTPTR  PREINC'D OUTPUT PTR
0DA6 C9                 RET
0DA7             *
0DA7             * MAG GAIN DECISION TIME.
0DA7             *
0DA7 01 E7 0D    MAGGAIN LXI   B,GNSAMPS  SAMPLE THE TRIPLET
0DAA 11 37 21           LXI    D,BXL
0DAD CD 84 0E           CALL   TRIPLET
0DB0             *
0DB0 11 E7 0D           LXI    D,GNSAMPS  LOW GAIN BY DEFAULT
0DB3 21 34 21           LXI    H,BXQTY
0DB6 0E 03              MVI    C,3
0DB8 D7                 RST    COPY/8
0DB9             *
0DB9 01 34 21           LXI    B,BXQTY  AND DECIDE 3 GAINS
0DBC 2A 37 21           LHLD   BXL
0DBF CD CB 0D           CALL   DECIDE
0DC2 2A 39 21           LHLD   BYL
0DC5 CD CB 0D           CALL   DECIDE
0DC8 2A 3B 21           LHLD   BZL
0DCB             *
0DCB             * DECIDE WHICH GAIN TO USE
0DCB             *
```

```
0DCB EB          DECIDE  XCHG    .       PUT VALUE IN [DE]
0DCC 2A A6 24            LHLD    LIMITS  H=POS. L = NEG LIMIT
0DCF 7A                  MOV     A,D     IF SMALL POSITIVE, GO
0DD0 B7                  ORA     A
0DD1 CA DC 0D            JZ      CHECK
0DD4 D6 0F               SUI     0FH     IF NOT SMALL NEGATIVE, SET LOW
0DD6 C2 E5 0D            JNZ     LOW
0DD9 93                  SUB     E       INVERT VALUE
0DDA 5F                  MOV     E,A
0DDB 65                  MOV     H,L     USE NEG LIMIT
0DDC             *
0DDC 7B          CHECK   MOV     A,E     IF VAL>= LIMIT, USE LOW GAIN
0DDD BC                  CMP     H
0DDE D2 E5 0D            JNC     LOW
0DE1 0A                  LDAX    B       USE HIGH GAIN
0DE2 F6 10               ORI     HIGAIN
0DE4 02                  STAX    B
0DE5 03          LOW     INX     B
0DE6 C9                  RET
0DE7             *
0DE7 02          GNSAMPS DB      BX
0DE8 01                  DB      BY
0DE9 00                  DB      BZ
0DEA             *
0DEA             * SAMPLE TIME. USE THE SAMPLE COUNT TO DETERMINE
0DEA             * WHICH MAG SAMPLES TO TAKE AND WHEN TO SHIFT, ETC.
0DEA             *
0DEA 21 2F 21    MAGSAMP LXI     H,SMPCNT  SMPCNT++
0DED 34                  INR     M
0DEE 01 34 21            LXI     B,BXQTY  SAMPLE AUTOGAINS
0DF1 11 3D 21            LXI     D,BXSAMP
0DF4 C3 84 0E            JMP     TRIPLET
0DF7             *
0DF7             * MAG ENCODING TIME
0DF7             *
0DF7 3A 2F 21    MAGENCD LDA     SMPCNT  IF 1ST SAMPLE TAKEN
0DFA FE 01               CPI     1
0DFC CC 43 0E            CZ      SAVLOW
0DFF             *
0DFF 2A 3D 21            LHLD    BXSAMP  SAVE AUTOGAIN VALUES
0E02 CD 58 0E            CALL    AGSTORE
0E05 2A 3F 21            LHLD    BYSAMP
0E08 CD 58 0E            CALL    AGSTORE
0E0B 2A 41 21            LHLD    BZSAMP
0E0E CD 58 0E            CALL    AGSTORE
0E11             *
0E11 21 30 21            LXI     H,TMPX  STORE THE THREE GAINS
0E14 11 34 21            LXI     D,BXQTY
0E17 CD 77 0E            CALL    GAINSET
0E1A CD 77 0E            CALL    GAINSET
```

```
0E1D CD 77 0E      0150          CALL   GAINSET
0E20               0151 *
0E20 3A 2F 21      0152          LDA    SMPCNT  IF AFTER THE 1ST SAMPLE
0E23 FE 01         0153          CPI    1       BUFFER THE MODE INFO
0E25 CA 38 0E      0154          JZ     PUTMODE
0E28 FE 08         0155          CPI    8       ON THE 8TH SAMPLE, PUT OUT
0E2A C0            0156          RNZ    .       THE GAINS
0E2B 2A 30 21      0157          LHLD   TMPX
0E2E 22 29 21      0158          SHLD   GAINX
0E31 3A 32 21      0159          LDA    TMPZ
0E34 32 2B 21      0160          STA    GAINZ
0E37 C9            0161          RET    .
0E38               0162 *
0E38 21 04 21      0163 PUTMODE LXI    H,MDBITS
0E3B 3A 33 21      0164          LDA    MODE
0E3E E6 0F         0165          ANI    0FH
0E40 B6            0166          ORA    M
0E41 77            0167          MOV    M,A
0E42 C9            0168          RET    .
0E43               0169 *
0E43 2A 37 21      0170 SAVLOW  LHLD   BXL     SAVE ALL LOW GAIN VERSIONS
0E46 CD 52 0E      0171          CALL   LGSTORE
0E49 2A 39 21      0172          LHLD   BYL
0E4C CD 52 0E      0173          CALL   LGSTORE
0E4F 2A 3B 21      0174          LHLD   BZL
0E52 11 2C 21      0175 LGSTORE LXI    D,LGPTR
0E55 C3 5B 0E      0176          JMP    MAGSTORE
0E58               0177 *
0E58               0178 * MAG STORE MECHANISM.
0E58               0179 * ON ENTRY: [DE]->STORAGE POINTER
0E58               0180 *           [HL]= 12-BIT VALUE TO STORE
0E58               0181 *
0E58 11 2D 21      0182 AGSTORE LXI    D,AGPTR
0E5B 1A            0183 MAGSTORE LDAX  D       UPDATE THE BUFFER POINTER
0E5C C6 03         0184          ADI    3       BY THE # NIBBLES
0E5E 12            0185          STAX   D
0E5F B7            0186          ORA    A
0E60 1F            0187          RAR    .       DIVIDE TO GET #BYTES
0E61 5F            0188          MOV    E,A     [DE]->BUFFER
0E62 DA 6E 0E      0189          JC     ODD
0E65 29            0190          DAD    H       ON EVEN STORES, LEFT ADJUST
0E66 29            0191          DAD    H       THE 12-BIT VALUE
0E67 29            0192          DAD    H
0E68 29            0193          DAD    H
0E69 EB            0194          XCHG   .
0E6A 72            0195          MOV    M,D
0E6B 23            0196          INX    H
0E6C 73            0197          MOV    M,E
0E6D C9            0198          RET    .
0E6E               0199 *
```

```
0E6E EB            ODD     XCHG   .       [DE]=VALUE, HL->BUFFER
0E6F 7A                    MOV    A,D     "OR" THESE BITS INTO BUFFER
0E70 E6 0F                 ANI    0FH     ON THE ODD STORES
0E72 B6                    ORA    M
0E73 77                    MOV    M,A
0E74 23                    INX    H
0E75 73                    MOV    M,E
0E76 C9                    RET    .
0E77               *
0E77               * STORE GAIN BIT FROM QTY IN MEM[DE]
0E77               *
0E77 1A            GAINSET LDAX   D       IF HIGAIN, SET CARRY
0E78 E6 10                 ANI    HIGAIN  ELSE CLEAR IT
0E7A CA 7E 0E              JZ     MG1
0E7D 37                    STC
0E7E 7E            MG1     MOV    A,M     PUT CARRY INTO MEM[HL]
0E7F 17                    RAL
0E80 77                    MOV    M,A
0E81 13                    INX    D
0E82 23                    INX    H
0E83 C9                    RET
0E84               *
0E84               * SAMPLE AND STORE A TRIPLET
0E84               *
0E84 CD 8A 0E      TRIPLET CALL   S1
0E87 CD 8A 0E              CALL   S1
0E8A 0A            S1      LDAX   B       GET QTY
0E8B 03                    INX    B
0E8C CD E6 00              CALL   SAMPLE
0E8F EB                    XCHG
0E90 73                    MOV    M,E
0E91 23                    INX    H
0E92 72                    MOV    M,D
0E93 23                    INX    H
0E94 EB                    XCHG   .
0E95 C9                    RET
0E96 00         V          DB     257     END-OF-MAG
0E97               *
0E97               * ENTER COMMAND VECTOR IN TABLE
0E97               *
0E97                       ORG    0F8H/4+CMDTAB
007E 70 0D                 DW     MAGCMD
0080               *
0080               * RAM SECTION
0080               *
0080                       ORG    MAGRAM
2100               LGBUF   DS     3*12+4/8  LOW GAIN BUFFER
2105               MDBITS  EQU    $-1     MODE BITS
2105               AGBUF   DS     8*3*12/8  AUTO GAIN BUFFER
2129               GAINX   DS     1       GAIN BYTES FOR X,Y,Z
```

```
212A  GAINY  DS     1
212B  GAINZ  DS     1
212C  *
212C  LGPTR  DS     1     LOW GAIN POINTER
212D  AGPTR  DS     1     AUTOGAIN POINTER
212E  OTPTR  DS     1     OUTPUT POINTER
212F  SMPCNT DS     1     SAMPLE COUNTER
2130  TMPX   DS     1     TEMP GAINS FOR X,Y,Z
2131  TMPY   DS     1
2132  TMPZ   DS     1
2133  MODE   DS     1     MODE BYTE (4 BITS)
2134  BXQTY  DS     1
2135  BYQTY  DS     1
2136  BZQTY  DS     1
2137  BXL    DS     2
2139  BYL    DS     2
213B  BZL    DS     2
213D  BXSAMP DS     2
213F  BYSAMP DS     2
2141  BZSAMP DS     2
2143  *
2143  * DEFINE WHERE MAG SAMPLES ARE
2143  *
2143  MAGDTA EQU    BXQTY
2143         COM    MAGDTA
2143  *
2143  * PUT GAIN LIMITS IN DSC (OTHER[3,4])
2143  *
2143  LIMITS EQU    024A6H
```

```
0000            *
0000            * CRRES FLIGHT PROGRAM---PLASMA DATA MANAGEMENT
0000            * WRITTEN BY PETER P HARVEY
0000            *
0000            * FILE : PLA.A  VERSION 6 (DEC 88)
0000            *
0000            PSW     EQU    6       8085 SPECIFIC INFORMATION
0000            SP      EQU    6
0000            *
0000            HIGAIN  EQU    10H     HIGAIN INDICATOR
0000            BYAMP   EQU    7       SPIN AXIS MEASUREMENT AMPLIFIER
0000            *
0000            * SHIFT REGISTER DEFINITION
0000            *
0000            PVALID  EQU    20H     PLA CALCULATION VALID IF 1
0000            PDISABL EQU    80H     PLA PACKAGE DISABLED
0000            PSEND   EQU    40H     PLA SENDING
0000            LPMODE  EQU    30H     LP INSTRUMENT INFO
0000            LPFREQ  EQU    0FH     LP SAMPLING FREQUENCY
0000            *
0000            MANT    EQU    3FH     MANTISSA PART OF RESULT
0000            SIGN    EQU    40H     SIGN OF RESULT
0000            OVER    EQU    80H     OVERFLOW ERROR BIT
0000            *
0000            * ENTRY POINTS
0000            *
0000                    ORG    PLA
0E98 C3 A8 0E           JMP    PLAINIT  INITIALIZATION
0E9B C3 30 0E           JMP    PLASAMP  SAMPLE TIME
0E9E            *
0E9E            * RETURN DIGITAL STATUS
0E9E            *
0E9E 3A 58 21   PLADSC  LDA    PMODE  SHOW THE MODE PART
0EA1 E6 3F              ANI    LPMODE+LPFREQ
0EA3 21 59 21           LXI    H,PSTAT  AND INTERNAL STATUS
0EA6 B6                 ORA    M
0EA7 C9                 RET
0EA8            *
0EA8            * INITIALIZE THE PLASMA  PACKAGE
0EA8            *
0EA8 21 50 21   PLAINIT LXI    H,PLARAM  CLEAR ALL VARS
0EAB 0E 0C              MVI    C,BYOFF-PLARAM
0EAD CF                 RST    ZERO/8
0EAE 11 F3 0F           LXI    D,INIOFF  COPY INIT OFFSETS
0EB1 0E 10              MVI    C,OFFEND-INIOFF
0EB3 D7                 RST    COPY/8
0EB4 C9                 RET
0EB5            *
0EB5            * PERFORM PACKAGE COMMANDS
```

```
0EB5                    *
0EB5 32 56 21   PLACMD  STA   PMODE
0EB8 07                 RLC   .      IF ALG DISABLE/ENABLE SET IT
0EB9 D0                 RNC
0EBA E6 80              ANI   PDISABL
0EBC 32 59 21           STA   PSTAT
0EBF C9                 RET
0EC0                    *
0EC0                    * SAMPLE TIME
0EC0                    *
0EC0 3A 59 21   PLASAMP LDA   PSTAT  IF DISABLED, RETURN
0EC3 E6 80              ANI   PDISABL
0EC5 C0                 RNZ
0EC6 C5                 PUSH  B
0EC7 21 5B 21           LXI   H,TIMER  ALTERNATE BETWEEN PART1 AND 2
0ECA 34                 INR   M
0ECB 7E                 MOV   A,M
0ECC 0F                 RRC
0ECD D2 F4 0E           JNC   PART2
0ED0                    *
0ED0 CD 24 0F           CALL  CVTBX  CONVERT BX INTO nT
0ED3 22 50 21           SHLD  ASCBX  SAVE ABS( S/C BX )
0ED6 32 56 21           STA   SGNBX
0ED9                    *
0ED9 CD 3C 0F           CALL  CVTBY  CONVERT BY INTO nT
0EDC 22 54 21           SHLD  ASCBZ  SAVE ABS( S/C BZ )
0EDF 32 57 21           STA   SGNBZ
0EE2                    *
0EE2 4F                 MOV   C,A    SAVE ITS SIGN
0EE3 3E 0B              MVI   A,11   MULTIPLY BY 11/256 (.044 APPX )
0EE5 EB                 XCHG
0EE6 CD 0A 06           CALL  MU21
0EE9 6C                 MOV   L,H
0EEA 67                 MOV   H,A
0EEB 79                 MOV   A,C
0EEC CD D5 0F           CALL  APPLY  APPLY SIGN
0EEF 22 52 21           SHLD  ASCBY  SAVE (11/256)*SCBY
0EF2 C1                 POP   B
0EF3 C9                 RET
0EF4                    *
0EF4 CD 30 0F   PART2   CALL  CVTBZ  CONVERT BZ INTO nT
0EF7 CD D5 0F           CALL  APPLY  [HL]=SIGNED SCBZ
0EFA EB                 XCHG  .      ADD IT TO SAVED VAL
0EFB 2A 52 21           LHLD  ASCBY
0EFE 19                 DAD   D
0EFF CD D4 0F           CALL  ABS16  TAKE ABSOLUTE VALUE
0F02                    *
0F02 97                 SUB   A      CLEAR "PSEND" BIT
0F03 32 59 21           STA   PSTAT
0F06 C1                 POP   B
```

```
0F07 29                DAD    H      IF ABS(SCBY)*8 > ABS(SCBX) RETURN
0F08 D8                RC
0F09 29                DAD    H
0F0A D8                RC
0F0B 29                DAD    H
0F0C D8                RC
0F0D 3A 50 21          LDA    ASCBX
0F10 95                SUB    L
0F11 3A 51 21          LDA    ASCBX+1
0F14 9C                SBB    H
0F15 D8                RC     .
0F16              *
0F16 C5                PUSH   B
0F17 CD 44 0F          CALL   ZCALC  CALCULATE THE VALUE TO SEND
0F1A CD 30 03          CALL   SETFLA
0F1D C1                POP    B
0F1E 3E 40             MVI    A,PSEND  INDICATE SENDING
0F20 32 59 21          STA    PSTAT
0F23 C9                RET
0F24              *
0F24              * CONVERSION ROUTINES
0F24              *
0F24 3A 34 21     CVTBX  LDA    BXQTY  CONVERT BX INTO NANOTESLA
0F27 2A 3D 21          LHLD   BXSAMP
0F2A 01 5C 21          LXI    B,BXOFF
0F2D C3 5F 0F          JMP    CONVERT  A=SIGN, HL=MAGNITUDE
0F30              *
0F30 3A 36 21     CVTBZ  LDA    BZQTY
0F33 2A 41 21          LHLD   BZSAMP
0F36 01 68 21          LXI    B,BZOFF
0F39 C3 5F 0F          JMP    CONVERT
0F3C              *
0F3C 3E 07        CVTBY  MVI    A,BYAMP  TEST WHETHER *6 IS ON
0F3E CD E4 02          CALL   TSTMUX
0F41 CA 56 0F          JZ     AMPOFF NO: GO
0F44 01 64 21     AMPON  LXI    B,BYOFF+4  IF AMPLIFIED
0F47 CD 59 0F          CALL   CONBY  CONVERT 1ST
0F4A 4F                MOV    C,A    SAVE SIGN OF BY
0F4B 3E 27             MVI    A,39   SCALE BY 39/256
0F4D EB                XCHG   .
0F4E CD 04 06          CALL   MU21
0F51 6C                MOV    L,H
0F52 67                MOV    H,A
0F53 79                MOV    A,C
0F54 2F                CMA    .      (AMP IS INVERTING)
0F55 C9                RET
0F56              *
0F56 01 60 21     AMPOFF LXI    B,BYOFF  IF UNAMPLIFIED
0F59 3A 35 21     CONBY  LDA    BYQTY
0F5C 2A 3F 21          LHLD   BYSAMP
```

```
0F5F                 *
0F5F                 * CONVERT MAGNETOMETER SAMPLE TO NANOTESLA
0F5F                 * ON ENTRY: [A]= MUX QTY (HI OR LO GAIN)
0F5F                 *           [HL]= VALUE
0F5F                 *           [BC]->OFFSET PAIR
0F5F                 *
0F5F E6 10           CONVERT ANI    HIGAIN  IF IN LOW GAIN, MULTIPLY
0F61 3E 00                   MVI    A,0
0F63 C2 6A 0F                JNZ    CV1     BY THE GAIN FACTOR
0F66 3E 33                   MVI    A,51
0F68 03                      INX    B       AND USE THE LO OFFSET
0F69 03                      INX    B
0F6A F5              CV1     PUSH   PSW
0F6B                 *
0F6B 7C                      MOV    A,H     EXTEND 12 BITS TO 16
0F6C FE 08                   CPI    8
0F6E DA 74 0F                JC     CVPOS
0F71 F6 F0                   ORI    0F0H
0F73 67                      MOV    H,A
0F74                 CVPOS   EQU    $
0F74                 *
0F74 0A                      LDAX   B       ADD OFFSET FROM MEM[BC]
0F75 5F                      MOV    E,A
0F76 03                      INX    B
0F77 0A                      LDAX   B
0F78 57                      MOV    D,A
0F79 19                      DAD    D
0F7A                 *
0F7A 7C                      MOV    A,H
0F7B 32 5A 21                STA    TEMP    SAVE SIGN
0F7E CD D4 0F                CALL   ABS16   CONVERT TO POSITIVE
0F81 F1                      POP    PSW
0F82 54                      MOV    D,H
0F83 5D                      MOV    E,L
0F84 CC 0A 06                CZ     MU21    IF LOW GAIN, AHL=A*DE
0F87 CD 93 0F                CALL   DIV4    AHL=AHL/4
0F8A B7                      ORA    A       IF A HAS ANYTHING, OVERFLOW
0F8B 3A 5A 21                LDA    TEMP    RETURN(SIGN)
0F8E C8                      RZ     .
0F8F 21 FF 7F                LXI    H,7FFFH
0F92 C9                      RET
0F93                 *
0F93 B7              DIV4    ORA    A       SHIFT RIGHT TWICE
0F94 1F                      RAR
0F95 23                      INX    H       ROUND OFF
0F96 CD 9B 0F                CALL   SRHL
0F99 B7                      ORA    A
0F9A 1F                      RAR
0F9B 4F              SRHL    MOV    C,A
0F9C 7C                      MOV    A,H
```

```
0F9D 1F               RAR
0F9E 67               MOV    H,A
0F9F 7D               MOV    A,L
0FA0 1F               RAR
0FA1 6F               MOV    L,A
0FA2 79               MOV    A,C
0FA3 C9               RET
0FA4            *
0FA4            * TIME TO MAKE THE CALCULATION OF BZ/BX
0FA4            * ON EXIT: [HL]= VALUE TO SEND TO LEPA
0FA4            *
0FA4 2A 50 21   ZCALC LHLD   ASCBX  A=ABS(BZ/2)/ABS(BX)
0FA7 EB               XCHG
0FA8 2A 54 21         LHLD   ASCBZ
0FAB B7               ORA    A      SHIFT RIGHT
0FAC CD 9B 0F         CALL   SRHL
0FAF CD DA 0F         CALL   QDIV   A=[HL]/[DE] (8-BITS)
0FB2 3C               INR    A      ROUND OFF LAST BIT
0FB3 B7               ORA    A
0FB4 1F               RAR    .
0FB5            *
0FB5 FE 40            CPI    MANT+1  IF GREATER THAN MANTISSA
0FB7 DA BE 0F         JC     MNTOK  CAN BE.
0FBA E6 3F            ANI    MANT   MASK MANT BITS
0FBC F6 80            ORI    OVER   THEN SET THE OVERFLOW
0FBE 5F         MNTOK MOV    E,A    SAVE THIS IN E
0FBF            *
0FBF 3A 56 21         LDA    SGNBX  COMPUTE THE SIGN DIFFERENCE
0FC2 21 57 21         LXI    H,SGNBZ
0FC5 AE               XRA    M
0FC6 7B               MOV    A,E    AND INSERT THE PLA SIGN
0FC7 F2 CC 0F         JP     PLAPOS
0FCA F6 40            ORI    SIGN
0FCC            PLAPOS EQU   $
0FCC 6F               MOV    L,A    SEND MODE AND RESULTS TO
0FCD            *
0FCD 3A 58 21         LDA    PMODE  THE LEPA INSTRUMENT
0FD0 F6 80            ORI    PVALID  SIGNIFY VALID BZ/BX
0FD2 67               MOV    H,A
0FD3 C9               RET    .
0FD4            *
0FD4            * ABSOLUTE VALUE
0FD4            *
0FD4 7C         ABS16 MOV    A,H    IF POSITIVE, RETURN
0FD5 B7         APPLY ORA    A
0FD6 F0               RP     .
0FD7 C3 8B 00         JMP    NEG16  ELSE [HL]= -[HL]
0FDA            *
0FDA            * QUICK DIVIDER
0FDA            * A = [HL]/[DE] TO 8 BITS
```

```
0FDA                 0250 *
0FDA 7A              0251 QDIV   MOV   A,D     [BC]= -DIVISOR
0FDB 2F              0252        CMA
0FDC 47              0253        MOV   B,A
0FDD 7B              0254        MOV   A,E
0FDE 2F              0255        CMA
0FDF 4F              0256        MOV   C,A
0FE0 03              0257        INX   B
0FE1                 0258 *
0FE1 3E 01           0259        MVI   A,1     SET MARKER FOR 8TH SHIFT
0FE3 C3 E9 0F        0260        JMP   QDIV1
0FE6                 0261 *
0FE6 17              0262 QDIVN  RAL   .       SHIFT RESULT INTO ACCUM
0FE7 D8              0263        RC    .       WHEN MARKER SETS CRY, RETURN
0FE8 29              0264        DAD   H       SHIFT REMAINDER
0FE9 54              0265 QDIV1  MOV   D,H     SAVE A COPY OF REMAINDER
0FEA 5D              0266        MOV   E,L
0FEB 09              0267        DAD   B       REM=REM-DIVISOR
0FEC DA E6 0F        0268        JC    QDIVN   IF POS, RESULT=1
0FEF EB              0269        XCHG  .       ELSE RESULT=0, GET OLD
0FF0 C3 E6 0F        0270        JMP   QDIVN   REMAINDER BACK AGAIN
0FF3                 0271 *
0FF3 A5 FF           0272 INIOFF DW    -91     BXH,BXL OFFSETS
0FF5 FC FF           0273        DW    -4
0FF7 F4 FF           0274        DW    -12     BY AMP OFF
0FF9 FE FF           0275        DW    -2
0FFB 60 00           0276        DW    +96     BY AMP ON
0FFD 01 00           0277        DW    +1
0FFF EA FF           0278        DW    -22     BZ
1001 FE FF           0279        DW    -2
1003                 0280 OFFEND EQU   $
1003 00          V   0281        DB    256     END OF PLA MODULE
1004                 0282 *
1004                 0283 * ENTER COMMAND VECTOR INTO TABLE
1004                 0284 *
1004                 0285        ORG   0D8H/4+CMDTAB
0076 B5 0E           0286        DW    PLACMD
0078                 0287 *
0078                 0288 * VARIABLES
0078                 0289 *
0078                 0290        ORG   PLARAM
2150                 0291 ASCBX  DS    2       ABSOLUTE VALUES OF S/C BX,BY,BZ
2152                 0292 ASCBY  DS    2
2154                 0293 ASCBZ  DS    2
2156                 0294 SGNBX  DS    1       SIGN OF BX
2157                 0295 SGNBZ  DS    1       SIGN OF BZ
2158                 0296 PMODE  DS    1       MODE INFORMATION
2159                 0297 PSTAT  DS    1       PACKAGE STATUS
215A                 0298 TEMP   DS    1
215B                 0299 TIMER  DS    1
```

```
215C            0300 BXOFF  DS     4
2160            0301 BYOFF  DS     4*2
2168            0302 BZOFF  DS     4
216C            0303 ENDPLA EQU    $
216C            0304 *
216C            0305 * EXTERNAL DATA
216C            0306 *
216C            0307        ORG    MAGDTA
2134            0308 BXQTY  DS     1       MAG MUX ADDRESSES
2135            0309 BYQTY  DS     1
2136            0310 BZQTY  DS     1
2137            0311 BXL    DS     2       AND LOW SAMPLES
2139            0312 BYL    DS     2
213B            0313 BZL    DS     2
213D            0314 BXSAMP DS     2       AND AUTOGAIN SAMPLES
213F            0315 BYSAMP DS     2
2141            0316 BZSAMP DS     2
```

```
0000                 ;
0000                 ; CRRES FLIGHT SOFTWARE---BURST TRIGGERING CONTROL
0000                 ; WRITTEN BY PETER R HARVEY
0000                 ;
0000                 ; FILE BUR.A
0000                 ;
0000                 PSW    EQU    6
0000                 SP     EQU    6
0000                 ;
0000                 ; BURST PROCESSOR COMMANDS
0000                 ;
0000                 BGO    EQU    0B400H  START SAMPLING
0000                 BSTOP  EQU    0B500H  STOP SAMPLING
0000                 BPAUSE EQU    0B600H  PAUSE SAMPLING
0000                 BCONT  EQU    0B700H  CONTINUE SAMPLING
0000                 BPLAY  EQU    0B800H  BEGIN PLAYBACK
0000                 BRESET EQU    0B900H  RESET BURST
0000                 ;
0000                        ORG    BUR
1008 C3 17 10               JMP    BURINIT  INITIALIZATION
100B C3 65 10               JMP    BURSAMP  SAMPLING
100E C3 38 11               JMP    BURPLAY  PLAYBACK
1011                 ;
1011                 ; RETURN BURST DIGITAL STATUS OF MODULE
1011                 ;
1011 21 70 21        BURDSC LXI    H,BURRAM
1014 DF                     RST    REF/8
1015 7E                     MOV    A,M
1016 C9                     RET
1017                 ;
1017                 ; INITIALIZE THE BURST TRIGGERING PACKAGE
1017                 ;
1017 97              BURINIT SUB   A      CLEAR THE PLAYBACK REQUEST
1018 32 70 21               STA    MODFREQ
101B 21 10 00               LXI    H,16   SET DEFAULT DURATION=4 SECONDS
101E 22 72 21               SHLD   WAITTIME
1021 C3 44 03               JMP    RBURST  RESET BURST TO START
1024                 ;
1024                 ; PERFORM TRIGGERING COMMANDS
1024                 ;
1024 11 96 21        ALGCMD LXI    D,ALGPARMS
1027 C3 2D 10               JMP    SETPARM
102A 11 71 21        CTLCMD LXI    D,CTLPARMS
102D 7C              SETPARM MOV   A,H    PUT THE VALUE INTO REGISTER
102E E6 03                  ANI    3      DESCRIBED BY THE REG FIELD
1030 EB                     XCHG   .
1031 DF                     RST    REF/8
1032                 ;
1032 97                     SUB    A      CLEAR TRIG MODE
```

```
1033 32 71 21           STA    BTMODE
1036 73                 MOV    M,E    SET NEW PARAMETER
1037               *
1037 3A 71 21           LDA    BTMODE  IF ANY TRIGGER SELECTED
103A E6 07              ANI    ALGBITS  THEN RESET IT.
103C C8                 RZ     .
103D 3E 40         START MVI   A,R1
103F CD AE 11           CALL   SETMODE
1042 B7                 ORA    A
1043 C9                 RET
1044               *
1044 97            BURCMD SUB  A      CLEAR TRIGGER ON ANY DIRECT COMMAND
1045 32 71 21           STA    BTMODE
1048 CD 4D 10           CALL   DOBUR  EXECUTE COMMAND
104B B7                 ORA    A      RETURN NO CARRY
104C C9            EXIT RET
104D               *
104D 7C            DOBUR MOV   A,H    GET THE 0BXH COMMAND
104E FE B9              CPI    BRESET/256  IF RESET, DO IT
1050 CA 17 10           JZ     BURINIT  AND CLEAR THIS PACKAGE
1053 FE B4              CPI    BGO/256
1055 CA 3D 10           JZ     START
1058 FE B5              CPI    BSTOP/256
105A CA E9 10           JZ     TRIGGER
105D FE B8              CPI    BPLAY/256
105F CA 1F 11           JZ     STPLAY
1062 C3 53 03           JMP    SEND
1065               *
1065               * SAMPLE CONDITIONS TO DECIDE IF WE SHOULD BURST
1065               * OR NOT.
1065               *
1065 CD A8 11      BURSAMP CALL GETMODE  RETRIEVE SAMPLING MODE
1068 0F                 RRC
1069 0F                 RRC
106A 0F                 RRC
106B E6 0E              ANI    7*2
106D 21 73 10           LXI    H,BSVECT
1070 C3 D5 10           JMP    BEX
1073               *
1073 4C 10         BSVECT DW   EXIT   OFF
1075 CB 10              DW     BURTST  SEARCH
1077 01 11              DW     BURCOL  COLLECT
1079 16 11              DW     BURWAIT  WAIT
107B 8D 10              DW     BR1
107D A0 10              DW     BR2
107F B9 10              DW     BR3
1081 83 10              DW     BR0
1083               *
1083 21 79 21      BR0  LXI    H,TEMP   DELAYED BR1
1086 35                 DCR    M
```

```
1087 C0                     RNZ
1088 3E 40                  MVI    A,R1
108A C3 AE 11               JMP    SETMODE
108D                   *
108D                   * RESET TRIGGERING (STAGES 1-3)
108D                   *
108D 21 00 B4          BR1  LXI    H,BGO  START THE BURST GOING
1090 CD 53 03               CALL   SEND
1093 97                     SUB    A      ZERO THE TEMP
1094 32 79 21               STA    TEMP
1097 3A 71 21               LDA    BTMODE  SAVE WHICH TRIGGER
109A 32 B6 21               STA    TRIGR
109D C3 C1 11               JMP    INCMODE
10A0                   *
10A0 CD 84 03          BR2  CALL   RECEIVE  GET THE 3 WORD INFO
10A3 C8                     RZ     .      (IF NOT READY, TRY NEXT TIME)
10A4 3A 70 21               LDA    MODFREQ  1ST IS THE "REAL FREQ" INFO
10A7 E6 F0                  ANI    -1-FREQBITS
10A9 B5                     ORA    L
10AA 32 70 21               STA    MODFREQ
10AD                   *
10AD CD 84 03          BR2W CALL   RECEIVE  NEXT IS THE TOTAL DURATION
10B0 CA AD 10               JZ     BR2W
10B3 22 B0 21               SHLD   DURATION  WHICH THE BURST
10B6 C3 C1 11               JMP    INCMODE
10B9                   *
10B9 CD 84 03          BR3  CALL   RECEIVE  CAN HOLD IN ITS CURRENT
10BC C8                     RZ
10BD 22 B2 21               SHLD   DURATION+2  STATE.
10C0 21 B7 21               LXI    H,ST   SAVE TIME WHEN THINGS BEGAN
10C3 CD E0 11               CALL   SAVETIME
10C6 3E 10                  MVI    A,SEARCH  AND BEGIN SEARCHING
10C8 C3 AE 11               JMP    SETMODE
10CB                   *
10CB                   * SEARCH PHASE.  USE COMMANDED ALGORITHM FOR SEARCH
10CB                   *
10CB 3A 71 21          BURTST LDA  BTMODE  GET THE ALGORITHM #
10CE E6 07                  ANI    ALGBITS
10D0 C8                     RZ     .
10D1 87                     ADD    A
10D2 21 D9 10               LXI    H,BATABLE-2  REFERENCE ALGORITHM
10D5 DF                BEX  RST    REF/8
10D6 7E                     MOV    A,M
10D7 23                     INX    H
10D8 66                     MOV    H,M
10D9 6F                     MOV    L,A
10DA E9                     PCHL   .
10DB                   *
10DB E9 10             BATABLE DW  TRIGGER THE TRIGGERING ALGORITHM LIST
10DD 73 11                  DW     VALCHK
```

```
10DF 9F 11              DW     MAGCHK
10E1 9F 11              DW     RAMALG
10E3 9F 11              DW     RAMALG
10E5 9F 11              DW     RAMALG
10E7 9F 11              DW     RAMALG
10E9               *
10E9               * EVENT TRIGGER
10E9               *
10E9 21 8C 21      TRIGGER LXI  H,VT   SAVE THE TIME OF THE EVENT
10EC CD E0 11           CALL   SAVETIME
10EF 2A 72 21           LHLD   WAITTIME  SET DELAY FOR THAT COMMANDED
10F2 22 7B 21           SHLD   DTIME+1  AS OPPOSED TO THE
10F5 97                 SUB    A      MEMORY CAPACITY
10F6 32 7A 21           STA    DTIME
10F9 32 7D 21           STA    DTIME+3
10FC 3E 20              MVI    A,COLLECT  MODE="COLLECT"
10FE C3 AE 11           JMP    SETMODE
1101               *
1101               * COLLECTION PHASE
1101               *
1101 CD C9 11      BURCOL CALL   CLKTICK  COUNT 1 CLOCK TIME
1104 D0                 RNC    .      IF NOT TIME TO QUIT, RETURN
1105 21 00 B5           LXI    H,BSTOP  COMMAND THE BURST PROCESSOR
1108 CD 53 03           CALL   SEND   TO STOP NOW.
110B 21 91 21           LXI    H,ET   SAVE THE END TIME
110E CD E0 11           CALL   SAVETIME
1111 3E 30              MVI    A,WAIT
1113 C3 AE 11           JMP    SETMODE
1116               *
1116               * WAIT FOR BURST CPU TO PROCESS ITS MEMORY
1116               *
1116 CD 84 03      BURWAIT CALL  RECEIVE  GET THE READY FOLLOWING A STOP
1119 C8                 RZ     .      IF NOT THERE, TRY NEXT TIME
111A 3E 00              MVI    A,OFF  TURN OFF SAMPLING SECTION
111C CD AE 11           CALL   SETMODE
111F               *
111F               * START PLAYBACK SEQUENCE IN STANDARD FORMAT
111F               *
111F 21 00 B8      STPLAY LXI   H,BPLAY  COMMAND THE BURST
1122 CD 53 03           CALL   SEND   TO PLAY BACK
1125 3E B1              MVI    A,0B1H  SET THE FORMAT CODE
1127 32 85 21           STA    FORMAT
112A 3E 85              MVI    A,HEADR*256/256  PUT OFFSET ADDR
112C 32 84 21           STA    HDINX  INTO THE HEADER INDEX
112F 3A 70 21           LDA    MODFREQ  START UP THE PLAYBACK
1132 F6 80              ORI    PLAYBIT  BY TURNING ON ITS BIT
1134 32 70 21           STA    MODFREQ
1137 C9                 RET
1138               *
1138               * RETRIEVE PLAYBACK DATA FOR THE TELEMETRY SYSTEM.
```

```
1138                *
1138 21 84 21       BURPLAY LXI   H,HDINX  IF HDINX < HDEND,
113B 7E                     MOV   A,M    THEN OUTPUT HEADER INFO
113C FE 9A                  CPI   XHEADR*256/256
113E DA 6D 11               JC    BPHEAD
1141 CD 84 03               CALL  RECEIVE  ELSE GRAB NEXT DATA
1144 C0                     RNZ   .      AND RETURN(HL) IF THERE
1145                *
1145 3A 70 21       ENPLAY  LDA   MODFREQ  IF THE PLAYBACK QUIT A WHILE
1148 E6 80                  ANI   PLAYBIT  BACK, JUST RETURN(0)
114A CA 69 11               JZ    ENPRO
114D 3A 70 21               LDA   MODFREQ  ELSE REMOVE PLAYBACK REQUEST
1150 E6 7F                  ANI   -1-PLAYBIT
1152 32 70 21               STA   MODFREQ
1155                *
1155 3E 10                  MVI   A,512/32  DELAY 1/2 SECOND
1157 32 79 21               STA   TEMP
115A 3A 71 21               LDA   BTMODE  IF AUTO-SEARCH MODE, RESTART
115D E6 80                  ANI   AUTOSEARCH  THE SEARCHING
115F 3E 70                  MVI   A,R0
1161 C2 66 11               JNZ   ENPNM
1164 3E 00                  MVI   A,OFF  ELSE TURN OFF
1166 CD AE 11       ENPNM   CALL  SETMODE
1169 21 00 00       ENPRO   LXI   H,0    RETURN(0) AS A TRAILER
116C C9                     RET
116D                *
116D 34             BPHEAD  INR   M      HDINX++ FOR NEXT TIME
116E 6F                     MOV   L,A    [HL]->THIS BYTE
116F 6E                     MOV   L,M    [HL]=BYTE
1170 26 00                  MVI   H,0
1172 C9                     RET   .
1173                *
1173                * BURST TRIGGER SECTION
1173                * VALUE CHECKING FOR SAMPLE > THRESHOLD
1173                *
1173 3A 96 21       VALCHK  LDA   MUXAD  SAMPLE THE MULTIPLEXOR
1176 CD E6 00               CALL  SAMPLE
1179 29                     DAD   H      SCALE TO 8 BITS
117A 29                     DAD   H
117B 29                     DAD   H
117C 29                     DAD   H
117D 7C                     MOV   A,H
117E CD 8A 11               CALL  ABS
1181 67                     MOV   H,A
1182 3A 97 21               LDA   THRESHOLD  IF THRES < SAMPLE(MUXAD)
1185 BC                     CMP   H      CALL THE TRIGGER START
1186 DC E9 10               CC    TRIGGER
1189 C9                     RET
118A                *
118A B7             ABS     ORA   A
```

```
118B F0                RP
118C 2F                CMA
118D 3C                INR    A
118E C9                RET
118F              *
118F              * MAG CHECKING ALGORITHM
118F              *
118F              LCONE  EQU    40H    "LOSS CONE BIT"
118F 97           MAGCHK SUB    A      GET PLA STATUS
1190 CD 9E 0E            CALL   PLADSC  AND TRIGGER WHEN
1193 E6 40               ANI    LCONE  LOSSCONE IS BEGINNING
1195 C8                  RZ     .
1196 21 79 21            LXI    H,TEMP
1199 BE                  CMP    M
119A C8                  RZ
119B 77                  MOV    M,A
119C C3 E9 10            JMP    TRIGGER
119F              *
119F              * RAM ALGORITHM
119F              *
119F 3A 75 21     RAMALG LDA    RAMCODE  CHECK THAT IT'S LOADED
11A2 FE AA               CPI    0AAH   OK
11A4 CA 76 21            JZ     RAMCODE+1
11A7 C9                  RET
11A8              *
11A8              * UTILITIES
11A8              *
11A8 3A 70 21     GETMODE LDA   MODFREQ  RETRIEVE SAMPLING MODE
11AB E6 70               ANI    SMPBITS
11AD C9                  RET
11AE              *
11AE E6 70        SETMODE ANI   SMPBITS  SET SAMPLE MODE
11B0 E5                  PUSH   H
11B1 6F                  MOV    L,A
11B2 3A 70 21            LDA    MODFREQ  INTO THE MODE/FREQ BYTE
11B5 E6 8F               ANI    -1-SMPBITS
11B7 B5                  ORA    L
11B8 32 70 21            STA    MODFREQ
11BB 6F                  MOV    L,A    NOW FORM A PMODE COMMAND
11BC 26 D8               MVI    H,PMODE  TO TELL THE PLASMA
11BE F7                  RST    6      INSTRUMENT
11BF E1                  POP    H
11C0 C9                  RET
11C1              *
11C1 21 70 21     INCMODE LXI   H,MODFREQ  UPDATE MODE INTERNALLY
11C4 7E                  MOV    A,M    BUT DON'T TELL LEPA ABOUT IT
11C5 C6 10               ADI    SMPBITS/7  (3-BIT FIELD)
11C7 77                  MOV    M,A
11C8 C9                  RET    .
11C9              *
```

```
11C9                 0300 * CLOCK TIMER.  DECREMENTS DTIME BY 32 MILLISECONDS
11C9                 0301 * RETURNS NO CARRY IF READY.
11C9                 0302 *
11C9 11 7A 21        0303 CLKTICK LXI   D,DTIME  DECREMENT THE DELAY TIMER
11CC 21 DC 11        0304         LXI   H,P32
11CF B7              0305         ORA   A      CLEAR CARRY
11D0 CD D3 11        0306         CALL  SUB2
11D3 CD D6 11        0307 SUB2    CALL  SUB1
11D6 1A              0308 SUB1    LDAX  D
11D7 9E              0309         SBB   M
11D8 12              0310         STAX  D
11D9 13              0311         INX   D
11DA 23              0312         INX   H
11DB C9              0313         RET
11DC                 0314 *
11DC 20 00           0315 P32     DW    32
11DE 00 00           0316         DW    0
11E0                 0317 *
11E0 11 1C 20        0318 SAVETIME LXI  D,SYSCLOCK  GET 5 BYTES OF
11E3 0E 05           0319         MVI   C,5    THE SYSTEM CLOCK AND STORE
11E5 D7              0320         RST   COPY/8
11E6 C9              0321         RET
11E7 00          V   0322         DB    256
11E8                 0323 *
11E8                 0324 * ENTER COMMAND VECTORS INTO TABLE
11E8                 0325 *
11E8                 0326         ORG   0A0H/4+CMDTAB
0068 2A 10           0327         DW    CTLCMD  #A0
006A 24 10           0328         DW    ALGCMD  #A8
006C 44 10           0329         DW    BURCMD  #B0
006E 44 10           0330         DW    BURCMD  #B8
0070                 0331 *
0070                 0332 * VARIABLES
0070                 0333 *
0070                 0334         ORG   BURRAM
2170                 0335 MODFREQ DS    1      MODE AND FREQ NIBBLES (MMMMFFFF)
2171                 0336 SMPBITS EQU   70H    SAMPLING MODE BITS
2171                 0337 PLAYBIT EQU   80H    PLAYBACK BIT
2171                 0338 FREQBITS EQU  0FH    FREQUENCY BITS
2171                 0339 OFF     EQU   0      SAMPLE MODE VALUES
2171                 0340 SEARCH  EQU   10H
2171                 0341 COLLECT EQU   20H
2171                 0342 WAIT    EQU   30H
2171                 0343 R1      EQU   40H    (RESETTING STAGES)
2171                 0344 R2      EQU   50H
2171                 0345 R3      EQU   60H
2171                 0346 R0      EQU   70H
2171                 0347 *
2171                 0348 CTLPARMS EQU  $      CONTROL PARAMS (4)
2171                 0349 BTMODE  DS    1      BURST TRIGGER MODE BITS
```

```
2172    ALGBITS EQU   7        3 PRELAUNCH ALGORITHMS + OFF
2172    AUTOSEARCH EQU 80H     AUTO-SEARCH MODE IF 1
2172    PULSED EQU    40H      PULSED SAMPLING MODE IF 1
2172    *
2172    WAITTIME DS   2        BURST DURATION
2174    SPARE  DS     1
2175    *
2175    RAMCODE DS    4
2179    TEMP   DS     1
217A    DTIME  DS     4        DELAY TIMER
217E    RFREQ  DS     2        REAL FREQUENCY CODE USED BY BURST
2180    DURATION DS   4        DURATION READ BACK (MSEC)
2184    HDINX  DS     1        HEADER INDEX
2185    *
2185    * PLAYBACK HEADER STORED IN MEMORY
2185    *
2185    HEADR  EQU    $
2185    FORMAT DS     1        FORMAT CODE
2186    TRIGR  DS     1        TRIGGER ALGORITHM
2187    ST     DS     5        START TIME
218C    VT     DS     5        EVENT TIME
2191    ET     DS     5        END TIME
2196    ALGPARMS DS   4
219A    XHEADR EQU    $
219A    *
219A    MUXAD  EQU    ALGPARMS+0  TRIGGER 1 PARAMETERS
219A    THRESHOLD EQU MUXAD+1
219A    *
219A    * EXTERNAL INFORMATION
219A    *
219A    SYSCLOCK EQU  BKGRAM+0
219A    PMODE  EQU    0D8H     PMODE COMMAND CODE
219A    *
219A           ORG    PLA
0E98    PLAINIT DS    3
0E9B    PLASAMP DS    3
0E9E    PLADSC DS     3
```

```
0000                * 
0000                * CRRES FLIGHT SOFTWARE --- SPIN FIT MANAGER
0000                * WRITTEN BY PETER R HARVEY
0000                * FILE : FIT.A
0000                * 
0000                FMODE  EQU    50H     FIT MODE #
0000                V12DI  EQU    02H     DISABLE V12 FITS
0000                FITXMIT EQU   08H     TRANSMIT ENABLE
0000                FITDI  EQU    20H     DISABLE FITS
0000                * 
0000                FLT    EQU    3       3-BYTE FLOATING POINT
0000                NULL   EQU    40H     NULL FLOAT VALUE IN 2ND BYTE
0000                NSAMPS EQU    32      # POINTS PER SPIN FIT
0000                AVPTS  EQU    4       # AHI/ALO TO AVERAGE
0000                V1ANG  EQU    128+32  DEGREES FROM SUN PULSE TO START FIT
0000                V3ANG  EQU    V1ANG+128  START V34 FIT 1/2 SPIN AWAY
0000                * 
0000                V12F   EQU    8       MAIN MULTIPLEXOR QUANTITIES
0000                V34    EQU    15
0000                HIGAIN EQU    10H     HIGH GAIN INDICATOR
0000                * 
0000                V12HEADER EQU 0A1H    V12 HEADER BYTE
0000                V34HEADER EQU 0A3H    V34 HEADER BYTE
0000                FILL   EQU    0       BYTE TO USE WHEN NOT SENDING
0000                READY  EQU    0AAH    READY CODE IN V12OUT/V34OUT
0000                PLAY   EQU    0BBH    PLAYING CODE
0000                DONE   EQU    0FFH    FINISHED CODE
0000                * 
0000                       ORG    FIT
11E8 C3 F4 11              JMP    FITINIT
11EB C3 29 12              JMP    FITSMP
11EE C3 A3 12              JMP    FITTEL
11F1 C3 DD 12              JMP    FITEXEC
11F4                * 
11F4 21 B0 22       FITINIT LXI   H,FITVARS  CLEAR ALL VARS/PARAMS
11F7 0E 07                 MVI    C,FVEND-FITVARS
11F9 CF                    RST    ZERO/8
11FA                * 
11FA 21 D7 22              LXI    H,V12PRM  INIT THE V12 PARAMETERS
11FD CD 03 12              CALL   PRMINIT
1200 21 F8 22              LXI    H,V34PRM  AND THE V34 PARAMETERS
1203 11 20 13       PRMINIT LXI   D,IPARAM
1206 0E 09                 MVI    C,IPRMX-IPARAM
1208 D7                    RST    COPY/8
1209                * 
1209 11 18 40       TBLNULL LXI   D,NULL*256+TBLNG  NULL ALL TABLE ENTRIES
120C 72             CLEAR  MOV    M,D
120D 23                    INX    H
120E 1D                    DCR    E
```

```
120F C2 0C 12              JNZ    CLEAR
1212 C9                    RET
1213               *
1213               *  FIT MODE SETTING COMMAND
1213               *
1213 5F            FITCMD MOV    E,A     A=E=ENABLE BITS
1214 29                   DAD    H       SHIFT DATA BITS LEFT
1215 A5                   ANA    L       D=ENABLE AND DATA
1216 57                   MOV    D,A
1217 7B                   MOV    A,E     A=FITMODE AND (NOT ENABLE)
1218 EE 22                XRI    V12DI+FITDI
121A 21 B6 22             LXI    H,FITMODE
121D A6                   ANA    M       AND WITH THE ENABLE BITS
121E B2                   ORA    D       OR WITH ENABLED BITS
121F 77                   MOV    M,A     REPLACE
1220               *
1220 E6 02                ANI    V12DI  IF CHANGING V12, CLEAR TABLE
1222 C0                   RNZ    .
1223 21 E0 22             LXI    H,V12TBL
1226 C3 05 12             JMP    TBLNULL  (AND RETURN NO CARRY)
1229               *
1229               * SPIN SYNCHRONOUS SAMPLING.
1229               * ON ENTRY: [A]=SUN ANGLE
1229               *
1229 D6 A0         FITSMP SUI    V1ANG  REMOVE SUN SENSOR TO BOOM 1 ANGLE
122B 5F                   MOV    E,A
122C E6 07                ANI    256/NSAMPS-1  IF NOT TIME, RETURN
122E C0                   RNZ    .
122F 3A B6 22             LDA    FITMODE  IF FITS DISABLED, RETURN
1232 E6 20                ANI    FITDI
1234 C0                   RNZ    .
1235               *
1235 7B                   MOV    A,E     MANAGE BUFFERS
1236 CD 6B 12             CALL   FITSYNC
1239               *
1239 3E 0F                MVI    A,V34  SAVE A SAMPLE OF V34
123B CD 90 12             CALL   GETSAMP
123E 3A B2 22             LDA    V34IN
1241 11 30 22             LXI    D,V34BUF
1244 CD 9D 12             CALL   STORE
1247 21 B2 22             LXI    H,V34IN
124A CD 64 12             CALL   INCR
124D               *
124D 3A B6 22             LDA    FITMODE  IF V12 DISABLED, GO
1250 E6 02                ANI    V12DI
1252 C0                   RNZ    .
1253 3E 08                MVI    A,V12F  SAVE A SAMPLE OF V12
1255 CD 90 12             CALL   GETSAMP
1258 3A B0 22             LDA    V12IN
125B 11 B0 22             LXI    D,V12BUF
```

```
125E CD 9D 12      0100         CALL   STORE
1261 21 B0 22      0101         LXI    H,V12IN
1264               0102 *
1264 7E            0103 INCR    MOV    A,M
1265 C6 02         0104         ADI    2
1267 E6 7F         0105         ANI    7FH
1269 77            0106         MOV    M,A
126A C9            0107         RET
126B               0108 *
126B               0109 * FITSYNC: ON 0 DEGREES FOR EITHER BOOM, SWITCH BUFFERS
126B               0110 *
126B FE 80         0111 FITSYNC CPI    V3ANG-V1ANG  IF V34 AT 0 DEGREES
126D 21 B2 22      0112         LXI    H,V34IN
1270 CA 7E 12      0113         JZ     FSY1
1273 B7            0114         ORA    A      IF V12 AT 0 DEGREES
1274 C0            0115         RNZ    .
1275 3A B6 22      0116         LDA    FITMODE  AND NOT DISABLED
1278 E6 02         0117         ANI    V12DI
127A C0            0118         RNZ    .
127B 21 B0 22      0119         LXI    H,V12IN  RESET V12OUTPUT
127E               0120 *
127E 7E            0121 FSY1    MOV    A,M    GET THE BUFFER POINTER
127F E6 3F         0122         ANI    NSAMPS*2-1  IF FINISHED WITH BUFFER
1281 CC 89 12      0123         CZ     FBUFOK  THEN SET THE OUTPUT POINTER
1284 7E            0124         MOV    A,M    GOOD OR NOT, CLEAR THE
1285 E6 C0         0125         ANI    -NSAMPS*2  LSB'S OF THE INDEX
1287 77            0126         MOV    M,A
1288 C9            0127         RET
1289               0128 *
1289 7E            0129 FBUFOK MOV     A,M    SET OUTPUT POINTER
128A EE 40         0130         XRI    NSAMPS*2  TO THE OTHER BUFFER
128C 2C            0131         INR    L
128D 77            0132         MOV    M,A
128E 2D            0133         DCR    L
128F C9            0134         RET
1290               0135 *
1290 CD 2A 01      0136 GETSAMP CALL   AGC    D=QTY TO SAMPLE
1293 7A            0137         MOV    A,D
1294 CD E6 00      0138         CALL   SAMPLE  TAKE IT
1297 7A            0139         MOV    A,D    PUT THE GAIN BIT INTO [HL]
1298 E6 10         0140         ANI    HIGAIN
129A B4            0141         ORA    H
129B 67            0142         MOV    H,A
129C C9            0143         RET
129D               0144 *
129D EB            0145 STORE   XCHG   .      DATA TO DE
129E DF            0146         RST    REF/8
129F 73            0147         MOV    M,E
12A0 23            0148         INX    H
12A1 72            0149         MOV    M,D
```

```
12A2 C9                    RET
12A3               *
12A3               * FIT TELEMETRY OUTPUT.
12A3               * ON EXIT: [A] = DATA OR 0 FILL
12A3               *
12A3 21 B4 22      FITTEL LXI    H,COUNT  IF NO DATA REMAINING
12A6 7E                   MOV    A,M
12A7 B7                   ORA    A       THEN CHECK IF MORE READY
12A8 CA B3 12             JZ     CHKDONE
12AB 35                   DCR    M       ELSE COUNT--
12AC 21 B5 22             LXI    H,PTR  RETURN THE NEXT DATA ITEM
12AF 34                   INR    M
12B0 6E                   MOV    L,M
12B1 7E                   MOV    A,M
12B2 C9                   RET
12B3               *
12B3 21 B1 22      CHKDONE LXI   H,V12OUT  IF OUTPUT STATUS=READY
12B6 16 A1                MVI    D,V12HEADER
12B8 1E B6                MVI    E,V12RES*256/256-1
12BA 7E                   MOV    A,M
12BB FE AA                CPI    READY
12BD CA D0 12             JZ     STARPLAY
12C0               *
12C0 21 B3 22             LXI    H,V34OUT
12C3 16 A3                MVI    D,V34HEADER
12C5 1E C6                MVI    E,V34RES*256/256-1
12C7 7E                   MOV    A,M
12C8 FE AA                CPI    READY
12CA CA D0 12             JZ     STARPLAY
12CD 3E 00                MVI    A,FILL
12CF C9                   RET
12D0               *
12D0 36 FF         STARPLAY MVI  M,DONE  CLEAR THE READY FLAG
12D2 7B                   MOV    A,E    SET THE POINTER TO OUTPUT BUFFER
12D3 32 B5 22             STA    PTR
12D6 3E 10                MVI    A,V34RES-V12RES  SET COUNT=LENGTH OF RESULTS
12D8 32 B4 22             STA    COUNT
12DB 7A                   MOV    A,D    RETURN HEADER CODE TO BEGIN
12DC C9                   RET
12DD               *
12DD               *  EXECUTIVE FIT CALCULATOR (FOREGROUND)
12DD               *
12DD 3A B1 22      FITEXEC LDA   V12OUT  IF OUTPUT BUFFER IS READY
12E0 B7                   ORA    A
12E1 F2 FE 12             JP     FIT12
12E4               *
12E4 3A B3 22             LDA    V34OUT  CHECK V34'S BUFFER TOO
12E7 B7                   ORA    A
12E8 F8                   RM     .
12E9               *
```

```
12E9 21 30 22       FIT34  LXI   H,V34BUF
12EC CD 13 13              CALL  STFIT
12EF 01 C7 22              LXI   B,V34RES
12F2 11 FB 22              LXI   D,V34PRM
12F5 CD C4 08              CALL  SPIN
12F8 3E AA                 MVI   A,READY
12FA 32 B3 22              STA   V34OUT
12FD C9                    RET
12FE                *
12FE 21 B0 21       FIT12  LXI   H,V12BUF
1301 CD 13 13              CALL  STFIT  [HL]->SAMPLE AREA
1304 01 B7 22              LXI   B,V12RES  [BC]->RESULTS AREA
1307 11 D7 22              LXI   D,V12PRM  [DE]->PARAMS AREA
130A CD C4 08              CALL  SPIN
130D 3E AA                 MVI   A,READY
130F 32 B1 22              STA   V12OUT
1312 C9                    RET
1313                *
1313 DF             STFIT  RST   REF/8  [HL]->SAMPLES
1314 3A B6 22              LDA   FITMODE  IF TRANSMIT ENABLED, DO IT
1317 E6 08                 ANI   FITXMIT
1319 C8                    RZ
131A 11 20 00              LXI   D,NSAMPS  [DE]=#SAMPLES
131D C3 CF 13              JMP   ELEXMIT  TRANSMIT THEM
1320                *
1320 3B             IPARAM DB    3BH    LO GAIN = 1/50.9 HIGAIN
1321 A0                    DB    0A0H
1322 F2                    DB    0F2H
1323 41                    DB    041H   ALPHA = 1.40
1324 E6                    DB    0E6H
1325 66                    DB    066H
1326 3F                    DB    03FH   BETA = 0.4
1327 CC                    DB    0CCH
1328 CD                    DB    0CDH
1329                IPRMX  EQU   $
1329 00          V         DB    256    END OF FIT
132A                *
132A                * ENTER COMMAND VECTOR INTO TABLE
132A                *
132A                       ORG   FMODE/4+CMDTAB
0054 13 12                 DW    FITCMD
0056                *
0056                * RAM
0056                *
0056                       ORG   FITRAM
21B0                SMPBUF EQU   $      SAMPLES BUFFER
21B0                V12BUF DS    NSAMPS*2*2
2230                V34BUF DS    NSAMPS*2*2
22B0                *
22B0                FITVARS EQU  $
```

```
22B0          0250 V12IN  DS     1
22B1          0251 V12OUT DS     1
22B2          0252 V34IN  DS     1
22B3          0253 V34OUT DS     1
22B4          0254 COUNT  DS     1      COUNT OF BYTES TO SEND OUT
22B5          0255 PTR    DS     1      POINTER TO RESULTS
22B6          0256 FITMODE DS    1      ENABLE/DISABLE BITS
22B7          0257 FVEND  EQU    $
22B7          0258 *
22B7          0259 V12RES DS     5*FLT+1  RESULT AREAS
22C7          0260 V34RES DS     5*FLT+1
22D7          0261 *
22D7          0262 TBLNG  EQU    AVPTS*2*FLT  AHI/ALO TABLE LENGTH
22D7          0263 V12PRM DS     3*FLT  PARAMETER AREA
22E0          0264 V12TBL DS     TBLNG  AHI/ALO TABLE AREA
22F8          0265 V34PRM DS     3*FLT
2301          0266 V34TBL DS     TBLNG
2319          0267 *
2319          0268 * EXTERNALS
2319          0269 *
2319          0281 ELEXMIT EQU   ELE+15
```

```
0000            *
0000            * CRRES FLIGHT SOFTWARE --- SAWTOOTH GENERATOR
0000            * WRITTEN BY PETER R HARVEY
0000            *
0000            * FILE SAW.A
0000            *
0000            PSW     EQU   6
0000            SAWCODE EQU   058H
0000            *
0000                    ORG   SAW
1330 C3 3C 13           JMP   SAWINIT
1333 C3 5F 13           JMP   SAWSTEP
1336            *
1336 21 A4 21   SAWDSC  LXI   H,SAWOFF
1339 DF                 RST   REF/8
133A 7E                 MOV   A,M
133B C9                 RET
133C            *
133C 11 B5 13   SAWINIT LXI   D,DEFALT  SET DEFAULTS
133F 21 A4 21           LXI   H,SAWOFF
1342 0E 06              MVI   C,6
1344 D7                 RST   COPY/8
1345 C9                 RET
1346            *
1346            * PERFORM SAWTOOTH COMMANDS
1346            *
1346 7C         SAWCMD  MOV   A,H    SELECT WHICH REG
1347 E6 07              ANI   7
1349 5D                 MOV   E,L
134A 21 A4 21           LXI   H,SAWOFF  REFERENCE OPTIONS
134D DF                 RST   REF/8
134E 73                 MOV   M,E    STORE
134F C9                 RET
1350            *
1350            * SAWTOOTH SYNCHRONIZER
1350            *
1350 21 A0 21   SAWSYNC LXI   H,BIASREG  LATCH IN
1353 11 A4 21           LXI   D,SAWOFF  THE COMMANDED VALUES
1356 0E 04              MVI   C,4
1358 D7                 RST   COPY/8
1359 3E 01              MVI   A,1    AT NEXT STEP, RESET BIAS
135B 32 A3 21           STA   DIVCNT
135E C9                 RET
135F            *
135F            * SAWTOOTH STEP
135F            *
135F 3A A5 21   SAWSTEP LDA   SAWDEL  IF NO DELTA, THEN
1362 B7                 ORA   A      THIS PACKAGE IS DISABLED
1363 C8                 RZ    .
```

```
1364                0050 *
1364 3A 1D 20       0051        LDA   FRAME  ON FRAME 30, LAST LINE
1367 FE FE          0052        CPI   -2     PERFORM A SYNC
1369 C2 74 13       0053        JNZ   SSOPT
136C 3A 1C 20       0054        LDA   WORD
136F FE E0          0055        CPI   224
1371 D4 50 13       0056        CNC   SAWSYNC
1374                0057 *
1374 3A A8 21       0058 SSOPT  LDA   OPTIONS  IF STEPPING DISABLED, QUIT
1377 0F             0059        RRC
1378 D0             0060        RNC
1379                0061 *
1379 21 A3 21       0062 STEP   LXI   H,DIVCNT  DIVIDE STEPS
137C 35             0063        DCR   M
137D C0             0064        RNZ
137E 3A A7 21       0065        LDA   SAWDIV  THEN RELOAD
1381 77             0066        MOV   M,A
1382                0067 *
1382 3A A8 21       0068        LDA   OPTIONS  IF BIASING ALLOWED
1385 E6 02          0069        ANI   2      DO IT
1387 C4 A5 13       0070        CNZ   BIASEM
138A                0071 *
138A 21 A2 21       0072        LXI   H,PERCNT
138D 35             0073        DCR   M
138E CA 9A 13       0074        JZ    FLIP
1391                0075 *
1391 3A A1 21       0076        LDA   DELREG  BIASREG += DELREG
1394 21 A0 21       0077        LXI   H,BIASREG
1397 86             0078        ADD   M
1398 77             0079        MOV   M,A
1399 C9             0080        RET
139A                0081 *
139A 3A A6 21       0082 FLIP   LDA   SAWPER  RESET PERIOD REGISTER
139D 77             0083        MOV   M,A
139E 21 A1 21       0084        LXI   H,DELREG  DELREG = -DELREG
13A1 97             0085        SUB   A
13A2 96             0086        SUB   M
13A3 77             0087        MOV   M,A
13A4 C9             0088        RET
13A5                0089 *
13A5 2A A0 21       0090 BIASEM LHLD  BIASREG
13A8 3A A9 21       0091        LDA   SENSOR
13AB 67             0092        MOV   H,A
13AC E5             0093        PUSH  H
13AD CD 63 01       0094        CALL  SETBIAS
13B0 E1             0095        POP   H
13B1 24             0096        INR   H
13B2 C3 63 01       0097        JMP   SETBIAS
13B5                0098 *
13B5                0099 * DEFAULT SETTINGS FOR SAWTOOTH
```

```
13B5             *
13B5 00            DEFALT DB    0      OFFSET
13B6 02                   DB    2      DELTA (PKG ENABLE TOO)
13B7 40                   DB    64 # STEPS UP THEN DOWN
13B8 02                   DB    2      64/N HZ STEPS
13B9 00                   DB    0      STEPPING/BIASING DISABLED
13BA 01                   DB    1      SENSORS 1 AND 2 GET SAWTOOTH
13BB 00         V         DB    256    SAWTOOTH END
13BC             *
13BC             * ENTER COMMAND VECTOR INTO TABLE
13BC             *
13BC                      ORG   SAWCODE/4+CMDTAB
0056 46 13                DW    SAWCMD
0058             *
0058             * SAWTOOTH VARIABLES
0058             *
0058                      ORG   SAWRAM
21A0               BIASREG DS   1      CURRENT BIAS VALUE
21A1               DELREG DS    1      CURRENT DELTA
21A2               PERCNT DS    1      PERIOD COUNTER
21A3               DIVCNT DS    1      DIVIDER COUNTER
21A4             *
21A4               SAWOFF DS    1      SAWTOOTH OFFSET
21A5               SAWDEL DS    1      DELTA (IF ZERO, PKG DISABLE)
21A6               SAWPER DS    1      #STEPS UP
21A7               SAWDIV DS    1      DIVIDER OF CALLS
21A8               OPTION DS    1      ......BS (B=BIAS ENABLE, S=STEP ENABLE)
21A9               SENSOR DS    1      WHICH BIAS PAIR (1 OR 3)
21AA               SPARE  DS    2
21AC             *
21AC                      ORG   BKGRAM  ACCESS SYSTEM CLOCK
201C               WORD   DS    1
201D               FRAME  DS    1
```

```
0000                *
0000                * CRRES FLIGHT PROGRAM---ELECTRIC FIELD MANAGEMENT
0000                * WRITTEN BY PETER R HARVEY
0000                *
0000                * FILE : ELE.A
0000                *
0000                PSW      EQU   6       8085 SPECIFIC INFORMATION
0000                SP       EQU   6
0000                *
0000                * FAST DIGITAL MONITOR DEFINITION
0000                *
0000                MBITS    EQU   00FH    MAIN STATUS
0000                BBITS    EQU   0B0H    BURST STATUS
0000                TESTFLAG EQU   40H     TEST/CAL MODE BIT
0000                MODEFLAG EQU   1       VOLTAGE/CURRENT MODE BIT
0000                PLAYFLAG EQU   80H     PLAYBACK ENABLE BIT
0000                MBXMIT   EQU   8       MAIN TRANSMIT OVERRIDE
0000                *
0000                MUX      EQU   2800H   MUX SETTING CMD
0000                RESET    EQU   3000H   RESET RELAY CMD
0000                SET      EQU   RESET+100H
0000                *
0000                         ORG   ELE
13C0 C3 D6 13                JMP   ELEINIT  INITIALIZATION
13C3 C3 BB 14                JMP   ELEFRAME  MINOR FRAME SYNC
13C6 C3 6A 15                JMP   ELESAMP  SAMPLE TIME
13C9 C3 31 15                JMP   ELETELEM  TELEMETRY TIME
13CC C3 42 15                JMP   ELEDSC  DIGITAL SUBCOM TIME
13CF C3 2E 16                JMP   ELEXMIT  REQUEST MAIN PLAYBACK
13D2                *
13D2 3A 77 20       ELESTAT  LDA   FDM     RETURN FAST STATUS
13D5 C9                      RET
13D6                *
13D6                * INITIALIZE THE ELECTRIC FIELD PACKAGE
13D6                *
13D6 21 50 20       ELEINIT  LXI   H,ELERAM
13D9 0E 45                   MVI   C,ELEND-ELERAM
13DB CF                      RST   ZERO/8
13DC C9                      RET
13DD                *
13DD                * PERFORM ELECTRIC FIELD CATEGORY COMMANDS
13DD                *
13DD 7C             SMCMD    MOV   A,H     A=MUX NUMBER
13DE E6 07                   ANI   7       CHECK IF IT'S AN ELE
13E0 FE 07                   CPI   7       CONTROLLED REGISTER.
13E2 3F                      CMC   .       IF NOT 0-6, ERROR.
13E3 D8                      RC    .
13E4 C3 CC 02                JMP   SETMUX  L=VALUE ALREADY
13E7                *
```

```
13E7 7C           RLCMD  MOV    A,H    CRY=1 TD SET RELAY
13E8 0F                  RRC    .
13E9 7D                  MDV    A,L    A=RELAY NUMBER TD FLIP
13EA C3 B8 01            JMP    SETRELAY
13ED                     *
13ED 7C           SETRB  MDV    A,H    SAVE 3 BITS OF MS BYTE
13EE E6 07               ANI    7
13F0 F6 20               ORI    ELERAM/2048*2048/256  TAKE 5 MSB'S
13F2 67                  MDV    H,A    DF THE RAM ADDRESSING
13F3 22 50 20            SHLD   RAMBASE  SET THE RAM BASE ADDRESS
13F6 C9                  RET
13F7                     *
13F7 C6 3E        SETMD  ADI    TESTFLAG-2  MDVE TEST FLAG DVER
13F9 32 94 20            STA    FDMNEXT
13FC 0F                  RRC    .
13FD 11 0A 14            LXI    D,VMLIST
1400 D2 06 14            JNC    SM1
1403 11 1C 14            LXI    D,IMLIST
1406 3E 01        SM1    MVI    A,1    ASK FOR INTERNAL BATCH CMDS
1408 E7                  RST    4
1409 C9                  RET
140A                     *
140A 12 31        VMLIST DW     SET+18
140C 13 31               DW     SET+19
140E 07 30               DW     RESET+7
1410 08 30               DW     RESET+8
1412 09 30               DW     RESET+9
1414 10 30               DW     RESET+16
1416 01 28               DW     MUX+0+1
1418 01 29               DW     MUX+100H+1
141A FF FF               DW     -1
141C                     *
141C 07 31        IMLIST DW     SET+7
141E 08 31               DW     SET+8
1420 09 31               DW     SET+9
1422 10 31               DW     SET+16
1424 12 30               DW     RESET+18
1426 13 30               DW     RESET+19
1428 00 28               DW     MUX+0+0
142A 00 29               DW     MUX+100H+0
142C FF FF               DW     -1
142E                     *
142E 7C           SETFOR MDV    A,H    IF QTY TYPE CMD, GD
142F E6 04               ANI    4
1431 C2 70 14            JNZ    SETFQ
1434 3E 01               MVI    A,1    IF VDLTAGE ENABLED,
1436 11 52 20            LXI    D,VHXPTR  EFFECT VDLT POINTERS
1439 CD 41 14            CALL   SETFM
143C 3E 02               MVI    A,2    IF CURRENT ENABLED,
143E 11 54 20            LXI    D,IHXPTR  EFFECT CURR POINTERS
```

```
1441                   *
1441 A4          SETFM  ANA    H       IF NOT ENABLED, RETURN
1442 C8                 RZ     .
1443 7D                 MOV    A,L     GET LEFT NIBBLE OF DATA
1444 0F                 RRC    .       FOR THE 1ST POINTER
1445 0F                 RRC
1446 0F                 RRC
1447 0F                 RRC
1448 CD 50 14           CALL   SF2
144B 7B                 MOV    A,E     MOVE TO LX POINTERS
144C C6 04              ADI    VLXPTR-VHXPTR
144E 5F                 MOV    E,A
144F 7D                 MOV    A,L     GET RIGHT NIBBLE FOR LX'S
1450                   *
1450 E5          SF2    PUSH   H       SAVE INST COMMAND
1451 CD 5D 14           CALL   ADRFMT  HL= &FORMAT(A)
1454 EB                 XCHG   .
1455 73                 MOV    M,E     STORE THE ADDRESS INTO HX/LX PTR
1456 23                 INX    H
1457 72                 MOV    M,D
1458 2B                 DCX    H
1459 EB                 XCHG
145A E1                 POP    H
145B B7                 ORA    A
145C C9                 RET
145D                   *
145D                   * ADDRESS THE FORMAT(A)
145D                   *
145D 21 3D 16    ADRFMT LXI    H,ROMFOR  POINT AT ROM FORMATS
1460 E6 0F              ANI    15
1462 FE 0A              CPI    10      IF FORMAT#10 THRU #15 THEN
1464 DA 6A 14           JC     ADRF1
1467 21 F5 1F           LXI    H,RAMFOR-160  USE RAM FORMATS
146A 87          ADRF1  ADD    A       EACH FORMAT IS 16 BYTES
146B 87                 ADD    A
146C 87                 ADD    A
146D 87                 ADD    A
146E DF                 RST    REF/8   HL=HL+A
146F C9                 RET
1470                   *
1470 7C          SETFQ  MOV    A,H     IF EVEN, THEN QTY INDEX=L
1471 0F                 RRC    .
1472 7D                 MOV    A,L
1473 D2 83 14           JNC    SETQX
1476                   *
1476 5D          SETQV  MOV    E,L
1477 3A 5B 20           LDA    TMINDEX  RAMFOR[TMINDEX]=L
147A 21 95 20           LXI    H,RAMFOR
147D DF                 RST    REF/8
147E 73                 MOV    M,E
```

```
147F 3A 5B 20        0150          LDA    TMINDEX  TMINDEX++
14B2 3C              015I          INR    A
14B3 FE 60           0152 SETDX    CPI    96
14B5 D0              0153          RNC
14B6 32 5B 20        0154          STA    TMINDEX
14B9 B7              0155          ORA    A      RETURN(NC)
14BA C9              0156          RET
14BB                 0157 *
14BB                 015B * MINOR FRAME SYNC.
14BB                 0159 *  ON ENTRY: A=MINOR FRAME NUMBER
14BB                 0160 *
14BB 21 7E 20        0161 ELEFRAME LXI    H,BUFF1  PICK UP ADDRESSES
148E 11 6A 20        0162          LXI    D,BUFFO  OF BOTH BUFFERS
1491 0F              0163          RRC    .      IF ODD FRAME, GO
1492 DA A5 14        0164          JC     ODDFRAME
1495                 0165 *
1495                 0166 * EVEN FRAME
1495                 0167 *
1495 EB              016B          XCHG   .      SET THE BUFFER POINTERS
1496 CD E0 14        0169          CALL   SETBUF
1499 CD EE 11        0170          CALL   FITTEL  GET THE SPIN FIT
149C 32 BB 20        0171          STA    SFR    ASYNCHRONOUS DATA
149F 3E FF           0172          MVI    A,-1   RESET THE SAMPLE COUNTER
14A1 32 5A 20        0173          STA    SMPCNT
14A4 C9              0174          RET    .
14A5                 0175 *
14A5                 0176 * ODD FRAME
14A5                 0177 *
14A5 CD E0 14        017B ODDFRAME CALL SETBUF  SET THE BUFFERS
14AB 3A 1D 20        0179          LDA    FRAME
14AB 3C              01B0          INR    A      IF FRAME=7 THEN
14AC E6 07           0181          ANI    7      RESET THE LX POINTER
14AE CC 09 15        01B2          CZ     RSTLX
14B1 CD 13 15        01B3          CALL   RSTHX  RESET HX EVERY ODD FRAME
14B4                 01B4 *
14B4                 0185 * CALCULATE THE FAST DIGITAL MONITOR
14B4                 01B6 *
14B4 97              01B7          SUB    A      GET BURST FDM BITS FROM
14B5 CD 11 10        01BB          CALL   BURDSC  THE BURST CONTROLLER
14BB E6 B0           01B9          ANI    BBITS
14BA 5F              0190          MOV    E,A
14BB                 0191 *
14BB CD EA 17        0192          CALL   SWPSTAT  PUT SWEEP STATUS IN L
14BE E6 01           0193          ANI    I
14C0 6F              0194          MOV    L,A
14C1                 0195 *
14C1 3E 02           0196          MVI    A,2    GET THE COMMAND COUNT BIT
14C3 E5              0197          PUSH   H
14C4 E7              019B          RST    4
14C5 E1              0199          POP    H
```

```
14C6 29                   DAD   H      SHIFT IT IN
14C7 DF                   RST   REF/8
14C8                ;
14C8 3A 94 20             LDA   FDMNEXT  GET THE I/V MODE COMMANDED
14CB E6 41                ANI   MODEFLAG+TESTFLAG
14CD 29                   DAD   H
14CE DF                   RST   REF/8
14CF B3                   ORA   E      PUT THESE WITH BURST BITS
14D0 5F                   MOV   E,A
14D1                ;
14D1 2A 5E 20             LHLD  XMTCNT  IF THE TRANSMIT COUNT!=0
14D4 7C                   MOV   A,H
14D5 B5                   ORA   L
14D6 CA DB 14             JZ    ORFDM
14D9 3E 88                MVI   A,PLAYFLAG+M8XMIT  TURN ON XMIT
14DB B3             ORFDM ORA   E
14DC 32 77 20             STA   FDM
14DF C9                   RET
14E0                ;
14E0                ; SET THE BUFFER POINTERS. (SAMPLE & TELEM)
14E0                ;  ON ENTRY: [HL]=BUFFER FOR TELEM
14E0                ;            [DE]=BUFFER FOR SAMPLING
14E0                ;
14E0 22 68 20       SETBUF SHLD TMPTR
14E3 EB                   XCHG  .      AND RESETTING THE
14E4 22 64 20             SHLD  HBPTR  HIGH AND LOW SAMPLE
14E7 11 0E 00             LXI   D,LSAMP  POINTERS TO THE
14EA 19                   DAD   D      OTHER BUFFER.
14EB 22 66 20             SHLD  LBPTR
14EE                ;
14EE                ; PLACE THE GAIN BITS INTO THE TM BUFFER
14EE                ;
14EE 2A 68 20             LHLD  TMPTR  PLACE THE GAINS IN
14F1 11 0C 00             LXI   D,12   INTO THE BUFFER AT
14F4 19                   DAD   D      AN OFFSET OF 12
14F5 3A 92 20             LDA   HGAINS
14F8 77                   MOV   M,A
14F9                ;
14F9 11 07 00             LXI   D,7    THEN PUT IN THE LOW
14FC 19                   DAD   D      GAINS
14FD 7E                   MOV   A,M
14FE E6 F0                ANI   0F0H
1500 5F                   MOV   E,A
1501 3A 93 20             LDA   LGAINS
1504 E6 0F                ANI   0FH
1506 B3                   ORA   E
1507 77                   MOV   M,A
1508 C9                   RET   .
1509                ;
1509                ; RESET THE HIGH AND LOW SAMPLE LIST POINTERS
```

```
1509                * DEPENDING UPON THE CURRENT MODE (E-FIELD OR LANGMUIR)
1509                *
1509 21 56 20       RSTLX  LXI    H,VLXPTR  POINT AT THE PAIR
150C CD 1D 15              CALL   INDEX  OF LISTS AND CHOOSE
150F 22 62 20              SHLD   LQPTR  ACC'ING TO MODE
1512 C9                    RET    .
1513                *
1513 21 52 20       RSTHX  LXI    H,VHXPTR  SAME FOR THE HIGH
1516 CD 1D 15              CALL   INDEX
1519 22 60 20              SHLD   HQPTR
151C C9                    RET    .
151D                *
151D 3A 77 20       INDEX  LDA    FDM    ADD 0 OR 2
1520 E6 01                 ANI    MODEFLAG  TO THE POINTER
1522 1E 02          IND1   MVI    E,2
1524 C2 28 15              JNZ    IND2
1527 5F                    MOV    E,A    DEPENDING UPON THE
1528 16 00          IND2   MVI    D,0    MODE SETTING.
152A 19                    DAD    D
152B                *
152B 7E                    MOV    A,M    PICK UP THE POINTER
152C 23                    INX    H
152D 66                    MOV    H,M
152E 6F                    MOV    L,A
152F 2B                    DCX    H      AND SUBTRACT THE 1ST
1530 C9                    RET    .
1531                *
1531                * E-FIELD TELEMETRY OUTPUT ROUTINES.
1531                *
1531 2A 68 20       ELETEL LHLD   TMPTR  [HL]->CURRENT DATA
1534 5E                    MOV    E,M    PICK UP 1 BYTE
1535 23                    INX    H
1536 87                    ORA    A      IF 1 BYTE ONLY
1537 CA 3D 15              JZ     FINTEL
153A 53                    MOV    D,E    ELSE GRAB ANOTHER
153B 5E                    MOV    E,M
153C 23                    INX    H
153D 22 68 20       FINTEL SHLD   TMPTR
1540 EB                    XCHG   .      PUT RESULT IN [HL]
1541 C9                    RET    .
1542                *
1542                * OUTPUT ELECTRIC FIELD DIGITAL SUBCOM
1542                * ON ENTRY: A= INDEX INTO ELE DIG SUB COM
1542                *
1542 FE 10          ELEDSC CPI    16     FROM 0 TO 15 ARE THE
1544 21 52 20              LXI    H,VHXPTR  HIGH RATE QTY LIST
1547 DA 54 15              JC     EDINX
154A D6 10                 SUI    16
154C FE 20                 CPI    32     FROM 16 TO 47 ARE
154E 21 56 20              LXI    H,VLXPTR  THE LOW QTY LIST
```

```
1551 D2 62 15               JNC    EDVAR
1554 F5               EDINX PUSH   PSW    INDEX TO GET HL
1555 3A 1E 20               LDA    CYCLE  PUT OUT I'S IN ODD CYCLES
1558 E6 01                  ANI    1
155A CD 22 15               CALL   IND1
155D F1                     POP    PSW    ADDRESS OF QTY LIST
155E 23                     INX    H
155F C3 67 15               JMP    ADDA
1562 D6 20            EDVAR SUI    32
1564 21 50 20               LXI    H,RAMBASE  FROM 48 ARE JUST VARS
1567 DF               ADDA  RST    REF/8
1568 7E                     MOV    A,M
1569 C9                     RET
156A                  *
156A                  * E-FIELD SAMPLING ROUTINE.
156A                  *
156A 21 5A 20         ELESAMP LXI  H,SMPCNT  SAMPLE COUNT++
156D 34                     INR    M
156E 7E                     MOV    A,M
156F 0F                     RRC    .      IF EVEN, SAMPLE HX
1570 DA 92 15               JC     LXBUR  ELSE SAMPLE LX
1573                  *
1573 2A 60 20         HXSAMP LHLD  HQPTR  INCREMENT LIST POINTER
1576 23                     INX    H
1577 22 60 20               SHLD   HQPTR
157A CD DA 15               CALL   GETQTY  [DE]=QUANTITY(MEM[HL])
157D                  *
157D 21 92 20               LXI    H,HGAINS  STORE THE GAIN
1580 CD B9 15               CALL   STOGAIN
1583                  *
1583 2A 64 20               LHLD   HBPTR  STORE [DE] IN BUFFER
1586 3A 5A 20               LDA    SMPCNT  LEFT OR RIGHT ADJUST
1589 E6 02                  ANI    2      DEPENDING UPON COUNT
158B CD C4 15               CALL   STOQTY
158E 22 64 20               SHLD   HBPTR
1591 C9                     RET    .
1592                  *
1592 3A 5A 20         LXBUR LDA    SMPCNT
1595 E6 02                  ANI    2
1597 CA 0B 10               JZ     BURSAMP
159A                  *
159A 2A 62 20         LXSAMP LHLD  LQPTR  INCREMENT LOW LIST
159D 23                     INX    H
159E 22 62 20               SHLD   LQPTR
15A1 CD DA 15               CALL   GETQTY  SAMPLE THE QUANTITY
15A4 21 93 20               LXI    H,LGAINS  STORE THE GAIN
15A7 CD B9 15               CALL   STOGAIN
15AA 2A 66 20               LHLD   LBPTR  STORE IN LOW BUFFER
15AD 3A 5A 20               LDA    SMPCNT  LEFT OR RIGHT ADJUSTED
15B0 E6 04                  ANI    4
```

```
15B2 CD C4 15          CALL   STOQTY
15B5 22 66 20          SHLD   LBPTR
15B8 C9                RET    .
15B9              !
15B9              ! STORE GAIN BIT FROM [DE] INTO [M]
15B9              !
15B9 7A           STOGAIN MOV A,D     IF GAIN=0, STORE 0
15BA E6 10             ANI    10H     BY RESETTING CARRY
15BC CA C0 15          JZ     STG0
15BF 37                STC    .       ELSE SET CARRY
15C0 7E           STG0 MOV    A,M     MOVE CRY INTO LSB
15C1 17                RAL    .
15C2 77                MOV    M,A
15C3 C9                RET    .
15C4              !
15C4              ! STORE A QUANTITY INTO THE BUFFER AT [HL]
15C4              !
15C4 C2 D1 15     STOQTY JNZ  ODD
15C7 EB           EVEN XCHG   .       STORE THE EVEN
15C8 29                DAD    H       BY SHIFTING THE
15C9 29                DAD    H       12 BITS TO THE LEFT
15CA 29                DAD    H
15CB 29                DAD    H
15CC EB                XCHG   .
15CD 72                MOV    M,D     AND STORING
15CE 23                INX    H       THE 12 BITS WITH
15CF 73                MOV    M,E     ZERO FOLLOWING
15D0 C9                RET    .       LEAVE HL->BYTE
15D1              !
15D1 7A           ODD  MOV    A,D     STORE THE ODD
15D2 E6 0F             ANI    0FH     BY PUTTING THE
15D4 B6                ORA    M       MSB'S DOWN FIRST
15D5 77                MOV    M,A
15D6 23                INX    H       AND THEN THE
15D7 73                MOV    M,E     LSB'S
15D8 23                INX    H
15D9 C9                RET    .
15DA              !
15DA              ! GET A QUANTITY.
15DA              !  ON ENTRY: [M] IS THE QUANTITY DESCRIPTOR.
15DA              !  ON EXIT : [DE]= THE 13 BIT VALUE OF THAT QTY
15DA              !
15DA 3A 77 20     GETQTY LDA  FDM     IF PLAYBACK MODE ENABLED
15DD A6                ANA    M       AND THE QTY IS ENABLED,
15DE FA 0A 16          JM     GETPLAY  THEN GET PLAYBACK
15E1              !
15E1 7E                MOV    A,M     IF A RAM QUANTITY,
15E2 E6 40             ANI    40H     THEN GET IT
15E4 C2 FF 15          JNZ    GETRAM
15E7              !
```

```
15E7                0400 ; GET AN ANALOG QUANTITY.
15E7                0401 ;
15E7 7E             0402 GETANA MOV    A,M     DETERMINE GAIN IN D
15E8 E6 3F          0403        ANI    3FH
15EA CD 2A 01       0404        CALL   AGC
15ED 1E 10          0405        MVI    E,10H   SET E TO RECORD HIGAIN/LOGAIN
15EF C2 F4 15       0406        JNZ    GA1
15F2 1E 00          0407        MVI    E,0
15F4                0408 ;
15F4 7A             0409 GA1    MOV    A,D     SAMPLE QTY(D)
15F5 CD E6 00       0410        CALL   SAMPLE  [HL]=ATOD(A)
15F8 7B             0411        MOV    A,E     PUT IN GAINBIT
15F9 E6 10          0412        ANI    10H
15FB B4             0413        ORA    H
15FC 57             0414        MOV    D,A     AND RETURN(DE)
15FD 5D             0415        MOV    E,L
15FE C9             0416        RET    .
15FF                0417 ;
15FF 7E             0418 GETRAM MOV    A,M     ADD THE OFFSET
1600 E6 3F          0419        ANI    03FH    FROM THE DESCRIPTOR
1602 2A 50 20       0420        LHLD   RAMBASE
1605 DF             0421        RST    REF/8
1606 5E             0422        MOV    E,M     LOAD TWO BYTES
1607 23             0423        INX    H
1608 56             0424        MOV    D,M
1609 C9             0425        RET    .
160A                0426 ;
160A                0427 ; GET BURST PLAYBACK DATA
160A                0428 ;
160A 3A 77 20       0429 GETPLAY LDA   FDM     IF MAIN XMIT, DO IT
160D E6 08          0430        ANI    MBXMIT
160F C2 17 16       0431        JNZ    GPMAIN
1612 CD 0E 10       0432        CALL   BURPLAY  [HL]=BURST PLAYBACK INFO
1615 EB             0433        XCHG   .       PUT IN DE
1616 C9             0434        RET    .
1617                0435 ;
1617 2A 5E 20       0436 GPMAIN LHLD   XMTCNT  DECREASE COUNT
161A 7C             0437        MOV    A,H     IF ZERO, RETURN(0)
161B B5             0438        ORA    L
161C EB             0439        XCHG   .
161D C8             0440        RZ     .
161E 1B             0441        DCX    D
161F EB             0442        XCHG
1620 22 5E 20       0443        SHLD   XMTCNT
1623 2A 5C 20       0444        LHLD   XMTPTR  THEN PICK UP 13 BITS
1626 5E             0445        MOV    E,M
1627 23             0446        INX    H
1628 56             0447        MOV    D,M
1629 23             0448        INX    H       AND UPDATE POINTER
162A 22 5C 20       0449        SHLD   XMTPTR
```

```
162D C9                    RET
162E                *
162E                * REQUEST TRANSMISSION.
162E                * ON ENTRY: [HL]->RAM AREA, [E]=COUNT OF SAMPLES
162E                *
162E 3A 77 20       ELEXMIT LDA   FDM     IF FDM XMIT GOING, RETURN
1631 E6 08                 ANI    MBXMIT
1633 37                    STC
1634 C0                    RNZ
1635 22 5C 20              SHLD   XMTPTR  SET POINTER
1638 EB                    XCHG   .
1639 22 5E 20              SHLD   XMTCNT  AND COUNTER
163C C9                    RET    .
163D                *
163D                * TELEMETRY TABLE DEFAULTS
163D                *
163D                BZ     EQU    0       MULTIPLEXOR ADDRESSES
163D                BY     EQU    1
163D                BX     EQU    2
163D                V3     EQU    3
163D                V2     EQU    4
163D                V1SC   EQU    5
163D                V1     EQU    6
163D                AGCF   EQU    7
163D                V12F   EQU    8
163D                F3     EQU    9
163D                F2     EQU    10
163D                F1     EQU    11
163D                V4     EQU    12
163D                AGCU   EQU    13
163D                V12U   EQU    14
163D                V34    EQU    15
163D                *
163D                PE     EQU    PLAYFLAG
163D                ROMFOR EQU    $       ROMFORMATS
163D                HXTABLE EQU   $
163D 08                    DB     V12F
163E 0F                    DB     V34
163F 88                    DB     V12F+PE
1640 8F                    DB     V34+PE
1641 08                    DB     V12F
1642 0F                    DB     V34
1643 88                    DB     V12F+PE
1644 8F                    DB     V34+PE
1645 08                    DB     V12F
1646 0F                    DB     V34
1647 88                    DB     V12F+PE
1648 8F                    DB     V34+PE
1649 08                    DB     V12F
164A 0F                    DB     V34
```

```
164B 88                  DB    V12F+PE
164C 8F                  DB    V34+PE
164D                *
164D                LXTABLE EQU   $
164D 86                  DB    V1+PE
164E 83                  DB    V3+PE
164F 87                  DB    ABCF+PE
1650 0B                  DB    F1
1651 84                  DB    V2+PE
1652 8C                  DB    V4+PE
1653 87                  DB    AGCF+PE
1654 09                  DB    F3
1655 86                  DB    V1+PE
1656 83                  DB    V3+PE
1657 87                  DB    AGCF+PE
1658 0A                  DB    F2
1659 84                  DB    V2+PE
165A 8C                  DB    V4+PE
165B 87                  DB    AGCF+PE
165C 09                  DB    F3
165D 86                  DB    V1+PE
165E 83                  DB    V3+PE
165F 87                  DB    AGCF+PE
1660 08                  DB    F1
1661 84                  DB    V2+PE
1662 8C                  DB    V4+PE
1663 87                  DB    AGCF+PE
1664 09                  DB    F3
1665 86                  DB    V1+PE
1666 83                  DB    V3+PE
1667 87                  DB    AGCF+PE
1668 0A                  DB    F2
1669 84                  DB    V2+PE
166A 8C                  DB    V4+PE
166B 87                  DB    AGCF+PE
166C 09                  DB    F3
166D 00        V         DB    256    ELE END
166E                *
166E                * ENTER COMMAND VECTORS INTO TABLE
166E                *
166E                     ORG   CMDTAB
0040 63 01               DW    SETBIAS  #0
0042 80 01               DW    SETSTUB  #8
0044 76 01               DW    SETGUARD  #10
0046 98 02               DW    SETVTRIM  #18
0048 AE 02               DW    SETFILTER  #20
004A DD 13               DW    SMCMD  #28
004C E7 13               DW    RLCMD  #30
004E                *
004E                     ORG   40H/4+CMDTAB
```

```
0050 2E 14         DW     SETFOR  *40
0052 ED 13         DW     SETRB  *48
0054          *
0054               ORG    68H/4+CMDTAB
005A F7 13         DW     SETMD  *68
005C          *
005C          * RAM SECTION
005C          *
005C               ORG    ELERAM
2050          RAMBASE DS  2       RAM VARIABLES BASE POINTER
2052          *
2052          VHXPTR DS   2       SAMPLE LIST POINTERS
2054          IHXPTR DS   2       FOR HIGH AND LOW
2056          VLXPTR DS   2
2058          ILXPTR DS   2
205A          SMPCNT DS   1       SAMPLE COUNT
205B          *
205B          TMINDEX DS  1       INDEX INTO RAM FORMAT
205C          XMTPTR DS   2       TRANSMIT POINTER
205E          XMTCNT DS   2       TRANSMIT COUNTER
2060          *
2060          HQPTR  DS   2       SAMPLE LIST POINTERS
2062          LQPTR  DS   2       FOR THE PRESENT MODE.
2064          HBPTR  DS   2       BUFFER POINTERS FOR
2066          LBPTR  DS   2       THE PRESENT MODE
2068          TMPTR  DS   2       TELEMETRY BUFFER POINTER.
206A          *
206A          BUFF0  DS   12+1+1+6  TELEMETRY BUFFER 0
207E          FDM    EQU  BUFF0+13  FAST DIGITAL MONITOR
207E          BUFF1  DS   12+1+1+6  TELEMETRY BUFFER 1
2092          SFR    EQU  BUFF1+13  SPIN-FIT RESULTS
2092          LSAMP  EQU  12+1+1  OFFSET BETWEEN HX AND LX
2092          *
2092          HGAINS DS   1       HIGH GAIN BITS
2093          LGAINS DS   1       LOW GAIN BITS
2094          FDMNEXT DS  1       FAST DIGITAL MONITOR (NEXT)
2095          RAMFOR DS   48*2
20F5          ELEND  EQU  $
20F5          *
20F5          * EXTERNALS
20F5          *
20F5               ORG    BUR
1008          BURINIT DS  3
100B          BURSAMP DS  3
100E          BURPLAY DS  3
1011          BURDSC DS   3
1014          *
1014               ORG    FIT
11E8          FITINI DS   3
11EB          FITSMP DS   3
```

```
11EE        0600 FITTEL DS     3
11F1        0601 *
11F1        0602        ORG    SWP
17E4        0603 SWPINIT DS    3
17E7        0604 SWPANG DS     3
17EA        0605 SWPSTAT DS    3
17ED        0606 *
17ED        0607        ORG    BKGRAM
201C        0608 WORD   DS     1
201D        0609 FRAME  DS     1
201E        0610 CYCLE  DS     1
```

```
0000                !
0000                ! CRRES FLIGHT SOFTWARE---MAIN PROGRAM LOADER
0000                ! FILE : LD.A
0000                !
0000                LDCODE EQU    0EBH   COMMAND NUMBER (5 BITS)
0000                !
0000                       ORG    LD
16B0                !
1680 21 30 29       LDINIT LXI    H,USER  POINT THE ADR REGISTER
1683 22 20 29              SHLD   ADR     TO THE USER LOADING AREA
1686 C9                    RET    .
1687                !
1687 7C             LDCMD  MOV    A,H     GET THE COMMAND AGAIN
1688 D6 EB                 SUI    LDCODE  REMOVE THE BIAS
168A CA 9B 16              JZ     SADRL   AND COUNT OFF EACH NUMBER
168D 3D                    DCR    A
168E CA A0 16              JZ     SADRH
1691 3D                    DCR    A
1692 CA A5 16              JZ     LOAD
1695 3D                    DCR    A
1696 CA AF 16              JZ     JUMP
1699 37                    STC    . IF UNKNOWN, RETURN(CRY)
169A C9                    RET
169B                !
169B 7D             SADRL  MOV    A,L     SET LOW ADDRESS
169C 32 20 29              STA    ADR
169F C9                    RET
16A0                !
16A0 7D             SADRH  MOV    A,L     SET HIGH ADDRESS
16A1 32 21 29              STA    ADR+1
16A4 C9                    RET    .
16A5                !
16A5 EB             LOAD   XCHG   .
16A6 2A 20 29              LHLD   ADR     MEM[ADR++] = VALUE
16A9 73                    MOV    M,E
16AA 23                    INX    H
16AB 22 20 29              SHLD   ADR
16AE C9                    RET    .
16AF                !
16AF 3A 30 29       JUMP   LDA    USER    EXECUTE USER PROGRAM
16B2 B5                    ADD    L       CHECK CODE PLUS COMMAND
16B3 FE AA                 CPI    0AAH    IS THE RIGHT VALUE
16B5 C0                    RNZ    .       IF NOT RIGHT, SIGNAL ERROR
16B6 97                    SUB    A       RESET THE CODE
16B7 32 30 29              STA    USER
16BA C3 31 29              JMP    USER+1
16BD 00          V         DB     256     END OF LD MODULE
16BE                !
16BE                ! ENTER COMMAND VECTOR INTO TABLE
```

```
16BE              0050 *
16BE              0051       ORG   LDCODE/4+CMDTAB
007A 87 16        0052       DW    LDCMD
007C              0053 *
007C              0054 * VARIABLES
007C              0055 *
007C              0056       ORG   LDRAM
2920              0057 ADR   DS    2      USER LOAD ADDRESS
2922              0058       DS    14
2930              0059 USER  DS    3F0H   USER PROGRAM LOADING AREA
```

```
0000                *
0000                * CRRES FLIGHT SOFTWARE --- DEPLOYMENT MODULE
0000                * WRITTEN BY PETER R HARVEY
0000                *
0000                * FILE: DEP.A
0000                *
0000                PSW     EQU    6
0000                *
0000                DEPLOY  EQU    0D0H    COMMAND NUMBER
0000                DEPOVER EQU    DEPLOY+4  DEPLOY SWITCH OVERRIDE
0000                *
0000                LCOVER  EQU    1       BOOM STATUS BITS
0000                TURNS   EQU    2
0000                RCOVER  EQU    4
0000                ENDWIRE EQU    8
0000                *
0000                * INTERNAL CODES FOR DEPLOY STATUS
0000                *
0000                CMDED   EQU    1       MOTOR IS COMMANDED TO RUN
0000                ACTUAL  EQU    2       MOTOR IS RUNNING
0000                *
0000                MOFF    EQU    0       NEITHER CMDED NOR RUNNING
0000                MPAUSE  EQU    CMDED   COMMANDED BUT PAUSED
0000                MSTOP   EQU    ACTUAL  RUNNING BUT SHOULD STOP
0000                MRUN    EQU    ACTUAL+CMDED  RUNNING AS COMMANDED
0000                *
0000                * CODE ENTRY POINTS
0000                *
0000                        ORG    DEP
16C4 C3 D0 16               JMP    DEPINIT
16C7 C3 0B 17               JMP    DEPSAMP
16CA                *
16CA                * DIGITAL SUB COMMUTATOR (STATUS)
16CA                *
16CA 21 44 20       DEPDSC  LXI    H,DEPSTAT  RETURN DEPSTAT ON
16CD DF                     RST    REF/8  FOR THE DIGITAL STATUS
16CE 7E                     MOV    A,M
16CF C9                     RET
16D0                *
16D0                * INITIALIZATION. TURN OFF BOTH BOOMS
16D0                *
16D0 CD E3 00       DEPINIT CALL   BOOMSTAT
16D3 32 45 20               STA    BOOMBITS
16D6 21 00 D0               LXI    H,DEPLOY*256+0
16D9 CD 02 17               CALL   DOVER  CLEAR OVERRIDE
16DC                *
16DC                * ACCEPT DEPLOY COMMANDS
16DC                * ON ENTRY: [HL]= COMMAND
16DC                * ON EXIT: CARRY SET IF NOT A DEPLOYMENT COMMAND
```

```
16DC          *
16DC 7C       DEPCMD MOV   A,H    CHECK THE INCOMING COMMAND
16DD FE D4           CPI   DEPOVER  OVERRIDE SWITCHES COMMAND?
16DF CA 02 17        JZ    DOVER
16E2 E6 FC           ANI   0FCH
16E4 D6 D0           SUI   DEPLOY  IF DEPLOY START/STOP, GO
16E6 37              STC   .      RETURN CARRY IF NOT US
16E7 C0              RNZ   .
16E8          *
16E8 7D              MOV   A,L    SET NEW LIMIT
16E9 32 48 20        STA   LIMIT
16EC          *
16EC 7C              MOV   A,H    LOOK UP WHAT TO DO WITH MOTORS
16ED E6 03           ANI   3
16EF 21 07 17        LXI   H,DEPTAB
16F2 DF              RST   REF/8
16F3 7E              MOV   A,M
16F4 32 44 20        STA   DEPSTAT
16F7          *
16F7 97              SUB   A
16F8 32 4A 20        STA   DEPCNT  DEPCNT=0
16FB 32 46 20        STA   DLEN1  ZERO DEPLOYED LENGTH COUNTS
16FE 32 47 20        STA   DLEN2
1701 C9              RET
1702          *
1702 7D       DOVER  MOV   A,L    SET OVERRIDE BITS
1703 32 49 20        STA   OVERRIDE
1706 C9              RET
1707          *
1707          * 2-BIT CODE TO STATE TABLE
1707          *
1707 22       DEPTAB DB    MSTOP*11H  TURN OFF BOTH MOTORS
1708 21              DB    MSTOP*10H+MPAUSE  TURN ON MOTOR 1
1709 12              DB    MPAUSE*10H+MSTOP  TURN ON MOTOR 2
170A 11              DB    MPAUSE*11H  TURN ON BOTH MOTORS
170B          *
170B          * DEPLOY SAMPLING.  MONITOR LENGTHS/CURRENTS OF MOTORS
170B          *                   TURN ON/OFF MOTORS AS COMMANDED.
170B          *
170B D5       DEPSAMP PUSH D      SAVE [DE] IN INTERRUPT
170C 21 4A 20        LXI   H,DEPCNT  UPDATE DEPLOY SAMPLE CNT
170F 34              INR   M      WHICH CHANGES THE MOTOR
1710 CD 68 17        CALL  SMPLENG  SAMPLE LENGTHS
1713 CD 18 17        CALL  DEPMON  CHECK BOOM DEPLOYMENT
1716 D1              POP   D
1717 C9              RET
1718          *
1718          * MONITOR BOOM DEPLOYMENT
1718          *
1718 CD 93 17 DEPMON CALL  GETSTAT  GET CURRENT MOTOR STATE
```

```
171B FE 00                    CPI    MOFF   IF OFF, QUIT
171D C8                       RZ     .
171E FE 01                    CPI    MPAUSE  IF MOTOR PAUSED, SEE IF
1720 CA 38 17                 JZ     TRYGO  IT CAN GO NOW
1723 FE 02                    CPI    MSTOP  IF MOTOR SHOULD STOP, DO IT
1725 CA 55 17                 JZ     STOP
1728                   *
1728                   * MOTOR IS COMMANDED AND RUNNING
1728                   *
1728 CD A8 17          KEEPGO CALL   LIMCHK  CHECK LIMITS
172B D2 55 17                 JNC    STOP   IF OVER LIMIT, STOP NOW
172E CD C8 17                 CALL   CMPLENG  IF LENGTH < OTHER, OK
1731 D8                       RC     .
1732 FE 03                    CPI    3      IF LENGTH-OTHER > 3, PAUSE
1734 D2 5A 17                 JNC    PAUSE
1737 C9                       RET
1738                   *
1738                   * TRY TO GO ONCE PAUSED
1738                   *
1738 CD A8 17          TRYGO  CALL   LIMCHK  IF BOOM AT LIMIT, DON'T
173B 3E 00                    MVI    A,MOFF  EVEN START IT.
173D D2 82 17                 JNC    SETSTAT
1740 CD C8 17                 CALL   CMPLENG  IF THIS BOOM IS SHORTER THAN
1743 DA 47 17                 JC     RUN    THE OTHER OK
1746 C0                       RNZ    .      IF EQUAL OK
1747                   *
1747                   * MOTOR CONTROL: A=NEW STATE FOR THE MOTOR
1747                   *
1747 3E 03             RUN    MVI    A,MRUN  SET THE STATE INFORMATION
1749 CD 82 17                 CALL   SETSTAT
174C 3A 4A 20                 LDA    DEPCNT  GET THE MTR BIT
174F E6 01                    ANI    1
1751 37                       STC    .      AND TURN ON THE MOTOR
1752 C3 1B 03                 JMP    SETMOTOR
1755                   *
1755 3E 00             STOP   MVI    A,MOFF
1757 C3 5C 17                 JMP    MTROFF
175A 3E 01             PAUSE  MVI    A,MPAUSE
175C CD 82 17          MTROFF CALL   SETSTAT
175F 3A 4A 20                 LDA    DEPCNT  GET MTR NUMBER
1762 E6 01                    ANI    1
1764 B7                       ORA    A      CLEAR CARRY FOR OFF
1765 C3 1B 03                 JMP    SETMOTOR
1768                   *
1768                   * SAMPLE THE LENGTHS OF THE BOOMS
1768                   *
1768 21 45 20          SMPLENG LXI   H,BOOMBITS  SAVE OLD BITS
176B 5E                       MOV    E,M
176C CD E3 00                 CALL   BOOMSTAT  GET NEW STATUS
176F 77                       MOV    M,A
```

```
1770                 *
1770 A8                      XRA    E      GET THE CHANGES
1771 A6                      ANA    M      WHICH ARE POSITIVE
1772 5F                      MOV    E,A
1773                 *
1773 3E 02                   MVI    A,TURNS  IF BDOM1 TURNS CTR
1775 21 46 20                LXI    H,DLEN1  IS A 1, INCREMENT DLEN1
1778 CD 7E 17                CALL   TRNCHK
177B                 *
177B 3E 20                   MVI    A,TURNS*16  CHECK BOOM2 AS WELL
177D 23                      INX    H
177E A3              TRNCHK  ANA    E      IF BIT IS 0, QUIT
177F C8                      RZ     .
1780 34                      INR    M      ELSE INCREMENT LENGTH
1781 C9                      RET
1782                 *
1782                 * SET/GET STATUS OF CURRENT MOTOR
1782                 *
1782 6F              SETSTAT MOV    L,A    SAVE NEW 2-BIT CODE
1783 3A 44 20                LDA    DEPSTAT  GET THE OTHER SIDE
1786 CD 9F 17                CALL   SWAP
1789 E6 30                   ANI    3*16   SAVE THEM
178B B5                      ORA    L      PUT IN NEW BITS
178C CD 9F 17                CALL   SWAP   REORIENT
178F 32 44 20                STA    DEPSTAT  AND SAVE
1792 C9                      RET
1793                 *
1793 3A 44 20        GETSTAT LDA    DEPSTAT  GET CURRENT STATUS
1796 CD 9F 17                CALL   SWAP
1799 E6 03                   ANI    3
179B C9                      RET
179C                 *
179C 3A 45 20        SWITCHES LDA   BOOMBITS   GET THE RIGHT SWITCHES
179F                 *
179F                 * SWAP REVERSES THE NIBBLES IN A BYTE IF MOTOR 2
179F                 *
179F 5F              SWAP    MOV    E,A    ADJUSTNPUT
17A0 3A 4A 20                LDA    DEPCNT  TEST WHICH MOTOR
17A3 0F                      RRC    .
17A4 7B                      MOV    A,E    RETURN SAME IF MOTOR 1
17A5 D0                      RNC    .
17A6 0F                      RRC    .      ELSE GET HIGH NIBBLE
17A7 0F                      RRC    .
17A8 0F                      RRC    .
17A9 0F                      RRC    .
17AA C9                      RET
17AB                 *
17AB                 * LIMIT CHECK THE CURRENT BOOM
17AB                 *
17AB CD 9C 17        LIMCHK  CALL   SWITCH  GET THE MICROS
```

```
17AE 21 49 20              LXI   H,OVERRIDE  ADD IN THE OVERRIDE BITS
17B1 B6                    ORA   M       TO DEFEAT AN ERRANT SWITCH
17B2 6F                    MOV   L,A
17B3 E6 01                 ANI   LCOVER  TEST THE COVERS
17B5 C8                    RZ    .       IF CLOSED, RETURN NC
17B6 7D                    MOV   A,L
17B7 E6 04                 ANI   RCOVER
17B9 C8                    RZ    .
17BA 7D                    MOV   A,L
17BB E6 88                 ANI   ENDWIRE*11H  IF END-OF-WIRE ON EITHER BOOM,
17BD EE 88                 XRI   ENDWIRE*11H  RETURN NO CARRY
17BF C0                    RNZ   .
17C0                     *
17C0 CD D6 17              CALL  REFLEN  GET BOOM LENGTH
17C3 21 48 20              LXI   H,LIMIT  AND COMPARE TO THE LIMIT
17C6 BE                    CMP   M
17C7 C9                    RET
17C8                     *
17C8                     * COMPARE BOOM LENGTHS IF BOTH BOOMS COMMANDED
17C8                     * IF ONLY 1 COMMANDED, RETURN CARRY
17C8                     * IF BOTH COMMANDED, RETURN THIS-THAT BOOM LENGTH
17C8                     *
17C8 3A 44 20            CMPLENG LDA   DEPSTAT  CHECK FOR BOTH COMMANDED
17CB E6 11                 ANI   CMDED*11H
17CD FE 11                 CPI   CMDED*11H
17CF 37                    STC
17D0 C0                    RNZ   .
17D1 CD D6 17              CALL  REFLEN  [A]=THIS BOOM, [L]=OTHER
17D4 95                    SUB   L
17D5 C9                    RET
17D6                     *
17D6                     * REFERENCE BOTH BOOM LENGTHS.
17D6                     * [A]=THIS BOOM, [L]=THAT BOOM.
17D6                     *
17D6 2A 46 20            REFLEN LHLD  DLEN1  H=DLEN2, L=DLEN1
17D9 3A 4A 20              LDA   DEPCNT
17DC 0F                    RRC   .
17DD 7C                    MOV   A,H    IF MOTOR 2, RETURN(LEN2,LEN1)
17DE D8                    RC    .
17DF 7D                    MOV   A,L    ELSE RETURN(LEN1,LEN2)
17E0 6C                    MOV   L,H
17E1 C9                    RET
17E2 00            V       DB    256    END OF DEPLOYMENT
17E3                     *
17E3                     * ENTER COMMAND VECTOR INTO TABLE
17E3                     *
17E3                       ORG   DEPLOY/4+CMDTAB
0074 DC 16                 DW    DEPCMD
0076                     *
0076                     * VARIABLES
```

```
0076            0250 *
0076            0251        ORG    DEPRAM
2044            0252 DEPSTAT DS    1      DEPLOY STATE [CODE1:CODE0]
2045            0253 BOOMBITS DS   1      BOOM STATUS BITS
2046            0254 DLEN1  DS     1      LENGTH OF BOOM 1
2047            0255 DLEN2  DS     1      LENGTH OF BOOM 2
2048            0256 LIMIT  DS     1      BOOM LENGTH UPPER LIMIT
2049            0257 OVERRIDE DS   1      SWITCH OVERRIDE BITS
204A            0258 DEPCNT DS     1      DEPLOY COUNTER (MTR IN LSB)
```

```
0000                 *
0000                 * CRRES FLIGHT PROGRAM --- BIAS SWEEPS
0000                 *
0000                 * FILE SWP.A
0000                 *
0000                 SWPCODE EQU   60H    COMMAND CODE FOR SWEEP MODULE
0000                 SWPR8   EQU   SWPCODE+2  COMMAND CODE FOR REBIASING
0000                 V1SET   EQU   68H    COMMAND CODE FOR ELE MODE SET
0000                 SAWENA  EQU   05CH   COMMAND CODE FOR SAWTOOTH OPTIONS
0000                 XMTCODE EQU   0EE0H  SWEEEEP XMIT CODE
0000                 *
0000                 V1ANG   EQU   128+32  SUN ANGLE WHEN BOOM1 SUNWARD
0000                 V3ANG   EQU   V1ANG+64  AND BOOM3 SUNWARD
0000                 V1FIT   EQU   V1ANG  V12 FITS START WHEN V1 SUNWARD
0000                 V3FIT   EQU   V1FIT+128  V34 FITS ARE OFF 180 DEGREES
0000                 V1SWP   EQU   V1ANG-64-8  SWEEP AT 112 DEGREES BEFORE SUN
0000                 V3SWP   EQU   V3ANG-64-8
0000                 *
0000                 V1      EQU   6      MUX ADDRESSES
0000                 V2F     EQU   4
0000                 R11     EQU   8
0000                 R12     EQU   V2F
0000                 V3      EQU   3
0000                 V4      EQU   12
0000                 *
0000                 PSW     EQU   6
0000                 SP      EQU   6
0000                 SRHL    EQU   10H    SHIFT RIGHT OPCODE
0000                 *
0000                         ORG   SWP
17E4 C3 04 18                JMP   SWPINIT
17E7 C3 31 18                JMP   SWPANG
17EA C3 FD 17                JMP   SWPSTAT
17ED C3 A2 18                JMP   SWPEXEC
17F0                 *
17F0                 * SWEEP DIGITAL STATUS
17F0                 * RETURNS A=OPTION[BOOM,A]
17F0                 *
17F0 FE 06           SWPDSC  CPI   6      IF 0-5, RETURN RAM12
17F2 DA F7 17                JC    SD1
17F5 C6 09                   ADI   RAM34-RAM12-6  ELSE RAM34
17F7 21 C3 24        SD1     LXI   H,RAM12
17FA DF                      RST   REF/B
17FB 7E                      MOV   A,M
17FC C9                      RET
17FD                 *
17FD                 * SWEEP FAST DIGITAL STATUS
17FD                 *
17FD 3A E2 24        SWPSTAT LDA   SWPREQ  RETURN LSB=1 FOR SWEEPING
```

```
1800 32 E3 24           STA    SWPOK  (SYNC TO WHEN THIS CALL IS MADE)
1803 C9                 RET
1804             *
1804             * INITIALIZATION/DEFAULT SETTINGS
1804             *
1804 21 C1 24    SWPINIT LXI   H,STATE  COPY FROM ROM TO RAM
1807 11 2D 18           LXI    D,ROMDEF
180A 0E 11              MVI    C,RAM34-RAM12+2
180C D7                 RST    COPY/8
180D 11 2F 18           LXI    D,ROMDEF+2
1810 0E 0F              MVI    C,RAM34-RAM12
1812 D7                 RST    COPY/8
1813 36 01              MVI    M,1    BOOM = 1
1815 3E 98              MVI    A,V3SWP*256/256  CHANGE SWEEP ANGLE OF V34
1817 32 D3 24           STA    RAM34+ANGSWP
181A C9                 RET
181B             *
181B             * COMMAND ENTRY IN [HL]
181B             *
181B 7C          SWPCMD MOV    A,H    IF REBIAS COMMAND, SETRESULT(L)
181C FE 62              CPI    SWPRB
181E CA 7F 18           JZ     REBIAS
1821 EB                 XCHG   .      [DE]=COMMAND
1822 21 C0 24           LXI    H,INDEX  [HL]->INDEX
1825 0F                 RRC    .      IF EVEN, SET INDEX
1826 D2 2F 18           JNC    SI
1829 34                 INR    M      ELSE SET VALUE(INDEX++)
182A 7E                 MOV    A,M
182B FE 22              CPI    BOOM-INDEX+1
182D D0                 RNC
182E DF                 RST    REF/8
182F 73          SI     MOV    M,E
1830 C9                 RET
1831             *
1831             * SWEEP SUN ANGLE
1831             *
1831 5F          SWPANG MOV    E,A    SAVE SUN ANGLE
1832 B7                 ORA    A      COUNT ZERO CROSSINGS
1833 CC 67 18           CZ     DNCNT
1836             *
1836 01 C1 24           LXI    B,STATE  IN SEARCH STATE,
1839 0A                 LDAX   B      CHECK ANGLE FOR START OF SWEEP
183A E6 03              ANI    3
183C CA 6F 18           JZ     CHKANG
183F D6 02              SUI    2      IF ANALYSIS NOT DONE, QUIT
1841 C0                 RNZ    .
1842             *
1842 CD 02 18           CALL   REFCON  IS IT TIME TO SET BIAS?
1845 23                 INX    H
1846 7E                 MOV    A,M    (CON[1]==ANGLE)
```

```
1847 93                 SUB    E
1848 C0                 RNZ    .      NO;RETURN
1849 02                 STAX   B      YES, STATE=0 (SEARCH AGAIN)
184A             *
184A 0E 10              MVI    C,10H  CHECK IF THIS MODE IS OK
184C CD E2 1A           CALL   CHKWTD  FOR SETTING BIASES
184F 2A E1 24           LHLD   BOOM
1852 C4 85 18           CNZ    SETRES  YES. SET BIASES(BOOM)
1855             *
1855 21 E1 24    SWITCH LXI    H,BOOM  SWITCH TO OTHER BOOM
1858 35                 DCR    M      IF 1, SET TO 3 AND VICE VERSA
1859 36 03              MVI    M,3
185B CA 60 18           JZ     SETMAX
185E 36 01              MVI    M,1
1860             *
1860 3A C2 24    SETMAX LDA    SPINMAX  SET DELAY IN #SPINS
1863 32 E4 24           STA    SPINCNT
1866 C9                 RET
1867             *
1867 21 E4 24    DNCNT  LXI    H,SPINCNT  COUNT DOWN
186A 35                 DCR    M
186B F0                 RP     .      IF ZERO OR GREATER, OK
186C 36 00              MVI    M,0
186E C9                 RET
186F             *
186F 3A E4 24    CHKANG LDA    SPINCNT  IF SPIN COUNTER=0
1872 B7                 ORA    A
1873 C0                 RNZ
1874 3E 01              MVI    A,ANGSWP  AND THE ANGLE=SWPANG
1876 CD FC 1A           CALL   GETOPT
1879 BB                 CMP    E
187A C0                 RNZ    .
187B 3E 01              MVI    A,1    SET READY TO SWEEP
187D 02                 STAX   B
187E C9                 RET
187F             *
187F 3A C1 24    REBIAS LDA    STATE  IF NOT DISABLED
1882 FE 03              CPI    3      THEN SET RESULT(BOOM=L) ON SENSORS
1884 C8                 RZ     .
1885             *
1885             * SET THE RESULT ON THE SENSORS OF BOOM[L]
1885             *
1885 55          SETRES MOV    D,L
1886 0E 10              MVI    C,10H  CHECK WHETHER TO USE RESULT
1888 CD DC 1A           CALL   CHKENA  OR THE ALTERNATE VALUES
188B 3E 03              MVI    A,ALT
188D CA 92 18           JZ     SR1
1890 3E 04              MVI    A,RESULT
1892 6A          SR1    MOV    L,D    HL->VARIABLES(BOOM L)
1893 CD F3 1A           CALL   REFL
```

```
1896 DF               RST   REF/8  L=ALTERNATE OR RESULT
1897 6E               MOV   L,M
1898 62               MOV   H,D
1899             *
1899             * BIAS A SENSOR PAIR
1899             * ON ENTRY: [H] = 1 OR 3, [L]=VALUE
1899             *
1899 E5          BIBOTH PUSH H
189A CD 63 01         CALL  SETBIAS
189D E1               POP   H
189E 24               INR   H
189F C3 63 01         JMP   SETBIAS
18A2             *
18A2             * SWEEP EXECUTIVE SECTION
18A2             *
18A2 3A C1 24    SWPEXEC LDA STATE  IF SWEEP ANGLE NOT MATCHED
18A5 3D               DCR   A       RETURN.
18A6 C0               RNZ   .
18A7             *
18A7 21 10 25         LXI   H,SWPBUF  SET WHERE DATA GOES
18AA 22 EE 24         SHLD  SWPPTR
18AD             *
18AD 3E 04            MVI   A,4     REQUEST SAWTOOTH MODE
18AF CD 36 13         CALL  SAWDSC
18B2 32 14 29         STA   SAWMODE
18B5             *
18B5             * VOLTAGE PHASE (DO A CURRENT SWEEP)
18B5             *
18B5 01 01 07         LXI   B,ISTEP*256+1  B=ISTEP,C=1
18B8 CD DA 18         CALL  SWEEP
18BB             *
18BB             * CURRENT PHASE (DO A VOLTAGE SWEEP)
18BB             *
18BB 01 04 09         LXI   B,VSTEP*256+4  B=VSTEP,C=4
18BE CD DA 18         CALL  SWEEP
18C1             *
18C1 97               SUB   A       REMOVE REQUEST
18C2 32 E2 24         STA   SWPREQ
18C5             *
18C5 2A 14 29         LHLD  SAWMODE  RESTORE THE OLD SAWTOOTH MODE
18C8 26 5C            MVI   H,SAWENA
18CA F7               RST   6
18CB             *
18CB 0E 01            MVI   C,1     IF THERE WAS A VOLTAGE PHASE
18CD CD DC 1A         CALL  CHKENA  THEN ANALYZE THE DATA
18D0 C4 87 19         CNZ   ANALYZE
18D3             *
18D3 21 C1 24         LXI   H,STATE  APPLY ANALYSIS DONE.
18D6 34               INR   M
18D7 C3 A5 1A         JMP   TRANSMIT
```

```
18DA              *
18DA              * PERFORM A SWEEP ON THE SELECTED BOOM SYSTEM IF ENABLED
18DA              * ON ENTRY: [B]= VSTEP OR ISTEP INDEX
18DA              *           [C]= 1 FOR V PHASE, 4 FOR I PHASE
18DA              *
18DA CD DC 1A     SWEEP  CALL  CHKENA  CHECK IF SWEEP ENABLED
18DD C8                  RZ
18DE              *
18DE 21 E2 24            LXI   H,SWPREQ  SET REQUEST=1
18E1 36 01               MVI   M,1    SET REQUEST=1
18E3 23                  INX   H
18E4 7E           SWPWT  MOV   A,M WAIT FOR THE OK
18E5 3D                  DCR   A
18E6 C2 E4 18            JNZ   SWPWT
18E9              *
18E9 CD E2 1A            CALL  CHKWTD  CHECK IF WE'LL NEED TO FLIP
18EC 32 15 29            STA   NEEDFLIP
18EF              *
18EF 78                  MOV   A,B    GET ISTEP/VSTEP INDEX
18F0 CD FC 1A            CALL  GETOPT  [HL]->ISTEP OR VSTEP
18F3 CD 1D 18            CALL  LD2
18F6 22 E9 24            SHLD  SWPDEL  (DEL AND BIAS)
18F9              *
18F9 CD 02 18            CALL  REFCON  GET THE SWPPAIR
18FC 79                  MOV   A,C    AND QTYA/8 FROM TABLE
18FD 3C                  INR   A      (INDEX=2 OR 5)
18FE DF                  RST   REF/8
18FF EB                  XCHG
1900 21 EB 24            LXI   H,SWPPAIR
1903 0E 03               MVI   C,3
1905 D7                  RST   COPY/8
1906              *
1906 21 01 5C            LXI   H,SAWENA*256+1  TURN OFF SAWTOOTH
1909 F7                  RST   6      BUT ALLOW IT TO KEEP TIME
190A              *
190A CD 25 19            CALL  FLIP   FLIP RELAYS IF WE HAVE TO.
190D 01 80 04            LXI   B,400H+128  WAIT 4 TIMES AT FIRST
1910 CD 4C 19     SWPLP  CALL  BIASEM
1913 CD 3D 19            CALL  WAIT
1916 CD 5B 19            CALL  ISAMP
1919 06 01               MVI   B,1    WAIT ONCE AFTER THAT
191B 0D                  DCR   C
191C C2 10 19            JNZ   SWPLP
191F 2A EB 24            LHLD  SWPPAIR  PUT PRIOR RESULT FOR THE
1922 CD 85 18            CALL  SETRES  SWEPT BOOM SYSTEM BACK OUT
1925              *
1925 3A 15 29     FLIP   LDA   NEEDFLIP  DO WE NEED TO FLIP?
1928 B7                  ORA   A
1929 C8                  RZ    .      NO. RETURN.
192A CD D2 13            CALL  ELESTAT  YES. INVERT THE MODE
```

```
192D 3C                 INR   A
192E E6 01              ANI   1
1930 6F                 MOV   L,A
1931 26 6B              MVI   H,VISET
1933 F7                 RST   6
1934 C9                 RET
1935               *
1935 3A 1C 20      WAITN LDA   WORD
1938 E6 07              ANI   7
193A C2 35 19           JNZ   WAITN
193D               *
193D 3A 1C 20      WAIT  LDA   WORD   SAMPLE IN WORDS
1940 D6 02              SUI   2      2,10,18...
1942 E6 07              ANI   7
1944 C2 3D 19           JNZ   WAIT
1947 05                 DCR   B
1948 C2 35 19           JNZ   WAITN
194B C9                 RET
194C               *
194C 2A EA 24      BIASEN LHLD SWPBIAS  BIAS SENSORS
194F CD 99 1B           CALL  BIBOTH
1952               *
1952 3A E9 24      STEPB LDA   SWPDEL  BIAS += DELTA
1955 21 EA 24           LXI   H,SWPBIAS
1958 86                 ADD   M
1959 77                 MOV   M,A
195A C9                 RET
195B               *
195B 3A EC 24      ISAMP LDA   QTYA   SAMPLE/STORE QTYA
195E CD 64 19           CALL  SAMSTO
1961 3A ED 24           LDA   QTYB
1964 CD 78 19      SAMSTO CALL SSAMP
1967 29                 DAD   H      SIGN EXTEND 12 TO 16 BITS
1968 29                 DAD   H      (SHIFTING LEFT, THEN RIGHT)
1969 29                 DAD   H
196A 29                 DAD   H
196B 10                 DB    SRHL
196C 10                 DB    SRHL
196D 10                 DB    SRHL
196E 10                 DB    SRHL
196F               *
196F EB                 XCHG  .      TO [DE]
1970 2A EE 24           LHLD  SWPPTR  PUT AWAY
1973 73                 MOV   M,E
1974 23                 INX   H
1975 72                 MOV   M,D
1976 23                 INX   H
1977 22 EE 24           SHLD  SWPPTR
197A C9                 RET
197B               *
```

```
197B B7          SSAMP   ORA    A       IF POSITIVE MUX QTY
197C F2 E6 00            JP     SAMPLE  TAKE A NORMAL SAMPLE
197F 2F                  CMA    .       INVERT MUX QTY
1980 3C                  INR    A
1981 CD E6 00            CALL   SAMPLE  ELSE TAKE SAMPLE
1984 C3 8B 00            JMP    NEG16  AND INVERT
1987             *
1987             * ANALYZE SWEEPS
1987             *
1987 3A E5 24    ANALYZE LDA    ANAVECT  IF USER VECTOR
198A FE AA               CPI    0AAH   IS SET, THEN GO
198C CA E6 24            JZ     ANAVECT+1
198F             *
198F 3E 02               MVI    A,ALGNO  SET ALGORITHM # 0
1991 CD FC 1A            CALL   GETOPT
1994 36 00               MVI    M,0
1996             *
1996 3E 0B               MVI    A,PTSREJ  ZERO THE # POINTS REJECTED
1998 CD FC 1A            CALL   GETOPT
199B 36 00               MVI    M,0
199D             *
199D 21 10 25            LXI    H,SWPBUF+0 GET ANA V1 OR V3
19A0 CD B6 19            CALL   ANA
19A3 F5                  PUSH   PSW     SAVE RESULT
19A4 21 12 25            LXI    H,SWPBUF+2  ADD ANA V2 OR V4
19A7 CD B6 19            CALL   ANA
19AA 5F                  MOV    E,A     AVERAGE THE RESULTS
19AB F1                  POP    PSW
19AC 83                  ADD    E
19AD 1F                  RAR
19AE 5F                  MOV    E,A
19AF 3E 04               MVI    A,RESULT  SET IT
19B1 CD FC 1A            CALL   GETOPT
19B4 73                  MOV    M,E
19B5 C9                  RET
19B6             *
19B6             * ANALYZE ONE SENSOR SWEEP
19B6             *
19B6 22 10 29    ANA     SHLD   BPTR   SAVE BUFFER START ADDRESS
19B9 3E 06               MVI    A,NREJ  REJECT(N)
19BB CD FC 1A            CALL   GETOPT
19BE CD E1 19            CALL   SMOOTH
19C1             *
19C1 3E 05               MVI    A,MAVG  FINDMIN(M)
19C3 CD FC 1A            CALL   GETOPT
19C6 CD 4C 1A            CALL   FINDMIN  A=INDEX OF MINIMUM
19C9 5F                  MOV    E,A
19CA             *
19CA FE FF               CPI    -1      IF NULL INDEX, USE ALTERNATE
19CC 3E 03               MVI    A,ALT
```

```
19CE CA FC 1A          JZ     GETOPT
19D1                 *
19D1                 * CALCULATE THE CURRENT FROM THE INDEX
19D1                 *
19D1 3E 07             MVI    A,ISTEP  A=INITIAL STEP
19D3 CD FC 1A          CALL   GETOPT
19D6 16 00             MVI    D,0     L= STEP*INDEX
19D8 E5                PUSH   H
19D9 CD 0A 06          CALL   MU21
19DC D1                POP    D
19DD                 *
19DD 13                INX    D       PICK UP INITIAL BIAS
19DE 1A                LDAX   D
19DF 85                ADD    L       ADD IT IN
19E0 C9                RET
19E1                 *
19E1                 * SMOOTH THE I/V CURVE
19E1                 * ON ENTRY: [A] = * TIMES TO OPERATE
19E1                 *
19E1 32 12 29        SMOOTH STA    NTEMP
19E4 3C                INR    A       IF -1, NO SMOOTHING
19E5 C8                RZ     .
19E6 06 7E             MVI    B,126   FOR B=126 TO 2
19E8 CD F3 19        SM1    CALL   SMOCHK  CHECK/REPLACE PT[B]
19EB 05                DCR    B
19EC 78                MOV    A,B
19ED FE 02             CPI    2
19EF D2 E8 19          JNC    SM1
19F2 C9                RET
19F3                 *
19F3 78              SMOCHK MOV    A,B     IF PT[B]-PT[B-1]+N>0
19F4 CD 0E 18          CALL   LOADV   THEN POINT IS OK.
19F7 C8                RZ     .       (SATURATED)
19F8 3A 12 29          LDA    NTEMP
19FB DF                RST    REF/8
19FC EB                XCHG   .
19FD 78                MOV    A,B
19FE 3D                DCR    A
19FF CD 0E 18          CALL   LOADV
1A02 C8                RZ     .
1A03 CD 88 00          CALL   NEG16
1A06 19                DAD    D
1A07 7C                MOV    A,H
1A08 B7                ORA    A
1A09 F0                RP     .
1A0A                 *
1A0A                 * SLOPE IS FOUND TO BE TOO NEGATIVE FROM [B-1] TO [B]
1A0A                 * DECIDE WHICH OF [B] AND [B-1] IS THE PROBLEM
1A0A                 *
1A0A 78                MOV    A,B     CALCULATE -[B]-[B-1]
```

```
1A0B CD 0E 1B          CALL   LOADV
1A0E EB                XCHG
1A0F 78                MOV    A,B
1A10 3D                DCR    A
1A11 CD 0E 1B          CALL   LOADV
1A14 19                DAD    D
1A15 CD 88 00          CALL   NEG16
1A18 EB                XCHG   .
1A19             *
1A19 78                MOV    A,B     +[B+1]
1A1A 3C                INR    A
1A1B CD 0E 1B          CALL   LOADV
1A1E C8                RZ     .
1A1F 19                DAD    D
1A20 EB                XCHG
1A21             *
1A21 78                MOV    A,B     +[B-2]
1A22 D6 02             SUI    2
1A24 CD 0E 1B          CALL   LOADV
1A27 C8                RZ     .
1A28 19                DAD    D
1A29             *
1A29 7C                MOV    A,H     IF POSITIVE, REPLACE [B]
1A2A B7                ORA    A
1A2B 78                MOV    A,B
1A2C F2 30 1A          JP     INTERP
1A2F 3D                DCR    A       ELSE REPLACE [B-1]
1A30             *
1A30             * INTERPOLATE POINT [A] FROM SURROUNDING POINTS
1A30             *
1A30 4F          INTERP MOV   C,A     INTERPOLATE PT[X]
1A31 3C                INR    A
1A32 CD 0E 1B          CALL   LOADV
1A35 EB                XCHG
1A36             *
1A36 79                MOV    A,C
1A37 3D                DCR    A
1A38 CD 0E 1B          CALL   LOADV
1A3B 19                DAD    D
1A3C 10                DB     SRHL
1A3D EB                XCHG
1A3E             *
1A3E 79                MOV    A,C
1A3F CD 22 1B          CALL   REFV
1A42 73                MOV    M,E
1A43 23                INX    H
1A44 72                MOV    M,D
1A45             *
1A45 3E 0B             MVI    A,PTSREJ
1A47 CD FC 1A          CALL   GETOPT
```

```
1A4A 34                    INR    M
1A4B C9                    RET
1A4C               ;
1A4C               ; FIND MIN DELTA-V
1A4C               ; ON ENTRY: [A] = # PTS TO AVERAGE (M)
1A4C               ; ON EXIT : [A] = INDEX OF MINIMUM DELTA V
1A4C               ;
1A4C 11 FF 7F      FINDMIN LXI    D,7FFFH  [DE]=MAX VALUE
1A4F 4F                    MOV    C,A     C = M
1A50 3E FF                 MVI    A,-1    MININX = -1
1A52 32 13 29              STA    MININX
1A55               ;
1A55 3E 7E                 MVI    A,127-1  START AT 127 - M - 1
1A57 91                    SUB    C
1A58 47                    MOV    B,A
1A59               ;
1A59 D5            DVLOP   PUSH   D
1A5A CD 74 1A              CALL   GETDV   COMPUTE DELTA V
1A5D D1                    POP    D
1A5E 7D                    MOV    A,L     COMPARE TO OLD MININUM
1A5F 93                    SUB    E
1A60 7C                    MOV    A,H
1A61 9A                    SBB    D
1A62 D2 6A 1A              JNC    DVCONT
1A65 EB                    XCHG   .       IF LESS, SAVE NEW MINIMUM
1A66 78                    MOV    A,B     AND RECORD THE INDEX
1A67 32 13 29              STA    MININX
1A6A 05            DVCONT  DCR    B
1A6B 78                    MOV    A,B     CONTINUE UNTIL B<M
1A6C B9                    CMP    C
1A6D D2 59 1A              JNC    DVLOP
1A70 3A 13 29              LDA    MININX  RETURN INDEX OF MINIMUM
1A73 C9                    RET
1A74               ;
1A74               ; COMPUTE DELTA-V FOR POINT AT [B]
1A74               ; ON ENTRY: [B] = INDEX INTO ARRAY
1A74               ;           [C] = #PTS TO AVERAGE
1A74               ; ON EXIT : [HL] = DELTA-V OR MAX IF SATURATED
1A74               ;
1A74 78            GETDV   MOV    A,B     COMPUTE SUM(B+1,M)
1A75 3C                    INR    A       MINUS SUM(B-M,M)
1A76 CD 87 1A              CALL   SUM
1A79 D8                    RC     .
1A7A E5                    PUSH   H
1A7B 78                    MOV    A,B
1A7C 91                    SUB    C
1A7D CD 87 1A              CALL   SUM
1A80 D1                    POP    D
1A81 D8                    RC     .
1A82 CD 8B 00              CALL   NEG16
```

```
1A85 19               DAD    D
1A86 C9               RET
1A87                *
1A87                * COMPUTE THE SUM OF 'C' SAMPLES
1A87                * ON ENTRY: [A]=INDEX INTO THE ARRAY
1A87                * ON EXIT : [HL]= SUM OR 7FFFH IF SATURATED (HIGH OR LOW)
1A87                *
1A87 11 00 00       SUM      LXI    D,0     SUM=0
1A8A C5                      PUSH   B
1A8B F5             SUM1     PUSH   PSW
1A8C CD 0E 1B                CALL   LOADV   ADD 1 VALUE
1A8F CA 9E 1A                JZ     OVER    IF SATURATED, RETURN
1A92 19                      DAD    D
1A93 EB                      XCHG   .
1A94 F1                      POP    PSW
1A95 3C                      INR    A       STEP TO NEXT
1A96 0D                      DCR    C       FOR C TIMES
1A97 C2 8B 1A                JNZ    SUM1
1A9A EB                      XCHG   .       RESULT TO [HL]
1A9B C1                      POP    B
1A9C B7                      ORA    A       RETURN NO CARRY
1A9D C9                      RET
1A9E                *
1A9E F1             OVER     POP    PSW
1A9F C1                      POP    B
1AA0 21 FF 3F                LXI    H,03FFFH  RETURN MAX
1AA3 37                      STC    .       WITH CARRY SET
1AA4 C9                      RET
1AA5                *
1AA5                * SWEEP TRANSMIT
1AA5                *
1AA5 0E 40          TRANSMIT MVI    C,40H   CHECK TRANSMIT ENABLE
1AA7 CD DC 1A                CALL   CHKENA
1AAA C8                      RZ
1AAB CD F0 1A                CALL   REFOPT
1AAE EB                      XCHG   .       COPY THE OPTIONS/RESULTS
1AAF 21 F2 24                LXI    H,SWPHDR+2  INTO THE HEADER
1AB2 0E 0F                   MVI    C,SWPBUF-SWPHDR-2/2  CONVERT
1AB4 CD D0 1A                CALL   EXPAND  FROM BYTE TO INT
1AB7                *
1AB7 21 E0 0E                LXI    H,XMTCODE  SET HEADER FOR WHICH SWEEP
1ABA 3A E1 24                LDA    BOOM
1ABD DF                      RST    REF/8
1ABE 22 F0 24                SHLD   SWPHDR
1AC1                *
1AC1 2A EE 24                LHLD   SWPPTR  COMPUTE LENGTH OF PLAYBACK
1AC4 11 10 DB                LXI    D,-SWPHDR
1AC7 19                      DAD    D       ADD,THEN DIV BY 2
1AC8 10                      DB     SRHL
1AC9 EB                      XCHG   .
```

```
1ACA 21 F0 24     0550        LXI   H,SWPHOR  TELL THE TELEMETRY FORMATTER
1ACD C3 CF 13     0551        JMP   ELEXMIT
1AD0              0552 *
1AD0 1A           0553 EXPAND LDAX  D       CONVERT BYTE TO INT ARRAY
1AD1 13           0554        INX   D
1AD2 77           0555        MOV   M,A
1AD3 23           0556        INX   H
1AD4 36 00        0557        MVI   M,0
1AD6 23           0558        INX   H
1AD7 0D           0559        DCR   C
1AD8 C2 D0 1A     0560        JNZ   EXPAND
1ADB C9           0561        RET
1ADC              0562 *
1ADC              0563 * OPTIONS REFERENCING
1ADC              0564 *
1ADC CD F0 1A     0565 CHKENA CALL  REFOPT  CHECK ENABLE OPTIONS
1ADF C3 E5 1A     0566        JMP   CHK1
1AE2 CD 02 1B     0567 CHKWTD CALL  REFCON  CHECK ENABLE CONSTANTS
1AE5 CD D2 13     0568 CHK1   CALL  ELESTAT  GET THE I/V MODE
1AE8 0F           0569        RRC
1AE9 79           0570        MOV   A,C
1AEA D2 EE 1A     0571        JNC   GM1
1AED 87           0572        ADD   A       IN IMODE, MOVE BITS OVER
1AEE A6           0573 GM1    ANA   M
1AEF C9           0574        RET
1AF0              0575 *
1AF0 2A E1 24     0576 REFOPT LHLD  BOOM    L=BOOM NUMBER (1,3 = V12/V34)
1AF3 2D           0577 REFL   DCR   L       IF 1, RETURN V12 LIST
1AF4 21 C3 24     0578        LXI   H,RAM12
1AF7 C8           0579        RZ
1AF8 21 D2 24     0580        LXI   H,RAM34
1AFB C9           0581        RET
1AFC              0582 *
1AFC CD F0 1A     0583 GETOPT CALL  REFOPT  [HL]->OPTIONS
1AFF DF           0584        RST   REF/8  [HL]-> OPTION[A]
1B00 7E           0585        MOV   A,M
1B01 C9           0586        RET
1B02              0587 *
1B02 2A E1 24     0588 REFCON LHLD  BOOM    RETURN CONSTANTS
1B05 2D           0589        DCR   L
1B06 21 3E 1B     0590        LXI   H,CON12
1B09 C8           0591        RZ
1B0A 21 46 1B     0592        LXI   H,CON34
1B0D C9           0593        RET
1B0E              0594 *
1B0E CD 22 1B     0595 LOADV  CALL  REFV    LOAD 16 BIT FROM ARRAY
1B11 CD 1D 1B     0596        CALL  LD2
1B14 7C           0597        MOV   A,H
1B15 FE 07        0598        CPI   7
1B17 C8           0599        RZ
```

```
1B18 E6 0F          0600        ANI    15
1B1A FE 08          0601        CPI    8
1B1C C9             0602        RET    .
1B1D                0603 *
1B1D 7E             0604 LD2    MOV    A,M
1B1E 23             0605        INX    H
1B1F 66             0606        MOV    H,M
1B20 6F             0607        MOV    L,A
1B21 C9             0608        RET
1B22                0609 *
1B22 2A 10 29       0610 REFV   LHLD   BPTR   REFERENCE ARRAY[A]
1B25 87             0611        ADD    A      (TIMES 2)
1B26 87             0612        ADD    A      (TIMES 4)
1B27 D2 2B 1B       0613        JNC    REFV1  IF NO CARRY, HL=HL+A
1B2A 24             0614        INR    H
1B2B DF             0615 REFV1  RST    REF/8
1B2C C9             0616        RET
1B2D                0617 *
1B2D                0618 * RAM AREA DEFINITIONS
1B2D                0619 *
1B2D                0620 OPT    EQU    0      OPTIONS
1B2D                0621 ANGSWP EQU    1      SWEEP ANGLE
1B2D                0622 ALGNO  EQU    2      ALGORITHM #
1B2D                0623 ALT    EQU    3      BIAS ALTERNATE
1B2D                0624 RESULT EQU    4      BIAS RESULT
1B2D                0625 MAVG   EQU    5      #POINTS TO AVERAGE
1B2D                0626 NREJ   EQU    6      NOISE PASS LIMIT
1B2D                0627 ISTEP  EQU    7      IBIAS STEP (VMODE)
1B2D                0628 IB0    EQU    8      INITIAL IBIAS (VMODE)
1B2D                0629 VSTEP  EQU    9      VBIAS STEP (IMODE)
1B2D                0630 VB0    EQU    10     INITIAL VOLTAGE (IMODE)
1B2D                0631 PTSREJ EQU    11     #POINTS REJECTED BY SMOOTHING
1B2D                0632 *
1B2D                0633 *  DEFAULT FOR V12 AND V34
1B2D                0634 *
1B2D 00             0635 ROMDEF DB     0      INITIAL STATE=0
1B2E 08             0636        DB     8      #SPINS BETWEEN SWITCHING (4 MINUTES)
1B2F                0637 *
1B2F CF             0638        DB     0CFH   USE ALTERNATE RESULT, ALL ELSE ENABLED
1B30 58             0639        DB     V1SWP  ANGLE TO SWEEP AT
1B31 00             0640        DB     0      ALG#0
1B32 F0             0641        DB     -16    ALTERNATE = -40 NANOAMPS
1B33 FF             0642        DB     -1     RESULT
1B34 05             0643        DB     5      M=5
1B35 08             0644        DB     8      N=8
1B36 02             0645        DB     2      ISTEP VALUE
1B37 81             0646        DB     -127   IBIAS0
1B38 02             0647        DB     2      STEP VALUE
1B39 81             0648        DB     -127   VBIAS0
1B3A 00             0649        DB     0
```

```
1B3B 00              0650        DB     0
1B3C 00              0651        DB     0
1B3D 00              0652        DB     0
1B3E                 0653 *
1B3E 16              0654 CON12  DB     016H
1B3F 9F              0655        DB     V1FIT-1  SET BIAS1&2 VALUES JUST BEFORE FIT
1B40 01              0656        DB     1       BIAS 1,2 PAIR IN VOLT PHASE
1B41 06              0657        DB     V1      MEASUREMENT QTYS
1B42 FC              0658        DB     -V2F
1B43 01              0659        DB     1       BIAS 1,2 PAIR IN CURR PHASE
1B44 0B              0660        DB     RI1
1B45 04              0661        DB     RI2
1B46                 0662 *
1B46 34              0663 CON34  DB     034H
1B47 1F              0664        DB     V3FIT-1*256/256  CHANGE BIAS34 BEFORE FIT
1B48 03              0665        DB     3       BIAS 3,4 IN VOLT PHASE
1B49 03              0666        DB     V3
1B4A 0C              0667        DB     V4
1B4B 01              0668        DB     1       BIAS 1,2 IN CURR PHASE
1B4C 0B              0669        DB     RI1     MEASURE RI1/2
1B4D 04              0670        DB     RI2
1B4E 00            V 0671        DB     256     END OF SWP
1B4F                 0672 *
1B4F                 0673 * ENTER COMMAND VECTOR INTO TABLE
1B4F                 0674 *
1B4F                 0675        ORG    SWPCODE/4+CMDTAB
005B 1B 1B           0676        DW     SWPCMD
005A                 0677 *
005A                 0678 * SWP RAM AREA
005A                 0679 *
005A                 0680        ORG    SWPRAM
24C0                 0681 INDEX  DS     1       COMMAND VALUE ENTRY
24C1                 0682 STATE  DS     1       STATE OF SWEEP MODULE
24C2                 0683 SPINMAX DS    1       #SPINS BETWEEN SWEEPS
24C3                 0684 RAM12  DS     15      PARAMETERS FOR BOOMS 1,2
24D2                 0685 RAM34  DS     15      PARAMETERS FOR BOOMS 3,4
24E1                 0686 BOOM   DS     1       BOOM SYSTEM (0=V12)
24E2                 0687 *
24E2                 0688 SWPREQ DS     1       SWEEP REQUEST
24E3                 0689 SWPOK  DS     1       SWEEP REQUEST OK
24E4                 0690 SPINCNT DS    1       SPIN COUNTER
24E5                 0691 ANAVECT DS    4       VECTOR = (AA,C3,XX,XX)
24E9                 0692 *
24E9                 0693 SWPDEL DS     1       BIAS DELTA
24EA                 0694 SWPBIAS DS    1       BIAS VALUE
24EB                 0695 SWPPAIR DS    1       SWP BIAS DACS TO USE
24EC                 0696 QTYA   DS     1       MUX QTYS
24ED                 0697 QTYB   DS     1
24EE                 0698 *
24EE                 0699 SWPPTR DS     2       SAMPLE POINTER
```

```
24F0          0700 SWPHDR DS     2*16   SWEEP HEADER INFO
2510          0701 SWPBUF DS     2*128*2*2  DATA AREA
2910          0702 SWPEND EQU    $
2910          0703 *
2910          0704 BPTR   DS     2      BUFFER POINTER
2912          0705 NTEMP  DS     1      N TEMPORARY
2913          0706 MININX DS     1      MINIMUM INDEX CALC
2914          0707 SAWMODE DS    1      SAWTOOTH MODE SAVED
2915          0708 NEEDFLIP DS   1      RELAYS NEED FLIPPING IF NZ
2916          0709 *
2916          0710 * EXTERNALS
2916          0711 *
2916          0712 ELEXMT EQU    ELE+0FH
2916          0713 ELESTAT EQU   ELE+12H
2916          0714 SAWDSC EQU    SAW+6
2916          0715 *
2916          0716 WORD   EQU    BKGRAM+0
```

```
0000            *
0000            * CRRES FLIGHT PROGRAM---EXECUTIVE CONTROL
0000            * WRITTEN BY PETER R HARVEY
0000            * FILE EXEC.A
0000            *
0000            * 8085 SPECIFIC INFORMATION
0000            *
0000            PSW      EQU    6
0000            SP       EQU    6
0000            *
0000            * RAM CONFIGURATION
0000            *
0000            RAM      EQU    2000H
0000            RAMSIZE  EQU    1000H
0000            STACK    EQU    RAM+RAMSIZE-1
0000            *
0000            * RESET VECTOR
0000            *
0000                     ORG    0
0000 C3 6D 1B            JMP    EXEINIT  FOREGROUND START
0003            BKGINIT  DS     3     BACKGROUND START
0006            *
0006            * MAIN PROCESSOR EXECUTIVE CONTROL
0006            *
0006                     ORG    EXEC
1B64 C3 FD 1B            JMP    EXEANG
1B67            *
1B67            * EXECUTIVE STATUS
1B67            *
1B67 21 A0 24   EXEDSC   LXI    H,VERSION  RETURN VARS
1B6A DF                  RST    REF/8
1B6B 7E                  MOV    A,M
1B6C C9                  RET
1B6D            *
1B6D            * BEGIN
1B6D            *
1B6D 31 FF 2F   EXEINIT  LXI    SP,STACK  INIT STACK POINTER
1B70 21 00 20            LXI    H,RAM  ZERO THE RAM
1B73 01 00 0F            LXI    B,RAMSIZE-256  EXCEPT FOR LAST PAGE
1B76 CF         CLEAR    RST    ZERO/8
1B77 05                  DCR    B
1B78 C2 76 1B            JNZ    CLEAR
1B7B            *
1B7B CD 9A 00            CALL   IOINIT  INIT THE IO MODULE
1B7E CD 03 00            CALL   BKGINIT  INIT THE BACKGROUND MGR
1B81 CD E4 17            CALL   SWPINIT  INIT EXECUTIVE MODULES
1B84 CD E8 11            CALL   FITINIT
1B87 11 40 1C            LXI    D,INISEQ  SEND INITIAL COMMANDS
1B8A CD 2A 1C            CALL   CMDSTRING
```

```
1B8D 2A AB 24          LHLD   VTIME  STAY IN INITIAL MODE
1B90 22 AC 24          SHLD   MODCTR  FOR THE FULL TIME
1B93 2A C7 1B          LHLD   DEFVECT  RESET THE EXEVECTOR
1B96 22 B0 24          SHLD   EXEVECT
1B99                 *
1B99                 * MAIN EXECUTIVE LOOP
1B99                 *
1B99 3A B0 24        EXLOOP LDA    EXEVECT  IF EXECUTIVE VECTOR
1B9C FE AA             CPI    0AAH   IS ENABLED, CALL THE ROUTINE
1B9E C2 A7 1B          JNZ    EXWT
1BA1 CD B1 24          CALL   EXEVECT+1
1BA4 C3 B5 1B          JMP    EXMGR
1BA7                 *
1BA7 3E 21           EXWT   MVI    A,21H  VERSION=2.1
1BA9 32 A0 24          STA    VERSION
1BAC 2A C5 1B          LHLD   LPPROG  WAIT - LOW POWER
1BAF 22 AE 24          SHLD   PROG
1BB2 CD AE 24          CALL   PROG
1BB5                 *
1BB5 CD F1 11        EXMGR  CALL   FITEXEC  CYCLE BETWEEN CALCULATING
1BB8 CD ED 17          CALL   SWPEXEC  FITS AND SWEEP DATA
1BBB CD 0A 1C          CALL   DECMODE  DECIDE THE MODE (I/V)
1BBE 97                SUB    A      CLEAR THE EXECUTIVE WATCHDOG
1BBF 32 B4 24          STA    EXWATCH
1BC2 C3 99 1B          JMP    EXLOOP
1BC5                 *
1BC5 76              LPPROG HLT    .      HALT IN RAM
1BC6 C9                RET    .      THEN RETURN AFTER INTERRUPT
1BC7 55              DEFVECT DB    055H   INVERT VECTOR ENABLE BITS
1BC8 C9                RET    .      AND PUT A RETURN IN CASE
1BC9                 *
1BC9                 *  CAL/TEST MODE PROGRAM
1BC9                 *
1BC9 7C              CALCMD MOV    A,H    IF EVEN, SEND THE CAL
1BCA 0F                RRC    .      COMMAND SEQUENCE
1BCB 11 B2 1C          LXI    D,CALSEQ
1BCE D2 2A 1C          JNC    CMDSTRING
1BD1                 *
1BD1 3A 1D 20        SYNCWT LDA    FRAME  WAIT FOR FRAME(L)
1BD4 BD                CMP    L
1BD5 C2 D1 1B          JNZ    SYNCWT
1BD8 C9                RET
1BD9                 *
1BD9                 * MODE CONTROL SECTION
1BD9                 *
1BD9                 VIANG  EQU    128+32
1BD9                 MODEFLAG EQU  1
1BD9                 *
1BD9 3E 98           MODINI MVI    A,VIANG-8  CHANGE ANGLE = 11 DEGREES
1BDB 32 A2 24          STA    CHGANG
```

```
1BDE             0100 *
1BDE 32 A1 24    0101 MODCMD STA    MODTIM  RECORD MODE TIMING
1BE1 CD B0 00    0102        CALL   UNARY  CONVERT VTIME
1BE4 2B          0103        DCX    H
1BE5 22 AB 24    0104        SHLD   VTIME
1BE8 3A A1 24    0105        LDA    MODTIM  THEN ITIME
1BEB 0F          0106        RRC
1BEC 0F          0107        RRC
1BED 0F          0108        RRC
1BEE 0F          0109        RRC
1BEF CD 80 00    0110        CALL   UNARY
1BF2 2B          0111        DCX    H
1BF3 22 AA 24    0112        SHLD   ITIME
1BF6 21 01 00    0113        LXI    H,1    SET FOR SWITCH NEXT TIME
1BF9 22 AC 24    0114        SHLD   MODCTR
1BFC C9          0115        RET
1BFD             0116 *
1BFD 21 A2 24    0117 EXEANG LXI    H,CHGANG  CHECK IF THE ANGLE
1C00 BE          0118        CMP    M      IS CORRECT.
1C01 C0          0119        RNZ    .
1C02 2A AC 24    0120        LHLD   MODCTR  THEN DOWN-COUNT THE MODE
1C05 2B          0121        DCX    H      COUNTER.
1C06 22 AC 24    0122        SHLD   MODCTR
1C09 C9          0123        RET
1C0A             0124 *
1C0A             0125 * DECIDE WHEN TO SWITCH MODES
1C0A             0126 *
1C0A 2A AC 24    0127 DECMODE LHLD  MODCTR  IF COUNT=0, SWITCH
1C0D 7C          0128        MOV    A,H
1C0E B5          0129        ORA    L
1C0F C0          0130        RNZ
1C10             0131 *
1C10 CD D2 13    0132        CALL   ELESTAT  GET THE CURRENT MODE
1C13 E6 01       0133        ANI    MODEFLAG
1C15 2A AB 24    0134        LHLD   VTIME  IF IMODE, TRY VTIME
1C18 11 2E 1C    0135        LXI    D,VCMDS  AND SEND V COMMANDS
1C1B C2 24 1C    0136        JNZ    TRYMODE
1C1E 2A AA 24    0137        LHLD   ITIME  IN VMODE, TRY IMODE
1C21 11 38 1C    0138        LXI    D,ICMDS  AND SEND I COMMANDS
1C24             0139 *
1C24 7C          0140 TRYMODE MOV   A,H    IF MODE TIME=0
1C25 B5          0141        ORA    L      DON'T SWITCH AT ALL
1C26 C8          0142        RZ     .
1C27 22 AC 24    0143        SHLD   MODCTR
1C2A             0144 *
1C2A 3E 01       0145 CMDSTRING MVI A,1
1C2C E7          0146        RST    4
1C2D C9          0147        RET
1C2E             0148 *
1C2E 02 50       0149 VCMDS  DW     5002H  ENABLE V12 FITS
```

```
1C30 00 5C          0150         DW    5C00H  DISABLE SAWTOOTH
1C32 00 68          0151         DW    6800H  GO TO VOLTAGE MODE
1C34 01 62          0152         DW    6201H  PUT OLD V12 RESULT OUT
1C36 FF FF          0153         DW    -1
1C38 03 50          0154 ICMDS   DW    5003H  DISABLE V12 FITS
1C3A 01 68          0155         DW    6801H  GO TO LANGMUIR PROBE MODE
1C3C 03 5C          0156         DW    5C03H  ENABLE SAWTOOTH STEP+BIAS
1C3E FF FF          0157         DW    -1
1C40                0158 *
1C40 01 43          0159 INISEQ  DW    4301H  FORMAT 0 1
1C42 F0 01          0160         DW    01F0H  BIAS 1-4 TO -40NAMPS
1C44 F0 02          0161         DW    02F0H
1C46 F0 03          0162         DW    03F0H
1C48 F0 04          0163         DW    04F0H
1C4A FF 09          0164         DW    09FFH  GUARDS 1-2 TO -1
1C4C FF 0A          0165         DW    0AFFH
1C4E FF 11          0166         DW    11FFH  STUBS 1-2 TO 0
1C50 FF 12          0167         DW    12FFH
1C52 FF 21          0168         DW    21FFH  FILTERS 1-7 TO 255
1C54 FF 22          0169         DW    22FFH
1C56 FF 23          0170         DW    23FFH
1C58 FF 24          0171         DW    24FFH
1C5A FF 25          0172         DW    25FFH
1C5C FF 26          0173         DW    26FFH
1C5E FF 27          0174         DW    27FFH
1C60 FF 19          0175         DW    19FFH  VTRIMS 1-2 TO -1
1C62 FF 1A          0176         DW    1AFFH
1C64 0A 31          0177         DW    310AH  SET 10 (GROUND OUTER BRAID)
1C66 0B 30          0178         DW    300BH  RESET 11 (BIAS REFS12=GND NOT P03)
1C68 0D 30          0179         DW    300DH  RESET 13 (GUARD FILTER OFF)
1C6A 0E 30          0180         DW    300EH  RESET 14 (STUB FILTER OFF)
1C6C 11 30          0181         DW    3011H  RESET 17 (BIAS3,4 REF=V3,V4 NOT GND)
1C6E 14 31          0182         DW    3114H  SET 20 (UNBIAS DC PREAMP 3)
1C70 15 31          0183         DW    3115H  SET 21 (UNBIAS DC PREAMP 4)
1C72 00 30          0184         DW    3000H  RESET 0 (STEERING FOR 20 AND 21)
1C74 01 2A          0185         DW    2A01H  MUX 2=1 (KAGC = V12/R11 )
1C76 01 2B          0186         DW    2B01H  MUX 3=1 (V1/SC = V1 )
1C78 00 48          0187         DW    4800H  RAMBASE 0X2000
1C7A 00 68          0188         DW    6800H  START OUT IN VMODE
1C7C 77 38          0189         DW    3877H  SWITCH EVERY 128 SPINS
1C7E FF C8          0190         DW    0C8FFH  COMMAND COUNTER (RESET/ERROR)
1C80 FF FF          0191         DW    -1     END OF LIST
1C82                0192 *
1C82 00 60          0193 CALSEQ  DW    6000H  DISABLE SWEEPS
1C84 03 61          0194         DW    6103H
1C86 C0 58          0195         DW    058C0H  SAWTOOTH OFFSET = -64
1C88 02 5C          0196         DW    05C02H  DISABLE SAWTOOTH BIASING
1C8A 03 68          0197         DW    6803H  SWITCH TO IMODE WITH TEST MARK
1C8C 01 29          0198         DW    2901H  MUX 1=1 TO GET V1=BIAS1
1C8E 00 80          0199         DW    08000H  BURST FREQUENCY 0
```

```
1C90 11 31          0200       DW     3111H  SET K17
1C92 01 30          0201       DW     3001H  RESET K1
1C94 05 B1          0202       DW     0B105H  BURST FORMAT = 5 (ALL QTYS)
1C96                0203 *
1C96 00 91          0204       DW     9100H  WAIT FOR FRAME(0)
1C98 9C 03          0205       DW     039CH  BIAS 3= -100
1C9A 64 04          0206       DW     0464H  BIAS 4= +100
1C9C 06 31          0207       DW     3106H  SET RELAY(6)
1C9E 20 91          0208       DW     9120H  WAIT 4 SECONDS
1CA0 64 03          0209       DW     0364H  BIAS 3= +100
1CA2 9C 04          0210       DW     049CH  BIAS 4= -100
1CA4 06 31          0211       DW     3106H  SET RELAY(6)
1CA6 40 91          0212       DW     9140H  WAIT 4 SECONDS
1CA8                0213 *
1CA8 03 5C          0214       DW     05C03H  SAWOPT = 60
1CAA 00 B4          0215       DW     0B400H  BURST 60
1CAC 80 91          0216       DW     09180H  WAIT FOR 8 SECONDS
1CAE 00 B5          0217       DW     0B500H  BURST STOP/PLAYBACK
1CB0 00 5C          0218       DW     05C00H  TURN OFF SAW
1CB2 00 68          0219       DW     6800H  VOLTAGE MODE
1CB4 00 60          0220       DW     6000H  REENABLE SWEEPS
1CB6 00 61          0221       DW     6100H
1CB8 01 62          0222       DW     6201H  PUT OUT BIAS1/2 RESULT
1CBA 03 62          0223       DW     6203H  AND  BIAS3/4 RESULTS
1CBC 11 30          0224       DW     3011H  RESET K17 (BIAS3,4 REF=3,4 NOT GND)
1CBE FF FF          0225       DW     -1
1CC0                0226 *
1CC0 4D 41 49 4E    0227       ASC    "MAIN2.1-HARVEY"  VERSION STAMP
     32 2E 31 2D
     48 41 52 56
     45 59
1CCE 00          V  0228       DB     256    END OF EXEC
1CCF                0229 *
1CCF                0230 * RAM AREA FOR THE EXECUTIVE
1CCF                0231 *
1CCF                0232       ORG    EXERAM
24A0                0233 VERSION DS    1
24A1                0234 MODTIM DS     1
24A2                0235 CHGANG DS     1
24A3                0236 OTHER  DS     5
24A8                0237 VTIME  DS     2
24AA                0238 ITIME  DS     2
24AC                0239 MODCTR DS     2
24AE                0240 PROG   DS     2      RAM AREA TO REST
24B0                0241 EXEVECT DS    4      FOREGROUND VECTOR (AA,C3,XX,XX)
24B4                0242 *
24B4                0243       COM    EXWATCH  EXECUTIVE WATCHDOG
24B4                0244 EXWATCH DS    1
24B5                0245 *
24B5                0246 * CONTROLLED MODULES
```

```
24B5           *
24B5                   ORG    LD
1680           LDINIT  DS     3
1683           *
1683                   ORG    SWP
17E4           SWPINIT DS     3
17E7           SWPANG  DS     3
17EA           SWPSTAT DS     3
17ED           SWPEXEC DS     3
17F0           *
17F0                   ORG    FIT
11E8           FITINIT DS     3
11EB           FITSMP  DS     3
11EE           FITTEL  DS     3
11F1           FITEXEC DS     3
11F4           *
11F4           ELESTAT EQU    ELE+12H  MODE STATUS
11F4           FRAME   EQU    BKGRAM+1  FRAME PART OF CLOCK
11F4           *
11F4           * DEFINE EXECUTIVE COMMANDS
11F4           *
11F4                   ORG    38H/4+CMDTAB
004E DE 1B             DW     MODCMD
0050           *
0050                   ORG    90H/4+CMDTAB
0064 C9 1B             DW     CALCMD
```

```
0000            0001 *
0000            0002 * CRRES FLIGHT PROGRAM---BACKGROUND MANAGEMENT
0000            0003 * WRITTEN BY PETER R HARVEY
0000            0004 *
0000            0005 * FILE : BKG.A
0000            0006 *
0000            0007 DIGIT  EQU    0F0H   DIGIT COMMAND CODE
0000            0008 ENTER  EQU    010H   'ENTER' SUB-CODE
0000            0009 CSUM   EQU    0C8H   CHECKSUM SET COMMAND
0000            0010 RESET  EQU    070H   SOFTWARE RESET COMMAND
0000            0011 *
0000            0012 LEN1   EQU    22H    BOOMLEN/TEMPS FROM MUX 22-27H
0000            0013 PSW    EQU    6      8085 SPECIFIC INFORMATION
0000            0014 SP     EQU    6
0000            0015 RES75  EQU    16
0000            0016 MSE    EQU    8
0000            0017 MSK75  EQU    4
0000            0018 MSK65  EQU    2
0000            0019 MSK55  EQU    1
0000            0020 *
0000            0021 WRDBIT EQU    MSK75
0000            0022 FRMBIT EQU    MSK65
0000            0023 CMDBIT EQU    MSK55
0000            0024 *
0000            0025 *   BACKROUND INITIALIZATION
0000            0026 *
0000            0027        ORG    3
0003 C3 E0 1C   0028        JMP    BKGINIT
0006            0029 *
0006            0030 *   BACKGROUND SERVICE FUNCTIONS
0006            0031 *
0006            0032        ORG    4*8
0020 C3 C1 1D   0033        JMP    BKGFNS
0023            0034 *
0023            0035 *   WATCHDOG TIMER INTERRUPT
0023            0036 *
0023            0037        ORG    4*8+4
0024 C3 00 00   0038 TRAP   JMP    0      RESET THE CPU
0027            0039 *
0027            0040 *   COMMAND INTERRUPT (RESTART 5.5)
0027            0041 *
0027            0042        ORG    5*8+4
002C F5         0043        PUSH   PSW    SAVE ACCUM
002D C3 2D 1E   0044        JMP    CMDSERV
0030            0045 *
0030            0046 *  SOFTWARE COMMAND INTERRUPT
0030            0047 *
0030            0048        ORG    6*8
0030 C3 61 1E   0049        JMP    CMD60
```

```
0033                *
0033                *   MAJOR FRAME INTERRUPT
0033                *
0033                        ORG     6*8+4
0034 F5                     PUSH    PSW
0035 C3 32 1D               JMP     MAJINT
0038                *
0038                *   WORD RATE CLOCK INTERRUPT (RESTART 7.5)
0038                *
0038                        ORG     7*8+4
003C F5                     PUSH    PSW     SAVE PROGRAM STATUS WORD
003D C3 42 1D               JMP     WRCINT
0040                *
0040                * BEGIN MODULE
0040                *
0040                        ORG     BKG
1CE0 21 C0 FF       BKGINIT LXI     H,0FFC0H  SET FRAME
1CE3 22 1D 20               SHLD    FRAME  SO 1ST MAJOR IS 7
1CE6 21 FF FF               LXI     H,-1   RESET HOURS AND DAYS
1CE9 22 1F 20               SHLD    HR225
1CEC                *
1CEC                * INITIALIZE THE PACKAGES WHICH OPERATE IN THE BACKGROUND
1CEC                *
1CEC CD C4 16               CALL    DEPINIT  DEPLOYMENT
1CEF CD 80 16               CALL    LDINIT  PROGRAM LOADING
1CF2 CD 50 0D               CALL    MAGINIT  MAGNETOMETER DATA
1CF5 CD 98 0E               CALL    PLAINIT  PLASMA DATA
1CF8 CD C0 13               CALL    ELEINIT  ELECTRIC FIELD/LANGMUIR PROBE DATA
1CFB CD 03 10               CALL    BURINIT  BURST CONTROL
1CFE CD 30 13               CALL    SAWINIT  SAWTOOTH CONTROL
1D01                *
1D01 21 00 00               LXI     H,0    CLEAR THE EXP OUTPUT
1D04 22 3C 20               SHLD    EXPCNT
1D07 CD F8 1D               CALL    STVECT AND THE BKG VECTOR
1D0A                *
1D0A 3E 0D                  MVI     A,MSE+WRDBIT+CMDBIT  DON'T ACCEPT
1D0C CD CF 00               CALL    SETMASK  ANYTHING UNTIL MAJOR FRM
1D0F FB                     EI      .
1D10 76                     HLT     .      LET THE FRAME INT OCCUR
1D11 3E 18                  MVI     A,RES75+MSE ENABLE WORD AND CMDS
1D13 CD CF 00               CALL    SETMASK
1D16                *
1D16                * NOW IN "NEITHERGROUND"
1D16                *
1D16 97                     SUB     A      RESET COMMAND COUNTERS
1D17 32 21 20               STA     GOODCNT
1D1A 32 22 20               STA     BADCNT
1D1D 32 2D 20               STA     STATUS  AND STATUS FLAG
1D20 D3 00                  OUT     0      RESET DIAGNOSTIC LEDS
1D22 D3 01                  OUT     1
```

```
1D24                  *
1D24 21 53 07                 LXI   H,30000/16  FAKE A SUNPERIOD
1D27 CD A3 1D                 CALL  SUNRES  OF 30 SECS TO START
1D2A                  *
1D2A 3E 50            KLYINIT MVI   A,8*10  10 SECS PER PULSE (.1HZ)
1D2C 32 32 20                 STA   KLYCNT
1D2F C3 2B 03                 JMP   SETKLY
1D32                  *
1D32                  * MAJOR FRAME INTERRUPT
1D32                  *
1D32 3A 1D 20         MAJINT LDA    FRAME  MAKE SURE MINOR=31
1D35 F6 1F                    ORI   31
1D37 32 1D 20                 STA   FRAME
1D3A 3E FC                    MVI   A,252  RESET WORD NUMBER
1D3C 32 1C 20                 STA   WORD
1D3F F1                       POP   PSW
1D40 FB                       EI
1D41 C9                       RET
1D42                  *
1D42                  * WORD INTERRUPT
1D42                  *
1D42 E5               WRCINT PUSH   H      SAVE [HL] REGISTER
1D43 D5                       PUSH  D      AND [DE] TOO
1D44 C5                       PUSH  B
1D45 FB                       EI
1D46                  *
1D46 3A 1C 20                 LDA   WORD   INCREMENT WORD BY 2
1D49 C6 02                    ADI   2
1D4B 32 1C 20                 STA   WORD
1D4E B7                       ORA   A      DIV BY 2 SINCE WE COUNT 16 BITS
1D4F 1F                       RAR   .      AT A TIME
1D50 CD 59 1D                 CALL  VECTOR  VECTOR TO TMTABLE(WORD)
1D53                  *
1D53 C1               EXIT    POP   B
1D54 D1                       POP   D      RETURN FROM THE INTERRUPT
1D55 E1                       POP   H
1D56 F1                       POP   PSW
1D57 FB                       EI    .
1D58 C9                       RET   .
1D59                  *
1D59                  * VECTOR TO TMTABLE(WORD)
1D59                  *
1D59 0F               VECTOR RRC    .      IF WORD IS ODD, NULL
1D5A DA 43 1F                 JC    NULL
1D5D 6F                       MOV   L,A    MAKE THE WORD (0-127)
1D5E 26 1F                    MVI   H,TMTABLE/256  INTO A TABLE ADDRESS
1D60 6E                       MOV   L,M    PICK UP ROUTINE ADDRESS
1D61 E9                       PCHL  .
1D62                  *
1D62                  * MINOR FRAME BOUNDARY (BEGINNING OF WORD 0)
```

```
1D62                *
1D62 21 1C 20       MINOR  LXI    H,WORD  INCREMENT WORD
1D65 3E 04                 MVI    A,4     COUNTER.
1D67 CD 80 1D              CALL   INC     (UP TO 4 BYTES)
1D6A CD B5 03              CALL   RWATCH  RESET HARDWARE WATCHDOG CIRCUIT
1D6D 21 B4 24              LXI    H,EXWATCH  UPDATE EXECUTIVE LOOP WATCHDOG
1D70 34                    INR    M       AND IF OVERFLOW (256), RESET CPU
1D71 CA 24 00              JZ     TRAP
1D74 3A 1D 20              LDA    FRAME   NOTIFY THE MODULES
1D77 CD 53 0D              CALL   MAGFRAME
1D7A 3A 1D 20              LDA    FRAME
1D7D C3 C3 13              JMP    ELEFRAME
1D80                *
1D80 23             INC    INX    H       STEP TO NEXT BYTE
1D81 34                    INR    M       INC IT
1D82 C0                    RNZ    .       RETURN IF DONE
1D83 3D                    DCR    A       GO ON
1D84 C2 80 1D              JNZ    INC
1D87 C9                    RET
1D88                *
1D88                * UPDATE SUN ANGLE/ CHECK FOR SUN PULSES
1D88                * [\]
1D88 2A 2E 20       SUNINC LHLD   SUNCTR
1D8B 23                    INX    H
1D8C 22 2E 20              SHLD   SUNCTR
1D8F 2A 30 20              LHLD   SUNDOWN  UPDATE THE SUN ANGLE
1D92 11 00 FF              LXI    D,-256   BY DIVIDING THE
1D95 19                    DAD    D       SUNPERIOD BY 256
1D96 D4 B4 1D              CNC    PHASE   IF NEG. BUMP ANGLE
1D99 22 30 20              SHLD   SUNDOWN
1D9C                *
1D9C CD D9 00              CALL   SUNSTAT  IF NOT SUN
1D9F C8                    RZ     .       PULSE YET, QUIT.
1DA0                *
1DA0 2A 2E 20              LHLD   SUNCTR  SET THE SUN PERIOD
1DA3 22 25 20       SUNRES SHLD   SUNPER
1DA6 2B                    DCX    H       RESET THE DOWN COUNTER
1DA7 22 30 20              SHLD   SUNDOWN
1DAA                *
1DAA 21 00 00              LXI    H,0     RESET THE SUN ANGLE
1DAD 22 23 20              SHLD   ANGLE
1DB0 22 2E 20              SHLD   SUNCTR  AND SUNINC COUNTER
1DB3 C9                    RET    .
1DB4                *
1DB4 EB             PHASE  XCHG   .       PUT REMAINDER IN [DE]
1DB5 2A 23 20              LHLD   ANGLE   ANGLE++
1DB8 23                    INX    H
1DB9 22 23 20              SHLD   ANGLE
1DBC 2A 25 20              LHLD   SUNPER  ADD NEW SUN PERIOD
1DBF 19                    DAD    D       TO THE REMAINDER FROM
```

```
1DC0 C9                       RET     .       LAST PERIOD.
1DC1                   *
1DC1                   * SERVICE ROUTINES
1DC1                   *
1DC1 FE 01             BKGFNS CPI     1       DETERMINE FUNCTION
1DC3 CA 15 1E                 JZ      BATCH
1DC6 FE 02                    CPI     2
1DC8 CA 0A 1E                 JZ      CCST
1DCB FE 03                    CPI     3
1DCD CA F8 1D                 JZ      STVECT
1DD0 FE 04                    CPI     4
1DD2 C0                       RNZ
1DD3                   *
1DD3                   * START EXPERIMENTAL OUTPUT
1DD3                   *
1DD3 E5                STEXP  PUSH    H
1DD4 CD E1 1D                 CALL    CHKEXP   CHECK IF GOING/STOPPED
1DD7 E1                       POP     H
1DD8 C0                       RNZ     .       IF ALREADY GOING, RETURN
1DD9 22 3A 20                 SHLD    EXPADR  SET DUMP ADDRESS
1DDC EB                       XCHG    .
1DDD 22 3C 20                 SHLD    EXPCNT  AND SIZE
1DE0 C9                       RET
1DE1                   *
1DE1 2A 3C 20          CHKEXP LHLD    EXPCNT
1DE4 7C                       MOV     A,H
1DE5 B5                       ORA     L
1DE6 C9                       RET
1DE7                   *
1DE7 CD E1 1D          EXPTEL CALL    CHKEXP  IF EXPCNT=0, RETURN(0)
1DEA C8                       RZ      .
1DEB 2B                       DCX     H       ELSE CNT--
1DEC 22 3C 20                 SHLD    EXPCNT
1DEF 2A 3A 20                 LHLD    EXPADR  AND RETURN MEM(ADR++)
1DF2 7E                       MOV     A,M
1DF3 23                       INX     H
1DF4 22 3A 20                 SHLD    EXPADR
1DF7 C9                       RET
1DF8                   *
1DF8                   * SET BKG VECTOR
1DF8                   *
1DF8 7C                STVECT MOV     A,H     IF ZERO, CLEAR BKGVECT
1DF9 B5                       ORA     L
1DFA 3E AA                    MVI     A,0AAH
1DFC C2 03 1E                 JNZ     STV1
1DFF 97                       SUB     A
1E00 21 4D 1F                 LXI     H,XTRA
1E03 22 36 20          STV1   SHLD    BKGVECT
1E06 32 38 20                 STA     BVCHK
1E09 C9                       RET
```

```
1E0A             *
1E0A             * COMMAND COUNT STATUS.  RETURNS 1 1F MORE TO GO.
1E0A             *
1E0A 21 21 20    CCST    LX1   H,GOODCNT  1F COUNT MATCHES
1E0D 3A 35 20            LDA   CMDCNT  THE #GOOD COMMANDS,
1E10 96                  SUB   M
1E11 C8                  RZ    .       RETURN(0) ELSE (1)
1E12 3E 01               MV1   A,1
1E14 C9                  RET
1E15             *
1E15             * COMMAND BATCH PROCESSOR
1E15             * ON ENTRY: [DE]-> COMMAND L1ST END1NG 1N -1
1E15             *
1E15 1A          BATCH   LDAX  D       HL=CMD
1E16 6F                  MOV   L,A
1E17 13                  1NX   D
1E18 1A                  LDAX  D
1E19 67                  MOV   H,A
1E1A 13                  1NX   D
1E1B 3C                  1NR   A       1F CMD= -1, QUIT
1E1C C8                  RZ
1E1D 3A 2D 20    BATWT   LDA   STATUS  1F COMMAND COM1NG 1N
1E20 FE AA               CPI   0AAH  WA1T T1LL DONE
1E22 CA 1D 1E            JZ    BATWT
1E25 D5                  PUSH  D
1E26 CD 61 1E            CALL  CMDGO
1E29 D1                  POP   D
1E2A C3 15 1E            JMP   BATCH
1E2D             *
1E2D             * COMMAND 1NTERRUPT SERV1CE ROUT1NE
1E2D             * [\]
1E2D 3E AA       CMDSERV MV1   A,0AAH  SET A SERV1CE FLAG
1E2F 32 2D 20            STA   STATUS
1E32 3E 09               MV1   A,MSE+CMDB1T  MASK OTHER CMDS
1E34 CD CF 00            CALL  SETMASK  UNTIL TH1S ONE
1E37 F1                  POP   PSW    1S SERV1CED.
1E38 FB                  EI    .
1E39 C9                  RET   .
1E3A             *
1E3A             * CHECK FOR COMMAND 1NPUTS.
1E3A             *
1E3A 3A 2D 20    CMDEXEC LDA   STATUS  1F COMMAND STATUS
1E3D FE AA               CP1   0AAH   1S ZERO, RETURN
1E3F C0                  RNZ   .
1E40 CD CD 00            CALL  GETMASK  1F A COMMAND
1E43 E6 10               AN1   CMDB1T*16  SHIFTING 1N, QU1T
1E45 C0                  RNZ   .       AND GET 1T NEXT TIME.
1E46             *
1E46 97                  SUB   A       RESET THE STATUS
1E47 32 2D 20            STA   STATUS
```

```
1E4A                *
1E4A CD D1 00              CALL   CMDIN  READ THE COMMAND REG
1E4D CD 93 00              CALL   MARK
1E50 3E 08                 MVI    A,MSE  RE-ENABLE CMDS
1E52 CD CF 00              CALL   SETMASK
1E55 CD 61 1E              CALL   CMDGO  EXECUTE IT
1E58 21 21 20              LXI    H,GOODCNT  COUNT GOOD OR BAD
1E5B D2 5F 1E              JNC    INRCNT
1E5E 23                    INX    H
1E5F 34             INRCNT INR    M
1E60 C9                    RET
1E61                *
1E61                * COMMAND DISTRIBUTION
1E61                *
1E61 EB             CMDGO  XCHG   .      PUT COMMAND IN [DE]
1E62 7A                    MOV    A,D    MASK UPPER 5 BITS
1E63 E6 F8                 ANI    0F8H
1E65 0F                    RRC    .
1E66 0F                    RRC    .
1E67 21 40 00              LXI    H,CMDTAB  REFER TO TABLE
1E6A DF                    RST    REF/8
1E6B                *
1E6B 7E                    MOV    A,M    PICK UP ADDRESS
1E6C 23                    INX    H
1E6D 66                    MOV    H,M
1E6E 6F                    MOV    L,A
1E6F                *
1E6F B4                    ORA    H      IF ADDRESS=0, RETURN CARRY
1E70 37                    STC
1E71 C8                    RZ     .
1E72                *
1E72 E5                    PUSH   H      PUT ON STACK
1E73 EB                    XCHG   .      [HL]=COMMAND
1E74 7D                    MOV    A,L    [A]=DATA PART
1E75 B7                    ORA    A      CLEAR CARRY
1E76 C9                    RET    .
1E77                *
1E77                * BACKGROUND MODULE COMMANDS
1E77                *
1E77 32 35 20       CSCMD  STA    CMDCNT   SET CMDCNT COMPARE REG=DATA
1E7A 21 00 00              LXI    H,0    CLEAR CMD COUNTERS
1E7D 22 21 20              SHLD   GOODCNT
1E80 C9                    RET    .
1E81                *
1E81 FE 10          DIGCMD CPI    ENTER  IF THE ENTER COMMAND
1E83 CA 95 1E              JZ     ENTDIG
1E86                *
1E86 2A 33 20              LHLD   DIGREG
1E89 29                    DAD    H      DIGREG=DIGREG*16 + A
1E8A 29                    DAD    H
```

```
1E8B 29               DAD   H
1E8C 29               DAD   H
1E8D E6 0F            ANI   15
1E8F B5               ORA   L
1E90 6F               MOV   L,A
1E91 22 33 20         SHLD  DIGREG
1E94 C9               RET   .      RETURN(NO CARRY)
1E95             *
1E95 2A 33 20    ENTDIG LHLD DIGREG HL=DIGIT REGISTER
1E98 97               SUB   A      DIGIT REGISTER=0
1E99 32 33 20         STA   DIGREG
1E9C 32 34 20         STA   DIGREG+1
1E9F C3 61 1E         JMP   CMDGO  EXECUTE COMMAND [HL]
1EA2             *
1EA2             * AUTONOMICS: KELLY AND TEMPERATURE UPDATES
1EA2             *
1EA2 21 32 20    AUTO LXI   H,KLYCNT
1EA5 35               DCR   M
1EA6 CC 2A 1D         CZ    KLYINIT
1EA9 CD C7 16         CALL  DEPSAMP
1EAC             *
1EAC 3A 1D 20         LDA   FRAME  EVERY MAJOR FRAME
1EAF E6 1F            ANI   31     SAMPLE 1 ANALOG SUBCOM VAL
1EB1 C0               RNZ
1EB2             *
1EB2             * SAMPLE BOOM LENGTHS AND 1 OF 4 TEMPERATURES
1EB2             *
1EB2 3A 1D 20    TEMPSAMP LDA FRAME COMPUTE WHICH ONE
1EB5 07               RLC
1EB6 07               RLC
1EB7 07               RLC
1EB8 E6 07            ANI   7
1EBA FE 06            CPI   6      IF 6 OR 7, QUIT
1EBC D0               RNC
1EBD F5               PUSH  PSW
1EBE C6 22            ADI   LEN1   ADD TO LEN1 MUX ADDRESS
1EC0 CD E6 00         CALL  SAMPLE
1EC3 29               DAD   H      CONVERT TO 8 BITS
1EC4 29               DAD   H
1EC5 29               DAD   H
1EC6 29               DAD   H
1EC7 5C               MOV   E,H
1EC8 F1               POP   PSW    GET THE 0-3 AGAIN
1EC9 21 27 20         LXI   H,SLEN1
1ECC DF               RST   REF/8
1ECD 73               MOV   M,E
1ECE C9               RET
1ECF             *
1ECF             * DIGITAL SUBCOM TABLE
1ECF             *
```

```
1ECF 32              0400 DSCTAB DB     48+2   QTYS IN PKG
1ED0 CC 13           0401        DW     ELEDSC  PKG ADDRESS
1ED2 0B              0402        DB     B
1ED3 67 1B           0403        DW     EXEDSC
1ED5 15              0404        DB     2+15+4
1ED6 74 02           0405        DW     IDDSC
1ED8 0C              0406        DB     12
1ED9 F0 17           0407        DW     SWPDSC
1EDB 04              0408        DB     4
1EDC 36 13           0409        DW     SAWDSC
1EDE 04              0410        DB     4
1EDF 11 10           0411        DW     BURDSC
1EE1 05              0412        DB     5
1EE2 CA 16           0413        DW     DEPDSC
1EE4 01              0414        DB     1
1EE5 9E 0E           0415        DW     PLADSC
1EE7 64              0416        DB     100
1EE8 F3 1F           0417        DW     BKGDSC
1EEA                 0418 *
1EEA                 0419 * SOFTWARE RESET
1EEA                 0420 *
1EEA 7D              0421 SOFTRESET MOV A,L    CHECK FOR COMMAND 7007
1EEB FE 07           0422        CPI    07H
1EED CA 00 00        0423        JZ     0
1EF0 37              0424        STC
1EF1 C9              0425        RET
1EF2 00            V 0426        DB     256    PART 1 ENDS
1EF3                 0427 *
1EF3                 0428 * TELEMETRY WORD FUNCTION TABLE
1EF3                 0429 *
1EF3                 0430        ORG    $/256+1*256  NEXT PAGE BOUNDARY
1F00                 0431 T      EQU    $
1F00 40              0432 TMTABLE DB    MNFR-T  0
1F01 4E              0433        DB     ET-T
1F02 5E              0434        DB     SPLO-T
1F03 56              0435        DB     MT-T
1F04 B6              0436        DB     ES-T
1F05 B6              0437        DB     ES-T
1F06 BF              0438        DB     SUN-T
1F07 AA              0439        DB     CMD-T
1F08                 0440 * 1
1F08 BC              0441        DB     PS-T
1F09 4E              0442        DB     ET-T
1F0A 7B              0443        DB     SPLIT-T
1F0B 56              0444        DB     MT-T
1F0C B6              0445        DB     ES-T
1F0D B6              0446        DB     ES-T
1F0E BF              0447        DB     SUN-T
1F0F AA              0448        DB     CMD-T
1F10                 0449 * 2
```

```
1F10 B9          0450      DB    MG-T
1F11 B0          0451      DB    ETMS-T
1F12 7B          0452      DB    SPLIT-T
1F13 B6          0453      DB    MTME-T
1F14 B6          0454      DB    ES-T
1F15 B6          0455      DB    ES-T
1F16 BF          0456      DB    SUN-T
1F17 AA          0457      DB    CMD-T
1F18             0458 # 3
1F18 BC          0459      DB    PS-T
1F19 4E          0460      DB    ET-T
1F1A 7B          0461      DB    SPLIT-T
1F1B 56          0462      DB    MT-T
1F1C B6          0463      DB    ES-T
1F1D B6          0464      DB    ES-T
1F1E BF          0465      DB    SUN-T
1F1F AA          0466      DB    CMD-T
1F20             0467 # 4
1F20 BC          0468      DB    KLYT-T
1F21 4E          0469      DB    ET-T
1F22 4D          0470      DB    XTRA-T
1F23 4D          0471      DB    XTRA-T
1F24 B6          0472      DB    ES-T
1F25 B6          0473      DB    ES-T
1F26 BF          0474      DB    SUN-T
1F27 AA          0475      DB    CMD-T
1F28             0476 # 5
1F28 BC          0477      DB    PS-T
1F29 4E          0478      DB    ET-T
1F2A 4D          0479      DB    XTRA-T
1F2B 4D          0480      DB    XTRA-T
1F2C B6          0481      DB    ES-T
1F2D B6          0482      DB    ES-T
1F2E BF          0483      DB    SUN-T
1F2F AA          0484      DB    CMD-T
1F30             0485 # 6
1F30 B9          0486      DB    MG-T
1F31 B0          0487      DB    ETMS-T
1F32 4D          0488      DB    XTRA-T
1F33 B9          0489      DB    ME-T
1F34 B6          0490      DB    ES-T
1F35 B6          0491      DB    ES-T
1F36 BF          0492      DB    SUN-T
1F37 AA          0493      DB    CMD-T
1F38             0494 # 7
1F38 BC          0495      DB    PS-T
1F39 4E          0496      DB    ET-T
1F3A 4D          0497      DB    XTRA-T
1F3B 4D          0498      DB    XTRA-T
1F3C B6          0499      DB    ES-T
```

```
1F3D 86                  DB     ES-T
1F3E 8F                  DB     SUN-T
1F3F AA                  DB     CMD-T
1F40              *
1F40              * ROUTINES FOR EACH WORD
1F40              *
1F40 C3 62 1D     MNFR   JMP    MINOR
1F43              *
1F43 3A 38 20     NULL   LDA    BVCHK  CHECK IF ENABLED
1F46 FE AA               CPI    0AAH
1F48 C0                  RNZ    .
1F49 2A 36 20            LHLD   BKGVECT
1F4C E9                  PCHL
1F4D              *
1F4D C9           XTRA   RET    .
1F4E              *
1F4E 3E 01        ET     MVI    A,1    GET 2 BYTES
1F50 CD C9 13            CALL   ELETELEM
1F53 C3 D5 00            JMP    TMOUT  AND OUTPUT 'EM
1F56              *
1F56 3E 01        MT     MVI    A,1    GET 2 MAG BYTES
1F58 CD 5F 0D            CALL   MAGTELEM
1F5B C3 D5 00            JMP    TMOUT
1F5E              *
1F5E              * IN FIRST SPLIT, GET 1 BYTE FROM ELE.
1F5E              * CHECK WHETHER MAG OR DSC GIVES THE OTHER.
1F5E              *
1F5E 97           SPLO   SUB    A      GET A BYTE OF ELE
1F5F CD C9 13            CALL   ELETELEM
1F62 E5                  PUSH   H
1F63 3A 1D 20            LDA    FRAME  IF ODD MINOR
1F66 0F                  RRC    .      THEN GET AN EXP BYTE
1F67 DA 71 1F            JC     SPLEXP
1F6A CD 8F 1F            CALL   DSC    GET A DSC BYTE IN L
1F6D 6F                  MOV    L,A
1F6E C3 81 1F            JMP    JOIN   AND JOIN BYTES
1F71 CD E7 1D     SPLEXP CALL   EXPTEL  GET EXP BYTE IN L
1F74 6F                  MOV    L,A
1F75 C3 81 1F            JMP    JOIN
1F78              *
1F78 97           SPLIT  SUB    A      GET 1 BYTE OF E-FIELD
1F79 CD C9 13            CALL   ELETELEM  IN L
1F7C E5                  PUSH   H
1F7D 97           SPLMAG SUB    A      GET 1 FROM MAG
1F7E CD 5F 0D            CALL   MAGTELEM
1F81              *
1F81 D1           JOIN   POP    D      PUT TOGETHER
1F82 63                  MOV    H,E
1F83 C3 D5 00            JMP    TMOUT  AND OUTPUT 'EM
1F86              *
```

```
1F86 C3 C6 13     ES    JMP   ELESAMP
1F89 C3 56 0D     MG    JMP   MAGGAIN
1F8C C3 9B 0E     PS    JMP   PLASAMP
1F8F CD 88 1D     SUN   CALL  SUNINC
1F92 3A 23 20           LDA   ANGLE  IF THE ANGLE CHANGED,
1F95 21 39 20           LXI   H,OLDANG  TAKE THE VECTOR
1F98 BE                 CMP   M
1F99 77                 MOV   M,A
1F9A C8                 RZ    .
1F9B CD EB 11           CALL  FITSMP  IF ANGLE CHANGED, FIT SAMPLE
1F9E 3A 23 20           LDA   ANGLE
1FA1 CD 64 1B           CALL  EXEANG  ELSE TELL EXECUTIVE
1FA4 3A 23 20           LDA   ANGLE   AND THE SWEEP
1FA7 C3 E7 17           JMP   SWPANG
1FAA              *
1FAA CD 33 13     CMD   CALL  SAWSTEP
1FAD C3 3A 1E           JMP   CMDEXEC
1FB0 CD 4E 1F     ETMS  CALL  ET
1FB3 C3 59 0D           JMP   MAGSAMP
1FB6 CD 56 1F     MTME  CALL  MT
1FB9 C3 5C 0D     ME    JMP   MAGENCD
1FBC C3 A2 1E     KLYT  JMP   AUTO
1FBF              *
1FBF              * OUTPUT THE DIGITAL SUBCOMMUTATOR VALUES
1FBF              *
1FBF 21 1D 20     DSC   LXI   H,FRAME  DECIDE WHETHER TO
1FC2 7E                 MOV   A,M    OUTPUT THE FRAME COUNTER.
1FC3 E6 1F              ANI   31     THIS IS DONE ON NEW MAJORS
1FC5 C2 D3 1F           JNZ   DSCI
1FC8 66                 MOV   H,M    PUT THE MAJOR FRAME NUMBER
1FC9 3A 1E 20           LDA   CYCLE  TOGETHER WITH HIGHER
1FCC 29                 DAD   H      TIME BITS.  3 FROM THE FRAME
1FCD 17                 RAL   .      AND 5 FROM THE NEXT BYTE
1FCE 29                 DAD   H
1FCF 17                 RAL   .
1FD0 29                 DAD   H
1FD1 17                 RAL   .
1FD2 C9                 RET
1FD3              *
1FD3 7E           DSCI  MOV   A,M    COMPUTE THE DSC INDEX
1FD4 E6 E0              ANI   0E0H   BY REMOVING THE TIMES
1FD6 0F                 RRC   .      WE OUTPUT THE FRAME COUNT
1FD7 0F                 RRC   .      PUT -(MAJOR-1) INTO LOW
1FD8 0F                 RRC   .      BITS AND ADD FRAME COUNT
1FD9 0F                 RRC   .
1FDA 2F                 CMA
1FDB 86                 ADD   M
1FDC              *
1FDC 0F                 RRC   .      CONVERT TO VALUES 0 TO 119
1FDD E6 7F              ANI   127
```

```
1FDF 21 CF 1E            LXI   H,OSCTAB  CHECK TABLE FOR WHERE TO GET
1FE2 BE          DSCF    CMP   M         THE OSC BYTE. IF < M, GET
1FE3 DA ED 1F            JC    DSCGO     IT FROM THE ROUTINE
1FE6 96                  SUB   M         ELSE DECREMENT THAT PART OF
1FE7 23                  INX   H         THE OSC INDEX. (EACH ROUTINE EXPECTS
1FE8 23                  INX   H         TO SEE 0-N IN ACCUM).
1FE9 23                  INX   H
1FEA C3 E2 1F            JMP   DSCF
1FED 23          DSCGO   INX   H
1FEE 5E                  MOV   E,M
1FEF 23                  INX   H
1FF0 56                  MOV   D,M
1FF1 EB                  XCHG  .
1FF2 E9                  PCHL  .
1FF3             *
1FF3             * BKG MODULE DIGITAL SUBCOM
1FF3             *
1FF3 21 1E 20    BKGDSC  LXI   H.CYCLE
1FF6 DF                  RST   REF/B
1FF7 7E                  MOV   A,M
1FF8 C9                  RET   .
1FF9 00         V        DB    257 BACKGROUND-END
1FFA             *
1FFA             * VARIABLES
1FFA             *
1FFA                     ORG   BKGRAM
201C             WORD    DS    1         WORD COUNTER
201D             FRAME   DS    1         MINOR/MAJOR FRAME COUNTER
201E             CYCLE   DS    1         SUB-COM CYCLE COUNTER
201F             HR225   DS    1         2.25 HOUR COUNTER
2020             DAY24   DS    1         24 DAY COUNTER
2021             *
2021             GOODCNT DS    1         GOOD COMMAND COUNTER
2022             BADCNT  DS    1         BAD COMMAND COUNTER
2023             *
2023             ANGLE   DS    2         SUN ANGLE
2025             SUNPER  DS    2         SUN PERIOD
2027             *
2027             SLEN1   DS    1         LENGTH MEASUREMENTS
2028             SLEN2   DS    1
2029             STMP1   DS    1         TEMPERATURE MEASUREMENTS
202A             STMP2   DS    1
202B             STMP3   DS    1
202C             STMP4   DS    1
202D             *
202D             STATUS  DS    1         COMMAND READY STATUS BYTE
202E             SUNCTR  DS    2         COUNT OF SUNINC CALLS
2030             SUNDOWN DS    2         DOWN COUNTER
2032             KLYCNT  DS    1         KELLEY GAIN TIMER
2033             DIGREG  DS    2         DIGIT COMMAND REGISTER
```

```
2035   CMDCNT DS     1      COMMAND COUNT COMPARE REG
2036   BKGVECT DS    2      BACKGROUND VECTOR
2038   BVCHK  DS     1      CHECK BYTE FOR VECTOR
2039   OLDANG DS     1      OLD SUN ANGLE
203A   EXPADR DS     2      EXP OUTPUT MEM ADDRESS
203C   EXPCNT DS     2      EXP OUTPUT COUNTER
203E   *
203E   * EXTERNAL REFERENCES
203E   *
203E          ORG    ELE
13C0   ELEINI DS     3      INITIALIZATION
13C3   ELEFRAME DS   3      MINOR FRAME
13C6   ELESAMP DS    3      SAMPLE TIME
13C9   ELETELEM DS   3      TELEMETRY TIME
13CC   ELEDSC DS     3      DIGITAL SUBCOM
13CF   *
13CF          ORG    MAG
0D50   MAGINIT DS    3
0D53   MAGFRAME DS   3
0D56   MAGGAIN DS    3
0D59   MAGSAMP DS    3
0D5C   MAGENCD DS    3
0D5F   MAGTELEM DS   3
0D62   *
0D62          ORG    PLA
0E98   PLAINIT DS    3
0E9B   PLASAMP DS    3
0E9E   PLADSC DS     3
0EA1   *
0EA1          ORG    DEP
16C4   DEPINT DS     3
16C7   DEPSAMP DS    3
16CA   DEPDSC DS     3
16CD   *
16CD          ORG    LO
16B0   LOINIT DS     3
16B3   *
16B3          ORG    FIT
11E8   FITINIT DS    3
11EB   FITSMP DS     3
11EE   FITTEL DS     3
11F1   *
11F1          ORG    BUR
1008   BURINIT DS    3
100B   BURSAMP DS    3
100E   BURTELEM DS   3
1011   BURDSC DS     3
1014   *
1014          ORG    SAW
1330   SAWINIT DS    3
```

```
1333            SAWSTEP DS    3
1336            SAWDSC DS     3
1339            *
1339                    ORG   SWP
17E4            SWPINIT DS    3
17E7            SWPANG DS     3
17EA            SWPSTAT DS    3
17ED            SWPEXEC DS    3
17F0            SWPDSC DS     3
17F3            *
17F3                    ORG   EXEC
1B64            EXEANG DS     3
1B67            EXEDSC DS     3
1B6A            *
1B6A            * DEFINE COMMANDS
1B6A            *
1B6A                    ORG   DIGIT/4+CMDTAB
007C 81 1E              DW    DIGCMD
007E            *
007E                    ORG   CSUM/4+CMDTAB
0072 77 1E              DW    CSCMD
0074            *
0074                    ORG   RESET/4+CMDTAB
005C EA 1E              DW    SOFTRESET
```

```
8000: C3 6D 1B C3 E0 1C 00 00 36 00 23 0D C2 08 00 C8
8010: 1A 77 13 23 0D C3 0F 00 85 6F D0 24 B7 C9 00 00
8020: C3 C1 1D 00 C3 00 00 00 00 00 00 00 F5 C3 2D 1E
8030: C3 61 1E 00 F5 C3 32 1D C9 00 00 00 F5 C3 42 1D
8040: 63 01 80 01 76 01 98 02 AE 02 DD 13 E7 13 DE 1B
8050: 2E 14 ED 13 13 12 46 13 1B 18 F7 13 EA 1E 00 00
8060: 00 00 00 00 C9 1B 00 00 2A 10 24 10 44 10 44 10
8070: 00 00 77 1E DC 16 B5 0E 00 00 87 16 81 1E 7D 0D
8080: 21 01 00 E6 0F C8 29 3D C3 85 00 7C 2F 67 7D 2F
8090: 6F 23 C9 7C D3 01 7D D3 00 C9 3E 20 32 00 20 CD
80A0: B5 03 3E 40 30 00 00 00 97 CD 18 03 3E 01 CD 18
80B0: 03 3E 3F D3 F3 32 14 20 3E 01 32 01 50 CD FA 02
80C0: 97 32 15 20 32 16 20 32 17 20 C3 08 02 20 C9 30
80D0: C9 2A FF AF C9 22 FF AF C9 DB 80 E6 80 21 02 20
80E0: BE 77 C9 DB 90 C9 6F 3E 0D 30 7D F6 80 D3 E0 3E
80F0: 19 3D C2 F1 00 7D D3 E0 3E 90 D3 F3 3E FF D3 50
8100: 76 3E 10 D3 F3 3E 08 30 7D 2A 00 50 FE 07 CA 16
8110: 01 FE 0D C2 20 01 7D E6 F0 6F DB 80 E6 0F B5 6F
8120: 7C E6 0F 67 3E F9 D3 E0 FB C9 57 FE 07 CA 5E 01
8130: FE 0D CA 5E 01 FE 10 DA 4C 01 FE 2E DA 5B 01 11
8140: 30 0E CA 48 01 11 40 0F 7B C3 4F 01 F6 10 5F CD
8150: E6 00 7C FE 07 C8 FE 08 C8 53 C9 E6 EF 57 CD E6
8160: 00 97 C9 1E 05 3E 03 CD AF 01 FE 02 2E 0E DA 87
8170: 01 2E 07 C3 87 01 1E 08 CD AD 01 2E 0B C3 87 01
8180: 1E 09 CD AD 01 2E 0D F5 CD 3D 02 E6 F0 B5 6F 7C
8190: EE 7F 67 F1 CD 81 02 2E 14 0F 7D D2 A0 01 F6 20
81A0: F3 D3 F3 E6 EF D3 F3 F6 10 D3 F3 FB C9 3E 01 25
81B0: A4 F5 83 5F 16 20 7D 12 F1 65 C9 F5 CD 4E 02 F1
81C0: 17 FE 04 DA 02 02 FE 0E DA FD 01 FE 24 DA EF 01
81D0: 2E 01 FE 28 DA D8 01 2D F5 1F 7D CD BB 01 F1 D6
81E0: 20 1F 37 F5 CD BB 01 CD A7 03 F1 B7 C3 BB 01 F5
81F0: 1F 3E 01 CD BB 01 F1 CD 02 02 C3 08 02 0F 07 D2
8200: 08 02 CD 19 02 C3 08 02 CD 3D 02 67 F6 0F 6F CD
8210: 81 02 3E 14 D3 F3 C3 A7 03 B7 1F D6 02 DA 29 02
8220: E6 0F C6 18 E6 27 C3 2C 02 3A 01 20 E6 3F 32 01
8230: 20 5F 3A 15 20 E6 03 0F 0F EE C0 B3 C9 CD 45 02
8240: 3E 10 C3 2C 02 3A 15 20 E6 83 32 15 20 C9 F5 FE
8250: 02 D4 45 02 F1 11 15 20 F5 FE 08 DA 64 02 13 D6
8260: 08 C3 59 02 CD 80 00 EB F1 7B DA 71 02 2F A6 77
8270: C9 B6 77 C9 21 03 20 85 6F 7E C9 C5 0E 08 C3 84
8280: 02 C5 0E 10 F5 29 3E 08 17 F3 D3 F3 F6 02 D3 F3
8290: FB 0D C2 85 02 F1 C1 C9 1E 03 CD AD 01 F5 EB 7A
82A0: EE 7F 57 2E 01 CD 00 03 F1 2E 18 C3 99 01 1E 0D
82B0: 3E 07 CD AF 01 54 3C E6 07 CD 80 00 CD 00 03 3E
82C0: 18 CD A0 01 3E 38 CD A0 01 C3 FA 02 C5 5D 21 F2
82D0: 02 E6 07 4F 06 00 09 3A 14 20 CD C1 03 32 14 20
82E0: C1 C3 FA 02 E6 07 5F 16 00 21 F2 02 19 3A 14 20
82F0: A6 C9 20 01 06 08 21 27 2F 10 21 00 00 11 00 00
8300: E5 3A 14 20 67 CD 7B 02 E1 EB CD 7B 02 7B 2F 67
```

```
8310: CD 7B 02 3E 18 D3 F3 C9 8F CD 80 00 CD 49 03 3E
8320: 0A CD A7 03 3D C2 21 03 C3 49 03 2E 80 C3 46 03
8330: E5 2E 40 CD 49 03 E1 7C 2F D3 D0 7D 2F D3 90 2E
8340: 40 C3 49 03 2E 20 CD 49 03 3A 00 20 AD D3 C0 32
8350: 00 20 C9 20 07 DC 7E 03 E5 C5 01 10 00 3E C0 30
8360: 05 37 CA 77 03 20 07 D2 60 03 00 3E 4D 30 3E 80
8370: 29 1F 30 0D C2 6E 03 3E 46 30 F8 C1 E1 C9 E5 CD
8380: 84 03 E1 C9 20 E6 80 C8 C5 EB 01 10 00 00 3E CD
8390: 30 20 04 A8 FA 91 03 0A 20 07 18 0D C2 97 03 EB
83A0: 3E 48 30 B7 C1 F8 C9 F5 D5 11 FE 01 18 7B B2 C2
83B0: AC 03 D1 F1 C9 3A 00 20 F6 10 D3 C0 EE 10 D3 C0
83C0: C9 4F 7E 2F A1 4F EB 1A 0F DA D0 03 29 C3 C8 03
83D0: 1A A5 B1 C9 4E 23 56 23 5E C9 71 23 72 23 73 C9
83E0: 7A B7 C8 46 23 7E B7 CA 70 05 23 6E 67 78 A9 F2
83F0: FD 03 CD B1 05 CD FD 03 79 EE 80 4F C9 78 81 D6
8400: 40 FA 76 05 4F CD F0 05 C3 5C 05 7A B7 C8 46 23
8410: 7E B7 CA 7B 05 23 6E 67 78 A9 F2 26 04 CD 81 05
8420: CD 26 04 C3 F8 03 79 90 C6 40 FA 76 05 4F C5 7C
8430: 2F 47 7D 2F 4F 03 62 6B 09 DA 56 04 EB 3E 10 CD
8440: 6F 04 29 DA 4C 04 09 DA 4C 04 C1 C9 C1 1C C0 14
8450: C0 11 00 80 0C C9 3E 10 11 FF FF CD 6F 04 C1 0C
8460: 37 7A 1F 57 7B 1F 5F D0 C3 4D 04 33 33 3D C8 29
8470: DA 81 04 EB 29 E8 1C E5 09 DA 6B 04 E1 1D C3 6D
8480: 04 EB 29 EB 09 1C C3 6D 04 C5 D5 CD 98 04 7A B7
8490: CA 95 04 79 07 D1 C1 C9 7E EE 80 47 C3 A0 04 46
84A0: 23 7E 23 6E 67 97 BC C8 DA CA 0C 05 79 90 87 F2
84B0: B8 04 78 41 4F EB 90 87 CA C2 04 0F FE 10 D0 CD
84C0: F4 04 78 A9 FA D2 04 19 EB D0 7A 1F 57 7B 1F 5F
84D0: 0C C9 7B 95 7A 9C DA E7 04 57 7B 95 5F 21 00 00
84E0: B2 C2 21 05 0E 00 C9 7D 93 5F 7C 9A 57 48 21 00
84F0: 00 C3 21 05 D6 08 DA FF 04 6C 26 00 C8 D6 08 C5
8500: 47 97 29 8F 04 C2 02 05 C1 6C 67 C9 EB 48 C9 7A
8510: B7 0E 60 F2 21 05 CD 69 06 CD 21 05 79 F6 80 4F
8520: C9 79 B7 F2 2F 05 E6 7F 4F CD 2F 05 C3 F8 03 7A
8530: B7 C2 59 05 B3 C2 4E 05 B4 C2 48 05 B5 C2 42 05
8540: 4A C9 55 06 18 C3 53 05 EB 06 10 C3 53 05 53 5C
8550: 65 06 08 79 90 4F DA 70 05 7A 6C 63 B7 FA 66 05
8560: 0D 29 8F F2 60 05 57 5C 7D 07 DC 4D 04 79 87 F0
8570: 0E 00 11 00 00 C9 FE C0 D2 70 05 0E 7F 11 FF FF
8580: C9 78 E6 7F 47 79 E6 7F 4F C9 79 EE 80 FA 97 05
8590: 4F CD 97 05 C3 69 06 E6 7F FE 41 DA C5 05 FE 60
85A0: D2 CC 05 21 00 00 D6 50 CB EB D2 BC 05 C6 10 CD
85B0: BC 05 EB 11 00 00 C9 29 2C E8 3D C8 29 EB DA B7
85C0: 05 29 C3 B9 05 11 00 00 21 00 00 C9 11 FF 7F 21
85D0: FF FF C9 7A B7 C8 41 62 6B C3 ED 03 7A B7 C8 79
85E0: E6 01 C4 CA 04 C5 CD 44 06 C1 79 0F C6 20 4F C9
85F0: 97 B8 CA 08 06 B5 CA 09 06 E5 CD 0A 06 6C 67 E3
8600: 7C CD 0A 06 D1 19 88 C9 EB 7C 21 00 00 44 87 D2
8610: 14 06 19 88 29 8F D2 1B 06 19 88 29 8F D2 22 06
8620: 19 88 29 8F D2 29 06 19 88 29 8F D2 30 06 19 88
```

```
8630: 29 8F D2 37 06 19 88 29 8F D2 3E 06 19 88 29 8F
8640: D0 19 88 C9 01 00 80 CD 54 06 78 0F 47 D2 47 06
8650: 51 1E 00 C9 D5 78 81 5F 16 00 C5 CD 0A 06 C1 D1
8660: 78 95 7A 9C D8 78 81 4F C9 CD 77 06 E8 CD 77 06
8670: EB 23 7C 85 C0 13 C9 7C 2F 67 7D 2F 6F C9 22 92
8680: 24 EB 22 94 24 32 90 24 3C 32 91 24 C9 3E 01 32
8690: 97 24 CD FF 06 D8 CD 5D 07 21 97 24 34 3A 90 24
86A0: BE C2 92 06 3A 90 24 32 9A 24 67 3A 91 24 6F CD
86B0: D7 07 3A 9A 24 6F 67 CD E3 07 2A 94 24 CD 8C 07
86C0: CD DA 03 3A 9A 24 3D C8 32 96 24 67 3A 9A 24 6F
86D0: CD D7 07 2A 94 24 CD BC 07 CD E0 03 79 EE 80 4F
86E0: 3A 96 24 67 3A 91 24 6F E5 CD E9 07 E1 CD DD 07
86F0: 3A 96 24 3D C2 C8 06 3A 9A 24 3D C2 A7 06 C9 3A
8700: 97 24 32 96 24 67 3A 97 24 6F CD EF 07 C2 1D 07
8710: 3A 96 24 3C 21 91 24 BE C2 02 07 37 C9 3A 96 24
8720: 21 97 24 BE C8 7E 32 9B 24 6F 3A 96 24 67 CD C8
8730: 07 EB 3A 97 24 67 3A 9B 24 6F CD C8 07 CD 4F 07
8740: 3A 9B 24 3C 21 91 24 BE DA 26 07 CA 26 07 C9 0E
8750: 03 1A 46 EB 12 70 23 13 0D C2 51 07 C9 3A 97 24
8760: 3C 32 98 24 67 3A 97 24 6F CD D7 07 3A 97 24 67
8770: 6F CD E3 07 21 9C 24 CD DA 03 3A 97 24 32 99 24
8780: 6F 3A 97 24 67 CD D7 07 21 9C 24 CD E0 03 79 EE
8790: 80 4F 3A 98 24 67 3A 99 24 6F E5 CD E9 07 E1 CD
87A0: DD 07 3A 99 24 3C 21 91 24 BE DA 7D 07 CA 7D 07
87B0: 3A 98 24 3C 21 91 24 BE C2 61 07 C9 2A 94 24 3A
87C0: 9A 24 3D 47 87 80 DF C9 25 2D 7C 87 87 84 85 6F
87D0: 87 85 2A 92 24 DF C9 CD C8 07 C3 D4 03 CD C8 07
87E0: C3 DA 03 CD C8 07 C3 0B 04 CD C8 07 C3 9F 04 CD
87F0: D7 07 7A B7 C8 79 E6 7F FE 37 D0 97 C9 21 91 08
8800: FE 33 DA 07 08 D6 30 DF C9 CD 16 08 C3 12 08 CD
8810: 1D 08 3E 33 DF C9 D6 18 D2 1D 08 C6 60 21 2B 08
8820: FE 33 DA 29 08 2F 3C C6 60 DF C9 41 80 00 40 F8
8830: 15 40 EC 83 40 D4 DB 40 B5 04 40 8E 3A 3F C3 EE
8840: 3E C7 C5 00 00 00 BE C7 C5 BF C3 EE C0 8E 3A C0
8850: 85 04 C0 D4 DB C0 EC 83 C0 FB 15 C1 80 00 41 80
8860: 00 40 F6 43 40 DA 83 40 B0 FC 40 80 00 3F 9E 08
8870: 3E 95 F6 3C 9B E5 00 00 00 3C 98 E5 3E 95 F6 3F
8880: 9E 08 40 80 00 40 B0 FC 40 DA 83 40 F6 43 41 80
8890: 00 00 00 00 3E C3 EF 3F 85 04 3F EC 83 40 80 00
88A0: 3F EC 83 3F 85 04 3E C3 EF 00 00 00 BE C3 EF BF
88B0: B5 04 BF EC 83 C0 80 00 BF EC 83 BF 85 04 BE C3
88C0: EF 00 00 00 22 20 23 EB 22 22 23 C5 3E C3 32 28
88D0: 23 CD E3 08 CD 0E 0C CD F8 08 D1 21 35 23 0E 10
88E0: C3 B9 0C 3E 04 11 35 23 CD 13 09 CD 24 09 CD AE
88F0: 09 CD 85 0B CD E1 09 C2 EE 08 C9 3E 02 11 3B 23
8900: CD 13 09 CD 62 09 CD D3 09 CD 85 08 CD ED 09 C2
8910: 06 09 C9 21 48 24 CD 7E 06 3E 03 CD DB 0C 11 46
8920: 23 C3 B6 0C CD 6D 0A 32 44 23 32 45 23 32 2D 23
8930: 21 39 09 CD 10 0D C3 0A 0B CD C1 0C 5E 23 56 E5
8940: CD 9F 0C E1 7E E6 10 C2 53 09 CD D0 0C CD DA 03
```

```
8950: C3 E9 0A 2A 22 23 CD E0 03 CD D0 0C CD DA 03 C3
8960: C2 0A 3E 80 32 2D 23 01 18 24 11 3F 24 21 35 23
8970: CD 94 09 01 18 24 11 30 24 21 38 23 CD 94 09 01
8980: 27 24 11 42 24 21 35 23 CD 94 09 01 27 24 11 33
8990: 24 21 38 23 C5 D5 CD D4 03 7A B7 CA AB 09 FE 40
89A0: CA AB 09 E1 CD E0 03 E1 C3 7E 0C E1 E1 C9 11 48
89B0: 24 21 0C 24 0E 3C CD DB 09 3A 45 23 B7 C2 C5 09
89C0: 3E 40 32 36 23 3A 44 23 21 45 23 96 C0 3E 40 32
89D0: 39 23 C9 11 48 24 21 12 24 0E 18 CD 88 0C C3 8D
89E0: 06 21 1D 0A CD 0F 0A CD 0A 0B C3 F3 09 21 29 0A
89F0: CD 0F 0A 21 46 23 CD D4 03 3E 06 CD DB 0C CD 9F
8A00: 04 21 46 23 CD DA 03 3A 2E 23 21 44 23 96 C9 3A
8A10: 44 23 32 2E 23 3E 80 32 2D 23 C3 10 0D CD 33 0A
8A20: D0 E6 10 CA E9 0A C3 C2 0A CD 33 0A D0 CD 4D 0B
8A30: C3 7D 0A CD C1 0C 23 7E FE 20 D0 E6 10 21 45 23
8A40: 7E C2 48 0A 3A 44 23 96 FE 03 DA 63 0A CD CA 0C
8A50: CD D4 03 7A B7 C8 79 E6 7F 4F 21 49 23 BE CC 89
8A60: 04 3F D0 CD C1 0C 23 7E F6 20 77 37 C9 21 0D 24
8A70: 0E 14 11 03 00 97 77 19 0D C2 76 0A C9 21 44 23
8A80: CD 8B 0C CD F5 0C 21 12 24 CD 7E 0C CD 07 0D 21
8A90: 15 24 CD 7E 0C 21 21 24 CD DA 03 CD FE 0C 21 24
8AA0: 24 CD 7E 0C CD E3 0C CD D0 0C CD E0 03 21 18 24
8AB0: CD 7E 0C CD EC 0C CD D0 0C CD E0 03 21 27 24 C3
8AC0: 7E 0C 21 45 23 CD 8B 0C CD E3 0C 21 3F 24 CD 7E
8AD0: 0C CD EC 0C 21 42 24 CD 7E 0C CD D0 0C CD D4 03
8AE0: 21 45 24 CD 7E 0C C3 7D 0A CD E3 0C 21 30 24 CD
8AF0: 7E 0C CD EC 0C 21 33 24 CD 7E 0C CD D0 0C CD D4
8B00: 03 21 36 24 CD 7E 0C C3 7D 0A 21 3F 24 11 0C 24
8B10: CD B6 0C 11 1B 24 CD B6 0C 21 30 24 11 0F 24 CD
8B20: B6 0C 11 1E 24 CD 86 0C 3A 45 23 B7 C2 30 0B 3C
8B30: CD 96 0C 21 39 24 CD DA 03 3A 44 23 21 45 23 96
8B40: C2 44 0B 3C CD 96 0C 21 2D 24 C3 DA 03 21 36 23
8B50: E6 10 C2 5B 0B 21 39 23 7E B7 C8 FE 40 C8 28 E5
8B60: CD D4 03 3A 2B 23 CD 1D 0B CD E0 03 21 18 24 CD
8B70: 7E 0C E1 CD D4 03 3A 2B 23 CD 16 0B CD E0 03 21
8B80: 27 24 C3 7E 0C 97 32 42 23 3A 44 23 3D CD 96 0C
8B90: 21 2F 23 CD DA 03 21 BD 0B CD 10 0D 21 41 23 CD
8BA0: D4 03 21 2F 23 CD 0B 04 CD DC 05 21 41 23 CD DA
8BB0: 03 21 46 23 CD E0 03 21 49 23 C3 DA 03 CD C1 0C
8BC0: 23 7E FE 20 D0 F5 CD E3 0C 21 3B 23 CD E0 03 21
8BD0: 32 23 CD DA 03 CD EC 0C 21 3E 23 CD E0 03 21 32
8BE0: 23 CD 9F 04 F1 E6 10 21 38 23 CA F0 0B 21 35 23
8BF0: CD 9F 04 CD D0 0C CD 98 04 CD CA 0C CD DA 03 CD
8C00: D3 05 21 41 23 CD 9F 04 21 41 23 C3 DA 03 3E 09
8C10: CD DB 0C EB 21 06 00 19 0E 12 CD BB 0C 21 35 23
8C20: 0E 06 CD B8 0C 3E 09 21 35 23 CD 32 0C 3E 0C 21
8C30: 3B 23 E5 0E 00 11 00 00 21 2C 23 36 00 F5 CD DB
8C40: 0C CD 70 0C F1 C6 06 FE 21 DA 3D 0C CD 53 0C E1
8C50: C3 DA 03 3A 2C 23 B7 CA 6D 0C C5 D5 CD 96 0C 21
8C60: 32 23 CD DA 03 D1 C1 21 32 23 C3 0B 04 16 40 C9
```

```
8C70: 23 7E FE 40 C8 2B CD 9F 04 21 2C 23 34 C9 E5 3A
8C80: 2D 23 A9 4F CD 9F 04 E1 C3 DA 03 3A 2D 23 87 FA
8C90: 94 0C 34 C9 35 C9 6F 26 00 11 00 00 C3 0F 05 21
8CA0: 00 00 7A E6 0F 57 FE 08 DA AE 0C F6 F0 57 CD 0F
8CB0: 05 79 D6 1B 4F C9 0E 03 7E 12 23 13 0D C2 88 0C
8CC0: C9 2A 20 23 3A 2C 23 C3 D6 0C 21 AC 23 C3 D3 0C
8CD0: 21 4C 23 3A 28 23 85 6F D0 24 C9 2A 22 23 85 6F
8CE0: D0 24 C9 3A 2B 23 CD 1D 08 C3 D4 03 3A 28 23 CD
8CF0: 16 08 C3 D4 03 3A 2B 23 CD 0F 08 C3 D4 03 3A 28
8D00: 23 CD 09 08 C3 D4 03 3A 2B 23 CD FD 07 C3 D4 03
8D10: 22 29 23 97 32 2C 23 32 28 23 CD 28 23 3A 2C 23
8D20: C6 20 32 2C 23 3A 2B 23 C6 30 32 2B 23 CD 28 23
8D30: 3A 2C 23 D6 1E 32 2C 23 3A 28 23 D6 2D FE 30 DA
8D40: 17 0D C9 00 00 00 00 00 00 00 00 00 00 00 00 00
8D50: C3 62 0D C3 96 0D C3 A7 0D C3 EA 0D C3 F7 0D C3
8D60: 85 0D 21 00 FE CD 7D 0D 21 26 29 22 A6 24 97 32
8D70: 2F 21 3E FD 32 2C 21 3E 07 32 2D 21 C9 32 33 21
8D80: 3E 07 C3 CC 02 B7 C4 8F 0D 53 CD 8F 0D EB C9 21
8D90: 2E 21 34 6E 5E C9 E6 03 CA A1 0D FE 02 CA 6E 0D
8DA0: C9 3E FF 32 2E 21 C9 01 E7 0D 11 37 21 CD 84 0E
8DB0: 11 E7 0D 21 34 21 0E 03 D7 01 34 21 2A 37 21 CD
8DC0: CB 0D 2A 39 21 CD C8 0D 2A 38 21 E8 2A A6 24 7A
8DD0: 87 CA DC 0D D6 0F C2 E5 0D 93 5F 65 78 8C D2 E5
8DE0: 0D 0A F6 10 02 03 C9 02 01 00 21 2F 21 34 01 34
8DF0: 21 11 3D 21 C3 84 0E 3A 2F 21 FE 01 CC 43 0E 2A
8E00: 3D 21 CD 58 0E 2A 3F 21 CD 58 0E 2A 41 21 CD 58
8E10: 0E 21 30 21 11 34 21 CD 77 0E CD 77 0E CD 77 0E
8E20: 3A 2F 21 FE 01 CA 38 0E FE 08 C0 2A 30 21 22 29
8E30: 21 3A 32 21 32 2B 21 C9 21 04 21 3A 33 21 E6 0F
8E40: B6 77 C9 2A 37 21 CD 52 0E 2A 39 21 CD 52 0E 2A
8E50: 3B 21 11 2C 21 C3 58 0E 11 2D 21 1A C6 03 12 87
8E60: 1F 5F DA 6E 0E 29 29 29 29 EB 72 23 73 C9 E8 7A
8E70: E6 0F 86 77 23 73 C9 1A E6 10 CA 7E 0E 37 7E 17
8E80: 77 13 23 C9 CD 8A 0E CD 8A 0E 0A 03 CD E6 00 E8
8E90: 73 23 72 23 E8 C9 00 00 C3 A8 0E C3 C0 0E 3A 58
8EA0: 21 E6 3F 21 59 21 86 C9 21 50 21 0E 0C CF 11 F3
8EB0: 0F 0E 10 D7 C9 32 58 21 07 D0 E6 80 32 59 21 C9
8EC0: 3A 59 21 E6 80 C0 C5 21 58 21 34 7E 0F D2 F4 0E
8ED0: CD 24 0F 22 50 21 32 56 21 CD 3C 0F 22 54 21 32
8EE0: 57 21 4F 3E 0B E8 CD 0A 06 6C 67 79 CD D5 0F 22
8EF0: 52 21 C1 C9 CD 30 0F CD D5 0F E8 2A 52 21 19 CD
8F00: D4 0F 97 32 59 21 C1 29 D8 27 D8 29 D8 3A 50 21
8F10: 95 3A 51 21 9C D8 C5 CD A4 0F CD 30 03 C1 3E 40
8F20: 32 59 21 C9 3A 34 21 2A 3D 21 01 5C 21 C3 5F 0F
8F30: 3A 36 21 2A 41 21 01 68 21 C3 5F 0F 3E 07 CD E4
8F40: 02 CA 56 0F 01 64 21 CD 59 0F 4F 3E 27 E8 CD 0A
8F50: 06 6C 67 79 2F C9 01 60 21 3A 35 21 2A 3F 21 E6
8F60: 10 3E 00 C2 6A 0F 3E 33 03 03 F5 7C FE 08 DA 74
8F70: 0F F6 F0 67 0A 5F 03 0A 57 19 7C 32 5A 21 CD D4
8F80: 0F F1 54 5D CC 0A 06 CD 93 0F 87 3A 5A 21 C8 21
```

```
8F90: FF 7F C9 87 1F 23 CD 9B 0F B7 1F 4F 7C 1F 67 7D
8FA0: 1F 6F 79 C9 2A 50 21 EB 2A 54 21 B7 CD 9B 0F CD
8FB0: DA 0F 3C 87 1F FE 40 DA BE 0F E6 3F F6 80 5F 3A
8FC0: 56 21 21 57 21 AE 7B F2 CC 0F F6 40 6F 3A 58 21
8FD0: F6 80 67 C9 7C B7 F0 C3 8B 00 7A 2F 47 7B 2F 4F
8FE0: 03 3E 01 C3 E9 0F 17 DB 29 54 5D 09 DA E6 0F EB
8FF0: C3 E6 0F A5 FF FC FF F4 FF FE FF 60 00 01 00 EA
9000: FF FE FF 00 00 00 00 00 C3 17 10 C3 65 10 C3 3B
9010: 11 21 70 21 DF 7E C9 97 32 70 21 21 10 00 22 72
9020: 21 C3 44 03 11 96 21 C3 2D 10 11 71 21 7C E6 03
9030: E8 DF 97 32 71 21 73 3A 71 21 E6 07 C8 3E 40 CD
9040: AE 11 B7 C9 97 32 71 21 CD 4D 10 B7 C9 7C FE B9
9050: CA 17 10 FE B4 CA 3D 10 FE B5 CA E9 10 FE 88 CA
9060: 1F 11 C3 53 03 CD A8 11 0F 0F 0F E6 0E 21 73 10
9070: C3 D5 10 4C 10 CB 10 01 11 16 11 8D 10 A0 10 89
9080: 10 83 10 21 79 21 35 C0 3E 40 C3 AE 11 21 00 84
9090: CD 53 03 97 32 79 21 3A 71 21 32 86 21 C3 C1 11
90A0: CD 84 03 C8 3A 70 21 E6 F0 B5 32 70 21 CD 84 03
90B0: CA AD 10 22 80 21 C3 C1 11 CD 84 03 C8 22 82 21
90C0: 21 87 21 CD E0 11 3E 10 C3 AE 11 3A 71 21 E6 07
90D0: C8 87 21 D9 10 DF 7E 23 66 6F E9 E9 10 73 11 8F
90E0: 11 9F 11 9F 11 9F 11 9F 11 21 8C 21 CD E0 11 2A
90F0: 72 21 22 7B 21 97 32 7A 21 32 7D 21 3E 20 C3 AE
9100: 11 CD C9 11 D0 21 00 85 CD 53 03 21 91 21 CD E0
9110: 11 3E 30 C3 AE 11 CD 84 03 C8 3E 00 CD AE 11 21
9120: 00 88 CD 53 03 3E 81 32 85 21 3E 85 32 84 21 3A
9130: 70 21 F6 80 32 70 21 C9 21 84 21 7E FE 9A DA 6D
9140: 11 CD 84 03 C0 3A 70 21 E6 80 CA 69 11 3A 70 21
9150: E6 7F 32 70 21 3E 10 32 79 21 3A 71 21 E6 80 3E
9160: 70 C2 66 11 3E 00 CD AE 11 21 00 00 C9 34 6F 6E
9170: 26 00 C9 3A 96 21 CD E6 00 29 29 29 29 7C CD 8A
9180: 11 67 3A 97 21 BC DC E9 10 C9 B7 F0 2F 3C C9 97
9190: CD 9E 0E E6 40 C8 21 79 21 8E C8 77 C3 E9 10 3A
91A0: 75 21 FE AA CA 76 21 C9 3A 70 21 E6 70 C9 E6 70
91B0: E5 6F 3A 70 21 E6 8F B5 32 70 21 6F 26 DB F7 E1
91C0: C9 21 70 21 7E C6 10 77 C9 11 7A 21 21 DC 11 B7
91D0: CD D3 11 CD D6 11 1A 9E 12 13 23 C9 20 00 00 00
91E0: 11 1C 20 0E 05 D7 C9 00 C3 F4 11 C3 29 12 C3 A3
91F0: 12 C3 DD 12 21 80 22 0E 07 CF 21 D7 22 CD 03 12
9200: 21 F8 22 11 20 13 0E 09 D7 11 18 40 72 23 1D C2
9210: 0C 12 C9 5F 29 A5 57 7B EE 22 21 B6 22 A6 B2 77
9220: E6 02 C0 21 E0 22 C3 09 12 D6 A0 5F E6 07 C0 3A
9230: 86 22 E6 20 C0 7B CD 6B 12 3E 0F CD 90 12 3A B2
9240: 22 11 30 22 CD 9D 12 21 B2 22 CD 64 12 3A B6 22
9250: E6 02 C0 3E 08 CD 90 12 3A B0 22 11 80 21 CD 9D
9260: 12 21 B0 22 7E C6 02 E6 7F 77 C9 FE 80 21 B2 22
9270: CA 7E 12 87 C0 3A B6 22 E6 02 C0 21 80 22 7E E6
9280: 3F CC 89 12 7E E6 C0 77 C9 7E EE 40 2C 77 2D C9
9290: CD 2A 01 7A CD E6 00 7A E6 10 B4 67 C9 EB DF 73
92A0: 23 72 C9 21 B4 22 7E 87 CA 83 12 35 21 B5 22 34
```

```
92B0: 6E 7E C9 21 B1 22 16 A1 1E B6 7E FE AA CA D0 12
92C0: 21 B3 22 16 A3 1E C6 7E FE AA CA D0 12 3E 00 C9
92D0: 36 FF 7B 32 B5 22 3E 10 32 B4 22 7A C9 3A B1 22
92E0: B7 F2 FE 12 3A B3 22 B7 F8 21 30 22 CD 13 13 01
92F0: C7 22 11 F8 22 CD C4 08 3E AA 32 B3 22 C9 21 B0
9300: 21 CD 13 13 01 B7 22 11 D7 22 CD C4 08 3E AA 32
9310: B1 22 C9 DF 3A B6 22 E6 08 C8 11 20 00 C3 CF 13
9320: 3B A0 F2 41 E6 66 3F CC CD 00 00 00 00 00 00 00
9330: C3 3C 13 C3 5F 13 21 A4 21 DF 7E C9 11 B5 13 21
9340: A4 21 0E 06 D7 C9 7C E6 07 5D 21 A4 21 DF 73 C9
9350: 21 A0 21 11 A4 21 0E 04 D7 3E 01 32 A3 21 C9 3A
9360: A5 21 B7 C8 3A 1D 20 FE FE C2 74 13 3A 1C 20 FE
9370: E0 D4 50 13 3A A8 21 0F D0 21 A3 21 35 C0 3A A7
9380: 21 77 3A A8 21 E6 02 C4 A5 13 21 A2 21 35 CA 9A
9390: 13 3A A1 21 21 A0 21 86 77 C9 3A A6 21 77 21 A1
93A0: 21 97 96 77 C9 2A A0 21 3A A9 21 67 E5 CD 63 01
93B0: E1 24 C3 63 01 00 02 40 02 00 01 00 00 00 00 00
93C0: C3 D6 13 C3 8B 14 C3 6A 15 C3 31 15 C3 42 15 C3
93D0: 2E 16 3A 77 20 C9 21 50 20 0E A5 CF C9 7C E6 07
93E0: FE 07 3F D8 C3 CC 02 7C 0F 7D C3 BB 01 7C E6 07
93F0: F6 20 67 22 50 20 C9 C6 3E 32 94 20 0F 11 0A 14
9400: D2 06 14 11 1C 14 3E 01 E7 C9 12 31 13 31 07 30
9410: 08 30 09 30 10 30 01 28 01 29 FF FF 07 31 08 31
9420: 09 31 10 31 12 30 13 30 00 28 00 29 FF FF 7C E6
9430: 04 C2 70 14 3E 01 11 52 20 CD 41 14 3E 02 11 54
9440: 20 A4 C8 7D 0F 0F 0F 0F CD 50 14 7B C6 04 5F 7D
9450: E5 CD 5D 14 EB 73 23 72 2B EB E1 B7 C9 21 3D 16
9460: E6 0F FE 0A DA 6A 14 21 F5 1F 87 87 87 87 DF C9
9470: 7C 0F 7D D2 83 14 5D 3A 5B 20 21 95 20 DF 73 3A
9480: 5B 20 3C FE 60 D0 32 5B 20 B7 C9 21 7E 20 11 6A
9490: 20 0F DA A5 14 EB CD E0 14 CD EE 11 32 8B 20 3E
94A0: FF 32 5A 20 C9 CD E0 14 3A 1D 20 3C E6 07 CC 09
94B0: 15 CD 13 15 97 CD 11 10 E6 B0 5F CD EA 17 E6 01
94C0: 6F 3E 02 E5 E7 E1 29 DF 3A 94 20 E6 41 29 DF B3
94D0: 5F 2A 5E 20 7C B5 CA DB 14 3E 88 B3 32 77 20 C9
94E0: 22 68 20 EB 22 64 20 11 0E 00 19 22 66 20 2A 68
94F0: 20 11 0C 00 19 3A 92 20 77 11 07 00 19 7E E6 F0
9500: 5F 3A 93 20 E6 0F B3 77 C9 21 56 20 CD 1D 15 22
9510: 62 20 C9 21 52 20 CD 1D 15 22 60 20 C9 3A 77 20
9520: E6 01 1E 02 C2 28 15 5F 16 00 19 7E 23 66 6F 2B
9530: C9 2A 68 20 5E 23 B7 CA 3D 15 53 5E 23 22 68 20
9540: EB C9 FE 10 21 52 20 DA 54 15 D6 10 FE 20 21 56
9550: 20 D2 62 15 F5 3A 1E 20 E6 01 CD 22 15 F1 23 C3
9560: 67 15 D6 20 21 50 20 DF 7E C9 21 5A 20 34 7E 0F
9570: DA 92 15 2A 60 20 23 22 60 20 CD DA 15 21 92 20
9580: CD B9 15 2A 64 20 3A 5A 20 E6 02 CD C4 15 22 64
9590: 20 C9 3A 5A 20 E6 02 CA 0B 10 2A 62 20 23 22 62
95A0: 20 CD DA 15 21 93 20 CD B9 15 2A 66 20 3A 5A 20
95B0: E6 04 CD C4 15 22 66 20 C9 7A E6 10 CA C0 15 37
95C0: 7E 17 77 C9 C2 D1 15 EB 29 29 29 29 EB 72 23 73
```

```
95D0: C9 7A E6 0F B6 77 23 73 23 C9 3A 77 20 A6 FA 0A
95E0: 16 7E E6 40 C2 FF 15 7E E6 3F CD 2A 01 1E 10 C2
95F0: F4 15 1E 00 7A CD E6 00 7B E6 10 B4 57 5D C9 7E
9600: E6 3F 2A 50 20 DF 5E 23 56 C9 3A 77 20 E6 08 C2
9610: 17 16 CD 0E 10 EB C9 2A 5E 20 7C B5 EB C8 1B E8
9620: 22 5E 20 2A 5C 20 5E 23 56 23 22 5C 20 C9 3A 77
9630: 20 E6 08 37 C0 22 5C 20 EB 22 5E 20 C9 08 0F 88
9640: 8F 08 0F 88 8F 08 0F B8 BF 0B 0F 88 BF 86 83 87
9650: 0B 84 8C 87 09 86 B3 87 0A 84 8C 87 09 86 83 87
9660: 0B 84 8C B7 09 86 83 87 0A 84 8C 87 09 00 00 00
9670: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
9680: 21 30 29 22 20 29 C9 7C D6 EB CA 9B 16 3D CA A0
9690: 16 3D CA A5 16 3D CA AF 16 37 C9 7D 32 20 29 C9
96A0: 7D 32 21 29 C9 EB 2A 20 29 73 23 22 20 29 C9 3A
96B0: 30 29 85 FE AA C0 97 32 30 29 C3 31 29 00 00 00
96C0: 00 00 00 00 C3 D0 16 C3 0B 17 21 44 20 DF 7E C9
96D0: CD E3 00 32 45 20 21 00 D0 CD 02 17 7C FE D4 CA
96E0: 02 17 E6 FC D6 D0 37 C0 7D 32 48 20 7C E6 03 21
96F0: 07 17 DF 7E 32 44 20 97 32 4A 20 32 46 20 32 47
9700: 20 C9 7D 32 49 20 C9 22 21 12 11 D5 21 4A 20 34
9710: CD 68 17 CD 18 17 D1 C9 CD 93 17 FE 00 C8 FE 01
9720: CA 36 17 FE 02 CA 55 17 CD A8 17 D2 55 17 CD C8
9730: 17 D8 FE 03 D2 5A 17 C9 CD A8 17 3E 00 D2 82 17
9740: CD C8 17 DA 47 17 C0 3E 03 CD 82 17 3A 4A 20 E6
9750: 01 37 C3 18 03 3E 00 C3 5C 17 3E 01 CD 82 17 3A
9760: 4A 20 E6 01 B7 C3 18 03 21 45 20 5E CD E3 00 77
9770: AB A6 5F 3E 02 21 46 20 CD 7E 17 3E 20 23 A3 C8
9780: 34 C9 6F 3A 44 20 CD 9F 17 E6 30 B5 CD 9F 17 32
9790: 44 20 C9 3A 44 20 CD 9F 17 E6 03 C9 3A 45 20 5F
97A0: 3A 4A 20 0F 7B D0 0F 0F 0F 0F C9 CD 9C 17 21 49
97B0: 20 B6 6F E6 01 C8 7D E6 04 C8 7D E6 88 EE 88 C0
97C0: CD D6 17 21 48 20 BE C9 3A 44 20 E6 11 FE 11 37
97D0: C0 CD D6 17 95 C9 2A 46 20 3A 4A 20 0F 7C D8 7D
97E0: 6C C9 00 00 C3 04 18 C3 31 18 C3 FD 17 C3 A2 18
97F0: FE 06 DA F7 17 C6 09 21 C3 24 DF 7E C9 3A E2 24
9800: 32 E3 24 C9 21 C1 24 11 2D 1B 0E 11 D7 11 2F 1B
9810: 0E 0F D7 36 01 3E 98 32 D3 24 C9 7C FE 62 CA 7F
9820: 18 EB 21 C0 24 0F D2 2F 18 34 7E FE 22 D0 DF 73
9830: C9 5F B7 CC 67 18 01 C1 24 0A E6 03 CA 6F 18 D6
9840: 02 C0 CD 02 1B 23 7E 93 C0 02 0E 10 CD E2 1A 2A
9850: E1 24 C4 85 18 21 E1 24 35 36 03 CA 60 18 36 01
9860: 3A C2 24 32 E4 24 C9 21 E4 24 35 F0 36 00 C9 3A
9870: E4 24 87 C0 3E 01 CD FC 1A BB C0 3E 01 02 C9 3A
9880: C1 24 FE 03 C8 55 0E 10 CD DC 1A 3E 03 CA 92 18
9890: 3E 04 6A CD F3 1A DF 6E 62 E5 CD 63 01 E1 24 C3
98A0: 63 01 3A C1 24 3D C0 21 10 25 22 EE 24 3E 04 CD
98B0: 36 13 32 14 29 01 01 07 CD DA 18 01 04 09 CD DA
98C0: 18 97 32 E2 24 2A 14 29 26 5C F7 0E 01 CD DC 1A
98D0: C4 87 19 21 C1 24 34 C3 A5 1A CD DC 1A C8 21 E2
98E0: 24 36 01 23 7E 30 C2 E4 18 CD E2 1A 32 15 29 78
```

```
9BF0: CD FC 1A CD 1D 1B 22 E9 24 CD 02 1B 79 3C DF EB
9900: 21 EB 24 0E 03 D7 21 01 5C F7 CD 25 19 01 B0 04
9910: CD 4C 19 CD 3D 19 CD 5B 19 06 01 0D C2 10 19 2A
9920: EB 24 CD B5 18 3A 15 29 B7 CB CD D2 13 3C E6 01
9930: 6F 26 6B F7 C9 3A 1C 20 E6 07 C2 35 19 3A 1C 20
9940: D6 02 E6 07 C2 3D 19 05 C2 35 19 C9 2A EA 24 CD
9950: 99 1B 3A E9 24 21 EA 24 B6 77 C9 3A EC 24 CD 64
9960: 19 3A ED 24 CD 7B 19 29 29 29 29 10 10 10 10 EB
9970: 2A EE 24 73 23 72 23 22 EE 24 C9 B7 F2 E6 00 2F
99B0: 3C CD E6 00 C3 BB 00 3A E5 24 FE AA CA E6 24 3E
9990: 02 CD FC 1A 36 00 3E 0B CD FC 1A 36 00 21 10 25
99A0: CD B6 19 F5 21 12 25 CD B6 19 5F F1 B3 1F 5F 3E
99B0: 04 CD FC 1A 73 C9 22 10 29 3E 06 CD FC 1A CD E1
99C0: 19 3E 05 CD FC 1A CD 4C 1A 5F FE FF 3E 03 CA FC
99D0: 1A 3E 07 CD FC 1A 16 00 E5 CD 0A 06 D1 13 1A B5
99E0: C9 32 12 29 3C CB 06 7E CD F3 19 05 7B FE 02 D2
99F0: EB 19 C9 7B CD 0E 1B CB 3A 12 29 DF EB 7B 3D CD
9A00: 0E 1B CB CD 8B 00 19 7C B7 F0 7B CD 0E 1B EB 7B
9A10: 3D CD 0E 1B 19 CD BB 00 EB 7B 3C CD 0E 1B C8 19
9A20: EB 78 D6 02 CD 0E 1B CB 19 7C B7 7B F2 30 1A 3D
9A30: 4F 3C CD 0E 1B EB 79 3D CD 0E 1B 19 10 EB 79 CD
9A40: 22 1B 73 23 72 3E 0B CD FC 1A 34 C9 11 FF 7F 4F
9A50: 3E FF 32 13 29 3E 7E 91 47 D5 CD 74 1A D1 7D 93
9A60: 7C 9A D2 6A 1A EB 78 32 13 29 05 7B B9 D2 59 1A
9A70: 3A 13 29 C9 7B 3C CD B7 1A DB E5 7B 91 CD B7 1A
9AB0: D1 DB CD BB 00 19 C9 11 00 00 C5 F5 CD 0E 1B CA
9A90: 9E 1A 19 EB F1 3C 0D C2 BB 1A EB C1 B7 C9 F1 C1
9AA0: 21 FF 3F 37 C9 0E 40 CD DC 1A CB CD F0 1A EB 21
9AB0: F2 24 0E 0F CD D0 1A 21 E0 0E 3A E1 24 DF 22 F0
9AC0: 24 2A EE 24 11 10 DB 19 10 EB 21 F0 24 C3 CF 13
9AD0: 1A 13 77 23 36 00 23 0D C2 D0 1A C9 CD F0 1A C3
9AE0: E5 1A CD 02 1B CD D2 13 0F 79 D2 EE 1A B7 A6 C9
9AF0: 2A E1 24 2D 21 C3 24 CB 21 D2 24 C9 CD F0 1A DF
9B00: 7E C9 2A E1 24 2D 21 3E 1B CB 21 46 1B C9 CD 22
9B10: 1B CD 1D 1B 7C FE 07 CB E6 0F FE 0B C9 7E 23 66
9B20: 6F C9 2A 10 29 B7 B7 D2 2B 1B 24 DF C9 00 0B CF
9B30: 5B 00 F0 FF 05 0B 02 B1 02 B1 00 00 00 00 16 9F
9B40: 01 06 FC 01 0B 04 34 1F 03 03 0C 01 0B 04 00 00
9B50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
9B60: 00 00 00 00 C3 FD 1B 21 A0 24 DF 7E C9 31 FF 2F
9B70: 21 00 20 01 00 0F CF 05 C2 76 1B CD 9A 00 CD 03
9BB0: 00 CD E4 17 CD EB 11 11 40 1C CD 2A 1C 2A AB 24
9B90: 22 AC 24 2A C7 1B 22 B0 24 3A B0 24 FE AA C2 A7
9BA0: 1B CD B1 24 C3 B5 1B 3E 21 32 A0 24 2A C5 1B 22
9BB0: AE 24 CD AE 24 CD F1 11 CD ED 17 CD 0A 1C 97 32
9BC0: B4 24 C3 99 1B 76 C9 55 C9 7C 0F 11 B2 1C D2 2A
9BD0: 1C 3A 1D 20 BD C2 D1 1B C9 3E 9B 32 A2 24 32 A1
9BE0: 24 CD B0 00 2B 22 AB 24 3A A1 24 0F 0F 0F 0F CD
9BF0: B0 00 2B 22 AA 24 21 01 00 22 AC 24 C9 21 A2 24
9C00: BE C0 2A AC 24 2B 22 AC 24 C9 2A AC 24 7C B5 C0
```

```
9C10: CD D2 13 E6 01 2A A8 24 11 2E 1C C2 24 1C 2A AA
9C20: 24 11 38 1C 7C B5 C8 22 AC 24 3E 01 E7 C9 02 50
9C30: 00 5C 00 68 01 62 FF FF 03 50 01 68 03 5C FF FF
9C40: 01 43 F0 01 F0 02 F0 03 F0 04 FF 09 FF 0A FF 11
9C50: FF 12 FF 21 FF 22 FF 23 FF 24 FF 25 FF 26 FF 27
9C60: FF 19 FF 1A 0A 31 0B 30 0D 30 0E 30 11 30 14 31
9C70: 15 31 00 30 01 2A 01 28 00 48 00 68 77 38 FF C8
9C80: FF FF 00 60 03 61 C0 58 02 5C 03 68 01 29 00 80
9C90: 11 31 01 30 05 B1 00 91 9C 03 64 04 06 31 20 91
9CA0: 64 03 9C 04 06 31 40 91 03 5C 00 B4 80 91 00 85
9CB0: 00 5C 00 68 00 60 00 61 01 62 03 62 11 30 FF FF
9CC0: 4D 41 49 4E 32 2E 31 2D 48 41 52 56 45 59 00 00
9CD0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
9CE0: 21 C0 FF 22 1D 20 21 FF FF 22 1F 20 CD C4 16 CD
9CF0: 80 16 CD 50 0D CD 98 0E CD C0 13 CD 08 10 CD 30
9D00: 13 21 00 00 22 3C 20 CD F8 1D 3E 0D CD CF 00 FB
9D10: 76 3E 18 CD CF 00 97 32 21 20 32 22 20 32 2D 20
9D20: D3 00 D3 01 21 53 07 CD A3 1D 3E 50 32 32 20 C3
9D30: 28 03 3A 1D 20 F6 1F 32 1D 20 3E FC 32 1C 20 F1
9D40: F8 C9 E5 D5 C5 FB 3A 1C 20 C6 02 32 1C 20 B7 1F
9D50: CD 59 1D C1 D1 E1 F1 FB C9 0F DA 43 1F 6F 26 1F
9D60: 6E E9 21 1C 20 3E 04 CD 80 1D CD B5 03 21 84 24
9D70: 34 CA 24 00 3A 1D 20 CD 53 0D 3A 1D 20 C3 C3 13
9D80: 23 34 C0 3D C2 80 1D C9 2A 2E 20 23 22 2E 20 2A
9D90: 30 20 11 00 FF 19 D4 84 1D 22 30 20 CD D9 00 C8
9DA0: 2A 2E 20 22 25 20 2B 22 30 20 21 00 00 22 23 20
9DB0: 22 2E 20 C9 E8 2A 23 20 23 22 23 20 2A 25 20 19
9DC0: C9 FE 01 CA 15 1E FE 02 CA 0A 1E FE 03 CA F8 1D
9DD0: FE 04 C0 E5 CD E1 1D E1 C0 22 3A 20 E8 22 3C 20
9DE0: C9 2A 3C 20 7C B5 C9 CD E1 1D C8 28 22 3C 20 2A
9DF0: 3A 20 7E 23 22 3A 20 C9 7C B5 3E AA C2 03 1E 97
9E00: 21 4D 1F 22 36 20 32 38 20 C9 21 21 20 3A 35 20
9E10: 96 C8 3E 01 C9 1A 6F 13 1A 67 13 3C C8 3A 2D 20
9E20: FE AA CA 1D 1E D5 CD 61 1E D1 C3 15 1E 3E AA 32
9E30: 2D 20 3E 09 CD CF 00 F1 F8 C9 3A 2D 20 FE AA C0
9E40: CD CD 00 E6 10 C0 97 32 2D 20 CD D1 00 CD 93 00
9E50: 3E 08 CD CF 00 CD 61 1E 21 21 20 D2 5F 1E 23 34
9E60: C9 E8 7A E6 F8 0F 0F 21 40 00 DF 7E 23 66 6F 84
9E70: 37 C8 E5 E8 7D 87 C9 32 35 20 21 00 00 22 21 20
9E80: C9 FE 10 CA 95 1E 2A 33 20 29 29 29 29 E6 0F 85
9E90: 6F 22 33 20 C9 2A 33 20 97 32 33 20 32 34 20 C3
9EA0: 61 1E 21 32 20 35 CC 2A 1D CD C7 16 3A 1D 20 E6
9EB0: 1F C0 3A 1D 20 07 07 07 E6 07 FE 06 D0 F5 C6 22
9EC0: CD E6 00 29 29 29 29 5C F1 21 27 20 DF 73 C9 32
9ED0: CC 13 08 67 18 15 74 02 0C F0 17 04 36 13 04 11
9EE0: 10 05 CA 16 01 9E 0E 64 F3 1F 7D FE 07 CA 00 00
9EF0: 37 C9 00 00 00 00 00 00 00 00 00 00 00 00 00 00
9F00: 40 4E 5E 56 86 86 8F AA 8C 4E 78 56 86 86 8F AA
9F10: 89 80 78 86 86 86 8F AA 8C 4E 78 56 86 86 8F AA
9F20: 8C 4E 4D 4D 86 86 8F AA 8C 4E 4D 4D 86 86 8F AA
```

```
9F30: 89 B0 4D B9 B6 86 8F AA 8C 4E 4D 4D 86 86 BF AA
9F40: C3 62 1D 3A 38 20 FE AA C0 2A 36 20 E9 C9 3E 01
9F50: CD C9 13 C3 D5 00 3E 01 CD 5F 0D C3 D5 00 97 CD
9F60: C9 13 E5 3A 1D 20 0F DA 71 1F CD BF 1F 6F C3 81
9F70: 1F CD E7 1D 6F C3 81 1F 97 CD C9 13 E5 97 CD 5F
9F80: 0D D1 63 C3 D5 00 C3 C6 13 C3 56 0D C3 9B 0E CD
9F90: 88 1D 3A 23 20 21 39 20 BE 77 C6 CD EB 11 3A 23
9FA0: 20 CD 64 1B 3A 23 20 C3 E7 17 CD 33 13 C3 3A 1E
9FB0: CD 4E 1F C3 59 0D CD 56 1F C3 5C 0D C3 A2 1E 21
9FC0: 1D 20 7E E6 1F C2 D3 1F 66 3A 1E 20 29 17 29 17
9FD0: 29 17 C9 7E E6 E0 0F 0F 0F 0F 2F B6 0F E6 7F 21
9FE0: CF 1E BE DA ED 1F 96 23 23 23 C3 E2 1F 23 5E 23
9FF0: 56 EB E9 21 1E 20 DF 7E C9 00 00 00 00 00 00 00
```

```
0000    *
0000    * CRRES FLIGHT SOFTWARE---BURST EXECUTIVE SECTION
0000    * WRITTEN BY PETER R HARVEY
0000    * FILE BEXEC.A
0000    *
0000    * 8085 SPECIFIC INFORMATION
0000    *
0000    PSW     EQU    6
0000    SP      EQU    6
0000    *
0000    * ROM ALLOCATIONS OF THE 4K AREA
0000    *
0000            ORG    40H
0040    BIO     DS     340H
0380    BEXEC   DS     0C0H
0440    BLD     DS     100H
0540    BSMP    DS     380H
08C0    BFMT    DS     180H
0A40    BCMP    DS     100H
0B40            DS     0C00H-$
0C00    FFP     DS     300H
0F00    *
0F00    * RAM ALLOCATIONS/EQUATES OF THE 2K AREA
0F00    *
0F00    RAM1    EQU    1000H  START ADDRESS
0F00    RAMSIZE EQU    2048   SIZE IN BYTES
0F00    MEM     EQU    8000H  BURST MEMORY BANK
0F00    *
0F00            ORG    RAM1
1000    BIORAM  DS     20H
1020    BEXRAM  DS     10H
1030    BFMTRAM DS     60H
1090    BCMPRAM DS     30H
10C0    BSMPRAM DS     140H
1200    BLDRAM  EQU    $
1200    STACK   EQU    RAM1+RAMSIZE
1200    *
1200    * DEFINE THESE GLOBALLY
1200    *
1200            COM    BIO
1200            COM    BEXEC
1200            COM    BLD
1200            COM    BSMP
1200            COM    BFMT
1200            COM    BCMP
1200            COM    FFP
1200            COM    BIORAM
1200            COM    BEXRAM
1200            COM    BLDRAM
```

```
1200                0050         COM    BSMPRAM
1200                0051         COM    BFMTRAM
1200                0052         COM    BCMPRAM
1200                0053 *
1200                0054 * RESET VECTOR
1200                0055 *
1200                0056         ORG    0
0000 C3 80 03       0057         JMP    BEXEC
0003                0058 *
0003                0059 * BURST PROCESSOR EXECUTIVE CONTROL
0003                0060 *
0003                0061         ORG    BEXEC
0380 31 00 18       0062         LXI    SP,STACK    INIT STACK POINTER
0383 21 11 11       0063         LXI    H,1111H
0386 CD FA 03       0064         CALL   MARK
0389 CD 40 00       0065         CALL   BIOINIT  INITIALIZE THE DRIVERS
038C CD 67 00       0066         CALL   D5MS
038F CD 40 04       0067         CALL   BLDINIT  INIT THE USER PROGRAM LDR
0392 CD 40 05       0068         CALL   BSMPINIT  INIT THE SAMPLING ROUTINES
0395 97             0069         SUB    A
0396 32 20 10       0070         STA    CMDERR
0399 21 00 00       0071         LXI    H,0
039C CD FA 03       0072         CALL   MARK
039F                0073 *
039F CD 4C 00       0074 BURST   CALL   RECEIVE  RECEIVE A COMMAND
03A2 C2 A8 03       0075         JNZ    BEX    IF READY, EXECUTE
03A5 CD C0 03       0076         CALL   WAIT   POWER DOWN-WAIT
03A8 C3 9F 03       0077         JMP    BURST
03AB                0078 *
03AB 22 21 10       0079 BEX     SHLD   CMDCPY
03AE CD FA 03       0080         CALL   MARK
03B1 CD 43 05       0081         CALL   BSMPCMD  IF A SAMPLE CMD
03B4 D2 9F 03       0082         JNC    BURST  OK.
03B7 CD 43 04       0083         CALL   BLDCMD  IF A USER PROGRAM LOAD
03BA DC CB 03       0084         CC     ERROR  OK ELSE ERROR
03BD C3 9F 03       0085         JMP    BURST
03C0                0086 *
03C0 2A C9 03       0087 WAIT    LHLD   PWRDN
03C3 22 23 10       0088         SHLD   PROG
03C6 C3 23 10       0089         JMP    PROG
03C9 76             0090 PWRDN   HLT
03CA C9             0091         RET
03CB                0092 *
03CB                0093 * DIAGNOSTIC OUTPUT ROUTINES
03CB                0094 *
03CB 21 20 10       0095 ERROR   LXI    H,CMDERR  UPDATE COMMAND ERROR
03CE 34             0096         INR    M
03CF C9             0097         RET
03D0                0098 *
03D0 11 00 10       0099 MEMTEST LXI    D,RAM1  START AT RAM START
```

```
03D3 0E 08                    MVI    C,8
03D5 2E 00                    MVI    L,0
03D7 CD E6 03           MTN   CALL   MTPAG
03DA 29                       DAD    H
03DB CA DF 03                 JZ     MTOK
03DE 2C                       INR    L
03DF 14                 MTOK  INR    D
03E0 7A                       MOV    A,D
03E1 0D                       DCR    C
03E2 C2 D7 03                 JNZ    MTN
03E5 C9                       RET
03E6                    *
03E6 1E 00              MTPAG MVI    E,0
03E8 CD F1 03           MTP1  CALL   MTLOC
03EB C0                       RNZ
03EC 1C                       INR    E
03ED C2 E8 03                 JNZ    MTP1
03F0 C9                       RET
03F1                    *
03F1 1A                 MTLOC LDAX   D      SAVE OLD VALUE
03F2 47                       MOV    B,A
03F3 2F                       CMA    .      FLIP ALL BITS
03F4 12                       STAX   D
03F5 1A                       LDAX   D
03F6 2F                       CMA
03F7 12                       STAX   D
03F8 B8                       CMP    B
03F9 C9                       RET
03FA                    *
03FA 7D                 MARK  MOV    A,L    OUTPUT TO DIAGNOSTIC LEDS
03FB D3 00                    OUT    0
03FD 7C                       MOV    A,H
03FE D3 01                    OUT    1
0400 C9                       RET
0401                    *
0401 42 55 52 53              ASC    'BURST 2-1-85 PR HARVEY'
     54 20 32 2D
     31 2D 38 35
     20 50 52 20
     48 41 52 56
     45 59
0417 00              V        DB     256    END OF BEXEC
0418                    *
0418                    * VARIABLES
0418                    *
0418                          ORG    BEXRAM
1020                    CMDERR DS    1      *COMMAND ERRORS FOUND
1021                    CMDCPY DS    2
1023                    PROG  DS     2      POWER DOWN PROGRAM
1025                    *
```

```
1025       * EXTERNAL MODULE DEFINITIONS
1025       *
1025               ORG    BIO
0040       BIOINIT DS     3
0043       GETMASK DS     3
0046       SETMASK DS     3
0049       RECSTAT DS     3
004C       RECEIVE DS     3
004F       SEND    DS     3
0052       ADPWR   DS     3
0055       SAMPLE  DS     3
0058       MEMPWR  DS     3
005B       MARSET  DS     3
005E       BANKSET DS     3
0061       MODESET DS     3
0064       SECOND  DS     3
0067       D5MS    DS     3
006A       READ    DS     3
006D       WRITE   DS     3
0070       *
0070               ORG    BLD
0440       BLDINIT DS     3
0443       BLDCMD  DS     3
0446       *
0446               ORG    BSMP
0540       BSMPINIT DS    3
0543       BSMPCMD DS     3
```

```
0000    *
0000    * CRRES FLIGHT SOFTWARE---BURST I/O DRIVER SECTION
0000    * WRITTEN BY PETER R HARVEY
0000    *
0000    * FILE BIO.A - COMPATIBLE WITH BREADBOARD, PROTOTYPE AND
0000    *              FLIGHT HARDWARE SYSTEMS.
0000    *
0000    * SYSTEM OPTIONS
0000    *
0000    COMLEN EQU    16     SERIAL COMMUNICATION BIT LENGTH
0000    *
0000    * SYSTEM DESCRIPTION
0000    *
0000    ROM1   EQU    0      4K ROM
0000    RAM1   EQU    1000H  2K RAM
0000    *
0000    * A/D INTERFACING FOR HIGH SPEED DIGITIZATIONS.
0000    *  TO MUX ADDRESS XXX, READ ADC+(XXX*2)
0000    *
0000    ADC    EQU    3000H  A/D DATA
0000    ADCTL  EQU    3001H  A/D CONTROL
0000    MUXAD  EQU    0FEH   MUX BITS
0000    HIGH   EQU    10H    HIGH GAIN OVERRIDE
0000    LOW    EQU    20H    LOW GAIN OVERRIDE
0000    BV34AC EQU    06H    STUB QTY
0000    GND    EQU    LOW+BV34AC*2  SET DC QTY ON MUX
0000    *
0000    KGAINS EQU    23H    KELLEY AUTO GAIN INPUT BITS
0000    KLYQTY EQU    13     KELLEY AUTO GAIN QTY NUMBER
0000    *
0000    FLIGHT EQU    1      FLIGHT BOARD == YES
0000    *
0000    * BURST MEMORY ADDRESSING
0000    *
0000    MEM    EQU    8000H  BURST MEMORY BANK --WRITE ADDRESS
0000    MEMRD  EQU    MEM+7000H  BURST MEMORY BANK -- READ ADDRESS
0000    HIMAR  EQU    70H    HIGH MEM ADDRESS BITS
0000    AUTOWRITE EQU 80H    AUTOWRITE TO MEMORY
0000    LOADADR EQU   40H    LOAD MAR
0000    UPPER6 EQU    3FH    UPPER 6 MAR BITS
0000    RELAYS EQU    20H    RELAY CONTROL BITS
0000    ENDAD  EQU    63H    END ADDR FOR WRAP-AROUND
0000    RLYCTL EQU    ENDAD  RELAY CONTROL TOO
0000    SRRLY  EQU    20H    SET/RESET RELAY
0000    LOMAR  EQU    0CFFFH  LOW 12 BITS OF MAR
0000    *
0000    * BREADBOARD I/O PORTS WHICH ARE DIFFERENT FROM ABOVE
0000    *
0000    A8155  EQU    20H    PROGRAMMABLE I/O CONTROL PORTS
```

```
0000            BB155    EQU   60H
0000            BBHIMAR  EQU   BB155+1  BREADBOARD HIGH MAR REGISTER
0000            BBRELAYS EQU   BB155+2  BREADBOARD RELAY CONTROLS
0000            *
0000            PSW      EQU   6        8085 INFORMATION
0000            SP       EQU   6
0000            SHLX     EQU   10H      STORE HL AT [DE]
0000            SLDE     EQU   18H      SHIFT LEFT [DE]
0000            RIM      EQU   20H      READ INT MASK
0000            SIM      EQU   30H      SET INT MASK
0000            SDO      EQU   80H      SERIAL DATA OUTPUT
0000            SDOEN    EQU   40H      SERIAL ENABLE
0000            RES75    EQU   10H      RESET 7.5 FLOP
0000            MSE      EQU   8        INTERRUPT MASK SET ENABLE
0000            DIS75    EQU   4        DISABLES FOR THE INTERRUPTS
0000            DIS65    EQU   2
0000            DIS55    EQU   1
0000            *
0000            AOUT     EQU   1        8155 [SA3001] PROGRAMMING
0000            BOUT     EQU   2
0000            COUT     EQU   12
0000            NOTM     EQU   040H     TIMER OFF
0000            TM       EQU   0C0H     TIMER ON
0000            *
0000            * INTERRUPTS
0000            *
0000                     ORG   6*8+4    2KHZ CLOCK
0034 E5         INT65    PUSH  H
0035 2A 04 10            LHLD  TIM1VECT
0038 E9                  PCHL  .
0039            *
0039                     ORG   7*8+4    INT 7.5
003C C3 C5 00            JMP   RST75
003F            *
003F            * BIO ENTRY POINTS
003F            *
003F                     ORG   BIO
0040 C3 7C 00            JMP   BIOINIT
0043 C3 AF 00            JMP   GETMASK
0046 C3 AA 00            JMP   SETMASK
0049 C3 FE 00            JMP   RECSTAT
004C C3 04 01            JMP   RECEIVE
004F C3 0E 01            JMP   SEND
0052            *
0052 C3 68 01            JMP   ADPWR
0055 C3 3A 01            JMP   SAMPLE
0058            *
0058 C3 2A 02            JMP   MEMPWR
005B C3 7C 01            JMP   MARSET
005E C3 AF 01            JMP   BANKSET
```

```
0061 C3 CF 01             JMP    MODESET
0064               *
0064 C3 67 02             JMP    SECOND
0067 C3 71 02             JMP    D5MS
006A               *
006A C3 DC 01             JMP    READ
006D C3 E2 01             JMP    WRITE
0070 C3 01 02             JMP    REWIND
0073 C3 17 02             JMP    MARGET
0076 C3 B1 00             JMP    SETVECT
0079 C3 C1 00             JMP    SETIO
007C               *
007C               * INITIALIZE THE BURST HARDWARE
007C               *
007C 3E 4C         BIOINIT MVI   A,NOTM+COUT  CONFIGURE I/O IN
007E D3 20                OUT    A8155  CASE THIS IS THE BREADBOARD
0080 3E 4F                MVI    A,NOTM+AOUT+BOUT+COUT
0082 D3 60                OUT    B8155
0084               *
0084 CD 43 02             CALL   RLYINIT  INIT RELAY CONTROL
0087 3E 50                MVI    A,SDOEN+RES75  SDO=0
0089 CD AA 00             CALL   SETMASK  AND 7.5 INITIALIZED
008C 97                   SUB    A      COMMAND STATUS=NOT READY
008D 32 03 10             STA    CMDSTAT
0090 32 08 10             STA    IOMODE  CMD-PENDING MODE OFF
0093 21 00 00             LXI    H,0
0096 CD B1 00             CALL   SETVECT  TURN OFF INTERUPTS
0099               *
0099 11 34 01             LXI    D,LPW1  PUT LOW POWER WAIT
009C 21 0E 10             LXI    H,LPWAIT  INTO MEMORY
009F 0E 06                MVI    C,LPWLEN
00A1               *
00A1 1A            COPY   LDAX   D
00A2 77                   MOV    M,A
00A3 13                   INX    D
00A4 23                   INX    H
00A5 0D                   DCR    C
00A6 C2 A1 00             JNZ    COPY
00A9 C9                   RET
00AA               *
00AA               * SET/READ PROCESSOR MASK
00AA               *
00AA 32 00 10      SETMASK STA   SDCOPY
00AD 30                   DB     SIM
00AE C9                   RET    .
00AF               *
00AF 20            GETMASK DB    RIM
00B0 C9                   RET    .
00B1               *
00B1 F3            SETVECT DI    .      DISABLE INTERRUPTS
```

```
00B2 22 04 10               SHLD   TIM1VECT  SET THE VECTOR
00B5 7C                     MOV    A,H      IF HL=0, DISABLE INTS
00B6 B5                     ORA    L
00B7 3E 0B                  MVI    A,MSE+DIS55+DIS65
00B9 CA BE 00               JZ     SV1
00BC 3E 09                  MVI    A,MSE+DIS55
00BE 30               SV1   DB     SIM
00BF FB                     EI     .        REENABLE INTERRUPTS
00C0 C9                     RET    .
00C1                  *
00C1 32 08 10         SETIO STA    IOMODE  SET COMMAND-PENDING MODE ON/OFF
00C4 C9                     RET    .
00C5                  *
00C5                  * SERIAL REQUEST FROM THE MAIN PROCESSOR
00C5                  *
00C5 F5               RST75 PUSH   PSW      SAVE REGISTERS USED
00C6 C5                     PUSH   B
00C7 D5                     PUSH   D
00C8 0E 10                  MVI    C,COMLEN  NUMBER OF BITS TO SHIFT
00CA 3E C0                  MVI    A,SDO+SDOEN   SDO<-1 (READY)
00CC 30                     DB     SIM
00CD 20               WTTM  DB     RIM      WAIT TILL SID=0 (START)
00CE 07                     RLC
00CF DA CD 00               JC     WTTM
00D2 00                     NOP    .        GIVE TM PROC SOME TIME
00D3                  *
00D3 3E 00            SH1   MVI    A,0      NOP OF 7 CYCLES
00D5 20                     DB     RIM      GET A BIT
00D6 07                     RLC
00D7 18                     DB     SLDE     SHIFT INTO DE
00D8 0D                     DCR    C        #BITS--
00D9 C2 D3 00               JNZ    SH1
00DC                  *
00DC EB                     XCHG   .        CMDREG=[DE]
00DD 22 01 10               SHLD   CMDREG
00E0 EB                     XCHG
00E1 3E 01                  MVI    A,1      CMDSTAT=READY
00E3 32 03 10               STA    CMDSTAT
00E6                  *
00E6 3A 00 10               LDA    SDCOPY  RESET SDO TO STATE PRIOR
00E9 E6 80                  ANI    SDO     TO INTERRUPT AND RESET THE
00EB F6 50                  ORI    SDOEN+RES75  7.5 INTERRUPT FLOP TOO.
00ED 30                     DB     SIM
00EE D1                     POP    D        RESTORE REGISTERS
00EF C1                     POP    B
00F0                  *
00F0 3A 08 10               LDA    IOMODE  IN CMD PENDING MODE, SET CARRY
00F3 0F                     RRC    .        WHEN RETURNING
00F4 DA FA 00               JC     CPMODE
00F7 F1                     POP    PSW      NORMAL MODE
```

```
00F8 FB              EI
00F9 C9              RET
00FA          !
00FA F1       CPMODE POP    PSW
00FB 37              STC
00FC FB              EI     .
00FD C9              RET    .
00FE          !
00FE          ! RETURN STATUS OF COMMUNICATIONS WITH TM PROCESSOR
00FE          !
00FE 21 03 10 RECSTAT LXI   H,CMDSTAT  RETURN ZERO IF NOT READY
0101 7E              MOV    A,M    AS WELL AS THE ADDRESS OF STAT
0102 B7              ORA    A
0103 C9              RET
0104          !
0104          ! RECEIVE DATA FROM THE TELEMETRY PROCESSOR
0104          !  ON EXIT: [HL]=DATA IF NOT ZERO, ELSE NOT READY
0104          !
0104 CD FE 00 RECEIVE CALL  RECSTAT  CHECK FOR STAT=READY
0107 C8              RZ     .
0108 36 00           MVI    M,0    STAT=NOT READY
010A 2A 01 10        LHLD   CMDREG  PICK UP THE COMMAND REGISTER
010D C9              RET    .
010E          !
010E          ! SEND [HL] TO THE TELEMETRY PROCESSOR
010E          !
010E E5       SEND   PUSH   H      SAVE MESSAGE OUT
010F C5              PUSH   B
0110 0E 10           MVI    C,COMLEN  C=#BITS TO SHIFT
0112 3E C0           MVI    A,SDO+SDOEN  SDO=1 (REQUEST)
0114 32 00 10        STA    SDCOPY
0117 30              DB     SIM
0118 F3              DI     .      IF NO INT, WE'VE GOT THE LINE.
0119          !
0119 CD 0E 10        CALL   LPWAIT
011C 3E 40           MVI    A,SDOEN  SDO=0 (START)
011E 30              DB     SIM
011F          !
011F 3E 80    SENDBIT MVI   A,SDOEN*2
0121 29              DAD    H
0122 1F              RAR
0123 30              DB     SIM
0124 0D              DCR    C
0125 C2 1F 01        JNZ    SENDBIT
0128          !
0128 C1              POP    B
0129 E1              POP    H      RESTORE HL AND DELAY
012A 3E 40           MVI    A,SDOEN  SDO=0 (NO MORE REQUEST)
012C 32 00 10        STA    SDCOPY
012F F6 10           ORI    RES75  AND RESET 7.5 INTERRUPT
```

```
0131 30                       DB     SIM
0132 FB                       EI     .
0133 C9                       RET    .
0134                   !
0134                   ! LOW POWER WAIT ROUTINE
0134                   ! (RUNS IN RAM AT LPWAIT)
0134                   !
0134 20                LPWI   DB     RIM
0135 07                       RLC
0136 D2 0E 10                 JNC    LPWAIT
0139 C9                       RET
013A                   LPWLEN EQU    $-LPWI
013A                   !
013A                   ! SAMPLE A QTY IN THE NORMAL FASHION
013A                   ! ON ENTRY: [A]=QTY TO SAMPLE
013A                   ! ON EXIT : [HL]=13 BIT SAMPLE (GAIN INCLUDED)
013A                   !
013A F5                SAMPLE PUSH   PSW     SAVE THE SAMPLE #
013B 87                       ADD    A       MOVE INTO THE LOW ADDRESS BITS
013C E6 FE                    ANI    MUXAD
013E 6F                       MOV    L,A
013F 26 30                    MVI    H,ADC/256
0141                   !
0141 3E 03                    MVI    A,3     CONFIGURE A/D FOR AUTO START MODE
0143 32 01 30                 STA    ADCTL
0146                   !
0146 7E                       MOV    A,M     READ THE A/D TO SET THE
0147 CD 61 01                 CALL   SMPDLA  GAIN DECISION AND SAMPLE/HOLD
014A                   !
014A 7E                       MOV    A,M     READ THE A/D TO CONVERT THE
014B CD 61 01                 CALL   SMPDLA  VALUE HELD BY THE SAMPLE/HOLD
014E                   !
014E 5E                       MOV    E,M     READ THE VALUE FROM THE A/D REGISTER
014F 2C                       INR    L
0150 66                       MOV    H,M
0151 6B                       MOV    L,E
0152                   !
0152 F1                       POP    PSW     RESTORE THE ORIGINAL QTY NUMBER
0153 FE 00                    CPI    KLYQTY
0155 C0                       RNZ    .
0156 29                       DAD    H       IF KELLEY, USE ONLY MS 8 BITS
0157 29                       DAD    H
0158 29                       DAD    H
0159 29                       DAD    H
015A 6C                       MOV    L,H
015B DB 23                    IN     KGAINS  THEN APPLY FOUR BITS
015D E6 0F                    ANI    0FH     FROM THE KELLEY PORT
015F 67                       MOV    H,A
0160 C9                       RET
0161                   !
```

```
0161 3E 02         SMPDLA MVI    A,2     DELAY A WHILE
0163 3D            SDLA   DCR    A
0164 C2 63 01             JNZ    SDLA
0167 C9                   RET
0168               *
0168               * A/D POWER CONTROL
0168               * ON ENTRY: CARRY=1 TO TURN ON, 0 TO TURN OFF.
0168               *
0168 DC 3A 02      ADPWR  CC     SRLY    IF ON, SET S/R RELAY
016B 3E 40                MVI    A,40H   PULSE THE 5V RELAY COIL
016D CD 58 02             CALL   PULSE
0170 3E 80                MVI    A,80H   AND THEN THE 12V RELAY
0172 CD 58 02             CALL   PULSE
0175 CD 48 02             CALL   RRLY
0178 3A 4C 30             LDA    ADC+GND  ADDRESS GROUND INPUT
017B C9                   RET
017C               *
017C               * SET THE MEMORY ADDRESS REGISTER
017C               * ON ENTRY: [AHL]=16 BIT ADDRESS TO SET
017C               *
017C C5            MARSET PUSH   B
017D E6 03                ANI    3       MASK TO 2 BITS
017F 47                   MOV    B,A
0180               *
0180 7C                   MOV    A,H     PICK UP 4 BITS FROM HL
0181 E6 F0                ANI    0F0H
0183 B0                   ORA    B       PUT EM TOGETHER
0184 07                   RLC    .       AND SHIFT TO 6 LSB'S
0185 07                   RLC
0186 07                   RLC
0187 07                   RLC
0188 F6 40                ORI    LOADADR  SET LOAD FLAG
018A CD A3 01             CALL   OUTHM   AND HIGH BITS
018D 2D                   DCR    L       SET LOW 12 BUT CORRECT
018E 22 FF CF             SHLD   LOMAR   FOR THE INCREMENT
0191 2C                   INR    L       TO THE LOW BYTE.
0192 3A 09 10             LDA    STBANK  RESTORE THE START BANK
0195 CD C0 01             CALL   SETSB   TO THE HIMAR LATCH.
0198               *
0198 29                   DAD    H       NOW RECORD THE MAR SETTING
0199 78                   MOV    A,B     IN A SHIFTED CONFIGURATION
019A 17                   RAL    .       (*2)
019B 32 0D 10             STA    MARREG+2
019E 22 0B 10             SHLD   MARREG
01A1 C1                   POP    B
01A2 C9                   RET
01A3               *
01A3 D5            OUTHM  PUSH   D
01A4 32 07 10             STA    MEMMODE  RECORD ALL HIMAR OUTPUTS
01A7 11 61 70             LXI    D,HIMAR*256+BBHIMAR  [DE]=BB+FLIGHT PORT #'S
```

```
01AA CD 86 02              CALL   DOUT
01AD D1                    POP    D
01AE C9                    RET
01AF               *
01AF               * SET BANK START AND END ADDRESSING
01AF               * ON ENTRY: [B]=START BANK # (0..5)
01AF               *           [C]=END BANK # (0..5)
01AF               *
01AF 79            BANKSET MOV    A,C    OUTPUT THE END BANK
01B0 E6 07                 ANI    7      MASKED TO PROPER BITS
01B2 32 0A 10              STA    ENBANK
01B5 EE 07                 XRI    7      AND COMPLEMENTED
01B7 D3 63                 OUT    ENDAD
01B9 32 06 10              STA    ENDCOPY
01BC               *
01BC 78                    MOV    A,B    MASK THE START ADDRESS INFO
01BD 32 09 10              STA    STBANK
01C0 87            SETSB   ADD    A      PUT IN THE 3 ZERO BITS
01C1 87                    ADD    A
01C2 87                    ADD    A
01C3 E6 3F         SETHM   ANI    UPPER6
01C5 4F                    MOV    C,A
01C6 3A 07 10              LDA    MEMMODE  DON'T EFFECT THE MEMMODE
01C9 E6 80                 ANI    AUTOWRITE
01CB B1                    ORA    C
01CC C3 A3 01              JMP    OUTHM
01CF               *
01CF               * SET/RESET THE MEMORY AUTOWRITE MODE
01CF               * ON ENTRY: [A]=1 FOR AUTOWRITE ELSE NO AUTOWRITE
01CF               *
01CF E6 01         MODESET ANI    1      USE ONLY THE LSB
01D1 0F                    RRC    .
01D2 47                    MOV    B,A    SAVE THE BIT
01D3 3A 07 10              LDA    MEMMODE  PICK UP THE CURRENT START
01D6 E6 3F                 ANI    UPPER6  BANK
01D8 B0                    ORA    B
01D9 C3 A3 01              JMP    OUTHM
01DC               *
01DC               * MEMORY FUNCTIONS
01DC               *
01DC 2A 00 F0      READ    LHLD   MEMRO
01DF C3 E5 01              JMP    INCMAR
01E2               *
01E2 22 00 80      WRITE   SHLD   MEM
01E5 E5            INCMAR  PUSH   H
01E6 2A 0B 10              LHLD   MARREG  INCREMENT THE MAR COPY
01E9 11 04 00              LXI    D,4    BY DOUBLE THE AMOUNT TO MAKE
01EC 19                    DAD    D      THE BANK # IN 1 BYTE.
01ED 22 0B 10              SHLD   MARREG
01F0 D2 FF 01              JNC    INCEX
```

```
01F3 21 0D 10            LXI   H,MARREG+2  IF MARBANK++ ==ENDBANK+1 VALUE
01F6 34                  INR   M
01F7 3A 0A 10            LDA   ENBANK  THEN REWIND
01FA 3C                  INR   A
01FB BE                  CMP   M
01FC CC 01 02            CZ    REWIND
01FF E1           INCEX  POP   H
0200 C9                  RET
0201              ;
0201 3A 09 10     REWIND LDA   STBANK  SET MAR TO BEGINNING OF BANK
0204 CD 0A 02            CALL  ADRBANK
0207 C3 7C 01            JMP   MARSET
020A              ;
020A 0F           ADRBANK RRC  .      CONVERT BANK INTO ADDRESS
020B C5                  PUSH  B
020C 47                  MOV   B,A
020D E6 80               ANI   80H
020F 67                  MOV   H,A
0210 2E 00               MVI   L,0
0212 78                  MOV   A,B
0213 E6 03               ANI   3
0215 C1                  POP   B
0216 C9                  RET
0217              ;
0217              ; READ-BACK MEMORY ADDRESS REGISTER
0217              ;
0217 3A 0D 10     MARGET LDA   MARREG+2  GET THE COPY
021A C5                  PUSH  B
021B B7                  ORA   A
021C 1F                  RAR   .      FROM THE RAM AND
021D 47                  MOV   B,A    DIVIDE BY 2
021E 2A 0B 10            LHLD  MARREG
0221 7C                  MOV   A,H
0222 1F                  RAR
0223 67                  MOV   H,A
0224 7D                  MOV   A,L
0225 1F                  RAR
0226 6F                  MOV   L,A
0227 78                  MOV   A,B
0228 C1                  POP   B
0229 C9                  RET
022A              ;
022A              ; MEMORY POWER CONTROL SECTION
022A              ; ON ENTRY: [A]= BANK NUMBER TO TURN ON OR OFF
022A              ;          CARRY=1 FOR ON, 0 FOR OFF
022A              ;
022A DC 3A 02     MEMPWR CC    SRLY   IF CRY, PREPARE SET/RESET RELAY
022D FE 06               CPI   6      IF ERRONEOUS BANK, SIMPLY RETURN
022F D0                  RNC   .
0230 CD 7B 02            CALL  UNARY  CONVERT TO UNARY
```

```
0233 7D                 MOV    A,L    SO THAT WE PULSE ONLY 1
0234 CD 58 02           CALL   PULSE  DO IT.
0237 C3 48 02           JMP    RRLY   AND GO TO LOW POWER AFTERWARD
023A               *
023A               * RELAY CONTROL SECTION
023A               *
023A F5            SRLY    PUSH   PSW
023B 3A 06 10              LDA    ENDCOPY  SET-RESET TO SET POSITION
023E F6 20                 ORI    SRRLY
0240 C3 4E 02              JMP    SRSET
0243               *
0243 3E FF         RLYINIT MVI    A,0FFH  RESET ALL RELAY COILS
0245 CD 61 02              CALL   OUTRELAYS
0248               *
0248 F5            RRLY    PUSH   PSW
0249 3A 06 10              LDA    ENDCOPY  RESET POSITION
024C E6 DF                 ANI    -1-SRRLY
024E D3 63         SRSET   OUT    RLYCTL
0250 32 06 10              STA    ENDCOPY
0253 CD 71 02              CALL   D5MS   DELAY FOR IT TO SETTLE OUT
0256 F1                    POP    PSW
0257 C9                    RET
0258               *
0258 2F            PULSE   CMA    .      OUTPUT IN COMPLEMENT FORM
0259 CD 61 02              CALL   OUTRELAYS
025C CD 71 02              CALL   D5MS   AND DELAY FOR THE FLIP
025F               *
025F 3E FF                 MVI    A,0FFH  RESET ALL COILS
0261 11 62 20      OUTRELAYS LXI  D,RELAYS*256+BBRELAYS
0264 C3 86 02              JMP    DOUT
0267               *
0267 06 C8         SECOND  MVI    B,200  DELAY 1 SECOND
0269 CD 71 02      SEC1    CALL   D5MS
026C 05                    DCR    B
026D C2 69 02              JNZ    SEC1
0270 C9                    RET
0271               *
0271 11 71 02      D5MS    LXI    D,5*3*1000/24  DELAY 5 MILLISECS AT 3MHZ
0274 1B            DELAY   DCX    D
0275 7B                    MOV    A,E
0276 B2                    ORA    D
0277 C2 74 02              JNZ    DELAY
027A C9                    RET
027B               *
027B               * CONVERT NUMBER IN A INTO UNARY IN HL
027B               *
027B 21 01 00      UNARY   LXI    H,1
027E E6 0F                 ANI    15
0280 C8            UNA1    RZ     .
0281 29                    DAD    H
```

```
0282 3D                     DCR    A
0283 C3 80 02               JMP    UNA1
0286                  *
0286                  * D OUTPUT.  THIS ROUTINE MAINTAINS COMPATIBILITY BETWEEN
0286                  *            THE BREADBOARD AND FLIGHT UNITS BY OUTPUTTING
0286                  *            THE ACCUM TO PORT(D) IF FLIGHT AND PORT(E) IF
0286                  *            THE BREADBOARD.
0286                  *
0286 F5               DOUT  PUSH   PSW     SAVE THE ACCUM
0287 3E 01                  MVI    A,FLIGHT  THE FLIGHT UNIT=1
0289 B7                     ORA    A
028A C2 8E 02               JNZ    DOFLGHT
028D 53                     MOV    D,E
028E F1               DOFLGHT POP  PSW     STORE THE DATA AT PORT(D)
028F 12                     STAX   D
0290 C9                     RET
0291 00          V          DB     256     END-OF-BIO MODULE
0292                  *
0292                  * VARIABLES FOR THE BURST DRIVERS
0292                  *
0292                        ORG    BIORAM
1000                  SDCOPY DS    1       SERIAL DATA COPY
1001                  CMDREG DS    2       COMMAND REGISTER
1003                  CMDSTAT DS   1       COMMAND STATUS (1=READY)
1004                  TIM1VECT DS  2       TIMER 1 VECTOR
1006                  ENDCOPY DS   1       COPY OF END ADDRESS
1007                  MEMMODE DS   1       MEMORY MODE REGISTER
1008                  IOMODE DS    1       CMD PENDING MODE
1009                  STBANK DS    1       START BANK OF MEMORY
100A                  ENBANK DS    1       END BANK
100B                  MARREG DS    3       RAM COPY OF MEMORY ADDRESS REG
100E                  LPWAIT DS    LPWLEN  LOW POWER WAIT ROUTINE
```

```
0000                 *
0000                 * CRRES FLIGHT SOFTWARE---BURST SAMPLING ROUTINES
0000                 * FILE : BSMP.A
0000                 *
0000                 PSW      EQU    6       PROCESSOR EQUATES
0000                 *
0000                 * ENTRY POINTS
0000                 *
0000                          ORG    BSMP
0540 C3 46 05                 JMP    BSMPINIT
0543 C3 63 05                 JMP    BSMPCMO
0546                 *
0546                 * INITIALIZE THE SAMPLING ROUTINES.
0546                 *
0546 CD C0 08        BSMPINIT CALL INIFMT  RESET ALL FORMATS
0549 97                       SUB    A       SET TO FORMAT(0)
054A CD C3 08                 CALL   SETFMT
054D 3E 0F                    MVI    A,15    SET TO HIGHEST RATE
054F CD 94 05                 CALL   SETRATE
0552 B7                       ORA    A       TURN OFF THE A/D POWER
0553 CD 52 00                 CALL   AOPWR
0556 97                       SUB    A       ZERO THE MODE
0557 32 D5 10                 STA    BMODE
055A 21 00 00                 LXI    H,0     ZERO THE MEMORY AODRESS
055D CD 5B 00                 CALL   MARSET
0560 21 45 B3                 LXI    H,0B345H  TURN ON BANKS 4 AND 5
0563                 *
0563 7C              BSMPCMO MOV     A,H     1F COMMANO BEGINS WITH
0564 E6 F0                    AN1    0F0H    0BXXXH, THEN WE'RE INTERESTED
0566 FE B0                    CPI    0B0H
0568 37                       STC    .
0569 C0                       RNZ    .
056A                 *
056A 7C                       MOV    A,H     CALCULATE WHICH OF THE SMP COMMANDS
056B D6 B0                    SUI    0B0H    THIS ONE IS.
056D FE 0B                    CPI    NCMDS   1F GREATER THAN KNOWN CMD
056F 3F                       CMC    .       RETURN(CARRY)
0570 D8                       RC     .
0571                 *
0571 EB                       XCHG   .       SAVE THE ORIGINAL COMMAND VALUE
0572 87                       ADD    A
0573 21 7E 05                 LXI    H,CMDTB  AND REFERENCE THE TABLE
0576 CD 51 08                 CALL   REF     FOR THE ROUTINES AODRESS
0579 CD 56 08                 CALL   LOOHL
057C 7B                       MOV    A,E     A=LOW VALUE OF COMMAND
057D E9                       PCHL
057E                 *
057E 94 05           CMDTB    DW     SETRATE  BFREQ
0580 C3 08                    DW     SETFMT  BFMT
```

```
0582 C6 08             DW     ADDFMT  BQTY
0584 9A 05             DW     BSELECT  BANKS
0586 E0 05             DW     GO     BGO
0588 2E 06             DW     STOP   BSTOP
058A 26 06             DW     PAUSE  BPAUSE
058C 19 06             DW     CONT   BCONTINUE
058E 46 06             DW     PLAY   BPLAY
0590 00 00             DW     0      BRESET
0592 D9 05             DW     MODSET  BMODE
0594              NCMDS  EQU    $-CMDTB/2  NUMBER OF COMMANDS
0594              *
0594              * SET THE RATE OF THE BURST
0594              *
0594 E6 0F        SETRATE ANI   15
0596 32 C4 10          STA    FREQ   SET NEW DESIRED FREQ
0599 C9                RET
059A              *
059A              * SELECT THE MEMORY BANKS TO USE
059A              *
059A 5F           BSELECT MOV   E,A
059B 0F                RRC    .      B=START BANK
059C 0F                RRC
059D 0F                RRC
059E 0F                RRC
059F E6 07             ANI    7
05A1 47                MOV    B,A
05A2 32 C6 10          STA    STBANK
05A5              *
05A5 7B                MOV    A,E    C=END BANK
05A6 E6 07             ANI    7
05A8 4F                MOV    C,A
05A9 32 C7 10          STA    ENBANK
05AC CD 5E 00          CALL   BANKSET
05AF              *
05AF              * MEMORY POWER CONTROLLER.  TURNS ON STBANK TO ENBANK
05AF              * AND TURNS OFF ANY OTHERS.
05AF              *
05AF 21 05 00     MEMPC  LXI    H,005H  TURN OFF BANKS 0 TO 5
05B2 06 00             MVI    B,0
05B4 CD C8 05          CALL   ONOFF
05B7              *
05B7 3A C6 10          LDA    STBANK  TURN ON FROM START TO END
05BA E6 07             ANI    7
05BC 67                MOV    H,A
05BD 3A C7 10          LDA    ENBANK
05C0 6F                MOV    L,A
05C1 06 01             MVI    B,1
05C3 CD C8 05          CALL   ONOFF
05C6 B7                ORA    A      RETURN(NO CARRY)
05C7 C9                RET
```

```
05C8              *
05C8              ONOFF  EQU   $
05C8 7D                  MOV   A,L    IF START > END, RETURN
05C9 BC                  CMP   H
05CA D8                  RC    .
05CB 78                  MOV   A,B    SET FOR ON OR OFF
05CC 0F                  RRC
05CD 7C                  MOV   A,H    SELECT BANK
05CE E5                  PUSH  H
05CF C5                  PUSH  B
05D0 CD 58 00            CALL  MEMPWR
05D3 C1                  POP   B
05D4 E1                  POP   H
05D5 24                  INR   H
05D6 C3 C8 05            JMP   ONOFF
05D9              *
05D9 32 D5 10     MODSET STA   BMODE
05DC 0F                  RRC   .
05DD C3 52 00            JMP   ADPWR
05E0              *
05E0              * BURST START/STOP/PAUSE/CONT CONTROLS
05E0              *
05E0              BOGUS  EQU   1800H  BOGUS SAMPLE (SATURATED HIGH GAIN)
05E0              SMARK  EQU   BOGUS+0A00AH  START MARKER
05E0              PMARK  EQU   BOGUS+0C00CH  PAUSE MARKER
05E0              EMARK  EQU   BOGUS+0E00EH  STOP MARKER
05E0              *
05E0 3A D5 10     GO     LDA   BMODE  TURN ON THE A/D IN
05E3 0F                  RRC   .
05E4 3F                  CMC
05E5 DC 52 00            CC    ADPWR  CASE IT IS OFF
05E8 CD E4 06            CALL  CMPROG  COMPILE THE SAMPLE PROGRAM
05EB 3A C5 10            LDA   REALF  SEND THE REAL FREQ BACK
05EE CD CF 06            CALL  SENDA
05F1 CD 65 07            CALL  GETDUR  CALCULATE DURATION OF BURST
05F4 22 C0 10            SHLD  DURATION
05F7 D5                  PUSH  D
05F8 CD 4F 00            CALL  SEND
05FB E1                  POP   H
05FC 22 C2 10            SHLD  DURATION+2
05FF CD 4F 00            CALL  SEND
0602              *
0602 CD 70 00     WRTST  CALL  REWIND  REPOSITION THE DIGITAL TAPE
0605 21 0A B8            LXI   H,SMARK  WRITE('START-MARK') INTO MEMORY
0608 CD 6D 00            CALL  WRITE  IN CASE OF SHORT BURST
060B              *
060B CD 70 00            CALL  REWIND  VERIFY THE MARK
060E CD 6A 00            CALL  READ
0611 11 0A B8            LXI   D,SMARK
0614 CD 58 08            CALL  EQ16
```

```
0617 37                STC
0618 C0                RNZ   .
0619             *
0619 3E 01       CONT  MVI   A,1    SHOW THE FILE OPEN
061B 32 C8 10          STA   FILESTAT
061E CD 1C 07          CALL  GETDLA  C=DELAY VALUE
0621 CD D6 10          CALL  SMPAREA  AND BEGIN
0624 97                SUB   A      RETURN(0)
0625 C9                RET
0626             *
0626 21 0C D8    PAUSE LXI   H,PMARK  WRITE('PAUSE-MARK')
0629 CD 6D 00          CALL  WRITE
062C 97                SUB   A      RETURN(0)
062D C9                RET
062E             *
062E 3A D5 10    STOP  LDA   BMODE  TURN OFF THE A/D CONVERTER
0631 0F                RRC   .      UNLESS COMMANDED TO STAY ON
0632 D4 52 00          CNC   ADPWR
0635 CD 9F 07          CALL  FCLOSE
0638 21 4B 4F          LXI   H,'OK'
063B D2 41 06          JNC   STOP1
063E 21 4F 4E          LXI   H,'NO'
0641 CD 4F 00    STOP1 CALL  SEND
0644 B7                ORA   A
0645 C9                RET
0646             *
0646             * BEGIN PLAYBACK TO THE MASTER COMPUTER
0646             *
0646 3A C8 10    PLAY  LDA   FILESTAT  IF FILE OPEN, CLOSE IT
0649 FE 01             CPI   1      TO GET THE FILE PARAMETERS
064B CC 9F 07          CZ    FCLOSE
064E             *
064E CD 8A 06          CALL  PHEAD  PLAY HEADER
0651 3A C8 10          LDA   FILESTAT  IF AN ERROR IN FILE, QUIT
0654 FE 02             CPI   2
0656 C8                RZ    .
0657 2A C9 10          LHLD  STADR  POSITION HEAD TO START MARK
065A 3A CB 10          LDA   STADR+2  OR THE FIRST VALID RECORD
065D CD 5B 00          CALL  MARSET
0660             *
0660 CD 49 00    PLOOP CALL  RECSTAT  IF COMMAND READY, QUIT THIS
0663 C2 81 06          JNZ   PLAYX
0666 CD 6A 00          CALL  READ   ELSE PLAY OUT DATA UNTIL END-MARK
0669 CD 4F 00          CALL  SEND
066C             *
066C CD 73 00          CALL  MARGET  IF THE ADDRESS MATCHES
066F EB                XCHG  .      WHERE WE FOUND THE ENDMARK
0670 47                MOV   B,A
0671 2A CC 10          LHLD  ENADR  QUIT
0674 CD 5B 08          CALL  EQ16
```

```
0677 C2 60 06          JNZ    PLOOP
067A 3A CE 10          LDA    ENADR+2
067D BB                CMP    B
067E C2 60 06          JNZ    PLOOP
0681             *
0681 97          PLAYX SUB    A      MAR = 0
0682 21 00 00          LXI    H,0
0685 CD 5B 00          CALL   MARSET
0688 B7                ORA    A      RETURN(0)
0689 C9                RET
068A             *
068A             * PLAY HEADER
068A             *
068A 3E B2       PHEAD MVI    A,0B2H  START HEADER PLAYBACK
068C CD CF 06          CALL   SENDA
068F 3E A1             MVI    A,0A1H  FORMAT 1
0691 CD CF 06          CALL   SENDA
0694 3A C5 10          LDA    REALF  REAL FREQUENCY CODE
0697 CD CF 06          CALL   SENDA
069A             *
069A 2A C9 10          LHLD   STADR  START ADDRESS
069D CD 4F 00          CALL   SEND
06A0 2A CA 10          LHLD   STADR+1
06A3 CD D8 06          CALL   DIV16
06A6 CD 4F 00          CALL   SEND
06A9             *
06A9 2A CC 10          LHLD   ENADR  PAGE OF END ADDRESS
06AC CD 4F 00          CALL   SEND
06AF 2A CD 10          LHLD   ENADR+1
06B2 CD D8 06          CALL   DIV16
06B5 CD 4F 00          CALL   SEND
06B8             *
06B8 CD C9 0B          CALL   ADRFMT  [DE]->LIST
06BB EB                XCHG   .
06BC CD CC 0B          CALL   LNGFMT  C=LENGTH OF FMT
06BF 4F                MOV    C,A
06C0 CD CF 06          CALL   SENDA  SEND LENGTH
06C3 79          SQTYLP MOV   A,C    IF NO QTYS LEFT, QUIT
06C4 B7                ORA    A
06C5 C8                RZ     .
06C6 1A                LDAX   D      SEND NEXT QTY DESCRIPTOR
06C7 CD CF 06          CALL   SENDA
06CA 13                INX    D
06CB 0D                DCR    C
06CC C3 C3 06          JMP    SQTYLP
06CF             *
06CF 6F          SENDA MOV    L,A
06D0 26 00             MVI    H,0
06D2 C3 4F 00          JMP    SEND
06D5 C3 CB 05          JMP    ONOFF
```

```
06D8            *
06D8 06 04      DIV16  MVI   B,4     SHIFT HL RIGHT 4 BITS
06DA 97                SUB   A
06DB 29         DV1    DAD   H
06DC 17                RAL   .
06DD 05                DCR   B
06DE C2 DB 06          JNZ   DV1
06E1 6C                MOV   L,H
06E2 67                MOV   H,A
06E3 C9                RET
06E4            *
06E4            * COMPILE THE SAMPLING LIST INTO A PROGRAM
06E4            *
06E4 CD 0B 07   CMPROG CALL  CHKFREQ  CHECK FREQUENCY
06E7 41                MOV   B,C     SAVE THE DELAY CODE
06E8 CD CC 08          CALL  LNGFMT  C=LENGTH OF THE FORMAT
06EB 4F                MOV   C,A
06EC CD C9 08          CALL  ADRFMT  [DE]->LIST OF QTYS
06EF EB                XCHG  .
06F0 21 D6 10          LXI   H,SMPAREA  [HL]->AREA TO PUT PROGRAM
06F3            *
06F3 3A C5 10          LDA   REALF   IF THE FREQUENCY IS IN THE
06F6 FE 08             CPI   INTYPE  INTERRUPT TIMES, GO
06F8 D2 00 07          JNC   INTCMP
06FB 06 02             MVI   B,2
06FD C3 40 0A          JMP   COMPILE
0700            *
0700 78         INTCMP MOV   A,B     GET THE DELAY CODE
0701 B7                ORA   A       IF ZERO DELAY, REQUEST THIS
0702 06 00             MVI   B,0     FROM THE COMPILER
0704 CA 40 0A          JZ    COMPILE
0707 04                INR   B       IF SOME DELAY, ASK FOR IT
0708 C3 40 0A          JMP   COMPILE
070B            *
070B            * CHECK THAT THE REQUESTED FREQUENCY IS POSSIBLE
070B            * SET REALF AT WHATEVER IS THE REAL FREQUENCY
070B            *
070B 3A C4 10   CHKFREQ LDA  FREQ    ASSUME THE REAL RATE IS
070E 32 C5 10          STA   REALF   THE REQUESTED RATE.
0711            *
0711 CD 1C 07   CHKRT  CALL  GETDLA  C=DELAY VALUE
0714 D0                RNC   .       IF VALID, RETURN
0715 21 C5 10          LXI   H,REALF  ELSE DECREASE THE FREQUENCY CODE
0718 35                DCR   M       AND TRY AGAIN
0719 C3 11 07          JMP   CHKRT
071C            *
071C 3A C5 10   GETDLA LDA   REALF   GET THE PROPER DELAY VALUE
071F 21 35 07          LXI   H,DLATBL  TO REGULATE SPEED
0722 CD 51 08          CALL  REF
0725 4E                MOV   C,M
```

```
0726           *
0726 3A C5 10         LDA    REALF  IF < INTERRUPT TYPE CODE
0729 FE 08            CPI    INTYPE  THEN RETURN(NO CRY)
072B 3F               CMC
072C D0               RNC
072D           *
072D CD CC 08         CALL   LNGFMT  ELSE SUBTRACT THE LENGTH
0730 47               MOV    B,A    OF THE LIST FROM THE DELAY
0731 79               MOV    A,C    SINCE SAMPLING ALWAYS DELAYS
0732 90               SUB    B      ITSELF.
0733 4F               MOV    C,A
0734 C9               RET    .
0735           *
0735 C8        DLATBL DB     200    10 HZ  CODES 0-7
0736 64               DB     100    20  (ALL INTERRUPT TIME-BASED)
0737 28               DB     40     50
0738 14               DB     20     100
0739 0A               DB     10     200
073A 04               DB     4      500
073B 02               DB     2      1000
073C 01               DB     1      2000
073D           *
073D           INTYPE EQU    $-DLATBL  ALL CODES BELOW ARE INTERRUPTS
073D 14               DB     20     CODES 8-15 (CPU CYCLE COUNTING)
073E 0A               DB     10
073F 06               DB     6
0740 04               DB     4
0741 03               DB     3
0742 02               DB     2
0743 01               DB     1
0744 00               DB     0
0745           FRMAX  EQU    $-DLATBL-2
0745           *
0745 0A 00     FRQTB  DW     10     FREQUENCIES
0747 14 00            DW     20
0749 32 00            DW     50
074B 64 00            DW     100
074D C8 00            DW     200
074F F4 01            DW     500
0751 E8 03            DW     1000
0753 D0 07            DW     2000
0755 A0 0B            DW     2976
0757 40 17            DW     5952
0759 C1 26            DW     9921
075B 21 3A            DW     14881
075D 81 4D            DW     19841
075F 42 74            DW     29762
0761 84 E8            DW     59524
0763 84 E8            DW     59524
0765           *
```

```
0765                * CALCULATE THE LENGTH OF TIME TO FILL MEMORY
0765                *
0765 3A C5 10       GETDUR LDA    REALF  REFERENCE FRQTB TO GET
0768 87                    ADD    A      THE #SAMPLES SETS/SECOND
0769 21 45 07              LXI    H,FRQTB
076C CD 51 08              CALL   REF
076F CD 56 08              CALL   LODHL
0772 E5                    PUSH   H
0773 CD CC 08              CALL   LNGFMT  GET THE #QTYS IN THE LIST
0776 D1                    POP    D
0777 CD 24 0C              CALL   MU21   AND MULTIPLY TO GET TOTAL
077A 16 00                 MVI    D,0    INPUT RATE TO MEMORY.
077C 5F                    MOV    E,A
077D CD 18 0C              CALL   FLT32  CONVERT TO FLOATING POINT
0780                *
0780 21 CF 10              LXI    H,TRATE  AND SAVE THE TOTAL RATE
0783 CD 03 0C              CALL   STOFP
0786                *
0786 3A C7 10              LDA    ENBANK  CALCULATE # BANKS
0789 21 C6 10              LXI    H,STBANK  TIMES 16K WORDS * 1000
078C 96                    SUB    M      WHICH IS NBANKS * 24 BITS
078D 3C                    INR    A
078E 57                    MOV    D,A
078F 1E 00                 MVI    E,0
0791 26 00                 MVI    H,0
0793 CD 18 0C              CALL   FLT32
0796                *
0796 21 CF 10              LXI    H,TRATE  NOW DIVIDE BY THE INPUT RATE
0799 CD 09 0C              CALL   FDIV   AND RETURN (DEHL) MILLISECONDS
079C C3 1B 0C              JMP    FIX32
079F                *
079F                * CLOSE THE FILE IN THE BURST MEMORY
079F                *
079F 21 0E F8       FCLOSE LXI    H,EMARK  WRITE("END-MARK")
07A2 CD 6D 00              CALL   WRITE
07A5                *
07A5 CD 70 00              CALL   REWIND  REPOSITION TAPE
07A8 11 0E F8              LXI    D,EMARK  AND FIND ITS ADDRESS
07AB CD 16 08              CALL   SEARCH
07AE D8                    RC     .      IF ERROR, RETURN NOW.
07AF                *
07AF CD 73 00              CALL   MARGET  ELSE FOUND IT.
07B2 22 CC 10              SHLD   ENADR  SAVE ITS ADDRESS
07B5 32 CE 10              STA    ENADR+2
07B8                *
07B8 CD 70 00              CALL   REWIND  REPOSITION TAPE
07BB CD 6A 00              CALL   READ   READ THE FIRST ENTRY
07BE 11 0A B8              LXI    D,SMARK  LOOKING FOR THE START MARK
07C1 CD 5B 08              CALL   EQ16   IF THERE, IT WAS A SHORT BURST
07C4 C2 D5 07              JNZ    FSLONG
```

```
07C7             *
07C7 CD 73 00    FENDSET CALL  MARGET  GET THE TAPE POSITION
07CA 22 C9 10            SHLD  STADR  AND SAVE FOR PLAYBACK
07CD 32 CB 10            STA   STADR+2
07D0 97                  SUB   A      SHOW THE BURST FILE CLOSED
07D1 32 C8 10            STA   FILESTAT
07D4 C9                  RET   .
07D5             *
07D5 2A CC 10    FSLONG LHLD   ENADR  START AT THE ENDMARKER
07D8 3A CE 10            LDA   ENADR+2
07DB CD 5B 00            CALL  MARSET
07DE             *
07DE CD CC 08            CALL  LNGFMT  C=LENGTH OF RECORD
07E1 4F                  MOV   C,A
07E2             *
07E2 CD 49 00    FSNREC CALL   RECSTAT  IF COMMAND READY, QUIT
07E5 C0                  RNZ   .
07E6 41                  MOV   B,C    START AT RECORD COUNT
07E7 CD 6A 00    FSIREC CALL   READ   READ NEXT VALUE
07EA 11 0C D8            LXI   D,PMARK  IF A PAUSE MARK
07ED CD 5B 08            CALL  EQ16   OR AN END MARK, QUIT.
07F0 CA 03 08            JZ    FSFND
07F3 11 0E F8            LXI   D,EMARK
07F6 CD 5B 08            CALL  EQ16
07F9 CA 03 08            JZ    FSFND
07FC 05                  DCR   B      ELSE CONTINUE WITHIN RECORD
07FD C2 E7 07            JNZ   FSIREC
0800 C3 E2 07            JMP   FSNREC  OR GET A NEW RECORD
0803             *
0803 79          FSFND  MOV    A,C    COMPUTE PARTIAL RECORD
0804 90                  SUB   B      LENGTH IN ORDER TO SKIP IT
0805 F5                  PUSH  PSW
0806 2A CC 10            LHLD  ENADR  POSITION TAPE TO ENDMARK
0809 3A CE 10            LDA   ENADR+2
080C CD 5B 00            CALL  MARSET
080F F1                  POP   PSW
0810 C4 47 08            CNZ   SKIP   SKIP(A) WORDS IF A!=0.
0813 C3 C7 07            JMP   FENDSET  SET FILE END TO CLOSE
0816             *
0816             * TAPE-LIKE FUNCTIONS
0816             *
0816 3E 03       SEARCH MVI    A,3    MAXCNT = 30000H
0818 32 D4 10            STA   MAXCNT+2  (*WORST CASE SEARCH LENGTH)
081B 21 00 00            LXI   H,0
081E 22 D2 10            SHLD  MAXCNT
0821             *
0821 CD 49 00    SEAR1  CALL   RECSTAT  IF COMMAND RECEIVED, QUIT
0824 37                  STC   .
0825 C0                  RNZ   .
0826 D5          SEAR2  PUSH   D
```

```
0827 CD 6A 00             CALL   READ   READ A VALUE THERE
082A D1                   POP    D
082B CD 5B 08             CALL   EQ16   IF SAME AS WE'RE LOOKING FOR
082E C8                   RZ     .      QUIT
082F                 *
082F 21 D2 10             LXI    H,MAXCNT  STOP COUNTING AFTER
0832 35                   DCR    M      THE MAX COUNT
0833 C2 26 06             JNZ    SEAR2
0836 23                   INX    H
0837 35                   DCR    M
0838 C2 21 08             JNZ    SEAR1
083B 23                   INX    H
083C 35                   DCR    M
083D C2 21 08             JNZ    SEAR1
0840 3E 02                MVI    A,2    ERROR IN FILE
0842 32 C8 10             STA    FILESTAT
0845 37                   STC
0846 C9                   RET
0847                 *
0847 F5              SKIP PUSH   PSW    SKIP # RECORDS
0848 CD 6A 00             CALL   READ
084B F1                   POP    PSW
084C 3D                   DCR    A
084D C2 47 08             JNZ    SKIP
0850 C9                   RET    .
0851                 *
0851 85              REF  ADD    L      [HL]=[HL]+A
0852 6F                   MOV    L,A
0853 D0                   RNC
0854 24                   INR    H
0855 C9                   RET
0856                 *
0856 7E              LODHL MOV   A,M    [HL]=MEM[HL]
0857 23                   INX    H
0858 66                   MOV    H,M
0859 6F                   MOV    L,A
085A C9                   RET
085B                 *
085B 7D              EQ16 MOV    A,L    COMPARE HL AND DE FOR =
085C BB                   CMP    E
085D C0                   RNZ
085E 7C                   MOV    A,H
085F BA                   CMP    D
0860 C9                   RET    .
0861                 *
0861 7D              MARK MOV    A,L
0862 D3 00                OUT    0
0864 7C                   MOV    A,H
0865 D3 01                OUT    1
0867 C9                   RET
```

```
0868 00        V  0500         DB     256    END-OF-MODULE
0869              0501 *
0869              0502 * VARIABLES
0869              0503 *
0869              0504         ORG    BSMPRAM
10C0              0505 DURATION DS    4      DURATION OF BURST IN MSEC
10C4              0506 FREQ    DS     1      DESIRED FREQUENCY
10C5              0507 REALF   DS     1      REAL FREQUENCY OF BURST USED
10C6              0508 STBANK  DS     1      START BANK #
10C7              0509 ENBANK  DS     1      END BANK #
10C8              0510 *
10C8              0511 FILESTAT DS    1      FILE STATUS (1=OPEN)
10C9              0512 STADR   DS     3      START OF BURST
10CC              0513 ENADR   DS     3      END OF BURST
10CF              0514 TRATE   DS     3      TOTAL INPUT RATE TO MEMORY
10D2              0515 MAXCNT  DS     3      MAXIMUM # SAMPLES TO SEARCH
10D5              0516 BMODE   DS     1      INTERNAL MODE (1=KEEP A/D ON)
10D6              0517 SMPAREA EQU    $      SAMPLE LIST AREA
10D6              0518 *
10D6              0519 * EXTERNAL MODULE DEFINITIONS
10D6              0520 *
10D6              0521         ORG    BIO
0040              0522 BIOINIT DS     3
0043              0523 GETMASK DS     3
0046              0524 SETMASK DS     3
0049              0525 RECSTAT DS     3
004C              0526 RECEIVE DS     3
004F              0527 SEND    DS     3
0052              0528 ADPWR   DS     3
0055              0529 SAMPLE  DS     3
0058              0530 MEMPWR  DS     3
005B              0531 MARSET  DS     3
005E              0532 BANKSET DS     3
0061              0533 MODESET DS     3
0064              0534 SECOND  DS     3
0067              0535 D5MS    DS     3
006A              0536 *
006A              0537 READ    DS     3
006D              0538 WRITE   DS     3
0070              0539 REWIND  DS     3
0073              0540 MARGET  DS     3
0076              0541 SETVECT DS     3
0079              0542 *
0079              0543         ORG    BCMP
0A40              0544 COMPILE DS     3
0A43              0545 *
0A43              0546         ORG    BFMT
08C0              0547 INIFMT  DS     3
08C3              0548 SETFMT  DS     3
08C6              0549 ADDFMT  DS     3
```

```
08C9            0550 ADRFMT DS     3
08CC            0551 LNGFMT DS     3
08CF            0552 ENDFMT DS     3
08D2            0553 *
08D2            0554        ORG    FFP
0C00            0555 LODFP  DS     3
0C03            0556 STOFP  DS     3
0C06            0557 FMUL   DS     3
0C09            0558 FDIV   DS     3
0C0C            0559 FADD   DS     3
0C0F            0560 FSUB   DS     3
0C12            0561 FCMP   DS     3
0C15            0562 FNEG   DS     3
0C18            0563 FLT32  DS     3
0C1B            0564 FIX32  DS     3
0C1E            0565 FSQUA  DS     3
0C21            0566 FSQRT  DS     3
0C24            0567 MU21   DS     3
```

```
0000               *
0000               * CRRES FLIGHT SOFTWARE---BURST PROGRAM LOADER
0000               * FILE : BLD.A
0000               *
0000               BLDCODE EQU   0BCH
0000               *
0000                       ORG   BLD
0440 C3 46 04              JMP   BLDINIT
0443 C3 4D 04              JMP   BLDCMD
0446               *
0446 21 02 12      BLDINIT LXI   H,USER  POINT THE ADR REGISTER
0449 22 00 12              SHLD  ADR     TO THE USER LOADING AREA
044C C9                    RET   .
044D               *
044D 7C            BLDCMD  MOV   A,H     CHECK FOR WHICH MEM LOAD CMD
044E E6 FC                 ANI   0FCH
0450 FE BC                 CPI   BLDCODE
0452 37                    STC
0453 C0                    RNZ   .
0454 EB                    XCHG
0455 7A                    MOV   A,D     GET THE COMMAND AGAIN
0456 D6 BC                 SUI   BLDCODE  REMOVE THE BIAS
0458 CA 69 04              JZ    SADRL   AND COUNT OFF EACH NUMBER
045B 3D                    DCR   A
045C CA 6E 04              JZ    SADRH
045F 3D                    DCR   A
0460 CA 73 04              JZ    LOAD
0463 3D                    DCR   A
0464 CA 7C 04              JZ    JUMP
0467 37                    STC   . IF UNKNOWN, RETURN(CRY)
0468 C9                    RET
0469               *
0469 7B            SADRL   MOV   A,E     SET LOW ADDRESS
046A 32 00 12              STA   ADR
046D C9                    RET
046E               *
046E 7B            SADRH   MOV   A,E     SET HIGH ADDRESS
046F 32 01 12              STA   ADR+1
0472 C9                    RET   .
0473               *
0473 2A 00 12      LOAD    LHLD  ADR     MEM[ADR++] = VALUE
0476 73                    MOV   M,E
0477 23                    INX   H
0478 22 00 12              SHLD  ADR
047B C9                    RET   .
047C               *
047C 3A 02 12      JUMP    LDA   USER    EXECUTE USER PROGRAM
047F FE AA                 CPI   0AAH    CHECK CODE TO VERIFY PROGRAM THERE
0481 C0                    RNZ   .       IF NOT RIGHT, SIGNAL ERROR
```

```
0482 97            0050        SUB   A      RESET THE CODE
0483 32 02 12      0051        STA   USER
0486 C3 03 12      0052        JMP   USER+1
0489               0053 *
0489               0054 * VARIABLES
0489               0055 *
0489               0056        ORG   BLDRAM
1200               0057 ADR    DS    2      USER LOAD ADDRESS
1202               0058 USER   DS    1024   USER PROGRAM LOADING AREA
```

```
0000            *
0000            * CRRES FLIGHT SOFTWARE---BURST FORMAT ROUTINES
0000            * WRITTEN BY PETER R HARVEY
0000            * FILE : BFMT.A
0000            *
0000            PSW    EQU    6
0000            EOL    EQU    0F0H   END OF LIST INDICATOR
0000            *
0000            * MULTIPLEXOR QUANTITIES
0000            *
0000            BZFAST EQU    0
0000            BXFAST EQU    1
0000            BYFAST EQU    2
0000            BV3    EQU    3
0000            BV4    EQU    4
0000            BV34   EQU    5
0000            BV34AC EQU    6
0000            BV1SC  EQU    7
0000            BVI2AC EQU    8
0000            BV2    EQU    9
0000            BV1    EQU    10
0000            BV12   EQU    11
0000            BDIRECT EQU   12
0000            BAGCU  EQU    I3
0000            BGUARD EQU    14
0000            BSTUB  EQU    15
0000            TESTQTY EQU   03FH
0000            *
0000                   ORG    BFMT
08C0 C3 DD 08          JMP    INIFMT
08C3 C3 D2 08          JMP    SETFMT
08C6 C3 24 09          JMP    ADDFMT
08C9 C3 F3 08          JMP    ADRFMT
08CC C3 1A 09          JMP    LNGFMT
08CF C3 0D 09          JMP    ENDFMT
08D2            *
08D2            * SET THE CURRENT FORMAT NUMBER
08D2            * ON ENTRY: A IS THE FORMAT TO USE
08D2            *
08D2 E6 0F      SETFMT ANI    I5
08D4 32 30 I0          STA    CURFMT
08D7 3E FF             MVI    A,-I   SET TO DELETE ON IST ADDFMT
08D9 32 31 I0          STA    INSFLAG
08DC C9                RET
08DD            *
08DD            * INIT ALL RAM FORMATS
08DD            *
08DD 21 33 10   INIFMT LXI    H,RAMLIST
08E0 II A4 09          LXI    D,RAMDEF
```

```
08E3 0E 06          MVI   C,RAMDX-RAMDEF+1
08E5           *
08E5 1A        IL1  LDAX  D      PUT THE END MARKER INTO MEM
08E6 13             INX   D
08E7 77             MOV   M,A
08E8 23             INX   H
08E9 0D             DCR   C
08EA C2 E5 08       JNZ   IL1
08ED 2D             DCR   L
08EE 7D             MOV   A,L    RECORD THE END OF MEMORY USED
08EF 32 32 10       STA   RLEND
08F2 C9             RET
08F3           *
08F3           * ADDRESS THE CURRENT FORMAT
08F3           * RETURNS [HL]=ADDRESS OF CURRENT FORMAT
08F3           *
08F3 3A 30 10  ADRFMT LDA  CURFMT  IF LIST# <10, THEN ITS A
08F6 FE 0A          CPI   10     LIST CONTAINED IN ROM
08F8 21 7C 09       LXI   H,ROMLIST
08FB DA 03 09       JC    AL1
08FE 21 33 10       LXI   H,RAMLIST  ELSE ITS A RAM LIST
0901 D6 0A          SUI   10
0903 B7        AL1  ORA   A
0904 C8             RZ
0905 CD 0D 09       CALL  ENDFMT  FIND THE END OF THE LIST
0908 23             INX   H      STEP OVER THE END MARKER
0909 3D             DCR   A      AND REPEAT TILL DONE
090A C3 03 09       JMP   AL1
090D           *
090D F5        ENDFMT PUSH PSW
090E 7E        EL1  MOV   A,M    SEARCH TILL THE END MARKER
090F FE F0          CPI   EOL
0911 CA 18 09       JZ    ELX
0914 23             INX   H
0915 C3 0E 09       JMP   EL1
0918 F1        ELX  POP   PSW
0919 C9             RET
091A           *
091A           * OBTAIN THE LENGTH OF THE CURRENT LIST
091A           *
091A CD F3 08  LNGFMT CALL ADRFMT  GET THE START ADDRESS
091D 5D             MOV   E,L
091E CD 0D 09       CALL  ENDFMT  AND THE ADDRESS OF THE ENDMARK
0921 7D             MOV   A,L
0922 93             SUB   E      RETURN LENGTH OF LIST
0923 C9             RET
0924           *
0924           * ADD A QTY TO THE CURRENT FORMAT
0924           * ON ENTRY: A HOLDS THE QTY
0924           *
```

```
0924 4F               ADDFMT MOV    C,A    ADD QTY TO CURRENT LIST
0925 3A 30 10                LDA    CURFMT  IF ROM FORMAT, QUIT
0928 FE 0A                   CPI    10
092A D8                      RC
092B                  *
092B 21 31 10                LXI    H,INSFLAG  IF 1ST ADD, DELETE ALL THE REST
092E 34                      INR    M
092F C2 3B 09                JNZ    ADEND
0932 CD F3 08                CALL   ADRFMT  HL->START OF THIS FORMAT
0935 CD 60 09         ADCLR  CALL   DELETE  DELETE 1 AT A TIME
0938 CA 35 09                JZ     ADCLR
093B                  *
093B CD F3 08         ADEND  CALL   ADRFMT  HL->END MARK FOR THIS FMT
093E CD 0D 09                CALL   ENDFMT
0941 79                      MOV    A,C
0942                  *
0942                  * INSERT A VALUE AT [HL] BY MOVING EVERYTHING ELSE OVER
0942                  *
0942 F5               INSERT PUSH   PSW    SAVE VALUE AND ADDRESS
0943 E5                      PUSH   H
0944 3A 32 10                LDA    RLEND  GET THE CURRENT RAMLIST END
0947 FE 78                   CPI    RLMAX*256/256  IF AT MAXIMUM QUIT
0949 D2 5C 09                JNC    INSX
094C 5F                      MOV    E,A    [DE]->LAST BYTE OF RAMLIST
094D 54                      MOV    D,H
094E                  *
094E 3C                      INR    A      RLEND++
094F 32 32 10                STA    RLEND
0952                  *
0952 1A               INS1   LDAX   D      MOVE LAST ELEMENT UP
0953 13                      INX    D
0954 12                      STAX   D
0955 1B                      DCX    D
0956 1B                      DCX    D
0957 7B                      MOV    A,E
0958 BD                      CMP    L
0959 D2 52 09                JNC    INS1
095C                  *
095C E1               INSX   POP    H
095D F1                      POP    PSW
095E 77                      MOV    M,A    PUT VALUE IN MEMORY
095F C9                      RET
0960                  *
0960                  * DELETE THE VALUE AT [HL]
0960                  * IF MEM[HL] IS AN END-OF-LIST, RETURN NOT ZERO
0960                  *
0960 7E               DELETE MOV    A,M    NEVER DELETE AN END OF LIST
0961 FE F0                   CPI    EOL
0963 CA 7A 09                JZ     RETNZ
0966                  *
```

```
0966 E5                   PUSH  H      SAVE THE START ADDRESS
0967 23          DEL1     INX   H      MOVE THE VALUE FROM 1 BEYOND
0968 7E                   MOV   A,M
0969 2B                   DCX   H
096A 77                   MOV   M,A
096B             *
096B 23                   INX   H
096C 3A 32 10             LDA   RLEND  IF RLEND WAS JUST MOVED
096F BD                   CMP   L      QUIT, ELSE LOOP AROUND
0970 C2 67 09             JNZ   DEL1
0973             *
0973 21 32 10             LXI   H,RLEND  RLEND--
0976 35                   DCR   M
0977 E1                   POP   H      RESTORE THE DELETE ADDRESS
0978 97                   SUB   A      RETURN ZERO
0979 C9                   RET
097A 3C          RETNZ    INR   A
097B C9                   RET
097C             *
097C 0B          ROMLIST DB     BV12   FMT#0
097D F0                   DB    EOL
097E             *
097E 0B                   DB    BV12   FMT#1
097F 07                   DB    BV1SC
0980 F0                   DB    EOL
0981             *
0981 0B                   DB    BV12   FMT#2
0982 05                   DB    BV34
0983 F0                   DB    EOL
0984             *
0984 0B                   DB    BV12   FMT#3
0985 05                   DB    BV34
0986 07                   DB    BV1SC
0987 F0                   DB    EOL
0988             *
0988 0B                   DB    BV12   FMT#4
0989 05                   DB    BV34
098A 01                   DB    BXFAST
098B 02                   DB    BYFAST
098C 00                   DB    BZFAST
098D F0                   DB    EOL
098E             *
098E 00                   DB    BZFAST  FMT#5
098F 01                   DB    BXFAST
0990 02                   DB    BYFAST
0991 03                   DB    BV3
0992 04                   DB    BV4
0993 05                   DB    BV34
0994 06 00                DW    BV34AC
0996 07                   DB    BV1SC
```

```
0997 08                DB     BV12AC
0998 09                DB     BV2
0999 0A                DB     BV1
099A 0B                DB     BV12
099B 0C                DB     BDIRECT
099C 0D                DB     BAGCU
099D 0E                DB     BGUARD
099E 0F                DB     BSTUB
099F F0                DB     EOL
09A0 F0                DB     EOL    FMT#6
09A1 F0                DB     EOL    FMT#7
09A2 F0                DB     EOL    FMT#8
09A3 F0                DB     EOL    FMT#9
09A4              *
09A4 F0           RAMDEF DB   EOL    FMT#10   RAM DEFAULT FORMATS 10-15
09A5 F0                DB     EOL    FMT#11
09A6 F0                DB     EOL    FMT#12
09A7 F0                DB     EOL    FMT#13
09A8 F0                DB     EOL    FMT#14
09A9 F0                DB     EOL    FMT#15
09AA              RAMDX  EQU  $-1
09AA 00      V         DB     256    END-OF-FMT
09AB              *
09AB              * VARIABLES
09AB              *
09AB                   ORG    BFMTRAM
1030              CURFMT DS   1      CURRENT LIST NUMBER
1031              INSFLAG DS  1      INSERT FLAG
1032              RLEND  DS   1      RAM LIST END-OF-MEMORY
1033              RAMLIST DS  64+6
1079              RLMAX  EQU  $-1
```

```
0000                *
0000                * CRRES FLIGHT SOFTWARE---BURST COMPILER SECTION
0000                * WRITTEN BY PETER R HARVEY
0000                *
0000                * FILE BCMP.A
0000                *
0000                ADC     EQU    3000H  A/D DATA
0000                ADCTL   EQU    3001H  A/D CONTROL
0000                MEM     EQU    8000H  BURST MEMORY BANK
0000                TESTQTY EQU    03FH   TEST QTY NUMBER
0000                PSW     EQU    6      8085 INFORMATION
0000                SP      EQU    6
0000                RIM     EQU    20H
0000                *
0000                * VARIABLES FOR THE BURST COMPILER
0000                *
0000                        ORG    BCMPRAM
1090                SLPTR   DS     2      SAMPLE LIST POINTER
1092                LPPTR   DS     2      LOOP POINTER
1094                TEST    EQU    SLPTR  TEST COUNTER
1094                RAMRTL  EQU    $
1094                *
1094                * BURST "S" COMPILER--- COMPILE A SAMPLE LIST INTO A
1094                *                       SAMPLE ROUTINE.
1094                * ON ENTRY: [HL]->CODE AREA
1094                *           [DE]->LIST TO SAMPLE
1094                *           C=LENGTH OF THE LIST
1094                *           B=0 FOR NO DELAY (HIGHEST FREQUENCY)
1094                *             1 FOR SOFTWARE TIMING
1094                *             2 FOR INTERRUPT TIMING
1094                *
1094                        ORG    BCMP
0A40 CD 18 0B       COMPILE CALL   STPUT  START PUTTING CODE AT [HL]
0A43 EB                     XCHG   .      [HL]->SAMPLE LIST
0A44 CD B7 0A               CALL   GENBEGIN  COMPILE A CALL TO INIT
0A47 CD D4 0A               CALL   GENTIM  GENERATE TIMING DELAY (IF NEEDED)
0A4A                *
0A4A 79                     MOV    A,C    IF THE LENGTH OF THE LIST
0A4B FE 03                  CPI    3      IS 3 OR MORE, SAMPLE Q3 THRU QN FIRST
0A4D D4 73 0A               CNC    CMPLIST
0A50                *
0A50 CD A6 0A               CALL   GENSMP  GENERATE SAMPLE FOR Q1
0A53 23                     INX    H
0A54 79                     MOV    A,C    IF LENGTH < 2, GO ON
0A55 FE 02                  CPI    2
0A57 DA 60 0A               JC     CMPEND
0A5A                *
0A5A CD CB 0A               CALL   GENSD  GENERATE DELAY BETWEEN Q1 AND Q2
0A5D CD A6 0A               CALL   GENSMP  SAMPLE Q2
```

```
0A60                 *
0A60 CD EE 0A        CMPEND CALL   GENFIN  GENERATE LOOP CODE
0A63 11 53 0B               LXI    D,ROMRTL
0A66 21 94 10               LXI    H,RAMRTL
0A69 0E 24                  MVI    C,RTLEN
0A6B CD 27 0B               CALL   COPY
0A6E 2A 90 10               LHLD   SLPTR  RETURN(NEXT AVAIL MEMORY ADDR)
0A71 B7                     ORA    A
0A72 C9                     RET
0A73                 *
0A73                 * COMPILE LIST Q3 THRU QN
0A73                 *
0A73 E5              CMPLIST PUSH  H      SAVE THE LIST ADDRESS
0A74 C5                     PUSH   B      AND ITS LENGTH
0A75 23                     INX    H      STEP PAST THE 1ST TWO ELEMENTS
0A76 23                     INX    H
0A77 0D                     DCR    C
0A78 0D                     DCR    C
0A79                 *
0A79 CD A6 0A        CL1    CALL   GENSMP  GENERATE A SAMPLE
0A7C CD CB 0A               CALL   GENSD  AND A SAMPLE-DELAY
0A7F 23                     INX    H      STEP TO NEXT LIST ELEMENT
0A80 0D                     DCR    C
0A81 C2 79 0A               JNZ    CL1
0A84 C1                     POP    B
0A85 E1                     POP    H
0A86 C9                     RET
0A87                 *
0A87                 * GENSMP -- CODE GENERATION ROUTINES
0A87                 *
0A87                 EOC    EQU    -1     END-OF-CODE
0A87                 *
0A87 3A A0 0A        GENBEGIN LDA  BC1    START BY LOADING THE
0A8A CD 1C 0B               CALL   PUT    FREQUENCY CODING.
0A8D 78                     MOV    A,B
0A8E CD 1C 0B               CALL   PUT
0A91                 *
0A91 11 A2 0A               LXI    D,BEGCODE  INSERT THE INIT ROUTINE
0A94 CD 00 0B               CALL   PUTN
0A97 E5                     PUSH   H
0A98 2A 90 10               LHLD   SLPTR  THEN RECORD WHERE THE LOOP GOES
0A9B 22 92 10               SHLD   LPPTR
0A9E E1                     POP    H
0A9F C9                     RET
0AA0                 *
0AA0 06 00           BC1    MVI    B,0
0AA2 CD 30 08        BEGCODE CALL  STARBURST
0AA5 FF                     DB     EOC
0AA6                 *
0AA6 7E              GENSMP MOV    A,M    IF TEST QTY, USE SPECIAL CODE
```

```
0AA7 FE 3F                 CPI   TESTQTY
0AA9 11 C3 0A              LXI   D,TSTCODE
0AAC CA 0D 0B              JZ    PUTN
0AAF 3A C0 0A              LDA   SMPCODE  PUT THE OPCODE
0AB2 CD 1C 0B              CALL  PUT
0AB5 7E                    MOV   A,M    THEN THE QTY*2
0AB6 87                    ADD   A
0AB7 CD 1C 0B              CALL  PUT
0ABA 3A C2 0A              LDA   SMPCODE+2
0ABD C3 1C 0B              JMP   PUT
0AC0 2A 00 B0      SMPCODE LHLD  MEM+ADC+0  EXAMPLE OF A/D SAMPLE
0AC3 2A 90 90      TSTCODE LHLD  MEM+TEST  EXAMPLE OF TEST SAMPLE
0AC6 2C                    INR   L
0AC7 22 90 10              SHLD  TEST
0ACA FF                    DB    EOC
0ACB               *
0ACB               * GENERATE SAMPLE DELAY (BETWEEN SAMPLES)
0ACB               *
0ACB 11 D1 0A      GENSD   LXI   D,SDCODE  COPY SOME CODE
0ACE C3 0D 0B              JMP   PUTN
0AD1 DB 7F         SDCODE  IN    7FH    DELAY 10 CYCLES
0AD3 FF                    DB    EOC
0AD4               *
0AD4               * GENERATE TIMING
0AD4               *
0AD4 78            GENTIM  MOV   A,B    GET THE TIMING INFO
0AD5 B7                    ORA   A      IF NO DELAY, NO CODE.
0AD6 C8                    RZ    .
0AD7 3D                    DCR   A      IF B=1, SOFTWARE DELAY
0AD8 11 E9 0A              LXI   D,SOFTIM
0ADB CA 0D 0B              JZ    PUTN
0ADE 11 E4 0A              LXI   D,INTTIM  ELSE INTERRUPT DELAY
0AE1 C3 0D 0B              JMP   PUTN
0AE4               *
0AE4 69            INTTIM  MOV   L,C    COUNT THE INTERRUPTS
0AE5 CD 9F 10              CALL  INTDLA
0AE8 FF                    DB    EOC
0AE9 69            SOFTIM  MOV   L,C    COUNT CYCLES
0AEA CD 94 10              CALL  SOFDLA
0AED FF                    DB    EOC
0AEE               *
0AEE               * GENERATE LOOP END CODE
0AEE               *
0AEE 11 06 0B      GENFIN  LXI   D,LPCODE
0AF1 CD 0D 0B              CALL  PUTN
0AF4 3A 92 10              LDA   LPPTR  PUT THE LOOP ADDRESS IN
0AF7 CD 1C 0B              CALL  PUT
0AFA 3A 93 10              LDA   LPPTR+1
0AFD CD 1C 0B              CALL  PUT
0B00 11 09 0B              LXI   D,FINCODE
```

```
0B03 C3 0D 0B        JMP     PUTN
0B06            *
0B06            LPCODE EQU     $ THE CODE WHICH IMPLEMENTS A LOOP
0B06 D2 FF FF           JNC     EOC    (COPY UP TO THE JNC OPCODE)
0B09 C3 48 0B   FINCODE JMP     ENDBURST
0B0C FF                 DB      EOC
0B0D            *
0B0D 1A         PUTN    LDAX    D      COPY CODE FROM MEM[DE]
0B0E FE FF              CPI     EOC
0B10 C8                 RZ
0B11 CD 1C 0B           CALL    PUT
0B14 13                 INX     D
0B15 C3 0D 0B           JMP     PUTN
0B18            *
0B18 22 90 10   STPUT   SHLD    SLPTR  SET SAMPLE LIST POINTER
0B1B C9                 RET
0B1C            *
0B1C E5         PUT     PUSH    H      SAVE [HL]
0B1D 2A 90 10           LHLD    SLPTR  PUT ACCUM INTO MEM AT SLPTR++
0B20 77                 MOV     M,A
0B21 23                 INX     H
0B22 22 90 10           SHLD    SLPTR
0B25 E1                 POP     H
0B26 C9                 RET
0B27            *
0B27 1A         COPY    LDAX    D
0B28 77                 MOV     M,A
0B29 13                 INX     D
0B2A 23                 INX     H
0B2B 0D                 DCR     C
0B2C C2 27 0B           JNZ     COPY
0B2F C9                 RET
0B30            *
0B30            * THE S COMPILER'S RUN TIME LIBRARY
0B30            * (SEE ABOVE FOR RESTART 1 AND 2)
0B30            *
0B30            * STARBURST INITIALIZES THE SYSTEM SO THAT THE COMPILED
0B30            * SAMPLING ROUTINE ALWAYS WORKS.
0B30            *
0B30            STARBURST EQU $
0B30 3E 01              MVI     A,1    SET MEMORY FOR AUTOWRITE
0B32 CD 61 00           CALL    MODESET
0B35 3E 03              MVI     A,3    SET A/D FOR AUTO-CONVERT
0B37 32 01 30           STA     ADCTL
0B3A 21 00 00           LXI     H,0    START TEST COUNTER AT 0
0B3D 22 90 10           SHLD    TEST
0B40 3E 01              MVI     A,1    SET I/O MODE TO SET CARRY
0B42 CD 79 00           CALL    IOMODE  WHEN COMMAND PENDING
0B45 FB                 EI      .      ALWAYS KEEP COMMAND INPUT (7.5) ENABLED
0B46 3A 90 10           LDA     BCMPRAM  REFERENCE RAM FOR LOW POWER
```

```
0B49 B7                      ORA    A      CLEAR CARRY
0B4A C9                      RET
0B4B                   *
0B4B 97                ENDBURST SUB A      REMOVE COMMAND-PENDING
0B4C CD 79 00                CALL   IOMODE  STATUS RETURN
0B4F 97                      SUB    A      CLEAR AUTOWRITE MODE
0B50 C3 61 00                JMP    MODESET
0B53                   *
0B53                   * THE DELAY ROUTINES
0B53                   *
0B53                   ROMRTL EQU   $
0B53                   SOFDLA EQU   RAMRTL
0B53 FB                      EI
0B54 2D                      DCR    L
0B55 C8                      RZ
0B56 DB 7F                   IN     7FH
0B58 00                      NOP
0B59 00                      NOP
0B5A D2 94 10                JNC    SOFDLA
0B5D C9                      RET
0B5E                   *
0B5E                   INTDLA EQU   $-ROMRTL+RAMRTL
0B5E D4 A7 10                CNC    INTWAIT
0B61 2D                      DCR    L
0B62 C2 9F 10                JNZ    INTDLA
0B65 C9                      RET
0B66                   *
0B66                   INTWAIT EQU  $-ROMRTL+RAMRTL
0B66 F3                      DI
0B67 F5                      PUSH   PSW
0B68 06 20                   MVI    B,20H
0B6A                   IWH    EQU   $-ROMRTL+RAMRTL
0B6A 20                      DB     RIM
0B6B A0                      ANA    B
0B6C C2 AB 10                JNZ    IWH
0B6F                   *
0B6F                   IWL    EQU   $-ROMRTL+RAMRTL
0B6F 20                      DB     RIM
0B70 A0                      ANA    B
0B71 CA B0 10                JZ     IWL
0B74 F1                      POP    PSW
0B75 FB                      EI
0B76 C9                      RET
0B77                   RTLEN  EQU   $-ROMRTL
0B77 00           V          DB     256    END-OF-BCMP MODULE
0B7B                   *
0B7B                   * EXTERNALS
0B7B                   *
0B7B                         ORG    BIO
0040                   BIOINIT DS   3
```

```
0043            0250 GETMASK DS    3
0046            0251 SETMASK DS    3
0049            0252 RECSTAT DS    3
004C            0253 RECEIVE DS    3
004F            0254 SEND    DS    3
0052            0255 ADPWR   DS    3
0055            0256 SAMPLE DS     3
0058            0257 MEMPWR DS     3
005B            0258 MARSET DS     3
005E            0259 BANKSET DS    3
0061            0260 MODESET DS    3
0064            0261 SECOND DS     3
0067            0262 D5MS    DS    3
006A            0263 READ    DS    3
006D            0264 WRITE   DS    3
0070            0265 REWIND DS     3
0073            0266 MARGET DS     3
0076            0267 SETVECT DS    3
0079            0268 IOMODE DS     3
```

```
0000           *
0000           * CRRES FLIGHT SOFTWARE---FAST FLOATING POINT
0000           *  WRITTEN BY PETER HARVEY
0000           *  FILE : BFFP.A
0000           *
0000           *  F.P. REGISTER IS CDE.
0000           *  FORMAT IS SIGN(1)+EXP(7)+MANTISSA(16)
0000           *  NO HIDDEN BIT
0000           *
0000           PSW    EQU    6
0000           SP     EQU    6
0000           *
0000                  ORG    FFP
0C00 C3 27 0C         JMP    LODFP
0C03 C3 2D 0C         JMP    STOFP
0C06 C3 33 0C         JMP    FMUL
0C09 C3 5E 0C         JMP    FDIV
0C0C C3 F2 0C         JMP    FADD
0C0F C3 EB 0C         JMP    FSUB
0C12 C3 DC 0C         JMP    FCMP
0C15 C3 4B 0C         JMP    FNEG
0C18 C3 62 0D         JMP    FLT32
0C1B C3 DD 0D         JMP    FIX32
0C1E C3 26 0E         JMP    FSQUA
0C21 C3 2F 0E         JMP    FSQRT
0C24 C3 5D 0E         JMP    MU21
0C27           *
0C27 4E        LODFP  MOV    C,M
0C28 23               INX    H
0C29 56               MOV    D,M
0C2A 23               INX    H
0C2B 5E               MOV    E,M
0C2C C9               RET    .
0C2D           *
0C2D 71        STOFP  MOV    M,C
0C2E 23               INX    H
0C2F 72               MOV    M,D
0C30 23               INX    H
0C31 73               MOV    M,E
0C32 C9               RET    .
0C33           *
0C33           * F.P. MULTIPLY ROUTINE
0C33           *
0C33 7A        FMUL   MOV    A,D     IF X=0, QUIT NOW
0C34 B7               ORA    A
0C35 C8               RZ     .
0C36 46               MOV    B,M     LOAD PARAM FROM MEM
0C37 23               INX    H       INTO BHL FORMAT
0C38 7E               MOV    A,M
```

```
0C39 B7                  ORA   A     IF ZERO THEN SET TO 0
0C3A CA C3 0D            JZ    RETO
0C3D 23                  INX   H     ELSE LOAD THE REST
0C3E 6E                  MOV   L,M
0C3F 67                  MOV   H,A
0C40             *
0C40 78          FMS33   MOV   A,B   IF SAME SIGN, GO
0C41 A9                  XRA   C
0C42 F2 50 0C            JP    FMU33
0C45 CD D4 0D            CALL  STRIP REMOVE SIGNS FROM B&C
0C48 CD 50 0C            CALL  FMU33 MULTIPLY THEN NEGATE
0C4B             *
0C4B             FNEG    EQU   $
0C4B 79          NEGFP   MOV   A,C   AND NEGATE F.P
0C4C EE 80               XRI   80H
0C4E 4F                  MOV   C,A
0C4F C9                  RET   .
0C50             *
0C50             * F.P. MULTIPLY POSITIVES ONLY
0C50             *
0C50 78          FMU33   MOV   A,B   ADD EXPONENTS
0C51 81                  ADD   C
0C52 D6 40               SUI   40H   ADJUST BACK TO EXCESS 64
0C54 FA C9 0D            JM    ERCHK IF MINUS, CHECK THE ERROR
0C57 4F                  MOV   C,A
0C58 CD 43 0E            CALL  MU22F [AHL.]= DE X HL
0C5B C3 AF 0D            JMP   NCHK  SHIFT UNTIL AHL NORMED,ROUND OFF
0C5E             *
0C5E             * F.P. DIVIDE
0C5E             *
0C5E 7A          FDIV    MOV   A,D   IF ZERO DIVIDEND, QUIT
0C5F B7                  ORA   A
0C60 C8                  RZ    .
0C61 46                  MOV   B,M   PICK UP DIVISOR
0C62 23                  INX   H
0C63 7E                  MOV   A,M
0C64 B7                  ORA   A     IF DIVISOR 0, OVERFLOW
0C65 CA CE 0D            JZ    OVERFLOW
0C68 23                  INX   H
0C69 6E                  MOV   L,M
0C6A 67                  MOV   H,A
0C6B             *
0C6B 78                  MOV   A,B   IF SAME SIGN, DO
0C6C A9                  XRA   C     SAME SIGNED VERSION
0C6D F2 79 0C            JP    FDU33
0C70 CD D4 0D            CALL  STRIP REMOVE SIGNS
0C73 CD 79 0C            CALL  FDU33 DIVIDE OUT
0C76 C3 4B 0C            JMP   NEGFP AND NEGATE
0C79             *
0C79 79          FDU33   MOV   A,C   EXP=C-B+40H
```

```
0C7A 90              SUB   B
0C7B C6 40           ADI   40H
0C7D FA C9 0D        JM    ERCHK
0C80 4F              MOV   C,A
0C81 C5              PUSH  B      SAVE EXPONENT
0C82          *
0C82 7C              MOV   A,H    BC=-DIVISOR
0C83 2F              CMA
0C84 47              MOV   B,A
0C85 7D              MOV   A,L
0C86 2F              CMA
0C87 4F              MOV   C,A
0C88 03              INX   B
0C89          *
0C89          * IF THE REMAINDER STARTS AS LARGE AS
0C89          * THE DIVISOR, THE FIRST BIT IS 1
0C89          *
0C89 62              MOV   H,D    HL=REMAINDER
0C8A 6B              MOV   L,E
0C8B 09              DAD   B      HL=REMAINDER-DIVISOR
0C8C DA A9 0C        JC    FBIT1
0C8F          *
0C8F          * IF REMAINDER LESS THAN DIVISOR, THE FIRST
0C8F          * BIT (INTEGER PART) IS ZERO.  DIVIDE FOR
0C8F          * FRACTIONAL PART WHICH WILL BE AUTOMATICALLY
0C8F          * NORMALIZED.
0C8F          *
0C8F EB              XCHG  .      HL=REMAINDER AGAIN
0C90 3E 10           MVI   A,16
0C92 CD C2 0C        CALL  FDSHF  [DE]=[HL]*2/[BC]
0C95 29              DAD   H      IF REMAINDER>8000H
0C96 DA 9F 0C        JC    DVROND  THEN ROUND UP
0C99 09              DAD   B      IF NEXT BIT WOULD BE 1
0C9A DA 9F 0C        JC    DVROND  THEN ROUND UP
0C9D C1              POP   B      RESTORE EXPONENT
0C9E C9              RET   .      NO NORMALIZATION REQD
0C9F          *
0C9F C1       DVROND POP   B      C=EXPONENT
0CA0 1C       ROND   INR   E ROUND OFF DE
0CA1 C0              RNZ   .      BUT DON'T PRODUCE
0CA2 14              INR   D      A ZERO
0CA3 C0              RNZ   .
0CA4 11 00 80        LXI   D,8000H  IF ZERO, THEN
0CA7 0C              INR   C      UP THE EXPONENT
0CA8 C9              RET   .
0CA9          *
0CA9          * FIRST BIT=1, DIVIDE OUT 16 MORE BITS
0CA9          * USING WHAT'S LEFT OF THE REMAINDER IN HL
0CA9          *
0CA9 3E 10    FBIT1  MVI   A,16
```

```
0CAB 11 FF FF             LXI    D,-1
0CAE CD C2 0C             CALL   FDSHF  [DE]=[HL]/[BC]
0CB1                *
0CB1 C1                   POP    B      RESTORE THE EXPONENT
0CB2 0C                   INR    C      ADJUST SINCE 1ST BIT=1
0CB3 37                   STC    .      RIGHT SHIFT A 1 INTO DE
0CB4 7A                   MOV    A,D
0CB5 1F                   RAR
0CB6 57                   MOV    D,A
0CB7 7B                   MOV    A,E
0CB8 1F                   RAR
0CB9 5F                   MOV    E,A
0CBA D0                   RNC    .      IF 17TH BIT WAS 0, STOP
0CBB C3 A0 0C             JMP    ROND   ELSE ROUND OFF
0CBE                *
0CBE                * DIVIDE NORMALIZED INTEGERS FOR F.P.
0CBE                *
0CBE 33             FDSTK INX    SP     REMOVE PARTIAL REMAINDER
0CBF 33                   INX    SP     FROM STACK
0CC0 3D             FDTST DCR    A      DECR BIT COUNTER
0CC1 C8                   RZ     .
0CC2 29             FDSHF DAD    H      BRING DOWN A BIT INTO REM
0CC3 DA D4 0C             JC     SUBIT  IF >=10000, THEN SUBTRACT
0CC6 EB                   XCHG   .
0CC7 29                   DAD    H      AND SHIFT RESULT REG
0CC8 EB                   XCHG
0CC9                *
0CC9 1C                   INR    E      ASSUME RESULT=1
0CCA E5             FDV22 PUSH   H      SAVE REMAINDER ON STK
0CCB 09                   DAD    B      IF REM<DIVISOR,LEAVE REM ALONE
0CCC DA BE 0C             JC     FDSTK
0CCF E1                   POP    H      ELSE RESTORE REMAINDER
0CD0 1D                   DCR    E      SET RESULT BIT=0
0CD1 C3 C0 0C             JMP    FDTST
0CD4                *
0CD4 EB             SUBIT XCHG   .      FINISH THE SHIFT
0CD5 29                   DAD    H
0CD6 EB                   XCHG   .
0CD7 09                   DAD    B      SUBTRACT DIVISOR
0CD8 1C                   INR    E      SET RESULT BIT
0CD9 C3 C0 0C             JMP    FDTST
0CDC                *
0CDC                * F.P. COMPARE
0CDC                * ON EXIT: ZERO SET IF EQUAL, CARRY IF LESS THAN
0CDC                *             CDE UNTOUCHED
0CDC                *
0CDC C5             FCMP  PUSH   B      SAVE CDE
0CDD D5                   PUSH   D
0CDE CD EB 0C             CALL   FSUB   SUBTRACT THE TWO
0CE1 7A                   MOV    A,D    IF RESULT=0, RET
```

```
0CE2 B7                 ORA   A
0CE3 CA E8 0C           JZ    FCMPX
0CE6 79                 MOV   A,C    IF NEGATIVE, THEN
0CE7 07                 RLC   .      SET CARRY, ELSE NO CARRY
0CE8 D1        FCMPX    POP   D      RESTORE CDE
0CE9 C1                 POP   B
0CEA C9                 RET   .
0CEB           *
0CEB           * F.P. SUB
0CEB           *
0CEB 7E        FSUB     MOV   A,M    INVERT SIGN OF 2ND
0CEC EE 80              XRI   80H    PARAMETER
0CEE 47                 MOV   B,A
0CEF C3 F3 0C           JMP   FAD1
0CF2           *
0CF2           * F.P. ADD
0CF2           *
0CF2 46        FADD     MOV   B,M    LOAD UP
0CF3 23        FAD1     INX   H
0CF4 7E                 MOV   A,M
0CF5 23                 INX   H
0CF6 6E                 MOV   L,M
0CF7 67                 MOV   H,A
0CF8 97                 SUB   A
0CF9 BC                 CMP   H      IF BHL=0, QUIT
0CFA C8                 RZ    .
0CFB BA                 CMP   D      IF CDE=0, QUIT
0CFC CA 5F 0D           JZ    SWITCH
0CFF           *
0CFF 79                 MOV   A,C    COMPUTE EXP DIFFERENCE
0D00 90                 SUB   B
0D01 87                 ADD   A
0D02 F2 0B 0D           JP    POSDX
0D05 78                 MOV   A,B    SWAP CDE FOR BHL
0D06 41                 MOV   B,C
0D07 4F                 MOV   C,A
0D08 EB                 XCHG  .
0D09 90                 SUB   B      COMPUTE EXP DIFFERENCE
0D0A 87                 ADD   A      AGAIN
0D0B CA 15 0D  POSDX    JZ    ADSUB
0D0E 0F                 RRC   .      DIV BY 2
0D0F FE 10              CPI   16     IF CDE>>BHL, QUIT
0D11 D0                 RNC
0D12 CD 47 0D           CALL  SHFHL  REDUCE HL A TIMES
0D15           *
0D15 78        ADSUB    MOV   A,B    IF SIGNS DIFFER, GO
0D16 A9                 XRA   C
0D17 FA 25 0D           JM    DIFFER
0D1A 19                 DAD   D      ADD DE TO HL
0D1B EB                 XCHG  .      IF NO CARRY,
```

```
0D1C D0               RNC    .      RETURN CDE
0D1D             *
0D1D 7A          RITE1 MOV   A,D    ELSE SHIFT RIGHT ONE
0D1E 1F               RAR    .      INCLUDING THE CARRY
0D1F 57               MOV    D,A
0D20 7B               MOV    A,E
0D21 1F               RAR
0D22 5F               MOV    E,A
0D23 0C               INR    C      ADJUST EXPONENT
0D24 C9               RET    .
0D25             *
0D25 7B          DIFFER MOV  A,E    IF DE<HL,
0D26 95               SUB    L      THEN NORM(B:HL-DE)
0D27 7A               MOV    A,D
0D28 9C               SBB    H
0D29 DA 3A 0D         JC     SUBD
0D2C             *
0D2C 57               MOV    D,A    ELSE NORM(C:DE-HL)
0D2D 7B               MOV    A,E
0D2E 95               SUB    L
0D2F 5F               MOV    E,A
0D30 21 00 00         LXI    H,0
0D33 B2               ORA    D      IF DE<>0,
0D34 C2 74 0D         JNZ    NORM   NORMALIZE WITH C
0D37 0E 00            MVI    C,0    IF HL-DE=0, RETURN
0D39 C9               RET    .
0D3A             *
0D3A 7D          SUBD  MOV   A,L    DE = DE - HL
0D3B 93               SUB    E
0D3C 5F               MOV    E,A
0D3D 7C               MOV    A,H
0D3E 9A               SBB    D
0D3F 57               MOV    D,A
0D40 48               MOV    C,B    USE BHL'S EXPONENT
0D41 21 00 00         LXI    H,0    AND SHIFT IN ZEROES
0D44 C3 74 0D         JMP    NORM   NORMALIZE
0D47             *
0D47             * SHIFT HL RIGHT A TIMES
0D47             *
0D47 D6 08       SHFHL SUI   8      IF LT 8, GO NOW
0D49 DA 52 0D         JC     LT8
0D4C 6C               MOV    L,H    SHIFT 8
0D4D 26 00            MVI    H,0
0D4F C8               RZ     .      IF EXACTLY 8, RETURN
0D50 D6 08            SUI    8      ELSE DO SECOND 8
0D52 C5          LT8   PUSH  B      SAVE EXPONENTS
0D53 47               MOV    B,A    SAVE INVERTED COUNTER
0D54 97               SUB    A      CLEAR ACCUM
0D55 29          SHF1  DAD   H      SHIFT LEFT
0D56 8F               ADC    A      INTO A FROM HL
```

```
0D57 04                   INR    B       COUNT UP TO 0
0D58 C2 55 0D             JNZ    SHF1
0D5B C1                   POP    B       RESTORE EXPS
0D5C 6C                   MOV    L,H
0D5D 67                   MOV    H,A
0D5E C9                   RET    .
0D5F               *
0D5F EB            SWITCH XCHG   .       CDE= BHL
0D60 48                   MOV    C,B
0D61 C9                   RET    .
0D62               *
0D62               * CONVERT 32 BIT DATA TO F.P. FORMAT
0D62               *
0D62 7A            FLT32  MOV    A,D     IF POSITIVE, JUST NORM
0D63 B7                   ORA    A
0D64 0E 60                MVI    C,64+32  WITH LSB=2**0 TO BEGIN
0D66 F2 74 0D             JP     NORM
0D69 CD BC 0E             CALL   NEG32   NEGATE DEHL
0D6C CD 74 0D             CALL   NORM    NOW NORMALIZE
0D6F 79                   MOV    A,C     AND NEGATE FP
0D70 F6 80                ORI    80H
0D72 4F                   MOV    C,A
0D73 C9                   RET    .
0D74               *
0D74               * NORMALIZE C:DEHL TO F.P. NORMAL FORM
0D74               *
0D74 79            NORM   MOV    A,C     IF C NEGATIVE, TRAP IT
0D75 B7                   ORA    A
0D76 F2 82 0D             JP     NORMP
0D79 E6 7F                ANI    7FH
0D7B 4F                   MOV    C,A
0D7C CD 82 0D             CALL   NORMP
0D7F C3 4B 0C             JMP    NEGFP   AND NEG LATER
0D82               *
0D82 7A            NORMP  MOV    A,D     IF WITHIN 8 BITS, GO NOW
0D83 B7                   ORA    A
0D84 C2 AC 0D             JNZ    NORM1
0D87 B3                   ORA    E       IF WITHIN 16, USE EHL
0D88 C2 A1 0D             JNZ    NRMEHL
0D8B B4                   ORA    H       IF WITHIN 24, USE HL
0D8C C2 9B 0D             JNZ    NRMHL
0D8F B5                   ORA    L       IF JUST L, USE IT
0D90 C2 95 0D             JNZ    NRML
0D93 4A                   MOV    C,D     ELSE CDE=0
0D94 C9                   RET    .
0D95               *
0D95 55            NRML   MOV    D,L     LOO FOR 3 BYTES
0D96 06 18                MVI    B,24    ADJUST EXP BY 24 BITS
0D98 C3 A6 0D             JMP    AJEXP
0D9B EB            NRMHL  XCHG   .       HLO FOR 3BYTES
```

```
0D9C 06 10              MVI   B,16   ADJUST EXP 16
0D9E C3 A6 0D           JMP   AJEXP
0DA1 53          NRMEHL MOV   D,E    SHIFT EHL TO DEH
0DA2 5C                 MOV   E,H
0DA3 65                 MOV   H,L
0DA4 06 08              MVI   B,8    ADJUST B BITS
0DA6 79          AJEXP  MOV   A,C    EXP=EXP-B
0DA7 90                 SUB   B
0DA8 4F                 MOV   C,A    IF PROBLEM, THEN UNDER
0DA9 DA C3 0D           JC    UNDERFLOW
0DAC             *
0DAC             * BIT BY BIT NORMALIZATION
0DAC             *
0DAC 7A          NORM1  MOV   A,D    AHL=DEH
0DAD 6C                 MOV   L,H
0DAE 63                 MOV   H,E
0DAF B7          NCHK   ORA   A      SHIFT AHL TILL NORMED
0DB0 FA B9 0D           JM    NRMFIN
0DB3 0D          NCHK1  DCR   C      EXP<-EXP-1
0DB4 29                 DAD   H
0DB5 8F                 ADC   A
0DB6 F2 B3 0D           JP    NCHK1
0DB9 57          NRMFIN MOV   D,A    DE=AH
0DBA 5C                 MOV   E,H
0DBB 7D                 MOV   A,L    IF MSB(L)=1, ROUND OFF DE
0DBC 07                 RLC   .
0DBD DC A0 0C           CC    ROND
0DC0 79                 MOV   A,C    IF EXP POSITIVE, OK
0DC1 B7                 ORA   A
0DC2 F0                 RP    .
0DC3             *
0DC3             * ERRORS : UNDERFLOW AND OVERFLOW
0DC3             *
0DC3             UNDERFLOW EQU $
0DC3 0E 00       RET0   MVI   C,0    RETURN CDE=0
0DC5 11 00 00           LXI   D,0
0DC8 C9                 RET   .
0DC9             *
0DC9 FE C0       ERCHK  CPI   0C0H   IF BETWEEN 080H AND 0BFH
0DCB D2 C3 0D           JNC   UNDERFLOW  THEN UNDERFLOW, ELSE OVER
0DCE             *
0DCE 0E 7F       OVERFLOW MVI C,7FH  RETURN CDE=MAXIMUM
0DD0 11 FF FF           LXI   D,-1
0DD3 C9                 RET   .
0DD4             *
0DD4 78          STRIP  MOV   A,B    REMOVE SIGNS FROM B
0DD5 E6 7F              ANI   7FH
0DD7 47                 MOV   B,A
0DD8 79                 MOV   A,C
0DD9 E6 7F              ANI   7FH
```

```
0DDB 4F                  MOV   C,A
0DDC C9                  RET   .
0DDD                *
0DDD                * FIX32:  FLT TO FIX CONVERSION
0DDD                *
0DDD 79             FIX32 MOV   A,C    IF NEGATIVE, INVERT
0DDE EE 80                XRI   80H    RESULTS
0DE0 FA EA 0D             JM    FIXPOS
0DE3 4F                   MOV   C,A
0DE4 CD EA 0D             CALL  FIXPOS
0DE7 C3 BC 0E             JMP   NEG32
0DEA                *
0DEA E6 7F          FIXPOS ANI  7FH    IF CDE<1, RETURN(0)
0DEC FE 41                CPI   41H
0DEE DA 18 0E             JC    ZERDH
0DF1 FE 60                CPI   60H    IF >2**31, MAX IT
0DF3 D2 1F 0E             JNC   MAXDH
0DF6                *
0DF6 21 00 00             LXI   H,0    ELSE SHIFT MANTISSA
0DF9 D6 50                SUI   40H+16  IF 2**16, QUIT
0DFB C8                   RZ    .
0DFC EB                   XCHG  .      DEHL=00XX, READY TO SHIFT
0DFD D2 0F 0E             JNC   SHDH   IF EXP WAS 51 TO 5F, 60
0E00 C6 10                ADI   16     ELSE 41-4F, SHIFT THEN
0E02 CD 0F 0E             CALL  SHDH   DIVIDE BY 2**16
0E05 EB                   XCHG
0E06 11 00 00             LXI   D,0
0E09 C9                   RET   .
0E0A                *
0E0A 29             SHCAR DAD   H      SHIFT DE PART
0E0B 2C                   INR   L      AND PUT IN CARRY
0E0C EB             DECRA XCHG  .      SWAP BACK HL
0E0D 3D                   DCR   A      IF COUNT=0, QUIT
0E0E C8                   RZ    .
0E0F 29             SHDH  DAD   H      SHIFT HL ONE BIT
0E10 EB                   XCHG  .      IF CARRY, THEN
0E11 DA 0A 0E             JC    SHCAR  UPDATE DE WITH CARRY
0E14 29                   DAD   H      ELSE WITHOUT CARRY
0E15 C3 0C 0E             JMP   DECRA
0E18                *
0E18 11 00 00       ZERDH LXI   D,0    DEHL=0
0E1B 21 00 00             LXI   H,0
0E1E C9                   RET   .
0E1F 11 FF 7F       MAXDH LXI   D,7FFFH  DEHL=MAXIMUM
0E22 21 FF FF             LXI   H,-1
0E25 C9                   RET   .
0E26                *
0E26                * SQUARE [CDE]
0E26                *
0E26 7A             FSQUA MOV   A,D    CHECK FOR 0
```

```
0E27 B7                    ORA   A
0E28 C8                    RZ    .
0E29 41                    MOV   B,C    BHL=CDE
0E2A 62                    MOV   H,D
0E2B 6B                    MOV   L,E
0E2C C3 40 0C              JMP   FMS33
0E2F                 *
0E2F 7A              FSQRT MOV   A,D    IF ZERO, QUIT
0E30 B7                    ORA   A
0E31 C8                    RZ    .
0E32 79                    MOV   A,C    IF ODD EXPONENT, SHIFT
0E33 E6 01                 ANI   1
0E35 C4 1D 0D              CNZ   RITE1
0E38 C5                    PUSH  B      SAVE EXPONENT
0E39 CD 97 0E              CALL  SQR2   DE=DE**1/2
0E3C C1                    POP   B
0E3D 79                    MOV   A,C    DIVIDE EXP BY 2
0E3E 0F                    RRC
0E3F C6 20                 ADI   20H    IN EXCESS 64
0E41 4F                    MOV   C,A
0E42 C9                    RET   .
0E43                 *
0E43                 * 16 X 16 MULTIPLY UNSIGNED.  OPTIMIZED FOR F.P.
0E43                 *  [AHL] = [HL] * [DE]  TOP 3 BYTES
0E43                 *
0E43 97              MU22F SUB   A      IF E=0, DO SHORT MULT
0E44 BB                    CMP   E
0E45 CA 5B 0E              JZ    SHORD
0E48 B5                    ORA   L      IF L=0, DO SHORT WITH H
0E49 CA 5C 0E              JZ    SHORH
0E4C E5                    PUSH  H      AHL= L*DE
0E4D CD 5D 0E              CALL  MU21
0E50 6C                    MOV   L,H    THROW AWAY LS BYTE
0E51 67                    MOV   H,A    SAVE UPPER BYTES
0E52 E3                    XTHL  .      SAVE EM, GET MS BYTE OF 1ST
0E53                 *
0E53 7C                    MOV   A,H    AHL=MSB*DE
0E54 CD 5D 0E              CALL  MU21
0E57 D1                    POP   D      GRAB THE TWO STORED
0E58 19                    DAD   D      ADD PARTIAL RESULTS
0E59 88                    ADC   B      FOR THREE BYTES (AHL)
0E5A C9                    RET   .
0E5B                 *
0E5B EB              SHORD XCHG  .      SHORT MULT
0E5C 7C              SHORH MOV   A,H    JUST MULT H*DE
0E5D                 *
0E5D                 * 16 X 8 MULTIPLY UNSIGNED
0E5D                 *  [AHL] <- A * [DE]
0E5D                 *  TAKES 198 TO 297 CYCLES
0E5D                 *
```

```
0E5D 21 00 00     MU21   LXI   H,0    ZERO RESULT REG
0E60 44                  MOV   B,H    B<-0
0E61              *
0E61 87           MULTX  ADD   A      SHIFT MSB TO CARRY
0E62 D2 67 0E            JNC   X2
0E65 19                  DAD   D      IF C=1, THEN ADD [DE]
0E66 88                  ADC   B      IF OVERFLOW,BUMP MSBYTE
0E67 29           X2     DAD   H      SHIFT FOR NEXT TEST
0E68              *
0E68 8F                  ADC   A      AND SO ON
0E69 D2 6E 0E            JNC   X4
0E6C 19                  DAD   D
0E6D 88                  ADC   B
0E6E 29           X4     DAD   H
0E6F              *
0E6F 8F                  ADC   A
0E70 D2 75 0E            JNC   X8
0E73 19                  DAD   D
0E74 88                  ADC   B
0E75 29           X8     DAD   H
0E76              *
0E76 8F                  ADC   A
0E77 D2 7C 0E            JNC   X10
0E7A 19                  DAD   D
0E7B 88                  ADC   B
0E7C 29           X10    DAD   H
0E7D              *
0E7D 8F                  ADC   A
0E7E D2 83 0E            JNC   X20
0E81 19                  DAD   D
0E82 88                  ADC   B
0E83 29           X20    DAD   H
0E84              *
0E84 8F                  ADC   A
0E85 D2 8A 0E            JNC   X40
0E88 19                  DAD   D
0E89 88                  ADC   B
0E8A 29           X40    DAD   H
0E8B              *
0E8B 8F                  ADC   A
0E8C D2 91 0E            JNC   X80
0E8F 19                  DAD   D
0E90 88                  ADC   B
0E91 29           X80    DAD   H
0E92              *
0E92 8F                  ADC   A
0E93 D0                  RNC
0E94 19                  DAD   D
0E95 88                  ADC   B
0E96 C9                  RET
```

```
0E97                *
0E97                * INTEGER SQUARE ROOT OF DE
0E97                *     [\]
0E97 01 00 80       SQR2   LXI    B,B000H  GUESS=80, ROOT0=0
0E9A CD A7 0E       SQRA1  CALL   APPX    CHECK APPROXIMATION
0E9D 78                    MOV    A,B     AND SHIFT APPX BIT
0E9E 0F                    RRC
0E9F 47                    MOV    B,A
0EA0 D2 9A 0E              JNC    SQRA1
0EA3 51                    MOV    D,C     DE=RESULT
0EA4 1E 00                 MVI    E,0
0EA6 C9                    RET    .
0EA7                *
0EA7 D5             APPX   PUSH   D       SAVE X
0EA8 78                    MOV    A,B     TRY NEW TEST BIT
0EA9 81                    ADD    C
0EAA 5F                    MOV    E,A
0EAB 16 00                 MVI    D,0
0EAD C5                    PUSH   B       SAVE BC
0EAE CD 5D 0E              CALL   MU21    AHL=A*DE
0EB1 C1                    POP    B
0EB2 D1                    POP    D       COMPARE TO X
0EB3 7B                    MOV    A,E     IF X <HL THEN TOO BIG
0EB4 95                    SUB    L
0EB5 7A                    MOV    A,D
0EB6 9C                    SBB    H
0EB7 D8                    RC     .
0EB8 78                    MOV    A,B     ELSE ADD TEST BIT TO C
0EB9 81                    ADD    C
0EBA 4F                    MOV    C,A
0EBB C9                    RET    .
0EBC                *
0EBC CD CA 0E       NEG32  CALL   INV16   INVERT DEHL
0EBF EB                    XCHG
0EC0 CD CA 0E              CALL   INV16
0EC3 EB                    XCHG   .
0EC4 23                    INX    H       AND ADD 1
0EC5                *
0EC5 7C                    MOV    A,H     IF HL=0, INCR DE
0EC6 B5                    ORA    L
0EC7 C0                    RNZ    .
0EC8 13                    INX    D
0EC9 C9                    RET    .
0ECA                *
0ECA 7C             INV16  MOV    A,H     INVERT HL
0ECB 2F                    CMA
0ECC 67                    MOV    H,A
0ECD 7D                    MOV    A,L
0ECE 2F                    CMA
0ECF 6F                    MOV    L,A
```

```
0ED0 C9          0600     RET   .
0ED1 00       V  0601     DB    256    END OF FFP
```

```
8000: C3 80 03 00 00 00 00 00 00 00 00 00 00 00 00 00
8010: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8020: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8030: 00 00 00 00 E5 2A 04 10 E9 00 00 00 C3 C5 00 00
8040: C3 7C 00 C3 AF 00 C3 AA 00 C3 FE 00 C3 04 01 C3
8050: 0E 01 C3 68 01 C3 3A 01 C3 2A 02 C3 7C 01 C3 AF
8060: 01 C3 CF 01 C3 67 02 C3 71 02 C3 DC 01 C3 E2 01
8070: C3 01 02 C3 17 02 C3 B1 00 C3 C1 00 3E 4C D3 20
8080: 3E 4F D3 60 CD 43 02 3E 50 CD AA 00 97 32 03 10
8090: 32 08 10 21 00 00 CD 81 00 11 34 01 21 0E 10 0E
80A0: 06 1A 77 13 23 0D C2 A1 00 C9 32 00 10 30 C9 20
80B0: C9 F3 22 04 10 7C B5 3E 0B CA BE 00 3E 09 30 F8
80C0: C9 32 08 10 C9 F5 C5 D5 0E 10 3E C0 30 20 07 DA
80D0: CD 00 00 3E 00 20 07 18 0D C2 D3 00 EB 22 01 10
80E0: EB 3E 01 32 03 10 3A 00 10 E6 80 F6 50 30 D1 C1
80F0: 3A 08 10 0F DA FA 00 F1 FB C9 F1 37 F8 C9 21 03
8100: 10 7E 87 C9 CD FE 00 C8 36 00 2A 01 10 C9 E5 C5
8110: 0E 10 3E C0 32 00 10 30 F3 CD 0E 10 3E 40 30 3E
8120: 80 29 1F 30 0D C2 1F 01 C1 E1 3E 40 32 00 10 F6
8130: 10 30 FB C9 20 07 D2 0E 10 C9 F5 87 E6 FE 6F 26
8140: 30 3E 03 32 01 30 7E CD 61 01 7E CD 61 01 5E 2C
8150: 66 68 F1 FE 0D C0 29 29 29 29 6C D8 23 E6 0F 67
8160: C9 3E 02 3D C2 63 01 C9 DC 3A 02 3E 40 CD 58 02
8170: 3E 80 CD 58 02 CD 48 02 3A 4C 30 C9 C5 E6 03 47
8180: 7C E6 F0 80 07 07 07 07 F6 40 CD A3 01 2D 22 FF
8190: CF 2C 3A 09 10 CD C0 01 29 78 17 32 0D 10 22 08
81A0: 10 C1 C9 D5 32 07 10 11 61 70 CD 86 02 D1 C9 79
81B0: E6 07 32 0A 10 EE 07 D3 63 32 06 10 78 32 09 10
81C0: 87 87 87 E6 3F 4F 3A 07 10 E6 80 81 C3 A3 01 E6
81D0: 01 0F 47 3A 07 10 E6 3F 80 C3 A3 01 2A 00 F0 C3
81E0: E5 01 22 00 80 E5 2A 08 10 11 04 00 19 22 08 10
81F0: D2 FF 01 21 0D 10 34 3A 0A 10 3C 8E CC 01 02 E1
8200: C9 3A 09 10 CD 0A 02 C3 7C 01 0F C5 47 E6 80 67
8210: 2E 00 78 E6 03 C1 C9 3A 0D 10 C5 87 1F 47 2A 08
8220: 10 7C 1F 67 7D 1F 6F 78 C1 C9 DC 3A 02 FE 06 D0
8230: CD 78 02 7D CD 58 02 C3 48 02 F5 3A 06 10 F6 20
8240: C3 4E 02 3E FF CD 61 02 F5 3A 06 10 E6 DF D3 63
8250: 32 06 10 CD 71 02 F1 C9 2F CD 61 02 CD 71 02 3E
8260: FF 11 62 20 C3 86 02 06 C8 CD 71 02 05 C2 69 02
8270: C9 11 71 02 18 78 82 C2 74 02 C9 21 01 00 E6 0F
8280: C8 29 3D C3 80 02 F5 3E 01 87 C2 8E 02 53 F1 12
8290: C9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
82A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
82B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
82C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
82D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
82E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
82F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8300: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
8310: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8320: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8330: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8340: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8350: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8360: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8370: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8380: 31 00 18 21 11 11 CD FA 03 CD 40 00 CD 67 00 CD
8390: 40 04 CD 40 05 97 32 20 10 21 00 00 CD FA 03 CD
83A0: 4C 00 C2 AB 03 CD C0 03 C3 9F 03 22 21 10 CD FA
83B0: 03 CD 43 05 D2 9F 03 CD 43 04 DC C8 03 C3 9F 03
83C0: 2A C9 03 22 23 10 C3 23 10 76 C9 21 20 10 34 C9
83D0: 11 00 10 0E 08 2E 00 CD E6 03 29 CA DF 03 2C 14
83E0: 7A 0D C2 D7 03 C9 1E 00 CD F1 03 C0 1C C2 E8 03
83F0: C9 1A 47 2F 12 1A 2F 12 B8 C9 7D D3 00 7C D3 01
8400: C9 42 55 52 53 54 20 32 2D 31 2D 38 35 20 50 52
8410: 20 48 41 52 56 45 59 00 00 00 00 00 00 00 00 00
8420: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8430: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8440: C3 46 04 C3 4D 04 21 02 12 22 00 12 C9 7C E6 FC
8450: FE 8C 37 C0 E8 7A D6 8C CA 69 04 3D CA 6E 04 3D
8460: CA 73 04 3D CA 7C 04 37 C9 7B 32 00 12 C9 7B 32
8470: 01 12 C9 2A 00 12 73 23 22 00 12 C9 3A 02 12 FE
8480: AA C0 97 32 02 12 C3 03 12 00 00 00 00 00 00 00
8490: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
84A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
84B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
84C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
84D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
84E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
84F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8500: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8510: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8520: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8530: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8540: C3 46 05 C3 63 05 CD C0 08 97 CD C3 08 3E 0F CD
8550: 94 05 B7 CD 52 00 97 32 D5 10 21 00 00 CD 5B 00
8560: 21 45 83 7C E6 F0 FE 80 37 C0 7C D6 B0 FE 08 3F
8570: D8 E8 87 21 7E 05 CD 51 08 CD 56 08 78 E9 94 05
8580: C3 08 C6 08 9A 05 E0 05 2E 06 26 06 19 06 46 06
8590: 00 00 D9 05 E6 0F 32 C4 10 C9 5F 0F 0F 0F 0F E6
85A0: 07 47 32 C6 10 78 E6 07 4F 32 C7 10 CD 5E 00 21
85B0: 05 00 06 00 CD C8 05 3A C6 10 E6 07 67 3A C7 10
85C0: 6F 06 01 CD C8 05 87 C9 7D BC D8 78 0F 7C E5 C5
85D0: CD 58 00 C1 E1 24 C3 C8 05 32 D5 10 0F C3 52 00
85E0: 3A D5 10 0F 3F DC 52 00 CD E4 06 3A C5 10 CD CF
85F0: 06 CD 65 07 22 C0 10 D5 CD 4F 00 E1 22 C2 10 CD
8600: 4F 00 CD 70 00 21 0A 88 CD 6D 00 CD 70 00 CD 6A
8610: 00 11 0A B8 CD 58 08 37 C0 3E 01 32 C8 10 CD 1C
8620: 07 CD D6 10 97 C9 21 0C D8 CD 6D 00 97 C9 3A D5
```

```
8630: 10 0F D4 52 00 CD 9F 07 21 48 4F D2 41 06 21 4F
8640: 4E CD 4F 00 87 C9 3A C8 10 FE 01 CC 9F 07 CD 8A
8650: 06 3A C8 10 FE 02 C8 2A C9 10 3A C8 10 CD 58 00
8660: CD 49 00 C2 81 06 CD 6A 00 CD 4F 00 CD 73 00 E8
8670: 47 2A CC 10 CD 58 08 C2 60 06 3A CE 10 88 C2 60
8680: 06 97 21 00 00 CD 58 00 87 C9 3E B2 CD CF 06 3E
8690: A1 CD CF 06 3A C5 10 CD CF 06 2A C9 10 CD 4F 00
86A0: 2A CA 10 CD D8 06 CD 4F 00 2A CC 10 CD 4F 00 2A
86B0: CD 10 CD D8 06 CD 4F 00 CD C9 08 EB CD CC 08 4F
86C0: CD CF 06 79 87 C8 1A CD CF 06 13 0D C3 C3 06 6F
86D0: 26 00 C3 4F 00 C3 C8 05 06 04 97 29 17 05 C2 D8
86E0: 06 6C 67 C9 CD 0B 07 41 CD CC 08 4F CD C9 08 E8
86F0: 21 D6 10 3A C5 10 FE 08 D2 00 07 06 02 C3 40 0A
8700: 78 B7 06 00 CA 40 0A 04 C3 40 0A 3A C4 10 32 C5
8710: 10 CD 1C 07 D0 21 C5 10 35 C3 11 07 3A C5 10 21
8720: 35 07 CD 51 08 4E 3A C5 10 FE 08 3F D0 CD CC 08
8730: 47 79 90 4F C9 C8 64 28 14 0A 04 02 01 14 0A 06
8740: 04 03 02 01 00 0A 00 14 00 32 00 64 00 C8 00 F4
8750: 01 E8 03 D0 07 A0 08 40 17 C1 26 21 3A 81 4D 42
8760: 74 84 E8 84 E8 3A C5 10 87 21 45 07 CD 51 08 CD
8770: 56 08 E5 CD CC 08 D1 CD 24 0C 16 00 5F CD 18 0C
8780: 21 CF 10 CD 03 0C 3A C7 10 21 C6 10 96 3C 57 1E
8790: 00 26 00 CD 18 0C 21 CF 10 CD 09 0C C3 18 0C 21
87A0: 0E F8 CD 6D 00 CD 70 00 11 0E F8 CD 16 08 D8 CD
87B0: 73 00 22 CC 10 32 CE 10 CD 70 00 CD 6A 00 11 0A
87C0: 88 CD 58 08 C2 D5 07 CD 73 00 22 C9 10 32 C8 10
87D0: 97 32 C8 10 C9 2A CC 10 3A CE 10 CD 58 00 CD CC
87E0: 08 4F CD 49 00 C0 41 CD 6A 00 11 0C D8 CD 58 08
87F0: CA 03 08 11 0E F8 CD 5B 08 CA 03 08 05 C2 E7 07
8800: C3 E2 07 79 90 F5 2A CC 10 3A CE 10 CD 58 00 F1
8810: C4 47 08 C3 C7 07 3E 03 32 D4 10 21 00 00 22 D2
8820: 10 CD 49 00 37 C0 D5 CD 6A 00 D1 CD 5B 08 C8 21
8830: D2 10 35 C2 26 08 23 35 C2 21 08 23 35 C2 21 08
8840: 3E 02 32 C8 10 37 C9 F5 CD 6A 00 F1 3D C2 47 08
8850: C9 85 6F D0 24 C9 7E 23 66 6F C9 7D 88 C0 7C 8A
8860: C9 7D D3 00 7C D3 01 C9 00 00 00 00 00 00 00 00
8870: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8880: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8890: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
88A0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
88B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
88C0: C3 DD 08 C3 D2 08 C3 24 09 C3 F3 08 C3 1A 09 C3
88D0: 0D 09 E6 0F 32 30 10 3E FF 32 31 10 C9 21 33 10
88E0: 11 A4 09 0E 06 1A 13 77 23 0D C2 E5 08 2D 7D 32
88F0: 32 10 C9 3A 30 10 FE 0A 21 7C 09 DA 03 09 21 33
8900: 10 D6 0A 87 C8 CD 0D 09 23 3D C3 03 09 F5 7E FE
8910: F0 CA 18 09 23 C3 0E 09 F1 C9 CD F3 08 5D CD 0D
8920: 09 7D 93 C9 4F 3A 30 10 FE 0A D8 21 31 10 34 C2
8930: 3B 09 CD F3 08 CD 60 09 CA 35 09 CD F3 08 CD 0D
8940: 09 79 F5 E5 3A 32 10 FE 78 D2 5C 09 5F 54 3C 32
```

```
8950: 32 10 1A 13 12 18 18 78 BD D2 52 09 E1 F1 77 C9
8960: 7E FE F0 CA 7A 09 E5 23 7E 28 77 23 3A 32 10 BD
8970: C2 67 09 21 32 10 35 E1 97 C9 3C C9 0B F0 08 07
8980: F0 08 05 F0 0B 05 07 F0 0B 05 01 02 00 F0 00 01
8990: 02 03 04 05 06 00 07 08 09 0A 08 0C 0D 0E 0F F0
89A0: F0 F0 F0 F0 F0 F0 F0 F0 F0 F0 00 00 00 00 00 00
89B0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
89C0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
89D0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
89E0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
89F0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8A00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8A10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8A20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8A30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8A40: CD 18 0B EB CD 87 0A CD D4 0A 79 FE 03 D4 73 0A
8A50: CD A6 0A 23 79 FE 02 DA 60 0A CD CB 0A CD A6 0A
8A60: CD EE 0A 11 53 08 21 94 10 0E 24 CD 27 0B 2A 90
8A70: 10 B7 C9 E5 C5 23 23 0D 0D CD A6 0A CD CB 0A 23
8A80: 0D C2 79 0A C1 E1 C9 3A A0 0A CD 1C 08 78 CD 1C
8A90: 0B 11 A2 0A CD 0D 0B E5 2A 90 10 22 92 10 E1 C9
8AA0: 06 00 CD 30 0B FF 7E FE 3F 11 C3 0A CA 0D 0B 3A
8AB0: C0 0A CD 1C 0B 7E 87 CD 1C 0B 3A C2 0A C3 1C 08
8AC0: 2A 00 B0 2A 90 90 2C 22 90 10 FF 11 D1 0A C3 0D
8AD0: 08 D8 7F FF 78 87 C8 3D 11 E9 0A CA 0D 08 11 E4
8AE0: 0A C3 0D 08 69 CD 9F 10 FF 69 CD 94 10 FF 11 06
8AF0: 0B CD 0D 0B 3A 92 10 CD 1C 08 3A 93 10 CD 1C 08
8B00: 11 09 0B C3 0D 0B D2 FF FF C3 4B 08 FF 1A FE FF
8B10: C8 CD 1C 08 13 C3 0D 08 22 90 10 C9 E5 2A 90 10
8B20: 77 23 22 90 10 E1 C9 1A 77 13 23 0D C2 27 08 C9
8B30: 3E 01 CD 61 00 3E 03 32 01 30 21 00 00 22 90 10
8B40: 3E 01 CD 79 00 F8 3A 90 10 87 C9 97 CD 79 00 97
8B50: C3 61 00 FB 20 C8 D8 7F 00 00 D2 94 10 C9 D4 A7
8B60: 10 2D C2 9F 10 C9 F3 F5 06 20 20 A0 C2 A8 10 20
8B70: A0 CA 80 10 F1 F8 C9 00 00 00 00 00 00 00 00 00
8B80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8B90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8BA0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8BB0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8BC0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8BD0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8BE0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8BF0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8C00: C3 27 0C C3 2D 0C C3 33 0C C3 5E 0C C3 F2 0C C3
8C10: E8 0C C3 DC 0C C3 4B 0C C3 62 0D C3 DD 0D C3 26
8C20: 0E C3 2F 0E C3 5D 0E 4E 23 56 23 5E C9 71 23 72
8C30: 23 73 C9 7A 87 C8 46 23 7E 87 CA C3 0D 23 6E 67
8C40: 78 A9 F2 50 0C CD D4 0D CD 50 0C 79 EE 80 4F C9
8C50: 78 81 D6 40 FA C9 0D 4F CD 43 0E C3 AF 0D 7A B7
8C60: C8 46 23 7E B7 CA CE 0D 23 6E 67 78 A9 F2 79 0C
```

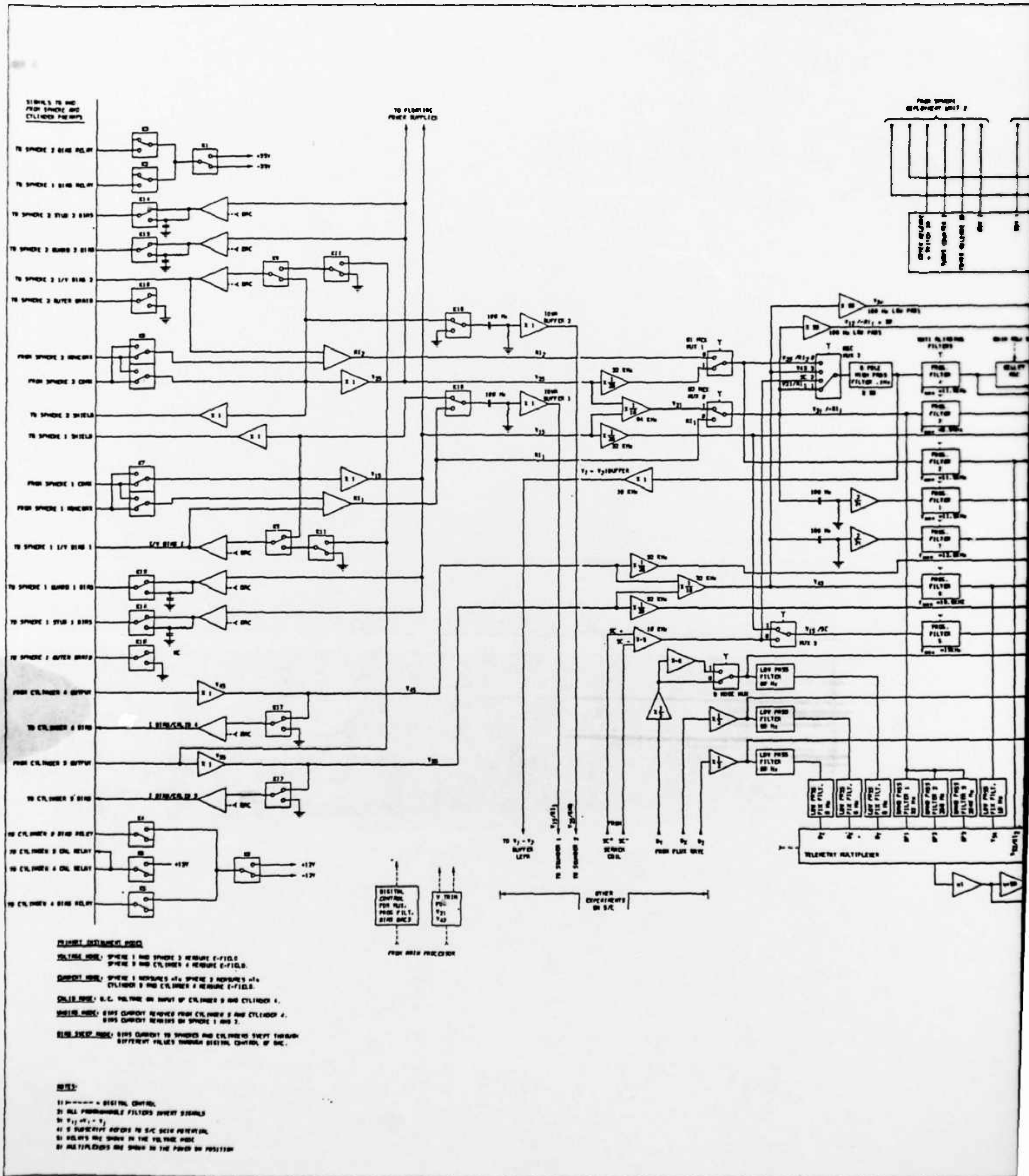
```
8C70: CD D4 0D CD 79 0C C3 48 0C 79 90 C6 40 FA C9 0D
8C80: 4F C5 7C 2F 47 7D 2F 4F 03 62 6B 09 DA A9 0C EB
8C90: 3E 10 CD C2 0C 29 DA 9F 0C 09 DA 9F 0C C1 C9 C1
8CA0: 1C C0 14 C0 11 00 80 0C C9 3E 10 11 FF FF CD C2
8CB0: 0C C1 0C 37 7A 1F 57 7B 1F 5F D0 C3 A0 0C 33 33
8CC0: 3D C8 29 DA D4 0C EB 29 EB 1C E5 09 DA 8E 0C E1
8CD0: 1D C3 C0 0C EB 29 EB 09 1C C3 C0 0C C5 D5 CD EB
8CE0: 0C 7A B7 CA EB 0C 79 07 D1 C1 C9 7E EE 80 47 C3
8CF0: F3 0C 46 23 7E 23 6E 67 97 8C C8 8A CA 5F 0D 79
8D00: 90 87 F2 0B 0D 78 41 4F EB 90 87 CA 15 0D 0F FE
8D10: 10 D0 CD 47 0D 78 A9 FA 25 0D 19 EB D0 7A 1F 57
8D20: 7B 1F 5F 0C C9 7B 95 7A 9C DA 3A 0D 57 7B 95 5F
8D30: 21 00 00 82 C2 74 0D 0E 00 C9 7D 93 5F 7C 9A 57
8D40: 48 21 00 00 C3 74 0D D6 08 DA 52 0D 6C 26 00 C8
8D50: D6 08 C5 47 97 29 8F 04 C2 55 0D C1 6C 67 C9 EB
8D60: 48 C9 7A 87 0E 60 F2 74 0D CD BC 0E CD 74 0D 79
8D70: F6 80 4F C9 79 87 F2 82 0D E6 7F 4F CD 82 0D C3
8D80: 4B 0C 7A B7 C2 AC 0D 83 C2 A1 0D B4 C2 98 0D B5
8D90: C2 95 0D 4A C9 55 06 18 C3 A6 0D EB 06 10 C3 A6
8DA0: 0D 53 5C 65 06 08 79 90 4F DA C3 0D 7A 6C 63 87
8DB0: FA 89 0D 0D 29 8F F2 83 0D 57 5C 7D 07 DC A0 0C
8DC0: 79 B7 F0 0E 00 11 00 00 C9 FE C0 D2 C3 0D 0E 7F
8DD0: 11 FF FF C9 78 E6 7F 47 79 E6 7F 4F C9 79 EE 80
8DE0: FA EA 0D 4F CD EA 0D C3 8C 0E E6 7F FE 41 DA 18
8DF0: 0E FE 60 D2 1F 0E 21 00 00 D6 50 C8 EB D2 0F 0E
8E00: C6 10 CD 0F 0E EB 11 00 00 C9 29 2C EB 3D C8 29
8E10: EB DA 0A 0E 29 C3 0C 0E 11 00 00 21 00 00 C9 11
8E20: FF 7F 21 FF FF C9 7A B7 C8 41 62 6B C3 40 0C 7A
8E30: 87 C8 79 E6 01 C4 1D 0D C5 CD 97 0E C1 79 0F C6
8E40: 20 4F C9 97 B2 CA 5B 0E 85 CA 5C 0E E5 CD 5D 0E
8E50: 6C 67 E3 7C CD 5D 0E D1 19 88 C9 EB 7C 21 00 00
8E60: 44 87 D2 67 0E 19 88 29 8F D2 6E 0E 19 88 29 8F
8E70: D2 75 0E 19 88 29 8F D2 7C 0E 19 88 29 8F D2 83
8E80: 0E 19 88 29 8F D2 8A 0E 19 88 29 8F D2 91 0E 19
8E90: 88 29 8F D0 19 88 C9 01 00 80 CD A7 0E 78 0F 47
8EA0: D2 9A 0E 51 1E 00 C9 D5 78 81 5F 16 00 C5 CD 5D
8EB0: 0E C1 D1 78 95 7A 9C D8 78 81 4F C9 CD CA 0E EB
8EC0: CD CA 0E EB 23 7C 85 C0 13 C9 7C 2F 67 7D 2F 6F
8ED0: C9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8EE0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8EF0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F00: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F10: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F20: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F30: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F40: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F50: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F60: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F70: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8F80: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
8F90: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8FA0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8FB0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8FC0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8FD0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8FE0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8FF0: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

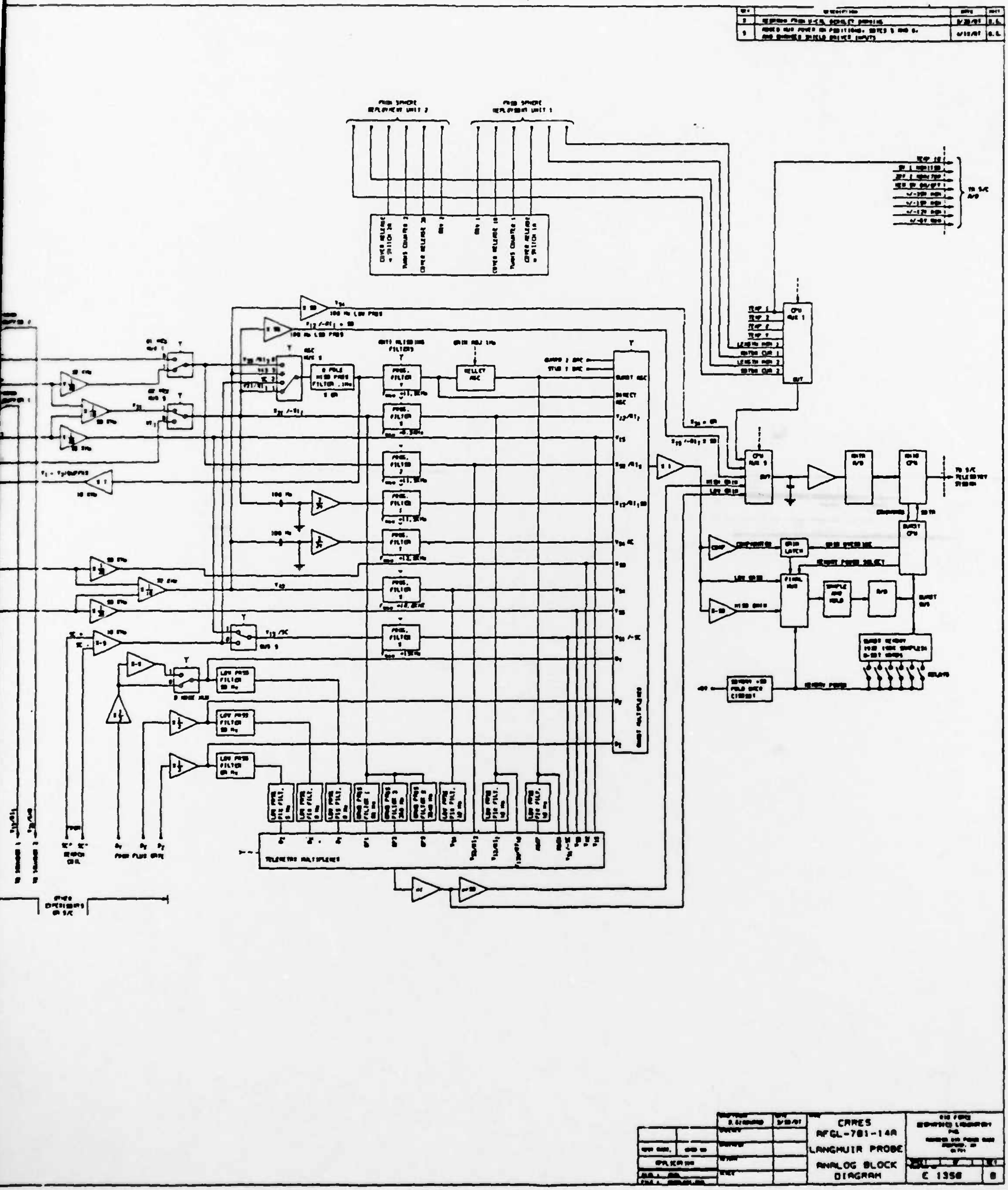
Appendix B.

Block Diagrams

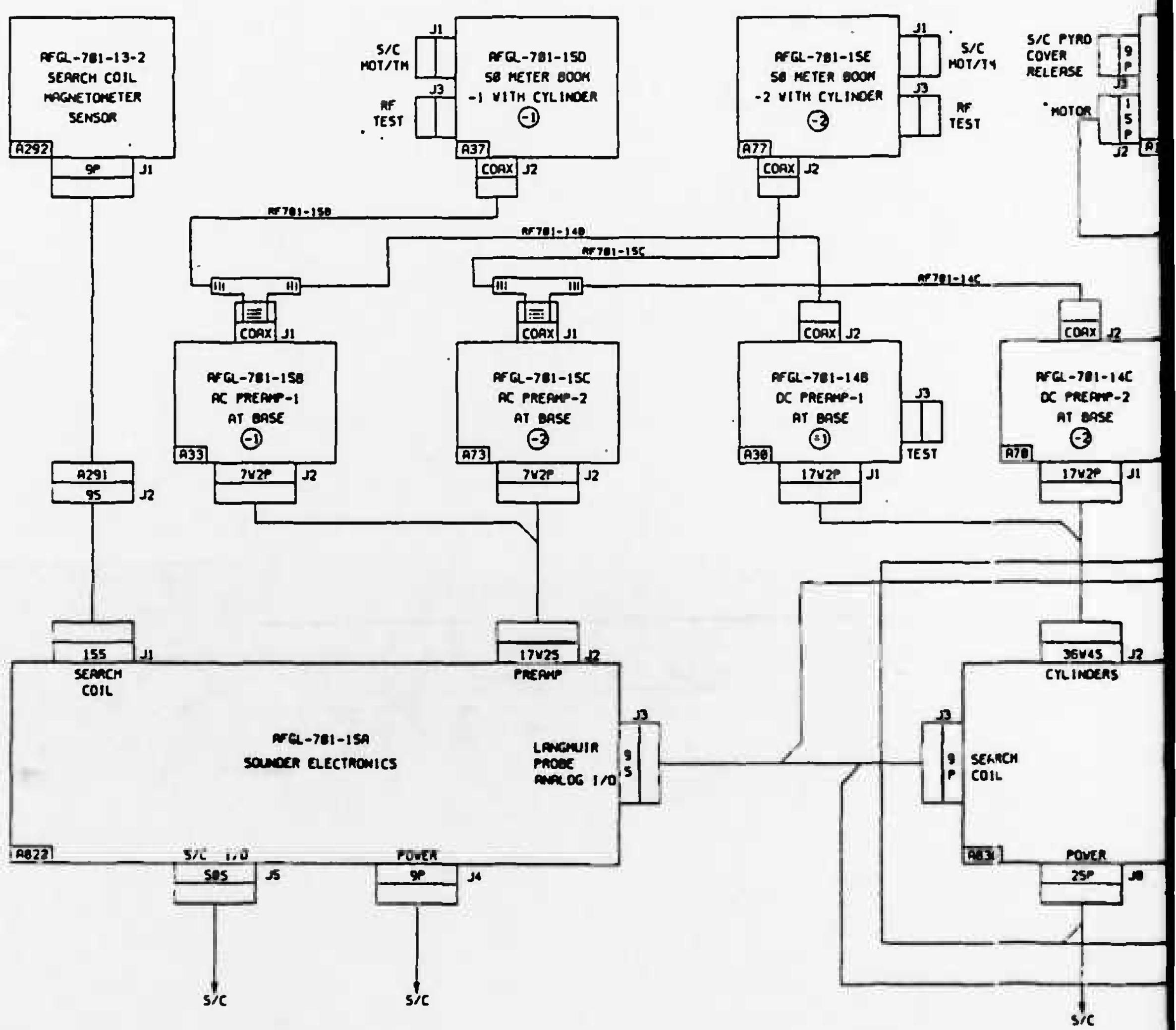
*U.S. GOVERNMENT PRINTING OFFICE: 1989-600-000/00013


TO FLOATING POWER SUPPLIES
VOLTAGE MODE: SPHERE 1 AND SPHERE 2 MEASURE E-FIELD
SPHERE 3 AND CYLINDER 4 MEASURE E-FIELD.
CALIB MODE: D.C. VOLTAGE ON INPUT OF CYLINDER 3 AND CYLINDER 4.
BIAS SWEEP MODE: BIAS CURRENT TO SPHERES AND CYLINDERS SWEPT THROUGH DIFFERENT VALUES THROUGH DIGITAL CONTROL OF DAC.
NOTES:
1) = DIGITAL CONTROL
2) ALL PROGRAMMABLE FILTERS INVERT SIGNALS
4) S SUBSCRIPT REFERS TO S/C SKIN POTENTIAL
5) RELAYS ARE SHOWN IN THE VOLTAGE MODE
6) MULTIPLEXERS ARE SHOWN IN THE POWER ON POSITION
TELEMETRY MULTIPLEXER
OTHER EXPERIMENTS ON S/C
FROM MAIN PROCESSOR

1 OF 2


CARES
AFGL-781-14A
LANGMUIR PROBE
ANALOG BLOCK
DIAGRAM
C 1358
B
TELEMETRY MULTIPLEXER

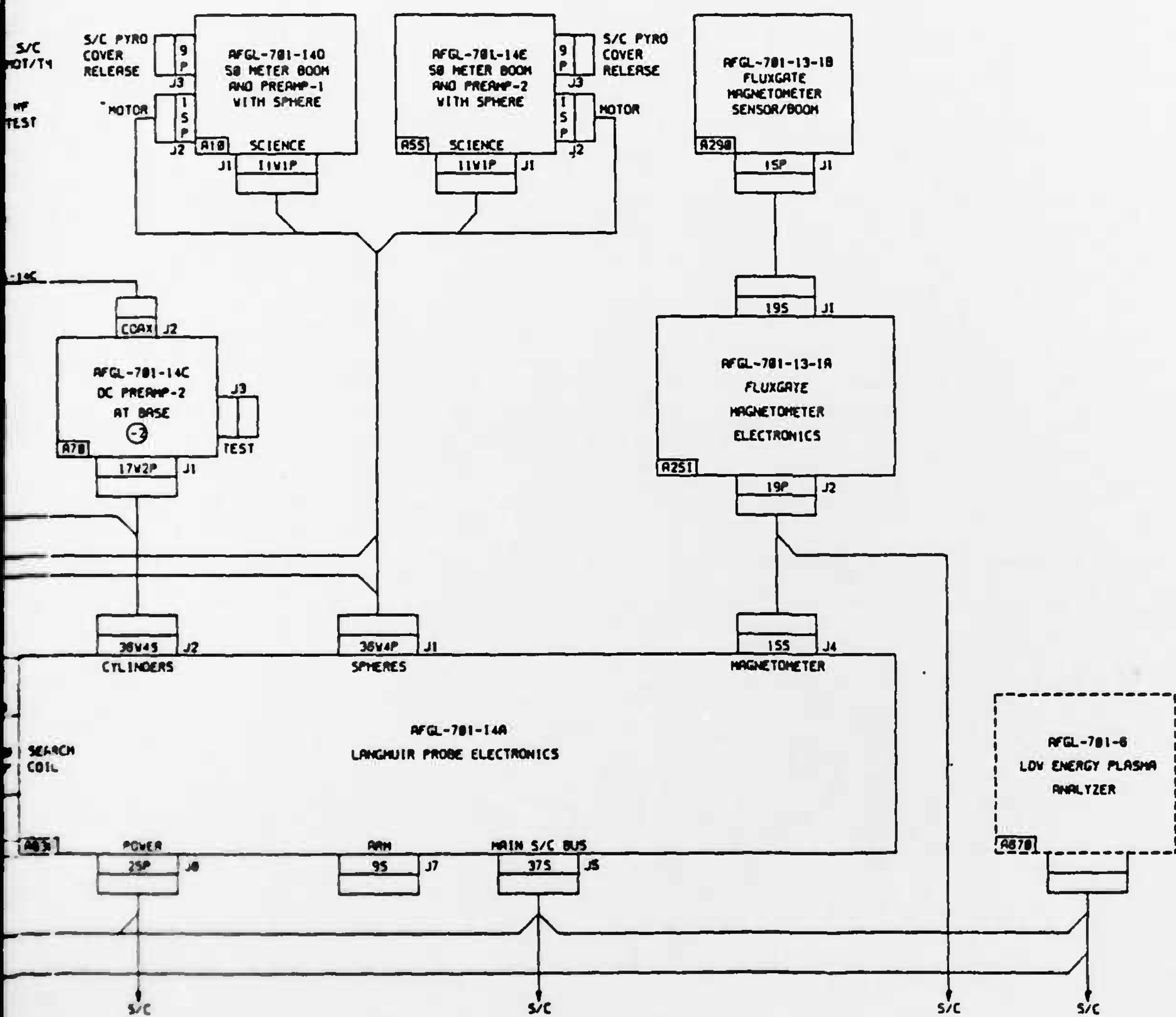
2 OF 2



NOTE: "TEE" CONNECTORS ARE OMNI SPECTRA
P/N 2041 6203 00

1 OF 2

REV	DESCRIPTION	DATE	INIT
H	REDRAWN. ADD COAX T'S. DEL -14A J6, -15D J4, -15E J4.	14 FEB 85	J.D.
I	ADDED A291 AND S/C NO.'S	01 JUL 85	J.D.



		DRAFTSMAN J. DULONG	DATE 2/8/85	NAME
		CHECKER		CRRES AFGL-701-13,-14,-15 INTERCONNECTION BLOCK DIAGRAM
NEXT ASSY.	USED ON	ENGINEER		AIR FORCE GEOPHYSICS LABORATORY PHG HANSCOM AIR FORCE BASE BEDFORD, MASS. 01731
APPLICATION		WEIGHT		SHEET OF
DISK : 0501 FILE : CRRESINC.DWG		SCALE		DRAWING NO. 01299 REV I

2 OF 2